SimCRS

1.01.1

Generated by Doxygen 1.6.1

Mon Jun 22 00:16:44 2015

Contents

1 SimCRS Documentation

1.1 Getting Started

- Main features
- Installation
- Linking with SimCRS
- Users Guide
- Tutorials
- Copyright and License
- Make a Difference
- Make a new release
- People

1.2 SimCRS at SourceForge

- Project page
- Download SimCRS
- Open a ticket for a bug or feature
- Mailing lists
- Forums
  - Discuss about Development issues
  - Ask for Help
  - Discuss SimCRS

1.3 SimCRS Development

- Git Repository (Subversion is deprecated)
- Coding Rules
- Documentation Rules
- Test Rules

1.4 External Libraries

- Boost (C++ STL extensions)
- Python
- MySQL client
- SOCI (C++ DB API)

1.5 Support SimCRS

1.6 About SimCRS

SimCRS is a C++ library of travel distribution classes and functions, exclusively targeting simulation purposes. N

SimCRS makes an extensive use of existing open-source libraries for increased functionality, speed and accuracy. In particular the Boost (*C++ Standard Extensions*) library is used.

The SimCRS library originates from the department of Operational Research and Innovation at Amadeus, Sophia Antipolis, France. SimCRS is released under the terms of the GNU Lesser General Public License (LGPL) for you to enjoy.

SimCRS should work on GNU/Linux, Sun Solaris, Microsoft Windows (with Cygwin, MinGW/MSYS, or Microsoft Visual C++ .NET) and Mac OS X operating systems.

Note:

(N) - The SimCRS library is **NOT** intended, in any way, to be used by any entity for production systems. If you want to report issue, bug or feature request, or if you just want to give feedback, have a look on the right-hand side of this page for the preferred reporting methods. In any case, please do not contact Amadeus directly for any matter related to SimCRS.

2 People

2.1 Project Admins

- Denis Arnaud <denis_arnaud@users.sourceforge.net> (N)
- Anh Quan Nguyen <quannaus@users.sourceforge.net> (N)

2.2 Developers

- Anh Quan Nguyen <quannaus@users.sourceforge.net> (N)
- Denis Arnaud <denis_arnaud@users.sourceforge.net> (N)
- Son Nguyen Kim <snguyenkim@users.sourceforge.net>
- Nicolas Bondoux <nbondoux@users.sourceforge.net> (N)

2.3 Retired Developers

- Patrick Grandjean <pgrandjean@users.sourceforge.net> (N)
- Ngoc-Thach Hoang <hoangngocthach@users.sourceforge.net> (N)

2.4 Contributors

- Emmanuel Bastien <ebastien@users.sourceforge.net> (N)
- Christophe Lacombe <ddtof@users.sourceforge.net> (N)

2.5 Distribution Maintainers

- Fedora/RedHat: Denis Arnaud <denis_arnaud@users.sourceforge.net> (N)
- Debian: Emmanuel Bastien <ebastien@users.sourceforge.net> (N)

Note:

(N) - Amadeus employees.

3 Coding Rules

In the following sections we describe the naming conventions which are used for files, classes, structures, local variables, and global variables.

3.1 Default Naming Rules for Variables

Variables names follow Java naming conventions. Examples:

- `lNumberOfPassengers`
- `lSeatAvailability`

3.2 Default Naming Rules for Functions

Function names follow Java naming conventions. Example:

- `int myFunctionName (const int& a, int b)`

3.3 Default Naming Rules for Classes and Structures

Each new word in a class or structure name should always start with a capital letter and the words should be separated with an under-score. Abbreviations are written with capital letters. Examples:

- `MyClassName`
- `MyStructName`

3.4 Default Naming Rules for Files

Files are named after the C++ class names.

Source files are named using `.cpp` suffix, whereas header files end with `.hpp` extension. Examples:

- `FlightDate.hpp`
- `SegmentDate.cpp`

3.5 Default Functionality of Classes

All classes that are configured by input parameters should include:

- default empty constructor
- one or more additional constructor(s) that takes input parameters and initializes the class instance
- setup function, preferably named `'setup'` or `'set_parameters'`

Explicit destructor functions are not required, unless they are needed. It shall not be possible to use any of the other member functions unless the class has been properly initiated with the input parameters.

4 Copyright and License

4.1 GNU LESSER GENERAL PUBLIC LICENSE

4.1.1 Version 2.1, February 1999

[This is the first released version of the Lesser GPL. It also counts
as the successor of the GNU Library Public License, version 2, hence
the version number 2.1.]

4.2 Preamble

The licenses for most software are designed to take away your freedom to share and change it. By contrast, the GNU General Public Licenses are intended to guarantee your freedom to share and change free software--to make sure the software is free for all its users.

This license, the Lesser General Public License, applies to some specially designated software packages--typically libraries--of the Free Software Foundation and other authors who decide to use it. You can use it too, but we suggest you first think carefully about whether this license or the ordinary General Public License is the better strategy to use in any particular case, based on the explanations below.

When we speak of free software, we are referring to freedom of use, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for this service if you wish); that you receive source code or can get it if you want it; that you can change the software and use pieces of it in new free programs; and that you are informed that you can do these things.

To protect your rights, we need to make restrictions that forbid distributors to deny you these rights or to ask you to surrender these rights. These restrictions translate to certain responsibilities for you if you distribute copies of the library or if you modify it.

For example, if you distribute copies of the library, whether gratis or for a fee, you must give the recipients all the rights that we gave you. You must make sure that they, too, receive or can get the source code. If you link other code with the library, you must provide complete object files to the recipients, so that they can relink them with the library after making changes to the library and recompiling it. And you must show them these terms so they know their rights.

We protect your rights with a two-step method: (1) we copyright the library, and (2) we offer you this license, which gives you legal permission to copy, distribute and/or modify the library.

To protect each distributor, we want to make it very clear that there is no warranty for the free library. Also, if the library is modified by someone else and passed on, the recipients should know that what they have is not the original version, so that the original author's reputation will not be affected by problems that might be introduced by others.

Finally, software patents pose a constant threat to the existence of any free program. We wish to make sure that a company cannot effectively restrict the users of a free program by obtaining a restrictive license from a patent holder. Therefore, we insist that any patent license obtained for a version of the library must be consistent with the full freedom of use specified in this license.

Most GNU software, including some libraries, is covered by the ordinary GNU General Public License. This license, the GNU Lesser General Public License, applies to certain designated libraries, and is quite different from the ordinary General Public License. We use this license for certain libraries in order to

permit linking those libraries into non-free programs.

When a program is linked with a library, whether statically or using a shared library, the combination of the two is legally speaking a combined work, a derivative of the original library. The ordinary General Public License therefore permits such linking only if the entire combination fits its criteria of freedom. The Lesser General Public License permits more lax criteria for linking other code with the library.

We call this license the "Lesser" General Public License because it does Less to protect the user's freedom than the ordinary General Public License. It also provides other free software developers Less of an advantage over competing non-free programs. These disadvantages are the reason we use the ordinary General Public License for many libraries. However, the Lesser license provides advantages in certain special circumstances.

For example, on rare occasions, there may be a special need to encourage the widest possible use of a certain library, so that it becomes a de-facto standard. To achieve this, non-free programs must be allowed to use the library. A more frequent case is that a free library does the same job as widely used non-free libraries. In this case, there is little to gain by limiting the free library to free software only, so we use the Lesser General Public License.

In other cases, permission to use a particular library in non-free programs enables a greater number of people to use a large body of free software. For example, permission to use the GNU C Library in non-free programs enables many more people to use the whole GNU operating system, as well as its variant, the GNU/Linux operating system.

Although the Lesser General Public License is Less protective of the users' freedom, it does ensure that the user of a program that is linked with the Library has the freedom and the wherewithal to run that program using a modified version of the Library.

The precise terms and conditions for copying, distribution and modification follow. Pay close attention to the difference between a "work based on the library" and a "work that uses the library". The former contains code derived from the library, whereas the latter must be combined with the library in order to run.

4.3 TERMS AND CONDITIONS FOR COPYING, DISTRIBUTION AND MODIFICATION

0. This License Agreement applies to any software library or other program which contains a notice placed by the copyright holder or other authorized party saying it may be distributed under the terms of this Lesser General Public License (also called "this License"). Each licensee is addressed as "you".

A "library" means a collection of software functions and/or data prepared so as to be conveniently linked with application programs (which use some of those functions and data) to form executables.

The "Library", below, refers to any such software library or work which has been distributed under these terms. A "work based on the Library" means either the Library or any derivative work under copyright law: that is to say, a work containing the Library or a portion of it, either verbatim or with modifications and/or translated straightforwardly into another language. (Hereinafter, translation is included without limitation in the term "modification".)

"Source code" for a work means the preferred form of the work for making modifications to it. For a library, complete source code means all the source code for all modules it contains, plus any associated interface definition files, plus the scripts used to control compilation and installation of the library.

Activities other than copying, distribution and modification are not covered by this License; they are outside its scope. The act of running a program using the Library is not restricted, and output from such a program is covered only if its contents constitute a work based on the Library (independent of the use of the Library in a tool for writing it). Whether that is true depends on what the Library does and what the program that uses the Library does.

1. You may copy and distribute verbatim copies of the Library's complete source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice and disclaimer of warranty; keep intact all the notices that refer to this License and to the absence of any warranty; and distribute a copy of this License along with the Library.

You may charge a fee for the physical act of transferring a copy, and you may at your option offer warranty protection in exchange for a fee.

2. You may modify your copy or copies of the Library or any portion of it, thus forming a work based on the Library, and copy and distribute such modifications or work under the terms of Section 1 above, provided that you also meet all of these conditions:

a) The modified work must itself be a software library.

b) You must cause the files modified to carry prominent notices stating that you changed the files and the date of any change.

c) You must cause the whole of the work to be licensed at no charge to all third parties under the terms of this License.

d) If a facility in the modified Library refers to a function or a table of data to be supplied by an application program that uses the facility, other than as an argument passed when the facility is invoked, then you must make a good faith effort to ensure that, in the event an application does not supply such function or table, the facility still operates, and performs whatever part of its purpose remains meaningful.

(For example, a function in a library to compute square roots has a purpose that is entirely well-defined independent of the application. Therefore, Subsection 2d requires that any application-supplied function or table used by this function must be optional: if the application does not supply it, the square root function must still compute square roots.)

These requirements apply to the modified work as a whole. If identifiable sections of that work are not derived from the Library, and can be reasonably considered independent and separate works in themselves, then this License, and its terms, do not apply to those sections when you distribute them as separate works. But when you distribute the same sections as part of a whole which is a work based on the Library, the distribution of the whole must be on the terms of this License, whose permissions for other licensees extend to the entire whole, and thus to each and every part regardless of who wrote it.

Thus, it is not the intent of this section to claim rights or contest your rights to work written entirely by you; rather, the intent is to exercise the right to control the distribution of derivative or collective works based on the Library.

In addition, mere aggregation of another work not based on the Library with the Library (or with a work based on the Library) on a volume of a storage or distribution medium does not bring the other work under the scope of this License.

3. You may opt to apply the terms of the ordinary GNU General Public License instead of this License to a given copy of the Library. To do this, you must alter all the notices that refer to this License, so that they refer to the ordinary GNU General Public License, version 2, instead of to this License. (If a newer version than version 2 of the ordinary GNU General Public License has appeared, then you can specify that version instead if you wish.) Do not make any other change in these notices.

Once this change is made in a given copy, it is irreversible for that copy, so the ordinary GNU General Public License applies to all subsequent copies and derivative works made from that copy.

This option is useful when you wish to copy part of the code of the Library into a program that is not a library.

4. You may copy and distribute the Library (or a portion or derivative of it, under Section 2) in object code or executable form under the terms of Sections 1 and 2 above provided that you accompany it with the complete corresponding machine-readable source code, which must be distributed under the terms of Sections 1 and 2 above on a medium customarily used for software interchange.

If distribution of object code is made by offering access to copy from a designated place, then offering equivalent access to copy the source code from the same place satisfies the requirement to distribute the source code, even though third parties are not compelled to copy the source along with the object code.

5. A program that contains no derivative of any portion of the Library, but is designed to work with the Library by being compiled or linked with it, is called a "work that uses the Library". Such a work, in isolation, is not a derivative work of the Library, and therefore falls outside the scope of this License.

However, linking a "work that uses the Library" with the Library creates an executable that is a derivative of the Library (because it contains portions of the Library), rather than a "work that uses the library". The executable is therefore covered by this License. Section 6 states terms for distribution of such executables.

When a "work that uses the Library" uses material from a header file that is part of the Library, the object code for the work may be a derivative work of the Library even though the source code is not. Whether this is true is especially significant if the work can be linked without the Library, or if the work is itself a library. The threshold for this to be true is not precisely defined by law.

If such an object file uses only numerical parameters, data structure layouts and accessors, and small macros and small inline functions (ten lines or less in length), then the use of the object file is unrestricted, regardless of whether it is legally a derivative work. (Executables containing this object code plus portions of the Library will still fall under Section 6.)

Otherwise, if the work is a derivative of the Library, you may distribute the object code for the work under the terms of Section 6. Any executables containing that work also fall under Section 6, whether or not they are linked directly with the Library itself.

6. As an exception to the Sections above, you may also combine or link a "work that uses the Library" with the Library to produce a work containing portions of the Library, and distribute that work under terms of your choice, provided that the terms permit modification of the work for the customer's own use and reverse engineering for debugging such modifications.

You must give prominent notice with each copy of the work that the Library is used in it and that the Library and its use are covered by this License. You must supply a copy of this License. If the work during execution displays copyright notices, you must include the copyright notice for the Library among them, as well as a reference directing the user to the copy of this License. Also, you must do one of these things:

a) Accompany the work with the complete corresponding machine-readable source code for the Library including whatever changes were used in the work (which must be distributed under Sections 1 and 2 above); and, if the work is an executable linked with the Library, with the complete machine-readable "work that uses the Library", as object code and/or source code, so that the user can modify the Library and then relink to produce a modified executable containing the modified Library. (It is understood that the user who changes the contents of definitions files in the Library will not necessarily be able to recompile the application to use the modified definitions.)

b) Use a suitable shared library mechanism for linking with the Library. A suitable mechanism is one that (1) uses at run time a copy of the library already present on the user's computer system, rather than copying library functions into the executable, and (2) will operate properly with a modified version of the library, if the user installs one, as long as the modified version is interface-compatible with the version that the work was made with.

c) Accompany the work with a written offer, valid for at least three years, to give the same user the materials specified in Subsection 6a, above, for a charge no more than the cost of performing this distribution.

d) If distribution of the work is made by offering access to copy from a designated place, offer equivalent access to copy the above specified materials from the same place.

e) Verify that the user has already received a copy of these materials or that you have already sent this user a copy.

For an executable, the required form of the "work that uses the Library" must include any data and utility

programs needed for reproducing the executable from it. However, as a special exception, the materials to be distributed need not include anything that is normally distributed (in either source or binary form) with the major components (compiler, kernel, and so on) of the operating system on which the executable runs, unless that component itself accompanies the executable.

It may happen that this requirement contradicts the license restrictions of other proprietary libraries that do not normally accompany the operating system. Such a contradiction means you cannot use both them and the Library together in an executable that you distribute.

7. You may place library facilities that are a work based on the Library side-by-side in a single library together with other library facilities not covered by this License, and distribute such a combined library, provided that the separate distribution of the work based on the Library and of the other library facilities is otherwise permitted, and provided that you do these two things:

a) Accompany the combined library with a copy of the same work based on the Library, uncombined with any other library facilities. This must be distributed under the terms of the Sections above.

b) Give prominent notice with the combined library of the fact that part of it is a work based on the Library, and explaining where to find the accompanying uncombined form of the same work.

8. You may not copy, modify, sublicense, link with, or distribute the Library except as expressly provided under this License. Any attempt otherwise to copy, modify, sublicense, link with, or distribute the Library is void, and will automatically terminate your rights under this License. However, parties who have received copies, or rights, from you under this License will not have their licenses terminated so long as such parties remain in full compliance.

9. You are not required to accept this License, since you have not signed it. However, nothing else grants you permission to modify or distribute the Library or its derivative works. These actions are prohibited by law if you do not accept this License. Therefore, by modifying or distributing the Library (or any work based on the Library), you indicate your acceptance of this License to do so, and all its terms and conditions for copying, distributing or modifying the Library or works based on it.

10. Each time you redistribute the Library (or any work based on the Library), the recipient automatically receives a license from the original licensor to copy, distribute, link with or modify the Library subject to these terms and conditions. You may not impose any further restrictions on the recipients' exercise of the rights granted herein. You are not responsible for enforcing compliance by third parties with this License.

11. If, as a consequence of a court judgment or allegation of patent infringement or for any other reason (not limited to patent issues), conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot distribute so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not distribute the Library at all. For example, if a patent license would not permit royalty-free redistribution of the Library by all those who receive copies directly or indirectly through you, then the only way you could satisfy both it and this License would be to refrain entirely from distribution of the Library.

If any portion of this section is held invalid or unenforceable under any particular circumstance, the balance of the section is intended to apply, and the section as a whole is intended to apply in other circumstances.

It is not the purpose of this section to induce you to infringe any patents or other property right claims or to contest validity of any such claims; this section has the sole purpose of protecting the integrity of the free software distribution system which is implemented by public license practices. Many people have made generous contributions to the wide range of software distributed through that system in reliance on consistent application of that system; it is up to the author/donor to decide if he or she is willing to distribute software through any other system and a licensee cannot impose that choice.

This section is intended to make thoroughly clear what is believed to be a consequence of the rest of this License.

12. If the distribution and/or use of the Library is restricted in certain countries either by patents or by

copyrighted interfaces, the original copyright holder who places the Library under this License may add an explicit geographical distribution limitation excluding those countries, so that distribution is permitted only in or among countries not thus excluded. In such case, this License incorporates the limitation as if written in the body of this License.

13. The Free Software Foundation may publish revised and/or new versions of the Lesser General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Library specifies a version number of this License which applies to it and "any later version", you have the option of following the terms and conditions either of that version or of any later version published by the Free Software Foundation. If the Library does not specify a license version number, you may choose any version ever published by the Free Software Foundation.

14. If you wish to incorporate parts of the Library into other free programs whose distribution conditions are incompatible with these, write to the author to ask for permission. For software which is copyrighted by the Free Software Foundation, write to the Free Software Foundation; we sometimes make exceptions for this. Our decision will be guided by the two goals of preserving the free status of all derivatives of our free software and of promoting the sharing and reuse of software generally.

4.3.1 NO WARRANTY

15. BECAUSE THE LIBRARY IS LICENSED FREE OF CHARGE, THERE IS NO WARRANTY FOR THE LIBRARY, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE LIBRARY "AS IS" WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE LIBRARY IS WITH YOU. SHOULD THE LIBRARY PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MAY MODIFY AND/OR REDISTRIBUTE THE LIBRARY AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE LIBRARY (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE LIBRARY TO OPERATE WITH ANY OTHER SOFTWARE), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

4.3.2 END OF TERMS AND CONDITIONS

4.4 How to Apply These Terms to Your New Programs

If you develop a new library, and you want it to be of the greatest possible use to the public, we recommend making it free software that everyone can redistribute and change. You can do so by permitting redistribution under these terms (or, alternatively, under the terms of the ordinary General Public License).

To apply these terms, attach the following notices to the library. It is safest to attach them to the start of each source file to most effectively convey the exclusion of warranty; and each file should have at least the "copyright" line and a pointer to where the full notice is found.

```
    <one line to give the library's name and a brief idea of what it does.>
```

```
    Copyright (C) <year>  <name of author>

    This library is free software; you can redistribute it and/or
    modify it under the terms of the GNU Lesser General Public
    License as published by the Free Software Foundation; either
    version 2.1 of the License, or (at your option) any later version.

    This library is distributed in the hope that it will be useful,
    but WITHOUT ANY WARRANTY; without even the implied warranty of
    MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the GNU
    Lesser General Public License for more details.

    You should have received a copy of the GNU Lesser General Public
    License along with this library; if not, write to the Free Software
    Foundation, Inc., 51 Franklin Street, Fifth Floor, Boston, MA  02110-1301  USA
```

Also add information on how to contact you by electronic and paper mail.

You should also get your employer (if you work as a programmer) or your school, if any, to sign a "copyright disclaimer" for the library, if necessary. Here is a sample; alter the names:

```
  Yoyodyne, Inc., hereby disclaims all copyright interest in the
  library 'Frob' (a library for tweaking knobs) written by James Random Hacker.

  <signature of Ty Coon>, 1 April 1990
  Ty Coon, President of Vice
```

That's all there is to it!

Source

5 Documentation Rules

5.1 General Rules

All classes in SimCRS should be properly documented with Doxygen comments in include (`.hpp`) files. Source (`.cpp`) files should be documented according to a normal standard for well documented C++ code.

An example of how the interface of a class shall be documented in SimCRS is shown here:

```
/*!
 * \brief Brief description of MyClass here
 *
 * Detailed description of MyClass here. With example code if needed.
 */
class MyClass {
public:
  //! Default constructor
  MyClass(void) { setup_done = false; }

  /*!
   * \brief Constructor that initializes the class with parameters
   *
   * Detailed description of the constructor here if needed
   *
   * \param[in] param1 Description of \a param1 here
   * \param[in] param2 Description of \a param2 here
   */
  MyClass(TYPE1 param1, TYPE2 param2) { setup(param1, param2); }

  /*!
```

```
   * \brief Setup function for MyClass
   *
   * Detailed description of the setup function here if needed
   *
   * \param[in] param1 Description of \a param1 here
   * \param[in] param2 Description of \a param2 here
   */
  void setup(TYPE1 param1, TYPE2 param2);

  /*!
   * \brief Brief description of memberFunction1
   *
   * Detailed description of memberFunction1 here if needed
   *
   * \param[in]     param1 Description of \a param1 here
   * \param[in]     param2 Description of \a param2 here
   * \param[in,out] param3 Description of \a param3 here
   * \return Description of the return value here
   */
  TYPE4 memberFunction1(TYPE1 param1, TYPE2 param2, TYPE3 &param3);

private:

  bool _setupDone;         /*!< Variable that checks if the class is properly
                                initialized with parameters */
  TYPE1 _privateVariable1; //!< Short description of _privateVariable1 here
  TYPE2 _privateVariable2; //!< Short description of _privateVariable2 here
};
```

5.2 File Header

All files should start with the following header, which include Doxygen's \file, \brief and \author tags, $Date$ and $Revisions$ CVS tags, and a common copyright note:

```
/*!
 * \file
 * \brief Brief description of the file here
 * \author Names of the authors who contributed to this code
 * \date Date
 *
 * Detailed description of the file here if needed.
 *
 * ------------------------------------------------------------------------
 *
 * SimCRS - C++ Simulated Travel Distribution System Library
 *
 * Copyright (C) 2009-2011  (\see authors file for a list of contributors)
 *
 * \see copyright file for license information
 *
 * ------------------------------------------------------------------------
 */
```

5.3 Grouping Various Parts

All functions must be added to a Doxygen group in order to appear in the documentation. The following code example defines the group 'my_group':

```
/*!
 * \defgroup my_group Brief description of the group here
 *
 * Detailed description of the group here
```

```
*/
```

The following example shows how to document the function `myFunction` and how to add it to the group `my_group:`

```
/*!
 * \brief Brief description of myFunction here
 * \ingroup my_group
 *
 * Detailed description of myFunction here
 *
 * \param[in] param1 Description of \a param1 here
 * \param[in] param2 Description of \a param2 here
 * \return Description of the return value here
 */
TYPE3 myFunction(TYPE1 param1, TYPE2 &param2);
```

6 Main features

A short list of the main features of SimCRS is given below sorted in different categories. Many more features and functions exist and for these we refer to the reference documentation.

6.1 Network generation

- Network/graph generation

6.2 Inventory generation

- Inventory generation

6.3 Finding travel solutions

- Matching of travel solutions with user requests

6.4 Distributed inventories

- Inventory independent partitions
- MPI-based distribution

6.5 Other features

- CSV input file parsing
- Memory handling

7 Make a Difference

Do not ask what SimCRS can do for you. Ask what you can do for SimCRS.

You can help us to develop the SimCRS library. There are always a lot of things you can do:

- Start using SimCRS
- Tell your friends about SimCRS and help them to get started using it
- If you find a bug, report it to us. Without your help we can never hope to produce a bug free code.
- Help us to improve the documentation by providing information about documentation bugs
- Answer support requests in the SimCRS discussion forums on SourceForge. If you know the answer to a question, help others to overcome their SimCRS problems.
- Help us to improve our algorithms. If you know of a better way (e.g. that is faster or requires less memory) to implement some of our algorithms, then let us know.
- Help us to port SimCRS to new platforms. If you manage to compile SimCRS on a new platform, then tell us how you did it.
- Send us your code. If you have a good SimCRS compatible code, which you can release under the LGPL, and you think it should be included in SimCRS, then send it to us.
- Become an SimCRS developer. Send us an e-mail and tell what you can do for SimCRS.

8 Make a new release

8.1 Introduction

This document describes briefly the recommended procedure of releasing a new version of SimCRS using a Linux development machine and the SourceForge project site.

The following steps are required to make a release of the distribution package.

8.2 Initialisation

Clone locally the full Git project:

```
cd ~
mkdir -p dev/sim
cd ~/dev/sim
git clone git://simcrs.git.sourceforge.net/gitroot/simcrs/simcrs simcrsgit
cd simcrsgit
git checkout trunk
```

8.3 Release branch maintenance

Switch to the release branch, on your local clone, and merge the latest updates from the trunk. Decide about the new version to be released.

```
cd ~/dev/sim/simcrsgit
git checkout releases
git merge trunk
```

Update the version in the various build system files, replacing the old version numbers by the correct ones:

```
vi CMakeLists.txt
vi autogen.sh
vi README
```

Update the version, add some news in the `NEWS` file, add a change-log in the `ChangeLog` file and in the RPM specification files:

```
vi NEWS
vi ChangeLog
vi simcrs.spec
```

8.4 Commit and publish the release branch

Commit the new release:

```
cd ~/dev/sim/simcrsgit
git add -A
git commit -m "[Release 0.5.0] Release of the 0.5.0 version of SimCRS."
git push
```

8.5 Create distribution packages

Create the distribution packages using the following command:

```
cd ~/dev/sim/simcrsgit
git checkout releases
rm -rf build && mkdir -p build
cd build
export INSTALL_BASEDIR=/home/user/dev/deliveries
export LIBSUFFIX_4_CMAKE="-DLIB_SUFFIX=64"
cmake -DCMAKE_INSTALL_PREFIX=${INSTALL_BASEDIR}/simcrs-0.5.0 \
 -DWITH_STDAIR_PREFIX=${INSTALL_BASEDIR}/stdair-stable \
 -DWITH_AIRRAC_PREFIX=${INSTALL_BASEDIR}/airsched-stable \
 -DWITH_AIRRAC_PREFIX=${INSTALL_BASEDIR}/airrac-stable \
 -DWITH_RMOL_PREFIX=${INSTALL_BASEDIR}/rmol-stable \
 -DWITH_RMOL_PREFIX=${INSTALL_BASEDIR}/airinv-stable \
 -DWITH_RMOL_PREFIX=${INSTALL_BASEDIR}/simfqt-stable \
 -DCMAKE_BUILD_TYPE:STRING=Debug -DINSTALL_DOC:BOOL=ON \
 ${LIBSUFFIX_4_CMAKE} ..
make check && make dist
make install
```

This will configure, compile and check the package. The output packages will be named, for instance, `simcrs-0.5.0.tar.gz` and `simcrs-0.5.0.tar.bz2`.

8.6 Upload the HTML documentation to SourceForge

In order to update the Web site files, either:

- `synchronise them with rsync and SSH`: Upload the just generated HTML (and PDF) documentation onto the `SourceForge Web site`.

  ```
  cd ~/dev/sim/simcrsgit/build
  git checkout releases
  rsync -aiv ${INSTALL_BASEDIR}/simcrs-0.5.0/share/doc/simcrs-0.5.0/html/ \
   your_sf_user,simcrs@web.sourceforge.net:htdocs/
  ```

where -aiv options mean:

  - -a: archive/mirror mode; equals -rlptgoD (no -H, -A, -X)
  - -v: increase verbosity
  - -i: output a change-summary for all updates
  - Note the trailing slashes (/) at the end of both the source and target directories. It means that the content of the source directory (doc/html), rather than the directory itself, has to be copied into the content of the target directory.

- or use the SourceForge Shell service.

8.7 Generate the RPM packages

Optionally, generate the RPM package (for instance, for Fedora/RedHat):

```
cd ~/dev/sim/simcrsgit/build
git checkout releases
make dist
```

To perform this step, rpm-build, rpmlint and rpmdevtools have to be available on the system.

```
cp ../simcrs.spec ~/dev/packages/SPECS \
  && cp simcrs-0.5.0.tar.bz2 ~/dev/packages/SOURCES
cd ~/dev/packages/SPECS
rpmbuild -ba simcrs.spec
cd ~/dev/packages
rpmlint -i SPECS/simcrs.spec SRPMS/simcrs-0.5.0-1.fc16.src.rpm \
  RPMS/noarch/simcrs-* RPMS/i686/simcrs-*
```

8.8 Update distributed change log

Update the NEWS and ChangeLog files with appropriate information, including what has changed since the previous release. Then commit and push the changes into the SimCRS's Git repository.

8.9 Create the binary package, including the documentation

Create the binary package, which includes HTML and PDF documentation, using the following command:

```
cd ~/dev/sim/simcrsgit/build
git checkout releases
make package
```

The output binary package will be named, for instance, simcrs-0.5.0-Linux.tar.bz2. That package contains both the HTML and PDF documentation. The binary package contains also the executables and shared libraries, as well as C++ header files, but all of those do not interest us for now.

8.10 Upload the files to SourceForge

Upload the distribution and documentation packages to the SourceForge server. Check SourceForge help page on uploading software.

8.11 Make a new post

- submit a new entry in the SourceForge project-related news feed
- make a new post on the SourceForge hosted WordPress blog
- and update, if necessary, Trac tickets.

8.12 Send an email on the announcement mailing-list

Finally, you should send an announcement to simcrs-announce@lists.sourceforge.net (see https://lists.sourceforge.net/lists/listinfo/simcrs-announce for the archives)

9 Installation

9.1 Table of Contents

9.2 Fedora/RedHat Linux distributions

Note that on Fedora/RedHat Linux distributions, RPM packages are available and can be installed with your usual package manager. For instance:

```
yum -y install simcrs-devel simcrs-doc
```

RPM packages can also be available on the SourceForge download site.

9.3 SimCRS Requirements

SimCRS should compile without errors or warnings on most GNU/Linux systems, on UNIX systems like Solaris SunOS, and on POSIX based environments for Microsoft Windows like Cygwin or MinGW with MSYS. It can be also built on Microsoft Windows NT/2000/XP/Vista/7 using Microsoft's Visual C++ .NET, but our support for this compiler is limited. For GNU/Linux, SunOS, Cygwin and MinGW we assume that you have at least the following GNU software installed on your computer:

- GNU Autotools:
  - `autoconf`,
  - `automake`,
  - `libtool`,
  - `make`, version 3.72.1 or later (check version with 'make --version')
- `GCC` - GNU C++ Compiler (g++), version 4.3.x or later (check version with 'gcc --version')
- `Boost` - C++ STL extensions, version 1.35 or later (check version with 'grep "define BOOST_LIB_VERSION" /usr/include/boost/version.hpp')
- `MySQL` - Database client libraries, version 5.0 or later (check version with 'mysql --version')
- `SOCI` - C++ database client library wrapper, version 3.0.0 or later (check version with 'soci-config --version')

Optionally, you might need a few additional programs: `Doxygen`, `LaTeX`, `Dvips` and `Ghostscript`, to generate the HTML and PDF documentation.

We strongly recommend that you use recent stable releases of the GCC, if possible. We do not actively work on supporting older versions of the GCC, and they may therefore (without prior notice) become unsupported in future releases of SimCRS.

9.4 Basic Installation

Briefly, the shell commands `'./cmake .. && make install'` should configure, build, and install this package. The following more-detailed instructions are generic; see the `'README'` file for instructions specific to this package. Some packages provide this `'INSTALL'` file but do not implement all of the features documented below. The lack of an optional feature in a given package is not necessarily a bug. More recommendations for GNU packages can be found in the info page corresponding to "Makefile Conventions: (standards)Makefile Conventions".

The `'cmake'` shell script attempts to guess correct values for various system-dependent variables used during compilation. It uses those values to create a `'Makefile'` in each directory of the package. It may also create one or more '.h' files containing system-dependent definitions. Finally, it creates a `'CMakeCache`.txt' cache file that you can refer to in the future to recreate the current configuration, and a file `'CMakeFiles'` containing compiler output (useful mainly for debugging `'cmake'`).

It can also use an optional file (typically called `'config`.cache' and enabled with `'--cache-file=config`.cache' or simply `'-C'`) that saves the results of its tests to speed up reconfiguring. Caching is disabled by default to prevent problems with accidental use of stale cache files.

If you need to do unusual things to compile the package, please try to figure out how 'configure' could check whether to do them, and mail diffs or instructions to the address given in the `'README'` so they can be considered for the next release. If you are using the cache, and at some point `'config`.cache' contains results you don't want to keep, you may remove or edit it.

The file 'CMakeLists.txt' is used to create the 'Makefile' files.

The simplest way to compile this package is:

1. 'cd' to the directory containing the package's source code and type './cmake ..' to configure the package for your system. Running 'cmake' is generally fast. While running, it prints some messages telling which features it is checking for.
2. Type 'make' to compile the package.
3. Optionally, type 'make check'to run any self-tests that come with the package, generally using the just-built uninstalled binaries.
4. Type 'make install' to install the programs and any data files and documentation. When installing into a prefix owned by root, it is recommended that the package be configured and built as a regular user, and only the 'make install' phase executed with root privileges.
5. You can remove the program binaries and object files from the source code directory by typing 'make clean'. To also remove the files that 'configure' created (so you can compile the package for a different kind of computer), type 'make distclean'. There is also a 'make maintainer-clean' target, but that is intended mainly for the package's developers. If you use it, you may have to get all sorts of other programs in order to regenerate files that came with the distribution.
6. Often, you can also type 'make uninstall' to remove the installed files again. In practice, not all packages have tested that uninstallation works correctly, even though it is required by the GNU Coding Standards.

9.5 Compilers and Options

Some systems require unusual options for compilation or linking that the 'cmake' script does not know about. Run './cmake --help' for details on some of the pertinent environment variables.

You can give 'cmake' initial values for configuration parameters by setting variables in the command line or in the environment. Here is an example:

```
./cmake  CC=c99 CFLAGS=-g LIBS=-lposix
```

See also:

Defining Variables for more details.

9.6 Compiling For Multiple Architectures

You can compile the package for more than one kind of computer at the same time, by placing the object files for each architecture in their own directory. To do this, you can use GNU 'make'. 'cd' to the directory where you want the object files and executables to go and

run the ‘configure’ script. ‘configure’ automatically checks for the source code in the directory that ‘configure’ is in and in ‘..’. This is known as a "VPATH" build.

With a non-GNU ‘make’, it is safer to compile the package for one architecture at a time in the source code directory. After you have installed the package for one architecture, use ‘make distclean’ before reconfiguring for another architecture.

On MacOS X 10.5 and later systems, you can create libraries and executables that work on multiple system types--known as "fat" or "universal" binaries--by specifying multiple ‘-arch’ options to the compiler but only a single ‘-arch’ option to the preprocessor. Like this:

```
./configure CC="gcc -arch i386 -arch x86_64 -arch ppc -arch ppc64" \
            CXX="g++ -arch i386 -arch x86_64 -arch ppc -arch ppc64" \
            CPP="gcc -E" CXXCPP="g++ -E"
```

This is not guaranteed to produce working output in all cases, you may have to build one architecture at a time and combine the results using the ‘lipo’ tool if you have problems.

9.7 Installation Names

By default, ‘make install’ installs the package’s commands under ‘/usr/local/bin’, include files under ‘/usr/local/include’, etc. You can specify an installation prefix other than ‘/usr/local’ by giving ‘configure’ the option ‘--prefix=PREFIX’, where PREFIX must be an absolute file name.

You can specify separate installation prefixes for architecture-specific files and architecture-independent files. If you pass the option ‘--exec-prefix=PREFIX’ to ‘configure’, the package uses PREFIX as the prefix for installing programs and libraries. Documentation and other data files still use the regular prefix.

In addition, if you use an unusual directory layout you can give options like ‘--bindir=DIR’ to specify different values for particular kinds of files. Run ‘configure --help’ for a list of the directories you can set and what kinds of files go in them. In general, the default for these options is expressed in terms of ‘${prefix}’, so that specifying just ‘--prefix’ will affect all of the other directory specifications that were not explicitly provided.

The most portable way to affect installation locations is to pass the correct locations to ‘configure’; however, many packages provide one or both of the following shortcuts of passing variable assignments to the ‘make install’ command line to change installation locations without having to reconfigure or recompile.

The first method involves providing an override variable for each affected directory. For example, ‘make install prefix=/alternate/directory’ will choose an alternate location for all directory configuration variables that were expressed in terms of ‘${prefix}’. Any directories that were specified during ‘configure’,

but not in terms of '${prefix}', must each be overridden at install time for the entire installation to be relocated. The approach of makefile variable overrides for each directory variable is required by the GNU Coding Standards, and ideally causes no recompilation. However, some platforms have known limitations with the semantics of shared libraries that end up requiring recompilation when using this method, particularly noticeable in packages that use GNU Libtool.

The second method involves providing the 'DESTDIR' variable. For example, 'make install DESTDIR=/alternate/directory' will prepend '/alternate/directory' before all installation names. The approach of 'DESTDIR' overrides is not required by the GNU Coding Standards, and does not work on platforms that have drive letters. On the other hand, it does better at avoiding recompilation issues, and works well even when some directory options were not specified in terms of '${prefix}' at 'configure' time.

9.8 Optional Features

If the package supports it, you can cause programs to be installed with an extra prefix or suffix on their names by giving 'cmake' the option '--program-prefix=PREFIX' or '--program-suffix=SUFFIX'.

Some packages pay attention to '--enable-FEATURE' options to 'configure', where FEATURE indicates an optional part of the package. They may also pay attention to '--with-PACKAGE' options, where PACKAGE is something like 'gnu-as' or 'x' (for the X Window System). The 'README' should mention any '--enable-' and '--with-' options that the package recognizes.

For packages that use the X Window System, 'configure' can usually find the X include and library files automatically, but if it doesn't, you can use the 'configure' options '--x-includes=DIR' and '--x-libraries=DIR' to specify their locations.

Some packages offer the ability to configure how verbose the execution of 'make' will be. For these packages, running './configure --enable-silent-rules' sets the default to minimal output, which can be overridden with 'make V=1'; while running './configure --disable-silent-rules' sets the default to verbose, which can be overridden with 'make V=0'.

9.9 Particular systems

On HP-UX, the default C compiler is not ANSI C compatible. If GNU CC is not installed, it is recommended to use the following options in order to use an ANSI C compiler:

```
./configure CC="cc -Ae -D_XOPEN_SOURCE=500"
```

and if that doesn't work, install pre-built binaries of GCC for HP-UX.

On OSF/1 a.k.a. Tru64, some versions of the default C compiler cannot parse its '<wchar.h>' header file. The option '-nodtk' can be used as

a workaround. If GNU CC is not installed, it is therefore recommended to try

```
./configure CC="cc"
```

and if that doesn't work, try

```
./configure CC="cc -nodtk"
```

On Solaris, don't put '/usr/ucb' early in your 'PATH'. This directory contains several dysfunctional programs; working variants of these programs are available in '/usr/bin'. So, if you need '/usr/ucb' in your 'PATH', put it _after_ '/usr/bin'.

On Haiku, software installed for all users goes in '/boot/common', not '/usr/local'. It is recommended to use the following options:

```
./cmake -DCMAKE_INSTALL_PREFIX=/boot/common
```

9.10 Specifying the System Type

There may be some features 'configure' cannot figure out automatically, but needs to determine by the type of machine the package will run on. Usually, assuming the package is built to be run on the _same_ architectures, 'configure' can figure that out, but if it prints a message saying it cannot guess the machine type, give it the '--build=TYPE' option. TYPE can either be a short name for the system type, such as 'sun4', or a canonical name which has the form CPU-COMPANY-SYSTEM

where SYSTEM can have one of these forms:

- OS
- KERNEL-OS

See the file 'config.sub' for the possible values of each field. If 'config.sub' isn't included in this package, then this package doesn't need to know the machine type.

If you are _building_ compiler tools for cross-compiling, you should use the option '--target=TYPE' to select the type of system they will produce code for.

If you want to _use_ a cross compiler, that generates code for a platform different from the build platform, you should specify the "host" platform (i.e., that on which the generated programs will eventually be run) with '--host=TYPE'.

9.11 Sharing Defaults

If you want to set default values for 'configure' scripts to share, you can create a site shell script called 'config.site' that gives

default values for variables like 'CC', 'cache_file', and 'prefix'. 'configure' looks for 'PREFIX/share/config.site' if it exists, then 'PREFIX/etc/config.site' if it exists. Or, you can set the 'CONFIG_SITE' environment variable to the location of the site script. A warning: not all 'configure' scripts look for a site script.

9.12 Defining Variables

Variables not defined in a site shell script can be set in the environment passed to 'configure'. However, some packages may run configure again during the build, and the customized values of these variables may be lost. In order to avoid this problem, you should set them in the 'configure' command line, using 'VAR=value'. For example:

```
./configure CC=/usr/local2/bin/gcc
```

causes the specified 'gcc' to be used as the C compiler (unless it is overridden in the site shell script).

Unfortunately, this technique does not work for 'CONFIG_SHELL' due to an Autoconf bug. Until the bug is fixed you can use this workaround:

```
CONFIG_SHELL=/bin/bash /bin/bash ./configure CONFIG_SHELL=/bin/bash
```

9.13 'cmake' Invocation

'cmake' recognizes the following options to control how it operates.

- '--help', '-h' print a summary of all of the options to 'cmake', and exit.
- '--help=short', '--help=recursive' print a summary of the options unique to this package's 'configure', and exit. The 'short' variant lists options used only in the top level, while the 'recursive' variant lists options also present in any nested packages.
- '--version', '-V' print the version of Autoconf used to generate the 'configure' script, and exit.
- '--cache-file=FILE' enable the cache: use and save the results of the tests in FILE, traditionally 'config.cache'. FILE defaults to '/dev/null' to disable caching.
- '--config-cache', '-C' alias for '--cache-file=config.cache'.
- '--quiet', '--silent', '-q' do not print messages saying which checks are being made. To suppress all normal output, redirect it to '/dev/null' (any error messages will still be shown).
- '--srcdir=DIR' look for the package's source code in directory DIR. Usually 'configure' can determine that directory automatically.
- '--prefix=DIR' use DIR as the installation prefix.

See also:

Installation Names for more details, including other options available for fine-tuning the installation locations.

- '--no-create', '-n' run the configure checks, but stop before creating any output files.

'cmake' also accepts some other, not widely useful, options. Run 'cmake' --help' for more details.

The 'cmake' script produces an ouput like this:

```
export LIBSUFFIX_4_CMAKE="-DLIB_SUFFIX=64"
export INSTALL_BASEDIR=/home/user/dev/deliveries
cmake -DCMAKE_INSTALL_PREFIX=${INSTALL_BASEDIR}/simcrs-0.1.0 \
 -DWITH_STDAIR_PREFIX=${INSTALL_BASEDIR}/stdair-stable \
 -DWITH_TRADEMGEN_PREFIX=${INSTALL_BASEDIR}/trademgen-stable \
 -DWITH_TRAVELCCM_PREFIX=${INSTALL_BASEDIR}/travelccm-stable \
 -DWITH_AIRSCHED_PREFIX=${INSTALL_BASEDIR}/airsched-stable \
 -DWITH_AIRRAC_PREFIX=${INSTALL_BASEDIR}/airrac-stable \
 -DWITH_RMOL_PREFIX=${INSTALL_BASEDIR}/rmol-stable \
 -DWITH_AIRINV_PREFIX=${INSTALL_BASEDIR}/airinv-stable \
 -DWITH_SIMFQT_PREFIX=${INSTALL_BASEDIR}/simfqt-stable \
 -DCMAKE_BUILD_TYPE:STRING=Debug -DINSTALL_DOC:BOOL=ON ${LIBSUFFIX_4_CMAKE} ..
-- The C compiler identification is GNU
-- The CXX compiler identification is GNU
-- Check for working C compiler: /usr/lib64/ccache/gcc
-- Check for working C compiler: /usr/lib64/ccache/gcc -- works
-- Detecting C compiler ABI info
-- Detecting C compiler ABI info - done
-- Check for working CXX compiler: /usr/lib64/ccache/c++
-- Check for working CXX compiler: /usr/lib64/ccache/c++ -- works
-- Detecting CXX compiler ABI info
-- Detecting CXX compiler ABI info - done
-- Requires Git without specifying any version
-- Current Git revision name: 7a1519ef5b14232c47fe1b1d46db4ae9e65e696d trunk
-- Requires Boost-1.41
-- Boost version: 1.47.0
-- Found the following Boost libraries:
--   regex
--   program_options
--   date_time
--   iostreams
--   serialization
--   filesystem
--   unit_test_framework
--   python
-- Found Boost version: 1.47.0
-- Found BoostWrapper: /usr/include (found suitable version "1.47.0", required is "1.41")
-- Requires Readline without specifying any version
-- Found Readline: /usr/include (found version "6.2")
-- Found Readline version: 6.2
-- Requires MySQL without specifying any version
-- Using mysql-config: /usr/bin/mysql_config
-- Found MySQL: /usr/lib64/mysql/libmysqlclient.so (found version "5.5.18")
-- Found MySQL version: 5.5.18
-- Requires SOCI-3.0
-- SOCI headers are not buried
-- Found SOCI: /usr/lib64/libsoci_core.so (found suitable version "3.1.0", required is "3.0")
-- Found SOCIMySQL: /usr/lib64/libsoci_mysql.so (found suitable version "3.1.0", required is "3.0")
-- Found SOCI with MySQL back-end support version: 3.1.0
-- Requires StdAir-0.43
-- Found StdAir version: 0.44.3
-- Requires AirSched-0.1
-- Found AirSched version: 0.1.3
```

```
-- Requires AirRAC-0.2
-- Found AirRAC version: 0.2.2
-- Requires RMOL-0.25
-- Found RMOL version: 0.25.2
-- Requires AirInv-0.1
-- Found AirInv version: 0.1.2
-- Requires SimFQT-0.1
-- Found SimFQT version: 0.1.2
-- Requires Doxygen without specifying any version
-- Found Doxygen: /usr/bin/doxygen
-- Found DoxygenWrapper: /usr/bin/doxygen (found version "1.7.5")
-- Found Doxygen version: 1.7.5
-- Had to set the linker language for 'simcrslib' to CXX
-- Test 'CRSTestSuite' to be built with 'CRSTestSuite.cpp'
--
-- ============================================================
-- ------------------------------------
-- ---      Project Information      ---
-- ------------------------------------
-- PROJECT_NAME ..................... : simcrs
-- PACKAGE_PRETTY_NAME .............. : SimCRS
-- PACKAGE .......................... : simcrs
-- PACKAGE_NAME ..................... : SIMCRS
-- PACKAGE_BRIEF .................... : C++ Simulated Travel-Oriented Distribution System Library
-- PACKAGE_VERSION .................. : 0.5.0
-- GENERIC_LIB_VERSION .............. : 0.5.0
-- GENERIC_LIB_SOVERSION ............ : 0.5
--
-- ------------------------------------
-- ---       Build Configuration     ---
-- ------------------------------------
-- Modules to build ................. : simcrs
-- Libraries to build/install ....... : simcrslib
-- Binaries to build/install ........ : simcrs
-- Modules to test .................. : simcrs
-- Binaries to test ................. : CRSTestSuitetst
--
-- * Module ......................... : simcrs
--   + Layers to build .............. : .;basic;bom;factory;command;service
--   + Dependencies on other layers .. :
--   + Libraries to build/install .... : simcrslib
--   + Executables to build/install .. : simcrs
--   + Tests to perform ............. : CRSTestSuitetst
--
-- BUILD_SHARED_LIBS ................ : ON
-- CMAKE_BUILD_TYPE ................. : Debug
--  * CMAKE_C_FLAGS .................. :
--  * CMAKE_CXX_FLAGS ................ : -Wall -Werror -DBOOST_VERSION=104700
--  * BUILD_FLAGS .................... :
--  * COMPILE_FLAGS .................. :
-- CMAKE_MODULE_PATH ................ : /home/user/dev/sim/simcrs/simcrsgithub/config/
-- CMAKE_INSTALL_PREFIX ............. : /home/user/dev/deliveries/simcrs-0.5.0
--
-- * Doxygen:
--   - DOXYGEN_VERSION ............. : 1.7.5
--   - DOXYGEN_EXECUTABLE ........... : /usr/bin/doxygen
--   - DOXYGEN_DOT_EXECUTABLE ....... : /usr/bin/dot
--   - DOXYGEN_DOT_PATH ............. : /usr/bin
--
-- ------------------------------------
-- ---  Installation Configuration   ---
-- ------------------------------------
-- INSTALL_LIB_DIR .................. : /home/user/dev/deliveries/simcrs-0.5.0/lib64
-- INSTALL_BIN_DIR .................. : /home/user/dev/deliveries/simcrs-0.5.0/bin
-- CMAKE_INSTALL_RPATH .............. : /home/user/dev/deliveries/simcrs-0.5.0/lib64
-- CMAKE_INSTALL_RPATH_USE_LINK_PATH . : ON
-- INSTALL_INCLUDE_DIR .............. : /home/user/dev/deliveries/simcrs-0.5.0/include
```

```
-- INSTALL_DATA_DIR ................. : /home/user/dev/deliveries/simcrs-0.5.0/share
-- INSTALL_SAMPLE_DIR ............... : /home/user/dev/deliveries/simcrs-0.5.0/share/simcrs/samples
-- INSTALL_DOC ...................... : ON
--
-- ---------------------------------------
-- ---    Packaging Configuration    ---
-- ---------------------------------------
-- CPACK_PACKAGE_CONTACT ............ : Denis Arnaud <denis_arnaud - at - users dot sourceforge dot net>
-- CPACK_PACKAGE_VENDOR ............. : Denis Arnaud
-- CPACK_PACKAGE_VERSION ............ : 0.5.0
-- CPACK_PACKAGE_DESCRIPTION_FILE .... : /home/user/dev/sim/simcrs/simcrsgithub/README
-- CPACK_RESOURCE_FILE_LICENSE ...... : /home/user/dev/sim/simcrs/simcrsgithub/COPYING
-- CPACK_GENERATOR .................. : TBZ2
-- CPACK_DEBIAN_PACKAGE_DEPENDS ..... :
-- CPACK_SOURCE_GENERATOR ........... : TBZ2;TGZ
-- CPACK_SOURCE_PACKAGE_FILE_NAME .... : simcrs-0.5.0
--
-- -------------------------------------
-- ---      External libraries      ---
-- -------------------------------------
--
-- * Boost:
--   - Boost_VERSION ................ : 104700
--   - Boost_LIB_VERSION ............ : 1_47
--   - Boost_HUMAN_VERSION .......... : 1.47.0
--   - Boost_INCLUDE_DIRS ........... : /usr/include
--   - Boost required components ..... : regex;program_options;date_time;iostreams;serialization;filesyste
--   - Boost required libraries ..... : /usr/lib64/libboost_regex-mt.so;/usr/lib64/libboost_iostreams-mt.
--
-- * Readline:
--   - READLINE_VERSION ............. : 6.2
--   - READLINE_INCLUDE_DIR ......... : /usr/include
--   - READLINE_LIBRARY ............. : /usr/lib64/libreadline.so
--
-- * MySQL:
--   - MYSQL_VERSION ................ : 5.5.18
--   - MYSQL_INCLUDE_DIR ............ : /usr/include/mysql
--   - MYSQL_LIBRARIES .............. : /usr/lib64/mysql/libmysqlclient.so
--
-- * SOCI:
--   - SOCI_VERSION ................. : 300100
--   - SOCI_LIB_VERSION ............. : 3_1_0
--   - SOCI_HUMAN_VERSION ........... : 3.1.0
--   - SOCI_INCLUDE_DIR ............. : /usr/include/soci
--   - SOCIMYSQL_INCLUDE_DIR ........ : /usr/include/soci/mysql
--   - SOCI_LIBRARIES ............... : /usr/lib64/libsoci_core.so
--   - SOCIMYSQL_LIBRARIES .......... : /usr/lib64/libsoci_mysql.so
--
-- * StdAir:
--   - STDAIR_VERSION ............... : 0.44.3
--   - STDAIR_BINARY_DIRS ........... : /home/user/dev/deliveries/stdair-0.44.3/bin
--   - STDAIR_EXECUTABLES ........... : stdair
--   - STDAIR_LIBRARY_DIRS .......... : /home/user/dev/deliveries/stdair-0.44.3/lib64
--   - STDAIR_LIBRARIES ............. : stdairlib;stdairuicllib
--   - STDAIR_INCLUDE_DIRS .......... : /home/user/dev/deliveries/stdair-0.44.3/include
--   - STDAIR_SAMPLE_DIR ............ : /home/user/dev/deliveries/stdair-0.44.3/share/stdair/samples
--
-- * AirSched:
--   - AIRSCHED_VERSION ............. : 0.1.3
--   - AIRSCHED_BINARY_DIRS ......... : /home/user/dev/deliveries/airsched-0.1.3/bin
--   - AIRSCHED_EXECUTABLES ......... : airsched
--   - AIRSCHED_LIBRARY_DIRS ........ : /home/user/dev/deliveries/airsched-0.1.3/lib64
--   - AIRSCHED_LIBRARIES ........... : airschedlib
--   - AIRSCHED_INCLUDE_DIRS ........ : /home/user/dev/deliveries/airsched-0.1.3/include
--
-- * AirRAC:
--   - AIRRAC_VERSION ............... : 0.2.2
```

```
--   - AIRRAC_BINARY_DIRS ............ : /home/user/dev/deliveries/airrac-0.2.2/bin
--   - AIRRAC_EXECUTABLES ............ : airrac
--   - AIRRAC_LIBRARY_DIRS ........... : /home/user/dev/deliveries/airrac-0.2.2/lib64
--   - AIRRAC_LIBRARIES .............. : airraclib
--   - AIRRAC_INCLUDE_DIRS ........... : /home/user/dev/deliveries/airrac-0.2.2/include
--
-- * RMOL:
--   - RMOL_VERSION .................. : 0.25.2
--   - RMOL_BINARY_DIRS .............. : /home/user/dev/deliveries/rmol-0.25.2/bin
--   - RMOL_EXECUTABLES .............. : rmol
--   - RMOL_LIBRARY_DIRS ............. : /home/user/dev/deliveries/rmol-0.25.2/lib
--   - RMOL_LIBRARIES ................ : rmollib
--   - RMOL_INCLUDE_DIRS ............. : /home/user/dev/deliveries/rmol-0.25.2/include
--
-- * AirInv:
--   - AIRINV_VERSION ................ : 0.1.2
--   - AIRINV_BINARY_DIRS ............ : /home/user/dev/deliveries/airinv-0.1.2/bin
--   - AIRINV_EXECUTABLES ............ : airinv;airinv_parseInventory
--   - AIRINV_LIBRARY_DIRS ........... : /home/user/dev/deliveries/airinv-0.1.2/lib
--   - AIRINV_LIBRARIES .............. : airinvlib
--   - AIRINV_INCLUDE_DIRS ........... : /home/user/dev/deliveries/airinv-0.1.2/include
--
-- * SimFQT:
--   - SIMFQT_VERSION ................ : 0.1.2
--   - SIMFQT_BINARY_DIRS ............ : /home/user/dev/deliveries/simfqt-0.1.2/bin
--   - SIMFQT_EXECUTABLES ............ : simfqt;simfqt_parseFareRules
--   - SIMFQT_LIBRARY_DIRS ........... : /home/user/dev/deliveries/simfqt-0.1.2/lib64
--   - SIMFQT_LIBRARIES .............. : simfqtlib
--   - SIMFQT_INCLUDE_DIRS ........... : /home/user/dev/deliveries/simfqt-0.1.2/include
--
-- Change a value with: cmake -D<Variable>=<Value>
-- ================================================================
--
-- Configuring done
-- Generating done
-- Build files have been written to: /home/user/dev/sim/simcrs/simcrsgithub/build
```

It is recommended that you check if your library has been compiled and linked properly and works as expected. To do so, you should execute the testing process 'make check'. As a result, you should obtain a similar report:

```
[  0%] Built target hdr_cfg_simcrs
[ 90%] Built target simcrslib
[100%] Built target CRSTestSuitetst
Scanning dependencies of target check_simcrstst
Test project /home/user/dev/sim/simcrs/simcrsgithub/build/test/simcrs
    Start 1: CRSTestSuitetst
1/1 Test #1: CRSTestSuitetst ..................   Passed    0.15 sec

100% tests passed, 0 tests failed out of 1

Total Test time (real) =   0.33 sec
[100%] Built target check_simcrstst
Scanning dependencies of target check
[100%] Built target check
```

Check if all the executed tests PASSED. If not, please contact us by filling a bug-report.

Finally, you should install the compiled and linked library, include files and (optionally) HTML and PDF documentation by typing:

```
make install
```

```
Depending on the PREFIX settings during configuration, you might need
the root (administrator) access to perform this step.

Eventually, you might invoke the following command

make clean

to remove all files created during compilation process, or even

cd ~/dev/sim/simcrsgit
rm -rf build && mkdir build
cd build

to remove everything.
```

10 Linking with SimCRS

10.1 Table of Contents

10.2 Introduction

There are two convenient methods of linking your programs with the SimCRS library. The first one employs the `'pkg-config'` command (see http://pkgconfig.freedesktop.org/), whereas the second one uses `'simcrs-config'` script. These methods are shortly described below.

10.3 Dependencies

The SimCRS library depends on several other C++ components.

10.3.1 StdAir

Among them, as for now, only StdAir has been packaged. The support for StdAir is taken in charge by a dedicated M4 macro file (namely, `'stdair.m4'`), from the configuration script (generated thanks to `'configure.ac'`).

10.3.2 Other Simulation-Related Components

SimCRS, as shown on the diagram below, depends on

- `AirSched`
- `SimFQT`
- `AirRAC`
- `RMOL`
- `AirInv`
- `AvlCal`
- `SimLFS`

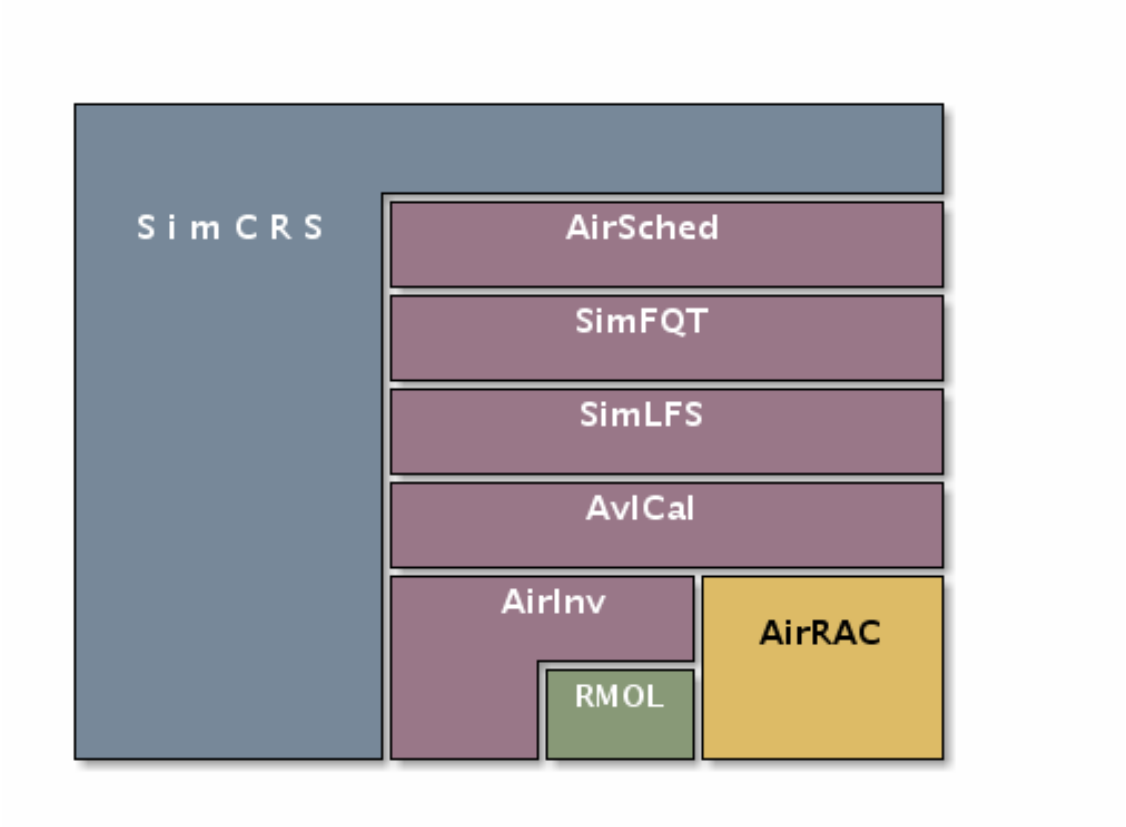


Figure 1: SimCRS Dependencies

10.4 Using the pkg-config command

'`pkg-config`' is a helper tool used when compiling applications and libraries. It helps you insert the correct compiler and linker options. The syntax of the '`pkg-config`' is as follows:

```
pkg-config <options> <library_name>
```

For instance, assuming that you need to compile an SimCRS based program 'my_prog.cpp', you should use the following command:

```
g++ `pkg-config --cflags simcrs` -o my_prog my_prog.cpp `pkg-config --libs simcrs
    `
```

For more information see the 'pkg-config' man pages.

10.5 Using the simcrs-config script

SimCRS provides a shell script called 'simcrs-config', which is installed by default in '$prefix/bin' ('/usr/local/bin') directory. It can be used to simplify compilation and linking of SimCRS based programs. The usage of this script is quite similar to the usage of the 'pkg-config' command.

Assuming that you need to compile the program 'my_prog.cpp' you can now do that with the following command:

```
g++ `simcrs-config --cflags` -o my_prog_opt my_prog.cpp `simcrs-config --libs`
```

A list of 'simcrs-config' options can be obtained by typing:

```
simcrs-config --help
```

If the 'simcrs-config' command is not found by your shell, you should add its location '$prefix/bin' to the PATH environment variable, e.g.:

```
export PATH=/usr/local/bin:$PATH
```

10.6 M4 macro for the GNU Autotools

A M4 macro file is delivered with SimCRS, namely 'simcrs.m4', which can be found in, e.g., '/usr/share/aclocal'. When used by a 'configure' script, thanks to he 'AM_PATH_SimCRS' macro (specified in the M4 macro file), the following Makefile variables are then defined:

- 'SimCRS_VERSION' (e.g., defined to 0.23.0)
- 'SimCRS_CFLAGS' (e.g., defined to '-I${prefix}/include')
- 'SimCRS_LIBS' (e.g., defined to '-L${prefix}/lib -lsimcrs')

10.7 Using SimCRS with dynamic linking

When using static linking some of the library routines in SimCRS are copied into your executable program. This can lead to unnecessary large executables. To avoid having too large executable files you may use dynamic linking instead. Dynamic linking means that the actual linking is performed when the program is executed. This requires that the system is able to locate the shared SimCRS library file during your program execution. If you install the SimCRS library using a non-standard prefix, the 'LD_LIBRARY_PATH' environment variable might be used to inform the linker of the dynamic library location, e.g.:

```
export LD_LIBRARY_PATH=<SimCRS installation prefix>/lib:$LD_LIBRARY_PATH
```

11 Test Rules

This section describes how the functionality of the SimCRS library should be verified. In the `'test/simcrs'` subdirectory, test source files are provided. All functionality should be tested using these test source files.

11.1 The Test Source Files

Each new SimCRS module/class should be accompanied with a test source file. The test source file is an implementation in C++ that tests the functionality of a function/class or a group of functions/classes called test suites. The test source file should test relevant parameter settings and input/output relations to guarantee correct functionality of the corresponding classes/functions. The test source files should be maintained using version control and updated whenever new functionality is added to the SimCRS library.

The test source file should print relevant data to a standard output that can be used to verify the functionality. All relevant parameter settings should be tested.

The test source file should be placed in the `'test/simcrs'` subdirectory and should have a name ending with `'TestSuite.cpp'`.

11.2 The Reference File

Consider a test source file named `'YieldTestSuite.cpp'`. A reference file named `'YieldTestSuite.ref'` should accompany the test source file. The reference file contains a reference printout of the standard output generated when running the test program. The reference file should be maintained using version control and updated according to the test source file.

11.3 Testing SimCRS Library

One can compile and execute all test programs from the `'test/simcrs'` sub-directory by typing:

```
% make check
```

after successful compilation of the SimCRS library.

12 Users Guide

12.1 Table of Contents

12.2 Introduction

The SimCRS library contains classes for airline business management. This document does not cover all the aspects of the SimCRS library. It does however explain the most important things you need to know in order to start using SimCRS.

12.3 Get Started

12.3.1 Get the SimCRS library

Clone locally the full Git project:

```
cd ~
mkdir -p dev/sim
cd ~/dev/sim
git clone git://simcrs.git.sourceforge.net/gitroot/simcrs/simcrs simcrsgit
cd simcrsgit
git checkout trunk
```

12.3.2 Build the SimCRS project

Link with StdAir, create the distribution package (say, 0.5.0) and compile using the following commands:

```
cd ~/dev/sim/simcrsgit
rm -rf build && mkdir -p build
cd build
cmake -DCMAKE_INSTALL_PREFIX=~/dev/deliveries/simcrs-0.5.0 \
 -DWITH_STDAIR_PREFIX=~/dev/deliveries/stdair-stable \
 -DCMAKE_BUILD_TYPE:STRING=Debug -DINSTALL_DOC:BOOL=ON ..
make
```

12.3.3 Build and Run the Tests

After building the SimCRS project, the following commands run the tests:

```
cd ~/dev/sim/simcrsgit
cd build
make check
```

As a result, you should obtain a similar report:

```
[  0%] Built target hdr_cfg_simcrs
[ 96%] Built target simcrslib
[100%] Built target AirlineScheduleTestSuitetst
Scanning dependencies of target check_simcrststst
Test project /home/dan/dev/sim/simcrs/simcrsgithub/build/test/simcrs
    Start 1: AirlineScheduleTestSuitetst
1/1 Test #1: AirlineScheduleTestSuitetst ......   Passed    0.15 sec

100% tests passed, 0 tests failed out of 1

Total Test time (real) =   0.40 sec
[100%] Built target check_simcrststst
Scanning dependencies of target check
[100%] Built target check
```

12.3.4 Install the SimCRS Project (Binaries, Documentation)

After the step Build the SimCRS project, to install the library and its header files, type:

```
cd ~/dev/sim/simcrsgit
cd build
make install
```

You can check that the executables and other required files have been copied into the given final directory:

```
cd ~dev/deliveries/simcrs-0.5.0
```

To generate the SimCRS project documentation, the commands are:

```
cd ~/dev/sim/simcrsgit
cd build
make doc
```

The SimCRS project documentation is available in the following formats: HTML, LaTeX. Those documents are available in a subdirectory:

```
cd ~/dev/sim/simcrsgit
cd build
cd doc
```

12.4 Input file of SimCRS Project

The schedule input file structure should look like the following sample:

Each line, beyond the header, represents a schedule entry, i.e., the specification of a given flight-period (see `SIMCRS::FlightPeriodStruct`). The fields are as follows:

- Flights section
  - AirlineCode (e.g., `BA`)
  - FlightNumber (e.g., `9`)
  - Start of the flight departure period (e.g., `2007-04-20`)
  - End of the flight departure period (e.g., `2007-06-30`)

  - Day-Of-the-Week for the flight departure period (DOW) (e.g., `0000011`)
  - Leg section
  - Segment section

- Leg section
  - BoardPoint (e.g., `LHR`)
  - OffPoint (e.g., `BKK`)
  - BoardTime (e.g., `22`:00)
  - ArrivalTime (e.g., `15`:15)
  - ArrivalDateOffSet (e.g., `+1`)
  - ElapsedTime (e.g., `11`:15)
  - Leg-cabin section
- Leg-cabin section
  - Cabin code (e.g., `F`, `J`, `W` or `Y`)
  - Capacity (e.g., respectively `5`, `12`, `20` or `300`)
- Segment section
  - Specificity flag:
    * `0` means that all the segments behave the same way, i.e., have got the same dressing (distribution and order of the booking classes per cabin)
    * `1` means that each segment bahave differently. The full specification of each of those segments must therefore be given.
  - Segment-cabin section
  - Fare family section
- Segment-cabin section
  - Cabin code (e.g., `F`, `J`, `W` or `Y`)
  - List of (one-letter-code) booking classes for the cabin (e.g, respectively `FA`, `JCDI`, `WT` or `YBHKMLSQ`)
- Fare family section
  - Fare family code (e.g., `1`)
  - List of (one-letter-code) booking classes for the fare family (e.g, respectively `FA`, `JCDI`, `WT` or `YBHKMLSQ`)

Some fare input examples (including the example above named `schedule03.csv`) are given in the `StdAir project`.

12.5 The schedule BOM Tree

The schedule-related Business Object Model (BOM) tree is a structure allowing to store all the `SIMCRS::FlightPeriodStruct` objects of the simulation. That is why parsing an input file, containing the specification for all the flight-periods, is more convenient (

See also:

the previous section Input file of SimCRS Project).

As it may be time consuming, and it for sure requires some know-how, to first build such a schedule input file, a small sample BOM tree is provided by default when needed.

12.5.1 Build of the schedule BOM tree

First, a BOM root object (i.e., a root for all the classes in the project) is instantiated by the `stdair::STDAIR_ServiceContext` context object, when the `stdair::STDAIR_Service` is itself instantiated (during the instantiation of the `SIMCRS::SIMCRS_Service` object).

The corresponding `type` (class) `stdair::BomRoot` is defined in the StdAir library.

Then, the BOM root can be either constructed thanks to the `SIMCRS::SIMCRS_Service::buildSampleBom()` method:

```
\textcolor{keywordtype}{void} buildSampleBom ();
```

or can be constructed using the schedule input file described above thanks to the `SIMCRS::SIMCRS_Service::parseAndLoad (const stdair::Filename_T&)` method:

```
\textcolor{keywordtype}{void} parseAndLoad (\textcolor{keyword}{const} stdair::ScheduleFilePath&,
```

12.5.2 Display of the schedule BOM tree

Note:

That feature (of BOM tree display) has not been implemented yet. Do not hesitate to open a ticket if you would like to have it implemented more quickly.

The schedule BOM tree can be displayed as done in the `batches::simcrs.cpp` program:

When the default BOM tree is used (`-b/--builtin` option of the main program `simcrs.cpp`), the schedule BOM tree display (for now, corresponding to `schedule01.csv` parsed by `SIMCRS::parseInventory`) should look like:

```
================================================================
BomRoot:  -- ROOT --
================================================================
++++++++++++++++++++++++++++++++++++++++++++++++
Inventory: SQ
++++++++++++++++++++++++++++++++++++++++++++++++
****************************************
FlightDate: SQ11, 2010-Jan-15
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-15, SIN-BKK, 2010-Jan-15, 08:20:00, 2010-Jan-15, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-15, SIN-BKK 2010-Jan-15, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 2, 298,
      9, 0, 0, 0, 0, 0,
****************************************
```

```
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-15, SIN-BKK 2010-Jan-15, Y, 1, 0, 0, 0, 2, 298, 0,
SQ11 2010-Jan-15, SIN-BKK 2010-Jan-15, Y, 2, 0, 0, 0, 2, 298, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-15, SIN-BKK 2010-Jan-15, Y, 1, Y, 300 (0), 0, 0, 0, 2, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-15, SIN-BKK 2010-Jan-15, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-16
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-16, SIN-BKK, 2010-Jan-16, 08:20:00, 2010-Jan-16, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-16, SIN-BKK 2010-Jan-16, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 1.83244e-319, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-16, SIN-BKK 2010-Jan-16, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-16, SIN-BKK 2010-Jan-16, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-16, SIN-BKK 2010-Jan-16, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-16, SIN-BKK 2010-Jan-16, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-17
****************************************
****************************************
```

```
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-17, SIN-BKK, 2010-Jan-17, 08:20:00, 2010-Jan-17, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-17, SIN-BKK 2010-Jan-17, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 1.58896e-319, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-17, SIN-BKK 2010-Jan-17, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-17, SIN-BKK 2010-Jan-17, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-17, SIN-BKK 2010-Jan-17, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-17, SIN-BKK 2010-Jan-17, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-18
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-18, SIN-BKK, 2010-Jan-18, 08:20:00, 2010-Jan-18, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-18, SIN-BKK 2010-Jan-18, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-18, SIN-BKK 2010-Jan-18, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-18, SIN-BKK 2010-Jan-18, Y, 2, 0, 0, 0, 0, 300, 0,
```

```
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-18, SIN-BKK 2010-Jan-18, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-18, SIN-BKK 2010-Jan-18, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-19
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-19, SIN-BKK, 2010-Jan-19, 08:20:00, 2010-Jan-19, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-19, SIN-BKK 2010-Jan-19, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-19, SIN-BKK 2010-Jan-19, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-19, SIN-BKK 2010-Jan-19, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-19, SIN-BKK 2010-Jan-19, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-19, SIN-BKK 2010-Jan-19, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-20
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-20, SIN-BKK, 2010-Jan-20, 08:20:00, 2010-Jan-20, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
```

```
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-20, SIN-BKK 2010-Jan-20, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-20, SIN-BKK 2010-Jan-20, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-20, SIN-BKK 2010-Jan-20, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-20, SIN-BKK 2010-Jan-20, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-20, SIN-BKK 2010-Jan-20, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-21
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-21, SIN-BKK, 2010-Jan-21, 08:20:00, 2010-Jan-21, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-21, SIN-BKK 2010-Jan-21, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-21, SIN-BKK 2010-Jan-21, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-21, SIN-BKK 2010-Jan-21, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-21, SIN-BKK 2010-Jan-21, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-21, SIN-BKK 2010-Jan-21, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
```

```
****************************************
FlightDate: SQ11, 2010-Jan-22
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-22, SIN-BKK, 2010-Jan-22, 08:20:00, 2010-Jan-22, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-22, SIN-BKK 2010-Jan-22, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-22, SIN-BKK 2010-Jan-22, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-22, SIN-BKK 2010-Jan-22, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-22, SIN-BKK 2010-Jan-22, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-22, SIN-BKK 2010-Jan-22, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-23
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-23, SIN-BKK, 2010-Jan-23, 08:20:00, 2010-Jan-23, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-23, SIN-BKK 2010-Jan-23, Y, 300, 300, 0, 0, 0, 0, 0, 0, 6.64029e-31
      9, 0, 300, 9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
```

```
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-23, SIN-BKK 2010-Jan-23, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-23, SIN-BKK 2010-Jan-23, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-23, SIN-BKK 2010-Jan-23, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-23, SIN-BKK 2010-Jan-23, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-24
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-24, SIN-BKK, 2010-Jan-24, 08:20:00, 2010-Jan-24, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-24, SIN-BKK 2010-Jan-24, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-24, SIN-BKK 2010-Jan-24, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-24, SIN-BKK 2010-Jan-24, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-24, SIN-BKK 2010-Jan-24, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-24, SIN-BKK 2010-Jan-24, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-25
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-25, SIN-BKK, 2010-Jan-25, 08:20:00, 2010-Jan-25, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
```

```
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-25, SIN-BKK 2010-Jan-25, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-25, SIN-BKK 2010-Jan-25, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-25, SIN-BKK 2010-Jan-25, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-25, SIN-BKK 2010-Jan-25, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-25, SIN-BKK 2010-Jan-25, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-26
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-26, SIN-BKK, 2010-Jan-26, 08:20:00, 2010-Jan-26, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-26, SIN-BKK 2010-Jan-26, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-26, SIN-BKK 2010-Jan-26, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-26, SIN-BKK 2010-Jan-26, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-26, SIN-BKK 2010-Jan-26, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
```

```
       0, 0, 0, 0,
SQ11 2010-Jan-26, SIN-BKK 2010-Jan-26, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-27
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-27, SIN-BKK, 2010-Jan-27, 08:20:00, 2010-Jan-27, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-27, SIN-BKK 2010-Jan-27, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-27, SIN-BKK 2010-Jan-27, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-27, SIN-BKK 2010-Jan-27, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-27, SIN-BKK 2010-Jan-27, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-27, SIN-BKK 2010-Jan-27, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-28
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-28, SIN-BKK, 2010-Jan-28, 08:20:00, 2010-Jan-28, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-28, SIN-BKK 2010-Jan-28, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
```

```
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-28, SIN-BKK 2010-Jan-28, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-28, SIN-BKK 2010-Jan-28, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-28, SIN-BKK 2010-Jan-28, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-28, SIN-BKK 2010-Jan-28, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-29
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-29, SIN-BKK, 2010-Jan-29, 08:20:00, 2010-Jan-29, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-29, SIN-BKK 2010-Jan-29, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-29, SIN-BKK 2010-Jan-29, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-29, SIN-BKK 2010-Jan-29, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-29, SIN-BKK 2010-Jan-29, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-29, SIN-BKK 2010-Jan-29, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-30
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
```

```
      apsed, Distance, Capacity,
SQ11 2010-Jan-30, SIN-BKK, 2010-Jan-30, 08:20:00, 2010-Jan-30, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-30, SIN-BKK 2010-Jan-30, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-30, SIN-BKK 2010-Jan-30, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-30, SIN-BKK 2010-Jan-30, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-30, SIN-BKK 2010-Jan-30, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-30, SIN-BKK 2010-Jan-30, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Jan-31
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Jan-31, SIN-BKK, 2010-Jan-31, 08:20:00, 2010-Jan-31, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Jan-31, SIN-BKK 2010-Jan-31, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Jan-31, SIN-BKK 2010-Jan-31, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Jan-31, SIN-BKK 2010-Jan-31, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
```

```
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Jan-31, SIN-BKK 2010-Jan-31, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Jan-31, SIN-BKK 2010-Jan-31, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-01
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-01, SIN-BKK, 2010-Feb-01, 08:20:00, 2010-Feb-01, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-01, SIN-BKK 2010-Feb-01, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-01, SIN-BKK 2010-Feb-01, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-01, SIN-BKK 2010-Feb-01, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-01, SIN-BKK 2010-Feb-01, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-01, SIN-BKK 2010-Feb-01, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-02
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-02, SIN-BKK, 2010-Feb-02, 08:20:00, 2010-Feb-02, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-02, SIN-BKK 2010-Feb-02, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
```

```
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-02, SIN-BKK 2010-Feb-02, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-02, SIN-BKK 2010-Feb-02, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-02, SIN-BKK 2010-Feb-02, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-02, SIN-BKK 2010-Feb-02, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-03
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-03, SIN-BKK, 2010-Feb-03, 08:20:00, 2010-Feb-03, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-03, SIN-BKK 2010-Feb-03, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-03, SIN-BKK 2010-Feb-03, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-03, SIN-BKK 2010-Feb-03, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-03, SIN-BKK 2010-Feb-03, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-03, SIN-BKK 2010-Feb-03, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-04
****************************************
```

```
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-04, SIN-BKK, 2010-Feb-04, 08:20:00, 2010-Feb-04, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-04, SIN-BKK 2010-Feb-04, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-04, SIN-BKK 2010-Feb-04, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-04, SIN-BKK 2010-Feb-04, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-04, SIN-BKK 2010-Feb-04, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-04, SIN-BKK 2010-Feb-04, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-05
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-05, SIN-BKK, 2010-Feb-05, 08:20:00, 2010-Feb-05, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-05, SIN-BKK 2010-Feb-05, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-05, SIN-BKK 2010-Feb-05, Y, 1, 0, 0, 0, 0, 300, 0,
```

```
SQ11 2010-Feb-05, SIN-BKK 2010-Feb-05, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-05, SIN-BKK 2010-Feb-05, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-05, SIN-BKK 2010-Feb-05, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-06
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-06, SIN-BKK, 2010-Feb-06, 08:20:00, 2010-Feb-06, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-06, SIN-BKK 2010-Feb-06, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-06, SIN-BKK 2010-Feb-06, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-06, SIN-BKK 2010-Feb-06, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-06, SIN-BKK 2010-Feb-06, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-06, SIN-BKK 2010-Feb-06, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-07
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-07, SIN-BKK, 2010-Feb-07, 08:20:00, 2010-Feb-07, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
```

```
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-07, SIN-BKK 2010-Feb-07, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-07, SIN-BKK 2010-Feb-07, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-07, SIN-BKK 2010-Feb-07, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-07, SIN-BKK 2010-Feb-07, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-07, SIN-BKK 2010-Feb-07, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-08
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-08, SIN-BKK, 2010-Feb-08, 08:20:00, 2010-Feb-08, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-08, SIN-BKK 2010-Feb-08, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-08, SIN-BKK 2010-Feb-08, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-08, SIN-BKK 2010-Feb-08, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-08, SIN-BKK 2010-Feb-08, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-08, SIN-BKK 2010-Feb-08, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
```

```
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-09
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-09, SIN-BKK, 2010-Feb-09, 08:20:00, 2010-Feb-09, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-09, SIN-BKK 2010-Feb-09, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-09, SIN-BKK 2010-Feb-09, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-09, SIN-BKK 2010-Feb-09, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-09, SIN-BKK 2010-Feb-09, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-09, SIN-BKK 2010-Feb-09, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-10
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-10, SIN-BKK, 2010-Feb-10, 08:20:00, 2010-Feb-10, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-10, SIN-BKK 2010-Feb-10, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
```

```
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-10, SIN-BKK 2010-Feb-10, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-10, SIN-BKK 2010-Feb-10, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-10, SIN-BKK 2010-Feb-10, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-10, SIN-BKK 2010-Feb-10, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-11
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-11, SIN-BKK, 2010-Feb-11, 08:20:00, 2010-Feb-11, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-11, SIN-BKK 2010-Feb-11, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-11, SIN-BKK 2010-Feb-11, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-11, SIN-BKK 2010-Feb-11, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-11, SIN-BKK 2010-Feb-11, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-11, SIN-BKK 2010-Feb-11, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-12
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-12, SIN-BKK, 2010-Feb-12, 08:20:00, 2010-Feb-12, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
```

```
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-12, SIN-BKK 2010-Feb-12, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-12, SIN-BKK 2010-Feb-12, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-12, SIN-BKK 2010-Feb-12, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-12, SIN-BKK 2010-Feb-12, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-12, SIN-BKK 2010-Feb-12, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-13
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-13, SIN-BKK, 2010-Feb-13, 08:20:00, 2010-Feb-13, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-13, SIN-BKK 2010-Feb-13, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-13, SIN-BKK 2010-Feb-13, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-13, SIN-BKK 2010-Feb-13, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
```

```
SQ11 2010-Feb-13, SIN-BKK 2010-Feb-13, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-13, SIN-BKK 2010-Feb-13, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-14
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-14, SIN-BKK, 2010-Feb-14, 08:20:00, 2010-Feb-14, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-14, SIN-BKK 2010-Feb-14, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-14, SIN-BKK 2010-Feb-14, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-14, SIN-BKK 2010-Feb-14, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-14, SIN-BKK 2010-Feb-14, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-14, SIN-BKK 2010-Feb-14, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-15
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-15, SIN-BKK, 2010-Feb-15, 08:20:00, 2010-Feb-15, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-15, SIN-BKK 2010-Feb-15, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
```

```
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-15, SIN-BKK 2010-Feb-15, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-15, SIN-BKK 2010-Feb-15, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-15, SIN-BKK 2010-Feb-15, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-15, SIN-BKK 2010-Feb-15, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-16
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-16, SIN-BKK, 2010-Feb-16, 08:20:00, 2010-Feb-16, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-16, SIN-BKK 2010-Feb-16, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-16, SIN-BKK 2010-Feb-16, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-16, SIN-BKK 2010-Feb-16, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-16, SIN-BKK 2010-Feb-16, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-16, SIN-BKK 2010-Feb-16, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-17
****************************************
****************************************
Leg-Dates:
----------
```

```
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-17, SIN-BKK, 2010-Feb-17, 08:20:00, 2010-Feb-17, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-17, SIN-BKK 2010-Feb-17, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-17, SIN-BKK 2010-Feb-17, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-17, SIN-BKK 2010-Feb-17, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-17, SIN-BKK 2010-Feb-17, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-17, SIN-BKK 2010-Feb-17, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-18
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-18, SIN-BKK, 2010-Feb-18, 08:20:00, 2010-Feb-18, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-18, SIN-BKK 2010-Feb-18, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-18, SIN-BKK 2010-Feb-18, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-18, SIN-BKK 2010-Feb-18, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
```

```
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-18, SIN-BKK 2010-Feb-18, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-18, SIN-BKK 2010-Feb-18, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-19
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-19, SIN-BKK, 2010-Feb-19, 08:20:00, 2010-Feb-19, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-19, SIN-BKK 2010-Feb-19, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-19, SIN-BKK 2010-Feb-19, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-19, SIN-BKK 2010-Feb-19, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-19, SIN-BKK 2010-Feb-19, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-19, SIN-BKK 2010-Feb-19, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-20
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-20, SIN-BKK, 2010-Feb-20, 08:20:00, 2010-Feb-20, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-20, SIN-BKK 2010-Feb-20, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
```

```
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-20, SIN-BKK 2010-Feb-20, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-20, SIN-BKK 2010-Feb-20, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-20, SIN-BKK 2010-Feb-20, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-20, SIN-BKK 2010-Feb-20, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-21
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-21, SIN-BKK, 2010-Feb-21, 08:20:00, 2010-Feb-21, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-21, SIN-BKK 2010-Feb-21, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-21, SIN-BKK 2010-Feb-21, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-21, SIN-BKK 2010-Feb-21, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-21, SIN-BKK 2010-Feb-21, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-21, SIN-BKK 2010-Feb-21, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-22
```

```
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-22, SIN-BKK, 2010-Feb-22, 08:20:00, 2010-Feb-22, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-22, SIN-BKK 2010-Feb-22, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-22, SIN-BKK 2010-Feb-22, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-22, SIN-BKK 2010-Feb-22, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-22, SIN-BKK 2010-Feb-22, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-22, SIN-BKK 2010-Feb-22, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-23
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-23, SIN-BKK, 2010-Feb-23, 08:20:00, 2010-Feb-23, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-23, SIN-BKK 2010-Feb-23, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
```

```
SQ11 2010-Feb-23, SIN-BKK 2010-Feb-23, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-23, SIN-BKK 2010-Feb-23, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-23, SIN-BKK 2010-Feb-23, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-23, SIN-BKK 2010-Feb-23, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-24
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-24, SIN-BKK, 2010-Feb-24, 08:20:00, 2010-Feb-24, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-24, SIN-BKK 2010-Feb-24, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-24, SIN-BKK 2010-Feb-24, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-24, SIN-BKK 2010-Feb-24, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-24, SIN-BKK 2010-Feb-24, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-24, SIN-BKK 2010-Feb-24, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-25
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-25, SIN-BKK, 2010-Feb-25, 08:20:00, 2010-Feb-25, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
```

```
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-25, SIN-BKK 2010-Feb-25, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-25, SIN-BKK 2010-Feb-25, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-25, SIN-BKK 2010-Feb-25, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-25, SIN-BKK 2010-Feb-25, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-25, SIN-BKK 2010-Feb-25, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-26
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-26, SIN-BKK, 2010-Feb-26, 08:20:00, 2010-Feb-26, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-26, SIN-BKK 2010-Feb-26, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-26, SIN-BKK 2010-Feb-26, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-26, SIN-BKK 2010-Feb-26, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-26, SIN-BKK 2010-Feb-26, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-26, SIN-BKK 2010-Feb-26, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
```

```
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-27
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-27, SIN-BKK, 2010-Feb-27, 08:20:00, 2010-Feb-27, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-27, SIN-BKK 2010-Feb-27, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-27, SIN-BKK 2010-Feb-27, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-27, SIN-BKK 2010-Feb-27, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-27, SIN-BKK 2010-Feb-27, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-27, SIN-BKK 2010-Feb-27, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ11, 2010-Feb-28
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ11 2010-Feb-28, SIN-BKK, 2010-Feb-28, 08:20:00, 2010-Feb-28, 11:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ11 2010-Feb-28, SIN-BKK 2010-Feb-28, Y, 300, 300, 0, 0, 0, 0, 0, 0, 0, 0, 300,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
```

```
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ11 2010-Feb-28, SIN-BKK 2010-Feb-28, Y, 1, 0, 0, 0, 0, 300, 0,
SQ11 2010-Feb-28, SIN-BKK 2010-Feb-28, Y, 2, 0, 0, 0, 0, 300, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ11 2010-Feb-28, SIN-BKK 2010-Feb-28, Y, 1, Y, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ11 2010-Feb-28, SIN-BKK 2010-Feb-28, Y, 2, M, 300 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-15
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-15, SIN-HND, 2010-Jan-15, 09:20:00, 2010-Jan-15, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-15, SIN-HND 2010-Jan-15, Y, 200, 200, 2.082e+121, 5.53287e-48, 5.20
      268e-90, 0, 1.31346e-47, 1.05119e-153, 2.78986e+179, 0, 200, 9, 3.66962e-62, 1.08
      54e-71, 6.74783e-67, 6.9835e-77, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-15, SIN-HND 2010-Jan-15, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-15, SIN-HND 2010-Jan-15, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-15, SIN-HND 2010-Jan-15, Y, 1, Y13856, 200 (0), 0, 0, 0, 0, 0 (0),
      0, 0, 0, 0, 0, 0,
SQ12 2010-Jan-15, SIN-HND 2010-Jan-15, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-16
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
```

```
SQ12 2010-Jan-16, SIN-HND, 2010-Jan-16, 09:20:00, 2010-Jan-16, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-16, SIN-HND 2010-Jan-16, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 2.63638e-319, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-16, SIN-HND 2010-Jan-16, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-16, SIN-HND 2010-Jan-16, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-16, SIN-HND 2010-Jan-16, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-16, SIN-HND 2010-Jan-16, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-17
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-17, SIN-HND, 2010-Jan-17, 09:20:00, 2010-Jan-17, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-17, SIN-HND 2010-Jan-17, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 2.39291e-319, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-17, SIN-HND 2010-Jan-17, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-17, SIN-HND 2010-Jan-17, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
```

```
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-17, SIN-HND 2010-Jan-17, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-17, SIN-HND 2010-Jan-17, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-18
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-18, SIN-HND, 2010-Jan-18, 09:20:00, 2010-Jan-18, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-18, SIN-HND 2010-Jan-18, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 2.14469e-319, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-18, SIN-HND 2010-Jan-18, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-18, SIN-HND 2010-Jan-18, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-18, SIN-HND 2010-Jan-18, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-18, SIN-HND 2010-Jan-18, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-19
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-19, SIN-HND, 2010-Jan-19, 09:20:00, 2010-Jan-19, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-19, SIN-HND 2010-Jan-19, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
```

```
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-19, SIN-HND 2010-Jan-19, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-19, SIN-HND 2010-Jan-19, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-19, SIN-HND 2010-Jan-19, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-19, SIN-HND 2010-Jan-19, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-20
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-20, SIN-HND, 2010-Jan-20, 09:20:00, 2010-Jan-20, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-20, SIN-HND 2010-Jan-20, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-20, SIN-HND 2010-Jan-20, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-20, SIN-HND 2010-Jan-20, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-20, SIN-HND 2010-Jan-20, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-20, SIN-HND 2010-Jan-20, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-21
****************************************
****************************************
```

```
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-21, SIN-HND, 2010-Jan-21, 09:20:00, 2010-Jan-21, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-21, SIN-HND 2010-Jan-21, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-21, SIN-HND 2010-Jan-21, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-21, SIN-HND 2010-Jan-21, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-21, SIN-HND 2010-Jan-21, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-21, SIN-HND 2010-Jan-21, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-22
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-22, SIN-HND, 2010-Jan-22, 09:20:00, 2010-Jan-22, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-22, SIN-HND 2010-Jan-22, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-22, SIN-HND 2010-Jan-22, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-22, SIN-HND 2010-Jan-22, Y, 2, 0, 0, 0, 0, 200, 0,
```

```
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-22, SIN-HND 2010-Jan-22, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-22, SIN-HND 2010-Jan-22, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-23
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-23, SIN-HND, 2010-Jan-23, 09:20:00, 2010-Jan-23, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-23, SIN-HND 2010-Jan-23, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-23, SIN-HND 2010-Jan-23, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-23, SIN-HND 2010-Jan-23, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-23, SIN-HND 2010-Jan-23, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-23, SIN-HND 2010-Jan-23, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-24
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-24, SIN-HND, 2010-Jan-24, 09:20:00, 2010-Jan-24, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
```

```
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-24, SIN-HND 2010-Jan-24, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-24, SIN-HND 2010-Jan-24, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-24, SIN-HND 2010-Jan-24, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-24, SIN-HND 2010-Jan-24, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-24, SIN-HND 2010-Jan-24, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-25
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-25, SIN-HND, 2010-Jan-25, 09:20:00, 2010-Jan-25, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-25, SIN-HND 2010-Jan-25, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-25, SIN-HND 2010-Jan-25, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-25, SIN-HND 2010-Jan-25, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-25, SIN-HND 2010-Jan-25, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-25, SIN-HND 2010-Jan-25, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
```

```
****************************************
FlightDate: SQ12, 2010-Jan-26
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-26, SIN-HND, 2010-Jan-26, 09:20:00, 2010-Jan-26, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-26, SIN-HND 2010-Jan-26, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-26, SIN-HND 2010-Jan-26, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-26, SIN-HND 2010-Jan-26, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-26, SIN-HND 2010-Jan-26, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-26, SIN-HND 2010-Jan-26, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-27
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-27, SIN-HND, 2010-Jan-27, 09:20:00, 2010-Jan-27, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-27, SIN-HND 2010-Jan-27, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
```

```
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-27, SIN-HND 2010-Jan-27, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-27, SIN-HND 2010-Jan-27, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-27, SIN-HND 2010-Jan-27, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-27, SIN-HND 2010-Jan-27, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-28
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-28, SIN-HND, 2010-Jan-28, 09:20:00, 2010-Jan-28, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-28, SIN-HND 2010-Jan-28, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-28, SIN-HND 2010-Jan-28, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-28, SIN-HND 2010-Jan-28, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-28, SIN-HND 2010-Jan-28, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-28, SIN-HND 2010-Jan-28, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-29
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-29, SIN-HND, 2010-Jan-29, 09:20:00, 2010-Jan-29, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
```

```
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-29, SIN-HND 2010-Jan-29, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-29, SIN-HND 2010-Jan-29, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-29, SIN-HND 2010-Jan-29, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-29, SIN-HND 2010-Jan-29, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-29, SIN-HND 2010-Jan-29, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-30
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-30, SIN-HND, 2010-Jan-30, 09:20:00, 2010-Jan-30, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-30, SIN-HND 2010-Jan-30, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-30, SIN-HND 2010-Jan-30, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-30, SIN-HND 2010-Jan-30, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-30, SIN-HND 2010-Jan-30, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
```

```
       0, 0, 0, 0,
SQ12 2010-Jan-30, SIN-HND 2010-Jan-30, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Jan-31
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Jan-31, SIN-HND, 2010-Jan-31, 09:20:00, 2010-Jan-31, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Jan-31, SIN-HND 2010-Jan-31, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Jan-31, SIN-HND 2010-Jan-31, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Jan-31, SIN-HND 2010-Jan-31, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Jan-31, SIN-HND 2010-Jan-31, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Jan-31, SIN-HND 2010-Jan-31, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-01
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-01, SIN-HND, 2010-Feb-01, 09:20:00, 2010-Feb-01, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-01, SIN-HND 2010-Feb-01, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
```

```
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-01, SIN-HND 2010-Feb-01, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-01, SIN-HND 2010-Feb-01, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-01, SIN-HND 2010-Feb-01, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-01, SIN-HND 2010-Feb-01, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-02
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-02, SIN-HND, 2010-Feb-02, 09:20:00, 2010-Feb-02, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-02, SIN-HND 2010-Feb-02, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-02, SIN-HND 2010-Feb-02, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-02, SIN-HND 2010-Feb-02, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-02, SIN-HND 2010-Feb-02, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-02, SIN-HND 2010-Feb-02, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-03
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
```

```
      apsed, Distance, Capacity,
SQ12 2010-Feb-03, SIN-HND, 2010-Feb-03, 09:20:00, 2010-Feb-03, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-03, SIN-HND 2010-Feb-03, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-03, SIN-HND 2010-Feb-03, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-03, SIN-HND 2010-Feb-03, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-03, SIN-HND 2010-Feb-03, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-03, SIN-HND 2010-Feb-03, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-04
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-04, SIN-HND, 2010-Feb-04, 09:20:00, 2010-Feb-04, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-04, SIN-HND 2010-Feb-04, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-04, SIN-HND 2010-Feb-04, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-04, SIN-HND 2010-Feb-04, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
```

```
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-04, SIN-HND 2010-Feb-04, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-04, SIN-HND 2010-Feb-04, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-05
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-05, SIN-HND, 2010-Feb-05, 09:20:00, 2010-Feb-05, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-05, SIN-HND 2010-Feb-05, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-05, SIN-HND 2010-Feb-05, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-05, SIN-HND 2010-Feb-05, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-05, SIN-HND 2010-Feb-05, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-05, SIN-HND 2010-Feb-05, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-06
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-06, SIN-HND, 2010-Feb-06, 09:20:00, 2010-Feb-06, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-06, SIN-HND 2010-Feb-06, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
```

```
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-06, SIN-HND 2010-Feb-06, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-06, SIN-HND 2010-Feb-06, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-06, SIN-HND 2010-Feb-06, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-06, SIN-HND 2010-Feb-06, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-07
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-07, SIN-HND, 2010-Feb-07, 09:20:00, 2010-Feb-07, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-07, SIN-HND 2010-Feb-07, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-07, SIN-HND 2010-Feb-07, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-07, SIN-HND 2010-Feb-07, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-07, SIN-HND 2010-Feb-07, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-07, SIN-HND 2010-Feb-07, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-08
****************************************
```

```
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-08, SIN-HND, 2010-Feb-08, 09:20:00, 2010-Feb-08, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-08, SIN-HND 2010-Feb-08, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-08, SIN-HND 2010-Feb-08, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-08, SIN-HND 2010-Feb-08, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-08, SIN-HND 2010-Feb-08, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-08, SIN-HND 2010-Feb-08, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-09
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-09, SIN-HND, 2010-Feb-09, 09:20:00, 2010-Feb-09, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-09, SIN-HND 2010-Feb-09, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-09, SIN-HND 2010-Feb-09, Y, 1, 0, 0, 0, 0, 200, 0,
```

```
SQ12 2010-Feb-09, SIN-HND 2010-Feb-09, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-09, SIN-HND 2010-Feb-09, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-09, SIN-HND 2010-Feb-09, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-10
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-10, SIN-HND, 2010-Feb-10, 09:20:00, 2010-Feb-10, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-10, SIN-HND 2010-Feb-10, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-10, SIN-HND 2010-Feb-10, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-10, SIN-HND 2010-Feb-10, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-10, SIN-HND 2010-Feb-10, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-10, SIN-HND 2010-Feb-10, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-11
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-11, SIN-HND, 2010-Feb-11, 09:20:00, 2010-Feb-11, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
```

```
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-11, SIN-HND 2010-Feb-11, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-11, SIN-HND 2010-Feb-11, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-11, SIN-HND 2010-Feb-11, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-11, SIN-HND 2010-Feb-11, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-11, SIN-HND 2010-Feb-11, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-12
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-12, SIN-HND, 2010-Feb-12, 09:20:00, 2010-Feb-12, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-12, SIN-HND 2010-Feb-12, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-12, SIN-HND 2010-Feb-12, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-12, SIN-HND 2010-Feb-12, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-12, SIN-HND 2010-Feb-12, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-12, SIN-HND 2010-Feb-12, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
```

```
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-13
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-13, SIN-HND, 2010-Feb-13, 09:20:00, 2010-Feb-13, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-13, SIN-HND 2010-Feb-13, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-13, SIN-HND 2010-Feb-13, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-13, SIN-HND 2010-Feb-13, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-13, SIN-HND 2010-Feb-13, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-13, SIN-HND 2010-Feb-13, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-14
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-14, SIN-HND, 2010-Feb-14, 09:20:00, 2010-Feb-14, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-14, SIN-HND 2010-Feb-14, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
```

```
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-14, SIN-HND 2010-Feb-14, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-14, SIN-HND 2010-Feb-14, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-14, SIN-HND 2010-Feb-14, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-14, SIN-HND 2010-Feb-14, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-15
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-15, SIN-HND, 2010-Feb-15, 09:20:00, 2010-Feb-15, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-15, SIN-HND 2010-Feb-15, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-15, SIN-HND 2010-Feb-15, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-15, SIN-HND 2010-Feb-15, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-15, SIN-HND 2010-Feb-15, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-15, SIN-HND 2010-Feb-15, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-16
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-16, SIN-HND, 2010-Feb-16, 09:20:00, 2010-Feb-16, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
```

```
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-16, SIN-HND 2010-Feb-16, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-16, SIN-HND 2010-Feb-16, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-16, SIN-HND 2010-Feb-16, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-16, SIN-HND 2010-Feb-16, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-16, SIN-HND 2010-Feb-16, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-17
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-17, SIN-HND, 2010-Feb-17, 09:20:00, 2010-Feb-17, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-17, SIN-HND 2010-Feb-17, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-17, SIN-HND 2010-Feb-17, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-17, SIN-HND 2010-Feb-17, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
```

```
SQ12 2010-Feb-17, SIN-HND 2010-Feb-17, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-17, SIN-HND 2010-Feb-17, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-18
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-18, SIN-HND, 2010-Feb-18, 09:20:00, 2010-Feb-18, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-18, SIN-HND 2010-Feb-18, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-18, SIN-HND 2010-Feb-18, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-18, SIN-HND 2010-Feb-18, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-18, SIN-HND 2010-Feb-18, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-18, SIN-HND 2010-Feb-18, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-19
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-19, SIN-HND, 2010-Feb-19, 09:20:00, 2010-Feb-19, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-19, SIN-HND 2010-Feb-19, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
```

```
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-19, SIN-HND 2010-Feb-19, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-19, SIN-HND 2010-Feb-19, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-19, SIN-HND 2010-Feb-19, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-19, SIN-HND 2010-Feb-19, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-20
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-20, SIN-HND, 2010-Feb-20, 09:20:00, 2010-Feb-20, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-20, SIN-HND 2010-Feb-20, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-20, SIN-HND 2010-Feb-20, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-20, SIN-HND 2010-Feb-20, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-20, SIN-HND 2010-Feb-20, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-20, SIN-HND 2010-Feb-20, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-21
****************************************
****************************************
Leg-Dates:
----------
```

```
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-21, SIN-HND, 2010-Feb-21, 09:20:00, 2010-Feb-21, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-21, SIN-HND 2010-Feb-21, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-21, SIN-HND 2010-Feb-21, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-21, SIN-HND 2010-Feb-21, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-21, SIN-HND 2010-Feb-21, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-21, SIN-HND 2010-Feb-21, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-22
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-22, SIN-HND, 2010-Feb-22, 09:20:00, 2010-Feb-22, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-22, SIN-HND 2010-Feb-22, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-22, SIN-HND 2010-Feb-22, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-22, SIN-HND 2010-Feb-22, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
```

```
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-22, SIN-HND 2010-Feb-22, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-22, SIN-HND 2010-Feb-22, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-23
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-23, SIN-HND, 2010-Feb-23, 09:20:00, 2010-Feb-23, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-23, SIN-HND 2010-Feb-23, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-23, SIN-HND 2010-Feb-23, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-23, SIN-HND 2010-Feb-23, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-23, SIN-HND 2010-Feb-23, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-23, SIN-HND 2010-Feb-23, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-24
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-24, SIN-HND, 2010-Feb-24, 09:20:00, 2010-Feb-24, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-24, SIN-HND 2010-Feb-24, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
```

```
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-24, SIN-HND 2010-Feb-24, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-24, SIN-HND 2010-Feb-24, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-24, SIN-HND 2010-Feb-24, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-24, SIN-HND 2010-Feb-24, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-25
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-25, SIN-HND, 2010-Feb-25, 09:20:00, 2010-Feb-25, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-25, SIN-HND 2010-Feb-25, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-25, SIN-HND 2010-Feb-25, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-25, SIN-HND 2010-Feb-25, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-25, SIN-HND 2010-Feb-25, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-25, SIN-HND 2010-Feb-25, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-26
```

```
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-26, SIN-HND, 2010-Feb-26, 09:20:00, 2010-Feb-26, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-26, SIN-HND 2010-Feb-26, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-26, SIN-HND 2010-Feb-26, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-26, SIN-HND 2010-Feb-26, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-26, SIN-HND 2010-Feb-26, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-26, SIN-HND 2010-Feb-26, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-27
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-27, SIN-HND, 2010-Feb-27, 09:20:00, 2010-Feb-27, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-27, SIN-HND 2010-Feb-27, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
```

```
SQ12 2010-Feb-27, SIN-HND 2010-Feb-27, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-27, SIN-HND 2010-Feb-27, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-27, SIN-HND 2010-Feb-27, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-27, SIN-HND 2010-Feb-27, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
****************************************
FlightDate: SQ12, 2010-Feb-28
****************************************
****************************************
Leg-Dates:
----------
Flight, Leg, BoardDate, BoardTime, OffDate, OffTime, Date Offset, Time Offset, El
      apsed, Distance, Capacity,
SQ12 2010-Feb-28, SIN-HND, 2010-Feb-28, 09:20:00, 2010-Feb-28, 12:00:00, 07:40:00
      , 0, -05:00:00, 6300, 0,
****************************************
****************************************
LegCabins:
----------
Flight, Leg, Cabin, OffedCAP, PhyCAP, RgdADJ, AU, UPR, SS, Staff, WL, Group, Comm
      Space, AvPool, Avl, NAV, GAV, ACP, ETB, BidPrice,
SQ12 2010-Feb-28, SIN-HND 2010-Feb-28, Y, 200, 200, 0, 0, 0, 0, 0, 0, 0, 0, 200,
      9, 0, 0, 0, 0, 0,
****************************************
****************************************
Buckets:
--------
Flight, Leg, Cabin, Yield, AU/SI, SS, AV,
****************************************
****************************************
SegmentCabins:
--------------
Flight, Segment, Cabin, FF, Bkgs, MIN, UPR, CommSpace, AvPool, BP,
SQ12 2010-Feb-28, SIN-HND 2010-Feb-28, Y, 1, 0, 0, 0, 0, 200, 0,
SQ12 2010-Feb-28, SIN-HND 2010-Feb-28, Y, 2, 0, 0, 0, 0, 200, 0,
****************************************
****************************************
Subclasses:
-----------
Flight, Segment, Cabin, FF, Subclass, MIN/AU (Prot), Nego, NS%, OB%, Bkgs, GrpBks
       (pdg), StfBkgs, WLBkgs, ETB, ClassAvl, RevAvl, SegAvl,
SQ12 2010-Feb-28, SIN-HND 2010-Feb-28, Y, 1, Y, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
SQ12 2010-Feb-28, SIN-HND 2010-Feb-28, Y, 2, M, 200 (0), 0, 0, 0, 0, 0 (0), 0, 0,
       0, 0, 0, 0,
****************************************
```

12.6 Exploring the Predefined BOM Tree

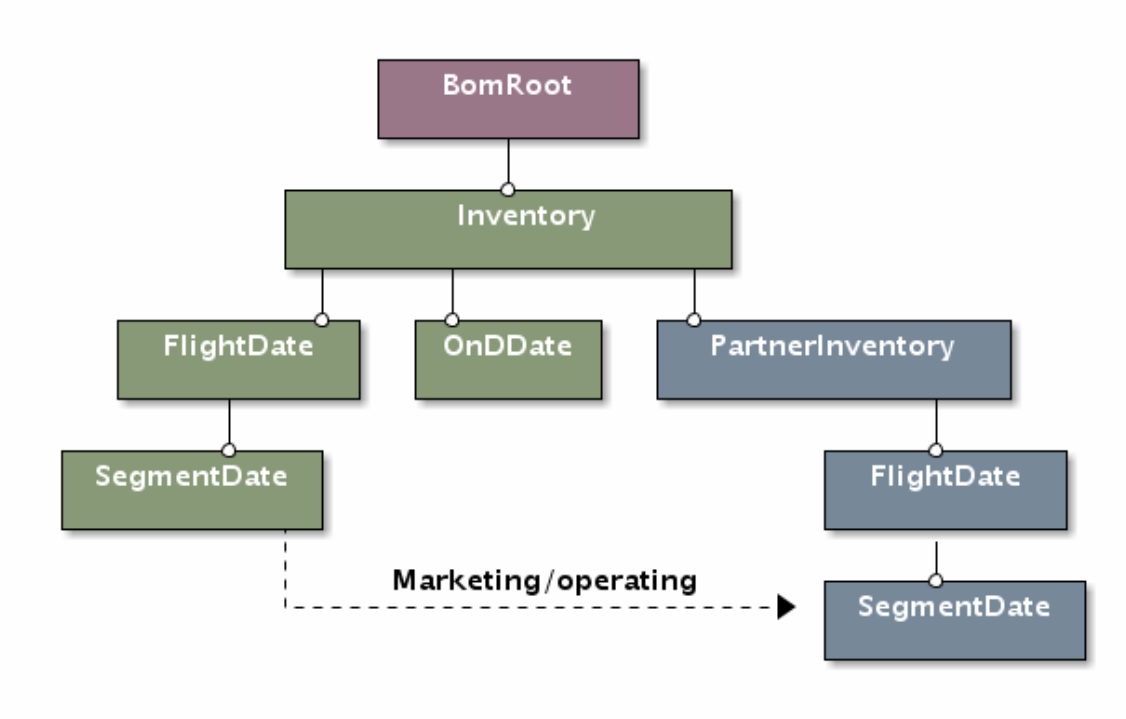


Figure 2: SimCRS BOM tree

`SimCRS` predefines a BOM (Business Object Model) tree specific to the airline IT arena.

12.6.1 Airline Network BOM Tree

- `SIMCRS::ReachableUniverse`
- `SIMCRS::OriginDestinationSet`
- `SIMCRS::SegmentPathPeriod`

12.6.2 Airline Schedule BOM Tree

- `stdair::Inventory`
- `stdair::FlightPeriod`
- `stdair::SegmentPeriod`
- `stdair::OnDPeriod`

12.7 Extending the BOM Tree

12.8 The travel solution calculation procedure

The project SimCRS aims at calculating a list of `travel solutions` for every incoming `booking request`.

13 Supported Systems

13.1 Table of Contents

13.2 Introduction

This page is intended to provide a list of SimCRS supported systems, i.e. the systems on which configuration, installation and testing process of the SimCRS library has been sucessful. Results are grouped based on minor release number. Therefore, only the latest tests for bug-fix releases are included. Besides, the information on this page is divided into sections dependent on the operating system.

Where necessary, some extra information is given for each tested configuration, e.g. external libraries installed, configuration commands used, etc.

If you manage to compile, install and test the SimCRS library on a system not mentioned below, please let us know, so we could update this database.

14 SimCRS Supported Systems (Previous Releases)

14.1 SimCRS 3.9.1

14.2 SimCRS 3.9.0

14.3 SimCRS 3.8.1

15 Tutorials

15.1 Table of Contents

15.2 Preparing the AirSched Project for Development

The source code for these examples can be found in the `batches` and `test/airsched` directories. They are compiled along with the rest of the `AirSched` project. See the Users Guide for more details on how to build the `AirSched` project.

15.3 Your first networkBuilde

15.3.1 Summary of the different steps

All the steps below can be found in the same order in the batch `AirSched.cpp` program.

First, we instanciate the AIRSCHED_Service object:

Then, we construct a default sample list of travel solutions and a default booking request (as mentionned in ug_procedure_bookingrequest and ug_procedure_travelsolution parts):

For basic use, the default BOM tree can be built using:

The main step is the network building (see The travel solution calculation procedure):

15.3.2 Result of the Batch Program

When the `AirSched.cpp` program is run (with the `-b` option), the log output file should look like:

What is interesting is to compare the travel solution list (here reduced to a single travel solution) displayed before:

and after the network building:

Between the two groups of dashes, we can see that a network option structure has been added by the network builder: the price is 450 EUR for the Y class, the ticket is refundable but there are exchange fees and the customer must stay over on saturday night.

Let's return to our default BOM tree display: the only network rule stored was a match for the travel solution into consideration (same origin airport, same destination airport, flight date included in the network rule date range, same airline "BA", ...).

By looking at the network rule trip type "RT", we can guess we face a round trip network: that means the price given in the default bom tree construction in `stdair::CmdBomManager.hpp` has been divided by 2 because we are considering either an inbound trip or an outbound one.

15.4 Network building with an input file

15.4.1 How to build a network input file?

The objective here is to build a network input file to network builde the default travel solution list built using:

This travel solution list, reduced to a singleton, can be displayed as done before:

We deduce:

- we need a network rule whose origin-destination couple is "LHR, SYD".
- the date range must include the date "2011-06-10".
- the time range must include the time "21:45".
- the airline operating is "BA", so it must be the airline pricing.

We can deduce a part of our network rule file :

We have no information about stay duration and advance purchase (such information are contained into the booking request): so let us put "0" to embrace all the requests possible.

No information for the point-of-sale and the channel too: let us consider all the channels ("IN", "DN", "IF" and DF") and all the points of sale (the origin "LHR", the destination "SYD" and the rest-of-the-world "ROW) existing. To access this information, we could look into the default booking request.

The input file is now:

Let us say we have just the Economy cabin "Y" and Bristish Airways prices ticket for class "Y".

No information about the trip type, so we duplicate all the network rules for both type: one-way "OW" and round-trip "RT" (to access this information, we could look to the default booking request).

The network options are all set to a default value "T" (meaning true) and the network values are chosen to be all distinct.

We obtain:

15.4.2 Building the BOM tree with an input file

The steps are the same as before Summary of the different steps except the bom tree must be built using the network input file :

15.4.3 Result of the Batch Program

When the `AirSched.cpp` program is run with the `-f` option linking with the file built just above:

```
~/AirSched -f ~/<YourFileName>.csv
```

the last lines of the log output should look like:

```
[D]~/AirSchedgit/AirSched/batches/AirSched.cpp:223: Travel solutions:
    [0] [0] BA, 9, 2011-06-10, LHR, SYD, 21:45 --- Y, 145, 1 1 1 ---
```

We have just one network option added to the travel solution. We can deduce from the price value 145 that the network builder used the network rule number 15 to price the travel solution. We have an inbound or outbound trip of a round trip: the total price 290 has been divided by 2.

16 Command-Line Test to Demonstrate How To Test the SimCRS Project

```
 */
// ////////////////////////////////////////////////////////////////////
// Import section
// ////////////////////////////////////////////////////////////////////
// STL
#include <sstream>
```

```
#include <fstream>
#include <string>
#include <cmath>
// Boost Unit Test Framework (UTF)
#define BOOST_TEST_DYN_LINK
#define BOOST_TEST_MAIN
#define BOOST_TEST_MODULE CRSTestSuite
#include <boost/test/unit_test.hpp>
// StdAir
#include <stdair/basic/BasLogParams.hpp>
#include <stdair/basic/BasDBParams.hpp>
#include <stdair/basic/BasFileMgr.hpp>
#include <stdair/bom/TravelSolutionStruct.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/service/Logger.hpp>
// SimFQT
#include <simfqt/SIMFQT_Types.hpp>
// SimCRS
#include <simcrs/SIMCRS_Service.hpp>
#include <simcrs/config/simcrs-paths.hpp>

namespace boost_utf = boost::unit_test;

// (Boost) Unit Test XML Report
std::ofstream utfReportStream ("CRSTestSuite_utfresults.xml");

struct UnitTestConfig {
  UnitTestConfig() {
    boost_utf::unit_test_log.set_stream (utfReportStream);
    boost_utf::unit_test_log.set_format (boost_utf::XML);
    boost_utf::unit_test_log.set_threshold_level (boost_utf::log_test_units);
    //boost_utf::unit_test_log.set_threshold_level (boost_utf::log_successful_tes
      ts);
  }

  ~UnitTestConfig() {
  }
};

// ////////////////////////////////////////////////////////////////////
const unsigned int testSimCRSHelper (const unsigned short iTestFlag,
                                     const stdair::Filename_T& iScheduleInputFile
      name,
                                     const stdair::Filename_T& iOnDInputFilename,

                                     const stdair::Filename_T& iFRAT5InputFilenam
      e,
                                     const stdair::Filename_T& iFFDisutilityInput
      Filename,
                                     const stdair::Filename_T& iYieldInputFilenam
      e,
                                     const stdair::Filename_T& iFareInputFilename
      ,
                                     const bool isBuiltin,
                                     const unsigned int iExpectedNbOfTravelSoluti
      ons,
                                     const unsigned int iExpectedPrice) {

  // CRS code
  const SIMCRS::CRSCode_T lCRSCode ("1P");

  // Output log File
  std::ostringstream oStr;
  oStr << "CRSTestSuite_" << iTestFlag << ".log";
  const stdair::Filename_T lLogFilename (oStr.str());

  // Set the log parameters
```

```
std::ofstream logOutputFile;
// Open and clean the log outputfile
logOutputFile.open (lLogFilename.c_str());
logOutputFile.clear();

// Initialise the list of classes/buckets
const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
SIMCRS::SIMCRS_Service simcrsService (lLogParams, lCRSCode);

stdair::Date_T lPreferredDepartureDate;;
stdair::Date_T lRequestDate;
stdair::TripType_T lTripType;

// Check wether or not a (CSV) input file should be read
if (isBuiltin == true) {

  // Build the default sample BOM tree
  simcrsService.buildSampleBom();

  lPreferredDepartureDate = boost::gregorian::from_string ("2010/02/08");
  lRequestDate = boost::gregorian::from_string ("2010/01/21");
  lTripType = "OW";

} else {

  // Build the BOM tree from parsing input files
  stdair::ScheduleFilePath lScheduleFilePath (iScheduleInputFilename);
  stdair::ODFilePath lODFilePath (iOnDInputFilename);
  stdair::FRAT5FilePath lFRAT5FilePath (iFRAT5InputFilename);
  stdair::FFDisutilityFilePath lFFDisutilityFilePath (iFFDisutilityInputFilenam
    e);
  const SIMFQT::FareFilePath lFareFilePath (iFareInputFilename);
  const AIRRAC::YieldFilePath lYieldFilePath (iYieldInputFilename);
  simcrsService.parseAndLoad (lScheduleFilePath, lODFilePath,
                              lFRAT5FilePath, lFFDisutilityFilePath,
                              lYieldFilePath, lFareFilePath);

  lPreferredDepartureDate = boost::gregorian::from_string ("2011/01/31");
  lRequestDate = boost::gregorian::from_string ("2011/01/22");
  lTripType = "RI";
}

// Create an empty booking request structure
const stdair::AirportCode_T lOrigin ("SIN");
const stdair::AirportCode_T lDestination ("BKK");
const stdair::AirportCode_T lPOS ("SIN");
const stdair::Duration_T lRequestTime (boost::posix_time::hours(10));
const stdair::DateTime_T lRequestDateTime (lRequestDate, lRequestTime);
const stdair::CabinCode_T lPreferredCabin ("Eco");
const stdair::PartySize_T lPartySize (3);
const stdair::ChannelLabel_T lChannel ("IN");
const stdair::DayDuration_T lStayDuration (7);
const stdair::FrequentFlyer_T lFrequentFlyerType ("M");
const stdair::Duration_T lPreferredDepartureTime (boost::posix_time::hours(10))
    ;
const stdair::WTP_T lWTP (1000.0);
const stdair::PriceValue_T lValueOfTime (100.0);
const stdair::ChangeFees_T lChangeFees (true);
const stdair::Disutility_T lChangeFeeDisutility (50);
const stdair::NonRefundable_T lNonRefundable (true);
const stdair::Disutility_T lNonRefundableDisutility (50);
const stdair::BookingRequestStruct lBookingRequest (lOrigin, lDestination,
                                                    lPOS,
                                                    lPreferredDepartureDate,
                                                    lRequestDateTime,
                                                    lPreferredCabin,
                                                    lPartySize, lChannel,
```

```
                                                    lTripType, lStayDuration,
                                                    lFrequentFlyerType,
                                                    lPreferredDepartureTime,
                                                    lWTP, lValueOfTime,
                                                    lChangeFees,
                                                    lChangeFeeDisutility,
                                                    lNonRefundable,
                                                    lNonRefundableDisutility);
  stdair::TravelSolutionList_T lTravelSolutionList =
    simcrsService.calculateSegmentPathList (lBookingRequest);

  // Price the travel solution
  simcrsService.fareQuote (lBookingRequest, lTravelSolutionList);

  //
  const unsigned int lNbOfTravelSolutions = lTravelSolutionList.size();

  // DEBUG
  std::ostringstream oMessageKeptTS;
  oMessageKeptTS << "The number of travel solutions for the booking request '"
                 << lBookingRequest.describe() << "' is actually "
                 << lNbOfTravelSolutions << ". That number is expected to be "
                 << iExpectedNbOfTravelSolutions << ".";
  STDAIR_LOG_DEBUG (oMessageKeptTS.str());

  BOOST_CHECK_EQUAL (lNbOfTravelSolutions, iExpectedNbOfTravelSolutions);

  BOOST_CHECK_MESSAGE (lNbOfTravelSolutions == iExpectedNbOfTravelSolutions,
                       oMessageKeptTS.str());

  stdair::TravelSolutionStruct& lTravelSolution = lTravelSolutionList.front();

  const stdair::FareOptionList_T& lFareOptionList =
    lTravelSolution.getFareOptionList();

  stdair::FareOptionStruct lFareOption = lFareOptionList.front();
  lTravelSolution.setChosenFareOption (lFareOption);

  // DEBUG
  std::ostringstream oMessageKeptFare;
  oMessageKeptFare
    << "The price given by the fare quoter for the booking request: '"
    << lBookingRequest.describe() << "' and travel solution: '"
    << lTravelSolution.describe() << "' is actually " << lFareOption.getFare()
    << " Euros. It is expected to be " << iExpectedPrice << " Euros.";
  STDAIR_LOG_DEBUG (oMessageKeptFare.str());

  BOOST_CHECK_EQUAL (std::floor (lFareOption.getFare() + 0.5), iExpectedPrice);

  BOOST_CHECK_MESSAGE (std::floor (lFareOption.getFare() + 0.5)
                       == iExpectedPrice, oMessageKeptFare.str());

  // DEBUG
  STDAIR_LOG_DEBUG ("A booking will now (attempted to) be made on the "
                    "travel solution '" << lTravelSolution.describe()
                    << "', for a party size of " << lPartySize << ".");

  const bool isSellSuccessful =
    simcrsService.sell (lTravelSolution, lPartySize);

  // Close the log file
  logOutputFile.close();

  return isSellSuccessful;

}
```

```
// /////////////// Main: Unit Test Suite //////////////

// Set the UTF configuration (re-direct the output to a specific file)
BOOST_GLOBAL_FIXTURE (UnitTestConfig);

// Start the test suite
BOOST_AUTO_TEST_SUITE (master_test_suite)


BOOST_AUTO_TEST_CASE (simcrs_simple_simulation_test) {

  // Schedule input filename
  const stdair::Filename_T lScheduleInputFilename (STDAIR_SAMPLE_DIR
                                                   "/rds01/schedule.csv");

  // O&D input filename
  const stdair::Filename_T lOnDInputFilename (STDAIR_SAMPLE_DIR "/ond01.csv");

  // FRAT5 curve input file name
  const stdair::Filename_T lFRAT5InputFilename (STDAIR_SAMPLE_DIR
                                                "/frat5.csv");

  // Fare family disutility curve input file name
  const stdair::Filename_T lFFDisutilityInputFilename (STDAIR_SAMPLE_DIR
                                                       "/ffDisutility.csv");

  // Yield input filename
  const stdair::Filename_T lYieldInputFilename (STDAIR_SAMPLE_DIR
                                                "/rds01/yield.csv");

  // Fare input filename
  const stdair::Filename_T lFareInputFilename (STDAIR_SAMPLE_DIR
                                               "/rds01/fare.csv");

  // State whether the BOM tree should be built-in or parsed from input files
  const bool isBuiltin = false;

  const unsigned int lExpectedPrice = 400;
  const unsigned int lExpectedNbOfTravelSolutions = 1;

  bool isSellSuccessful = false;

  BOOST_CHECK_NO_THROW (isSellSuccessful =
                        testSimCRSHelper (0,
                                          lScheduleInputFilename,
                                          lOnDInputFilename,
                                          lFRAT5InputFilename,
                                          lFFDisutilityInputFilename,
                                          lYieldInputFilename,
                                          lFareInputFilename,
                                          isBuiltin,
                                          lExpectedNbOfTravelSolutions,
                                          lExpectedPrice));

  // DEBUG
  std::ostringstream oMessageSell;
  const std::string isSellSuccessfulStr = (isSellSuccessful == true)?"Yes":"No";
  oMessageSell << "Was the sell successful? Answer: " << isSellSuccessfulStr;
  STDAIR_LOG_DEBUG (oMessageSell.str());

  BOOST_CHECK_EQUAL (isSellSuccessful, true);

  BOOST_CHECK_MESSAGE (isSellSuccessful == true, oMessageSell.str());


}
```

```
BOOST_AUTO_TEST_CASE (simcrs_simple_default_bom_simulation_test) {

  // State whether the BOM tree should be built-in or parsed from input files
  const bool isBuiltin = true;

  const unsigned int lExpectedPrice = 900;
  const unsigned int lExpectedNbOfTravelSolutions = 1;

  bool isSellSuccessful = false;

  BOOST_CHECK_NO_THROW (isSellSuccessful =
                        testSimCRSHelper (1,
                                          " ", " ", " ", " ", " ", " ",
                                          isBuiltin,
                                          lExpectedNbOfTravelSolutions,
                                          lExpectedPrice));

  // DEBUG
  std::ostringstream oMessageSell;
  const std::string isSellSuccessfulStr = (isSellSuccessful == true)?"Yes":"No";
  oMessageSell << "Was the sell successful? Answer: " << isSellSuccessfulStr;
  STDAIR_LOG_DEBUG (oMessageSell.str());

  BOOST_CHECK_EQUAL (isSellSuccessful, true);

  BOOST_CHECK_MESSAGE (isSellSuccessful == true, oMessageSell.str());


}

// End the test suite
BOOST_AUTO_TEST_SUITE_END()

/*!
```

17 Directory Hierarchy

17.1 Directories

This directory hierarchy is sorted roughly, but not completely, alphabetically:

18 Namespace Index

18.1 Namespace List

Here is a list of all namespaces with brief descriptions:

19 Class Index

19.1 Class Hierarchy

This inheritance list is sorted roughly, but not completely, alphabetically:

std::basic_fstream< char >
std::basic_fstream< wchar_t >
std::basic_ifstream< char >
std::basic_ifstream< wchar_t >
std::basic_ios< char >
std::basic_ios< wchar_t >
std::basic_iostream< char >
std::basic_iostream< wchar_t >
std::basic_istream< char >
std::basic_istream< wchar_t >
std::basic_istringstream< char >
std::basic_istringstream< wchar_t >
std::basic_ofstream< char >
std::basic_ofstream< wchar_t >
std::basic_ostream< char >
std::basic_ostream< wchar_t >
std::basic_ostringstream< char >
std::basic_ostringstream< wchar_t >
std::basic_string< char >
std::basic_string< wchar_t >
std::basic_stringstream< char >
std::basic_stringstream< wchar_t >

20 Class Index

20.1 Class List

Here are the classes, structs, unions and interfaces with brief descriptions:

21 File Index

21.1 File List

Here is a list of all files with brief descriptions:

22 Directory Documentation

22.1 simcrs/basic/ Directory Reference

Files

- file BasConst.cpp
- file BasConst_General.hpp

- file BasConst_SIMCRS_Service.hpp

22.2 simcrs/batches/ Directory Reference

Files

- file simcrs.cpp

22.3 simcrs/bom/ Directory Reference

Files

- file BomAbstract.cpp
- file BomAbstract.hpp

22.4 simcrs/command/ Directory Reference

Files

- file DistributionManager.cpp
- file DistributionManager.hpp

22.5 simcrs/config/ Directory Reference

Files

- file simcrs-paths.hpp.in

22.6 simcrs/factory/ Directory Reference

Files

- file FacBomAbstract.cpp
- file FacBomAbstract.hpp
- file FacServiceAbstract.cpp
- file FacServiceAbstract.hpp
- file FacSimcrsServiceContext.cpp
- file FacSimcrsServiceContext.hpp
- file FacSupervisor.cpp
- file FacSupervisor.hpp

22.7 simcrs/service/ Directory Reference

Files

- file ServiceAbstract.cpp
- file ServiceAbstract.hpp
- file SIMCRS_Service.cpp
- file SIMCRS_ServiceContext.cpp
- file SIMCRS_ServiceContext.hpp

22.8 test/simcrs/ Directory Reference

Files

- file CRSTestSuite.cpp

22.9 simcrs/ Directory Reference

Directories

- directory basic
- directory batches
- directory bom
- directory command
- directory config
- directory factory
- directory service

Files

- file SIMCRS_Service.hpp
- file SIMCRS_Types.hpp

22.10 test/ Directory Reference

Directories

- directory simcrs

23 Namespace Documentation

23.1 AIRINV Namespace Reference

23.2 SIMCRS Namespace Reference

Classes

- class BomAbstract
- class DistributionManager

  Command wrapping the travel distribution (CRS/GDS) process.

- class FacBomAbstract
- class FacServiceAbstract
- class FacSimcrsServiceContext
- class FacSupervisor
- class ServiceAbstract
- class SIMCRS_ServiceContext

  Class holding the context of the Simcrs services.

- class SIMCRS_Service
- class BookingException
- class AvailabilityRetrievalException

Typedefs

- typedef std::string CRSCode_T
- typedef boost::shared_ptr< SIMCRS_Service > SIMCRS_ServicePtr_T

Variables

- const std::string DEFAULT_CRS_CODE = "1S"

23.2.1 Typedef Documentation

23.2.1.1 typedef std::string SIMCRS::CRSCode_T

CRS code (identifier of the CRS; not actually used for now).

Definition at line 39 of file SIMCRS_Types.hpp.

23.2.1.2 typedef boost::shared_ptr<SIMCRS_Service> SIMCRS::SIMCRS_ServicePtr_T

(Smart) Pointer on the SimCRS service handler.

Definition at line 44 of file SIMCRS_Types.hpp.

23.2.2 Variable Documentation

23.2.2.1 const std::string SIMCRS::DEFAULT_CRS_CODE = "1S"

Default CRS code for the SIMCRS_Service.

Definition at line 10 of file BasConst.cpp.

23.3 stdair Namespace Reference

Forward declarations.

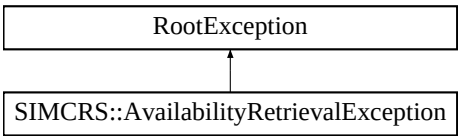
23.3.1 Detailed Description

Forward declarations.

24 Class Documentation

24.1 SIMCRS::AvailabilityRetrievalException Class Reference

`#include <simcrs/SIMCRS_Types.hpp>`Inheritance diagram for SIMCRS::AvailabilityRetrievalException::



24.1.1 Detailed Description

Specific exception related to availability calculation.

Definition at line 31 of file SIMCRS_Types.hpp.

The documentation for this class was generated from the following file:

- simcrs/SIMCRS_Types.hpp

24.2 SIMCRS::BomAbstract Class Reference

`#include <simcrs/bom/BomAbstract.hpp>`

Public Member Functions

- virtual void toStream (std::ostream &ioOut) const =0
- virtual void fromStream (std::istream &ioIn)=0
- virtual std::string toString () const =0
- virtual std::string describeKey () const =0
- virtual std::string describeShortKey () const =0

Protected Member Functions

- BomAbstract ()
- BomAbstract (const BomAbstract &)
- virtual ~BomAbstract ()

Friends

- class FacBomAbstract

24.2.1 Detailed Description

Base class for the Business Object Model (BOM) layer.

Definition at line 14 of file BomAbstract.hpp.

24.2.2 Constructor & Destructor Documentation

24.2.2.1 SIMCRS::BomAbstract::BomAbstract () [inline, protected]

Protected Default Constructor to ensure this class is abtract.

Definition at line 40 of file BomAbstract.hpp.

24.2.2.2 SIMCRS::BomAbstract::BomAbstract (const BomAbstract &) [inline, protected]

Definition at line 41 of file BomAbstract.hpp.

24.2.2.3 virtual SIMCRS::BomAbstract::~BomAbstract () [inline, protected, virtual]

Destructor.

Definition at line 44 of file BomAbstract.hpp.

24.2.3 Member Function Documentation

24.2.3.1 virtual void SIMCRS::BomAbstract::toStream (std::ostream & *ioOut*) const [pure virtual]

Dump a Business Object into an output stream.

Parameters:

ostream& the output stream.

24.2.3.2 virtual void SIMCRS::BomAbstract::fromStream (std::istream & *ioIn*) [pure virtual]

Read a Business Object from an input stream.

Parameters:

istream& the input stream.

Referenced by operator>>().

24.2.3.3 virtual std::string SIMCRS::BomAbstract::toString () const [pure virtual]

Get the serialised version of the Business Object.

24.2.3.4 virtual std::string SIMCRS::BomAbstract::describeKey () const [pure virtual]

Get a string describing the whole key (differentiating two objects at any level).

24.2.3.5 virtual std::string SIMCRS::BomAbstract::describeShortKey () const [pure virtual]

Get a string describing the short key (differentiating two objects at the same level).

24.2.4 Friends And Related Function Documentation

24.2.4.1 friend class FacBomAbstract [friend]

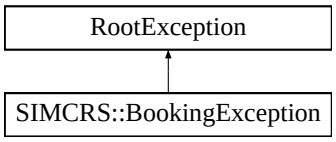
Definition at line 15 of file BomAbstract.hpp.

The documentation for this class was generated from the following file:

- simcrs/bom/BomAbstract.hpp

24.3 SIMCRS::BookingException Class Reference

`#include <simcrs/SIMCRS_Types.hpp>`Inheritance diagram for SIMCRS::BookingException::



24.3.1 Detailed Description

Specific exception related to bookings made against the CRS.

Definition at line 25 of file SIMCRS_Types.hpp.

The documentation for this class was generated from the following file:

- simcrs/SIMCRS_Types.hpp

24.4 SIMCRS::DistributionManager Class Reference

Command wrapping the travel distribution (CRS/GDS) process.

`#include <simcrs/command/DistributionManager.hpp>`

Friends

- class SIMCRS_Service

24.4.1 Detailed Description

Command wrapping the travel distribution (CRS/GDS) process.

Definition at line 30 of file DistributionManager.hpp.

24.4.2 Friends And Related Function Documentation

24.4.2.1 friend class SIMCRS_Service [friend]

Definition at line 31 of file DistributionManager.hpp.

The documentation for this class was generated from the following files:

- simcrs/command/DistributionManager.hpp
- simcrs/command/DistributionManager.cpp

24.5 SIMCRS::FacBomAbstract Class Reference

`#include <simcrs/factory/FacBomAbstract.hpp>`

Public Types

- typedef std::vector< BomAbstract * > BomPool_T

Static Public Member Functions

- static std::size_t getID (const BomAbstract *)
- static std::size_t getID (const BomAbstract &)
- static std::string getIDString (const BomAbstract *)
- static std::string getIDString (const BomAbstract &)

Protected Member Functions

- FacBomAbstract ()
- FacBomAbstract (const FacBomAbstract &)
- virtual ~FacBomAbstract ()

Protected Attributes

- BomPool_T _pool

Friends

- class FacSupervisor

24.5.1 Detailed Description

Base class for Factory layer.

Definition at line 17 of file FacBomAbstract.hpp.

24.5.2 Member Typedef Documentation

24.5.2.1 typedef std::vector<BomAbstract*> SIMCRS::FacBomAbstract::BomPool_T

Define the list (pool) of Bom objects.

Definition at line 22 of file FacBomAbstract.hpp.

24.5.3 Constructor & Destructor Documentation

24.5.3.1 SIMCRS::FacBomAbstract::FacBomAbstract () [inline, protected]

Default Constructor.

This constructor is protected to ensure the class is abstract.

Definition at line 41 of file FacBomAbstract.hpp.

24.5.3.2 SIMCRS::FacBomAbstract::FacBomAbstract (const FacBomAbstract &) [inline, protected]

Definition at line 42 of file FacBomAbstract.hpp.

24.5.3.3 SIMCRS::FacBomAbstract::~FacBomAbstract () [protected, virtual]

Destructor.

Definition at line 16 of file FacBomAbstract.cpp.

24.5.4 Member Function Documentation

24.5.4.1 std::size_t SIMCRS::FacBomAbstract::getID (const BomAbstract * *iBomAbstract_ptr*) [static]

Return the ID corresponding to the given object pointer.

Definition at line 35 of file FacBomAbstract.cpp.

Referenced by getID(), and getIDString().

24.5.4.2 std::size_t SIMCRS::FacBomAbstract::getID (const BomAbstract & *iBomAbstract*) [static]

Return the ID corresponding to the given object reference.

Definition at line 43 of file FacBomAbstract.cpp.

References getID().

24.5.4.3 std::string SIMCRS::FacBomAbstract::getIDString (const BomAbstract * *iBomAbstract_ptr*) [static]

Return the ID, as a string, corresponding to the given object pointer.

Definition at line 48 of file FacBomAbstract.cpp.

References getID().

Referenced by getIDString().

24.5.4.4 std::string SIMCRS::FacBomAbstract::getIDString (const BomAbstract & *iBomAbstract*) [static]

Return the ID, as a string, corresponding to the given object reference.

Definition at line 56 of file FacBomAbstract.cpp.

References getIDString().

24.5.5 Friends And Related Function Documentation

24.5.5.1 friend class FacSupervisor [friend]

Definition at line 18 of file FacBomAbstract.hpp.

24.5.6 Member Data Documentation

24.5.6.1 BomPool_T SIMCRS::FacBomAbstract::_pool [protected]

List of instantiated Business Objects

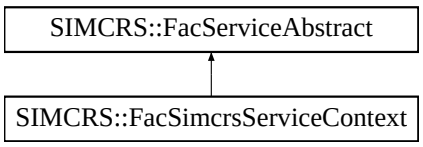
Definition at line 53 of file FacBomAbstract.hpp.

The documentation for this class was generated from the following files:

- simcrs/factory/FacBomAbstract.hpp
- simcrs/factory/FacBomAbstract.cpp

24.6 SIMCRS::FacServiceAbstract Class Reference

`#include <simcrs/factory/FacServiceAbstract.hpp>`Inheritance diagram for SIMCRS::FacServiceAbstract::



Public Types

- typedef std::vector< ServiceAbstract * > ServicePool_T

Public Member Functions

- virtual ~FacServiceAbstract ()
- void clean ()

Protected Member Functions

- FacServiceAbstract ()

Protected Attributes

- ServicePool_T _pool

24.6.1 Detailed Description

Base class for the (Service) Factory layer.

Definition at line 16 of file FacServiceAbstract.hpp.

24.6.2 Member Typedef Documentation

24.6.2.1 typedef std::vector<ServiceAbstract*> SIMCRS::FacServiceAbstract::ServicePool_T

Define the list (pool) of Service objects.

Definition at line 20 of file FacServiceAbstract.hpp.

24.6.3 Constructor & Destructor Documentation

24.6.3.1 SIMCRS::FacServiceAbstract::~FacServiceAbstract () [virtual]

Destructor.

Definition at line 13 of file FacServiceAbstract.cpp.

References clean().

24.6.3.2 SIMCRS::FacServiceAbstract::FacServiceAbstract () [inline, protected]

Default Constructor.

This constructor is protected to ensure the class is abstract.

Definition at line 31 of file FacServiceAbstract.hpp.

24.6.4 Member Function Documentation

24.6.4.1 void SIMCRS::FacServiceAbstract::clean ()

Destroyed all the object instantiated by this factory.

Definition at line 18 of file FacServiceAbstract.cpp.

References _pool.

Referenced by ~FacServiceAbstract().

24.6.5 Member Data Documentation

24.6.5.1 ServicePool_T SIMCRS::FacServiceAbstract::_pool [protected]

List of instantiated Business Objects

Definition at line 34 of file FacServiceAbstract.hpp.

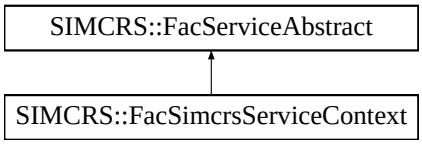
Referenced by clean(), and SIMCRS::FacSimcrsServiceContext::create().

The documentation for this class was generated from the following files:

- simcrs/factory/FacServiceAbstract.hpp
- simcrs/factory/FacServiceAbstract.cpp

24.7 SIMCRS::FacSimcrsServiceContext Class Reference

`#include <simcrs/factory/FacSimcrsServiceContext.hpp>`Inheritance diagram for SIMCRS::FacSimcrsServiceContext::



Public Types

- typedef std::vector< ServiceAbstract * > ServicePool_T

Public Member Functions

- ~FacSimcrsServiceContext ()
- SIMCRS_ServiceContext & create (const std::string &iTravelDatabaseName)
- void clean ()

Static Public Member Functions

- static FacSimcrsServiceContext & instance ()

Protected Member Functions

- FacSimcrsServiceContext ()

Protected Attributes

- ServicePool_T _pool

24.7.1 Detailed Description

Factory for Bucket.

Definition at line 18 of file FacSimcrsServiceContext.hpp.

24.7.2 Member Typedef Documentation

24.7.2.1 typedef std::vector<ServiceAbstract*> SIMCRS::FacServiceAbstract::ServicePool_T [inherited]

Define the list (pool) of Service objects.

Definition at line 20 of file FacServiceAbstract.hpp.

24.7.3 Constructor & Destructor Documentation

24.7.3.1 SIMCRS::FacSimcrsServiceContext::~FacSimcrsServiceContext ()

Destructor.

The Destruction put the _instance to NULL in order to be clean for the next FacSimcrsServiceContext::instance()

Definition at line 16 of file FacSimcrsServiceContext.cpp.

24.7.3.2 SIMCRS::FacSimcrsServiceContext::FacSimcrsServiceContext () [inline, protected]

Default Constructor.

This constructor is protected in order to ensure the singleton pattern.

Definition at line 42 of file FacSimcrsServiceContext.hpp.

Referenced by instance().

24.7.4 Member Function Documentation

24.7.4.1 FacSimcrsServiceContext & SIMCRS::FacSimcrsServiceContext::instance () [static]

Provide the unique instance.

The singleton is instantiated when first used

Returns:

FacSimcrsServiceContext&

Definition at line 21 of file FacSimcrsServiceContext.cpp.

References FacSimcrsServiceContext().

24.7.4.2 SIMCRS_ServiceContext & SIMCRS::FacSimcrsServiceContext::create (const std::string & *iTravelDatabaseName*)

Create a new SIMCRS_ServiceContext object.

This new object is added to the list of instantiated objects.

Returns:

SIMCRS_ServiceContext& The newly created object.

Definition at line 34 of file FacSimcrsServiceContext.cpp.

References SIMCRS::FacServiceAbstract::_pool.

24.7.4.3 void SIMCRS::FacServiceAbstract::clean () [inherited]

Destroyed all the object instantiated by this factory.

Definition at line 18 of file FacServiceAbstract.cpp.

References SIMCRS::FacServiceAbstract::_pool.

Referenced by SIMCRS::FacServiceAbstract::~FacServiceAbstract().

24.7.5 Member Data Documentation

24.7.5.1 ServicePool_T SIMCRS::FacServiceAbstract::_pool [protected, inherited]

List of instantiated Business Objects

Definition at line 34 of file FacServiceAbstract.hpp.

Referenced by SIMCRS::FacServiceAbstract::clean(), and create().

The documentation for this class was generated from the following files:

- simcrs/factory/FacSimcrsServiceContext.hpp
- simcrs/factory/FacSimcrsServiceContext.cpp

24.8 SIMCRS::FacSupervisor Class Reference

`#include <simcrs/factory/FacSupervisor.hpp>`

Public Types

- typedef std::vector< FacBomAbstract * > BomFactoryPool_T
- typedef std::vector< FacServiceAbstract * > ServiceFactoryPool_T

Public Member Functions

- void registerBomFactory (FacBomAbstract ∗)
- void registerServiceFactory (FacServiceAbstract ∗)
- void cleanBomLayer ()
- void cleanServiceLayer ()
- ∼FacSupervisor ()

Static Public Member Functions

- static FacSupervisor & instance ()
- static void cleanFactory ()

Protected Member Functions

- FacSupervisor ()
- FacSupervisor (const FacSupervisor &)

24.8.1 Detailed Description

Singleton class to register and clean all Factories.

Definition at line 17 of file FacSupervisor.hpp.

24.8.2 Member Typedef Documentation

24.8.2.1 typedef std::vector<FacBomAbstract∗> SIMCRS::FacSupervisor::BomFactoryPool_T

Define the pool (list) of factories.

Definition at line 21 of file FacSupervisor.hpp.

24.8.2.2 typedef std::vector<FacServiceAbstract∗> SIMCRS::FacSupervisor::ServiceFactoryPool_T

Definition at line 22 of file FacSupervisor.hpp.

24.8.3 Constructor & Destructor Documentation

24.8.3.1 SIMCRS::FacSupervisor::∼FacSupervisor ()

Destructor

The static instance is deleted (and reset to NULL) by the static cleanFactory() method.

Definition at line 41 of file FacSupervisor.cpp.

References cleanBomLayer(), and cleanServiceLayer().

24.8.3.2 SIMCRS::FacSupervisor::FacSupervisor () [protected]

Default Constructor.

This constructor is protected to ensure the singleton pattern.

Definition at line 16 of file FacSupervisor.cpp.

Referenced by instance().

24.8.3.3 SIMCRS::FacSupervisor::FacSupervisor (const FacSupervisor &) [inline, protected]

Definition at line 66 of file FacSupervisor.hpp.

24.8.4 Member Function Documentation

24.8.4.1 FacSupervisor & SIMCRS::FacSupervisor::instance () [static]

Provides the unique instance.

The singleton is instantiated when first used.

Returns:

FacSupervisor&

Definition at line 20 of file FacSupervisor.cpp.

References FacSupervisor().

24.8.4.2 void SIMCRS::FacSupervisor::registerBomFactory (FacBomAbstract * *ioFacBomAbstract_ptr*)

Register a newly instantiated concrete factory for the Bom layer.

When a concrete Factory is firstly instantiated this factory have to register itself to the FacSupervisor

Parameters:

FacAbstract& the concrete Factory to register.

Definition at line 30 of file FacSupervisor.cpp.

24.8.4.3 void SIMCRS::FacSupervisor::registerServiceFactory (FacServiceAbstract * *ioFacServiceAbstract_ptr*)

Register a newly instantiated concrete factory for the Service layer.

When a concrete Factory is firstly instantiated this factory have to register itself to the FacSupervisor.

Parameters:

FacServiceAbstract& the concrete Factory to register.

Definition at line 36 of file FacSupervisor.cpp.

24.8.4.4 void SIMCRS::FacSupervisor::cleanBomLayer ()

Clean all created object.

Call the clean method of all the instantiated factories for the Bom layer.

Definition at line 47 of file FacSupervisor.cpp.

Referenced by cleanFactory(), and ~FacSupervisor().

24.8.4.5 void SIMCRS::FacSupervisor::cleanServiceLayer ()

Clean all Service created object.

Call the clean method of all the instantiated factories for the Service layer.

Definition at line 61 of file FacSupervisor.cpp.

Referenced by cleanFactory(), and ~FacSupervisor().

24.8.4.6 void SIMCRS::FacSupervisor::cleanFactory () [static]

Clean the static instance.

The singleton is deleted.

Definition at line 75 of file FacSupervisor.cpp.

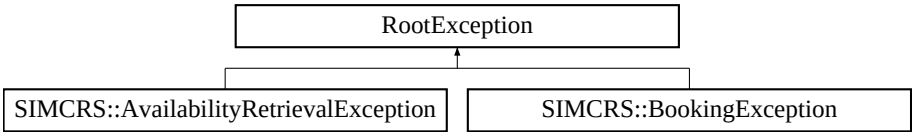
References cleanBomLayer(), and cleanServiceLayer().

The documentation for this class was generated from the following files:

- simcrs/factory/FacSupervisor.hpp
- simcrs/factory/FacSupervisor.cpp

24.9 RootException Class Reference

Inheritance diagram for RootException::

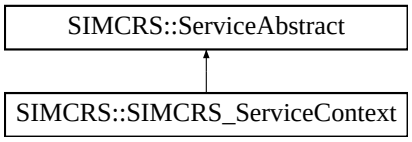


The documentation for this class was generated from the following file:

- simcrs/SIMCRS_Types.hpp

24.10 SIMCRS::ServiceAbstract Class Reference

`#include <simcrs/service/ServiceAbstract.hpp>`Inheritance diagram for SIMCRS::ServiceAbstract::



Public Member Functions

- virtual ~ServiceAbstract ()
- virtual void toStream (std::ostream &ioOut) const
- virtual void fromStream (std::istream &ioIn)

Protected Member Functions

- ServiceAbstract ()

24.10.1 Detailed Description

Base class for the Service layer.

Definition at line 14 of file ServiceAbstract.hpp.

24.10.2 Constructor & Destructor Documentation

24.10.2.1 virtual SIMCRS::ServiceAbstract::~ServiceAbstract () [inline, virtual]

Destructor.

Definition at line 18 of file ServiceAbstract.hpp.

24.10.2.2 SIMCRS::ServiceAbstract::ServiceAbstract () [inline, protected]

Protected Default Constructor to ensure this class is abtract.

Definition at line 30 of file ServiceAbstract.hpp.

24.10.3 Member Function Documentation

24.10.3.1 virtual void SIMCRS::ServiceAbstract::toStream (std::ostream & *ioOut*) const [inline, virtual]

Dump a Business Object into an output stream.

Parameters:

ostream& the output stream.

Definition at line 22 of file ServiceAbstract.hpp.

24.10.3.2 virtual void SIMCRS::ServiceAbstract::fromStream (std::istream & *ioIn*) [inline, virtual]

Read a Business Object from an input stream.

Parameters:

istream& the input stream.

Definition at line 26 of file ServiceAbstract.hpp.

Referenced by operator>>().

The documentation for this class was generated from the following file:

- simcrs/service/ServiceAbstract.hpp

24.11 SIMCRS::SIMCRS_Service Class Reference

`#include <simcrs/SIMCRS_Service.hpp>`

Public Member Functions

- SIMCRS_Service (const stdair::BasLogParams &, const stdair::BasDBParams &, const CRSCode_T &)
- SIMCRS_Service (const stdair::BasLogParams &, const CRSCode_T &)
- SIMCRS_Service (stdair::STDAIR_ServicePtr_T, SEVMGR::SEVMGR_ServicePtr_T, const CRSCode_T &)
- void parseAndLoad (const stdair::ScheduleFilePath &, const stdair::ODFilePath &, const stdair::FRAT5FilePath &, const stdair::FFDisutilityFilePath &, const AIRRAC::YieldFilePath &, const SIMFQT::FareFilePath &)
- void initSnapshotAndRMEvents (const stdair::Date_T &iStartDate, const stdair::Date_T &iEndDate)
- ~SIMCRS_Service ()
- stdair::TravelSolutionList_T calculateSegmentPathList (const stdair::BookingRequestStruct &)
- void fareQuote (const stdair::BookingRequestStruct &, stdair::TravelSolutionList_T &)
- void calculateAvailability (stdair::TravelSolutionList_T &)
- bool sell (const stdair::TravelSolutionStruct &, const stdair::PartySize_T &)
- void takeSnapshots (const stdair::SnapshotStruct &)
- bool playCancellation (const stdair::CancellationStruct &)
- void optimise (const stdair::RMEventStruct &)
- bool sell (const std::string &iSegmentDateKey, const stdair::ClassCode_T &, const stdair::PartySize_T &)
- void buildSampleBom ()
- void clonePersistentBom ()
- void buildComplementaryLinks (stdair::BomRoot &)
- void buildSampleTravelSolutions (stdair::TravelSolutionList_T &)
- stdair::BookingRequestStruct buildSampleBookingRequest (const bool isForCRS=false)
- std::string jsonHandler (const stdair::JSONString &) const
- std::string csvDisplay () const
- std::string csvDisplay (const stdair::TravelSolutionList_T &) const

- std::string list (const stdair::AirlineCode_T &iAirlineCode="all", const stdair::FlightNumber_T &iFlightNumber=0) const
- std::string csvDisplay (const stdair::AirlineCode_T &, const stdair::FlightNumber_T &, const stdair::Date_T &iDepartureDate) const

24.11.1 Detailed Description

Interface for the SIMCRS Services.

Definition at line 42 of file SIMCRS_Service.hpp.

24.11.2 Constructor & Destructor Documentation

24.11.2.1 SIMCRS::SIMCRS_Service::SIMCRS_Service (const stdair::BasLogParams & *iLogParams*, const stdair::BasDBParams & *iDBParams*, const CRSCode_T & *iCRSCode*)

Constructor.

The init() method is called; see the corresponding documentation for more details.

A reference on an output stream is given, so that log outputs can be directed onto that stream.

Moreover, database connection parameters are given, so that a session can be created on the corresponding database.

Parameters:

const stdair::BasLogParams& Parameters for the output log stream.

const stdair::BasDBParams& Parameters for the database access.

const CRSCode_T& Code of the owner of the distribution system.

Definition at line 81 of file SIMCRS_Service.cpp.

24.11.2.2 SIMCRS::SIMCRS_Service::SIMCRS_Service (const stdair::BasLogParams & *iLogParams*, const CRSCode_T & *iCRSCode*)

Constructor.

The init() method is called; see the corresponding documentation for more details.

Moreover, a reference on an output stream is given, so that log outputs can be directed onto that stream.

Parameters:

const stdair::BasLogParams& Parameters for the output log stream.

const CRSCode_T& Code of the owner of the distribution system.

Definition at line 51 of file SIMCRS_Service.cpp.

24.11.2.3 SIMCRS::SIMCRS_Service::SIMCRS_Service (stdair::STDAIR_ServicePtr_T *ioSTDAIR_Service_ptr*, SEVMGR::SEVMGR_ServicePtr_T *ioSEVMGR_Service_ptr*, const CRSCode_T & *iCRSCode*)

Constructor.

The init() method is called; see the corresponding documentation for more details.

Moreover, as no reference on any output stream is given, it is assumed that the StdAir log service has already been initialised with the proper log output stream by some other methods in the calling chain (for instance, when the SIMCRS_Service is itself being initialised by another library service such as TVLSIM_-Service).

Parameters:

stdair::STDAIR_ServicePtr_T Reference on the STDAIR service.

SEVMGR::SEVMGR_ServicePtr_T Reference on the SEVMGR_Service.

const stdair::RandomSeed_T& Seed for the random generation.

const CRSCode_T& Code of the owner of the distribution system.

Definition at line 113 of file SIMCRS_Service.cpp.

24.11.2.4 SIMCRS::SIMCRS_Service::~SIMCRS_Service ()

Destructor.

Definition at line 144 of file SIMCRS_Service.cpp.

24.11.3 Member Function Documentation

24.11.3.1 void SIMCRS::SIMCRS_Service::parseAndLoad (const stdair::ScheduleFilePath & *iScheduleInputFilepath*, const stdair::ODFilePath & *iODInputFilepath*, const stdair::FRAT5FilePath & *iFRAT5InputFilepath*, const stdair::FFDisutilityFilePath & *iFFDisutilityInputFilepath*, const AIRRAC::YieldFilePath & *iYieldInputFilepath*, const SIMFQT::FareFilePath & *iFareInputFilepath*)

Parse the schedule, O&D, fare and yield input files.

The CSV files, describing the airline schedule, O&Ds, fares and yields for the simulator, are parsed and instantiated in memory accordingly.

Parameters:

const stdair::ScheduleFilePath Filename of the input schedule file.

const stdair::ODFilePath Filename of the input O&D file.

const stdair::FRAT5FilePath& Filename of the input FRAT5 file.

const stdair::FFDisutilityFilePath& Filename of the input FF disutility file.

const AIRRAC::YieldFilePath& Filename of the input yield file.

const SIMFQT::FareFilePath& Filename of the input fare file.

Definition at line 324 of file SIMCRS_Service.cpp.

References buildComplementaryLinks(), and clonePersistentBom().

Referenced by main().

24.11.3.2 void SIMCRS::SIMCRS_Service::initSnapshotAndRMEvents (const stdair::Date_T & *iStartDate*, const stdair::Date_T & *iEndDate*)

Initialise the snapshot and RM events for the inventories.

Parameters:

const stdair::Date_T& Start date of the simulation.

const stdair::Date_T& End date of the simulation.

Definition at line 635 of file SIMCRS_Service.cpp.

24.11.3.3 stdair::TravelSolutionList_T SIMCRS::SIMCRS_Service::calculateSegmentPathList (const stdair::BookingRequestStruct & *iBookingRequest*)

Construct the list of travel solutions corresponding to the booking request.

Definition at line 739 of file SIMCRS_Service.cpp.

Referenced by main().

24.11.3.4 void SIMCRS::SIMCRS_Service::fareQuote (const stdair::BookingRequestStruct & *iBookingRequest*, stdair::TravelSolutionList_T & *ioTravelSolutionList*)

Calculate the fare of each travel solutions in the list.

Definition at line 775 of file SIMCRS_Service.cpp.

Referenced by main().

24.11.3.5 void SIMCRS::SIMCRS_Service::calculateAvailability (stdair::TravelSolutionList_T & *ioTravelSolutionList*)

Compute the availability for each travel solution in the list.

Definition at line 806 of file SIMCRS_Service.cpp.

24.11.3.6 bool SIMCRS::SIMCRS_Service::sell (const stdair::TravelSolutionStruct & *iTravelSolution*, const stdair::PartySize_T & *iPartySize*)

Register a booking.

Definition at line 839 of file SIMCRS_Service.cpp.

Referenced by main().

24.11.3.7 void SIMCRS::SIMCRS_Service::takeSnapshots (const stdair::SnapshotStruct & *iSnapshot*)

Take inventory snapshots.

Definition at line 925 of file SIMCRS_Service.cpp.

24.11.3.8 bool SIMCRS::SIMCRS_Service::playCancellation (const stdair::CancellationStruct & *iCancellation*)

Play cancellation.

Definition at line 886 of file SIMCRS_Service.cpp.

24.11.3.9 void SIMCRS::SIMCRS_Service::optimise (const stdair::RMEventStruct & *iRMEvent*)

Optimise (revenue management) an flight-date/network-date

Definition at line 944 of file SIMCRS_Service.cpp.

24.11.3.10 bool SIMCRS::SIMCRS_Service::sell (const std::string & *iSegmentDateKey*, const stdair::ClassCode_T & *iClassCode*, const stdair::PartySize_T & *iPartySize*)

Register a booking.

Parameters:

const std::string& Key for the segment on which the sale is made

const stdair::ClassCode_T& Class code where the sale is made

const stdair::PartySize_T& Party size

Returns:

bool Whether or not the sale was successfull

Definition at line 593 of file SIMCRS_Service.cpp.

24.11.3.11 void SIMCRS::SIMCRS_Service::buildSampleBom ()

Build a sample BOM tree, and attach it to the BomRoot instance.

As for now, the BOM sample tree is the one built by the AirInv component.

See also:

AIRINV::AIRINV_Master_Service and stdair::CmdBomManager for more details.

Definition at line 402 of file SIMCRS_Service.cpp.

References buildComplementaryLinks(), and clonePersistentBom().

Referenced by main().

24.11.3.12 void SIMCRS::SIMCRS_Service::clonePersistentBom ()

Clone the persistent BOM object.

Definition at line 481 of file SIMCRS_Service.cpp.

References buildComplementaryLinks().

Referenced by buildSampleBom(), and parseAndLoad().

24.11.3.13 void SIMCRS::SIMCRS_Service::buildComplementaryLinks (stdair::BomRoot & *ioBomRoot*)

Build all the complementary links in the given bom root object.

Note:

Do nothing for now.

Definition at line 548 of file SIMCRS_Service.cpp.

Referenced by buildSampleBom(), clonePersistentBom(), and parseAndLoad().

24.11.3.14 void SIMCRS::SIMCRS_Service::buildSampleTravelSolutions (stdair::TravelSolutionList_T & *ioTravelSolutionList*)

Build a sample list of travel solutions.

As of now (March 2011), that list is made of the following travel solutions:

- BA9
- LHR-SYD
- 2011-06-10
- Q
- WTP: 900
- Change fee: 20; Non refundable; Saturday night stay

See also:

stdair::CmdBomManager for more details.

Parameters:

TravelSolutionList_T& Sample list of travel solution structures. It should be given empty. It is altered with the returned sample.

Definition at line 554 of file SIMCRS_Service.cpp.

24.11.3.15 stdair::BookingRequestStruct SIMCRS::SIMCRS_Service::buildSampleBookingRequest (const bool *isForCRS* = `false`)

Build a sample booking request structure.

As of now (March 2011), the sample booking request is made of the following parameters:

- Return trip (inbound): LHR-SYD (POS: LHR, Channel: DN),
- Departing 10-JUN-2011 around 8:00, staying 7 days
- Requested on 15-MAY-2011 at 10:00
- Economy cabin, 3 persons, FF member

- WTP: 1000.0 EUR
- Dis-utility: 100.0 EUR/hour

As of now (March 2011), the CRS-related booking request is made of the following parameters:

- Return trip (inbound): SIN-BKK (POS: SIN, Channel: IN),
- Departing 30-JAN-2010 around 10:00, staying 7 days
- Requested on 22-JAN-2010 at 10:00
- Economy cabin, 3 persons, FF member
- WTP: 1000.0 EUR
- Dis-utility: 100.0 EUR/hour

See also:

stdair::CmdBomManager for more details.

Parameters:

const bool isForCRS Whether the sample booking request is for CRS.

Returns:

BookingRequestStruct& Sample booking request structure.

Definition at line 574 of file SIMCRS_Service.cpp.

Referenced by main().

24.11.3.16 std::string SIMCRS::SIMCRS_Service::jsonHandler (const stdair::JSONString & *iJSONString*) const

Dispatch the JSon command string to the AirInv service. (Only AirInv has json export commands for now).

Parameters:

const stdair::JSONString& Input string which contained the JSon command string.

Returns:

std::string Output string in which the asking objects are logged/dumped with a JSon format.

Definition at line 615 of file SIMCRS_Service.cpp.

24.11.3.17 std::string SIMCRS::SIMCRS_Service::csvDisplay () const

Recursively display (dump in the returned string) the objects of the BOM tree.

Returns:

std::string Output string in which the BOM tree is logged/dumped.

Definition at line 654 of file SIMCRS_Service.cpp.

Referenced by main().

24.11.3.18 std::string SIMCRS::SIMCRS_Service::csvDisplay (const stdair::TravelSolutionList_T & *ioTravelSolutionList*) const

Display (dump in the returned string) the full list of travel solution structures.

Returns:

std::string Output string in which the list of travel solutions is logged/dumped.

Definition at line 675 of file SIMCRS_Service.cpp.

24.11.3.19 std::string SIMCRS::SIMCRS_Service::list (const stdair::AirlineCode_T & *iAirlineCode* = "all", const stdair::FlightNumber_T & *iFlightNumber* = 0) const

Display the list of flight-dates (contained within the BOM tree) corresponding to the parameters given as input.

Parameters:

const AirlineCode& Airline for which the flight-dates should be displayed. If set to "all" (the default), all the inventories will be displayed.

const FlightNumber_T& Flight number for which all the departure dates should be displayed. If set to 0 (the default), all the flight numbers will be displayed.

Returns:

std::string Output string in which the BOM tree is logged/dumped.

Definition at line 695 of file SIMCRS_Service.cpp.

24.11.3.20 std::string SIMCRS::SIMCRS_Service::csvDisplay (const stdair::AirlineCode_T & *iAirlineCode*, const stdair::FlightNumber_T & *iFlightNumber*, const stdair::Date_T & *iDepartureDate*) const

Recursively display (dump in the returned string) the flight-date corresponding to the parameters given as input.

Parameters:

const stdair::AirlineCode_T& Airline code of the flight to display

const stdair::FlightNumber_T& Flight number of the flight to display.

const stdair::Date_T& Departure date of the flight to display.

Returns:

std::string Output string in which the BOM tree is logged/dumped.

Definition at line 716 of file SIMCRS_Service.cpp.

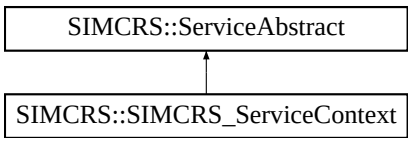
The documentation for this class was generated from the following files:

- simcrs/SIMCRS_Service.hpp
- simcrs/service/SIMCRS_Service.cpp

24.12 SIMCRS::SIMCRS_ServiceContext Class Reference

Class holding the context of the Simcrs services.

`#include <simcrs/service/SIMCRS_ServiceContext.hpp>`Inheritance diagram for SIMCRS::SIMCRS_ServiceContext::



Public Member Functions

- virtual void toStream (std::ostream &ioOut) const
- virtual void fromStream (std::istream &ioIn)

Friends

- class SIMCRS_Service
- class FacSimcrsServiceContext

24.12.1 Detailed Description

Class holding the context of the Simcrs services.

Definition at line 32 of file SIMCRS_ServiceContext.hpp.

24.12.2 Member Function Documentation

24.12.2.1 virtual void SIMCRS::ServiceAbstract::toStream (std::ostream & *ioOut*) const [inline, virtual, inherited]

Dump a Business Object into an output stream.

Parameters:

ostream& the output stream.

Definition at line 22 of file ServiceAbstract.hpp.

24.12.2.2 virtual void SIMCRS::ServiceAbstract::fromStream (std::istream & *ioIn*) [inline, virtual, inherited]

Read a Business Object from an input stream.

Parameters:

istream& the input stream.

Definition at line 26 of file ServiceAbstract.hpp.

Referenced by operator>>().

24.12.3 Friends And Related Function Documentation

24.12.3.1 friend class SIMCRS_Service [friend]

The SIMCRS_Service class should be the sole class to get access to ServiceContext content: general users do not want to bother with a context interface.

Definition at line 38 of file SIMCRS_ServiceContext.hpp.

24.12.3.2 friend class FacSimcrsServiceContext [friend]

Definition at line 39 of file SIMCRS_ServiceContext.hpp.

The documentation for this class was generated from the following files:

- simcrs/service/SIMCRS_ServiceContext.hpp
- simcrs/service/SIMCRS_ServiceContext.cpp

25 File Documentation

25.1 doc/local/authors.doc File Reference

25.2 doc/local/codingrules.doc File Reference

25.3 doc/local/copyright.doc File Reference

25.4 doc/local/documentation.doc File Reference

25.5 doc/local/features.doc File Reference

25.6 doc/local/help_wanted.doc File Reference

25.7 doc/local/howto_release.doc File Reference

25.8 doc/local/index.doc File Reference

25.9 doc/local/installation.doc File Reference

25.10 doc/local/linking.doc File Reference

25.11 doc/local/test.doc File Reference

25.12 doc/local/users_guide.doc File Reference

25.13 doc/local/verification.doc File Reference

25.14 doc/tutorial/tutorial.doc File Reference

25.15 simcrs/basic/BasConst.cpp File Reference

```
#include <simcrs/basic/BasConst_General.hpp>
#include <simcrs/basic/BasConst_SIMCRS_Service.hpp>
```

Namespaces

- namespace SIMCRS

Variables

- const std::string SIMCRS::DEFAULT_CRS_CODE = "1S"

25.16 BasConst.cpp

```
00001 // ////////////////////////////////////////////////////////////////////
00002 // Import section
00003 // ////////////////////////////////////////////////////////////////////
00004 #include <simcrs/basic/BasConst_General.hpp>
00005 #include <simcrs/basic/BasConst_SIMCRS_Service.hpp>
00006
00007 namespace SIMCRS {
00008
00010   const std::string DEFAULT_CRS_CODE = "1S";
00011
00012 }
```

25.17 simcrs/basic/BasConst_General.hpp File Reference

Namespaces

- namespace SIMCRS

25.18 BasConst_General.hpp

```
00001 #ifndef __SIMCRS_BAS_BASCONST_GENERAL_HPP
00002 #define __SIMCRS_BAS_BASCONST_GENERAL_HPP
00003
00004 // ////////////////////////////////////////////////////////////////////
00005 // Import section
00006 // ////////////////////////////////////////////////////////////////////
00007
00008 namespace SIMCRS {
00009
00010 }
00011 #endif // __SIMCRS_BAS_BASCONST_GENERAL_HPP
```

25.19 simcrs/basic/BasConst_SIMCRS_Service.hpp File Reference

```
#include <string>
```

Namespaces

- namespace SIMCRS

25.20 BasConst_SIMCRS_Service.hpp

```
00001 #ifndef __SIMCRS_BAS_BASCONST_SIMCRS_SERVICE_HPP
00002 #define __SIMCRS_BAS_BASCONST_SIMCRS_SERVICE_HPP
00003
00004 // //////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////
00007 #include <string>
00008
00009 namespace SIMCRS {
00010
00012   extern const std::string DEFAULT_CRS_CODE;
00013
00014 }
00015 #endif // __SIMCRS_BAS_BASCONST_SIMCRS_SERVICE_HPP
```

25.21 simcrs/batches/simcrs.cpp File Reference

```
#include <sstream>
#include <fstream>
#include <string>
#include <boost/program_options.hpp>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/basic/BasLogParams.hpp>
#include <stdair/basic/BasDBParams.hpp>
#include <stdair/basic/BasFileMgr.hpp>
#include <stdair/bom/TravelSolutionStruct.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/service/Logger.hpp>
#include <simfqt/SIMFQT_Types.hpp>
#include <simcrs/SIMCRS_Service.hpp>
#include <simcrs/config/simcrs-paths.hpp>
```

Functions

- const std::string K_SIMCRS_DEFAULT_LOG_FILENAME ("simcrs.log")
- const std::string K_SIMCRS_DEFAULT_SCHEDULE_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/schedule01.csv")
- const std::string K_SIMCRS_DEFAULT_OND_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/ond01.csv")
- const std::string K_SIMCRS_DEFAULT_FRAT5_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/frat5.csv")
- const std::string K_SIMCRS_DEFAULT_FF_DISUTILITY_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/ffDisutility.csv")
- const std::string K_SIMCRS_DEFAULT_YIELD_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/yieldstore01.csv")
- const std::string K_SIMCRS_DEFAULT_FARE_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/fare01.csv")
- const std::string K_SIMCRS_DEFAULT_DB_USER ("dsim")
- const std::string K_SIMCRS_DEFAULT_DB_PASSWD ("dsim")
- const std::string K_SIMCRS_DEFAULT_DB_DBNAME ("sim_dsim")
- const std::string K_SIMCRS_DEFAULT_DB_HOST ("localhost")
- const std::string K_SIMCRS_DEFAULT_DB_PORT ("3306")
- template<class T >
  std::ostream & operator<< (std::ostream &os, const std::vector< T > &v)
- int readConfiguration (int argc, char *argv[ ], bool &ioIsBuiltin, stdair::Filename_T &ioScheduleInputFilename, stdair::Filename_T &ioOnDInputFilename, stdair::Filename_T &ioFRAT5Filename, stdair::Filename_T &ioFFDisutilityFilename, stdair::Filename_T &ioYieldInputFilename, stdair::Filename_T &ioFareInputFilename, stdair::Filename_T &ioLogFilename, std::string &ioDBUser, std::string &ioDBPasswd, std::string &ioDBHost, std::string &ioDBPort, std::string &ioDBDBName)
- int main (int argc, char *argv[ ])

Variables

- const bool K_SIMCRS_DEFAULT_BUILT_IN_INPUT = false
- const int K_SIMCRS_EARLY_RETURN_STATUS = 99

25.21.1 Function Documentation

25.21.1.1 const std::string K_SIMCRS_DEFAULT_LOG_FILENAME ("simcrs.log")

Default name and location for the log file.

Referenced by readConfiguration().

25.21.1.2 const std::string K_SIMCRS_DEFAULT_SCHEDULE_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/schedule01.csv")

Default name and location for the (CSV) schedule input file.

Referenced by readConfiguration().

25.21.1.3 const std::string K_SIMCRS_DEFAULT_OND_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/ond01.csv")

Default name and location for the (CSV) O&D input file.

Referenced by readConfiguration().

25.21.1.4 const std::string K_SIMCRS_DEFAULT_FRAT5_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/frat5.csv")

FRAT5 curve input file name

Referenced by readConfiguration().

25.21.1.5 const std::string K_SIMCRS_DEFAULT_FF_DISUTILITY_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/ffDisutility.csv")

Fare family disutility curve input file name

Referenced by readConfiguration().

25.21.1.6 const std::string K_SIMCRS_DEFAULT_YIELD_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/yieldstore01.csv")

Default name and location for the (CSV) yield input file.

Referenced by readConfiguration().

25.21.1.7 const std::string K_SIMCRS_DEFAULT_FARE_INPUT_FILENAME (STDAIR_SAMPLE_DIR"/fare01.csv")

Default name and location for the (CSV) fare input file.

Referenced by readConfiguration().

25.21.1.8 const std::string K_SIMCRS_DEFAULT_DB_USER ("dsim")

Default name and location for the MySQL database.

Referenced by readConfiguration().

25.21.1.9 const std::string K_SIMCRS_DEFAULT_DB_PASSWD ("dsim")

Referenced by readConfiguration().

25.21.1.10 const std::string K_SIMCRS_DEFAULT_DB_DBNAME ("sim_dsim")

Referenced by readConfiguration().

25.21.1.11 const std::string K_SIMCRS_DEFAULT_DB_HOST ("localhost")

Referenced by readConfiguration().

25.21.1.12 const std::string K_SIMCRS_DEFAULT_DB_PORT ("3306")

Referenced by readConfiguration().

25.21.1.13 template<class T > std::ostream& operator<< (std::ostream & *os*, const std::vector< T > & *v*) [inline]

Definition at line 80 of file simcrs.cpp.

25.21.1.14 int readConfiguration (int *argc*, char * *argv*[ ], bool & *ioIsBuiltin*, stdair::Filename_T & *ioScheduleInputFilename*, stdair::Filename_T & *ioOnDInputFilename*, stdair::Filename_T & *ioFRAT5Filename*, stdair::Filename_T & *ioFFDisutilityFilename*, stdair::Filename_T & *ioYieldInputFilename*, stdair::Filename_T & *ioFareInputFilename*, stdair::Filename_T & *ioLogFilename*, std::string & *ioDBUser*, std::string & *ioDBPasswd*, std::string & *ioDBHost*, std::string & *ioDBPort*, std::string & *ioDBDBName*)

Read and parse the command line options.

Definition at line 90 of file simcrs.cpp.

References K_SIMCRS_DEFAULT_BUILT_IN_INPUT, K_SIMCRS_DEFAULT_DB_DBNAME(), K_SIMCRS_DEFAULT_DB_HOST(), K_SIMCRS_DEFAULT_DB_PASSWD(), K_SIMCRS_DEFAULT_DB_PORT(), K_SIMCRS_DEFAULT_DB_USER(), K_SIMCRS_DEFAULT_FARE_INPUT_FILENAME(), K_SIMCRS_DEFAULT_FF_DISUTILITY_INPUT_FILENAME(), K_SIMCRS_DEFAULT_FRAT5_INPUT_FILENAME(), K_SIMCRS_DEFAULT_LOG_FILENAME(), K_SIMCRS_DEFAULT_OND_INPUT_FILENAME(), K_SIMCRS_DEFAULT_SCHEDULE_INPUT_FILENAME(), K_SIMCRS_DEFAULT_YIELD_INPUT_FILENAME(), K_SIMCRS_EARLY_RETURN_STATUS, PACKAGE_NAME, PACKAGE_VERSION, and PREFIXDIR.

Referenced by main().

25.21.1.15 int main (int *argc*, char * *argv*[ ])

Definition at line 313 of file simcrs.cpp.

References SIMCRS::SIMCRS_Service::buildSampleBom(), SIMCRS::SIMCRS_Service::buildSampleBookingRequest(), SIMCRS::SIMCRS_Service::calculateSegmentPathList(), SIMCRS::SIMCRS_Service::csvDisplay(), SIMCRS::SIMCRS_Service::fareQuote(), K_SIMCRS_EARLY_RETURN_STATUS, SIMCRS::SIMCRS_Service::parseAndLoad(), readConfiguration(), and SIMCRS::SIMCRS_Service::sell().

25.21.2 Variable Documentation

25.21.2.1 const bool K_SIMCRS_DEFAULT_BUILT_IN_INPUT = false

Default for the BOM tree building. The BOM tree can either be built-in or provided by an input file. That latter must then be given with input file options (-s, -o, -f, -y).

Definition at line 67 of file simcrs.cpp.

Referenced by readConfiguration().

25.21.2.2 const int K_SIMCRS_EARLY_RETURN_STATUS = 99

Early return status (so that it can be differentiated from an error).

Definition at line 87 of file simcrs.cpp.

Referenced by main(), and readConfiguration().

25.22 simcrs.cpp

```
00001 // STL
00002 #include <sstream>
00003 #include <fstream>
00004 #include <string>
00005 // Boost (Extended STL)
00006 #include <boost/program_options.hpp>
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/basic/BasLogParams.hpp>
00010 #include <stdair/basic/BasDBParams.hpp>
00011 #include <stdair/basic/BasFileMgr.hpp>
00012 #include <stdair/bom/TravelSolutionStruct.hpp>
00013 #include <stdair/bom/BookingRequestStruct.hpp>
00014 #include <stdair/service/Logger.hpp>
00015 // SimFQT
00016 #include <simfqt/SIMFQT_Types.hpp>
00017 // SimCRS
00018 #include <simcrs/SIMCRS_Service.hpp>
00019 #include <simcrs/config/simcrs-paths.hpp>
00020
00021 // ///////// Constants //////
00025 const std::string K_SIMCRS_DEFAULT_LOG_FILENAME ("simcrs.log");
00026
00030 const std::string K_SIMCRS_DEFAULT_SCHEDULE_INPUT_FILENAME (STDAIR_SAMPLE_DIR
00031                                                             "/schedule01.csv");
00032
00036 const std::string K_SIMCRS_DEFAULT_OND_INPUT_FILENAME (STDAIR_SAMPLE_DIR
00037                                                        "/ond01.csv");
00038
00042 const std::string K_SIMCRS_DEFAULT_FRAT5_INPUT_FILENAME (STDAIR_SAMPLE_DIR
00043                                                          "/frat5.csv");
00047 const std::string K_SIMCRS_DEFAULT_FF_DISUTILITY_INPUT_FILENAME (
      STDAIR_SAMPLE_DIR
00048                                                                 "/ffDisutility.csv
      ");
00049
00053 const std::string K_SIMCRS_DEFAULT_YIELD_INPUT_FILENAME (STDAIR_SAMPLE_DIR
00054                                                           "/yieldstore01.csv");
00055
00059 const std::string K_SIMCRS_DEFAULT_FARE_INPUT_FILENAME (STDAIR_SAMPLE_DIR
00060                                                         "/fare01.csv");
00061
00067 const bool K_SIMCRS_DEFAULT_BUILT_IN_INPUT = false;
00068
00072 const std::string K_SIMCRS_DEFAULT_DB_USER ("dsim");
00073 const std::string K_SIMCRS_DEFAULT_DB_PASSWD ("dsim");
00074 const std::string K_SIMCRS_DEFAULT_DB_DBNAME ("sim_dsim");
00075 const std::string K_SIMCRS_DEFAULT_DB_HOST ("localhost");
00076 const std::string K_SIMCRS_DEFAULT_DB_PORT ("3306");
00077
00078 // ///////// Parsing of Options & Configuration /////////
00079 // A helper function to simplify the main part.
00080 template<class T> std::ostream& operator<< (std::ostream& os,
00081                                             const std::vector<T>& v) {
00082   std::copy (v.begin(), v.end(), std::ostream_iterator<T> (std::cout, " "));
00083   return os;
00084 }
00085
00087 const int K_SIMCRS_EARLY_RETURN_STATUS = 99;
00088
00090 int readConfiguration (int argc, char* argv[],
00091                        bool& ioIsBuiltin,
00092                        stdair::Filename_T& ioScheduleInputFilename,
00093                        stdair::Filename_T& ioOnDInputFilename,
00094                        stdair::Filename_T& ioFRAT5Filename,
```

```
00095                         stdair::Filename_T& ioFFDisutilityFilename,
00096                         stdair::Filename_T& ioYieldInputFilename,
00097                         stdair::Filename_T& ioFareInputFilename,
00098                         stdair::Filename_T& ioLogFilename,
00099                         std::string& ioDBUser, std::string& ioDBPasswd,
00100                         std::string& ioDBHost, std::string& ioDBPort,
00101                         std::string& ioDBDBName) {
00102   // Default for the built-in input
00103   ioIsBuiltin = K_SIMCRS_DEFAULT_BUILT_IN_INPUT;
00104
00105   // Declare a group of options that will be allowed only on command line
00106   boost::program_options::options_description generic ("Generic options");
00107   generic.add_options()
00108     ("prefix", "print installation prefix")
00109     ("version,v", "print version string")
00110     ("help,h", "produce help message");
00111
00112   // Declare a group of options that will be allowed both on command
00113   // line and in config file
00114   boost::program_options::options_description config ("Configuration");
00115   config.add_options()
00116     ("builtin,b",
00117      "The sample BOM tree can be either built-in or parsed from input files. In t
      hat latter case, the input files must be specified as well (e.g., -s/--schedule,
       -o/--ond, -f/--fare, -y/--yield)")
00118     ("schedule,s",
00119      boost::program_options::value< std::string >(&ioScheduleInputFilename)->defa
      ult_value(K_SIMCRS_DEFAULT_SCHEDULE_INPUT_FILENAME),
00120      "(CVS) input file for the schedules")
00121     ("ond,o",
00122      boost::program_options::value< std::string >(&ioOnDInputFilename)->default_v
      alue(K_SIMCRS_DEFAULT_OND_INPUT_FILENAME),
00123      "(CVS) input file for the O&D definitions")
00124     ("frat5,F",
00125      boost::program_options::value< std::string >(&ioFRAT5Filename)->default_valu
      e(K_SIMCRS_DEFAULT_FRAT5_INPUT_FILENAME),
00126      "(CSV) input file for the FRAT5 Curve")
00127     ("ff_disutility,D",
00128      boost::program_options::value< std::string >(&ioFFDisutilityFilename)->defau
      lt_value(K_SIMCRS_DEFAULT_FF_DISUTILITY_INPUT_FILENAME),
00129      "(CSV) input file for the FF disutility Curve")
00130     ("yield,y",
00131      boost::program_options::value< std::string >(&ioYieldInputFilename)->default
      _value(K_SIMCRS_DEFAULT_YIELD_INPUT_FILENAME),
00132      "(CVS) input file for the yields")
00133     ("fare,f",
00134      boost::program_options::value< std::string >(&ioFareInputFilename)->default_
      value(K_SIMCRS_DEFAULT_FARE_INPUT_FILENAME),
00135      "(CVS) input file for the fares")
00136     ("log,l",
00137      boost::program_options::value< std::string >(&ioLogFilename)->default_value(
      K_SIMCRS_DEFAULT_LOG_FILENAME),
00138      "Filepath for the logs")
00139     ("user,u",
00140      boost::program_options::value< std::string >(&ioDBUser)->default_value(
      K_SIMCRS_DEFAULT_DB_USER),
00141      "SQL database username")
00142     ("passwd,p",
00143      boost::program_options::value< std::string >(&ioDBPasswd)->default_value(
      K_SIMCRS_DEFAULT_DB_PASSWD),
00144      "SQL database password")
00145     ("host,H",
00146      boost::program_options::value< std::string >(&ioDBHost)->default_value(
      K_SIMCRS_DEFAULT_DB_HOST),
00147      "SQL database hostname")
00148     ("port,P",
00149      boost::program_options::value< std::string >(&ioDBPort)->default_value(
```

```
      K_SIMCRS_DEFAULT_DB_PORT),
00150      "SQL database port")
00151     ("dbname,m",
00152      boost::program_options::value< std::string >(&ioDBDBName)->default_value(
      K_SIMCRS_DEFAULT_DB_DBNAME),
00153      "SQL database name")
00154     ;
00155
00156   // Hidden options, will be allowed both on command line and
00157   // in config file, but will not be shown to the user.
00158   boost::program_options::options_description hidden ("Hidden options");
00159   hidden.add_options()
00160     ("copyright",
00161      boost::program_options::value< std::vector<std::string> >(),
00162      "Show the copyright (license)");
00163
00164   boost::program_options::options_description cmdline_options;
00165   cmdline_options.add(generic).add(config).add(hidden);
00166
00167   boost::program_options::options_description config_file_options;
00168   config_file_options.add(config).add(hidden);
00169
00170   boost::program_options::options_description visible ("Allowed options");
00171   visible.add(generic).add(config);
00172
00173   boost::program_options::positional_options_description p;
00174   p.add ("copyright", -1);
00175
00176   boost::program_options::variables_map vm;
00177   boost::program_options::
00178     store (boost::program_options::command_line_parser (argc, argv).
00179            options (cmdline_options).positional(p).run(), vm);
00180
00181   std::ifstream ifs ("simcrs.cfg");
00182   boost::program_options::store (parse_config_file (ifs, config_file_options),
00183                                  vm);
00184   boost::program_options::notify (vm);
00185
00186   if (vm.count ("help")) {
00187     std::cout << visible << std::endl;
00188     return K_SIMCRS_EARLY_RETURN_STATUS;
00189   }
00190
00191   if (vm.count ("version")) {
00192     std::cout << PACKAGE_NAME << ", version " << PACKAGE_VERSION << std::endl;
00193     return K_SIMCRS_EARLY_RETURN_STATUS;
00194   }
00195
00196   if (vm.count ("prefix")) {
00197     std::cout << "Installation prefix: " << PREFIXDIR << std::endl;
00198     return K_SIMCRS_EARLY_RETURN_STATUS;
00199   }
00200
00201   if (vm.count ("builtin")) {
00202     ioIsBuiltin = true;
00203   }
00204   const std::string isBuiltinStr = (ioIsBuiltin == true)?"yes":"no";
00205   std::cout << "The BOM should be built-in? " << isBuiltinStr << std::endl;
00206
00207   //
00208   std::ostringstream oErrorMessageStr;
00209   oErrorMessageStr << "Either the -b/--builtin option, or the combination of "
00210                    << "the -s/--schedule, -o/--ond, -f/--fare and -y/--yield "
00211                    << "options must be specified";
00212
00213   if (ioIsBuiltin == false) {
00214     if (vm.count ("schedule")) {
```

```
00215       ioScheduleInputFilename = vm["schedule"].as< std::string >();
00216       std::cout << "Schedule input filename is: " << ioScheduleInputFilename
00217                 << std::endl;
00218
00219     } else {
00220       // The built-in option is not selected. However, no schedule input file
00221       // is specified
00222       std::cerr << oErrorMessageStr.str() << std::endl;
00223     }
00224
00225     if (vm.count ("ond")) {
00226       ioOnDInputFilename = vm["ond"].as< std::string >();
00227       std::cout << "O&D input filename is: " << ioOnDInputFilename << std::endl;
00228
00229     } else {
00230       // The built-in option is not selected. However, no schedule input file
00231       // is specified
00232       std::cerr << oErrorMessageStr.str() << std::endl;
00233     }
00234
00235     if (vm.count ("frat5")) {
00236       ioFRAT5Filename = vm["frat5"].as< std::string >();
00237       std::cout << "FRAT5 input filename is: " << ioFRAT5Filename << std::endl;
00238
00239     } else {
00240       // The built-in option is not selected. However, no frat5 input file
00241       // is specified
00242       std::cerr << oErrorMessageStr.str() << std::endl;
00243     }
00244
00245     if (vm.count ("ff_disutility")) {
00246       ioFFDisutilityFilename = vm["ff_disutility"].as< std::string >();
00247       std::cout << "FF disutility input filename is: "
00248                 << ioFFDisutilityFilename << std::endl;
00249
00250     } else {
00251       // The built-in option is not selected. However, no ff
00252       // disutility input file is specified
00253       std::cerr << oErrorMessageStr.str() << std::endl;
00254     }
00255
00256     if (vm.count ("yield")) {
00257       ioYieldInputFilename = vm["yield"].as< std::string >();
00258       std::cout << "Yield input filename is: " << ioYieldInputFilename
00259                 << std::endl;
00260
00261     } else {
00262       // The built-in option is not selected. However, no schedule input file
00263       // is specified
00264       std::cerr << oErrorMessageStr.str() << std::endl;
00265     }
00266
00267     if (vm.count ("fare")) {
00268       ioFareInputFilename = vm["fare"].as< std::string >();
00269       std::cout << "Fare input filename is: " << ioFareInputFilename
00270                 << std::endl;
00271
00272     } else {
00273       // The built-in option is not selected. However, no schedule input file
00274       // is specified
00275       std::cerr << oErrorMessageStr.str() << std::endl;
00276     }
00277   }
00278
00279   if (vm.count ("log")) {
00280     ioLogFilename = vm["log"].as< std::string >();
00281     std::cout << "Log filename is: " << ioLogFilename << std::endl;
```

```
00282   }
00283
00284   if (vm.count ("user")) {
00285     ioDBUser = vm["user"].as< std::string >();
00286     std::cout << "SQL database user name is: " << ioDBUser << std::endl;
00287   }
00288
00289   if (vm.count ("passwd")) {
00290     ioDBPasswd = vm["passwd"].as< std::string >();
00291     //std::cout << "SQL database user password is: " << ioDBPasswd << std::endl;
00292   }
00293
00294   if (vm.count ("host")) {
00295     ioDBHost = vm["host"].as< std::string >();
00296     std::cout << "SQL database host name is: " << ioDBHost << std::endl;
00297   }
00298
00299   if (vm.count ("port")) {
00300     ioDBPort = vm["port"].as< std::string >();
00301     std::cout << "SQL database port number is: " << ioDBPort << std::endl;
00302   }
00303
00304   if (vm.count ("dbname")) {
00305     ioDBDBName = vm["dbname"].as< std::string >();
00306     std::cout << "SQL database name is: " << ioDBDBName << std::endl;
00307   }
00308
00309   return 0;
00310 }
00311
00312 // ///////// M A I N ////////////
00313 int main (int argc, char* argv[]) {
00314
00315   // State whether the BOM tree should be built-in or parsed from an
00316   // input file
00317   bool isBuiltin;
00318
00319   // Schedule input filename
00320   stdair::Filename_T lScheduleInputFilename;
00321
00322   // O&D input filename
00323   stdair::Filename_T lOnDInputFilename;
00324
00325   // FRAT5 input filename
00326   std::string lFRAT5InputFilename;
00327
00328   // FF disutility input filename
00329   std::string lFFDisutilityInputFilename;
00330
00331   // Yield input filename
00332   stdair::Filename_T lYieldInputFilename;
00333
00334   // Fare input filename
00335   stdair::Filename_T lFareInputFilename;
00336
00337   // Output log File
00338   stdair::Filename_T lLogFilename;
00339
00340   // SQL database parameters
00341   std::string lDBUser;
00342   std::string lDBPasswd;
00343   std::string lDBHost;
00344   std::string lDBPort;
00345   std::string lDBDBName;
00346
00347   // CRS code
00348   const SIMCRS::CRSCode_T lCRSCode ("1P");
```

```
00349
00350   // Call the command-line option parser
00351   const int lOptionParserStatus =
00352     readConfiguration (argc, argv, isBuiltin,
00353                        lScheduleInputFilename, lOnDInputFilename,
00354                        lFRAT5InputFilename, lFFDisutilityInputFilename,
00355                        lYieldInputFilename, lFareInputFilename, lLogFilename,
00356                        lDBUser, lDBPasswd, lDBHost, lDBPort, lDBDBName);
00357
00358   if (lOptionParserStatus == K_SIMCRS_EARLY_RETURN_STATUS) {
00359     return 0;
00360   }
00361
00362   // Set the database parameters
00363   const stdair::BasDBParams lDBParams (lDBUser, lDBPasswd, lDBHost, lDBPort,
00364                                        lDBDBName);
00365
00366   // Set the log parameters
00367   std::ofstream logOutputFile;
00368   // Open and clean the log outputfile
00369   logOutputFile.open (lLogFilename.c_str());
00370   logOutputFile.clear();
00371
00372   // Initialise the list of classes/buckets
00373   const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
00374   SIMCRS::SIMCRS_Service simcrsService (lLogParams, lCRSCode);
00375
00376   // Check wether or not (CSV) input files should be read
00377   if (isBuiltin == true) {
00378
00379     // Build the sample BOM tree
00380     simcrsService.buildSampleBom();
00381
00382   } else {
00383     // Build the BOM tree from parsing input files
00384     stdair::ScheduleFilePath lScheduleFilePath (lScheduleInputFilename);
00385     stdair::ODFilePath lODFilePath (lOnDInputFilename);
00386     stdair::FRAT5FilePath lFRAT5FilePath (lFRAT5InputFilename);
00387     stdair::FFDisutilityFilePath lFFDisutilityFilePath (lFFDisutilityInputFilenam
      e);
00388     const SIMFQT::FareFilePath lFareFilePath (lFareInputFilename);
00389     const AIRRAC::YieldFilePath lYieldFilePath (lYieldInputFilename);
00390     simcrsService.parseAndLoad (lScheduleFilePath, lODFilePath,
00391                                 lFRAT5FilePath, lFFDisutilityFilePath,
00392                                 lYieldFilePath, lFareFilePath);
00393   }
00394
00395   // TODO (issue #37707): instead of building a sample, read the parameters
00396   //      from the command-line options, and build the corresponding
00397   //      booking request
00398   const bool isForCRS = true;
00399   const stdair::BookingRequestStruct& lBookingRequest =
00400     simcrsService.buildSampleBookingRequest (isForCRS);
00401
00402   // Calculate the travel solutions corresponding to the given booking request
00403   stdair::TravelSolutionList_T lTravelSolutionList =
00404     simcrsService.calculateSegmentPathList (lBookingRequest);
00405
00406   // Check whether everything was fine
00407   if (lTravelSolutionList.empty() == true) {
00408     STDAIR_LOG_ERROR ("No travel solution has been found for: "
00409                       << lBookingRequest.display());
00410     return -1;
00411   }
00412
00413   // Price the travel solution
00414   simcrsService.fareQuote (lBookingRequest, lTravelSolutionList);
```

```
00415
00416   // Choose a random travel solution: the first one.
00417   stdair::TravelSolutionStruct& lChosenTravelSolution =
00418     lTravelSolutionList.front();
00419
00420   // Get the segment path of the travel solution.
00421   const stdair::KeyList_T& lsegmentDateKeyList =
00422     lChosenTravelSolution.getSegmentPath();
00423
00424   const stdair::FareOptionList_T& lFareOptionList =
00425     lChosenTravelSolution.getFareOptionList();
00426
00427   // Check whether everything was fine
00428   if (lFareOptionList.empty() == true) {
00429     STDAIR_LOG_ERROR ("No fare option for the chosen travel solution: "
00430                       << lChosenTravelSolution.display());
00431     return -1;
00432   }
00433
00434   //
00435   const stdair::FareOptionStruct& lFareOption = lFareOptionList.front();
00436   lChosenTravelSolution.setChosenFareOption (lFareOption);
00437
00438   // DEBUG
00439   const std::string& lSegmentDateKey = lsegmentDateKeyList.front();
00440   STDAIR_LOG_DEBUG ("The chosen travel solution is: " << lSegmentDateKey
00441                     << ", the fare is: " << lFareOption.getFare() << " Euros.");
00442
00443   // Make a booking (reminder: party size is 3)
00444   const stdair::PartySize_T lPartySize (3);
00445   const bool isSellSuccessful =
00446     simcrsService.sell (lChosenTravelSolution, lPartySize);
00447
00448   // DEBUG
00449   STDAIR_LOG_DEBUG ("Sale ('" << lBookingRequest << "'): "
00450                     << " successful? " << isSellSuccessful);
00451
00452   // DEBUG: Display the whole BOM tree
00453   const std::string& lCSVDump = simcrsService.csvDisplay();
00454   STDAIR_LOG_DEBUG (lCSVDump);
00455
00456   // Close the Log outputFile
00457   logOutputFile.close();
00458
00459   /*
00460     Note: as that program is not intended to be run on a server in
00461     production, it is better not to catch the exceptions. When it
00462     happens (that an exception is throwned), that way we get the
00463     call stack.
00464   */
00465
00466   return 0;
00467 }
```

25.23 simcrs/bom/BomAbstract.cpp File Reference

```
#include <simcrs/bom/BomAbstract.hpp>
```

Namespaces

- namespace SIMCRS

25.24 BomAbstract.cpp

```
00001 // ////////////////////////////////////////////////////////////////////
00002 // Import section
00003 // ////////////////////////////////////////////////////////////////////
00004 // SIMCRS
00005 #include <simcrs/bom/BomAbstract.hpp>
00006
00007 namespace SIMCRS {
00008
00009 }
```

25.25 simcrs/bom/BomAbstract.hpp File Reference

```
#include <iosfwd>
#include <string>
```

Classes

- class SIMCRS::BomAbstract

Namespaces

- namespace SIMCRS

Functions

- template<class charT , class traits >
  std::basic_ostream< charT, traits > & operator<< (std::basic_ostream< charT, traits > &ioOut, const SIMCRS::BomAbstract &iBom)
- template<class charT , class traits >
  std::basic_istream< charT, traits > & operator>> (std::basic_istream< charT, traits > &ioIn, SIMCRS::BomAbstract &ioBom)

25.25.1 Function Documentation

25.25.1.1 template<class charT , class traits > std::basic_ostream<charT, traits>& operator<< (std::basic_ostream< charT, traits > & *ioOut*, const SIMCRS::BomAbstract & *iBom*) [inline]

Piece of code given by Nicolai M. Josuttis, Section 13.12.1 "Implementing Output Operators" (p653) of his book "The C++ Standard Library: A Tutorial and Reference", published by Addison-Wesley.

Definition at line 56 of file BomAbstract.hpp.

25.25.1.2 template<class charT , class traits > std::basic_istream<charT, traits>& operator>> (std::basic_istream< charT, traits > & *ioIn*, SIMCRS::BomAbstract & *ioBom*) [inline]

Piece of code given by Nicolai M. Josuttis, Section 13.12.1 "Implementing Output Operators" (pp655-657) of his book "The C++ Standard Library: A Tutorial and Reference", published by Addison-Wesley.

Definition at line 84 of file BomAbstract.hpp.

References SIMCRS::BomAbstract::fromStream().

25.26 BomAbstract.hpp

```
00001 #ifndef __SIMCRS_BOM_BOMABSTRACT_HPP
00002 #define __SIMCRS_BOM_BOMABSTRACT_HPP
00003
00004 // //////////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////////
00007 // STL
00008 #include <iosfwd>
00009 #include <string>
00010
00011 namespace SIMCRS {
00012
00014   class BomAbstract {
00015     friend class FacBomAbstract;
00016   public:
00017     // ////////// Display support methods /////////
00020     virtual void toStream (std::ostream& ioOut) const = 0;
00021
00024     virtual void fromStream (std::istream& ioIn) = 0;
00025
00027     virtual std::string toString() const = 0;
00028
00031     virtual std::string describeKey() const = 0;
00032
00035     virtual std::string describeShortKey() const = 0;
00036
00037
00038   protected:
00040     BomAbstract() {}
00041     BomAbstract(const BomAbstract&) {}
00042
00044     virtual ~BomAbstract() {}
00045   };
00046 }
00047
00053 template <class charT, class traits>
00054 inline
00055 std::basic_ostream<charT, traits>&
00056 operator<< (std::basic_ostream<charT, traits>& ioOut,
00057             const SIMCRS::BomAbstract& iBom) {
00063   std::basic_ostringstream<charT,traits> ostr;
00064   ostr.copyfmt (ioOut);
00065   ostr.width (0);
00066
00067   // Fill string stream
00068   iBom.toStream (ostr);
00069
00070   // Print string stream
00071   ioOut << ostr.str();
00072
00073   return ioOut;
00074 }
00075
00081 template <class charT, class traits>
00082 inline
00083 std::basic_istream<charT, traits>&
00084 operator>> (std::basic_istream<charT, traits>& ioIn,
00085             SIMCRS::BomAbstract& ioBom) {
00086   // Fill Bom object with input stream
00087   ioBom.fromStream (ioIn);
00088   return ioIn;
00089 }
00090
00091 #endif // __SIMCRS_BOM_BOMABSTRACT_HPP
```

25.27 simcrs/command/DistributionManager.cpp File Reference

```
#include <cassert>
#include <stdair/bom/FareOptionStruct.hpp>
#include <stdair/bom/TravelSolutionStruct.hpp>
#include <stdair/bom/CancellationStruct.hpp>
#include <stdair/service/Logger.hpp>
#include <airinv/AIRINV_Master_Service.hpp>
#include <simcrs/command/DistributionManager.hpp>
```

Namespaces

- namespace SIMCRS

25.28 DistributionManager.cpp

```
00001 // //////////////////////////////////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // StdAir
00007 #include <stdair/bom/FareOptionStruct.hpp>
00008 #include <stdair/bom/TravelSolutionStruct.hpp>
00009 #include <stdair/bom/CancellationStruct.hpp>
00010 #include <stdair/service/Logger.hpp>
00011 // Airline Inventory
00012 #include <airinv/AIRINV_Master_Service.hpp>
00013 // SimCRS
00014 #include <simcrs/command/DistributionManager.hpp>
00015
00016 namespace SIMCRS {
00017
00018   // //////////////////////////////////////////////////////////////////
00019   void DistributionManager::
00020   calculateAvailability (AIRINV::AIRINV_Master_Service& ioAIRINV_Master_Service,
00021                          stdair::TravelSolutionList_T& ioTravelSolutionList) {
00022     for (stdair::TravelSolutionList_T::iterator itTS =
00023            ioTravelSolutionList.begin();
00024          itTS != ioTravelSolutionList.end(); ++itTS) {
00025       stdair::TravelSolutionStruct& lCurrentTravelSolution = *itTS;
00026
00027       // Forward the work to the dedicated service.
00028       ioAIRINV_Master_Service.calculateAvailability (lCurrentTravelSolution);
00029     }
00030   }
00031
00032   // //////////////////////////////////////////////////////////////////
00033   bool DistributionManager::
00034   sell (AIRINV::AIRINV_Master_Service& ioAIRINV_Master_Service,
00035         const stdair::TravelSolutionStruct& iTravelSolution,
00036         const stdair::NbOfSeats_T& iPartySize) {
00037     bool hasSaleBeenSuccessful = false;
00038
00039     const stdair::ClassObjectIDMapHolder_T& lClassObjectIDMapHolder =
00040       iTravelSolution.getClassObjectIDMapHolder();
00041     if (lClassObjectIDMapHolder.size() > 0) {
00042       const stdair::FareOptionStruct& lChosenFareOption =
00043         iTravelSolution.getChosenFareOption ();
00044       const stdair::ClassList_StringList_T& lClassPath =
00045         lChosenFareOption.getClassPath();
00046       stdair::ClassList_StringList_T::const_iterator itClassKeyList =
00047         lClassPath.begin();
00048       for (stdair::ClassObjectIDMapHolder_T::const_iterator itClassObjectIDMap =
00049              lClassObjectIDMapHolder.begin();
00050            itClassObjectIDMap != lClassObjectIDMapHolder.end();
00051            ++itClassObjectIDMap, ++itClassKeyList) {
00052         const stdair::ClassObjectIDMap_T& lClassObjectIDMap =
00053           *itClassObjectIDMap;
00054
00055         // TODO: Removed this hardcode.
00056         std::ostringstream ostr;
00057         const stdair::ClassList_String_T& lClassList = *itClassKeyList;
00058         assert (lClassList.size() > 0);
00059         ostr << lClassList.at(0);
00060         const stdair::ClassCode_T lClassCode (ostr.str());
00061         stdair::ClassObjectIDMap_T::const_iterator itClassID =
00062           lClassObjectIDMap.find (lClassCode);
00063         assert (itClassID != lClassObjectIDMap.end());
00064         const stdair::BookingClassID_T& lClassID = itClassID->second;
00065
```

```
00066         hasSaleBeenSuccessful =
00067           ioAIRINV_Master_Service.sell (lClassID, iPartySize);
00068       }
00069     } else {
00070       const stdair::KeyList_T& lSegmentDateKeyList =
00071         iTravelSolution.getSegmentPath();
00072       const stdair::FareOptionStruct& lChosenFareOption =
00073         iTravelSolution.getChosenFareOption ();
00074       const stdair::ClassList_StringList_T& lClassPath =
00075         lChosenFareOption.getClassPath();
00076       stdair::ClassList_StringList_T::const_iterator itClassKeyList =
00077         lClassPath.begin();
00078       for (stdair::KeyList_T::const_iterator itKey= lSegmentDateKeyList.begin();
00079            itKey != lSegmentDateKeyList.end(); ++itKey, ++itClassKeyList) {
00080         const std::string& lSegmentDateKey = *itKey;
00081
00082         // TODO: Removed this hardcode.
00083         std::ostringstream ostr;
00084         const stdair::ClassList_String_T& lClassList = *itClassKeyList;
00085         assert (lClassList.size() > 0);
00086         ostr << lClassList.at(0);
00087         const stdair::ClassCode_T lClassCode (ostr.str());
00088
00089         hasSaleBeenSuccessful =
00090           ioAIRINV_Master_Service.sell (lSegmentDateKey, lClassCode,
00091                                         iPartySize);
00092       }
00093     }
00094
00095     return hasSaleBeenSuccessful;
00096   }
00097
00098   // ////////////////////////////////////////////////////////////////////
00099   bool DistributionManager::
00100   playCancellation (AIRINV::AIRINV_Master_Service& ioAIRINV_Master_Service,
00101                     const stdair::CancellationStruct& iCancellation) {
00102     bool hasCancellationBeenSuccessful = false;
00103
00104     const stdair::PartySize_T& lPartySize = iCancellation.getPartySize();
00105     const stdair::BookingClassIDList_T& lClassIDList =
00106       iCancellation.getClassIDList();
00107
00108     for (stdair::BookingClassIDList_T::const_iterator itClassID =
00109            lClassIDList.begin(); itClassID != lClassIDList.end(); ++itClassID) {
00110       const stdair::BookingClassID_T& lClassID = *itClassID;
00111
00112       hasCancellationBeenSuccessful =
00113         ioAIRINV_Master_Service.cancel (lClassID, lPartySize);
00114     }
00115     return hasCancellationBeenSuccessful;
00116   }
00117
00118 }
```

25.29 simcrs/command/DistributionManager.hpp File Reference

```
#include <stdair/stdair_basic_types.hpp>
#include <stdair/bom/TravelSolutionTypes.hpp>
#include <airinv/AIRINV_Types.hpp>
#include <simcrs/SIMCRS_Types.hpp>
```

Classes

- class SIMCRS::DistributionManager

  Command wrapping the travel distribution (CRS/GDS) process.

Namespaces

- namespace stdair

  Forward declarations.

- namespace AIRINV
- namespace SIMCRS

25.30 DistributionManager.hpp

```
00001 #ifndef __SIMCRS_CMD_DISTRIBUTIONMANAGER_HPP
00002 #define __SIMCRS_CMD_DISTRIBUTIONMANAGER_HPP
00003
00004 // //////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/bom/TravelSolutionTypes.hpp>
00010 // Airinv
00011 #include <airinv/AIRINV_Types.hpp>
00012 // Simcrs
00013 #include <simcrs/SIMCRS_Types.hpp>
00014
00015 // Forward declarations
00016 namespace stdair {
00017   struct TravelSolutionStruct;
00018   struct CancellationStruct;
00019 }
00020
00021 namespace AIRINV {
00022   class AIRINV_Master_Service;
00023 }
00024
00025 namespace SIMCRS {
00026
00030   class DistributionManager {
00031     friend class SIMCRS_Service;
00032   private:
00034     static void calculateAvailability (AIRINV::AIRINV_Master_Service&,
00035                                        stdair::TravelSolutionList_T&);
00036
00038     static bool sell (AIRINV::AIRINV_Master_Service&,
00039                       const stdair::TravelSolutionStruct&,
00040                       const stdair::NbOfSeats_T&);
00041
00043     static bool playCancellation (AIRINV::AIRINV_Master_Service&,
00044                                   const stdair::CancellationStruct&);
00045
00046   private:
00048     DistributionManager() {}
00049     DistributionManager(const DistributionManager&) {}
00051     ~DistributionManager() {}
00052   };
00053
00054 }
00055 #endif // __SIMCRS_CMD_DISTRIBUTIONMANAGER_HPP
```

25.31 simcrs/config/simcrs-paths.hpp.in File Reference

Defines

- #define PACKAGE "@PACKAGE@"
- #define PACKAGE_NAME "@PACKAGE_NAME@"
- #define PACKAGE_VERSION "@PACKAGE_VERSION@"
- #define PREFIXDIR "@prefix@"
- #define EXEC_PREFIX "@exec_prefix@"
- #define BINDIR "@bindir@"
- #define LIBDIR "@libdir@"
- #define LIBEXECDIR "@libexecdir@"
- #define SBINDIR "@sbindir@"
- #define SYSCONFDIR "@sysconfdir@"
- #define INCLUDEDIR "@includedir@"
- #define DATAROOTDIR "@datarootdir@"
- #define DATADIR "@datadir@"
- #define DOCDIR "@docdir@"
- #define MANDIR "@mandir@"
- #define INFODIR "@infodir@"
- #define HTMLDIR "@htmldir@"
- #define PDFDIR "@pdfdir@"
- #define STDAIR_SAMPLE_DIR "@sampledir@"

25.31.1 Define Documentation

25.31.1.1 #define PACKAGE "@PACKAGE@"

Definition at line 4 of file simcrs-paths.hpp.in.

25.31.1.2 #define PACKAGE_NAME "@PACKAGE_NAME@"

Definition at line 5 of file simcrs-paths.hpp.in.

Referenced by readConfiguration().

25.31.1.3 #define PACKAGE_VERSION "@PACKAGE_VERSION@"

Definition at line 6 of file simcrs-paths.hpp.in.

Referenced by readConfiguration().

25.31.1.4 #define PREFIXDIR "@prefix@"

Definition at line 7 of file simcrs-paths.hpp.in.

Referenced by readConfiguration().

25.31.1.5 #define EXEC_PREFIX "@exec_prefix@"

Definition at line 8 of file simcrs-paths.hpp.in.

25.31.1.6 #define BINDIR "@bindir@"

Definition at line 9 of file simcrs-paths.hpp.in.

25.31.1.7 #define LIBDIR "@libdir@"

Definition at line 10 of file simcrs-paths.hpp.in.

25.31.1.8 #define LIBEXECDIR "@libexecdir@"

Definition at line 11 of file simcrs-paths.hpp.in.

25.31.1.9 #define SBINDIR "@sbindir@"

Definition at line 12 of file simcrs-paths.hpp.in.

25.31.1.10 #define SYSCONFDIR "@sysconfdir@"

Definition at line 13 of file simcrs-paths.hpp.in.

25.31.1.11 #define INCLUDEDIR "@includedir@"

Definition at line 14 of file simcrs-paths.hpp.in.

25.31.1.12 #define DATAROOTDIR "@datarootdir@"

Definition at line 15 of file simcrs-paths.hpp.in.

25.31.1.13 #define DATADIR "@datadir@"

Definition at line 16 of file simcrs-paths.hpp.in.

25.31.1.14 #define DOCDIR "@docdir@"

Definition at line 17 of file simcrs-paths.hpp.in.

25.31.1.15 #define MANDIR "@mandir@"

Definition at line 18 of file simcrs-paths.hpp.in.

25.31.1.16 #define INFODIR "@infodir@"

Definition at line 19 of file simcrs-paths.hpp.in.

25.31.1.17 #define HTMLDIR "@htmldir@"

Definition at line 20 of file simcrs-paths.hpp.in.

25.31.1.18 #define PDFDIR "@pdfdir@"

Definition at line 21 of file simcrs-paths.hpp.in.

25.31.1.19 #define STDAIR_SAMPLE_DIR "@sampledir@"

Definition at line 22 of file simcrs-paths.hpp.in.

25.32 simcrs-paths.hpp.in

```
00001 #ifndef __SIMCRS_PATHS_HPP__
00002 #define __SIMCRS_PATHS_HPP__
00003
00004 #define PACKAGE "@PACKAGE@"
00005 #define PACKAGE_NAME "@PACKAGE_NAME@"
00006 #define PACKAGE_VERSION "@PACKAGE_VERSION@"
00007 #define PREFIXDIR "@prefix@"
00008 #define EXEC_PREFIX "@exec_prefix@"
00009 #define BINDIR "@bindir@"
00010 #define LIBDIR "@libdir@"
00011 #define LIBEXECDIR "@libexecdir@"
00012 #define SBINDIR "@sbindir@"
00013 #define SYSCONFDIR "@sysconfdir@"
00014 #define INCLUDEDIR "@includedir@"
00015 #define DATAROOTDIR "@datarootdir@"
00016 #define DATADIR "@datadir@"
00017 #define DOCDIR "@docdir@"
00018 #define MANDIR "@mandir@"
00019 #define INFODIR "@infodir@"
00020 #define HTMLDIR "@htmldir@"
00021 #define PDFDIR "@pdfdir@"
00022 #define STDAIR_SAMPLE_DIR "@sampledir@"
00023
00024 #endif // __SIMCRS_PATHS_HPP__
```

25.33 simcrs/factory/FacBomAbstract.cpp File Reference

```
#include <cassert>
#include <sstream>
#include <boost/functional/hash/hash.hpp>
#include <simcrs/bom/BomAbstract.hpp>
#include <simcrs/factory/FacBomAbstract.hpp>
```

Namespaces

- namespace SIMCRS

25.34 FacBomAbstract.cpp

```
00001 // //////////////////////////////////////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // Boost (STL Extension)
00008 #include <boost/functional/hash/hash.hpp>
00009 // Simcrs
00010 #include <simcrs/bom/BomAbstract.hpp>
00011 #include <simcrs/factory/FacBomAbstract.hpp>
00012
00013 namespace SIMCRS {
00014
00015   // //////////////////////////////////////////////////////////////////////
00016   FacBomAbstract::~FacBomAbstract() {
00017     clean ();
00018   }
00019
00020   // //////////////////////////////////////////////////////////////////////
00021   void FacBomAbstract::clean() {
00022     for (BomPool_T::iterator itBom = _pool.begin();
00023          itBom != _pool.end(); itBom++) {
00024       BomAbstract* currentBom_ptr = *itBom;
00025       assert (currentBom_ptr != NULL);
00026
00027       delete (currentBom_ptr); currentBom_ptr = NULL;
00028     }
00029
00030     // Empty the pool of Factories
00031     _pool.clear();
00032   }
00033
00034   // //////////////////////////////////////////////////////////////////////
00035   std::size_t FacBomAbstract::getID (const BomAbstract* iBomAbstract_ptr) {
00036     const void* lPtr = iBomAbstract_ptr;
00037     boost::hash<const void*> ptr_hash;
00038     const std::size_t lID = ptr_hash (lPtr);
00039     return lID;
00040   }
00041
00042   // //////////////////////////////////////////////////////////////////////
00043   std::size_t FacBomAbstract::getID (const BomAbstract& iBomAbstract) {
00044     return getID (&iBomAbstract);
00045   }
00046
00047   // //////////////////////////////////////////////////////////////////////
00048   std::string FacBomAbstract::getIDString(const BomAbstract* iBomAbstract_ptr) {
00049     const std::size_t lID = getID (iBomAbstract_ptr);
00050     std::ostringstream oStr;
00051     oStr << lID;
00052     return oStr.str();
00053   }
00054
00055   // //////////////////////////////////////////////////////////////////////
00056   std::string FacBomAbstract::getIDString (const BomAbstract& iBomAbstract) {
00057     return getIDString (&iBomAbstract);
00058   }
00059
00060 }
```

25.35 simcrs/factory/FacBomAbstract.hpp File Reference

```
#include <string>
#include <vector>
```

Classes

- class SIMCRS::FacBomAbstract

Namespaces

- namespace SIMCRS

25.36 FacBomAbstract.hpp

```
00001 #ifndef __SIMCRS_FAC_FACBOMABSTRACT_HPP
00002 #define __SIMCRS_FAC_FACBOMABSTRACT_HPP
00003
00004 // //////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////
00007 // STL
00008 #include <string>
00009 #include <vector>
00010
00011 namespace SIMCRS {
00012
00013   // Forward declarations
00014   class BomAbstract;
00015
00017   class FacBomAbstract {
00018     friend class FacSupervisor;
00019   public:
00020
00022     typedef std::vector<BomAbstract*> BomPool_T;
00023
00025     static std::size_t getID (const BomAbstract*);
00026
00028     static std::size_t getID (const BomAbstract&);
00029
00032     static std::string getIDString (const BomAbstract*);
00033
00036     static std::string getIDString (const BomAbstract&);
00037
00038   protected:
00041     FacBomAbstract() {}
00042     FacBomAbstract(const FacBomAbstract&) {}
00043
00045     virtual ~FacBomAbstract();
00046
00047   private:
00049     void clean();
00050
00051   protected:
00053     BomPool_T _pool;
00054   };
00055 }
00056 #endif // __SIMCRS_FAC_FACBOMABSTRACT_HPP
```

25.37 simcrs/factory/FacServiceAbstract.cpp File Reference

```
#include <cassert>
#include <simcrs/service/ServiceAbstract.hpp>
#include <simcrs/factory/FacServiceAbstract.hpp>
```

Namespaces

- namespace SIMCRS

25.38 FacServiceAbstract.cpp

```
00001 // ////////////////////////////////////////////////////////////////////
00002 // Import section
00003 // ////////////////////////////////////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // SIMCRS
00007 #include <simcrs/service/ServiceAbstract.hpp>
00008 #include <simcrs/factory/FacServiceAbstract.hpp>
00009
00010 namespace SIMCRS {
00011
00012   // ////////////////////////////////////////////////////////////////////
00013   FacServiceAbstract::~FacServiceAbstract() {
00014     clean ();
00015   }
00016
00017   // ////////////////////////////////////////////////////////////////////
00018   void FacServiceAbstract::clean() {
00019     for (ServicePool_T::iterator itService = _pool.begin();
00020          itService != _pool.end(); itService++) {
00021       ServiceAbstract* currentService_ptr = *itService;
00022       assert (currentService_ptr != NULL);
00023
00024       delete (currentService_ptr); currentService_ptr = NULL;
00025     }
00026
00027     // Empty the pool of Service Factories
00028     _pool.clear();
00029   }
00030
00031 }
```

25.39 simcrs/factory/FacServiceAbstract.hpp File Reference

```
#include <vector>
```

Classes

- class SIMCRS::FacServiceAbstract

Namespaces

- namespace SIMCRS

25.40 FacServiceAbstract.hpp

```
00001 #ifndef __SIMCRS_FAC_FACSERVICEABSTRACT_HPP
00002 #define __SIMCRS_FAC_FACSERVICEABSTRACT_HPP
00003
00004 // //////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////
00007 // STL
00008 #include <vector>
00009
00010 namespace SIMCRS {
00011
00012   // Forward declarations
00013   class ServiceAbstract;
00014
00016   class FacServiceAbstract {
00017   public:
00018
00020     typedef std::vector<ServiceAbstract*> ServicePool_T;
00021
00023     virtual ~FacServiceAbstract();
00024
00026     void clean();
00027
00028   protected:
00031     FacServiceAbstract() {}
00032
00034     ServicePool_T _pool;
00035   };
00036
00037 }
00038 #endif // __SIMCRS_FAC_FACSERVICEABSTRACT_HPP
```

25.41 simcrs/factory/FacSimcrsServiceContext.cpp File Reference

```
#include <cassert>
#include <simcrs/factory/FacSupervisor.hpp>
#include <simcrs/factory/FacSimcrsServiceContext.hpp>
#include <simcrs/service/SIMCRS_ServiceContext.hpp>
```

Namespaces

- namespace SIMCRS

25.42 FacSimcrsServiceContext.cpp

```
00001 // //////////////////////////////////////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // SIMCRS Common
00007 #include <simcrs/factory/FacSupervisor.hpp>
00008 #include <simcrs/factory/FacSimcrsServiceContext.hpp>
00009 #include <simcrs/service/SIMCRS_ServiceContext.hpp>
00010
00011 namespace SIMCRS {
00012
00013   FacSimcrsServiceContext* FacSimcrsServiceContext::_instance = NULL;
00014
00015   // //////////////////////////////////////////////////////////////////////
00016   FacSimcrsServiceContext::~FacSimcrsServiceContext () {
00017     _instance = NULL;
00018   }
00019
00020   // //////////////////////////////////////////////////////////////////////
00021   FacSimcrsServiceContext& FacSimcrsServiceContext::instance () {
00022
00023     if (_instance == NULL) {
00024       _instance = new FacSimcrsServiceContext();
00025       assert (_instance != NULL);
00026
00027       FacSupervisor::instance().registerServiceFactory (_instance);
00028     }
00029     return *_instance;
00030   }
00031
00032   // //////////////////////////////////////////////////////////////////////
00033   SIMCRS_ServiceContext& FacSimcrsServiceContext::
00034   create (const std::string& iTravelDatabaseName) {
00035     SIMCRS_ServiceContext* aSIMCRS_ServiceContext_ptr = NULL;
00036
00037     aSIMCRS_ServiceContext_ptr =
00038       new SIMCRS_ServiceContext (iTravelDatabaseName);
00039     assert (aSIMCRS_ServiceContext_ptr != NULL);
00040
00041     // The new object is added to the Bom pool
00042     _pool.push_back (aSIMCRS_ServiceContext_ptr);
00043
00044     return *aSIMCRS_ServiceContext_ptr;
00045   }
00046
00047 }
```

25.43 simcrs/factory/FacSimcrsServiceContext.hpp File Reference

```
#include <string>
#include <simcrs/factory/FacServiceAbstract.hpp>
```

Classes

- class SIMCRS::FacSimcrsServiceContext

Namespaces

- namespace SIMCRS

25.44 FacSimcrsServiceContext.hpp

```
00001 #ifndef __SIMCRS_FAC_FACSIMCRSSERVICECONTEXT_HPP
00002 #define __SIMCRS_FAC_FACSIMCRSSERVICECONTEXT_HPP
00003
00004 // //////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////
00007 // STL
00008 #include <string>
00009 // Simcrs
00010 #include <simcrs/factory/FacServiceAbstract.hpp>
00011
00012 namespace SIMCRS {
00013
00015   class SIMCRS_ServiceContext;
00016
00018   class FacSimcrsServiceContext : public FacServiceAbstract {
00019   public:
00020
00024     static FacSimcrsServiceContext& instance();
00025
00030     ~FacSimcrsServiceContext();
00031
00035     SIMCRS_ServiceContext& create (const std::string& iTravelDatabaseName);
00036
00037
00038   protected:
00042     FacSimcrsServiceContext () {}
00043
00044   private:
00046     static FacSimcrsServiceContext* _instance;
00047   };
00048
00049 }
00050 #endif // __SIMCRS_FAC_FACSIMCRSSERVICECONTEXT_HPP
```

25.45 simcrs/factory/FacSupervisor.cpp File Reference

```
#include <cassert>
#include <simcrs/factory/FacBomAbstract.hpp>
#include <simcrs/factory/FacServiceAbstract.hpp>
#include <simcrs/factory/FacSupervisor.hpp>
```

Namespaces

- namespace SIMCRS

25.46 FacSupervisor.cpp

```
00001 // //////////////////////////////////////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // SIMCRS
00007 #include <simcrs/factory/FacBomAbstract.hpp>
00008 #include <simcrs/factory/FacServiceAbstract.hpp>
00009 #include <simcrs/factory/FacSupervisor.hpp>
00010
00011 namespace SIMCRS {
00012
00013   FacSupervisor* FacSupervisor::_instance = NULL;
00014
00015   // //////////////////////////////////////////////////////////////////////
00016   FacSupervisor::FacSupervisor () {
00017   }
00018
00019   // //////////////////////////////////////////////////////////////////////
00020   FacSupervisor& FacSupervisor::instance() {
00021     if (_instance == NULL) {
00022       _instance = new FacSupervisor();
00023     }
00024
00025     return *_instance;
00026   }
00027
00028   // //////////////////////////////////////////////////////////////////////
00029   void FacSupervisor::
00030   registerBomFactory (FacBomAbstract* ioFacBomAbstract_ptr) {
00031     _bomPool.push_back (ioFacBomAbstract_ptr);
00032   }
00033
00034   // //////////////////////////////////////////////////////////////////////
00035   void FacSupervisor::
00036   registerServiceFactory (FacServiceAbstract* ioFacServiceAbstract_ptr) {
00037     _svcPool.push_back (ioFacServiceAbstract_ptr);
00038   }
00039
00040   // //////////////////////////////////////////////////////////////////////
00041   FacSupervisor::~FacSupervisor() {
00042     cleanBomLayer();
00043     cleanServiceLayer();
00044   }
00045
00046   // //////////////////////////////////////////////////////////////////////
00047   void FacSupervisor::cleanBomLayer() {
00048     for (BomFactoryPool_T::const_iterator itFactory = _bomPool.begin();
00049          itFactory != _bomPool.end(); itFactory++) {
00050       const FacBomAbstract* currentFactory_ptr = *itFactory;
00051       assert (currentFactory_ptr != NULL);
00052
00053       delete (currentFactory_ptr); currentFactory_ptr = NULL;
00054     }
00055
00056     // Empty the pool of Bom Factories
00057     _bomPool.clear();
00058   }
00059
00060   // //////////////////////////////////////////////////////////////////////
00061   void FacSupervisor::cleanServiceLayer() {
00062     for (ServiceFactoryPool_T::const_iterator itFactory = _svcPool.begin();
00063          itFactory != _svcPool.end(); itFactory++) {
00064       const FacServiceAbstract* currentFactory_ptr = *itFactory;
00065       assert (currentFactory_ptr != NULL);
```

```
00066
00067       delete (currentFactory_ptr); currentFactory_ptr = NULL;
00068     }
00069
00070     // Empty the pool of Service Factories
00071     _svcPool.clear();
00072   }
00073
00074   // ////////////////////////////////////////////////////////////////////
00075   void FacSupervisor::cleanFactory () {
00076         if (_instance != NULL) {
00077                 _instance->cleanBomLayer();
00078                 _instance->cleanServiceLayer();
00079         }
00080     delete (_instance); _instance = NULL;
00081   }
00082
00083 }
```

25.47 simcrs/factory/FacSupervisor.hpp File Reference

```
#include <vector>
```

Classes

- class SIMCRS::FacSupervisor

Namespaces

- namespace SIMCRS

25.48 FacSupervisor.hpp

```
00001 #ifndef __SIMCRS_FAC_FACSUPERVISOR_HPP
00002 #define __SIMCRS_FAC_FACSUPERVISOR_HPP
00003
00004 // //////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////
00007 // STL
00008 #include <vector>
00009
00010 namespace SIMCRS {
00011
00012   // Forward declarations
00013   class FacBomAbstract;
00014   class FacServiceAbstract;
00015
00017   class FacSupervisor {
00018   public:
00019
00021     typedef std::vector<FacBomAbstract*> BomFactoryPool_T;
00022       typedef std::vector<FacServiceAbstract*> ServiceFactoryPool_T;
00023
00027     static FacSupervisor& instance();
00028
00033     void registerBomFactory (FacBomAbstract*);
00034
00039     void registerServiceFactory (FacServiceAbstract*);
00040
00044     void cleanBomLayer();
00045
00049     void cleanServiceLayer();
00050
00053     static void cleanFactory ();
00054
00058     ~FacSupervisor();
00059
00060
00061   protected:
00065     FacSupervisor ();
00066     FacSupervisor (const FacSupervisor&) {}
00067
00068
00069   private:
00071     static FacSupervisor* _instance;
00072
00074     BomFactoryPool_T _bomPool;
00075
00077     ServiceFactoryPool_T _svcPool;
00078   };
00079 }
00080 #endif // __SIMCRS_FAC_FACSUPERVISOR_HPP
```

25.49 simcrs/service/ServiceAbstract.cpp File Reference

```
#include <simcrs/service/ServiceAbstract.hpp>
```

Namespaces

- namespace SIMCRS

25.50 ServiceAbstract.cpp

```
00001 // ////////////////////////////////////////////////////////////////////
00002 // Import section
00003 // ////////////////////////////////////////////////////////////////////
00004 // SIMCRS
00005 #include <simcrs/service/ServiceAbstract.hpp>
00006
00007 namespace SIMCRS {
00008
00009 }
```

25.51 simcrs/service/ServiceAbstract.hpp File Reference

```
#include <iosfwd>
```

Classes

- class SIMCRS::ServiceAbstract

Namespaces

- namespace SIMCRS

Functions

- template<class charT , class traits >
  std::basic_ostream< charT, traits > & operator<< (std::basic_ostream< charT, traits > &ioOut, const SIMCRS::ServiceAbstract &iService)
- template<class charT , class traits >
  std::basic_istream< charT, traits > & operator>> (std::basic_istream< charT, traits > &ioIn, SIMCRS::ServiceAbstract &ioService)

25.51.1 Function Documentation

25.51.1.1 template<class charT , class traits > std::basic_ostream<charT, traits>& operator<< (std::basic_ostream< charT, traits > & *ioOut*, const SIMCRS::ServiceAbstract & *iService*) [inline]

Piece of code given by Nicolai M. Josuttis, Section 13.12.1 "Implementing Output Operators" (p653) of his book "The C++ Standard Library: A Tutorial and Reference", published by Addison-Wesley.

Definition at line 42 of file ServiceAbstract.hpp.

25.51.1.2 template<class charT , class traits > std::basic_istream<charT, traits>& operator>> (std::basic_istream< charT, traits > & *ioIn*, SIMCRS::ServiceAbstract & *ioService*) [inline]

Piece of code given by Nicolai M. Josuttis, Section 13.12.1 "Implementing Output Operators" (pp655-657) of his book "The C++ Standard Library: A Tutorial and Reference", published by Addison-Wesley.

Definition at line 70 of file ServiceAbstract.hpp.

References SIMCRS::ServiceAbstract::fromStream().

25.52 ServiceAbstract.hpp

```
00001 #ifndef __SIMCRS_SVC_SERVICEABSTRACT_HPP
00002 #define __SIMCRS_SVC_SERVICEABSTRACT_HPP
00003
00004 // //////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////
00007 // STL
00008 #include <iosfwd>
00009 //#include <sstream>
00010
00011 namespace SIMCRS {
00012
00014   class ServiceAbstract {
00015   public:
00016
00018     virtual ~ServiceAbstract() {}
00019
00022     virtual void toStream (std::ostream& ioOut) const {}
00023
00026     virtual void fromStream (std::istream& ioIn) {}
00027
00028   protected:
00030     ServiceAbstract() {}
00031   };
00032 }
00033
00039 template <class charT, class traits>
00040 inline
00041 std::basic_ostream<charT, traits>&
00042 operator<< (std::basic_ostream<charT, traits>& ioOut,
00043             const SIMCRS::ServiceAbstract& iService) {
00049   std::basic_ostringstream<charT,traits> ostr;
00050   ostr.copyfmt (ioOut);
00051   ostr.width (0);
00052
00053   // Fill string stream
00054   iService.toStream (ostr);
00055
00056   // Print string stream
00057   ioOut << ostr.str();
00058
00059   return ioOut;
00060 }
00061
00067 template <class charT, class traits>
00068 inline
00069 std::basic_istream<charT, traits>&
00070 operator>> (std::basic_istream<charT, traits>& ioIn,
00071             SIMCRS::ServiceAbstract& ioService) {
00072   // Fill Service object with input stream
00073   ioService.fromStream (ioIn);
00074   return ioIn;
00075 }
00076
00077 #endif // __SIMCRS_SVC_SERVICEABSTRACT_HPP
```

25.53 simcrs/service/SIMCRS_Service.cpp File Reference

```
#include <cassert>
#include <sstream>
#include <boost/make_shared.hpp>
#include <stdair/stdair_exceptions.hpp>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_json.hpp>
#include <stdair/basic/BasChronometer.hpp>
#include <stdair/basic/BasFileMgr.hpp>
#include <stdair/bom/BomManager.hpp>
#include <stdair/bom/BookingRequestStruct.hpp>
#include <stdair/bom/TravelSolutionStruct.hpp>
#include <stdair/bom/CancellationStruct.hpp>
#include <stdair/bom/BomRoot.hpp>
#include <stdair/bom/Inventory.hpp>
#include <stdair/service/Logger.hpp>
#include <stdair/STDAIR_Service.hpp>
#include <sevmgr/SEVMGR_Service.hpp>
#include <airinv/AIRINV_Master_Service.hpp>
#include <airtsp/AIRTSP_Service.hpp>
#include <simfqt/SIMFQT_Service.hpp>
#include <simcrs/basic/BasConst_SIMCRS_Service.hpp>
#include <simcrs/command/DistributionManager.hpp>
#include <simcrs/factory/FacSimcrsServiceContext.hpp>
#include <simcrs/service/SIMCRS_ServiceContext.hpp>
#include <simcrs/SIMCRS_Service.hpp>
```

Namespaces

- namespace SIMCRS

25.54 SIMCRS_Service.cpp

```
00001 // //////////////////////////////////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 #include <sstream>
00007 // Boost
00008 #include <boost/make_shared.hpp>
00009 // Standard Airline Object Model
00010 #include <stdair/stdair_exceptions.hpp>
00011 #include <stdair/stdair_basic_types.hpp>
00012 #include <stdair/stdair_json.hpp>
00013 #include <stdair/basic/BasChronometer.hpp>
00014 #include <stdair/basic/BasFileMgr.hpp>
00015 #include <stdair/bom/BomManager.hpp>
00016 #include <stdair/bom/BookingRequestStruct.hpp>
00017 #include <stdair/bom/TravelSolutionStruct.hpp>
00018 #include <stdair/bom/CancellationStruct.hpp>
00019 #include <stdair/bom/BomRoot.hpp>
00020 #include <stdair/bom/Inventory.hpp>
00021 #include <stdair/service/Logger.hpp>
00022 #include <stdair/STDAIR_Service.hpp>
00023 // SEvMgr
00024 #include <sevmgr/SEVMGR_Service.hpp>
00025 // Airline Inventory
00026 #include <airinv/AIRINV_Master_Service.hpp>
00027 // Airline Schedule
00028 #include <airtsp/AIRTSP_Service.hpp>
00029 // Fare Quote
00030 #include <simfqt/SIMFQT_Service.hpp>
00031 // SimCRS
00032 #include <simcrs/basic/BasConst_SIMCRS_Service.hpp>
00033 #include <simcrs/command/DistributionManager.hpp>
00034 #include <simcrs/factory/FacSimcrsServiceContext.hpp>
00035 #include <simcrs/service/SIMCRS_ServiceContext.hpp>
00036 #include <simcrs/SIMCRS_Service.hpp>
00037
00038 namespace SIMCRS {
00039
00040   // //////////////////////////////////////////////////////////////////
00041   SIMCRS_Service::SIMCRS_Service() : _simcrsServiceContext (NULL) {
00042     assert (false);
00043   }
00044
00045   // //////////////////////////////////////////////////////////////////
00046   SIMCRS_Service::SIMCRS_Service (const SIMCRS_Service& iService) {
00047     assert (false);
00048   }
00049
00050   // //////////////////////////////////////////////////////////////////
00051   SIMCRS_Service::SIMCRS_Service (const stdair::BasLogParams& iLogParams,
00052                                   const CRSCode_T& iCRSCode)
00053     : _simcrsServiceContext (NULL) {
00054
00055     // Initialise the StdAir service handler
00056     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00057       initStdAirService (iLogParams);
00058
00059     // Initialise the service context
00060     initServiceContext (iCRSCode);
00061
00062     // Add the StdAir service context to the SimCRS service context
00063     // \note SIMCRS owns the STDAIR service resources here.
00064     const bool ownStdairService = true;
00065     addStdAirService (lSTDAIR_Service_ptr, ownStdairService);
```

```
00066
00067     // Initalise the SimFQT service.
00068     initSIMFQTService();
00069
00070     // Initalise the AirTSP service.
00071     initAIRTSPService();
00072
00073     // Initalise the AirInv service.
00074     initAIRINVService();
00075
00076     // Initialise the (remaining of the) context
00077     initSimcrsService();
00078   }
00079
00080   // ////////////////////////////////////////////////////////////////////
00081   SIMCRS_Service::SIMCRS_Service (const stdair::BasLogParams& iLogParams,
00082                                   const stdair::BasDBParams& iDBParams,
00083                                   const CRSCode_T& iCRSCode)
00084     : _simcrsServiceContext (NULL) {
00085
00086     // Initialise the STDAIR service handler
00087     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00088       initStdAirService (iLogParams, iDBParams);
00089
00090     // Initialise the service context
00091     initServiceContext (iCRSCode);
00092
00093     // Add the StdAir service context to the SIMCRS service context
00094     // \note SIMCRS owns the STDAIR service resources here.
00095     const bool ownStdairService = true;
00096     addStdAirService (lSTDAIR_Service_ptr, ownStdairService);
00097
00098     // Initalise the SIMFQT service.
00099     initSIMFQTService();
00100
00101     // Initalise the AIRTSP service.
00102     initAIRTSPService();
00103
00104     // Initalise the AIRINV service.
00105     initAIRINVService();
00106
00107     // Initialise the (remaining of the) context
00108     initSimcrsService();
00109   }
00110
00111   // ////////////////////////////////////////////////////////////////////
00112   SIMCRS_Service::
00113   SIMCRS_Service (stdair::STDAIR_ServicePtr_T ioSTDAIR_Service_ptr,
00114                   SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_Service_ptr,
00115                   const CRSCode_T& iCRSCode)
00116     : _simcrsServiceContext (NULL) {
00117
00118     // Initialise the service context
00119     initServiceContext (iCRSCode);
00120
00121     // Store the STDAIR service object within the (AIRINV) service context
00122     // \note AirInv does not own the STDAIR service resources here.
00123     const bool doesNotOwnStdairService = false;
00124     addStdAirService (ioSTDAIR_Service_ptr, doesNotOwnStdairService);
00125
00126     //Add the SEvMgr service to the TRADEMGEN service context.
00127     const bool doesNotOwnSEVMGRService = false;
00128     addSEVMGRService (ioSEVMGR_Service_ptr, doesNotOwnSEVMGRService);
00129
00130     // Initalise the SIMFQT service.
00131     initSIMFQTService();
00132
```

```
00133     // Initalise the AIRTSP service.
00134     initAIRTSPService();
00135
00136     // Initalise the AIRINV service.
00137     initAIRINVService();
00138
00139     // Initialise the (remaining of the) context
00140     initSimcrsService();
00141   }
00142
00143   // ////////////////////////////////////////////////////////////////////
00144   SIMCRS_Service::~SIMCRS_Service() {
00145     // Delete/Clean all the objects from memory
00146     finalise();
00147   }
00148
00149   // ////////////////////////////////////////////////////////////////////
00150   void SIMCRS_Service::finalise() {
00151     assert (_simcrsServiceContext != NULL);
00152     // Reset the (Boost.)Smart pointer pointing on the STDAIR_Service object.
00153     _simcrsServiceContext->reset();
00154   }
00155
00156   // ////////////////////////////////////////////////////////////////////
00157   void SIMCRS_Service::initServiceContext (const CRSCode_T& iCRSCode) {
00158     // Initialise the service context
00159     SIMCRS_ServiceContext& lSIMCRS_ServiceContext =
00160       FacSimcrsServiceContext::instance().create (iCRSCode);
00161     _simcrsServiceContext = &lSIMCRS_ServiceContext;
00162   }
00163
00164   // ////////////////////////////////////////////////////////////////////
00165   void SIMCRS_Service::
00166   addStdAirService (stdair::STDAIR_ServicePtr_T ioSTDAIR_Service_ptr,
00167                     const bool iOwnStdairService) {
00168
00169     // Retrieve the SimCRS service context
00170     assert (_simcrsServiceContext != NULL);
00171     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00172
00173     // Store the StdAir service object within the (SimCRS) service context
00174     lSIMCRS_ServiceContext.setSTDAIR_Service (ioSTDAIR_Service_ptr,
00175                                               iOwnStdairService);
00176   }
00177
00178   // ////////////////////////////////////////////////////////////////////
00179   void SIMCRS_Service::
00180   addSEVMGRService (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_Service_ptr,
00181                     const bool iOwnSEVMGRService) {
00182
00183     // Retrieve the SimCRS service context
00184     assert (_simcrsServiceContext != NULL);
00185     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00186
00187     // Store the STDAIR service object within the (TRADEMGEN) service context
00188     lSIMCRS_ServiceContext.setSEVMGR_Service (ioSEVMGR_Service_ptr,
00189                                               iOwnSEVMGRService);
00190   }
00191
00192   // ////////////////////////////////////////////////////////////////////
00193   stdair::STDAIR_ServicePtr_T SIMCRS_Service::
00194   initStdAirService (const stdair::BasLogParams& iLogParams) {
00195
00203     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00204       boost::make_shared<stdair::STDAIR_Service> (iLogParams);
00205
00206     return lSTDAIR_Service_ptr;
```

```
00207   }
00208
00209   // ////////////////////////////////////////////////////////////////////
00210   stdair::STDAIR_ServicePtr_T SIMCRS_Service::
00211   initStdAirService (const stdair::BasLogParams& iLogParams,
00212                      const stdair::BasDBParams& iDBParams) {
00213
00221     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00222       boost::make_shared<stdair::STDAIR_Service> (iLogParams, iDBParams);
00223
00224     return lSTDAIR_Service_ptr;
00225   }
00226
00227   // ////////////////////////////////////////////////////////////////////
00228   void SIMCRS_Service::initAIRTSPService() {
00229
00230     // Retrieve the SimCRS service context
00231     assert (_simcrsServiceContext != NULL);
00232     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00233
00234     // Retrieve the StdAir service context
00235     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00236       lSIMCRS_ServiceContext.getSTDAIR_ServicePtr();
00237
00245     AIRTSP::AIRTSP_ServicePtr_T lAIRTSP_Service_ptr =
00246       boost::make_shared<AIRTSP::AIRTSP_Service> (lSTDAIR_Service_ptr);
00247
00248     // Store the AIRTSP service object within the (SimCRS) service context
00249     lSIMCRS_ServiceContext.setAIRTSP_Service (lAIRTSP_Service_ptr);
00250   }
00251
00252   // ////////////////////////////////////////////////////////////////////
00253   void SIMCRS_Service::initSIMFQTService() {
00254
00255     // Retrieve the SimCRS service context
00256     assert (_simcrsServiceContext != NULL);
00257     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00258
00259     // Retrieve the StdAir service context
00260     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00261       lSIMCRS_ServiceContext.getSTDAIR_ServicePtr();
00262
00270     SIMFQT::SIMFQT_ServicePtr_T lSIMFQT_Service_ptr =
00271       boost::make_shared<SIMFQT::SIMFQT_Service> (lSTDAIR_Service_ptr);
00272
00273     // Store the SIMFQT service object within the (SimCRS) service context
00274     lSIMCRS_ServiceContext.setSIMFQT_Service (lSIMFQT_Service_ptr);
00275   }
00276
00277   // ////////////////////////////////////////////////////////////////////
00278   void SIMCRS_Service::initAIRINVService() {
00279
00280     // Retrieve the SimCRS service context
00281     assert (_simcrsServiceContext != NULL);
00282     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00283
00284     // Retrieve the StdAir service context
00285     stdair::STDAIR_ServicePtr_T lSTDAIR_Service_ptr =
00286       lSIMCRS_ServiceContext.getSTDAIR_ServicePtr();
00287
00295     AIRINV::AIRINV_Master_ServicePtr_T lAIRINV_Master_Service_ptr;
00296     const bool ownSEVMGRService =
00297       lSIMCRS_ServiceContext.getOwnSEVMGRServiceFlag();
00298     if (ownSEVMGRService == false) {
00299       // Retrieve the SEVMGR service
00300       SEVMGR::SEVMGR_ServicePtr_T lSEVMGR_Service_ptr =
00301         lSIMCRS_ServiceContext.getSEVMGR_ServicePtr();
```

```
00302       assert (lSEVMGR_Service_ptr != NULL);
00303       lAIRINV_Master_Service_ptr =
00304         boost::make_shared<AIRINV::AIRINV_Master_Service> (lSTDAIR_Service_ptr,
00305                                                            lSEVMGR_Service_ptr);
00306     } else {
00307       lAIRINV_Master_Service_ptr =
00308         boost::make_shared<AIRINV::AIRINV_Master_Service> (lSTDAIR_Service_ptr);
00309     }
00310     assert (lAIRINV_Master_Service_ptr != NULL);
00311
00312     // Store the AIRINV service object within the (SimCRS) service context
00313     lSIMCRS_ServiceContext.setAIRINV_Service (lAIRINV_Master_Service_ptr);
00314   }
00315
00316   // //////////////////////////////////////////////////////////////////
00317   void SIMCRS_Service::initSimcrsService() {
00318     // Do nothing at this stage. A sample BOM tree may be built by
00319     // calling the buildSampleBom() method
00320   }
00321
00322   // //////////////////////////////////////////////////////////////////
00323   void SIMCRS_Service::
00324   parseAndLoad (const stdair::ScheduleFilePath& iScheduleInputFilepath,
00325                 const stdair::ODFilePath& iODInputFilepath,
00326                 const stdair::FRAT5FilePath& iFRAT5InputFilepath,
00327                 const stdair::FFDisutilityFilePath& iFFDisutilityInputFilepath,
00328                 const AIRRAC::YieldFilePath& iYieldInputFilepath,
00329                 const SIMFQT::FareFilePath& iFareInputFilepath) {
00330
00331     // Retrieve the SimCRS service context
00332     if (_simcrsServiceContext == NULL) {
00333       throw stdair::NonInitialisedServiceException ("The SimCRS service "
00334                                                     "has not been initialised");
00335     }
00336     assert (_simcrsServiceContext != NULL);
00337
00338     // Retrieve the SimCRS service context and whether it owns the Stdair
00339     // service
00340     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00341     const bool doesOwnStdairService =
00342       lSIMCRS_ServiceContext.getOwnStdairServiceFlag();
00343
00344     // Retrieve the StdAir service object from the (SimCRS) service context
00345     stdair::STDAIR_Service& lSTDAIR_Service =
00346       lSIMCRS_ServiceContext.getSTDAIR_Service();
00347
00348     // Retrieve the persistent BOM root object.
00349     stdair::BomRoot& lPersistentBomRoot =
00350       lSTDAIR_Service.getPersistentBomRoot();
00351
00361     AIRTSP::AIRTSP_Service& lAIRTSP_Service =
00362       lSIMCRS_ServiceContext.getAIRTSP_Service();
00363     lAIRTSP_Service.parseAndLoad (iScheduleInputFilepath);
00364
00371     AIRINV::AIRINV_Master_Service& lAIRINV_Service =
00372       lSIMCRS_ServiceContext.getAIRINV_Service();
00373     lAIRINV_Service.parseAndLoad (iScheduleInputFilepath, iODInputFilepath,
00374                                   iFRAT5InputFilepath,
00375                                   iFFDisutilityInputFilepath,
00376                                   iYieldInputFilepath);
00377
00381     SIMFQT::SIMFQT_Service& lSIMFQT_Service =
00382       lSIMCRS_ServiceContext.getSIMFQT_Service();
00383     lSIMFQT_Service.parseAndLoad (iFareInputFilepath);
00384
00389     buildComplementaryLinks (lPersistentBomRoot);
00390
```

```
00395     if (doesOwnStdairService == true) {
00396       //
00397       clonePersistentBom ();
00398     }
00399   }
00400
00401   // ////////////////////////////////////////////////////////////////////
00402   void SIMCRS_Service::buildSampleBom() {
00403
00404     // Retrieve the SimCRS service context
00405     if (_simcrsServiceContext == NULL) {
00406       throw stdair::NonInitialisedServiceException ("The SimCRS service "
00407                                                     "has not been initialised");
00408     }
00409     assert (_simcrsServiceContext != NULL);
00410
00411     // Retrieve the SimCRS service context and whether it owns the Stdair
00412     // service
00413     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00414     const bool doesOwnStdairService =
00415       lSIMCRS_ServiceContext.getOwnStdairServiceFlag();
00416
00417     // Retrieve the StdAir service object from the (SimCRS) service context
00418     stdair::STDAIR_Service& lSTDAIR_Service =
00419       lSIMCRS_ServiceContext.getSTDAIR_Service();
00420
00421     // Retrieve the persistent BOM root object.
00422     stdair::BomRoot& lPersistentBomRoot =
00423       lSTDAIR_Service.getPersistentBomRoot();
00424
00429     if (doesOwnStdairService == true) {
00430       //
00431       lSTDAIR_Service.buildSampleBom();
00432     }
00433
00443     AIRTSP::AIRTSP_Service& lAIRTSP_Service =
00444       lSIMCRS_ServiceContext.getAIRTSP_Service();
00445     lAIRTSP_Service.buildSampleBom();
00446
00453     AIRINV::AIRINV_Master_Service& lAIRINV_Service =
00454       lSIMCRS_ServiceContext.getAIRINV_Service();
00455     lAIRINV_Service.buildSampleBom();
00456
00460     SIMFQT::SIMFQT_Service& lSIMFQT_Service =
00461       lSIMCRS_ServiceContext.getSIMFQT_Service();
00462     lSIMFQT_Service.buildSampleBom();
00463
00468     buildComplementaryLinks (lPersistentBomRoot);
00469
00474     if (doesOwnStdairService == true) {
00475       //
00476       clonePersistentBom ();
00477     }
00478   }
00479
00480   // ////////////////////////////////////////////////////////////////////
00481   void SIMCRS_Service::clonePersistentBom () {
00482
00483     // Retrieve the SimCRS service context
00484     if (_simcrsServiceContext == NULL) {
00485       throw stdair::NonInitialisedServiceException ("The SimCRS service "
00486                                                     "has not been initialised");
00487     }
00488     assert (_simcrsServiceContext != NULL);
00489
00490     // Retrieve the SimCRS service context and whether it owns the Stdair
00491     // service
```

```
00492     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00493     const bool doesOwnStdairService =
00494       lSIMCRS_ServiceContext.getOwnStdairServiceFlag();
00495
00496     // Retrieve the StdAir service object from the (SimCRS) service context
00497     stdair::STDAIR_Service& lSTDAIR_Service =
00498       lSIMCRS_ServiceContext.getSTDAIR_Service();
00499
00504     if (doesOwnStdairService == true) {
00505       //
00506       lSTDAIR_Service.clonePersistentBom ();
00507     }
00508
00518     AIRTSP::AIRTSP_Service& lAIRTSP_Service =
00519       lSIMCRS_ServiceContext.getAIRTSP_Service();
00520     lAIRTSP_Service.clonePersistentBom ();
00521
00528     AIRINV::AIRINV_Master_Service& lAIRINV_Service =
00529       lSIMCRS_ServiceContext.getAIRINV_Service();
00530     lAIRINV_Service.clonePersistentBom ();
00531
00535     SIMFQT::SIMFQT_Service& lSIMFQT_Service =
00536       lSIMCRS_ServiceContext.getSIMFQT_Service();
00537     lSIMFQT_Service.clonePersistentBom ();
00538
00543     stdair::BomRoot& lBomRoot = lSTDAIR_Service.getBomRoot();
00544     buildComplementaryLinks (lBomRoot);
00545   }
00546
00547   // ////////////////////////////////////////////////////////////////////
00548   void SIMCRS_Service::buildComplementaryLinks (stdair::BomRoot& ioBomRoot) {
00549     // Currently, no more things to do by TravelCCM at that stage.
00550   }
00551
00552   // ////////////////////////////////////////////////////////////////////
00553   void SIMCRS_Service::
00554   buildSampleTravelSolutions(stdair::TravelSolutionList_T& ioTravelSolutionList){

00555
00556     // Retrieve the SimCRS service context
00557     if (_simcrsServiceContext == NULL) {
00558       throw stdair::NonInitialisedServiceException ("The SimCRS service "
00559                                                     "has not been initialised");
00560     }
00561     assert (_simcrsServiceContext != NULL);
00562
00563     // Retrieve the StdAir service object from the (SimCRS) service context
00564     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00565     stdair::STDAIR_Service& lSTDAIR_Service =
00566       lSIMCRS_ServiceContext.getSTDAIR_Service();
00567
00568     // Delegate the BOM building to the dedicated service
00569     lSTDAIR_Service.buildSampleTravelSolutions (ioTravelSolutionList);
00570   }
00571
00572   // ////////////////////////////////////////////////////////////////////
00573   stdair::BookingRequestStruct SIMCRS_Service::
00574   buildSampleBookingRequest (const bool isForCRS) {
00575
00576     // Retrieve the SimCRS service context
00577     if (_simcrsServiceContext == NULL) {
00578       throw stdair::NonInitialisedServiceException ("The SimCRS service "
00579                                                     "has not been initialised");
00580     }
00581     assert (_simcrsServiceContext != NULL);
00582
00583     // Retrieve the StdAir service object from the (SimCRS) service context
```

```
00584     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00585     stdair::STDAIR_Service& lSTDAIR_Service =
00586       lSIMCRS_ServiceContext.getSTDAIR_Service();
00587
00588     // Delegate the BOM building to the dedicated service
00589     return lSTDAIR_Service.buildSampleBookingRequest (isForCRS);
00590   }
00591
00592   // ////////////////////////////////////////////////////////////////////
00593   bool SIMCRS_Service::sell (const std::string& iSegmentDateKey,
00594                              const stdair::ClassCode_T& iClassCode,
00595                              const stdair::PartySize_T& iPartySize) {
00596
00597     // Retrieve the SimCRS service context
00598     if (_simcrsServiceContext == NULL) {
00599       throw stdair::NonInitialisedServiceException ("The SimCRS service "
00600                                                     "has not been initialised");
00601     }
00602     assert (_simcrsServiceContext != NULL);
00603     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00604
00605     // Retrieve the AIRINV Master service.
00606     AIRINV::AIRINV_Master_Service& lAIRINV_Master_Service =
00607       lSIMCRS_ServiceContext.getAIRINV_Service();
00608
00609     return lAIRINV_Master_Service.sell (iSegmentDateKey, iClassCode,
00610                                         iPartySize);
00611   }
00612
00613   // ////////////////////////////////////////////////////////////////////
00614   std::string SIMCRS_Service::
00615   jsonHandler (const stdair::JSONString& iJSONString) const {
00616
00617     // Retrieve the SimCRS service context
00618     if (_simcrsServiceContext == NULL) {
00619       throw stdair::NonInitialisedServiceException ("The SimCRS service "
00620                                                     "has not been initialised");
00621     }
00622     assert (_simcrsServiceContext != NULL);
00623     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00624
00625     // Retrieve the AIRINV Master service.
00626     AIRINV::AIRINV_Master_Service& lAIRINV_Master_Service =
00627       lSIMCRS_ServiceContext.getAIRINV_Service();
00628
00629     return lAIRINV_Master_Service.jsonHandler (iJSONString);
00630
00631   }
00632
00633   // ////////////////////////////////////////////////////////////////////
00634   void SIMCRS_Service::
00635   initSnapshotAndRMEvents (const stdair::Date_T& iStartDate,
00636                            const stdair::Date_T& iEndDate) {
00637
00638     // Retrieve the SimCRS service context
00639     if (_simcrsServiceContext == NULL) {
00640       throw stdair::NonInitialisedServiceException ("The SimCRS service has "
00641                                                     "not been initialised");
00642     }
00643     assert (_simcrsServiceContext != NULL);
00644     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00645
00646     // Retrieve the AIRINV Master service.
00647     AIRINV::AIRINV_Master_Service& lAIRINV_Master_Service =
00648       lSIMCRS_ServiceContext.getAIRINV_Service();
00649
00650     lAIRINV_Master_Service.initSnapshotAndRMEvents (iStartDate, iEndDate);
```

```
00651   }
00652
00653   // ////////////////////////////////////////////////////////////////////
00654   std::string SIMCRS_Service::csvDisplay() const {
00655
00656     // Retrieve the SimCRS service context
00657     if (_simcrsServiceContext == NULL) {
00658       throw stdair::NonInitialisedServiceException ("The SimCRS service "
00659                                                     "has not been initialised");
00660     }
00661     assert (_simcrsServiceContext != NULL);
00662
00663     // Retrieve the StdAir service object from the (SimCRS) service context
00664     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00665     stdair::STDAIR_Service& lSTDAIR_Service =
00666       lSIMCRS_ServiceContext.getSTDAIR_Service();
00667     const stdair::BomRoot& lBomRoot = lSTDAIR_Service.getBomRoot();
00668
00669     // Delegate the BOM building to the dedicated service
00670     return lSTDAIR_Service.csvDisplay(lBomRoot);
00671   }
00672
00673   // ////////////////////////////////////////////////////////////////////
00674   std::string SIMCRS_Service::
00675   csvDisplay (const stdair::TravelSolutionList_T& ioTravelSolutionList) const {
00676
00677     // Retrieve the SimCRS service context
00678     if (_simcrsServiceContext == NULL) {
00679       throw stdair::NonInitialisedServiceException ("The SimCRS service "
00680                                                     "has not been initialised");
00681     }
00682     assert (_simcrsServiceContext != NULL);
00683
00684     // Retrieve the StdAir service object from the (SimCRS) service context
00685     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00686     stdair::STDAIR_Service& lSTDAIR_Service =
00687       lSIMCRS_ServiceContext.getSTDAIR_Service();
00688
00689     // Delegate the BOM building to the dedicated service
00690     return lSTDAIR_Service.csvDisplay (ioTravelSolutionList);
00691   }
00692
00693   // ////////////////////////////////////////////////////////////////////
00694   std::string SIMCRS_Service::
00695   list (const stdair::AirlineCode_T& iAirlineCode,
00696         const stdair::FlightNumber_T& iFlightNumber) const {
00697
00698     // Retrieve the SimCRS service context
00699     if (_simcrsServiceContext == NULL) {
00700       throw stdair::NonInitialisedServiceException ("The SimCRS service has "
00701                                                     "not been initialised");
00702     }
00703     assert (_simcrsServiceContext != NULL);
00704     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00705
00706     // Retrieve the AIRINV Master service.
00707     AIRINV::AIRINV_Master_Service& lAIRINV_Master_Service =
00708       lSIMCRS_ServiceContext.getAIRINV_Service();
00709
00710     // Delegate the BOM display to the dedicated service
00711     return lAIRINV_Master_Service.list (iAirlineCode, iFlightNumber);
00712   }
00713
00714   // ////////////////////////////////////////////////////////////////////
00715   std::string SIMCRS_Service::
00716   csvDisplay (const stdair::AirlineCode_T& iAirlineCode,
00717               const stdair::FlightNumber_T& iFlightNumber,
```

```
00718             const stdair::Date_T& iDepartureDate) const {
00719
00720     // Retrieve the SimCRS service context
00721     if (_simcrsServiceContext == NULL) {
00722       throw stdair::NonInitialisedServiceException ("The SimCRS service has "
00723                                                     "not been initialised");
00724     }
00725     assert (_simcrsServiceContext != NULL);
00726     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00727
00728     // Retrieve the AIRINV Master service.
00729     AIRINV::AIRINV_Master_Service& lAIRINV_Master_Service =
00730       lSIMCRS_ServiceContext.getAIRINV_Service();
00731
00732     // Delegate the BOM display to the dedicated service
00733     return lAIRINV_Master_Service.csvDisplay (iAirlineCode, iFlightNumber,
00734                                               iDepartureDate);
00735   }
00736
00737   // ////////////////////////////////////////////////////////////////////
00738   stdair::TravelSolutionList_T SIMCRS_Service::
00739   calculateSegmentPathList(const stdair::BookingRequestStruct& iBookingRequest){
00740
00741     // Retrieve the SimCRS service context
00742     if (_simcrsServiceContext == NULL) {
00743       throw stdair::NonInitialisedServiceException ("The SimCRS service "
00744                                                     "has not been initialised");
00745     }
00746     assert (_simcrsServiceContext != NULL);
00747
00748     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00749
00750     stdair::TravelSolutionList_T oTravelSolutionList;
00751
00752     // Get a reference on the AIRTSP service handler
00753     AIRTSP::AIRTSP_Service& lAIRTSP_Service =
00754       lSIMCRS_ServiceContext.getAIRTSP_Service();
00755
00756     // Delegate the booking to the dedicated service
00757     stdair::BasChronometer lTravelSolutionRetrievingChronometer;
00758     lTravelSolutionRetrievingChronometer.start();
00759
00760     lAIRTSP_Service.buildSegmentPathList (oTravelSolutionList,
00761                                           iBookingRequest);
00762
00763     // DEBUG
00764     const double lSegmentPathRetrievingMeasure =
00765       lTravelSolutionRetrievingChronometer.elapsed();
00766     STDAIR_LOG_DEBUG ("Travel solution retrieving: "
00767                       << lSegmentPathRetrievingMeasure << " - "
00768                       << lSIMCRS_ServiceContext.display());
00769
00770     return oTravelSolutionList;
00771   }
00772
00773   // ////////////////////////////////////////////////////////////////////
00774   void SIMCRS_Service::
00775   fareQuote (const stdair::BookingRequestStruct& iBookingRequest,
00776              stdair::TravelSolutionList_T& ioTravelSolutionList) {
00777
00778     // Retrieve the SimCRS service context
00779     if (_simcrsServiceContext == NULL) {
00780       throw stdair::NonInitialisedServiceException ("The SimCRS service has "
00781                                                     "not been initialised");
00782     }
00783     assert (_simcrsServiceContext != NULL);
00784
```

```
00785     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00786
00787     // Get a reference on the SIMFQT service handler
00788     SIMFQT::SIMFQT_Service& lSIMFQT_Service =
00789       lSIMCRS_ServiceContext.getSIMFQT_Service();
00790
00791     // Delegate the action to the dedicated command
00792     stdair::BasChronometer lFareQuoteRetrievalChronometer;
00793     lFareQuoteRetrievalChronometer.start();
00794
00795     lSIMFQT_Service.quotePrices (iBookingRequest, ioTravelSolutionList);
00796
00797     // DEBUG
00798     const double lFareQuoteRetrievalMeasure =
00799       lFareQuoteRetrievalChronometer.elapsed();
00800     STDAIR_LOG_DEBUG ("Fare Quote retrieving: " << lFareQuoteRetrievalMeasure
00801                       << " - " << lSIMCRS_ServiceContext.display());
00802   }
00803
00804   // ////////////////////////////////////////////////////////////////////
00805   void SIMCRS_Service::
00806   calculateAvailability (stdair::TravelSolutionList_T& ioTravelSolutionList) {
00807
00808     // Retrieve the SimCRS service context
00809     if (_simcrsServiceContext == NULL) {
00810       throw stdair::NonInitialisedServiceException ("The SimCRS service has "
00811                                                     "not been initialised");
00812     }
00813     assert (_simcrsServiceContext != NULL);
00814
00815     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00816
00817     // Retrieve the CRS code
00818     //const CRSCode_T& lCRSCode = lSIMCRS_ServiceContext.getCRSCode();
00819
00820     // Retrieve the AIRINV Master service.
00821     AIRINV::AIRINV_Master_Service& lAIRINV_Master_Service =
00822       lSIMCRS_ServiceContext.getAIRINV_Service();
00823
00824     // Delegate the availability retrieval to the dedicated command
00825     stdair::BasChronometer lAvlChronometer;
00826     lAvlChronometer.start();
00827
00828     DistributionManager::calculateAvailability (lAIRINV_Master_Service,
00829                                                 ioTravelSolutionList);
00830
00831     // DEBUG
00832     const double lAvlMeasure = lAvlChronometer.elapsed();
00833     STDAIR_LOG_DEBUG ("Availability retrieval: " << lAvlMeasure << " - "
00834                       << lSIMCRS_ServiceContext.display());
00835   }
00836
00837   // ////////////////////////////////////////////////////////////////////
00838   bool SIMCRS_Service::
00839   sell (const stdair::TravelSolutionStruct& iTravelSolution,
00840         const stdair::PartySize_T& iPartySize) {
00841     bool hasSaleBeenSuccessful = false;
00842
00843     // Retrieve the SimCRS service context
00844     if (_simcrsServiceContext == NULL) {
00845       throw stdair::NonInitialisedServiceException ("The SimCRS service has "
00846                                                     "not been initialised");
00847     }
00848     assert (_simcrsServiceContext != NULL);
00849
00850     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00851
```

```
00852     // Retrieve the CRS code
00853     //const CRSCode_T& lCRSCode = lSIMCRS_ServiceContext.getCRSCode();
00854
00855     // Retrieve the AIRINV Master service.
00856     AIRINV::AIRINV_Master_Service& lAIRINV_Master_Service =
00857       lSIMCRS_ServiceContext.getAIRINV_Service();
00858
00859     // Delegate the booking to the dedicated command
00860     stdair::BasChronometer lSellChronometer;
00861     lSellChronometer.start();
00862
00863     hasSaleBeenSuccessful = DistributionManager::sell (lAIRINV_Master_Service,
00864                                                        iTravelSolution,
00865                                                        iPartySize);
00866
00867     // DEBUG
00868     STDAIR_LOG_DEBUG ("Made a sell of " << iPartySize
00869                       << " persons on the following travel solution: "
00870                       << iTravelSolution.describe()
00871                       << " with the chosen fare option: "
00872                       << iTravelSolution.getChosenFareOption().describe()
00873                       << ". Successful? " << hasSaleBeenSuccessful);
00874
00875     // DEBUG
00876     const double lSellMeasure = lSellChronometer.elapsed();
00877     STDAIR_LOG_DEBUG ("Booking sell: " << lSellMeasure << " - "
00878                       << lSIMCRS_ServiceContext.display());
00879
00880     return hasSaleBeenSuccessful;
00881   }
00882
00883
00884   // ////////////////////////////////////////////////////////////////////
00885   bool SIMCRS_Service::
00886   playCancellation (const stdair::CancellationStruct& iCancellation) {
00887     bool hasCancellationBeenSuccessful = false;
00888
00889     // Retrieve the SimCRS service context
00890     if (_simcrsServiceContext == NULL) {
00891       throw stdair::NonInitialisedServiceException ("The SimCRS service has "
00892                                                     "not been initialised");
00893     }
00894     assert (_simcrsServiceContext != NULL);
00895
00896     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00897
00898     // Retrieve the CRS code
00899     //const CRSCode_T& lCRSCode = lSIMCRS_ServiceContext.getCRSCode();
00900
00901     // Retrieve the AIRINV Master service.
00902     AIRINV::AIRINV_Master_Service& lAIRINV_Master_Service =
00903       lSIMCRS_ServiceContext.getAIRINV_Service();
00904
00905     // Delegate the booking to the dedicated command
00906     stdair::BasChronometer lCancellationChronometer;
00907     lCancellationChronometer.start();
00908
00909     hasCancellationBeenSuccessful =
00910       DistributionManager::playCancellation (lAIRINV_Master_Service,
00911                                              iCancellation);
00912
00913     // DEBUG
00914     STDAIR_LOG_DEBUG ("Made a cancellation of " << iCancellation.describe());
00915
00916     // DEBUG
00917     const double lCancellationMeasure = lCancellationChronometer.elapsed();
00918     STDAIR_LOG_DEBUG ("Booking cancellation: " << lCancellationMeasure << " - "
```

```
00919                       << lSIMCRS_ServiceContext.display());
00920
00921     return hasCancellationBeenSuccessful;
00922   }
00923
00924   // ////////////////////////////////////////////////////////////////////
00925   void SIMCRS_Service::takeSnapshots (const stdair::SnapshotStruct& iSnapshot) {
00926
00927     // Retrieve the SimCRS service context
00928     if (_simcrsServiceContext == NULL) {
00929       throw stdair::NonInitialisedServiceException ("The SimCRS service has "
00930                                                     "not been initialised");
00931     }
00932     assert (_simcrsServiceContext != NULL);
00933     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00934
00935     // Retrieve the AIRINV Master service.
00936     AIRINV::AIRINV_Master_Service& lAIRINV_Master_Service =
00937       lSIMCRS_ServiceContext.getAIRINV_Service();
00938
00939     lAIRINV_Master_Service.takeSnapshots (iSnapshot);
00940   }
00941
00942   // ////////////////////////////////////////////////////////////////////
00943   void SIMCRS_Service::
00944   optimise (const stdair::RMEventStruct& iRMEvent) {
00945
00946     // Retrieve the SimCRS service context
00947     if (_simcrsServiceContext == NULL) {
00948       throw stdair::NonInitialisedServiceException ("The SimCRS service has "
00949                                                     "not been initialised");
00950     }
00951     assert (_simcrsServiceContext != NULL);
00952     SIMCRS_ServiceContext& lSIMCRS_ServiceContext = *_simcrsServiceContext;
00953
00954     // Retrieve the AIRINV Master service.
00955     AIRINV::AIRINV_Master_Service& lAIRINV_Master_Service =
00956       lSIMCRS_ServiceContext.getAIRINV_Service();
00957
00958     lAIRINV_Master_Service.optimise (iRMEvent);
00959   }
00960 }
```

25.55 simcrs/service/SIMCRS_ServiceContext.cpp File Reference

```
#include <cassert>
#include <stdair/STDAIR_Service.hpp>
#include <stdair/service/Logger.hpp>
#include <simcrs/basic/BasConst_SIMCRS_Service.hpp>
#include <simcrs/service/SIMCRS_ServiceContext.hpp>
```

Namespaces

- namespace SIMCRS

25.56 SIMCRS_ServiceContext.cpp

```
00001 // //////////////////////////////////////////////////////////////////
00002 // Import section
00003 // //////////////////////////////////////////////////////////////////
00004 // STL
00005 #include <cassert>
00006 // Standard Airline Object Model
00007 #include <stdair/STDAIR_Service.hpp>
00008 #include <stdair/service/Logger.hpp>
00009 // SimCRS
00010 #include <simcrs/basic/BasConst_SIMCRS_Service.hpp>
00011 #include <simcrs/service/SIMCRS_ServiceContext.hpp>
00012
00013 namespace SIMCRS {
00014
00015   // //////////////////////////////////////////////////////////////////
00016   SIMCRS_ServiceContext::SIMCRS_ServiceContext ()
00017     : _ownStdairService (false), _ownSEVMGRService (true),
00018       _CRSCode (DEFAULT_CRS_CODE) {
00019   }
00020
00021   // //////////////////////////////////////////////////////////////////
00022   SIMCRS_ServiceContext::SIMCRS_ServiceContext (const SIMCRS_ServiceContext&)
00023     : _ownStdairService (false), _ownSEVMGRService (true) {
00024   }
00025
00026   // //////////////////////////////////////////////////////////////////
00027   SIMCRS_ServiceContext::SIMCRS_ServiceContext (const CRSCode_T& iCRSCode)
00028     : _ownSEVMGRService (true),_CRSCode (iCRSCode) {
00029   }
00030
00031   // //////////////////////////////////////////////////////////////////
00032   SIMCRS_ServiceContext::~SIMCRS_ServiceContext() {
00033   }
00034
00035   // //////////////////////////////////////////////////////////////////
00036   const std::string SIMCRS_ServiceContext::shortDisplay() const {
00037     std::ostringstream oStr;
00038     oStr << "SIMCRS_ServiceContext [" << _CRSCode
00039          << "] - Owns StdAir service: " << _ownStdairService;
00040     return oStr.str();
00041   }
00042
00043   // //////////////////////////////////////////////////////////////////
00044   const std::string SIMCRS_ServiceContext::display() const {
00045     std::ostringstream oStr;
00046     oStr << shortDisplay();
00047     return oStr.str();
00048   }
00049
00050   // //////////////////////////////////////////////////////////////////
00051   const std::string SIMCRS_ServiceContext::describe() const {
00052     return shortDisplay();
00053   }
00054
00055   // //////////////////////////////////////////////////////////////////
00056   void SIMCRS_ServiceContext::reset() {
00057
00058     // The shared_ptr<>::reset() method drops the refcount by one.
00059     // If the count result is dropping to zero, the resource pointed to
00060     // by the shared_ptr<> will be freed.
00061
00062     // Reset the StdAir shared pointer
00063     _stdairService.reset();
00064
00065     // Reset the SimFQT shared pointer
```

```
00066     _simfqtService.reset();
00067 
00068      // Reset the AirTSP shared pointer
00069     _airtspService.reset();
00070 
00071     // Reset the AirInv shared pointer
00072     _airinvService.reset();
00073 
00074     // Reset the SEvMgr shared pointer
00075     _sevmgrService.reset();
00076   }
00077 
00078 }
```

25.57 simcrs/service/SIMCRS_ServiceContext.hpp File Reference

```
#include <string>
#include <map>
#include <boost/shared_ptr.hpp>
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_service_types.hpp>
#include <sevmgr/SEVMGR_Types.hpp>
#include <airinv/AIRINV_Types.hpp>
#include <airtsp/AIRTSP_Types.hpp>
#include <simfqt/SIMFQT_Types.hpp>
#include <simcrs/SIMCRS_Types.hpp>
#include <simcrs/service/ServiceAbstract.hpp>
```

Classes

- class SIMCRS::SIMCRS_ServiceContext

  Class holding the context of the Simcrs services.

Namespaces

- namespace SIMCRS

25.58 SIMCRS_ServiceContext.hpp

```
00001 #ifndef __SIMCRS_SVC_SIMCRSSERVICECONTEXT_HPP
00002 #define __SIMCRS_SVC_SIMCRSSERVICECONTEXT_HPP
00003
00004 // //////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////
00007 // STL
00008 #include <string>
00009 #include <map>
00010 // Boost
00011 #include <boost/shared_ptr.hpp>
00012 // StdAir
00013 #include <stdair/stdair_basic_types.hpp>
00014 #include <stdair/stdair_service_types.hpp>
00015 // SEvMgr
00016 #include <sevmgr/SEVMGR_Types.hpp>
00017 // AirInv
00018 #include <airinv/AIRINV_Types.hpp>
00019 // AirTSP
00020 #include <airtsp/AIRTSP_Types.hpp>
00021 // SimFQT
00022 #include <simfqt/SIMFQT_Types.hpp>
00023 // SimCRS
00024 #include <simcrs/SIMCRS_Types.hpp>
00025 #include <simcrs/service/ServiceAbstract.hpp>
00026
00027 namespace SIMCRS {
00028
00032   class SIMCRS_ServiceContext : public ServiceAbstract {
00038     friend class SIMCRS_Service;
00039     friend class FacSimcrsServiceContext;
00040
00041   private:
00042     // ////////////////// Getters ////////////////////
00048     const CRSCode_T& getCRSCode() const {
00049       return _CRSCode;
00050     }
00051
00055     stdair::STDAIR_ServicePtr_T getSTDAIR_ServicePtr() const {
00056       return _stdairService;
00057     }
00058
00062     stdair::STDAIR_Service& getSTDAIR_Service() const {
00063       assert (_stdairService != NULL);
00064       return *_stdairService;
00065     }
00066
00070     const bool getOwnStdairServiceFlag() const {
00071       return _ownStdairService;
00072     }
00073
00077     SEVMGR::SEVMGR_ServicePtr_T getSEVMGR_ServicePtr() const {
00078       return _sevmgrService;
00079     }
00080
00084     SEVMGR::SEVMGR_Service& getSEVMGR_Service() const {
00085       assert (_sevmgrService != NULL);
00086       return *_sevmgrService;
00087     }
00088
00092     const bool getOwnSEVMGRServiceFlag() const {
00093       return _ownSEVMGRService;
00094     }
00095
00099     AIRINV::AIRINV_Master_Service& getAIRINV_Service() const {
```

```
00100       assert (_airinvService != NULL);
00101       return *_airinvService;
00102     }
00103
00107     AIRTSP::AIRTSP_Service& getAIRTSP_Service() const {
00108       assert (_airtspService != NULL);
00109       return *_airtspService;
00110     }
00111
00115     SIMFQT::SIMFQT_Service& getSIMFQT_Service() const {
00116       assert (_simfqtService != NULL);
00117       return *_simfqtService;
00118     }
00119
00120
00121   private:
00122     // ////////////////// Setters ////////////////////
00126     void setCRSCode (const CRSCode_T& iCRSCode) {
00127       _CRSCode = iCRSCode;
00128     }
00129
00133     void setSTDAIR_Service (stdair::STDAIR_ServicePtr_T ioSTDAIR_ServicePtr,
00134                             const bool iOwnStdairService) {
00135       _stdairService = ioSTDAIR_ServicePtr;
00136       _ownStdairService = iOwnStdairService;
00137     }
00138
00142     void setSEVMGR_Service (SEVMGR::SEVMGR_ServicePtr_T ioSEVMGR_ServicePtr,
00143                             const bool iOwnSEVMGRService) {
00144       _sevmgrService = ioSEVMGR_ServicePtr;
00145       _ownSEVMGRService = iOwnSEVMGRService;
00146     }
00147
00151     void setAIRINV_Service (AIRINV::AIRINV_Master_ServicePtr_T ioServicePtr) {
00152       _airinvService = ioServicePtr;
00153     }
00154
00158     void setAIRTSP_Service (AIRTSP::AIRTSP_ServicePtr_T ioServicePtr) {
00159       _airtspService = ioServicePtr;
00160     }
00161
00165     void setSIMFQT_Service (SIMFQT::SIMFQT_ServicePtr_T ioServicePtr) {
00166       _simfqtService = ioServicePtr;
00167     }
00168
00169
00170   private:
00171     // ////////////////////// Display Methods /////////////////////
00175     const std::string shortDisplay() const;
00176
00180     const std::string display() const;
00181
00185     const std::string describe() const;
00186
00187
00188   private:
00190
00193     SIMCRS_ServiceContext (const CRSCode_T& iCRSCode);
00197     SIMCRS_ServiceContext();
00201     SIMCRS_ServiceContext (const SIMCRS_ServiceContext&);
00202
00206     ~SIMCRS_ServiceContext();
00207
00211     void reset();
00212
00213
00214   private:
```

```
00218     stdair::STDAIR_ServicePtr_T _stdairService;
00219
00223     bool _ownStdairService;
00224
00228     SEVMGR::SEVMGR_ServicePtr_T _sevmgrService;
00229
00233     bool _ownSEVMGRService;
00234
00238     AIRTSP::AIRTSP_ServicePtr_T _airtspService;
00239
00243     AIRINV::AIRINV_Master_ServicePtr_T _airinvService;
00244
00248     SIMFQT::SIMFQT_ServicePtr_T _simfqtService;
00249
00250
00251   private:
00252     // ///////////////// Attributes //////////////////
00258     CRSCode_T _CRSCode;
00259   };
00260
00261 }
00262 #endif // __SIMCRS_SVC_SIMCRSSERVICECONTEXT_HPP
```

25.59 simcrs/SIMCRS_Service.hpp File Reference

```
#include <stdair/stdair_basic_types.hpp>
#include <stdair/stdair_file.hpp>
#include <stdair/stdair_service_types.hpp>
#include <stdair/bom/TravelSolutionTypes.hpp>
#include <sevmgr/SEVMGR_Types.hpp>
#include <simfqt/SIMFQT_Types.hpp>
#include <airrac/AIRRAC_Types.hpp>
#include <simcrs/SIMCRS_Types.hpp>
```

Classes

- class SIMCRS::SIMCRS_Service

Namespaces

- namespace stdair

  Forward declarations.

- namespace SIMCRS

25.60 SIMCRS_Service.hpp

```
00001 #ifndef __SIMCRS_SVC_SIMCRS_SERVICE_HPP
00002 #define __SIMCRS_SVC_SIMCRS_SERVICE_HPP
00003
00004 // //////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////
00007 // StdAir
00008 #include <stdair/stdair_basic_types.hpp>
00009 #include <stdair/stdair_file.hpp>
00010 #include <stdair/stdair_service_types.hpp>
00011 #include <stdair/bom/TravelSolutionTypes.hpp>
00012 // SEvMgr
00013 #include <sevmgr/SEVMGR_Types.hpp>
00014 // SimFQT
00015 #include <simfqt/SIMFQT_Types.hpp>
00016 // AIRRAC
00017 #include <airrac/AIRRAC_Types.hpp>
00018 // SimCRS
00019 #include <simcrs/SIMCRS_Types.hpp>
00020
00022 namespace stdair {
00023   class BomRoot;
00024   struct BasLogParams;
00025   struct BasDBParams;
00026   struct BookingRequestStruct;
00027   struct CancellationStruct;
00028   struct SnapshotStruct;
00029   struct RMEventStruct;
00030   class JSONString;
00031 }
00032
00033 namespace SIMCRS {
00034
00036   class SIMCRS_ServiceContext;
00037
00038
00042   class SIMCRS_Service {
00043   public:
00044     // ////////////////// Constructors and Destructors /////////////////
00061     SIMCRS_Service (const stdair::BasLogParams&, const stdair::BasDBParams&,
00062                     const CRSCode_T&);
00063
00076     SIMCRS_Service (const stdair::BasLogParams&, const CRSCode_T&);
00077
00096     SIMCRS_Service (stdair::STDAIR_ServicePtr_T, SEVMGR::SEVMGR_ServicePtr_T,
00097                     const CRSCode_T&);
00098
00099
00113     void parseAndLoad (const stdair::ScheduleFilePath&,
00114                        const stdair::ODFilePath&,
00115                        const stdair::FRAT5FilePath&,
00116                        const stdair::FFDisutilityFilePath&,
00117                        const AIRRAC::YieldFilePath&,
00118                        const SIMFQT::FareFilePath&);
00119
00126     void initSnapshotAndRMEvents (const stdair::Date_T& iStartDate,
00127                                   const stdair::Date_T& iEndDate);
00128
00132     ~SIMCRS_Service();
00133
00134
00135   public:
00136     // ///////////////// Business Methods /////////////////
00141     stdair::TravelSolutionList_T
00142     calculateSegmentPathList (const stdair::BookingRequestStruct&);
```

```
00143
00147     void fareQuote (const stdair::BookingRequestStruct&,
00148                     stdair::TravelSolutionList_T&);
00149
00153     void calculateAvailability (stdair::TravelSolutionList_T&);
00154
00158     bool sell (const stdair::TravelSolutionStruct&, const stdair::PartySize_T&);
00159
00163     void takeSnapshots (const stdair::SnapshotStruct&);
00164
00168     bool playCancellation (const stdair::CancellationStruct&);
00169
00173     void optimise (const stdair::RMEventStruct&);
00174
00183     bool sell (const std::string& iSegmentDateKey, const stdair::ClassCode_T&,
00184                const stdair::PartySize_T&);
00185
00195     void buildSampleBom ();
00196
00200     void clonePersistentBom ();
00201
00206     void buildComplementaryLinks (stdair::BomRoot&);
00207
00227     void buildSampleTravelSolutions (stdair::TravelSolutionList_T&);
00228
00259     stdair::BookingRequestStruct
00260     buildSampleBookingRequest (const bool isForCRS = false);
00261
00262
00263   public:
00264     // ///////////////// Export support methods /////////////////
00274     std::string jsonHandler (const stdair::JSONString&) const;
00275
00276   public:
00277     // ///////////////// Display support methods /////////////////
00285     std::string csvDisplay() const;
00286
00294     std::string csvDisplay (const stdair::TravelSolutionList_T&) const;
00295
00309     std::string list (const stdair::AirlineCode_T& iAirlineCode = "all",
00310                       const stdair::FlightNumber_T& iFlightNumber = 0) const;
00311
00323     std::string csvDisplay (const stdair::AirlineCode_T&,
00324                             const stdair::FlightNumber_T&,
00325                             const stdair::Date_T& iDepartureDate) const;
00326
00327
00328   private:
00329     // /////// Construction and Destruction helper methods ///////
00333     SIMCRS_Service();
00334
00338     SIMCRS_Service (const SIMCRS_Service&);
00339
00349     stdair::STDAIR_ServicePtr_T initStdAirService (const stdair::BasLogParams&,
00350                                                    const stdair::BasDBParams&);
00351
00361     stdair::STDAIR_ServicePtr_T initStdAirService (const stdair::BasLogParams&);
00362
00366     void initAIRTSPService();
00367
00371     void initSIMFQTService();
00372
00376     void initAIRINVService();
00377
00386     void addStdAirService (stdair::STDAIR_ServicePtr_T,
00387                            const bool iOwnStdairService);
00388
```

```
00394     void addSEVMGRService (SEVMGR::SEVMGR_ServicePtr_T,
00395                            const bool iOwnSEVMGRService);
00396
00403     void initServiceContext (const CRSCode_T&);
00404
00409     void initSimcrsService();
00410
00414     void finalise();
00415
00416
00417   private:
00418     // ///////// Service Context /////////
00422     SIMCRS_ServiceContext* _simcrsServiceContext;
00423   };
00424 }
00425 #endif // __SIMCRS_SVC_SIMCRS_SERVICE_HPP
```

25.61 simcrs/SIMCRS_Types.hpp File Reference

```
#include <exception>
#include <string>
#include <boost/shared_ptr.hpp>
#include <stdair/stdair_exceptions.hpp>
```

Classes

- class SIMCRS::BookingException
- class SIMCRS::AvailabilityRetrievalException

Namespaces

- namespace SIMCRS

Typedefs

- typedef std::string SIMCRS::CRSCode_T
- typedef boost::shared_ptr< SIMCRS_Service > SIMCRS::SIMCRS_ServicePtr_T

25.62 SIMCRS_Types.hpp

```
00001 #ifndef __SIMCRS_SIMCRS_TYPES_HPP
00002 #define __SIMCRS_SIMCRS_TYPES_HPP
00003
00004 // //////////////////////////////////////////////////////////////////
00005 // Import section
00006 // //////////////////////////////////////////////////////////////////
00007 // STL
00008 #include <exception>
00009 #include <string>
00010 // Boost
00011 #include <boost/shared_ptr.hpp>
00012 // StdAir
00013 #include <stdair/stdair_exceptions.hpp>
00014
00015 namespace SIMCRS {
00016
00017   // Forward declarations
00018   class SIMCRS_Service;
00019
00020
00021   // ********* Exceptions ***********
00025   class BookingException : public stdair::RootException {
00026   };
00027
00031   class AvailabilityRetrievalException : public stdair::RootException {
00032   };
00033
00034
00035   // ******** Type definitions specific to SimCRS *********
00039   typedef std::string CRSCode_T;
00040
00044   typedef boost::shared_ptr<SIMCRS_Service> SIMCRS_ServicePtr_T;
00045
00046 }
00047 #endif // __SIMCRS_SIMCRS_TYPES_HPP
00048
```

25.63 test/simcrs/CRSTestSuite.cpp File Reference

25.64 CRSTestSuite.cpp

```
00001
00005 // ////////////////////////////////////////////////////////////////////
00006 // Import section
00007 // ////////////////////////////////////////////////////////////////////
00008 // STL
00009 #include <sstream>
00010 #include <fstream>
00011 #include <string>
00012 #include <cmath>
00013 // Boost Unit Test Framework (UTF)
00014 #define BOOST_TEST_DYN_LINK
00015 #define BOOST_TEST_MAIN
00016 #define BOOST_TEST_MODULE CRSTestSuite
00017 #include <boost/test/unit_test.hpp>
00018 // StdAir
00019 #include <stdair/basic/BasLogParams.hpp>
00020 #include <stdair/basic/BasDBParams.hpp>
00021 #include <stdair/basic/BasFileMgr.hpp>
00022 #include <stdair/bom/TravelSolutionStruct.hpp>
00023 #include <stdair/bom/BookingRequestStruct.hpp>
00024 #include <stdair/service/Logger.hpp>
00025 // SimFQT
00026 #include <simfqt/SIMFQT_Types.hpp>
00027 // SimCRS
00028 #include <simcrs/SIMCRS_Service.hpp>
00029 #include <simcrs/config/simcrs-paths.hpp>
00030
00031 namespace boost_utf = boost::unit_test;
00032
00033 // (Boost) Unit Test XML Report
00034 std::ofstream utfReportStream ("CRSTestSuite_utfresults.xml");
00035
00039 struct UnitTestConfig {
00041   UnitTestConfig() {
00042     boost_utf::unit_test_log.set_stream (utfReportStream);
00043     boost_utf::unit_test_log.set_format (boost_utf::XML);
00044     boost_utf::unit_test_log.set_threshold_level (boost_utf::log_test_units);
00045     //boost_utf::unit_test_log.set_threshold_level (boost_utf::log_successful_tes
      ts);
00046   }
00047
00049   ~UnitTestConfig() {
00050   }
00051 };
00052
00053 // ////////////////////////////////////////////////////////////////////
00057 const unsigned int testSimCRSHelper (const unsigned short iTestFlag,
00058                                      const stdair::Filename_T& iScheduleInputFile
      name,
00059                                      const stdair::Filename_T& iOnDInputFilename,

00060                                      const stdair::Filename_T& iFRAT5InputFilenam
      e,
00061                                      const stdair::Filename_T& iFFDisutilityInput
      Filename,
00062                                      const stdair::Filename_T& iYieldInputFilenam
      e,
00063                                      const stdair::Filename_T& iFareInputFilename
      ,
00064                                      const bool isBuiltin,
00065                                      const unsigned int iExpectedNbOfTravelSoluti
      ons,
00066                                      const unsigned int iExpectedPrice) {
00067
00068   // CRS code
```

```
00069   const SIMCRS::CRSCode_T lCRSCode ("1P");
00070
00071   // Output log File
00072   std::ostringstream oStr;
00073   oStr << "CRSTestSuite_" << iTestFlag << ".log";
00074   const stdair::Filename_T lLogFilename (oStr.str());
00075
00076   // Set the log parameters
00077   std::ofstream logOutputFile;
00078   // Open and clean the log outputfile
00079   logOutputFile.open (lLogFilename.c_str());
00080   logOutputFile.clear();
00081
00082   // Initialise the list of classes/buckets
00083   const stdair::BasLogParams lLogParams (stdair::LOG::DEBUG, logOutputFile);
00084   SIMCRS::SIMCRS_Service simcrsService (lLogParams, lCRSCode);
00085
00086   stdair::Date_T lPreferredDepartureDate;;
00087   stdair::Date_T lRequestDate;
00088   stdair::TripType_T lTripType;
00089
00090   // Check wether or not a (CSV) input file should be read
00091   if (isBuiltin == true) {
00092
00093     // Build the default sample BOM tree
00094     simcrsService.buildSampleBom();
00095
00096     lPreferredDepartureDate = boost::gregorian::from_string ("2010/02/08");
00097     lRequestDate = boost::gregorian::from_string ("2010/01/21");
00098     lTripType = "OW";
00099
00100   } else {
00101
00102     // Build the BOM tree from parsing input files
00103     stdair::ScheduleFilePath lScheduleFilePath (iScheduleInputFilename);
00104     stdair::ODFilePath lODFilePath (iOnDInputFilename);
00105     stdair::FRAT5FilePath lFRAT5FilePath (iFRAT5InputFilename);
00106     stdair::FFDisutilityFilePath lFFDisutilityFilePath (iFFDisutilityInputFilenam
      e);
00107     const SIMFQT::FareFilePath lFareFilePath (iFareInputFilename);
00108     const AIRRAC::YieldFilePath lYieldFilePath (iYieldInputFilename);
00109     simcrsService.parseAndLoad (lScheduleFilePath, lODFilePath,
00110                                 lFRAT5FilePath, lFFDisutilityFilePath,
00111                                 lYieldFilePath, lFareFilePath);
00112
00113     lPreferredDepartureDate = boost::gregorian::from_string ("2011/01/31");
00114     lRequestDate = boost::gregorian::from_string ("2011/01/22");
00115     lTripType = "RI";
00116   }
00117
00118   // Create an empty booking request structure
00119   const stdair::AirportCode_T lOrigin ("SIN");
00120   const stdair::AirportCode_T lDestination ("BKK");
00121   const stdair::AirportCode_T lPOS ("SIN");
00122   const stdair::Duration_T lRequestTime (boost::posix_time::hours(10));
00123   const stdair::DateTime_T lRequestDateTime (lRequestDate, lRequestTime);
00124   const stdair::CabinCode_T lPreferredCabin ("Eco");
00125   const stdair::PartySize_T lPartySize (3);
00126   const stdair::ChannelLabel_T lChannel ("IN");
00127   const stdair::DayDuration_T lStayDuration (7);
00128   const stdair::FrequentFlyer_T lFrequentFlyerType ("M");
00129   const stdair::Duration_T lPreferredDepartureTime (boost::posix_time::hours(10))
      ;
00130   const stdair::WTP_T lWTP (1000.0);
00131   const stdair::PriceValue_T lValueOfTime (100.0);
00132   const stdair::ChangeFees_T lChangeFees (true);
00133   const stdair::Disutility_T lChangeFeeDisutility (50);
```

```
00134   const stdair::NonRefundable_T lNonRefundable (true);
00135   const stdair::Disutility_T lNonRefundableDisutility (50);
00136   const stdair::BookingRequestStruct lBookingRequest (lOrigin, lDestination,
00137                                                       lPOS,
00138                                                       lPreferredDepartureDate,
00139                                                       lRequestDateTime,
00140                                                       lPreferredCabin,
00141                                                       lPartySize, lChannel,
00142                                                       lTripType, lStayDuration,
00143                                                       lFrequentFlyerType,
00144                                                       lPreferredDepartureTime,
00145                                                       lWTP, lValueOfTime,
00146                                                       lChangeFees,
00147                                                       lChangeFeeDisutility,
00148                                                       lNonRefundable,
00149                                                       lNonRefundableDisutility);
00150   stdair::TravelSolutionList_T lTravelSolutionList =
00151     simcrsService.calculateSegmentPathList (lBookingRequest);
00152
00153   // Price the travel solution
00154   simcrsService.fareQuote (lBookingRequest, lTravelSolutionList);
00155
00156   //
00157   const unsigned int lNbOfTravelSolutions = lTravelSolutionList.size();
00158
00159   // DEBUG
00160   std::ostringstream oMessageKeptTS;
00161   oMessageKeptTS << "The number of travel solutions for the booking request '"
00162                  << lBookingRequest.describe() << "' is actually "
00163                  << lNbOfTravelSolutions << ". That number is expected to be "
00164                  << iExpectedNbOfTravelSolutions << ".";
00165   STDAIR_LOG_DEBUG (oMessageKeptTS.str());
00166
00167   BOOST_CHECK_EQUAL (lNbOfTravelSolutions, iExpectedNbOfTravelSolutions);
00168
00169   BOOST_CHECK_MESSAGE (lNbOfTravelSolutions == iExpectedNbOfTravelSolutions,
00170                        oMessageKeptTS.str());
00171
00176   stdair::TravelSolutionStruct& lTravelSolution = lTravelSolutionList.front();
00177
00182   const stdair::FareOptionList_T& lFareOptionList =
00183     lTravelSolution.getFareOptionList();
00184
00193   stdair::FareOptionStruct lFareOption = lFareOptionList.front();
00194   lTravelSolution.setChosenFareOption (lFareOption);
00195
00196   // DEBUG
00197   std::ostringstream oMessageKeptFare;
00198   oMessageKeptFare
00199     << "The price given by the fare quoter for the booking request: '"
00200     << lBookingRequest.describe() << "' and travel solution: '"
00201     << lTravelSolution.describe() << "' is actually " << lFareOption.getFare()
00202     << " Euros. It is expected to be " << iExpectedPrice << " Euros.";
00203   STDAIR_LOG_DEBUG (oMessageKeptFare.str());
00204
00205   BOOST_CHECK_EQUAL (std::floor (lFareOption.getFare() + 0.5), iExpectedPrice);
00206
00207   BOOST_CHECK_MESSAGE (std::floor (lFareOption.getFare() + 0.5)
00208                        == iExpectedPrice, oMessageKeptFare.str());
00209
00217   // DEBUG
00218   STDAIR_LOG_DEBUG ("A booking will now (attempted to) be made on the "
00219                     "travel solution '" << lTravelSolution.describe()
00220                     << "', for a party size of " << lPartySize << ".");
00221
00222   const bool isSellSuccessful =
00223     simcrsService.sell (lTravelSolution, lPartySize);
```

```
00224
00225   // Close the log file
00226   logOutputFile.close();
00227
00228   return isSellSuccessful;
00229
00230 }
00231
00232
00233 // ///////////////// Main: Unit Test Suite ///////////////
00234
00235 // Set the UTF configuration (re-direct the output to a specific file)
00236 BOOST_GLOBAL_FIXTURE (UnitTestConfig);
00237
00238 // Start the test suite
00239 BOOST_AUTO_TEST_SUITE (master_test_suite)
00240
00241
00244 BOOST_AUTO_TEST_CASE (simcrs_simple_simulation_test) {
00245
00246   // Schedule input filename
00247   const stdair::Filename_T lScheduleInputFilename (STDAIR_SAMPLE_DIR
00248                                                    "/rds01/schedule.csv");
00249
00250   // O&D input filename
00251   const stdair::Filename_T lOnDInputFilename (STDAIR_SAMPLE_DIR "/ond01.csv");
00252
00253   // FRAT5 curve input file name
00254   const stdair::Filename_T lFRAT5InputFilename (STDAIR_SAMPLE_DIR
00255                                                "/frat5.csv");
00256
00257   // Fare family disutility curve input file name
00258   const stdair::Filename_T lFFDisutilityInputFilename (STDAIR_SAMPLE_DIR
00259                                                        "/ffDisutility.csv");
00260
00261   // Yield input filename
00262   const stdair::Filename_T lYieldInputFilename (STDAIR_SAMPLE_DIR
00263                                                "/rds01/yield.csv");
00264
00265   // Fare input filename
00266   const stdair::Filename_T lFareInputFilename (STDAIR_SAMPLE_DIR
00267                                               "/rds01/fare.csv");
00268
00269   // State whether the BOM tree should be built-in or parsed from input files
00270   const bool isBuiltin = false;
00271
00277   const unsigned int lExpectedPrice = 400;
00278   const unsigned int lExpectedNbOfTravelSolutions = 1;
00279
00280   bool isSellSuccessful = false;
00281
00282   BOOST_CHECK_NO_THROW (isSellSuccessful =
00283                         testSimCRSHelper (0,
00284                                           lScheduleInputFilename,
00285                                           lOnDInputFilename,
00286                                           lFRAT5InputFilename,
00287                                           lFFDisutilityInputFilename,
00288                                           lYieldInputFilename,
00289                                           lFareInputFilename,
00290                                           isBuiltin,
00291                                           lExpectedNbOfTravelSolutions,
00292                                           lExpectedPrice));
00293
00294   // DEBUG
00295   std::ostringstream oMessageSell;
00296   const std::string isSellSuccessfulStr = (isSellSuccessful == true)?"Yes":"No";
00297   oMessageSell << "Was the sell successful? Answer: " << isSellSuccessfulStr;
```

```
00298   STDAIR_LOG_DEBUG (oMessageSell.str());
00299
00300   BOOST_CHECK_EQUAL (isSellSuccessful, true);
00301
00302   BOOST_CHECK_MESSAGE (isSellSuccessful == true, oMessageSell.str());
00303
00304
00305 }
00306
00307
00311 BOOST_AUTO_TEST_CASE (simcrs_simple_default_bom_simulation_test) {
00312
00313   // State whether the BOM tree should be built-in or parsed from input files
00314   const bool isBuiltin = true;
00315
00321   const unsigned int lExpectedPrice = 900;
00322   const unsigned int lExpectedNbOfTravelSolutions = 1;
00323
00324   bool isSellSuccessful = false;
00325
00326   BOOST_CHECK_NO_THROW (isSellSuccessful =
00327                         testSimCRSHelper (1,
00328                                           " ", " ", " ", " ", " ", " ",
00329                                           isBuiltin,
00330                                           lExpectedNbOfTravelSolutions,
00331                                           lExpectedPrice));
00332
00333   // DEBUG
00334   std::ostringstream oMessageSell;
00335   const std::string isSellSuccessfulStr = (isSellSuccessful == true)?"Yes":"No";
00336   oMessageSell << "Was the sell successful? Answer: " << isSellSuccessfulStr;
00337   STDAIR_LOG_DEBUG (oMessageSell.str());
00338
00339   BOOST_CHECK_EQUAL (isSellSuccessful, true);
00340
00341   BOOST_CHECK_MESSAGE (isSellSuccessful == true, oMessageSell.str());
00342
00343
00344 }
00345
00346 // End the test suite
00347 BOOST_AUTO_TEST_SUITE_END()
00348
00349
```