Global Scientific JOURNALS
GSJ: Volume 10, Issue 2, February 2022, Online: ISSN 2320-9186
www.globalscientificjournal.com

Study of Data Warehouse Architecture

Mohmmed Asharaff
Research Scholar, Division of International Programmes, Universidad Azteca -Azteca University, Chalco, Mexico
byjumannar@gmail.com

Abstract

One of the most important components of decision support, which has become a rising emphasis of the database industry, is data warehousing. Many commercial products and services are now available, and these services are provided by all of the main database management system manufacturers. Decision support, in contrast to normal online transaction processing programs, places some specific demands on database technology. Back-end tools for extracting, cleaning, and loading data into a data warehouse, front-end client tools for querying and data analysis, metadata management and warehouse management, and back-end tools for extracting, cleaning, and loading data into a data warehouse are all defined in this paper, as well as back-end tools for extracting, cleaning, and loading data into a data warehouse.

Keywords: Data warehouse, decision support, on-line transaction processing, database and front-end client tools.

Introduction

A data warehouse is a subject-oriented, integrated, time-variant, and non-volatile data collection that helps management make decisions.

Subject-Oriented: A data warehouse can be used to investigate a specific topic. For example, "sales" can be a specific topic.

Integrated: A data warehouse combines information from various sources. For example, source A and source B may have distinct ways of identifying a product, but there will only be one way of identifying a product in a data warehouse.

Time-Variant: A data warehouse is used to store historical data. In a data warehouse, for example, data from 3 months, 6 months, 12 months, or even older can be retrieved. This is in contrast to a transactional system, which typically just keeps the most recent data. A transaction system, for example, may save a client's most recent address, whereas a data warehouse may store all addresses linked with a customer.

Non-volatile: Once data is stored in a data warehouse, it cannot be changed. In a data warehouse, historical data should never be

changed.

Overall Architecture

A relational database management system server serves as the central repository for informational data in the data warehouse architecture. Data and processing for operational purposes are kept distinct from data warehouse processing. This primary data store is surrounded by a number of critical components that work together to make the entire environment functional, managed, and available to both operational systems and end-user query and analysis tools.

The warehouse's raw data is often derived from operational applications. Data is cleansed and turned into an integrated structure and format when it enters the warehouse. Conversion, summarization, filtering, and condensing of data may all be part of the transformation process. Because the data is historical, the warehouse must be able to store and manage enormous volumes of data as well as multiple data architectures for the same database throughout time.

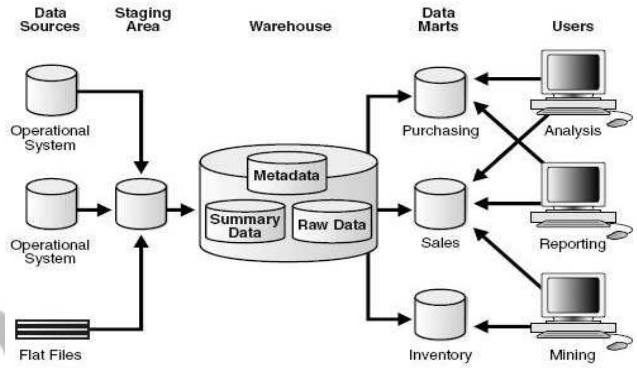

Fig 1. Architecture of data warehouse

Major Components of Data Warehousing

a. Data Warehouse Database

The data-warehousing environment's cornerstone is the core data warehouse database. The relational database management system (RDBMS) is nearly always used to create this database. Traditional RDBMS systems are geared for transactional database processing, hence this type of deployment is frequently limited. Certain data warehouse characteristics, such as enormous database sizes, ad hoc query processing, and the necessity for flexible user view generation, including aggregates, multi-table joins, and drill-downs, have fueled the development of several technology approaches to the data warehouse database. These strategies include:

- Scalable parallel relational database systems that use shared memory, shared disk, or shared-nothing models on a variety of multiprocessor configurations (symmetric multiprocessors, massively parallel processors, and/or clusters of uni- or multiprocessors).
- A novel way to speeding up a typical RDBMS by avoiding relational table scans with new index structures.

- Multidimensional databases (MDDBs) rely on proprietary database technology; a dimensional data model, on the other hand, can be implemented using a standard relational database management system (RDBMS). Multidimensional databases are intended to circumvent any limits imposed by the relational data model on the warehouse. MDDBs provide on-line analytical processing (OLAP) tools, which are part of a larger group of data warehousing components known as data query, reporting, analysis, and mining tools. Tools for sourcing, acquiring, cleaning, and transformation

Extraction of data from operational systems and conversion to a format suitable for informational applications that run on the data warehouse takes up a large portion of the deployment work.

All of the conversions, summarizations, key modifications, structural changes, and condensations required to transform heterogeneous data into information that can be used by the decision support tool are performed by the data sourcing, cleanup, transformation, and migration tools. COBOL programs, MVS job-control language (JCL), UNIX scripts, and SQL data definition language (DDL) are among the programs and control statements they create to transport data into the data warehouse for numerous operating systems. The meta data is also maintained by these technologies. The following features are available:

- Removing unneeded data from operational databases
- Converting to standard data names and definitions
- Creating defaults for missing data
- Adapting to changes in source data definitions

Data sourcing, cleansing, extraction, transformation, and migration tools must cope with a number of key difficulties, including:

- Database inconsistency. Data models, data access languages, data navigation, operations, concurrency, integrity, and recovery are all significantly varied between DBMSs.
- Data heterogeneity is a problem. Homonyms, synonyms, unit compatibility (US vs metric), multiple properties for the same thing, and different ways of describing the same fact are all examples of how data is described and used in different models.

These programs can help you save a lot of time and work. However, there are some severe flaws. Many available tools, for example, are generally useful for simpler data extraction. For more intricate data extraction methods, bespoke extract techniques are frequently required.

b.Meta Data

The data warehouse's meta data is information about the data warehouse. It's used to create, maintain, manage, and access the data warehouse. There are three types of meta data:

- Technical meta data, which offers information on warehouse data for warehouse designers and administrators to utilize in developing and managing warehouses.
- Business meta data, which contains information that provides users with an easy-to-understand view of the data warehouse's information.

Meta data, on the other hand, gives users interactive access to assist them interpret information and find data. One of the challenges with meta data is that many data extraction tools' meta data gathering capabilities are still in their infancy. As a result, creating a meta data interface for users is frequently required, which can result in some duplication of labor.

A meta data repository and supporting software are used to handle meta data. Meta data repository management software, which is commonly installed on a workstation, can be used to map source data to target databases, develop code for data transformations, integrate and convert data, and regulate data movement.

Users' techniques to reviewing the results of their requests for information are likely to move from relatively basic manual analysis for trends and exceptions to agent-driven analysis based on user-defined thresholds as their interactions with the data warehouse grow. The meta data repository also stores the definitions of these thresholds, the configuration parameters for the software agents that use them, and the information directory showing where the proper sources for the information may be accessed.

c. Access Tools

Data warehousing's main goal is to give information to corporate users for strategic decision-making. Front-end tools are used by these users to interface with the data warehouse. Although many end users develop skill in the tools, many of these tools require the assistance of an information specialist. Query and reporting tools, application development tools, online analytical processing tools, and data mining tools are the four primary categories of tools.

d. OLAP Tools

Users can examine data using complex, multidimensional perspectives with OLAP tools, which are based on the principles of dimensional data models and related databases. Product performance and profitability, the efficiency of a sales program or marketing campaign, sales forecasting, and capacity planning are just a few examples of typical commercial applications. These tools are based on the assumption that data is structured in a multidimensional model.

The capacity to successfully use information is a vital success component for today's

businesses. Data mining is the process of employing artificial intelligence, statistical, and mathematical approaches to identify new significant connections, patterns, and trends in massive amounts of data kept in a warehouse

e. Data Marts

In the data warehouse sector, the concept of a data mart is generating a lot of buzz and drawing a lot of attention. Data marts are frequently promoted as a cost-effective and time-saving alternative to building a data warehouse. However, different people interpret the phrase "data mart" differently. A data store that is subordinate to an integrated data warehouse is a strict definition of this phrase. The data mart refers to a set of data (commonly referred to as a subject area) that has been developed for a certain set of consumers. A data mart could be a collection of data that has been denormalized, summarized, or aggregated. Rather than a physically separate data repository, such a set could be stored on the data warehouse in some cases. In most cases, however, the data mart is a physically separate data store that resides on a different database server, which is frequently a local area network that serves a specific user group. Sometimes a data mart is just relational OLAP technology, which generates analysis. These data marts, also known as dependent data marts, because their data is sourced from the data warehouse, have a high value because different users can access the information views derived from the single integrated version of the data regardless of how they are deployed or how many different enabling technologies are used.

Unfortunately, false claims regarding data marts' ease of use and low cost can lead to companies or suppliers wrongly portraying them as a replacement for data warehouses. This point of view describes independent data marts as fragmented point solutions to a variety of enterprise business concerns. In the context of a larger technological or application architecture, this style of solution should be used sparingly. Indeed, it lacks the data integration component that is at the heart of the data warehousing notion. Each data mart makes its own assumptions about how to consolidate data, and data from different data marts may not be consistent.

Furthermore, the concept of an independent data mart is risky because, as soon as the first data mart is built, other companies, groups, and subject areas within the company begin to establish their own data marts. As a result, you end up with a situation where various operational systems feed multiple non-integrated data marts with data content, task scheduling, connectivity, and management that often overlap. In other words, you've turned a difficult many-to-one problem of constructing a data warehouse from operational and external data sources into a

sourcing and management nightmare.

f. Data Warehouse Administration and Management

Data warehouses can be up to four times the size of equivalent operational databases, with some approaching terabytes in size depending on how much history must be stored. They are not synced with the operational data in real time, but they can be updated as frequently as once a day if the application requires it.

Furthermore, almost all data warehouse packages come with gateways that allow users to access numerous enterprise data sources without having to rewrite applications to interpret and use the data. Furthermore, in a heterogeneous data warehouse, the various databases are located on different platforms, necessitating the employment of inter-networking solutions. It is self-evident that this environment must be managed.

Security and priority management, data quality checks, managing and updating meta data, auditing and reporting data warehouse usage and status, purging data, replicating, sub setting, and distributing data, backup and recovery, and data warehouse storage management are all part of managing data warehouses.

Information Delivery System

The information delivery component is used to allow users to subscribe to data warehouse information and have it delivered to one or more destinations according to a user-defined schedule. To put it another way, the information delivery system delivers data and other information objects housed in warehouses to other data warehouses and end-user products like spreadsheets and local databases. Information may be delivered based on the time of day or the completion of an external event. The delivery systems component's reasoning is based on the notion that once the data warehouse is deployed and operational, its customers do not need to be aware of its location or upkeep. They only require the report or an analytical view of data at a particular point in time. With the widespread use of the Internet and the World Wide Web, a distribution system like this may take advantage of the Internet's ease by distributing warehouse-enabled information to thousands of end-users via the ubiquitous global network.

In fact, the Web is transforming the data warehousing landscape because the aims of the Web and data warehousing are essentially similar at a high level: simple access to information. When the right information reaches people who need it, when they need it, and when they need it the most, the value of data warehousing is maximized. However, many businesses have struggled to provide end users with the access they require due to sophisticated client/server systems. When people are physically separated from the data warehouse, the problems become even more difficult to fix. By providing consumers with universal and very inexpensive access to data, the Web alleviates many of these concerns.

When you combine this access with the capacity to deliver essential information on demand, you get a web-based information delivery system that enables users from all over the world to conduct sophisticated business-critical analysis and participate in group decision-making.

Back End Tools and Utilities

For populating warehouses, data warehousing systems use a variety of data extraction and cleaning tools, as well as load and refresh utilities.

a. Data Extraction

Gateways and standard interfaces (such as Information Builders EDA/SQL, ODBC, Oracle Open Connect, Sybase Enterprise Connect, Informix Enterprise Gateway) are commonly used to extract data from "external" sources.

b. Data Cleaning

Because a data warehouse is used to make decisions, it is critical that the data in the warehouse be accurate. However, because enormous amounts of data from various sources are involved, there is a considerable risk of data inaccuracies and abnormalities. As a result, instruments that assist in the detection and correction of data abnormalities can be quite profitable.

Inconsistent field lengths, descriptions, value assignments, missing entries, and violations of integrity constraints are all examples of situations where data cleaning is required. Optional fields in data entry forms, not unexpectedly, are a major source of data inconsistency.

Data cleaning tools are divided into three categories, each with its own set of features. Simple transformation rules can be set in data migration tools, such as "replace the string gender by sex." Prism's Warehouse Manager is an example of a popular tool of this type. Data scrubbing tools scrub data using domain-specific information (for example, postal addresses). To clean data from different sources, they frequently use parsing and fuzzy matching algorithms. Some programs allow you to select a source's "relative cleanliness." This category includes tools like Integrity and Trillum. Scanning data with data auditing technologies enables the discovery of rules and relationships (or the detection of violations of established rules). As a result, such technologies could be classified as data mining tools. For example, a program like this can notice a worrisome tendency (based on statistical analysis) that a particular auto dealer has never had any complaints.

c. Load

Data must be loaded into the warehouse after it has been extracted, cleaned, and transformed. Checking integrity constraints; sorting; summarization, aggregation, and other computation to produce the derived tables stored in the warehouse; establishing indices and other access paths; and partitioning to numerous destination storage locations may all require additional preparation. Batch load utilities are commonly used for this purpose. A load utility

must allow the system administrator to monitor status, cancel, suspend, and continue a load, and restart after failure without losing data integrity, in addition to populating the warehouse.

Data warehouse load utilities must deal with substantially bigger data quantities than operational databases. The warehouse can only be brought offline for a short period of time (typically at night) to refresh it. Sequential loads might take weeks or months to complete, for example, loading a terabyte of data. As a result, pipelined and partitioned parallelism are frequently used.

A full load has the benefit of being able to be viewed as a large batch transaction that creates a new database. The current database can still allow queries while the load transaction is in progress; when the load transaction commits, the current database is replaced with the new one. When periodic checkpoints are used, the operation can be restarted from the last checkpoint if a failure occurs during the load.

Even with parallelism, though, a full load may take too long. To limit the amount of data that must be included into the warehouse, most commercial utilities (for example, RedBrick Table Management Utility) use incremental loading during refresh. Only the tuples that have been changed are added. The load process, on the other hand, is now more difficult to manage. Because the incremental load interferes with ongoing queries, it is handled as a series of smaller transactions (which commit at regular intervals, such as every 1000 records or every few seconds), but this series of transactions must now be coordinated to ensure that derived data and indices are consistent with the base data.

d. **Refresh**

The process of refreshing a warehouse entail propagating source data update to the base data and derived data stored in the warehouse. When it comes to refreshing, there are two factors to consider: when to refresh and how to renew. The warehouse is usually replenished on a regular basis (e.g., daily or weekly). It is only necessary to propagate every update if some OLAP queries require current data (for example, up-to-the-minute market quotes). The warehouse administrator determines the refresh policy based on user needs and traffic, and it may differ for different sources.

The qualities of the source and the capabilities of the database servers may also influence refresh approaches. Extracting a full source file or database is normally prohibitively expensive, but in the case of legacy data sources, it may be the only option. Most modern database systems have replication servers that provide incremental update propagation from a primary database to one or more copies. When the sources change, these replication servers can be utilized to incrementally refresh the warehouse. Data shipping and transaction shipping are the two most used replication mechanisms.

A table in the warehouse is handled as a remote snapshot of a table in the source

database in data shipping (e.g., Oracle Replication Server, Praxis OmniReplicator). When the source table changes, rowtriggers are used to update the snapshot log table, and an automatic refresh schedule (or a manual refresh method) is set up to transfer the modified data to the remote snapshot.

Instead of triggers and a separate snapshot log table, the ordinary transaction log is utilized in transaction shipping (e.g., in Sybase Replication Server and Microsoft SQL Server). The transaction log is scanned at the source site for updates to replicated tables, and those log records are sent to a replication server, which packages the associated transactions and updates the replicas. Transaction shipping has the advantage of not requiring triggers, which can put a strain on operational source databases. However, because there are no standard APIs for accessing the transaction log, it cannot usually be used easily across DBMSs from various vendors.

Replication servers like these have been used to keep data warehouses up to date. The refresh cycles, on the other hand, must be carefully selected such that the volume of data does not overrun the incremental load utility.

The Three Major Advantages are

1. Combining data from many sources.
2. Conducting new types of analyses; and
3. Minimizing the cost of historical data access.

The Three Major Disadvantages Are

The biggest disadvantage is that maintaining a data warehouse can be expensive, which can be an issue if the warehouse is underutilized. Managers appear to have false expectations about the benefits of having a data warehouse.

Conclusion

This paper defines back-end tools for extracting, cleaning, and loading data into a data warehouse, front-end client tools for querying and data analysis, and tools for metadata management and warehouse management, as well as back-end tools for extracting, cleaning, and loading data into a data warehouse.

Reference

1. http://www.1keydata.com
2. Devlin, B.A., and P.T. Murphy, "An architecture for a business and information system," IBM Systems Journal, Vol. 27, No 1. 1988.
3. Power, D., "What are the advantages and disadvantages of Data Warehouses?" DSS News, Vol. 1, No. 7, July 31, 2000.
4. Zhuge, Y., H. Garcia-Molina, J. Hammer, J. Widom, “View Maintenance in a Warehousing Environment, Proc. of SIGMOD Conf., 1995.
5. Roussopoulos, N., et al., “The Maryland ADMS Project: Views R Us.” Data Eng. Bulletin, Vol. 18, No.2, June 1995.