Proceedings of the 5th WSEAS Int. Conf. on Artificial Intelligence, Knowledge Engineering and Data Bases, Madrid, Spain, February 15-17, 2006 (pp342-347)

Design and Implementation of a National Data Warehouse

Carlo DELL'AQUILA, Ezio LEFONS, and Filippo TANGORRA
Dipartimento di Informatica
Università di Bari
via Orabona 4, 70126 Bari
ITALY

Abstract: - The Information Technology is an essential support in the decisional process to improve manager' phenomena knowledge, that is often approximate and ill-structured. Tools underlying decision support systems (as OLAP systems, data mining, and data warehouses) have a central role in enterprise information systems. In this paper, we present the design and the implementation of a national data warehouse.

Key Words: - Data marts, data warehouses, train booking, railway transportation analysis.

1 Introduction

In last years, there has been an explosive growth in the use of data warehousing technology in order to construct successful decision support infrastructures [2, 4, 11, 12]. For many years, the quest for competitive advantages has prompted many organizations to attempt the paradigm to shift from data processing to the new exciting arena of information analysis. Increasing successful implementations, more robust and functional extraction software, improving price-to-performance equipment ratios, and improved training for IT staff are surely data warehousing growth motivators.

Data warehouse (DW) is a sophisticated system that gives benefits only to organizations with a high degree of IT maturity. An organization embracing data warehousing prematurely, *i.e.*, while it is still working to meet its operational information needs, can easily meet obstacles. Poor support from the management staff, in a lot of cases unable to appreciate the need of data warehousing, or insufficient resources and funds owing to pressing needs for operational systems could be pitfalls. Moreover, user expectations could not be met due to immature operational systems and insufficient resources, unable to allow the development of satisfactory ETL processes.

Once the warehouse is built, it has to be maintained and tuned [13]. In fact, because of the heterogeneity of sources, data coming into the warehouse can change their form in time. Likewise, user's needs can change in time. In addition, historical data inevitably grow in time. For these all, a warehouse needs constant tuning.

Here, we present some guidelines and principles both in the design and the implementation of the data warehouse used to accomplish our national railway transportation analyses. The study has been conducted throughout TeleSistemi Ferroviari (*TSF*), an Italian IT solutions provider. The overall architecture of the data warehouse is briefly presented in Section 2. In Section 3, two typical warehouse schemas - namely, the Star schema and the Snowflake schema - with their entities (fact and dimension tables) are discussed. Section 4 deals with the warehousing design from the database viewpoint. Then, the conversion of data collected during the logical design into physical database structures is dealt with materialized views. Section 5 contains conclusions.

2 Data warehouse architecture

The architecture overview of the referring application context - the Italian railway system for the booking process [5, 10]-is shown in Figure 1.

It comprises the usual different levels of data flow corresponding to the sequence of steps for adapting the data to the decision-maker needs [3, 7-12].

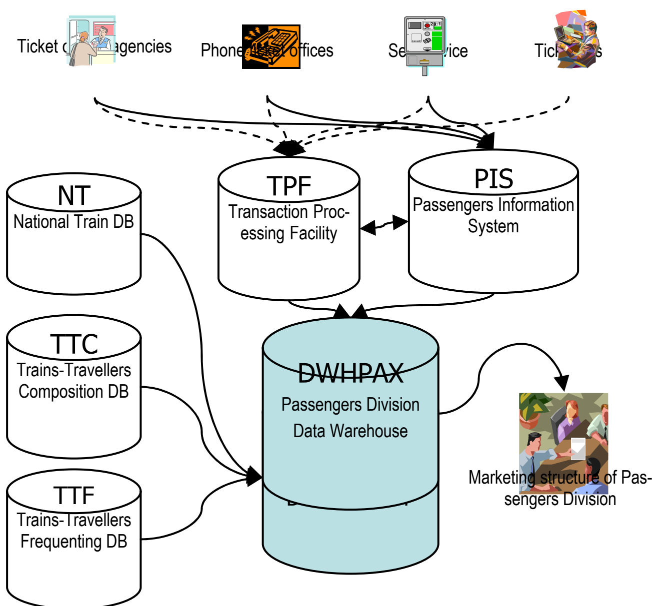

Fig. 1. Italian railway DW architecture.

The source data flows dealt with by Transaction Processing Facility (*TPF*) come from the Sale System of the operational Passengers Information System (*PIS*) and from the set of information concurring in the booking analysis. Besides *PIS/TPF*, the other operational systems involved are the National Train database (*NT*), which contains the train route and railway kilometres, the Trains-Travellers Composition database (*TTC*), which contains saleable delivered services, associated trains and antenna trains (coupling/release), and the Trains-Travellers Frequenting database (*TTF*), which contains the train registry. Further details can be found in [9-10].

3 The design process

The design process must be oriented to the end users' needs through the two main types of objects commonly used in dimensional data warehousing, *viz.*, fact and dimension tables.

Fact tables are the large tables in the warehouse schema that store business measurements. Fact tables typically contain facts and foreign keys to the dimension tables. Fact tables represent data, usually numeric and additive, that can be analyzed and examined. Therefore, the fact table has columns that contain both numeric facts (measurements) and foreign keys (FK) to dimension tables. The fact table contains either detail-level facts or facts that have been aggregated (summary tables).

A dimension table is a structure, often composed of one or more hierarchies, that categorizes data. Several distinct dimensions, combined with facts, enable to answer business questions. Dimension data are collected at the lowest level of detail and then hierarchically aggregated into higher level totals that are more useful for analysis. Data warehouses use a dimensional model where multidimensional data structures are basic elements. Multidimensional data structures are based on the separation of quantitative and qualitative data.

The *star schema* is the most common and natural way to model the data warehouse. In fact, only one join has to be executed to establish the relationship between the fact table and any one of the dimension tables. The star schema shown in Figure 2 represents a part of the data marts of the data warehouse of the customer *Trenitalia*, the Italian main train service company, realized in collaboration with *TSF* (railway telesystems company), the IT solutions provider. The central node, or the CMM_DMT_TPF_TTC_TPF_ELE table, is the fact while the other tables are the dimensions. Their semantics is reported in Tables 1 to 5, respectively.

On the other hand, a *snowflake schema* is a transformation of a star schema based on the third normal form. Redundancies are eliminated by the normalization of the dimension tables, by grouping dimension data into multiple tables instead of one large table. For every dimension hierarchy, there exists a separate table (dimension table). Figure 3 is an example of snowflake schema extracted from the overall data marts model. The central node, or the fact, is the CMM_DMT_TPF_TPF table and has five dimension nodes. Their semantics is reported in Tables 6 to 8, 3, 4, and 9, respectively. On its hand, the CMM_DMT_TPF_ANA_TRE_TPF_GIO_MVT dimension has been implemented with a materialized view and it has five "sub-dimensions" tables whose semantics is reported in Tables 10, 5, and 11 to 13, respectively.

The complete data marts ER model of the national railway data warehouse is given in Figure 4.

4 Physical design

Physical design decisions are mainly driven by query performance and database maintenance aspects. During the logical design phase, a model for the data warehouse consisting of entities, attributes and relationships is created, as shown in the previous section.

During the physical design process, the translation of the expected logical schemas into actual database structures occurs. The structures mentioned are those provided by the Oracle DBMS, which is used by *TSF* provider for the implementation of the data warehouse subject of the present case study [1, 6, 14-15].

Tables and Partitioned Tables

Tables are the basic unit of data storage. They are the containers for the expected amount of raw data in a data warehouse.

Using partitioned tables instead of nonpartitioned ones addresses the key problem of supporting very large data volumes by allowing their decomposition into smaller and more manageable pieces. The main design criterion for partitioning is manageability, though performance benefits will be seen in most cases because of partition pruning or intelligent parallel processing.

Partitioned tables allow data to be broken down into smaller, more manageable pieces called partitions, or even subpartitions. Each partition can be managed individually, and can function independently of the other partitions, thus providing a structure that can be better tuned for availability and performance. When using parallel execution, partitions provide another means of parallelization. Operations on partitioned tables are performed in parallel by assigning different parallel execution servers to different partitions of the table.

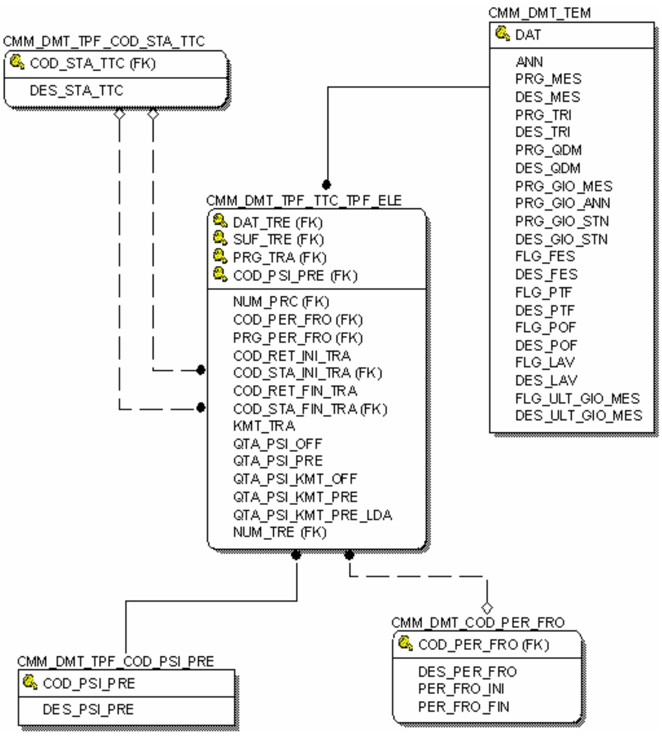

Fig. 2. A Star Schema.

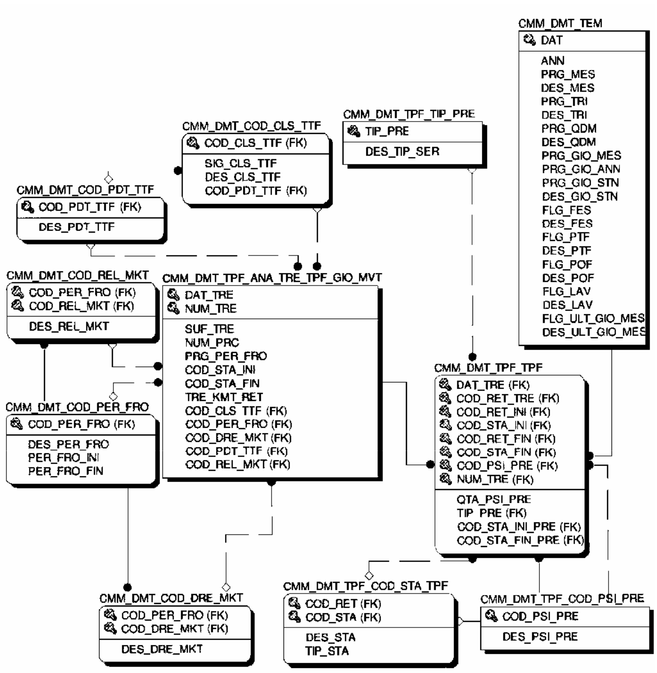

Fig. 3. A Snowflake schema.

CMM_DMT_TPF_TTC_TPF_ELE FACT TABLE	
FIELD NAME	**DESCRIPTION**
DAT_TRE (FK)	Train Date
SUF_TRE (FK)	Train Suffix
PRG_TRA (FK)	Elementary Route Number of the Train Route Sort
COD_PSI_PRE (FK)	Booked Seat Code
NUM_PRC (FK)	Train Route Number
COD_PER_FRO (FK)	Railway Period Code
PRG_PER_FRO (FK)	Railway Period Number
COD_RET_INI_TRA	Elementary Route Leaving Railway Code
COD_STA_INI_TRA (FK)	Elementary Route Leaving Station Code
COD_RET_FIN_TRA	Elementary Route Final Railway Code
COD_STA_FIN_TRA (FK)	Elementary Route Final Station Code
KMT_TRA	Railway Kilometre Train
QTA_PSI_OFF	Total Number of Delivered Seats
QTA_PSI_PRE	Total Number of Booked Seats
QTA_PSI_KMT_OFF	Total Number of Delivered Seats per Km
QTA_PSI_KMT_PRE	Total Number of Booked Seats per Km
QTA_PSI_KMT_PRE_LFA	Total Number of Booked Seats per Km in the Waiting List
NUM_TRE (FK)	Train Number

Tab. 1. *TPF/TTC* elementary fact data mart.

CMM_DMT_TPF_COD_STA_TTC DIMENSION TABLE	
FIELD NAME	**DESCRIPTION**
COD_STA_TTC (FK)	TTC Station Code
DES_STA_TTC	TCC Station Name

Tab. 2. *TTC* station data mart.

CMM_DMT_TEM DIMENSION TABLE	
FIELD NAME	**DESCRIPTION**
DAT	Date
ANN	Year
PRG_MES	Month Number
DES_MES	Month Name
PRG_TRI	Quarter Number
DES_TRI	Quarter Description
PRG_QDM	Four-month Period Number
DES_QDM	Four-month Period Description
PRG_GIO_MES	Day Number in the Month
PRG_GIO_ANN	Day Number in the Year
PRG_GIO_STN	Day Number of the Week
DES_GIO_STN	Day Name
FLG_FES	Holiday Flag
DES_FES	Holiday Description
FLG_PTF	Extra Long Holiday Flag
DES_PTF	Extra Long Holiday Description
FLG_POF	After-holiday Flag
DES_POF	After-holiday Description
FLG_LAV	Workday Flag
DES_LAV	Workday Description
FLG_ULT_GIO_MES	Last Day of the Month Flag
DES_ULT_GIO_MES	Last Day of the Month Description

Tab. 3. Time data mart.

CMM_DMT_TPF_COD_PSI_PRE DIMENSION TABLE	
FIELD NAME	**DESCRIPTION**
COD_PSI_PRE	Booked Seat Code
DES_PSI_PRE	Booked Seat Description

Tab. 4. Booked place data mart.

CMM_DMT_TPF_COD_PER_FRO DIMENSIONTABLE	
FIELD NAME	DESCRIPTION
COD_PER_FRO (FK)	Railway Period Code
DES_PER_FRO	Railway Period Description
PER_FRO_INI	Start Railway Period
PER_FRO_FIN	End Railway Period

Tab. 5. Railway period data mart.

CMM_DMT_TPF_TPF FACT TABLE	
FIELD NAME	DESCRIPTION
DAT_TRE (FK)	Train Date
COD_RET_TRE (FK)	Train Railway Code
COD_RET_INI (FK)	Leaving Station Railway Code
COD_STA_INI (FK)	Leaving Station Code
COD_RET_FIN (FK)	Final Station Railway Code
COD_STA_FIN (FK)	Final Station Code
COD_PSI_PRE (FK)	Booked Seat Code
NUM_TRE (FK)	Train Number
QTA_PSI_PRE	Booked Seats Total Number
TIP_PRE (FK)	Booking Type
COD_STA_INI_PRE (FK)	Booked Leaving Station Code
COD_STA_FIN_PRE (FK)	Booked Final Station Code

Tab. 6. Transaction Processing Facility data mart.

CMM_DMT_TPF_ANA_TRE_TPF_GIO_MVT DIMENSION TABLE	
FIELD NAME	DESCRIPTION
DAT_TRE	Train Date
NUM_TRE	Train Number
SUF_TRE	Train Suffix
NUM_PRC	Train Route Number
PRG_PER_FRO	Railway Period Number
COD_STA_INI	Leaving Station Code
COD_STA_FIN	Final Station Code
TRE_KMT_RET	Railway Kilometre Train
COD_CLS_TTF (FK)	Train Classification Code
COD_PER_FRO (FK)	Railway Period Code
COD_DRE_MKT (FK)	Marketing Directrix Code
COD_PDT_TTF (FK)	Product Code
COD_REL_MKT (FK)	Marketing Relation Code

Tab. 7. Daily train movement data mart.

CMM_DMT_TPF_TIP_PRE DIMENSION TABLE	
FIELD NAME	DESCRIPTION
TIP_PRE	Booking Type
DES_TIP_SER	Service Type Description

Tab. 8. Booking type data mart.

CMM_DMT_TPF_COD_STA_TPF DIMENSION TABLE	
FIELD NAME	DESCRIPTION
COD_RET (FK)	Station Railway Code
COD_STA (FK)	Station Code
DES_STA	Station Name
TIP_STA	Station Type

Tab. 9. Railway station data mart.

CMM_DMT_COD_DRE_MKT DIMENSION TABLE	
FIELD NAME	DESCRIPTION
COD_PER_FRO (FK)	Railway Period Code
COD_DRE_MKT (FK)	Marketing Directrix Code
DES_DRE_MKT	Marketing Directrix Description

Tab. 10. Directrix marketing data mart.

CMM_DMT_COD_REL_MKT DIMENSION TABLE	
FIELD NAME	DESCRIPTION
COD_PER_FRO (FK)	Railway Period Code
COD_REL_MKT (FK)	Marketing Relation Code
DES_REL_MKT	Marketing Relation Description

Tab. 11. Relation marketing data mart.

CMM_DMT_COD_PDT_TTF DIMENSION TABLE	
FIELD NAME	DESCRIPTION
COD_PDT_TTF (FK)	Product Code
DES_PDT_TTF	Product Description

Tab. 12. *TTF* product data mart.

CMM_DMT_COD_CLS_TTF DIMENSION TABLE	
FIELD NAME	DESCRIPTION
COD_CLS_TTF (FK)	Train Classification Code
SIG_CLS_TTF	Train Classification Signature
DES_CLS_TTF	Train Classification Description
COD_PDT_TTF (FK)	Product Code

Tab. 13. *TTF* classification data mart.

Partitions and subpartitions of a table all share the same logical attributes. However, they can have different physical attributes (such as the belonging tablespaces). Storing partitions in separate tablespaces enables: the reduction of the possibility of data corruption in multiple partitions, backup and recover independent for each partition, controlling the partitions mapping to disk drives (important for balancing I/O load), improving manageability, availability and performance.

Materialized Views

Materialized views are used in data warehouses to increase the speed of queries on very large databases. Queries to large databases often involve joins between tables and/or aggregations such as SUM, which are expensive operations in terms of time and processing power. The type of materialized view determines how the materialized view is refreshed and used by query rewrite. Materialized views can be created also in order to replicate data, as snapshots. When used for query rewrite, they improve query performance by precalculating expensive aggregations and joins prior to execution and storing the results in the database.

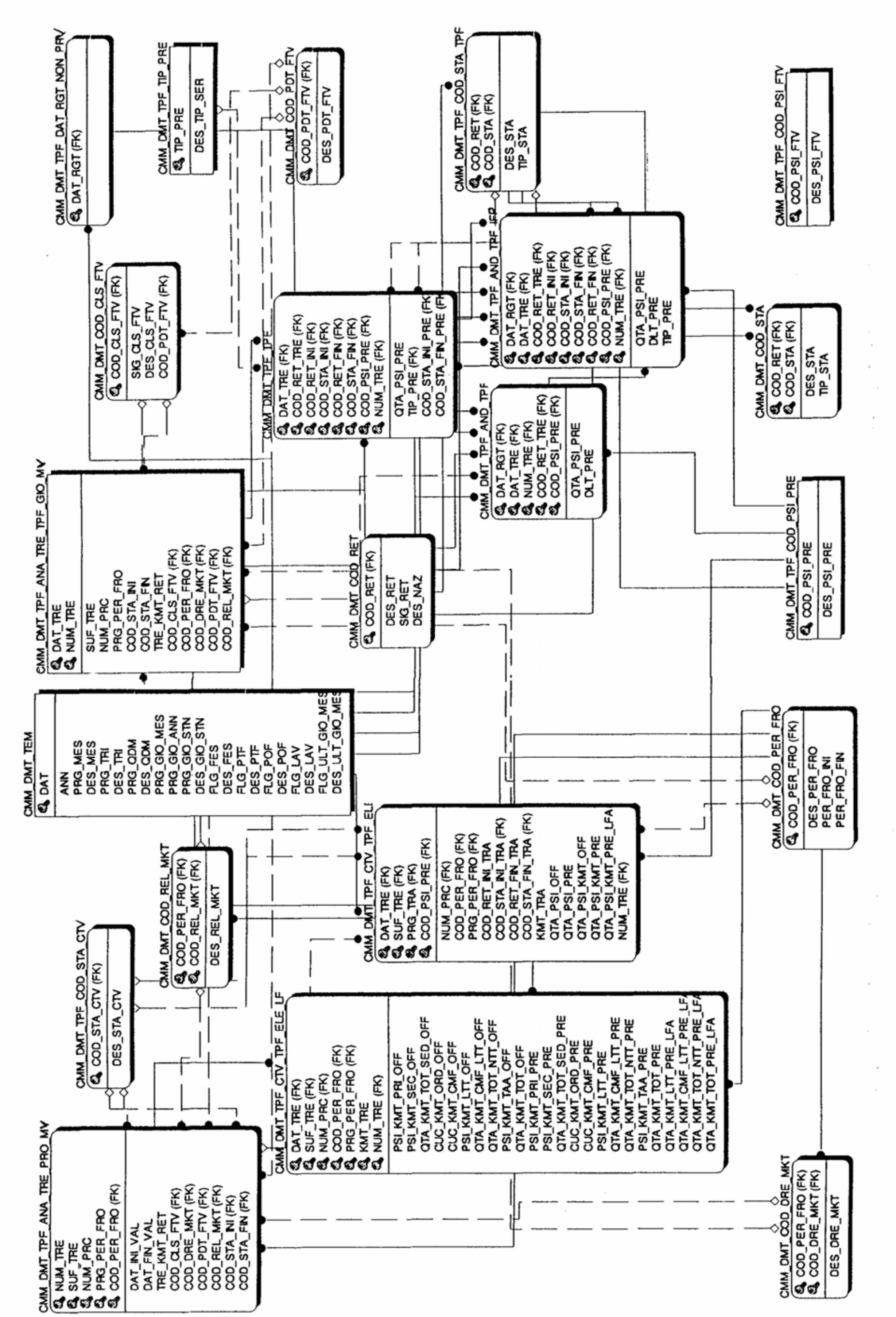

Fig. 4. Data Marts ER Model.

The query optimizer automatically recognizes when an existing materialized view can and should be used to satisfy a request. If the recognition is successful, then the data are taken directly from the materialized views and not from the underlying tables.

Summaries and query rewrite
Typically, data flow from one or more online operational sources (OLTP databases) into a data warehouse on a monthly, weekly, or daily basis. Usually, the vast majority of the data is stored in a few very large fact tables. One technique employed in data warehouses to improve performance is the creation of summaries. Summaries are special kinds of aggregate views that improve query execution times by pre-calculating expensive joins and aggregate operations prior to execution and storing the results in a table. These can be implemented with materialized views, which can perform a number of roles, such as improving query performance or providing replicated data, *i.e.*, behaving like snapshots.
The type of materialized view determines how the materialized view is refreshed and used by query rewrite. When an existing materialized view satisfies a request, queries are moved to it and not to the underlying table: this improves response. A materialized view definition can include a number of aggregations (sum, count, average, variance, min, max) and joins. If it has to be used by query rewrite, then it must be stored in the same database as the fact or detail tables on which it relies.
The materialized views can be with aggregates and/or with only joins. In data warehouses, materialized views normally contain aggregates whereas some ones contain only joins and no aggregates. The advantage of the latter is that expensive joins will be pre-calculated.

5 Concluding remarks

Decision support systems technology and applications evolved significantly in last years. The design and the implementation of a national data warehouse presented in this paper address some of problems of this area. In particular, data warehousing is useful for organizations producing or manipulating large or huge amounts of data that need to be suitably and profitably analyzed. Nevertheless, organizations attempting to embrace data warehousing must have a high IT maturity degree, because of the costs that data warehouse maintenance implies in terms of constant optimization due to the frequent modifications of user's needs. For increasing the diffusion of this profitable technology, we claim the need of a standardization activity for the life cycle of data warehouse building in order to limit their costs.

References

[1] R. Baylis, K. Rich, and J. Fee, *Oracle 9i Database Administrator's Guide*, Release 1 (9.0.1), Oracle Corporation 2001.

[2] M. Boehnlein and A. Ulbrich-vom Ende, Deriving Initial Data Warehouse Structures from Conceptual Data Models of the Underlying Operational Information Systems, *DOLAP '99* Proc. *ACM,* pp. 15-21.

[3] S. Chaundhuri, U. Dayal, and V. Ganti, Database technology for decision support systems, *IEEE Computer*, Vol. 34, No 12, 2001, pp. 48-55.

[4] C. P. Chua and R. Green, *Data Warehousing Fundamentals*, Oracle Corporation 1999.

[5] L. Cupertino, *Building a Successful Data Warehouse: Design, Implementation, and Tuning*, Thesis dissertation 2005.

[6] M. Cyran and C. Dialeris Green, *Oracle 9i Database Performance Guide and Reference*, Release 1 (9.0.1), Oracle Corporation 2001.

[7] C. dell'Aquila, E. Lefons, and F. Tangorra, Decisional portal using approximate query processing, *WSEAS Transactions on Computers*, Vol. 2, No 2, 2003, pp. 486-492.

[8] C. dell'Aquila, E. Lefons, and F. Tangorra, Approximate query processing in decision support system environment, *WSEAS Transactions on Computers*, Vol. 3, No 3, 2004, pp. 581-586.

[9] C. dell'Aquila, E. Lefons, and F. Tangorra, Backend and Front-end Solutions for a Data Warehousing Case Study, *WSEAS Transactions on Computers*, vol. 4, no. 10, 2005, pp. 1259-1269.

[10] C. dell'Aquila, E. Lefons, and F. Tangorra, Data warehousing System for National Railway Traffic, *WSEAS Transactions on Information Science and Applications*, vol. 2, no. 6, 2005, pp. 726-735.

[11] M. Janesch, *Implementing the Best Data Warehousing Tuning Techniques for Your Environment*, Innovative Consulting 2001.

[12] M. Jarke, M. Lenzerini, Y. Vassiliou, and P. Vassiliadis, *Fundamentals of Data Warehouses*, Springer-Verlag, 2003.

[13] R. Kimball and M. Ross, *The Data Warehouse Toolkit*, 2nd edition, John Wiley & Sons, 2002.

[14] P. Lane, V. Shupmann, *Oracle 9i Warehousing Guide*, Release 1 (9.0.1), Oracle Corporation 2001

[15] L. McGeen Lusher, *Oracle 9i Database Concepts*, Release 1 (9.0.1), Oracle Corporation 2001.