TP161, 05/02

Rational. software

Data Modeling for Data Warehouses

Davor Gornik

Table of Contents

OLTP and Data Warehouse – Where is the Difference?

In our daily life we use plenty of applications generating new data, altering data, deleting data, and of course in most cases reading and analyzing data. Just imagine the simple email application. We have stored addresses, and probably some documents. We can decide if we store sent emails, but will probably delete them after some amount of time including all of the documents we sent. How about addresses we deleted, or changed some time ago? We will not see them any more.

Email program is in most cases not a very complex database, but a good simple example of an OLTP (online transaction processing system) in a single user environment. It uses all of the so-called CRUD operations (create, read, update, delete) to access the data. When a certain size of the data storage is reached, the size remains almost unchanged, because of removal of the older data from the storage.

Data warehouse is a completely different kind of application. It is not used to run current operations like sending email. It is used for analyzing the data and discovering new value out of the existing data, mainly to be able to predict the future. Data warehouse is not a universal structure to solve every problem. It has to be focused on one problem area, like in-flight service, customer revenues, etc.

The interesting thing about the data warehouse is that the database itself is steadily growing. The data is not deleted, nor altered. We do not try to keep the redundancy out of the database (because data is added to the warehouse in a process of data cleansing, which checks the correctness of data) to reduce the complexity and raise the performance of the read operations.

To be able to analyze the data in the data warehouse, the data is stored in a multidimensional structure called star schema. If the star has to be expanded, we call it a snowflake. This white paper will explain the modeling of the star schema and a snowflake using IBM Rational Rose®.

The Flight Service data mart example

To better explain the modeling of a data warehouse, this white paper will use an example of a simple data mart (which is a data warehouse or part of a data warehouse) analyzing the passenger's behavior and satisfaction flying with the airline Happy Flying and Landing.

We will store the passengers and the data about every single flight, the chosen menu, and the satisfaction of the passengers on the flights.

Glossary of a data warehouse

The data warehouse introduces new terminology expanding the traditional data-modeling glossary. For the sake of completeness I will introduce the most common terms.

Data Warehouse

A data warehouse is a collection of data supporting management decisions. The data is subject oriented, integrated, nonvolatile, and time variant.

The data warehouse is the collection of snapshots from all of the operational environments and external sources. It is not up to the minute accurate, because it has to be extracted from the operational environment, which is done on a regular time base.

Data Mart

A data warehouse limited to a single subject area, such as customer, department, location, etc. Data marts can be dependent, when they receive the data from the data warehouse, or independent, when they receive the data from the operational systems.

Fact

The unit of information in a data warehouse. The fact is a cell in the multidimensional space, limited by the units of analysis.

Facts are stored in a table (when used in a relational database) or are a cell in a multidimensional database.

Every fact contains the basic information about the fact (revenue, value, satisfaction note, etc.), and relates to the dimensions.

In some cases the pure occurring of the fact is information enough for the data warehouse, when all of the necessary information is stored in the dimensions. We talk then about a factless fact.

Dimension

The axes of a coordinate system that bind the space defined by that coordinate system. The coordinate system in a data warehouse defines the data cells, which contain facts.

An example of a coordinate system is the Cartesian coordinate system with x and y dimensions.

In a data warehouse, one of the dimensions is always time.

Data Mining

The process of discovering new information out of data in a data warehouse, which cannot be retrieved within the operational system, is called data mining.

Analytical space

The amount of data in a data warehouse used for data mining to discover new information and support management decisions.

Slicing

A technique used in a data warehouse to limit the analytical space in one dimension to a subset of the data.

Dicing

A technique used in a data warehouse to limit the analytical space in more dimensions to a subset of data.

Star Schema

A schema realizing a multidimensional analysis space using a relational database is called a star.

The star schema will be discussed further later on in this white paper.

Snowflake

When the dimensions of a start schema have to be normalized because of any reasons, the schema evolves to a snowflake.

Using IBM Rational Rose to model a Star Schema

The basic form of a star schema has to realize a multidimensional space (often called a dice), using the basic capabilities of a relational database.

First we need an understanding of a multidimensional space.

Multidimensional analysis space

A geometrical dice is an example of 3 dimensional spaces with all 3 dimensions of the same size. Imagining a cube with each dimension of three units we get $3^3 = 27$ cells of equal structure.

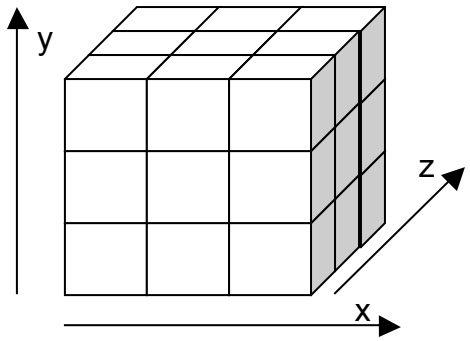

Figure 1 A dice with dimensions x, y, and z

The multidimensional analysis space (or a data warehouse dice) differs just in details from a geometrical space:

- The dimensions are not limited to just 3. However it is not easy to handle a cube with lots of dimensions, which results in most of the implementations to be limited to 6 or 7 dimensions. Do not expect a good graphic representation of more than 4 dimensions – and if you find one do not forget to tell me about it.
- The dimensions are not of the same size and units. The size can differ extremely from few units to several million units. The units can be a day, a customer, department, etc.
- The cell, which equals the (1 times 1 times 1 etc.) sub dice, contains facts.

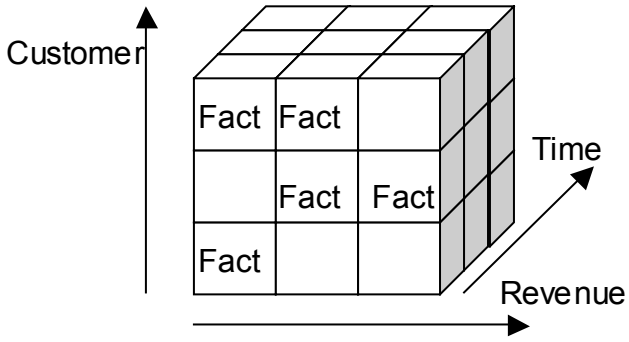

Figure 2 A 3 dimensional data cube

The data cube needs lots of memory to store all the facts. The cell has to be reserved regardless of the presence of the contained fact.

This is the reason to use relational database and a star schema, which is able to optimize the storage and remain the flexibility of data structures.

Star Schema

The basic idea of the start schema is to retain the multidimensional capability of the cube while adding the flexibility of smaller data storage.

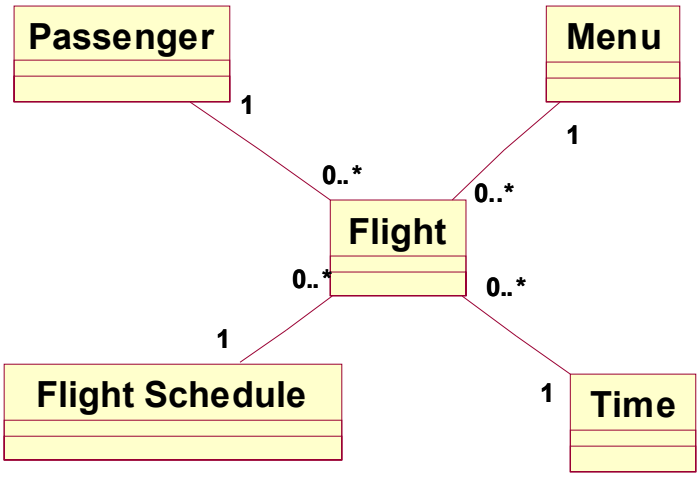

Figure 3 A Star Schema

In the figure 3, the star schema represents a dice of 4 dimensions (Passenger, Menu, Flight Schedule, and Time) with the fact Flight. Basically the fact has to refer to every dimension, to be placed in a cell of the cube.

The dimensions in our example are:

- Passenger describes the each individual passenger of the airline identified by the frequent flyer number. Passengers not participating at the frequent flyer program are not part of the data warehouse.
- Flight Schedule is the schedule of all regular flights.
- Menu is the menu served on the flights. Only a basic classification of menus is important for data mining.
- Time is the time of the flight.

The fact Flight describes a single flight of a Passenger at a distinct Time choosing a Menu.

The analytical space can be the whole dice, or we can slice the analytical space according to the dimensions in smaller pieces.

Each dimension is described according to an object, which is represented by a class, which is names according to a business subject. This is most important for the success of the data warehouse, as the user of the warehouse (manager, analyst, marketing) will be inexperienced with terms of information technology.

The fact itself is another object of the business intelligence, represented again by a class.

The fact refers to every dimension. The association between the fact and the dimension is always one to any, which means each fact is associated with exactly one unit of the single dimension, and each unit of the dimension (each Passenger, Time, etc.) can be associated with any number of facts (including 0).

Transforming the object model into the data model does the implementation of the star schema using Rational Rose. Here we see the result of the transformation.

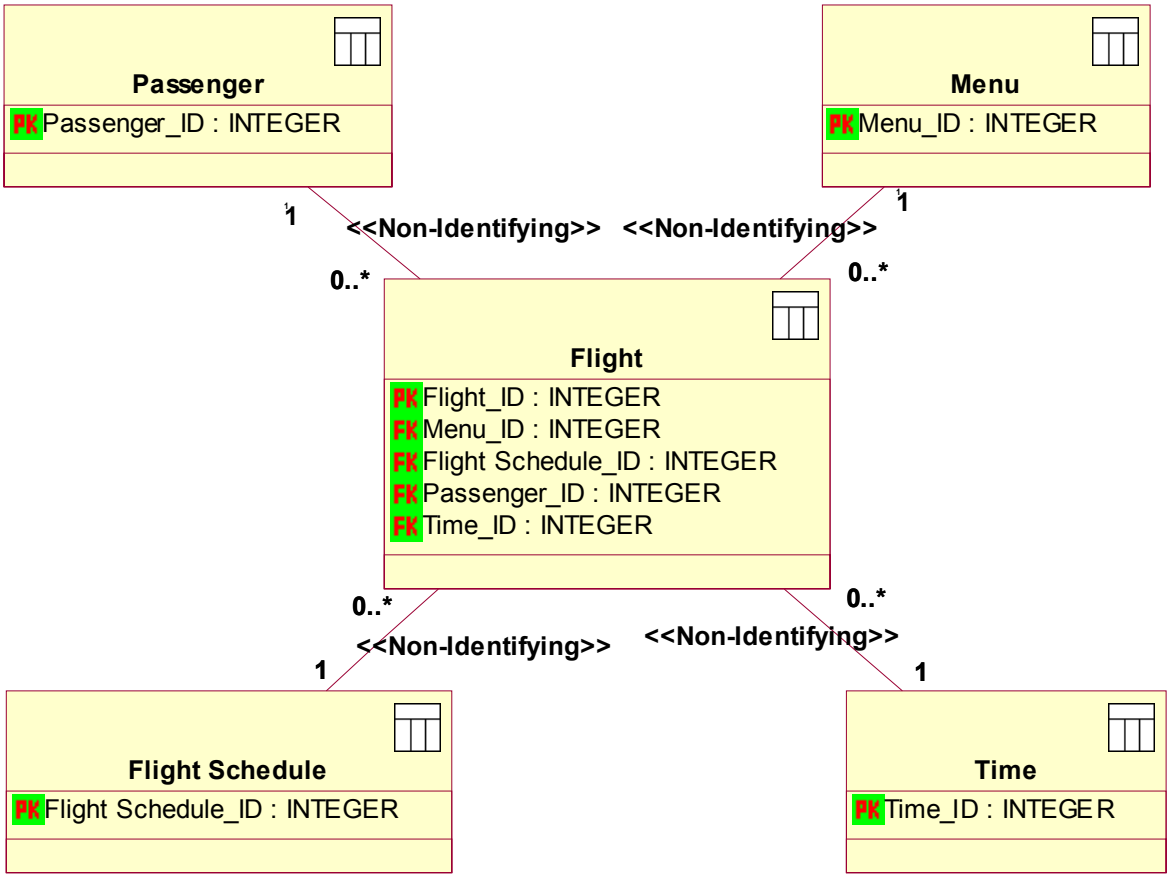

Figure 4 The Implementation of the star Schema using Rational Rose

In the figure 4, the automatically generated primary and foreign key constraints were hidden from the display.

The dimensions of the star schema are independent tables. Rational Rose generates the primary key of the dimensions when the object model is transferred to the data model.

The fact table refers to the dimensions using the key migration from the dimension tables. Rational Rose generates the foreign keys when the data model is generated.

Slicing and dicing in a star schema is a limitation (selection) of dimensions. It is a run-time issue, not a modeling one, but the model has to recognize the need of it.

Snowflake

The basic star schema does not satisfy all needs of the data mining. We need more complex dimensions, like for example time. The analyst wants to recognize the patterns according to week, month, quarter, etc.

The dimensions have to be normalized. We will not want the redundancy of the dimension table, which would make the slicing of the data more complicated. The schema we get with such a process is called a snowflake.

Lets see an example of a simple snowflake. We will normalize the time dimension to weeks, months, and quarters.

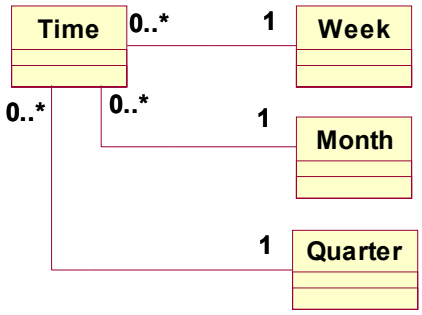

Figure 5 Normalized Time dimension

We would like to be able to slice the cube with every of the additional normalized dimensions: week, month, and quarter. In this example we assume quarter as a parallel hierarchy to month, which means we cannot assume a quarter to be an aggregation of several months. For this reason we preselect the dimension of time using one of the normalized tables, which is a simple addition to the OLAP queries.

The resulting snowflake adds the normalized dimension.

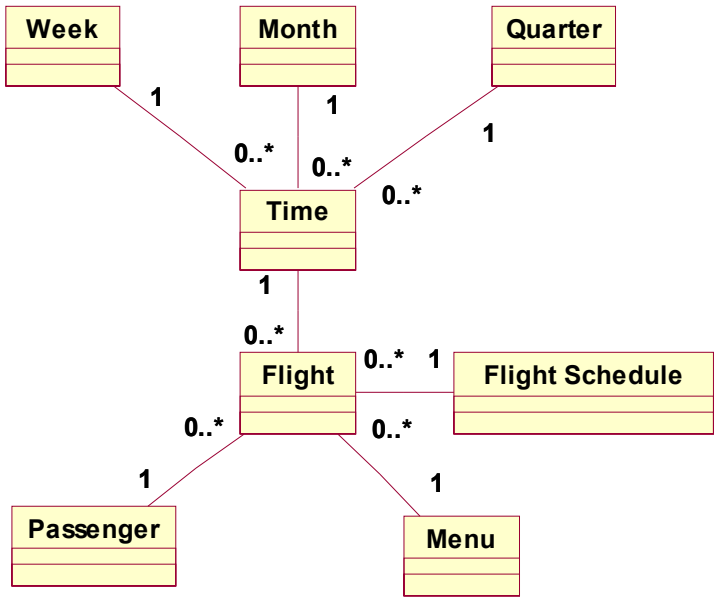

Figure 6 The Snowflake with the normalized dimension of Time and the fact Flight

Of course all of the dimensions can be normalized like the time example, which results in more complex data mart schema.

The implementation schema (data model) developed by Rational Rose out of this snowflake is complete.

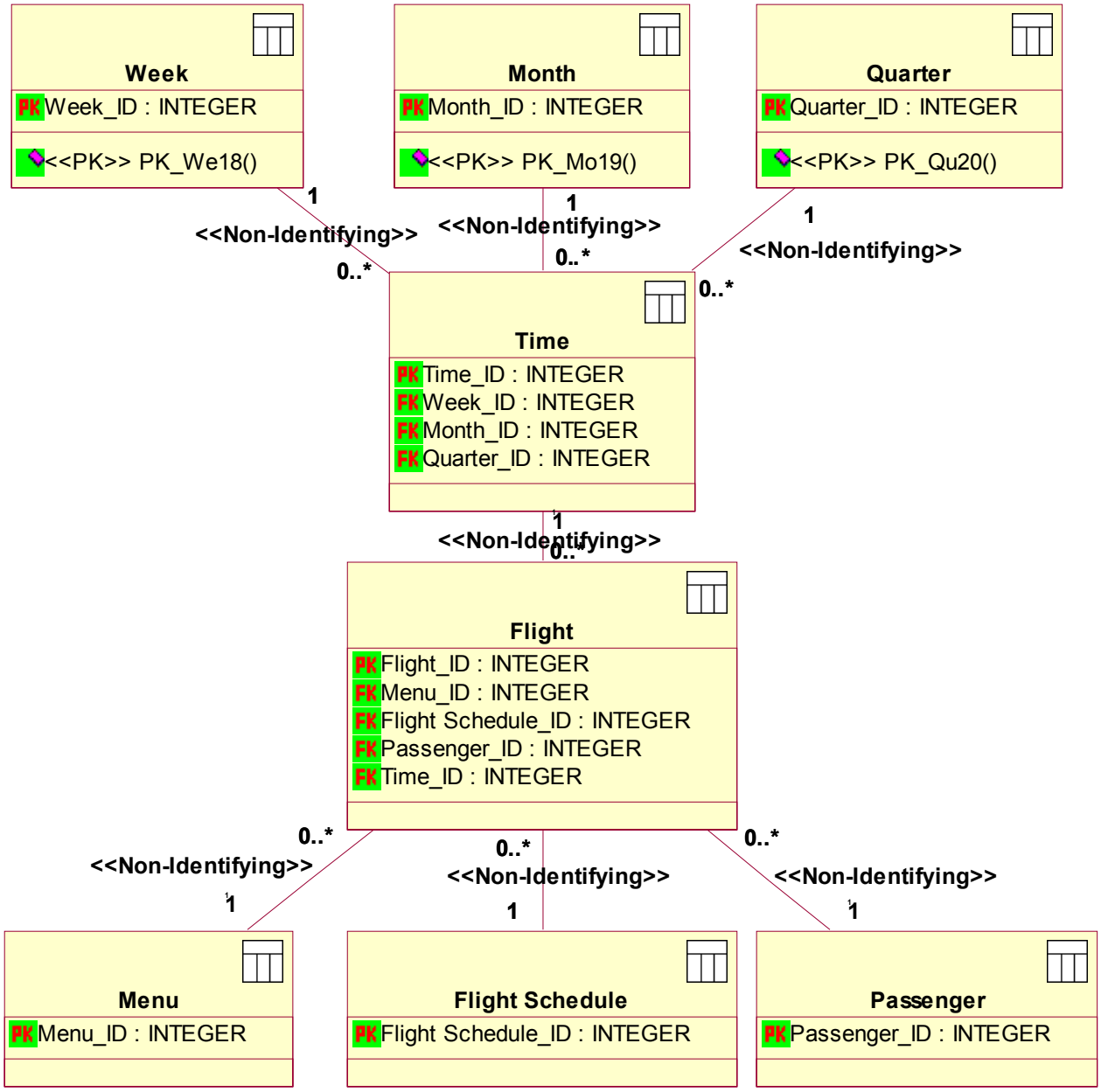

Figure 7 A data model of a snowflake with normalized Time dimension

Again, the generated constraints have been hidden displaying this diagram.

Slicing in a snowflake is possible not just at the basic Time dimension, but also at normalized Week, Month, and Quarter.

Many-to-Many Relationships

On a flight we do not have to have just one meal. There can be several of them on a long flight. In this case we would not have a one-to-many association between the Flight fact and the Menu dimension. We have to have a many-to-many association. However such an association cannot be implemented in the star schema.

A special form of a snowflake implement the necessary structure of the data to realize this need.

First of all we change our model to a many-to-many association between the fact and the dimension. With Rational Rose this is just a change in the cardinality of the association.

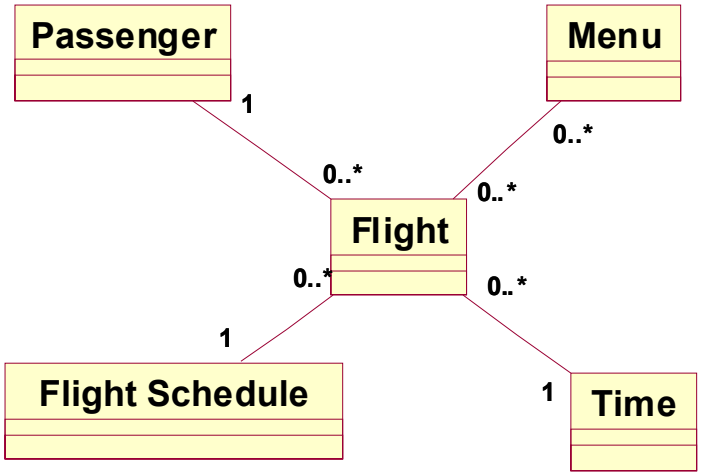

Figure 8 The star schema with many-to-many dimension on Menu

We are not able to implement a many-to-many association with a relational database. The implementation uses another kind of snowflake to implement the many-to-many relation.

In the following figure we focus at the part of already developed snowflake, which deals with the many-to-many dimension.

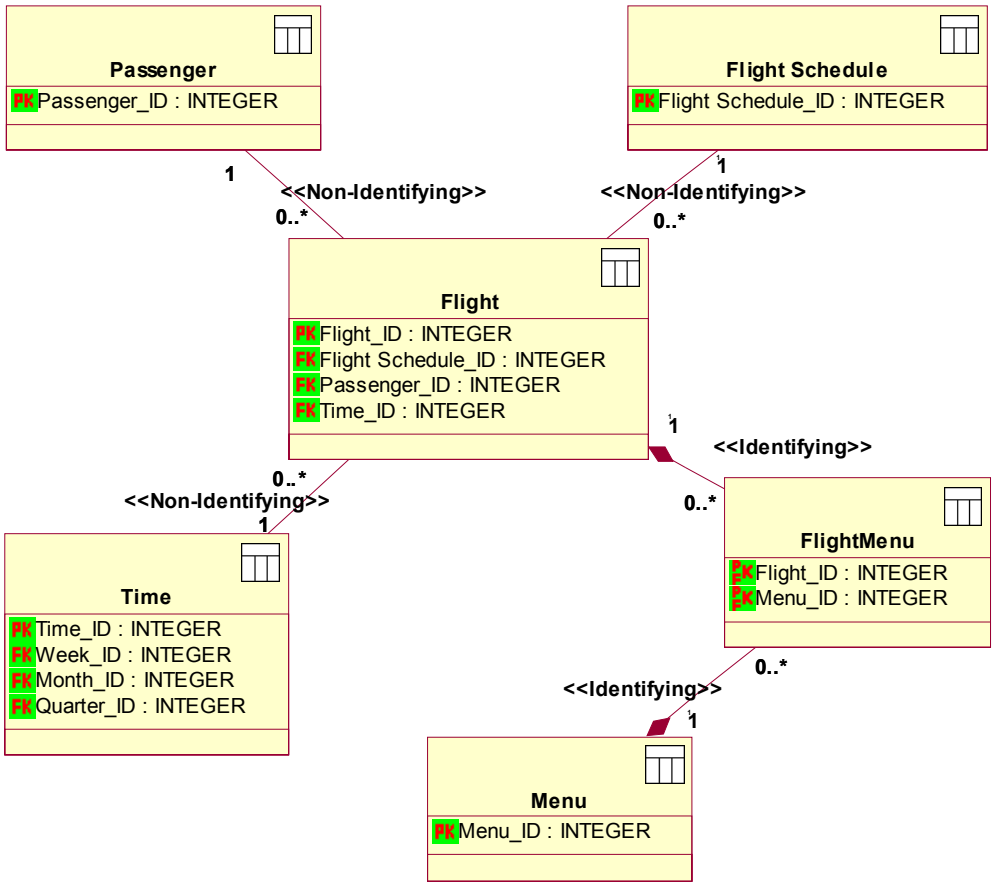

Figure 9 The snowflake resolves the many dimension of the Menu

Rational Rose generates the additional dimension table FlightMenu, which refers to the dimension Menu and the fact Flight.

Identifying relationships are used to resolve the many-to-many association.

The most important for the architect of a snowflake is to recognize such a relationship. The simple object view gives him the opportunity for the understanding of the concept, whereas the generated data view dives deeper into the implementation.

Hierarchies

Data mining discovers information from the data that is hidden at the surface of the operational systems. One of the questions we would like to ask could be the dependence of the chosen menus on the demographics.

The demography data can be built on the Passenger dimension as a hierarchy. Passenger can be grouped according to the ZIP code, then according to the country.

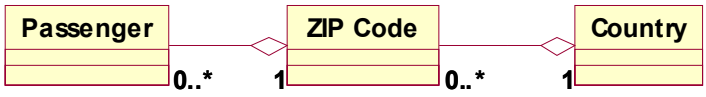

Figure 10 The hierarchy of the Passenger

The hierarchy is specified using an aggregation. The aggregation defines the containment. The Country contains the ZIP codes and the ZIP Code can contain multiple Passengers.

The resulting implementation used foreign keys to implement aggregations.

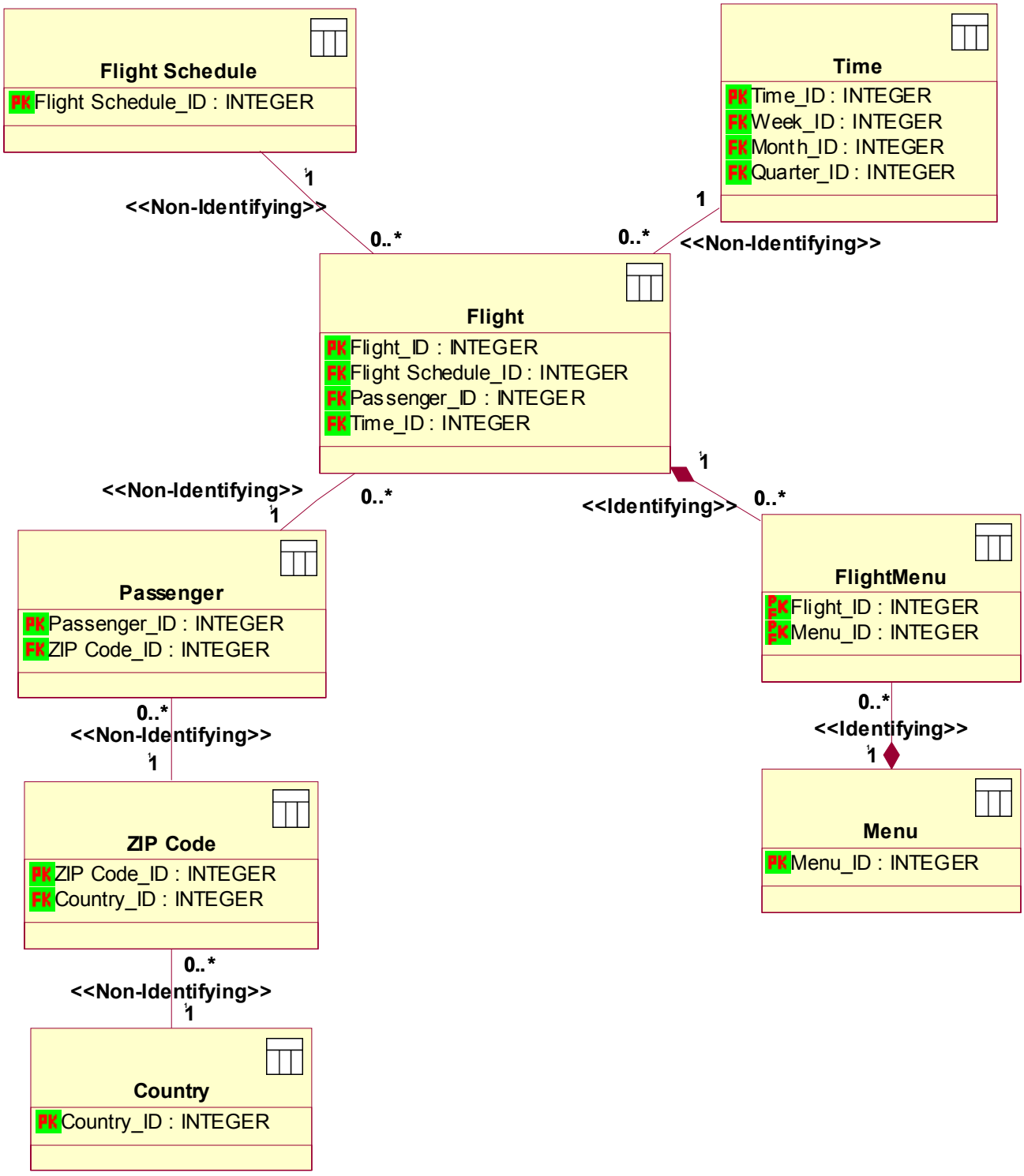

Figure 11 Snowflake implements the aggregation of a Passenger dimension

Again the generated constraints have been hidden from the figure.

Using aggregation the dimensions can be used at any level defined. The analytical space can be sliced by Passenger, ZIP Code, or Country.

Conforming Dimensions

The snowflake becomes more and more complex as the architect of the warehouse adds more and more details. The design process has to stop at some level to remain the data warehouse perform well enough.

Still a star or a snowflake schema focuses just on one fact – in the example the Flight. How about complex relationships?

For each fact we have to design a schema of its own. They have to have a common dimension – we call it conforming dimension – if we want to make complex queries.

Lets define a second star schema with Pilot as a dimension and PilotFlights as a fact. We have additional dimensions Flight Schedule and Time.

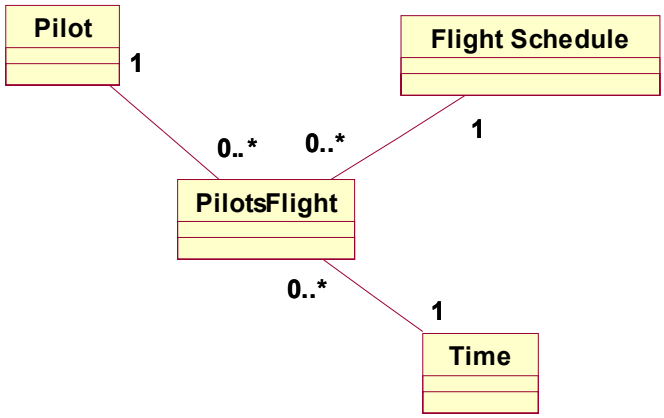

Figure 12 The Pilots star schema

The second schema can be used stand alone or combined with the Passenger schema to query the satisfaction of the Passengers according to the pilots using the conforming dimensions.

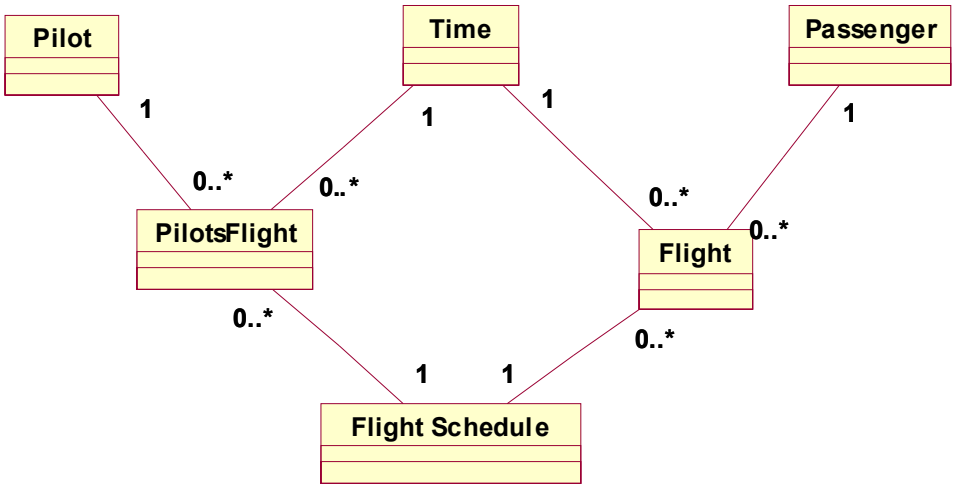

Figure 13 Conforming dimension Time and Flight Schedule

The relationship between Pilots and Passengers is easy even in the simple structure of a data warehouse using the conforming dimensions.

A data warehouse combines lots of small star schemas and snowflakes into a big data warehouse schema while developing the data model.

The data of the fact and dimensions

We want to evaluate the satisfaction rates of the Passengers at the Flights. They rate it with a note from not satisfactorily to excellent. This note is stored in the fact table Flights as an attribute (column).

If we want to build an average note, we have to scale the note to a value we can calculate with. We can map the notes to 0 to 10. With this we can calculate an average note. The averages should be stored in the dimension tables, to be used for a simple slicing, where we want to slice just in one dimension.

Rational Rose generates the attributes for the implementation according to data types of the target database. The object model is used to define the source of the data for the database.

Summary

IBM Rational Rose can be best used for designing a data warehouse implementation.

The object model defines the object related global structure of the schema and the whole data warehouse including the source of the data. It represents the object related view at the data warehouse and hides the implementation details.

The data model is the implementation model of the data warehouse. The data model can be generated out of the object model and vice versa.

Rational Rose is the best tool for designing the star as well as the snowflake schemas. It is variable enough to support any concept data warehouse will need, and offers the functionality the data architect and the database administrator needs to tune the data warehouse.

Rational Rose offers the power to analyze the business and develop the requirements to the implementation of a data warehouse.

IBM software integrated solutions
IBM Rational supports a wealth of other offerings from IBM software. IBM software solutions can give you the power to achieve your priority business and IT goals.

- *DB2® software helps you leverage information with solutions for data enablement, data management, and data distribution.*
- *Lotus® software helps your staff be productive with solutions for authoring, managing, communicating, and sharing knowledge.*
- *Tivoli® software helps you manage the technology that runs your e-business infrastructure.*
- *WebSphere® software helps you extend your existing business-critical processes to the Web.*
- *Rational® software helps you improve your software development capability with tools, services, and best practices.*

Rational software from IBM
Rational software from IBM helps organizations create business value by improving their software development capability. The Rational software development platform integrates software engineering best practices, tools, and services. With it, organizations thrive in an on demand world by being more responsive, resilient, and focused. Rational's standards-based, cross-platform solution helps software development teams create and extend business applications, embedded systems and software products. Ninety-eight of the Fortune 100 rely on Rational tools to build better software, faster. Additional information is available at www.rational.com and www.therationaledge.com, the monthly e-zine for the Rational community.

IBM Corporation
Software Group
Route 100
Somers, NY 10589
U.S.A.

Printed in the United States of America
01-03 All Rights Reserved.
Made in the U.S.A.

The IBM home page on the Internet can be found at **ibm.com**