

Book Title: New Trends in Database Systems. Methods, Tools, Applications

Chapter: Data warehouse and OLAP technology

Author: Marta E. Zorrilla Pantaleón

Department of Mathematics, Statistics and Computation, University of Cantabria.

Avda. de los Castros s/n 39005 Santander, Spain

marta.zorrilla@unican.es

Table of contents

1.	Business Intelligence overview	2
2.	Data warehouses	3
2.1.	Components of a Data warehouse	4
2.2.	Data warehouse lifecycle.....	6
2.3.	“e-Learning data webhouse” case study.....	7
2.4.	Dimensional modelling	9
2.4.1.	Fact and dimension tables	10
2.4.2.	Additional considerations.....	11
2.5.	Data warehouse vs data mart.....	13
3.	OLAP technology.....	14
3.1.	Concepts and terminology	14
3.2.	OLAP cube design.....	16
3.3.	OLAP storage: MOLAP, ROLAP, HOLAP.....	17
4.	Business Intelligence Applications.....	19
5.	Summary.....	21
6.	References	21

1. BUSINESS INTELLIGENCE OVERVIEW

In a market as competitive and global as the current one, information has become one of the main managerial assets. Until recently, the automated management of business processes (Invoice Management, Supply Chain Management, Enterprise Resource Planning, and so on) of companies and organizations by means of transactional systems had been sufficient to meet their information needs and do their work, but now they need to integrate and handle all these information in order to analyze it in line with the business aims that are to be achieved at every stage.

To help organizations with these tasks, at the end of the nineties, Business Intelligence (BI) tools appeared in the software market, which enable the handling, consolidating and analyzing of large volumes of data, transforming these into valuable information for decision-making. In other words, these tools help analysts, managers and executives to have a more comprehensive knowledge of the factors affecting their business (metrics, key performance indicators, behaviour patterns, analysis of trends ...). Thus, they can modify their business processes in order to achieve targets such as increasing sales, reducing fraud, improving client relationship, increasing quality of the services offered and reducing costs.

In short, the Business Intelligence field provides companies with a framework to:

- Define and measure business key performance indicators (KPI) and understand their behaviour.
- Process, summarize, report and distribute the relevant information on time.
- Manage and share business knowledge with the organization.
- Analyse and optimise the processes that act on business key performance indicators.

In order to carry out these tasks, Business Intelligence solutions encompass a wide range of techniques and technologies, as we will see in section 2.1: the data warehouse database as integrated repository of strategic information, the OLAP (On-Line Analytical Processing) technology for the exploration of information under different perspectives, dashboard, scorecard and reporting tools for the analysis and visualization of information and trends, and data mining techniques to discover meaningful patterns and rules in large volumes of data by automatic or semi-automatic means. Before continuing, it should be said that some authors, such as Kimball [12] consider that the “data warehouse” is the platform for business intelligence (DW/BI); however other authors such as Inmon [5] consider that the data warehouse is simply the database where the business data are consolidated and stored. In this work, we follow Kimball’s idea.

Although Business Intelligence tools appeared to help companies tackle the most difficult business decisions, nowadays they are used in many other areas, such as in scientific applications (astronomy, bioinformatics, drug discovery, ...), in governmental applications (surveillance, crime detection, profiling tax cheaters, ...) or in the Web (search engines, e-commerce, e-learning, web and text mining, ...).

This chapter focuses on the explanation of data warehousing and OLAP technologies from a practical point of view, using a real case study developed in the University of Cantabria,

a BI solution to monitoring and analysing the learners' behaviour in e-learning platforms, such as WebCT, Moodle or Caroline.

This chapter is organized as follows. First, we introduce the data warehousing field, justifying the necessity of its implementation to help organizations analyze their data. Next, we describe the data warehouse architecture and indicate the steps for its design and building. After that, we summarize the basic concepts of dimensional modelling, the data model on which the data warehouse database is based. Then, we study OLAP technology, which enables analysts, managers, and executives to gain insight into data through fast, consistent, interactive access to a wide variety of possible views of information and, finally, we describe different alternatives to build BI applications with which the end-users analyse and assess the behaviour of their business indicators.

2. DATA WAREHOUSES

Now that we know “how” data warehousing can contribute to helping companies, we must ask ourselves why they are necessary when corporations have operational systems in which their data are stored conveniently and which, besides, allow them to manage their daily activity. The question is why is it necessary to design, build and maintain another information system? Can they not use the current ones? There are several reasons which justify it:

- First, the information in companies is usually distributed in different data sources and, very often, the same information is gathered in a variety of formats, codes, etc. Even more, inconsistent data and duplicated registers are generally found. For this reason, a new repository is required so that all this information is consistent, organized and standardized.
- Besides, the aim of each data source is not oriented towards making decisions but to registering transactions (retail sales, order management, human resources management, etc.) This is why it is necessary to extract and transform some of the operational data to calculate the key indicators with which analysts measure business performance.
- On the other hand, this operational information is generally stored in relational databases normalized until third normal form, whose structure is not suitable to answer rapidly complex queries (accesses to several tables, aggregation and summarization operations...).

All these reasons lead us to the designing of a specific information system driven by the needs of business users, fed from operational data sources and built and presented from a simple perspective.

As can be observed succinctly in Table 1, data warehouses have got profoundly different needs, clients, structures and rhythms to the operational systems.

To conclude, it must be said that one cannot make the mistake of building a data warehouse as a mere copy of the data from the operational systems stored on an independent hardware platform for performance reasons. It must offer users useful, consistent, understandable information by means of expressive and easy-to-use graphic interfaces.

Table 1. OLTP vs OLAP

	Operational (OLTP)	Analytical (OLAP)
Users	Clients and administrative and technical professionals	Executive and Business professionals
Function	Daily operative (transactional)	Making decisions (analytical)
Data	Dynamic, current, detailed, process-oriented	Static, historical, aggregated, consolidated, business-oriented
# users	Thousands	Hundreds
# acceded rows	Dozens	Millions
DB size	100MB-GB	100GB-TB
DB design	Transactional (3NF or greater). Normalized. Application-oriented.	Simple database design (usually dimensional). De-normalized. Subject-oriented.
Metric	Transaction throughput	Query throughput, response

2.1. Components of a Data warehouse

Now that the goals and the main characteristics of data warehouses have been outlined, the components that make up a complete warehousing environment must be described. As illustrated in Figure I, there are four components: data sources, data staging area, data area, data access tools.

Data Sources: These store the transactions of the business. Although they are generally internal to the organization, they can also include external sources such as surveys, studies carried out by statistical centres, or files proceeding from the web, etc. These sources must never be modified, are only queried in order to extract the information of interest.

Data Staging Area: This area is composed of a storage space and the set of processes, commonly called ETL processes [7], with which data are extracted, transformed and loaded in the *data area*, and later also updated as a consequence of the necessary synchronization tasks between the operational and analytical systems. This storage area is used to transform data (correcting misspellings, resolving domain conflicts, dealing with missing elements or parsing into standard formats), integrate data from different sources, write down audit information (when the last load was done, how many rows were added or updated, ...) and assign warehouse keys, among others.

Its physical structure is not necessarily normalized; it must answer to the needs of the ETL processes. The design and the programming of these processes are costly in terms of time and money. These tasks can take up 60-70% of the project time, even more, the data warehouse success depends, to a great extent on them, because a failure in the definition of the process can lead to showing wrong data to executives and, consequently, inappropriate decisions are made.

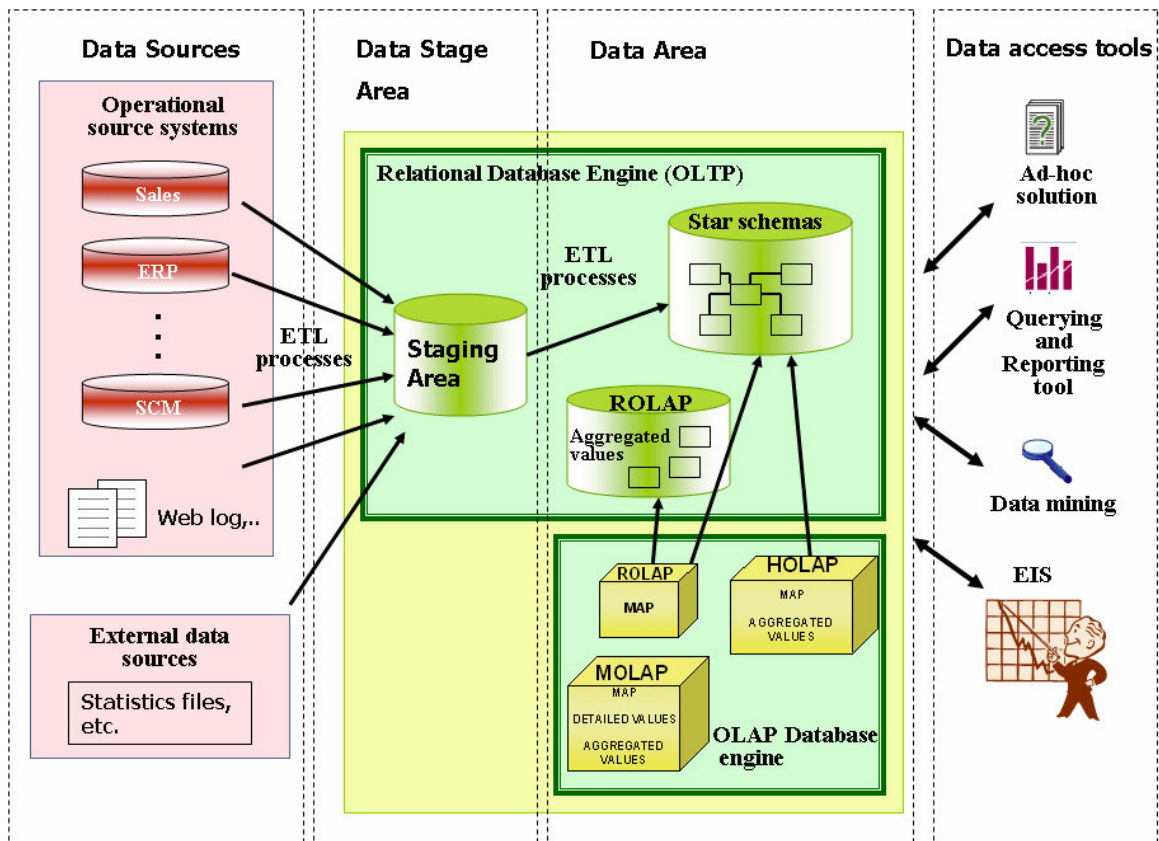


Figure I. Components of a data warehouse

Data area: Data area is where data are organized, stored, and made available for direct querying by users, reports and other analytical applications. The information is not only stored in detail but also summarized according to the different business perspectives. For example, the admissions department of a university will need to know the number of students registered in each degree in every academic year, whereas the professorship will be interested in knowing how this same number can be broken up by academic level or by age band.

These data can be stored either in a relational database management system or in a multidimensional system (OLAP), although generally both are used: the relational database, to store the detail information and the OLAP database, to build the cubes which include the aggregations and the indicators whose calculation is complex such as average years to graduate or number of drop-outs. The OLAP engine usually feeds from the relational database.

In both engines, the physical data model follows a dimensional structure based on star and snowflake schemas. Here is where Bill Inmon's and Ralph Kimball's perspectives differ. For Bill Inmon [4], the information in the data warehouse is stored in third normal form, whereas for Ralph Kimball [12], the information is always stored in a dimensional model. According to Kimball, the use of normalized modelling in the data area defeats the whole purpose of data warehousing, namely, understandable, resilient to change and high-performance retrieval of data.

Data access tools: These are the tools that allow business users access to the information stored in the data warehouse database by means of intuitive, graphical and easy-to-use

interfaces. These are mainly focused on the data analysis through queries and reports and the knowledge extraction by means of semi-automatic techniques (data mining).

There are different alternatives for developing the applications which allow analysts and executives to analyse the strategic data. One can choose commercial querying and reporting products such as Microstrategy, Cognos, Business Objects, Actuate, and so on which enable the rapid development of end-user tools, or build ad-hoc tools using specific APIs such as JDBC, OLEDB, OLEDB for OLAP or XML for Analysis. On the other hand, there are also the Executive Information Systems (EIS) which are packaged solutions generally focusing on a certain subject (fraud detection, promotion effectiveness...). These require a fair amount of customization.

Related to knowledge extraction, data mining tools such as Weka, Clementine or DBMiner allow us to explore large volumes of data in search of useful and hidden patterns and predictable information that an expert cannot extract directly, showing this knowledge by means of meaningful patterns and/or rules. Their integration in the end-user applications is generally ad-hoc.

Approximately, between 80 and 90 percent of the potential users will be served by pre-built parameter-driven analytic applications, with only a few using the query capabilities or knowledge extraction.

2.2. Data warehouse lifecycle

It is not easy to face the task of designing and developing a Business Intelligence solution for the first time. That is why it is advisable to have a process model which allows one to know the different tasks that have to be done and the deliverables that must be obtained.

The proposed lifecycle is an iterative approach based on seven steps:

1. Identify business requirements and their associate value. This is generally gathered in a requirements document which will include, at the very least, a list of data elements, example questions and a list of desired reports that would help to answer the analytical questions. This can also include information about data sources (availability, constraints, quality...) which will be used and, should it exist, a list of software and hardware requirements.
2. Design a single, integrated, easy-to-use, high performing information model that gathers the identified business requirements and build it physically in a relational database. It is recommended that this follow a dimensional schema.
3. Design the data stage schema and programme the ETL processes to load data in the dimensional schema. Likewise, implement the synchronization processes between data sources and the data warehouse database and, also develop the auditing system.
4. Design and build the OLAP cubes with the metrics, the complex calculations and the key performance indicators that the business experts require to make decisions.
5. Develop the user interface that allows business users to exploit the data (query tools, static and dynamic reports, scorecards, and so on) and extract the knowledge (patterns, rules...).
6. Define the system maintenance plan and,
7. Test the BI solution developed and deploy it.

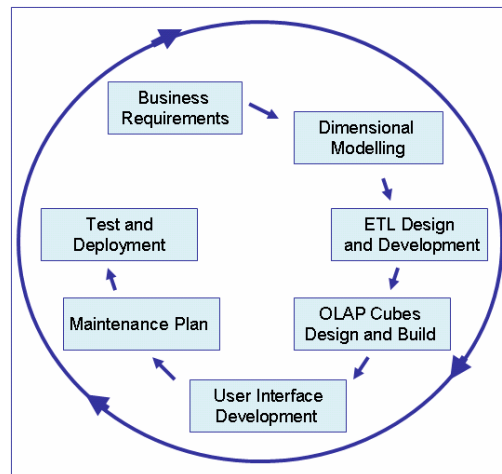


Figure II. DW/BI lifecycle

As in the development of any software solution, there are, of course, other transversal tasks to be carried out such as the definition and management of the project plan (phases, milestones, human resources, deliverables, risks and contingencies) and the selection of the development and exploitation tools.

Undertaking a global project on Business Intelligence is a risky task that, in many real cases, has led to failure. Therefore, it is advisable to do it by business areas. From amongst them all, we have to choose that which is most likely to succeed and, before starting the project, it is good to prepare the staff mentally creating realistic expectations in them.

Another important aspect to guarantee the project's success is the fact of involving operational users, IT technicians and business analysts so that the different but complementary viewpoints can allow the development of a better system. Whereas users guarantee the quality of data, IT technicians provide the computer systems' support and help in data sources description, and experts and business analysts assure the quality of the end-product.

2.3. "e-Learning data webhouse" case study

The best way to understand the principles of dimension modelling is to work with real cases. Here we propose a case developed in the University of Cantabria, with which the learners' behaviour in e-learning platforms is assessed ([22][23][24]).

Nowadays, most universities, colleges and high schools are virtualising their teaching through e-learning platforms benefiting from their many advantages such as work "any-time, anywhere", use collaborative tools, support different styles of learning (collaborative learning, discussion-led learning, student-centred learning, resource-based learning), etc. However, these tools do not cover all teaching aspects since, they do not usually provide tools which allow educators to thoroughly track and assess all the activities performed by all learners, nor to evaluate the structure of the course content and its effectiveness in the learning process ([11][13][21][24]). Consequently, the specific tools to help undertake this task must be developed. Here is where business intelligence techniques applied to the Web ([4][15][19][20][21]) play their role, given that these can generate statistics, analytic models and uncover meaningful patterns from data.

As mentioned in Section 2.2, the first step is to identify the business requirements that, in our case, can be summarized into answering questions such as:

- Regarding follow-up of the course: When do students connect to the system? Do they work online? Could the value of a session be measured in relation to learning objectives? This would help teachers to carry out continual assessment.
- Regarding the course: How often do they use collaborative tools? What are the sequences of visited pages in each session, in which order, and how long do students stay in each one? What are the most frequent paths? This will allow teachers to discover if students follow the sequence established by them, to detect not-visited pages or to know which the preferred resources of the students (pdf, videotutorials, etc) are among other things. In this way, teachers will have information to modify the structure of their courses and to adapt them to learners' behaviour.
- Regarding students: What are the students' profiles? Is there a relationship between their behaviour and their qualifications? Who leaves the course and when?

Now these questions must be transcribed into indicators which allow us to measure and assess the situation under different perspectives (date, time, course, learner ...) such as e-learning session duration, number of visited pages, average number of sessions, and so on. And from there, to design the dimensional schema, build the OLAP cubes and develop the teachers' tools.

In Figure III, the architecture of our solution is shown. As one can observe, the information of virtual courses, academic data as well as the e-learning platform log files which register the activity made in this tool (visited pages, date, time and name of student) are used as operational data sources. Then come the ETL processes which extract, transform and load this information onto the designed dimensional database (web house), on which the OLAP cubes are built to be used by the professor's tool. The whole description of the system is found in [10], regrettably only available in Spanish.

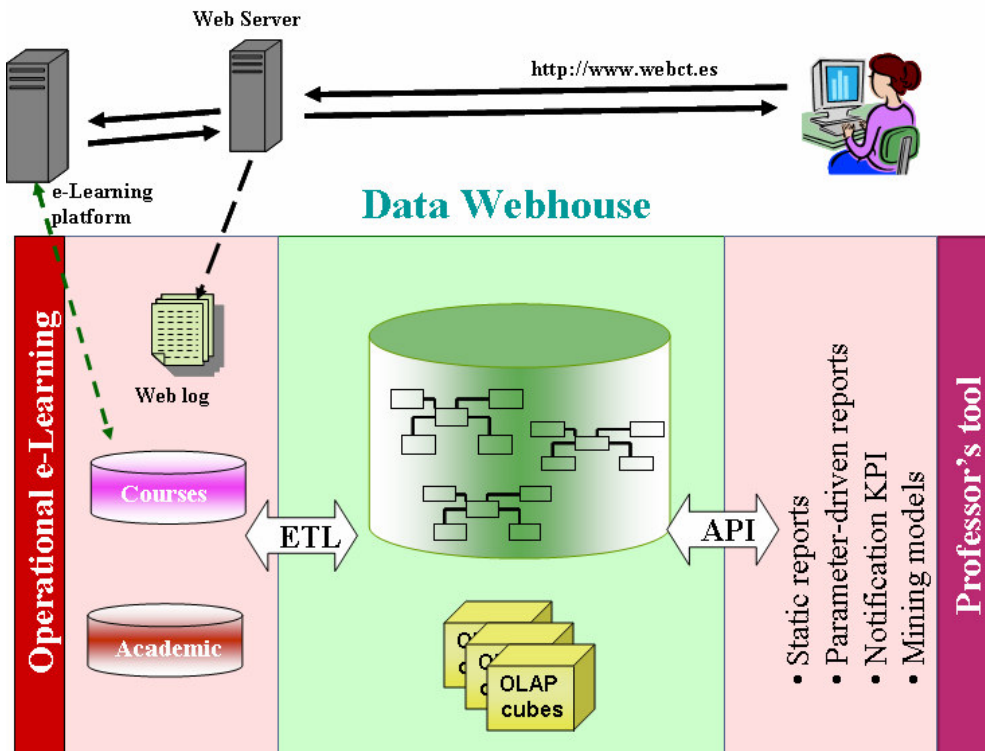


Figure III. E-learning Data Webhouse

In the next section, the basic concepts of dimensional modelling are explained based on this case study.

2.4. Dimensional modelling

Dimensional modelling [6] has been broadly accepted as the dominant technique for the implementation of data warehouse databases. Its success is mainly due to the fact that it allows users to understand databases easily and software to navigate databases efficiently (query performance).

A dimensional schema is made up of a central fact table and its associate dimensions. It is also called *star schema* because it looks like a star with the fact table in the middle and the dimensions serving as the points on the star. Figure IV shows a star diagram which gathers solely the clickstream data extracted from our e-learning platform logs. The level of detail chosen or *grain* for this fact table is a row for each completed learner session. A learner session is defined as the time spent by a student since he or she connects to a certain course of the system until he leaves it. Due to the fact that the http protocol is stateless, sometimes we do not know if the session has finished, for this reason, if a session is inactive for more than a certain period of time (this depends on the kind of course), this is closed.

From a relational data modelling perspective, we can say that the dimensional model consists of a normalized fact table with denormalized dimension tables.

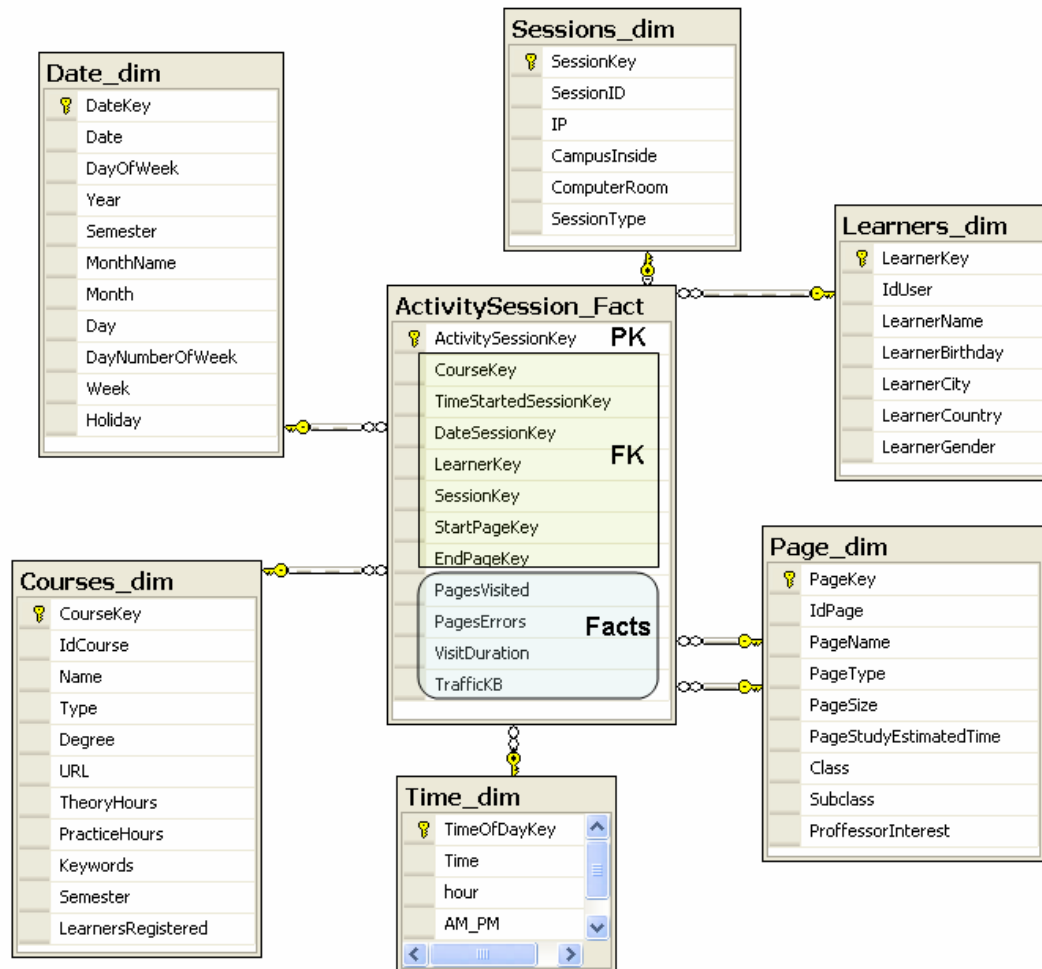


Figure IV. Star schema

2.4.1. Fact and dimension tables

The fact table contains the *measures* associated with a specific business process such as taking an order, enrolling at the university or, as in our case, evaluating e-learning sessions. These measurements or *facts* are generally numeric and additive like the number of pages visited, the traffic generated or the time spent on the e-learning session (see Figure IV), meaning they can be summed up across all dimensions. Although these can also be semi-additive, that means they are additive according to all dimensions except for the time one (such as stock in inventory or parts of a budget) or non-additive (such as unit prices or temperatures). In this last case, they can be aggregated calculating average values.

Besides, it is necessary to include as facts, the measures (numerator and denominator independently) with which one can calculate indicators that must be shown as ratios or percentages by remembering the ratio of the sums, not the sum of the ratios. On the other hand, it is sometimes interesting to also store measures derived from other facts (*derived or computed facts*) with a view to improving the query performance, such as PagesRequested calculated from PagesVisited plus PagesErrors.

Another important aspect of the design of a dimensional schema is the issue of granularity. This refers to the level of detail or summarization of the units of data in the star schema. The more detail there is, the lower the level of granularity. For example, a simple transaction (a learner session) would be at a low level of granularity. A summary of all

transactions for the month would be at a high level of granularity. It is recommended that fact tables be designed to store data at the lowest level of detail for the set of associated dimensions. If, for performance reasons, it is necessary to store more aggregated data, it must be done in another star diagram, which must be loaded from the data warehouse database; or, otherwise, an OLAP cube must be built which, besides performance, will offer other advantages as we can see in Section 3.

From a physical point of view, a fact table typically contains measures and the foreign keys to dimension tables, and, generally, its primary key is autonumeric (see Figure IV).

On the other hand, the dimension tables are the foundation of the dimensional model, containing descriptive information relevant to analyse the fact table attributes from different perspectives. For example, knowing the time invested monthly in each course or weekly by a learner, comparing the number of students connected on work days as compared with non-working days, etc. The best attributes to include in a dimension table are textual and discrete. A dimension table can be referenced several times, each one with a different role. For example, start page and end page of the session in our star schema.

In general, fact tables tend to include few attributes and an incredible number of rows, millions of rows, whereas dimension tables contain many attributes and a reduced number of rows compared with the fact tables.

Usually, a star does not have more than 15 dimension tables, with at least one of them being temporal (usually, date dimension). An excess of dimensions, more than 25, indicates that several are not independent. In this case, simpler dimensions must be combined.

Besides the *star schemas*, dimensional modelling includes the *snowflakes schemas*. These present the same structure as star schemas with the exception of the dimensional hierarchy which is explicitly represented by normalizing the dimension tables (see Time dimension and TimePeriod dimension in Figure V). This leads to advantages in maintaining the dimension tables and saving storage space; however, it reduces effectiveness of browsing the dimensions. Another additional advantage is that the normalized dimensions contribute to the building of *fact constellations* which are more complex structures in which multiple fact tables share dimensional tables. For example, ActivitySession_fact table and ActivityPage_fact table form a fact constellation since they share dimensions (See Figure V). Fact constellation requires that every dimension have the same meaning with each possible fact table to which it is joined (*conformed dimensions*).

2.4.2. Additional considerations

In the previous section, the basic dimensional modelling concepts have been described. They are applicable to a wide range of business scenarios, but there are other aspects that must be taken into account.

Sometimes, the business processes track events without any real measure, such as if a student used a collaborative tool or not, or did an exercise on a specific day or not. The way of gathering these facts is by means of a fact table with an attribute that contains the number 1 when the event happened or 0 otherwise. This provides users with an easy way to count the number of events by adding up this attribute. This kind of fact tables is called *factless fact tables* (See Figure VI).

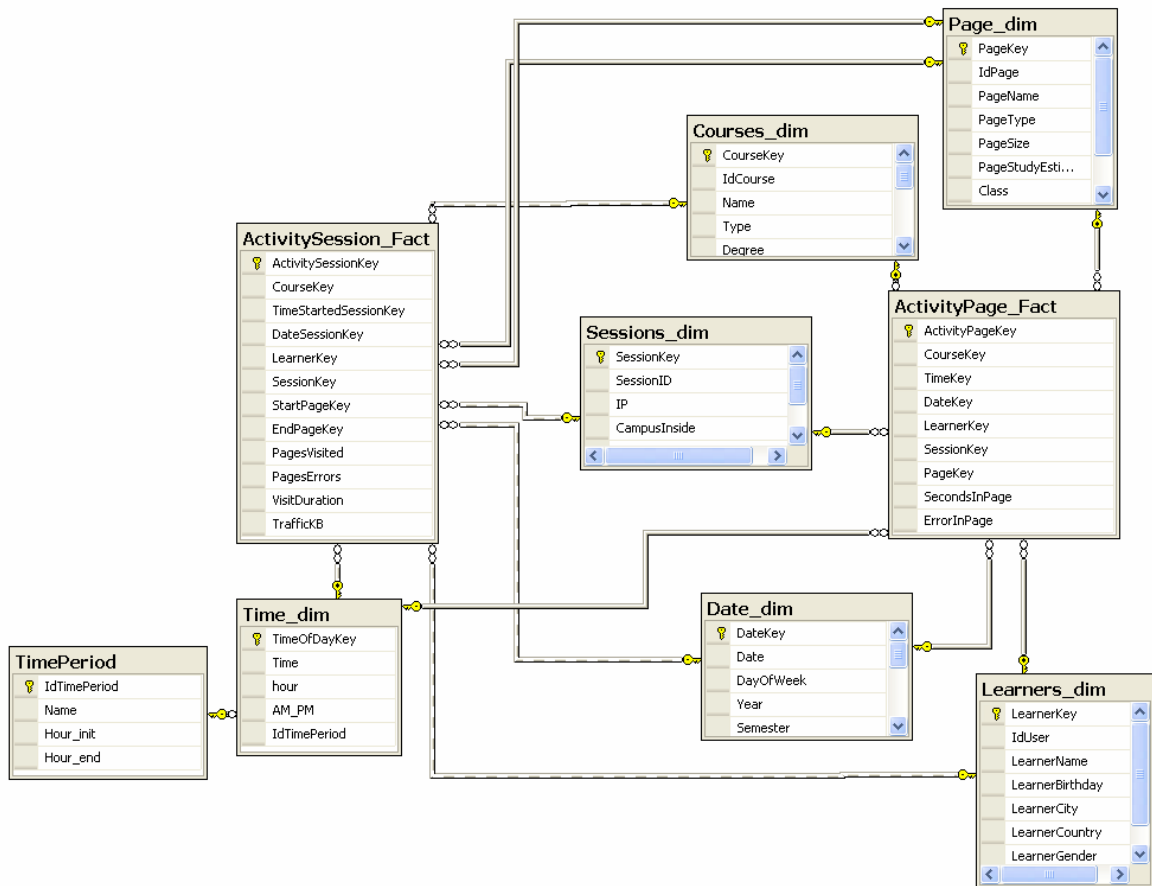


Figure V. Fact Constellation

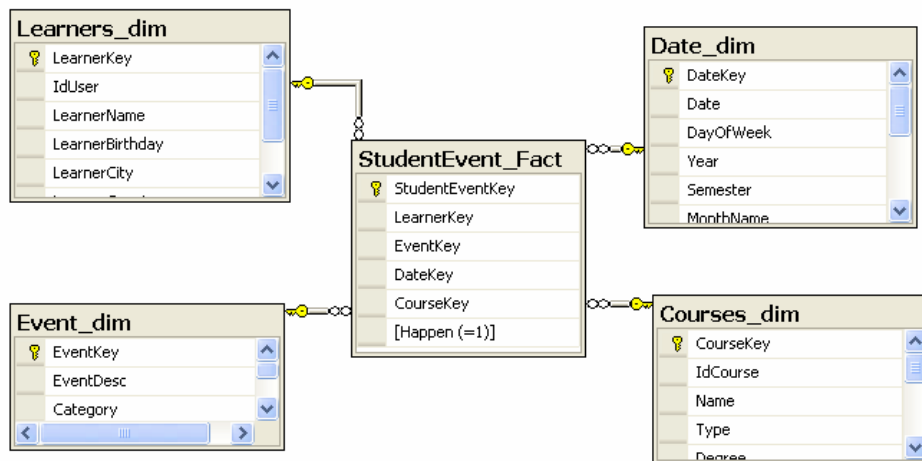


Figure VI. FactLess fact table

On the other hand, the transaction identifiers of the operational systems such as order numbers, number applicant, etc. that usually give rise to dimensions without attributes are represented in dimensional modelling as attributes in the fact table itself when the grain of the table is the document itself or a line item in the document (eg. order number). They are called *degenerate dimensions*. They are not facts, neither do they measure events nor are

they additive, but serve to connect back to the operational system and to know all the rows that belong to the same transaction.

On occasions, the fact table design suggests the inclusion in the fact table of textual attributes or flags (sex, civil status, age range, income...) that take on a small range of discrete values. In this situation, the designer can choose one of these possible solutions: leaving the attributes in the fact table, making separate dimensions for each attribute or defining a new dimension, called *junk dimension*, which combines all the possible values of this type of attributes. The latter could be the most appropriate solution but, the rest of the database schema must be taken into account (*conformed dimensions*).

Although we would like to think that the attribute values of the dimension tables are fixed, in fact they change over time. For example, in a generic professor dimension, the date of graduation should not change but the professional category might change several times. The decision on which dimensions and attributes must be tracked depends on the business. There are two alternatives to gather this information in the dimension tables. The first alternative is adding a new row to the dimension table each time that the values of the attributes change, registering the date on which the change occurs. Sometimes, it turns out to be more efficient, for performance query reasons, to also add a flag attribute which indicates the active row. This option allows one to follow the history but it poses some difficulties for data navigation. The second alternative consists in adding a column to the dimension table to keep the new and old attribute value. This solution is used rather infrequently because it involves changing the physical tables and is not very scalable.

A final aspect to take into consideration is choosing the primary key of each table in the dimensional schema. It is recommended that a whole new set of keys be created in the data warehouse database, separated from the keys in the operational systems. This makes the integration of different operational systems easier and, besides, protects the data warehouse database from changes in the source systems. These keys are called *surrogate keys* and are usually numeric. They can be observed in Figure VI marked with a yellow key. The mapping between operational and surrogate keys is registered in the staging area. Finally, the dimension tables usually include an attribute which collects the primary key of the transactional system where the data are from (eg. attribute CourseID in Course_dim table). This attribute can be used for the synchronization tasks between the operational and analytical databases and also for the follow-up of the changes in the dimension attributes.

2.5. Data warehouse vs data mart

In this section, we are going to differentiate between two terms, data warehouse and data mart, which are frequently used as synonymous. As mentioned in Section 1, we consider that the data warehouse is the whole organization's analytical system which includes the staging area, the data area and the presentation area (data access tools). Within data area, we find the data warehouse database which collects all information about subjects (learners, courses, sessions, registration...) that span the entire organization, and, on the other hand, we can also find data marts.

A data mart is defined as a data warehouse information subset, subject-oriented such as academic registration or student activity data mart which is usually stored independently of the data warehouse database for performance reasons. Data marts generally contain the information of a star diagram whose grain can be thicker than the grain in the data warehouse database or with the same level of detail. In its most simplistic form, a data mart represents data from a single business process.

Due to the fact that most user groups do not need to accede to the whole of the information, decomposing the data warehouse database into different data marts usually improves the performance of the queries by reducing the data volume. For this reason, the use of data marts is justified for: segmenting the information into different hardware platforms (portability); facilitating access to the query tools; dividing data to better control the accesses and improving reply-times. Only one thing must be taken into account, in order to assure data consistency, data marts must be loaded from data warehouse database and not from data sources.

3. OLAP TECHNOLOGY

OLAP technology [2][9][14][17] appeared in the 90's in order to meet the needs that managers and business analysts had to search accurate, update and complete information in their corporative systems (generally with large volumes of data) quickly, and handle and explore this information under different perspectives by means of simple and user-friendly formats.

Although relational databases are a great place to store and manage data, as has already been said, their internal data structures and the SQL query language have not been designed to answer complex business queries quickly. In our case study, questions such as the following have arisen:

- How long is our learner connected by course and by month? Which position in the group does each student occupy according to the average?
- What is the connected learner distribution over time (time of the day, day of the week, month and year)?
- How many learning sessions do our learners establish over time? Is it true that the more students know about computer science, the longer they are connected when studying?
- How many pages are visited on average in each session? Which are the top 10 pages visited by course over different periods of time? Which ones have never been visited?

To solve the performance problem, OLAP technology builds multidimensional structures called *cubes* that contain pre-calculated summary data (*aggregations*). In such a way that querying existing aggregated data is close to instantaneous compared to doing cold queries (no cache) with no pre-calculated summaries in place. For example, a typical aggregation would be adding up learners' sessions figures from course level, degree level or university level.

And, in order to help analysts do their job, the OLAP technology uses a specific query language, called MDX (Multi-Dimensional eXpressions), which includes operators and functions that allow, in a simple way, one to build the typical business experts' queries; and, besides, it incorporates navigation and data manipulation capacities so that the end-user may analyse data easily and quickly.

3.1. Concepts and terminology

The *cube* is the central element on which multidimensional databases are founded. This can be defined as a subset of data from a database designed under a relational or dimensional schema that is stored in a multidimensional structure. It must be pointed out, that if the cubes are built from a dimensional schema, its design is easier, since it consists

of a simple mapping: dimensions of the star diagram into cube dimensions and fact tables of the star diagram into cube fact tables.

Although the term cube suggests three dimensions, a cube can have more. In Figure VII, we show an OLAP cube example. This has been built from the ActivitySession fact table using three dimension tables: Date_dim, Courses_dim and Learners_dim. You can observe that the front face of the cube has been divided into 25 small squares which are called *cells*. The cell is an entity from which you can retrieve data that is pertinent to an intersection of the *dimension members*. The number of cells depends on the number of *hierarchies* within each dimension of the cube and the number of members in each hierarchy.

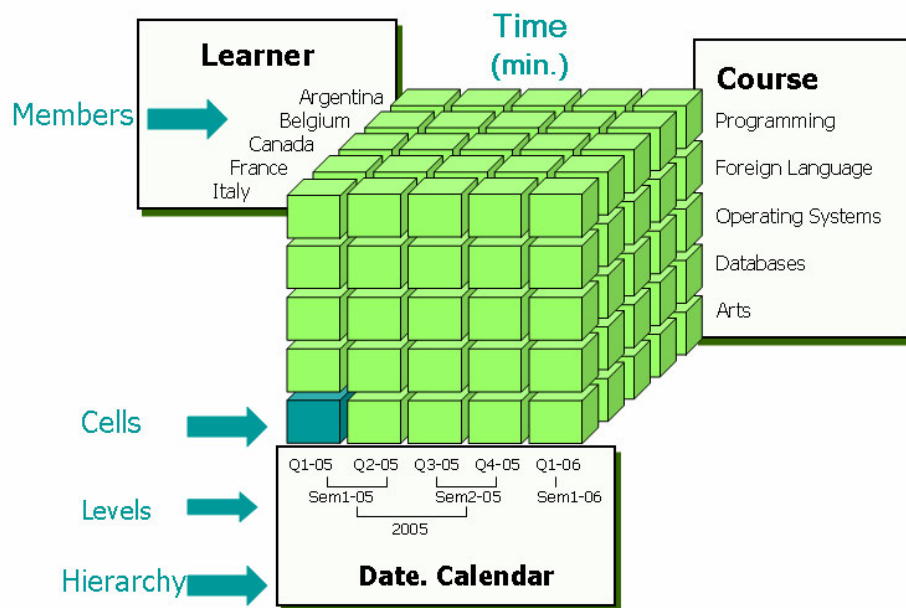


Figure VII. Cube components

As you can image, cells hold the data values of all measures of the cube. The *measures* are always quantitative entities. These can be the same as in fact table or others calculated based on them. These are always defined by applying an aggregation function to them such as sum, average, count, max, min ...

The term *member* refers to each discrete value in a dimension. And this corresponds to one or more occurrences of that value in the underlying dimension table. For example, Course dimension might contain the following members: [ALL], Programming, Foreign Language, and so on. The [ALL] member represents the total for all of the members in a dimension.

Each *dimension* may have one or more *hierarchies* and each hierarchy may contain one or more *levels*. The hierarchy is a way of grouping members and is defined in terms of their levels. In Figure VII, you can see Date.Calendar hierarchy which has as members: 2005, Sem1-05, Sem2-05, Q1-05, etc. and these are grouped into the following levels: quarters into semesters and years. So, we can say that levels describe the member order from the most summarized level of data to the most detailed level.

In summary, an OLAP cube is a container of detail values from one or more fact tables, along with all possible aggregations for one or more dimension hierarchies.

But, the really interesting point of the OLAP cubes is not only their capacity to make queries that, in a sense, can be done by selections, projections, concatenations and traditional groupings on the relational database; but also their operators of refinement and manipulation of queries.

Typical OLAP operations include *rollup*, which means, increasing the level of aggregation along one or more dimension hierarchies; *drill-down*, decreasing the level of aggregation or, put differently, increasing detail; *slice and dice*, taking a projection of the data on a subset of dimensions for selected values of the other dimensions, which means defining subcubes; and *pivot*, re-orienting the multidimensional view of data. Figure VIII shows graphically how these operations work.

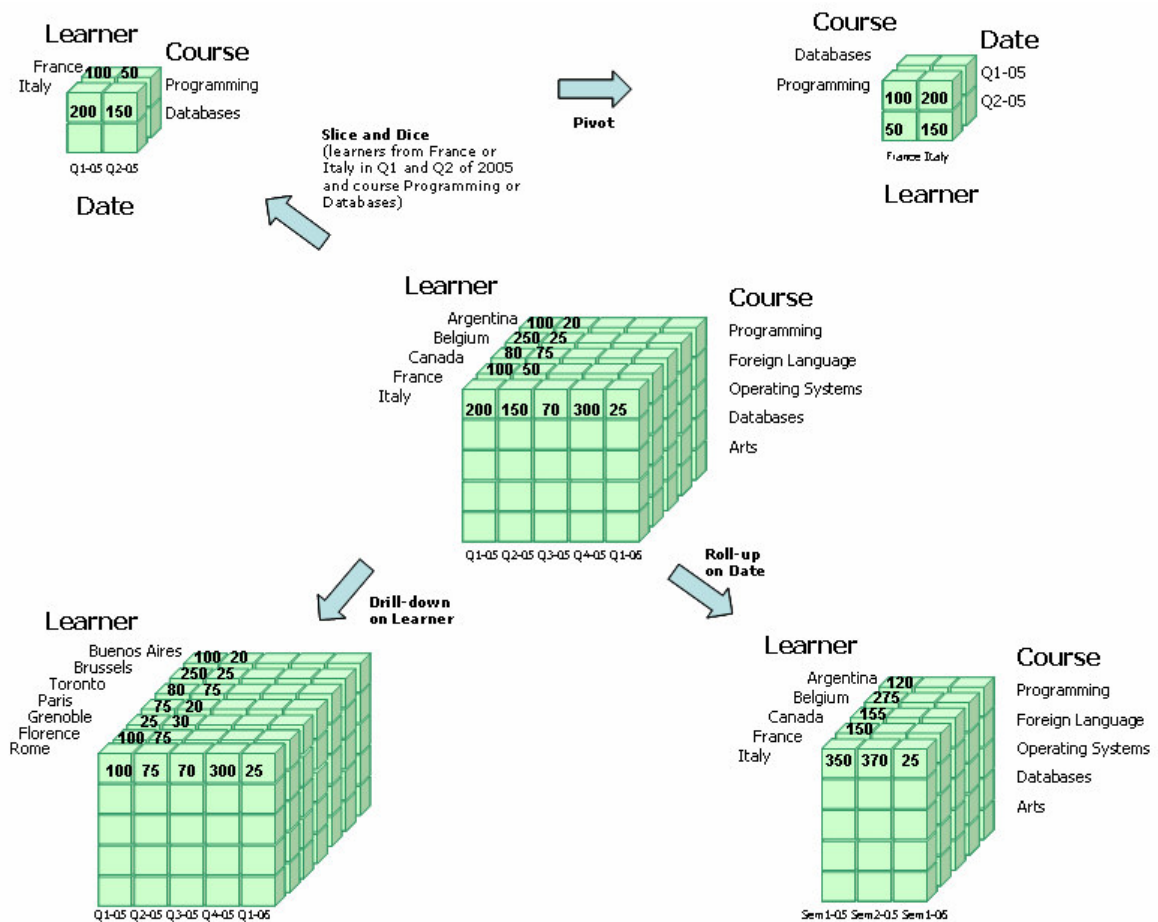


Figure VIII. OLAP cube operations

3.2. OLAP cube design

Cube definition will depend on the indicators and metrics that the end-user needs to observe and on the dimensions or perspectives under which he needs to assess them. This means that different cubes may be built on one or more fact tables of the data warehouse database, each one of them oriented to answer the questions of the different users of the organization.

The steps to define OLAP cubes are summarized as follows:

First, choose the fact tables which contain the measures to be included in the cube and, next, select the aggregation function to be applied to them (usually sum for additive measures and count or average for non-additive ones).

Second, decide on the dimensions that the cube will include and define the hierarchies. The cube can have more than a hierarchy per dimension (Year→Semester→Quarter and Year→Month→Day in the Date dimension). And, besides, a dimension can be shared by several cubes.

Next, add, if necessary, other measures to the cube. These are generally calculated as a combination of measures that are already in the cube (*calculated measures*), for example, the addition, subtraction or multiplication of two or more base measures; or else, they come out as a consequence of applying another aggregation function on an existing measure. *Calculated measures* are stored in the cube.

Finally, other measures whose calculus depends on the members of the dimension that the user is observing at each moment must be included. They are called *calculated members*. For example, the ratio of certain measures according to specific business conditions, the contribution of a measure as a percentage of the total, the cumulative total of a measure over time, the ranking and sorting of measures under different criteria such as knowing the top 5 pages that have contributed more in terms of visit duration in a specific course. Additionally, to define calculated members it may be necessary to create previously a set of dimension members which fulfil certain conditions, such as students with a high mark in order to be compared with the rest of the course.

Only the definition of calculated member is cached in the cube, their values are obtained in browsing time.

Calculated members are defined in MDX language. MDX is a pseudoacronym for MultiDimensional eXpressions. Just as SQL (Structure Query Language) is a query language used to retrieve data from relational data sources, MDX is a query language created for retrieving data from OLAP data sources. MDX is a standard language which is part of the OLEDB for OLAP specification.

Once the dimensions and the cube are defined, they must be processed. This means, creating the multidimensional structure with all the aggregations defined. There are three alternatives: MOLAP, ROLAP and HOLAP which are explained in the following section.

3.3. OLAP storage: MOLAP, ROLAP, HOLAP

Because a cube appears to contain every possible summarized value at every possible level of detail for every dimension, the cube must appear to contain not only the possible detail combinations but also all possible summary values. The maximum size of the cube will be $(n_1+1)*(n_2+1)* \dots *(n_d+1)$ where n_i is the number of rows of i dimension and d is the number of dimensions.

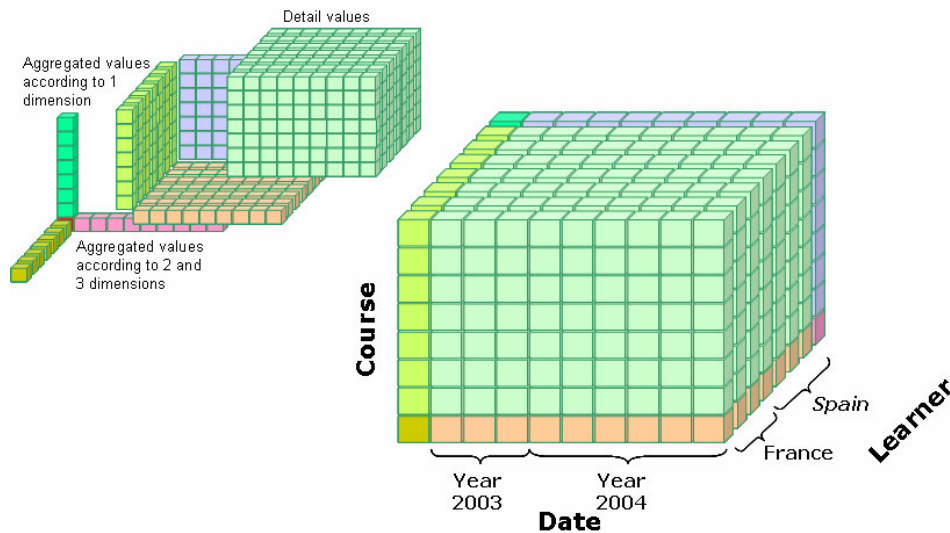


Figure IX. MOLAP cube (detail and aggregated values)

A cube consists of three logical components: a map, detail values and aggregated values (see Figure I). The map stores all the information about the logical and physical definition of the cube. The detail values correspond to the lowest-level members of each dimension. The aggregated values are summarized values for higher levels in dimension hierarchies.

An OLAP cube stores the cube map within OLAP server and offers 3 options to store detail and aggregated values: ROLAP, HOLAP or MOLAP.

- ROLAP (Relational OLAP) leaves detail values in the relational fact table and stores the aggregated values in the relational database as well.
- HOLAP (Hybrid OLAP) leaves detail values in the relational fact table and stores the aggregated values in the cube.
- MOLAP (Multidimensional OLAP) stores both, detail and aggregated values within the cube.

MOLAP provides better query performance as compared to the other storage types. Three potential disadvantages are: storage needed for large databases, the inability to see new data entering your data warehouse and it does not scale to large numbers of dimensions or data sets (limited to 10 dimensions or 5 GBs).

ROLAP has the ability to handle large cubes only limited by the relational backend. The most important disadvantage is its slow query performance, given that the OLAP engine translates MDX query to one or more SQL queries. Besides, unlike the MOLAP cube, the ROLAP cube is non-portable, so the user must work in a connected environment.

HOLAP combines the best of MOLAP and ROLAP modes. If the queries are on aggregated data, the answer will be faster than if it retrieves detail data and it requires less storage space. It is a solution which balances performance cost and storage cost.

Since a cube is a logical entity, you can use different storage modes for different portions of a cube. The final configuration will have to be decided on according to the end-users' necessities.

4. BUSINESS INTELLIGENCE APPLICATIONS

Once the whole data structure (data stage and data area) is built, one must continue with the development of the user's interface. That is to say, the tool with which business experts can do, among others, the following tasks:

- Trend analysis and detection
- Key indicator measurement, tracking and monitoring
- Drill-down analysis
- Problem monitoring
- Competitive analysis

As indicated in section 2.1, there are different alternatives: one can purchase a packaged application and customize it so that data are taken from data area; or, use a commercial reporting and querying tool with which this interface can be built quickly and easily; or, otherwise, programme ad-hoc tools which allow end-users to interact with the strategic information.

As always, all alternatives have advantages and disadvantages. The packaged applications present elaborate interfaces that meet the needs of business experts, but their customization is generally costly in terms of time and money. The development of ad-hoc tools allows the creation of flexible interfaces adapted to the users' needs but its average cost in time/person is higher and its implementation and deployment requires longer. And the use of the commercial querying and reporting tools, which is the most commonly used option, provides programmers with many components that contribute to the rapid development of the applications although these tools present certain limitations to personalise the environment or add non-supported functions.

The spreadsheet programmes are another less sophisticated alternative, but also very used to manage exclusively OLAP cubes. They allow business analysts to explore and analyse data with a tool they are familiar with, although they lose the whole system perspective. They are suitable to work with the strategic data disconnected from the analytical system.

BI applications generally include a broad spectrum of reports and analyses, ranging from very simple, fixed-format reports to sophisticated analytical applications with complex embedded algorithms and domain expertise.

Standard reports are the basic end of the BI applications. They are relatively simple, predefined-format and parameter-driven reports. They provide users with a core set of information about what is going on in a particular area of the business. Generally 80-90% of the end-users work with standard reports.

An example based on our case study can be observed in Figure X. It shows two reports which summarize the global usage analysis of the course (image on the left) and the usage per learner (image on the right). Both include the number of sessions, average time per session and number of pages per session measures and additionally, the second one also presents the average values of the course so that instructors can compare the measures. These reports are very useful because they allow teachers to evaluate the usage of their course, to detect if a student is close to giving up, if the students connect to the system frequently, or if the study effort is higher than they planned.

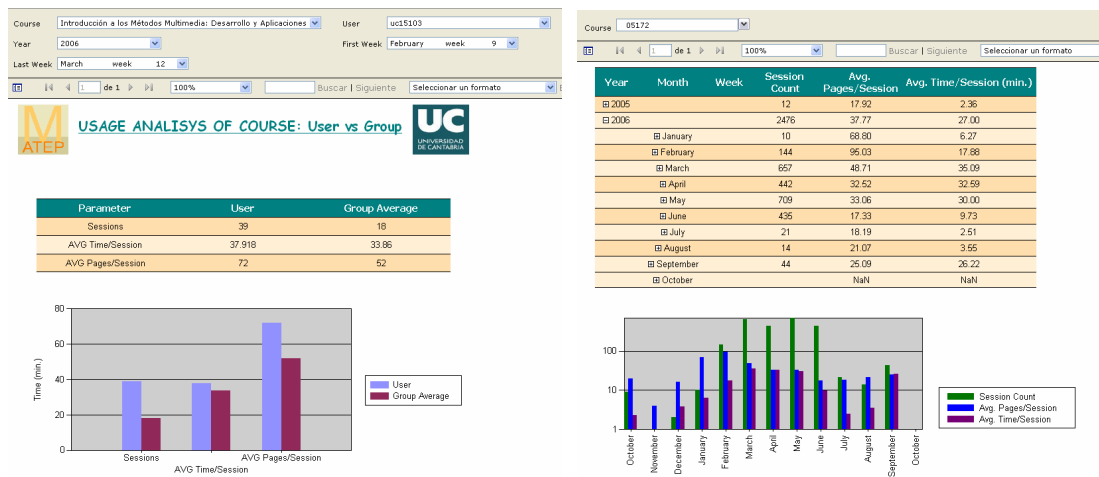


Figure X. Global usage course and usage per learner

On the other hand, BI applications can also include tools with which end-users create their own queries and reports. This utility that allows business users to dive in and explore the data in an independent and autonomous way, regrettably, is only used by a 10% of the end-users. Probably because learning the use of the tool and the meaning of the data is beyond the interest of these people.

Finally, BI applications can also integrate analytic applications. They are focused on a specific business process and encapsulate a certain amount of domain expertise about how to analyze and interpret that process. They may include complex, code-based algorithms or data mining models that help identify underlying issues or opportunities. Generally, these models are obtained by experts in data mining techniques ([1][3][16][18]) and included in the BI application as rules.

As an example, Figure XI contains the graphic result of a clustering of e-learning sessions carried out on a specific virtual course of our platform, with the aim of, later, finding relationships between students and web-navigation behavior. Four clusters are identified: the first gathers very short sessions in a regular day probably to read the news; the second is the same as the first but staying longer (reading /writing mail, studying contents, doing activities, etc.); the third gathers longer sessions (48 min. on average) on days previous to the submission of a task; and, finally, the fourth collects very long sessions from students who work at the last moment (close to a submission deadline). This clustering allows teachers to conclude that the distance learners' behavior is similar to their behavior in traditional teaching.

To finish, the communication between the end-user tools and the data warehouse data area is generally established by using some of the following Application Programming Interfaces (API): ODBC or JDBC to connect to relational databases; OLEDB to access to different data sources (flat files, databases, ...); OLEDB for OLAP and OLEDB for DM to connect to multidimensional or data mining servers respectively, but limited to MS Windows platform; or XML for Analysis API that allows client applications to query DM and OLAP servers from any platform to any other platform. It must be mentioned that other APIs are still under development such as Java Data Mining API (JDM) and Java OLAP (JOLAP).



Figure XI. Web-navigation behaviour cluster

5. SUMMARY

Data warehousing and OLAP technologies are the key elements for the development of business intelligence systems, which enable analysts, managers and executives to analyze, track and assess the key performance indicators of their business so that they can make faster and better strategic decisions.

Nowadays, nobody questions the fact that the building of DW/BI solutions is essential for organizations in order to be more and more competitive. But their design and building are not trivial; this is why this chapter has been written.

This chapter makes it clear, from a practical point of view, what a Business Intelligence solution is, why it is needed, which its main components are, and how to confront its design and development. Besides, it explains in detail the main concepts of the dimensional model on which the data warehouse database is generally based and the basic terminology and notions of OLAP technology. Both of them are essential to build analytical applications which meet users' needs and offer them useful, accurate, consistent and understandable information through simple and user-friendly interfaces. The chapter also presents different alternatives to build these interfaces.

Lastly, it is important not to forget the remarkable role that data mining techniques play within the Business Intelligence field, although these have not been dealt with in this chapter.

6. REFERENCES

- [1] Han, J. Data mining: Concepts and Techniques. Morgan Kaufmann. 2006
- [2] Harinath, S., Quinn, S. Professional SQL Server Analysis Services 2005 with MDX. Wiley Publishing Inc. 2006.
- [3] Hernández Orallo, J., Ramírez Quintana, M^a J., Ferri Ramírez, C. Introducción a la minería de datos. Pearson Prentice Hall. 2004.

- [4] Hu, X., Cercone, N. A data warehouse/online analytic processing framework for web usage mining and business intelligence reporting. *International Journal of Intelligent Systems*, Volume 19, Issue 7, Pages 585 – 606. 2004
- [5] Inmon, W. H. *Building the Data Warehouse*. Wiley & Son. 2002.
- [6] Kimball, R., Ross, M. *The data warehouse toolkit: the complete guide to dimensional modelling*. John Wiley & Sons, cop. 2002.
- [7] Kimball, R., Caserta, J. *The Data Warehouse ETL Toolkit*. John Wiley & Sons, 2002.
- [8] Kimball, R. et al. *The Data Warehouse Lifecycle Toolkit: Tools and Techniques for Designing, Developing, and Deploying Data Warehouses*. John Wiley & Sons, 1998.
- [9] Jacobson, R. *Microsoft SQL Server 2000 Analysis Services step by step. OLAP Train*, Microsoft Press, cop. 2000.
- [10] Martín Fraile, L. *Monitoring and analysis tool for e-Learning platforms*. Final Degree Project directed by Zorrilla Pantaleón, M. University of Cantabria. 2007.
- [11] Mazza, R., Dimitrova, V.: *CourseVis: A graphical student monitoring tool for supporting instructors in web-based distance courses*. *International Journal of Human-Computer Studies* 65(2), 125–139, 2007.
- [12] Mundy, J., Thorthwaite, W., Kimball, R. *The Microsoft Data Warehouse Toolkit: with SQL Server 2005 and the Microsoft Business Intelligence Toolset*. Wiley Publishing Inc. 2006.
- [13] Romero, C., Ventura, S.: *Data mining in E-Learning*. Volume 4 of *Advances in Management Information*. WIT Press. 2006
- [14] Spofford, G. et al. *MDX Solutions: With Microsoft SQL Server Analysis Services 2005 and Hyperion Essbase*. Wiley Publishing, 2006.
- [15] Srivastava, J., Cooley, R., Deshpande, M., Tan, P. *Web usage mining: discovery and applications of usage patterns from Web data*. *SIGKDD Explor.*, vol. 1, nº 2. ACM Press, 2000.
- [16] Tan, P., Steinbach, M., Kumar, V. *Introduction to data mining*. Pearson Prentice Hall. 2006.
- [17] Thomsen, E. *OLAP Solutions: Building Multidimensional Information Systems*, Second Edition, John Wiley & Sons, 2002.
- [18] Witten, I., Frank, E. *Data mining. Practical machine learning tools and techniques*. Morgan Kaufmann. 2005.
- [19] Xue, H., Liu, ZW., Shun, XM. *Research and realization of medicine materials sales BI system*. In *Proc. Of the International Symposium on Distributed Computing and Applications to Business, Engineering and Science 2006*, Vol.2, pp. 950-953, 2006.
- [20] Zaïane, O., Xin, M., Han, J. *Discovering Web Access Patterns and Trends by Applying OLAP and Data Mining Technology on Web Logs*, in *Proc. ADL'98 (Advances in Digital Libraries)*, Santa Barbara, April 1998.
- [21] Zaïane, O. *Web Usage Mining for a Better Web-Based Learning Environment*. *Proc. of Conference on Advantage Technology for Education*. Alberta, Canada. 2001.

- [22] Zorrilla, M. E., Menasalvas, E., Marín, D., Mora, E., Segovia, J. Web usage mining project for improving web-based learning sites. LNCS 3643, pp. 205 – 210, Springer-Verlag, 2005.
- [23] Zorrilla, M., Millán, S., Menasalvas, E. Data webhouse to support web intelligence in e-learning environments. In Proc. of the IEEE International Conference on Granular Computing. Beijing, China, 2005.
- [24] Zorrilla, M. E., Marín, D., Álvarez, E. Towards virtual course evaluation using Web Intelligence. LNCS 4739, pp. 392–399. Springer-Verlag Berlin Heidelberg, 2007.