Business Intelligence | Meerza Ahmed
Tuesday, October 24, 2023 1:00 PM

Lecture 10: Link | Syllabus
Class Week 10/15

Database Systems by Coronel & Morris

Chapter 13: Business Intelligence & Data Warehouses

* 13.1 *The Need for Data Analysis*
Organizations are always looking for a competitive advantage through product development, market positioning, sales promotions, and customer service. Thanks to the Internet, customers are more informed than ever about the products they want and the prices they are willing to pay. So than, how can companies survive on lower margins and still make a profit? The key is in having the right data at the right time to support the decision-making process.

* 13.2 *Business Intelligence*
Business intelligence (BI) is a term that describes a comprehensive, cohesive, and integrated set of tools and processes used to capture, collect, integrate, store, and analyze data with the purpose of generating and presenting information to support business decision making. BI is a framework that allows a business to transform data into information, information into knowledge, and knowledge into wisdom

* 13.2-a *Business Intelligence Architecture*
BI architecture is composed of many interconnected parts: people, processes, data, and technology working together to facilitate and enhance a business's management and governance.
BI framework has six basic components:
- ETL tools - Data extraction, transformation, and loading (ETL)
- Data Store - Generally represented by Data warehouse or a data mart, usually optimized for data analysis and query speed
- Query & Reporting - used by the data analysis to create queries that access the database and create reports.
- Data Visualization - used to present data in meaningful way visually using charts and graphs
- Data Monitoring and alerting - this allows for the real-time monitoring of business activities.
- Data Analytics - Performs data mining and data analysis using the data in the data stores.

* Governance is a method or process of government. In this case, BI provides a method for controlling and monitoring business health and for consistent decision making. Furthermore, having such governance creates accountability for business decisions

* 13.2-b *Business Intelligence Architecture*
Improved decision making is the main goal of BI, but BI provides other benefits :
- Integrating architecture - This is the integrated umbrella for a disparate mix of IT systems within an organization
- Common User Interface - A common user interface familiar to all company members providing up to date information
- Common Data repository - This provides a framework to integrate under a common environment with a single version
- Improved organizational Performances - An increased bottom line due to less waste, increased sales, turnover and much more

* 13.3 *Business Support Data*
Although BI is used at strategic and tactical managerial levels within organizations, its effectiveness depends on the quality of data gathered at the operational level. Yet, operational data is seldom well suited to the decision support tasks.
Operational data is well-suited for decision support tasks its is also stored in a relational database with a normalized structure, and optimized to support transactions linked to daily operations.
Decision Support Data differs from operational data in its time span and granularity. Drill down is the decomposing of data to a lower level and Roll up is the aggregating a data into a higher level.

CONTRASTING OPERATIONAL AND DECISION SUPPORT DATA CHARACTERISTICS

CHARACTERISTIC	OPERATIONAL DATA	DECISION SUPPORT DATA
Data currency	Current operations Real-time data	Historic data Snapshot of company data Time component (week/month/year)
Granularity	Atomic-detailed data	Summarized data
Summarization level	Low; some aggregate yields	High; many aggregation levels
Data model	Highly normalized Mostly relational DBMSs	Non-normalized Complex structures Some relational, but mostly multidimensional DBMSs
Transaction type	Mostly updates	Mostly query
Transaction volumes	High-update volumes	Periodic loads and summary calculations
Transaction speed	Updates are critical	Retrievals are critical
Query activity	Low to medium	High
Query scope	Narrow range	Broad range
Query complexity	Simple to medium	Very complex
Data volumes	Hundreds of gigabytes	Terabytes to petabytes

* 13.3-b *Decision Support Database Requirements*

A decision support database is a specialized DBMS tailored to provide fast answers to complex queries. There are three main requirements for a decision support database :

- Database Schema - Must support complex, non-normalized data representations. Data must also be aggerated and summarized and Queries must be able to extract multidimensional time slices.
- Data Extraction and Filtering - Allows batch and schedule data extraction, Supports different data sources and checks for inconsistent data or data validation rules. Helps advanced integration, aggregation, and classification.
- Database Size - the DBMS must be capable of supporting very large databases (VLDBs). To support a VLDB adequately, the DBMS might be required to support advanced storage technologies, and even more importantly, to support multiple -processor technologies, such as a symmetric multiprocessor (SMP) or a massively parallel processor (MPP)

* 13.4 *The Data Warehouse*

Bill Inmon, the acknowledged "father" of the data warehouse, defines the term as "an integrated, subject-oriented, time-variant, nonvolatile collection of data that provides support for decision making."

CHARACTERISTICS OF DATA WAREHOUSE DATA AND OPERATIONAL DATABASE DATA

CHARACTERISTIC	OPERATIONAL DATABASE DATA	DATA WAREHOUSE DATA
Integrated	Similar data can have different representations or meanings. For example, Social Security numbers may be stored as ###-##-#### or as #########, and a given condition may be labeled as T/F or 0/1 or Y/N. A sales value may be shown in thousands or in millions.	Provide a unified view of all data elements with a common definition and representation for all business units.
Subject-oriented	Data is stored with a functional, or process, orientation. For example, data may be stored for invoices, payments, and credit amounts.	Data is stored with a subject orientation that facilitates multiple views of the data and decision making. For example, sales may be recorded by product, division, manager, or region.
Time-variant	Data is recorded as current transactions. For example, the sales data may be the sale of a product on a given date, such as $342.78 on 12-MAY-2016.	Data is recorded with a historical perspective in mind. Therefore, a time dimension is added to facilitate data analysis and various time comparisons.
Nonvolatile	Data updates are frequent and common. For example, an inventory amount changes with each sale. Therefore, the data environment is fluid.	Data cannot be changed. Data is added only periodically from historical systems. Once the data is properly stored, no changes are allowed. Therefore, the data environment is relatively static.

* This table summarizes the differences between data warehouses and operational databases. In summary, the data warehouse is a read-only database optimized for data analysis and query processing. Typically, data is extracted from various sources and are then transformed and integrated—in other words, passed through a data filter— before being loaded into the data warehouse. As mentioned, this process is known as ETL.

* 13.4-a *Data Marts*

A data mart is a small, single-subject data warehouse subset that provides decision support to a small group of people. In addition, a data mart could be created from data extracted from a larger data warehouse for the specific purpose of supporting faster data access to a target group or function. That is, data marts and data warehouses can coexist within a business intelligence environment

* 13.4-b *The Twelve Rules That Define a Data Warehouse*

TWELVE RULES FOR A DATA WAREHOUSE

RULE NO.	DESCRIPTION
1	The data warehouse and operational environments are separated.
2	The data warehouse data is integrated.
3	The data warehouse contains historical data over a long time.
4	The data warehouse data is snapshot data captured at a given point in time.
5	The data warehouse data is subject oriented.
6	The data warehouse data is mainly read-only with periodic batch updates from operational data. No online updates are allowed.
7	The data warehouse development life cycle differs from classical systems development. Data warehouse development is data-driven; the classical approach is process-driven.
8	The data warehouse contains data with several levels of detail: current detail data, old detail data, lightly summarized data, and highly summarized data.
9	The data warehouse environment is characterized by read-only transactions to very large data sets. The operational environment is characterized by numerous update transactions to a few data entities at a time.
10	The data warehouse environment has a system that traces data sources, transformations, and storage.
11	The data warehouse's metadata is a critical component of this environment. The metadata identifies and defines all data elements. The metadata provides the source, transformation, integration, storage, usage, relationships, and history of each data element.
12	The data warehouse contains a chargeback mechanism for resource usage that enforces optimal use of the data by end users.

* 13.5 *Star Schemas*

The star schema is a data-modeling technique used to map multidimensional decision support data into a relational database. In effect, the star schema creates the near equivalent of a multidimensional database schema from the existing relational database. Star schemas yield an easily implemented model for multidimensional data analysis while preserving the relational structures on which the operational database is built.

* The basic star schema has four components: facts, dimensions, attributes, and attribute hierarchies

1. Facts - Numeric measurements (values) that represent a specific business aspect or activity. Fact tables contains facts that are linked through their dimensions, which are explained in the next section. Facts can also be computed or derived at run time. Such computed or derived facts are sometimes called metrics to differentiate them from stored facts.

2. Dimensions - Qualifying characteristics that provide additional perspectives to a given fact. Dimensions are the magnifying glass through which you study the facts. Such dimensions are normally stored in dimension tables.

3. Attributes - Used to search, filter, or classify facts. Dimensions provide descriptive characteristics about the facts through their attributes. Therefore, the data warehouse designer must define common business attributes that will be used by the data analyst to narrow a search, group information, or describe dimensions.

4. Attribute Hierarchies - Attributes within dimensions can be ordered in a well-defined attribute hierarchy. The attribute hierarchy provides a top-down data organization that is used for two main purposes: aggregation and drill-down/roll-up data analysis.

FIGURE 13.7 THREE-DIMENSIONAL VIEW OF SALES

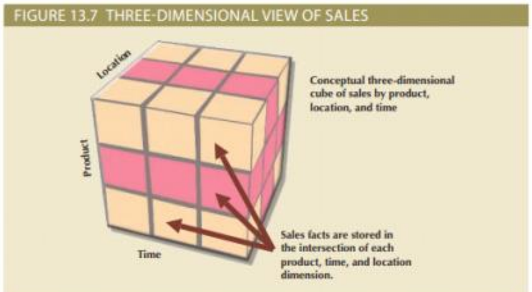
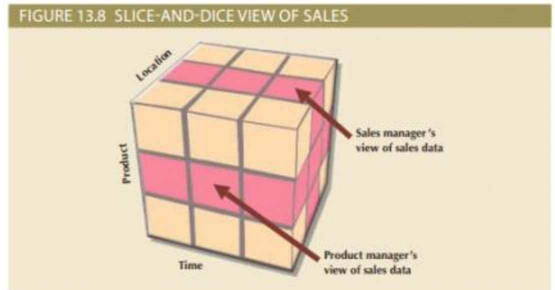


FIGURE 13.8 SLICE-AND-DICE VIEW OF SALES



* The Professor's Summery of the Star Schema

Fact tables that we see in the middle of star/snowflake schema, are ALWAYS denormalized, with multiple repeating values in the columns that link to dimensions - eg. multiple date values, product values, location values, POS terminal # values etc (because each row in a fact table contains those columns as raw 'facts').

Dimension tables, in a star schema are ALSO denormalized - eg. location dimension, with city, state, region columns, will have repeating values for states (because many cities are in each state), and repeating region values (because many states are in each region).

Dimension tables in a snowflake schema are normalized, because we create a chain (hierarchy) of them using the star's dimension columns.

The fact table ALWAYS stays denormalized. Such a fact table is said to employ star schema, if we use star-like denormalized columns for BI - eg. to find out how much of a product we sold in a city, we'd query the fact rows for city name, and if we need it, can also do state-level analyses (because states are listed in the location dimension table).

Using a snowflake schema, doing location analysis for a product at a city level is similar to the above paragraph - we simply look for the city name, and if necessary, get extra info about the city (eg tax rate) by looking at the dimension table. BUT to do state level analysis, we need to follow the city->state link, and use the state-level dimension table ie traverse a branch of the snowflake.

To avoid traversing those branches in a snowflake, we trade off ('waste') space by creating extra 'copies' of the fact table, where a column such as city (lowest value in the hierarchy of 'location') is REPLACED instead with 'state' values, and in another copy, with 'region' values. This lets us do star-like analyses again, because a fact row directly points the state table, and in another copy, directly points to the region table - no traversing the chain necessary (at the expense of extra storage).

Which schema (star or snowflake) is used to model the warehouse, determines whether we maintain denormalized (or normalized) dimension tables [fact tables always stay denormalized]. 'For BI purposes, the idea is to take the 'single unified view' of data which is in the fact table (which contains numerous columns (think of a single Amazon purchase order item) - they can be categorized into dimensions, and in each dimension, even be hierarchically grouped - an example would be 'location'), and DERIVE additional tables, with data pre-aggregated along those (hierarchies of) dimensions. This lets us slice-and-dice (along dimensions), and zoom in/out (along just one dimension), all without expensive querying at runtime (on billions of rows), because the 'group by' calculations have been done already (that resulted in those aggregated data tables).'

* **13.6** *Online Analytical Processing*

Online analytical processing (OLAP) is a BI style whose systems share three main characteristics:

- Multidimensional data analysis techniques - The most distinctive characteristic of modern OLAP tools is their capacity for multidimensional analysis, in which data is processed and viewed as part of a multidimensional structure. This type of data analysis is particularly attractive to business decision makers because they tend to view business data as being related to other business data.

- Advanced database support - To provide a seamless interface, OLAP tools map the data elements from the data warehouse and the operational database to their own data dictionaries. This metadata is used to translate end-user data analysis requests into the proper (optimized) query codes, which are then directed to the appropriate data sources
- Easy-to-use end-user interfaces - The end-user analytical interface is one of the most critical OLAP components. When properly implemented, an analytical interface permits the user to navigate the data in a way that simplifies and accelerates decision making or data analysis

* 13.6-d *OLAP Architecture*

The OLAP architecture is designed to meet ease-of-use requirements while keeping the system flexible.

An OLAP system has three main architectural components :

1. Graphical user interface (GUI)
2. Analytical processing logic
3. Data-processing logic

* 13.6-e *Relational OLAP*

Relational online analytical processing (ROLAP) provides OLAP functionality by using relational databases and familiar relational query tools to store and analyze multidimensional data.

ROLAP adds the following extensions to traditional RDBMS technology:

- Multidimensional data schema support within the RDBMS
- Data access language and query performance optimized for multidimensional data
- Support for very large databases (VLDBs

* 13.6-f *Multidimensional OLAP*

Multidimensional online analytical processing (MOLAP) extends OLAP functionality to multidimensional database management systems (MDBMSs). An MDBMS uses proprietary techniques to store data in matrix-like n-dimensional arrays. MOLAP's premise is that multidimensional databases are best suited to manage, store, and analyze multidimensional data. Conceptually, MDBMS end users visualize the stored data as a three-dimensional cube known as a data cube.

RELATIONAL VS. MULTIDIMENSIONAL OLAP

CHARACTERISTIC	ROLAP	MOLAP
Schema	Uses star schema Additional dimensions can be added dynamically	Uses data cubes Multidimensional arrays, row stores, column stores Additional dimensions require re-creation of the data cube
Database size	Medium to large	Large
Architecture	Client/server Standards-based	Client/server Open or proprietary, depending on vendor
Access	Supports ad hoc requests Unlimited dimensions	Limited to predefined dimensions Proprietary access languages
Speed	Good with small data sets; average for medium-sized to large data sets	Faster for large data sets with predefined dimensions

* 13.7 *SQL Extensions for OLAP*

The proliferation of OLAP tools has fostered the development of SQL extensions to support multidimensional data analysis.

* The ROLLUP Extension - is used with the GROUP BY clause to generate aggregates by different dimensions. As you know, the GROUP BY clause will generate only one aggregate for each new value combination of attributes listed in the GROUP BY clause. The ROLLUP extension goes one step further; it enables you to get a subtotal for each column listed except for the last one, which gets a grand total instead.

The syntax of the GROUP BY ROLLUP command sequence is as follows:

```
SELECT column1 [, column2, …], aggregate_function(expression)
FROM table1 [, table2, …]
[WHERE condition]
GROUP BY ROLLUP (column1 [, column2, …])
[HAVING condition]
[ORDER BY column1 [, column2, …]]
```

* The CUBE Extension - is also used with the GROUP BY clause to generate aggregates by the listed columns, including the last one. The CUBE extension enables you to get a subtotal for each column listed in the expression, in addition to a grand total for

the last column listed.

The syntax of the GROUP BY CUBE command sequence is as follows:

```
SELECT column1 [, column2, …], aggregate_function(expression)
FROM table1 [, table2, …]
[WHERE condition]
GROUP BY CUBE (column1 [, column2, …])
[HAVING condition]
[ORDER BY column1 [, column2, …]]
```