Deriving Initial Data Warehouse Structures from the Conceptual Data Models of the Underlying Operational Information Systems

Dipl.-Wirtsch.Inf. Michael Böhnlein, Dipl.-Inf. Achim Ulbrich-vom Ende,

University of Bamberg, chair of information systems, especially system engineering and database application, Feldkirchenstr. 21, D-96045 Bamberg, Tel. +49-(0)951-863-2514,

E-Mail: {michael.boehnlein | achim.ulbrich}@sowi.uni-bamberg.de

Abstract

In recent years the major problem of conceptual data modelling for business needs has been the construction of large scale data schemes for operational systems. But in fact multidimensional data structures used for decision support applications in data warehouses have rather different requirements to data modelling techniques. Instead of creating the data model from the application specific requirements as in the case of operational systems, the development of data warehouse structures implicates the consideration of userdefined information requirements as well as the underlying operational source systems. In this paper we present a new approach which involves the creation of multidimensional structures from the data models of the underlying operational information systems. This ensures the more effective modelling of data warehouse structures. We would like to point out that the special features of the Structured Entity Relationship Model (SERM), which is the conceptual basis of the data modelling technique used by the SAP Corporation and an extension of the conventional Entity Relationship Model (ERM), are not only useful for the development of big operational systems but can also help with the derivation of data warehouse structures. To illustrate the usability of this approach we show the derivation of the warehouse structures of a conceptual data model of an airlines flight reservation system. The empirical background of our approach is the CEUS-HB project (*Computerbased Decision Support System for the Higher Universities of Bavaria*), by which a hierarchical distributed data warehouse system for the Bavarian universities is being developed. The project is initiated and supported by the Bavarian Ministry of Science, Research and Art. We are members of an interdisciplinary research group of the Bavarian State Institute For Higher Education Research (Prof. Dr. H.-U. Küpper) and the University of Bamberg (Prof. Dr. E. J. Sinz).

1. Introduction

During the last decade data warehouse systems have become an essential component of modern decision support systems. According to a survey done by the Meta Group, in 1996 alone 4.5 billion US-Dollar was spent on data warehouse projects. For the year 2000 a turnover of 15 billion US-Dollar is expected in the warehousing industry. This reflects positively on the basic functionality of these systems. Data warehouse systems offer efficient access to integrated and historical data from different, partly heterogeneous and autonomous information sources for helping managers in planning and decision making. This is achieved by extracting the relevant data from the operational sources, cleaning the data and storing it redundantly in a single repository. The data within the data warehouse is cleaned, consolidated, aggregated and accumulate in multidimensional data structures to support direct querying and multidimensional analysis.

The development of data warehouse systems is rather different from the development of conventional operational systems. The latter has only to fulfil the application specific business needs. The design of data warehouse systems not only involves that consideration begiven to the information requirements of the user, but also to the structure of the underlying source systems, to the consistency requirements of the user to query performance enhancing techniques. These factors have static as well as dynamic aspects, illustrated by possible changes in user requirements and variations in the structure of the underlying source tables. This clearly illustrates the need for a comprehensive data warehouse modelling technique. We have been pursuing this goal and through this paper we aim to describe the first part of our work: the derivation of initial data warehouse structures from the underlying source systems. For the reason that most source systems are relational, we had to find a suitable ERM extension for the identification. In our opinion the Structured Entity Relationship Model (SERM) is the best alternative. It forms the basis of the data modelling technique used by the SAP Corporation. For the derivation of data warehouse structures we assume the conceptual data models of the operational source system designed on the basis of SERM. This is not a restriction, as every consistent entity relationship diagram can be transformed into a structured entity relationship diagram. If no ERM documentation of the source systems is available, the SERM offers a comparatively speaking easier redesign functionality.

On the subject of application development most authors distinguish between conceptual, logical and physical development phases (e.g. [Voss99]). For classical data modelling techniques there exist modelling methods for all three development phases. Currently the

main emphasis in the development of data warehouses is put on the logical and physical layer ([LaQA97]). In most recent times some modelling concepts have been extended to the conceptual, business oriented issues ([Bulo96]). In the absence of a standard we only aim to describe the relational alternative for multidimensional data modelling - the star schema. Considering that this approach can also be extended to other (e.g. multidimensional) modelling alternatives, this is not a restriction.

We begin by looking at related work in data warehouse modelling, after which we briefly mention the conceptual data modelling technique of the SERM and explain the ERM. Section 3 contains an overview of the basic elements of multidimensional data structures and their representation in the logical modelling diagrams of the star scheme. The main focus of this paper is the derivation of initial data warehouse structures as presented in section 4. This illustrates the practical application of our approach through a conceptual data model of a flight reservation system. The result of the derivation is shown by means of star scheme. The paper is concluded with a summary and an outlook on future research topics.

2. Related Work on Data Modelling for Data Warehouses

There exists no set standard for the conceptional modelling of data warehouses so far. Some of the basic methods which have been proposed include: *Application Design for Analytical Processing Technologies* (ADAPT) ([Bulo96]), "*sichtenspezifische Modellierung"* ([GaGl97]) and *Dimensional Modelling* ([GoMR98]). There is general consensus that classical ER modelling techniques are not suitable for warehouse design. „Entity relation models cannot be used as the basis for enterprise data warehouses.“ ([Kimb96]). „ERD provides us with no good way of modeling hypercubes – the basic building blocks of OLAP databases.“ ([Bulo96]). For this reason we do not use a conceptual modelling technique for data warehouse structures in this paper. The basic building blocks of multidimensional data structures as a central basis of data warehouses are briefly discussed in this section.

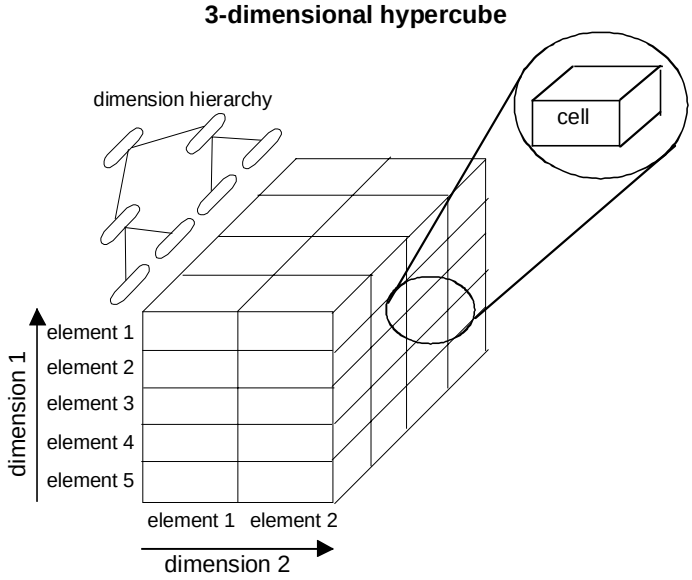


Figure 1: Basic elements of multidimensional data structures

The basic idea of multidimensional data structures is the separation of quantitative and qualitative data. Quantitative measurable facts, called *measures* or *measured facts*, are analysed from various viewpoints based on the qualitative content of the data ([Kena95]). For example the turnover of a company could be analysed by examining the product structure, sales structure and time. The combination of some qualitative aspects to a common coherence is called *dimension*. This classification leads to a n-dimensional data structure, called *hypercube* (multidimensional cube). Two-dimensional structures can be represented as tables, three-dimensional structures as cubes (figure 1) and four-dimensional structures as tesseracts. With traditional visualisation techniques it is not possible to clearly display hypercubes with more than three dimensions. Every dimension has a set of dimension elements, e.g. a product dimension of an automobile company could include different types of cars. The intersection of dimension elements for every dimension in the cube forms a cell with the quantitative measure. Dimension elements can be arranged hierarchically. The measures in the hypercube are summarized along the dimensional hierarchies according to mathematical rules. The most frequently applied consolidation rule is the summarization, e.g. the turnover of different products has to be summarized to a single product group. But it is also possible to use more complex rules. In this paper no mention is made of any special case of dimensional modelling, like parallel hierarchies, proportional aggregation or unbalanced trees ([Holt98]).

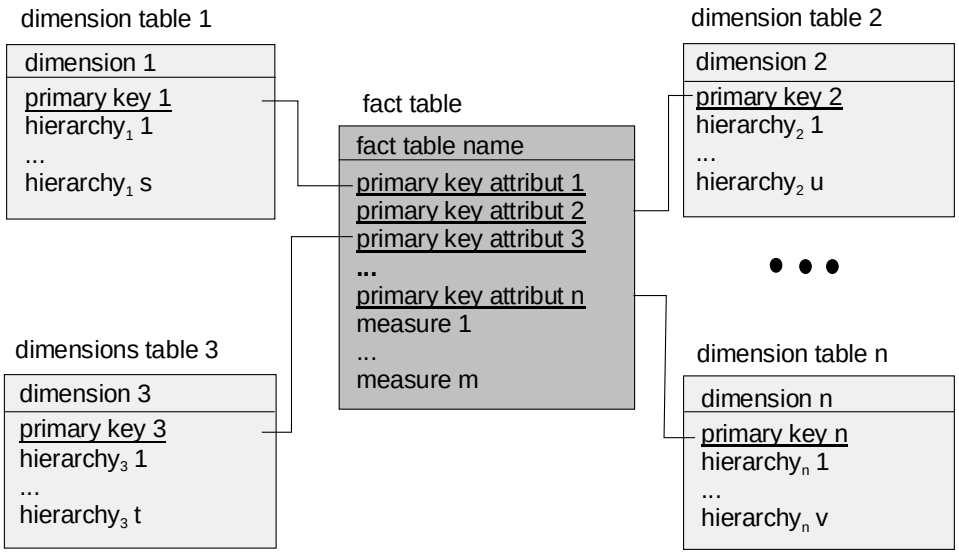


Figure 2: Star scheme

On the logical layer the techniques of star and snowflake models and their variants have become a well known representation of multidimensional data structures in relational based systems ([Hahn98]).

In the star scheme there are two different variants of relation types: a *fact table* and at least one *dimension table* to describe a hypercube ([Rade95], [Rade96]). As the name of this technique indicates, the visualisation of a star scheme looks like a star (figure 2). The measures are stored in the fact table and the dimension tables include the hierarchical structure of the qualitative attributes. Connecting the dimension tables and the fact table is made possible by storing the primary keys of the dimension tables in the fact table, where they become part of the primary key. The hierarchies of the dimensions are modelled in the corresponding dimension table. However there is no information about the hierarchy levels stored in the basic structure of the star scheme. Additionally the representation of a dimension by a single table leads to redundant, denormalized data and potential inconsistencies caused by anomalies. In data warehouse environments this cause no problem as exclusive read-only access to the data is common in most cases, except for the warehouse maintenance.

A *snowflake scheme* is a transformation of the star scheme based on the third normal form. The redundancies are eliminated by the normalization of the dimension tables. For every dimension hierarchy there exists a separate table (*dimension table*). Connections between the dimension hierarchy tables are realized by passing the primary key to the underlying table in the hierarchy.

The selection of a suitable model is based on the trade-off between storage costs and query performance. The star scheme tends to offer better query performance due to fewer join operations between the fact and the dimension tables. This is only achieved by a higher storage cost. The snowflake scheme involves more join operations but needs less storage capacity due to the normalization.

For both models there exist a lot of different variants. For example the *galaxy scheme* is a star scheme with several fact tables ([McGu96]). The *starflake scheme* ([AnMu97]) or *degenerated snowflake scheme* is a combination of the star and the snowflake scheme in which a part of the dimension tables are denormalized. Within the *fact constellation scheme* ([Rade95]) there are additional fact tables for specific dimensions including already summarized data.

A different approach to deriving data warehouse structures from ER-diagrams is discussed in [GoMR98]. The major disadvantage of this approach is having to find a first point of reference for the derivation in the ER-diagram. This difficulty is clearly illustrated when the conceptual data models of the underlying operational systems are very complex. The SERM allows for a better graphical representation of, and is more suitable for the derivation of complex schemes. The structure of the SERM is described in the following section.

3. Conceptual Data Modelling for Operational Business Applications Systems

The most commonly applied modelling technique for data in operational business systems is the Entity Relationship Model (ERM) developed by Chen in 1976 ([Chen76]). The popularity and the acceptance of the ERM can be attributed to the natural expressions that resemble terms of objects in the real world. Furthermore this data modelling technique has a wide range of applications. The entity type (E-type) and the relationship type (R-type) are the basic components of the ERM. The E-type represents a set of entities, which include objects in the real world and abstractions that can be characterized by their specific properties. An R-types is the generalization of the link between two or more entities. In ER-diagrams E- and R-types are connected by edges. To indicate the complexity of the relationship between entities the (1,m,n)-notation is used in the basic design of the ERM. From a graph-theoretical point of view, the ER diagram describes general bipartite graphs with nodes all with the same ranking.

A comprehensive extension of the ERM is the *Structured Entity Relationship Model* (*SERM*) ([Sinz87], [Sinz88]). It eliminates some of the representational

and analytical disadvantages found in the ERM ([Sinz92b]). The major goals of the SERM ([FeSi98]) are:

- Designing extensive data models

  Due to the fact, that in ER diagrams all nodes have the same ranking, it is very difficult to find a suitable starting point in the analysis of a data model. In SERM the nodes are arranged in such a way as to indicate their interdependencies in a hierarchical way. This leads, from a graph theoretical point of view, to a quasi hierarchical (acyclic and directed) graph as opposed to the bipartite graph of the ERM.

- Visualisation of the order of dependencies between data object types

  Existence dependencies and sequences of existence dependencies are clearly represented. In contrast to the ERM, where relations between E-types are modelled, modelling in SERM means constructing the data model based on the principle of dependency.

- Avoiding inconsistencies

  The hierarchical structure of a SER-diagram prevents the modelling of a cycle, a special kind of closed loop. These cycles are syntactically correct but lead semantically to inconsistent data models.

- Avoiding unnecessary relationships

  In contrast to ERM the creation of a relational database from a conceptual data model in SERM can be done with very little structural transformations. Due to the introduction of foreign keys and the directed inheritance thereof, the transformation into a relational database structure is more easily achieved.

Besides the E- and the R-type the SERM also includes an *entity-relationship-type* (ER-type). This is a combination of an E-type and a R-type with a (1,1)-relationship. The ER-type has two faces: from the left side it is a R-type symbol and from the right side it is an E-type symbol. Different kinds of edges between the data object types correspond with the specification in (min,max) notation (figure 3). According to the representation rules in the SER-diagram every edge is directed, that runs from the rectangle to the rhomb; from the left to the right.

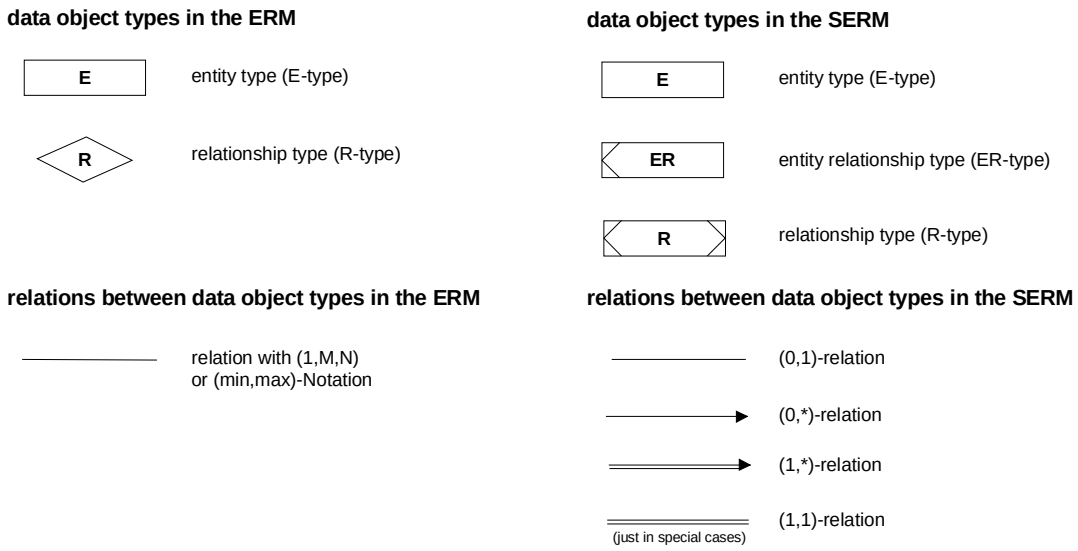


Figure 3: Modelling components in ERM and SERM

A given ERM without inconsistencies can be transformed into a SERM. A correct transformation from ERM to SERM is shown in [Gros92]. A shortcut of the transformation rules is shown in Appendix A.2. The SERM forms the basis of the data modelling technique SAP-SERM which the company SAP Corporation use in their information modelling method SIMO.

Most important for the identification of initial data warehouse structures in our approach is the feature of the SERM to visualise existency dependencies between data object types. The basic data object type is located on the left hand side of the SER-diagram and all dependent data object types to the right of it.

4. Derivation of Initial Data Warehouse Structures from a Data Model of an Airline

We now introduce a typical airline as an example of an application domain to visualize the derivation of initial data warehouse structures. The goal of the airline is to improve its turnover from the previous year. Further objectives are maximization of profits and service performance. We assume that the turnover of the airline consists mainly of flight bookings.

With the available part of the data model of a central flight reservation system for several airlines we point out specific restrictions and assumptions of the application domain (figure 4)[1].

[1] The corresponding ER-diagram is shown in Appendix A.3.

An airplane is a specific aircraft type and is assigned exactly to one airline and can be used for different flights. However, we assume that a specific flight is flown only by the same aircraft. A flight consists mostly of different flight intervals, which correspond to specific stages between two airports. A stage can occur in different flight intervals and therefore can be served by different aircrafts. Airports are characterized by their geographical location. Therefore the data object types *city*, *state* and *country* are explicitly mentioned. A specific reservation is associated with a specific booking class. The number of available seats for a booking class depends on the aircraft. In this example we assume that the fares remain constant, related only to the flight interval and the corresponding booking class. Furthermore each passenger reservation transaction corresponds explicitly to a specific flight interval and a specific booking class. The combination of several flight intervals, which is often called trip reservation, is only implicitly considered. In the illustration PK and FK indicates the inheritance of the primary or the foreign key respectively.

The characteristic attributes and an extension of the data object types is shown in the appendix (figure 6).

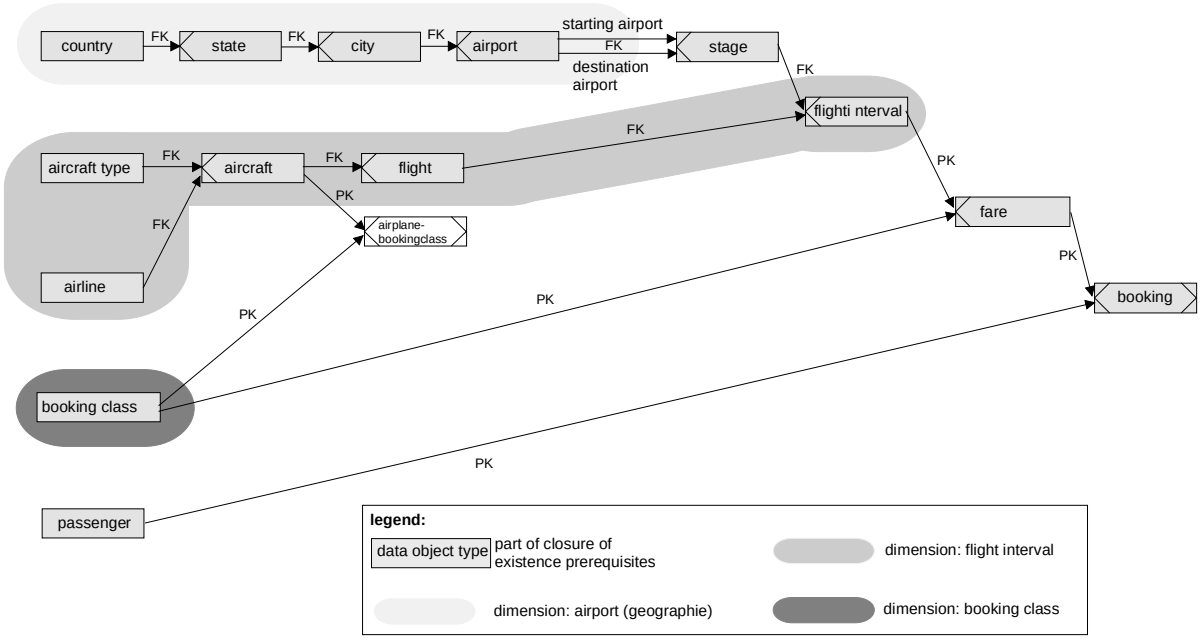


Figure 4: Identification of initial data warehouse structures in the SER-diagram

We now introduce a detailed description of the derivation of initial data warehouse structures from the conceptual scheme of an airline. As a basis we take the already introduced conceptual data model of an airline in the SERM. We assume a model in third normal form for further considerations. We observe the following three stages:

a) Identification of business measures

b) Identification of dimensions and dimension hierarchies

c) Identification of integrity constraints along the dimension hierarchies

ad a) Identification of business measures

Business measures are determined by the argumentation chain: goals – services – measures.

We assume as given the commitment to goals of a particular company. We differentiate between goals and objectives. While goals describe the nature and purpose of the service production, objectives comprise of the nature and extent of achieved goals. The airline in question pursues the goal of increasing its turnover from the previous year. Objectives include the maximization of profits and service performance. Next we examine the production program of the company. The airline offers a reservation service for transportation. A service is always immaterial. How can the offered service be measured? The measurement of a service has to be carried out by adequate quantitative values. Service performance and achievement of the set goals should be evaluated through adequate measures. One has to distinguish between basis and derived measures. Business events describe hints for discovering adequate measures. The number of reservations is a meaningful basic measure to evaluate the offered transport services. This value corresponds to the business event "reserving flight by a passenger". With additional information about the fare we can determine the derived measure *reservation turnover*.

The identified measures, which form the core of the multidimensional data structure, are assigned to one or more data object types of the conceptual model in SERM. The structure of the existency dependencies in the SERM is helpful for further considerations. The diagram is so designed that independent data object types are positioned on the left hand side and dependent data object types are positioned to the right. The resulting structure simplifies the visual identification of existency dependencies and sequences of existency dependencies. This offers another dimension to the classical paradigm of dividing object types in structural and transactional data. From the aspect of existency dependencies this paradigm is generalized to a visualization of related object types into pairs ([Sinz92b]). The data object types on the left hand side are “more structural” and the data object types on the right hand side “more transactional”. For this reason we have to assign measures to dependent data object types. One can get information about the measure “number of reservations” through the data object

type *booking* in the conceptual SERM scheme. The derived measure "reservation turnover" can be determined from the data object types *booking* and *fare*.

ad b) Identification of dimensions and dimension hierarchies

The clear visualization of existency dependencies in a conceptual SERM scheme is helpful for the identification of potential dimensions and dimension hierarchies. To identify candidates for dimensions and dimension hierarchies we first have to determine the closure of existency prerequisites C_{EX}. The starting point is the data object type assigned to the chosen measure. Then we have to pass through the edges in the opposite direction from right to left and gradually enclose all data object types with existency prerequisites to the data object types of our measures. In the example (figure 4) we determine the closure of existency dependencies for the *number of reservations* and the *reservation turnover* starting from the data object types *booking* and *fare*. The existency prerequisite of *fare* is *flight interval* and *booking class*, for *booking* it is *passenger*. The data object type *passenger* has no further existency prerequisites. Furthermore *flight interval* depends on *stages* and *flights*. Existency prerequisites for *stages* is *airport*, *city*, *state* and *country*. Correspondingly to the other existency prerequisites for *flight* the data object types *aircraft*, *airline* and *aircraft type* are added to the closure. The data object type *aircraft-booking class* which comprises the number of seats in an airplane related to a specific reservation class, is not part of the closure. The resulting closure of existency prerequisites for the data object type *booking* is:

C_{EX} (booking, fare) = {booking, fare, passenger, flight interval, booking class, stage, flight, airport, aircraft, city, aircraft type, airline, state, country}

To help with the understanding of the described algorithm we also show a formal specification in Appendix A.4. The closure of existency dependencies is made more visual through a gray filling of the data object types in figure 4. The emphasis is the sub graph that consists of data object types and the related edges from the closure of existency prerequisites.

Discovering dimensions and dimension hierarchies requires creativity and considerable knowledge of the application domain. With our approach the development of initial data warehouse structures is supported. Starting with the data object types of the chosen measure we examine object types along the edges of the graph in the direction right to left for potential dimensions and dimension hierarchies. Data object types can form the base for building a dimension. Decisive criterion is that the data object types have attributes as these are important for analytical purposes. The data object type *booking class* comprises of information about the booking class of a passenger and the specific airline. This information

is important for the evaluation of the number of reservations and the reservation turnover. Therefore we create the dimension *booking class*. For the reason that our model is in the third normal form a clue for identifying dimension hierarchies in a single data object type, cannot be found. It is therefore not possible to derive any information about hierarchies in the dimension *booking class* from our conceptual scheme.

We now examine the data object *type flight interval* and its existence prerequisites. It is not necessary that every data object type has information about potential dimensions and dimension hierarchies. (0,*)- and (1,*)-relations between two or more adjacent data object types give a formal hint at hierarchies within a dimension. They must build a factual and content aggregation path for analytical purposes. The data object types *country*, *state*, *city* and *airport* form a chain of such (0,*)-relations. They describe the geographical location of an airport with a meaningful hierarchy for analytical purposes. This leads to the construction of the dimension *airport (geography)*. The distinction between start and destination airport requires the double use of the dimension *airport (geography)* to build our cube. The data object type *stage* does not have to be considered as the essential information about the start and destination airports is already included.

The data object types *aircraft type*, *aircraft*, *flight*, *flight interval* and *airline*, *aircraft*, *flight*, *flight interval* also form two chains of (0,*)-relations. They fulfill the formal criterion of a dimension hierarchy and build a meaningful aggregation path. *Flight interval* is chosen as the title of the dimension. The intersection of the data object types *aircraft*, *flight* and *flight interval* in both chains indicates a potential parallel hierarchy within the dimension. It is possible to build aggregations with *flight interval*, *flight*, *aircraft*, *airline* and *totals* as well as aggregations with *flight interval*, *flight*, *aircraft*, *aircraft type* and *totals*. In general several edges leading to the same data object type could give a hint as to the existence of parallel hierarchies within a dimension.

An important indication for appropriate candidates for dimensions from specific data object types of a conceptual model is the recurrent economic standard dimensions with the according consolidation paths ([BeSc93]). For example these are company structure (sphere of activity, organisational structure and juristical units), product structure (product family, product group, article), regional structure (country, area, customer) and customer structure (customer groups).

There is an additional dimension which is an essential part of almost every data warehouse structure. It is the *time dimension*. Due to the fact that operational systems normally do not

store any historical sales transaction for longer time periods, the time dimension can not be found in the operational data model. In some instances the time is modelled as an attribute in the relation of the corresponding measure. In the case of the flight reservation system the data object type *booking* includes the attribute *booking date* (figure 6). Normally the hierarchical structure of the time dimension is given by the analytical actuality requirements. We assume the following hierarchy: *day - month - quarter - financial year*.

Our example is restricted to these four dimensions: *time - airport (start and departure) - flight interval - booking class*. An additional dimension could be formed by the demographic aspects of the passengers.

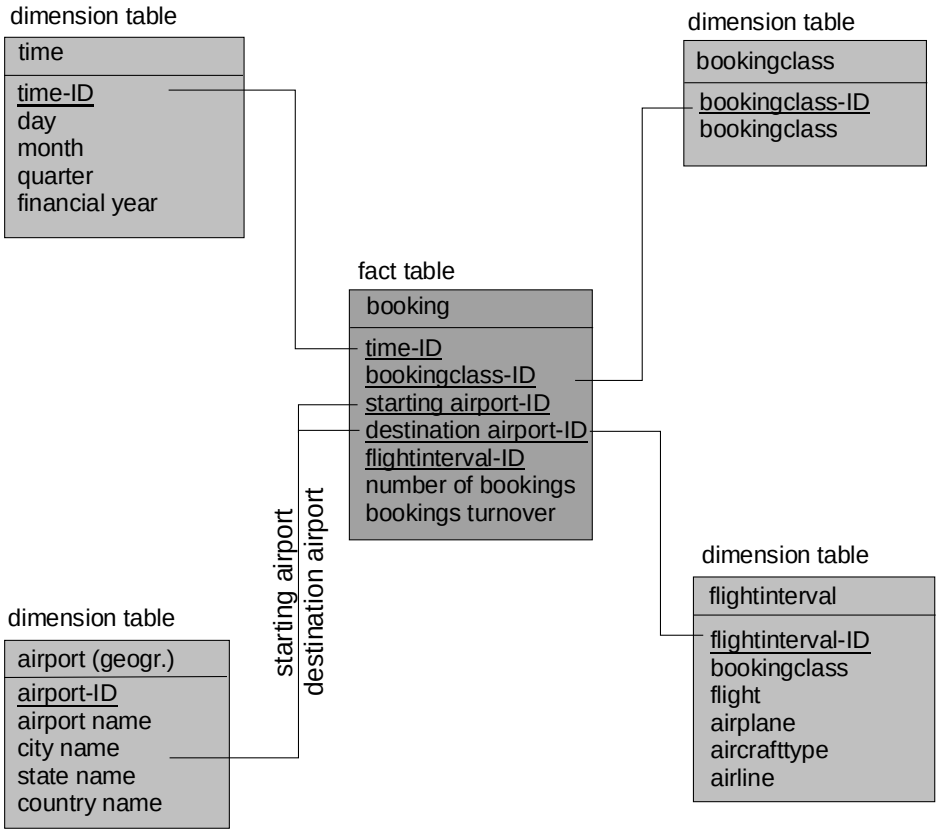


Figure 5: Star scheme of the flight reservation system

ad c) Identification of integrity constraints along the dimension hierarchies

The consolidation of data along the dimension hierarchies depends on specific integrity constraints. At best the data elements of a lower hierarchical level are summarized to the higher hierarchical level. Other more complex computation rules could also be used, e.g. the median. These consolidation rules can not be represented in standard star and snowflake schemes. The correctness of the consolidation rules can only be proven by specific analysement tools.

In figure 5 we show the transformation of the identified multidimensional data structures into the star scheme.

The fact table of the star scheme contains both selected measures „number of reservations“ and „reservation turnover“ as attributes. For each of the four dimensions a separate dimension table is build. A hierarchy level of a dimension is realized by a corresponding attribute. The attributes *flight interval*, *flight*, *aircraft*, *aircraft type* and *airline* in the dimension table *flight interval* are the hierarchy levels. The information about the order of hierarchy levels can not be depicted. The dimensions *time*, *booking class*, *airport (geography*) are designed analogous. The primary keys of the dimensions are included in the fact table and where they together form the primary key of the fact table. The primary key of the dimension *airport* is assigned twice to the fact table to distinguish between the start and destination airports. A star scheme is transformed to a snowflake scheme through denormalization of the dimension tables. A dimension hierarchy is build through separate tables. The primary key of the higher hierarchical table is inherited to the hierarchical lower table as a foreign key.

5. Summary and Outlook

In this paper we presented a new approach to derive initial data warehouse structures from the conceptual schemes of operational sources. We have shown that the data modelling technique of SERM is very suitable for this purpose because of the explicit visualisation of existency dependencies. The usability of our approach has been proven through the example of an airline. The automatic creation of multidimensional tables is not practical, because of varying requirements and creative scope. At the moment we are developing a design tool that supports semiautomatic derivation of multidimensional data structures from conceptual source schemes.

We have limited our approach to the data modelling perspective, that has to be realized by a conceptual data model. Our next step is the extension of our approach to business process

models ([Kimb97]). Furthermore the adequate representation of maintenance and consistency requirements in the data warehouse has to be investigated for an all-embracing modelling technique. This is very important for incorporating changes to the structural data within dimensions and their effect on aggregations.

Research has to be done to develop a suitable conceptual modelling technique, which will enable an adequate representation of the multidimensional world. This should form the foundation for discussions between the business and the information technology staff. For this reason a conceptual data warehouse modelling approach should be independent from the underlying database technology. Relational, multidimensional and hybrid database technologies should be served without changes to the conceptual models.

Literature

[AnMu97] Anahory, S.; Murray, D.: Data Warehousing in the Real World, Addison-Wesley, Harlow, 1997.

[Chen76] Chen, P. P.-S.: The Entity-Relationship Model – Toward a Unified View of Data, in: ACM Transactions on Database Systems, Vol. 1, No. 1, 1976, p. 9-36.

[Bulo96] Bulos, D.: A New Dimension, in: Database Programming & Design, 6/1996.

[Codd93] Codd, E. F.; Codd, S. B.; Salley, C. T.: Providing OLAP (On-line Analytical Processing to User-Analysts: An IT Mandate, E.F. Codd & Associates, White Paper, 1993.

[FeSi98] Ferstl, O. K.; Sinz, E. J.: Grundlagen der Wirtschaftsinformatik, third edition, Oldenbourg, 1998.

[GaGl97] Gabriel, R.; Gluchowski, P.: Semantische Modellierungstechniken für multidimensionale Datenstrukturen, in: HMD - Theorie und Praxis der Wirtschaftsinformatik 195, 1997, p. 18-37.

[GoMR98] Golfarelli, M.; Maio, D.; Rizzi, S.: Conceptual Design of Data Warehouses from E/R Schemes, Proceedings of the Hawaii International Conference on System Sciences, 6.-9. Januar, Kona, Hawai, 1998.

[Gros92] Gross, H.-P.: Eine semantiktreue Transformation vom Entity-Relationship-Modell in das Strukturierte Entity-Relationship-Modell; Bamberger Beiträge zur Wirtschaftsinformatik, Nr. 9, march 1992.

[Hahn98] Hahne, M.: Logische Datenmodellierung für das Data Warehouse, in: Chamoni, P.; Gluchowski, P. (editors): Analytische Informationssysteme, Springer, Berlin, 1998, p. 104-122.

[Holt98] Holthuis, J.: Der Aufbau von Data Warehouse Systemen - Konzeption - Datenmodellierung - Vorgehen, DUV, Wiesbaden, 1998

[Inmo96] Inmon, W. H.: Building the Data Warehouse, Second Edition, Wiley & Sons, New York, 1996.

[Kena95] o.V.: An Introduction to Multidimensional Database Technology, White Paper, Kenan Technologies, 1995.

[Kimb96] Kimball, R: Data Warehouse Toolkit – Practical Techniques for Building Dimensional Data Warehouse, Wiley & Sons, New York, 1996.

[Kimb97] Kimball, R.: A Dimensional Modeling Manifesto, in: DBMS online, http://www.dbmsmag.com/9708d15.html.

[LaQA97] Labio, W. J.; Quass, D.; Adelberg, B.: Physical Database Design for Data Warehousing, in: Proceedings of the 13th International Conference on Data Engineering, 7.-11. April, Binghamton, UK, 1997.

[McGu98] McGuff, F.: Hitchhiker's Guide to Decision Support, http://members.aol.com/fmcguff/dwmodel/frtoc.htm.

[Pete94] Peterson, S.: Stars: A Pattern Language for Query Optimized Scheme, http://c2.com/ppr/stars.html.

[Rade95] Raden, N.: Star Scheme 101, http://members.aol.com/nraden.str.htm.

[Rade96] Raden, N.: Modeling the Data Warehouse, http://techweb.cmp.com/iw/564/64oldat.htm.

[Sinz87] Sinz, E.J.: Datenmodellierung betrieblicher Probleme und ihre Unterstützung durch ein wissensbasiertes Entwicklungssystem, habilitation paper, Regensburg, 1987.

[Sinz88] Sinz, E.J.: Das Strukturierte Entity-Relationship Model (SER-Modell), in: Angewandte Informatik, volume 30, issue 5, 1998, p. 191-202.

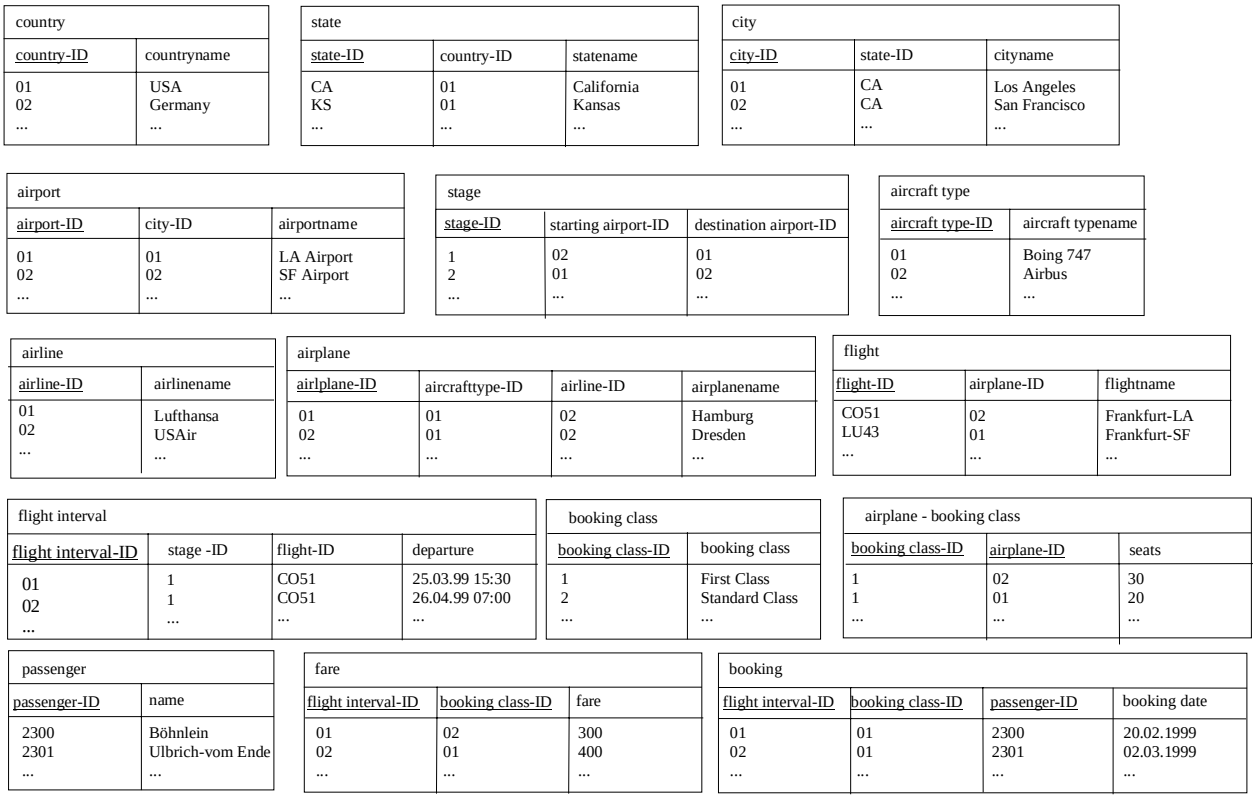
[Sinz92a] Sinz, E.J.: Datenmodellierung im Strukturierten Entity-Relationship-Modell (SERM), in: Müller-Ettrich, G. (editor): Fachliche Analyse von Informationssystemen, Addison-Wesley, Bonn, 1992.

[Sinz92b] Sinz, E.J.: Datenmodellierung im Strukturierten Entity-Relationship-Modell (SERM); Bamberger Beiträge zur Wirtschaftsinformatik, Nr. 10, May 1992.

[Voss99] Vossen, G.: Datenbankmodelle, Datenbanksprachen und Datenbankmanagementsysteme, München, 3. edition, Oldenbourg, 1999.

Appendix A

A.1 Extension of the flight reservation example



country

country-ID	countryname
01	USA
02	Germany
...	...

state

state-ID	country-ID	statename
CA	01	California
KS	01	Kansas
...	...	...

city

city-ID	state-ID	cityname
01	CA	Los Angeles
02	CA	San Francisco
...	...	...

airport

airport-ID	city-ID	airportname
01	01	LA Airport
02	02	SF Airport
...	...	...

stage

stage-ID	starting airport-ID	destination airport-ID
1	02	01
2	01	02
...	...	...

aircraft type

aircraft type-ID	aircraft typename
01	Boing 747
02	Airbus
...	...

airline

airline-ID	airlinename
01	Lufthansa
02	USAir
...	...

airplane

airlplane-ID	aircrafttype-ID	airline-ID	airplanename
01	01	02	Hamburg
02	01	02	Dresden
...	...	...	...

flight

flight-ID	airplane-ID	flightname
CO51	02	Frankfurt-LA
LU43	01	Frankfurt-SF
...	...	...

flight interval

flight interval-ID	stage -ID	flight-ID	departure
01	1	CO51	25.03.99 15:30
02	1	CO51	26.04.99 07:00
...	...	...	...

booking class

booking class-ID	booking class
1	First Class
2	Standard Class
...	...

airplane - booking class

booking class-ID	airplane-ID	seats
1	02	30
1	01	20
...	...	...

passenger

passenger-ID	name
2300	Böhnlein
2301	Ulbrich-vom Ende
...	...

fare

flight interval-ID	booking class-ID	fare
01	02	300
02	01	400
...	...	...

booking

flight interval-ID	booking class-ID	passenger-ID	booking date
01	01	2300	20.02.1999
02	01	2301	02.03.1999
...	...	...	...

Figure 6: Attributes and extension of the flight booking example

Primary key attributes are underlined. Foreign key attributes are explicitly part of the data object types. For each data object type a possible extension consisting of two data objects is shown.

A.2 Corresponding structures in ERM and SERM

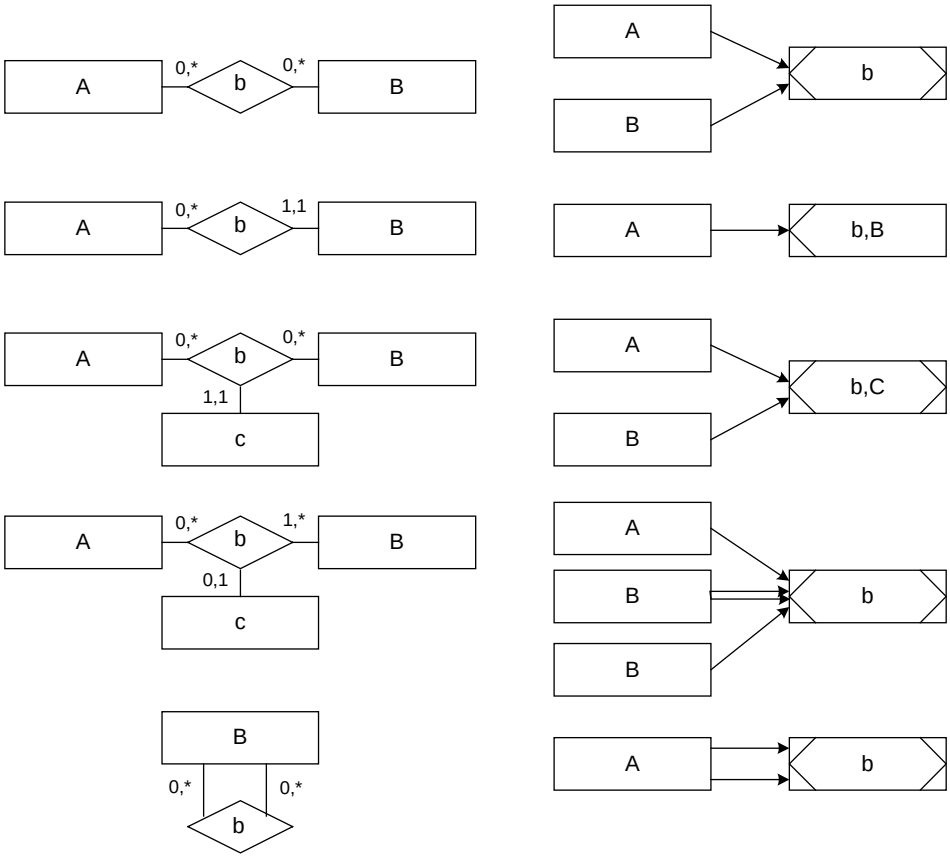


Figure 7: Corresponding structures in ERM and SERM

A.3 ER-diagram of the flight reservation example:

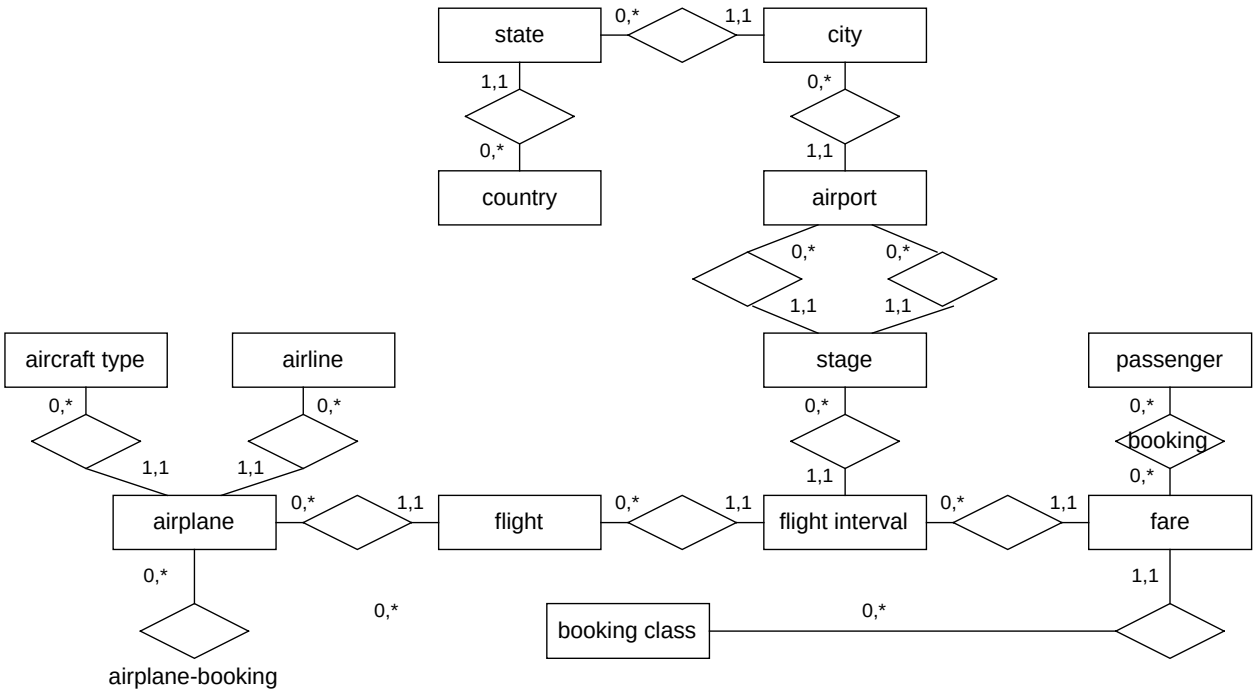


Figure 8: ER-diagram of a flight reservation system

A.4 Algorithm for the closure of existence prerequisites C_{EX}

Input: Set of data object types **U**,

Initial data object types $\mathbf{X} \subseteq \mathbf{U}$

Set of existence prerequisites[2] **F** of **U**

Output: $\mathbf{C_{EX}}(X)$ for **F** and **U**

Algorithm: $X^{(0)} := X$;

$i := 0$;

REPEAT

$i := i + 1$;

$X^{(i)} := X^{(i-1)} \cup$

$\{A \mid (Y,Z) \in F, A \in Y, Z \subseteq X^{(i-1)}\}$

until $X^{(i)} = X^{(i-1)}$;

$\mathbf{C_{EX}} = X^{(i)}$.

[2] Element (Y,Z) of F means, that data object type Y is a existence prerequisite for data object type Z