International Journal of Advance Research in Science and Engineering
Volume No. 10, Issue No. 06, June 2021
www.ijarse.com
IJARSE
ISSN 2319 - 8354

Data Warehousing and OLAP Technology

[1]Anuag Solanki , [2]Prashant Singh, [3]Dhyanender Jain, [4]Amit Kumar Pandey

[1,2,3,4] Department of Information Technology

[1,2,3,4]Dr. Akhilesh Das Gupta Institute of Technology & Management, New Delhi, India.

anurag.solanki0021@gmail.com[1],prashant.ert@gmail.com[2],dhyanendra.jain@gmail.com[3], amitpandey33@gmail.com[4]

Abstract

Decision support, which is increasingly at the heart of database operations, requires data warehousing and online analytical processing (OLAP). Many commercial products and services are now available, and all major database management system manufacturers now offer these services. Unlike standard online transaction processing systems, business intelligence has some unique database technology requirements. The goal of this article is to provide an overview of data storage technologies and OLAP, with a focus on their emerging needs.

Keywords: OLAP, Business intelligence, Data storage technologies, Data warehousing.

Introduction:

Data warehousing is a set of decision support technologies aimed at making better decisions (executives, managers, knowledge workers and analysts) to make more informed and timely judgments. There has been tremendous growth in the past three years, both in terms of revenue and employment META Group, Warehousing, Hardware and Database Market Software and Tools Market is expected to grow from $2B to $3B USD grow by 2020. From $5 billion in 1995 to $8 billion in 1998, the industry has grown significantly. DataTechnology for warehousing has been effectively implemented. Used across a variety of sectors including manufacturing (e.g. retail (for example order fulfillment and customer service), wholesale (for order fulfillment and customer service), wholesale (for order fulfillment and user profiling and inventory management services (for damage, risk and credit analysis) , transport, map analysis and fraud detection (to manage a fleet), Telecommunications (for call analysis and fraud detection), utilities (for power consumption analysis), and healthcare (for outcomes analysis) are all examples of industries that use data analytics. A data roadmap is presented in this work. Warehousing technologies, with an emphasis on the unique demands that data warehouses make on database management systems (DBMSs).

2. Architecture and End-to-End Process

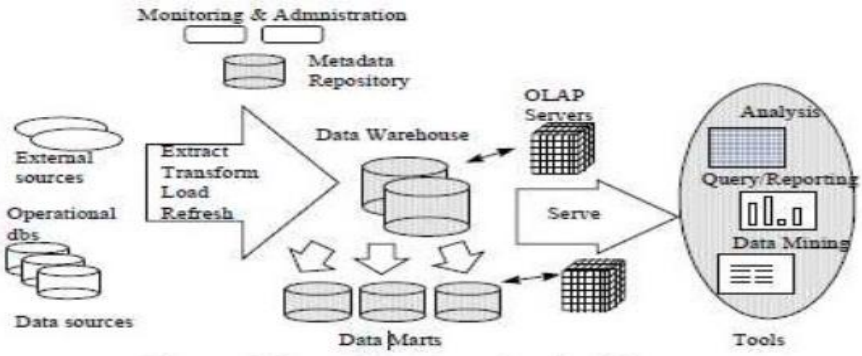

Figure 1. Data Warehousing Architecture

It includes tools for extracting data from multiple operational databases and external sources, cleaning, transforming, and integrating that data, loading data into the data warehouse, updating the warehouse periodically to reflect source updates, and deleting data from the warehouse , possibly in a slower archiving camp. In addition to the primary warehouse, there can be many departmental data marts. One or more warehouse servers store and manage data in the warehouse and data marts, and present multidimensional representations of data to a variety of front-end tools such as query tools, report writers, analysis tools, and data mining tools.

Finally, there are tools for monitoring and controlling the warehousing system, as well as a repository for storing and managing information.

3. Back End Tools and Utilities

For populating warehouses, data warehousing systems employ a range of data extraction and cleaning tools, as well as load and refresh utilities. Gateways and standard interfaces (such as Information Builders EDA/SQL, ODBC, Oracle Open Connect, Sybase Enterprise Connect, and Informix Enterprise Gateway) are commonly used to extract data from "external" sources.

Cleaning of data

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 many 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 may 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 cause of data inconsistency.

Load

Data must be put into the warehouse once 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 pathways; 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, halt, and continue a load, and restart after failure without losing data integrity, in addition to populating the warehouse.

Refreshing

A warehouse entails propagating adjustments to source data in order to update 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's only necessary if some OLAP queries require current data (for example, up-to-the-minute market quotations).

Every update must be propagated. The warehouse administrator determines the refresh policy based on user demands and traffic, and it may change for various sources.

4. Front-end tools and conceptual model

The multidimensional view of data in the warehouse is a common conceptual paradigm that impacts front-end tools, database design, and OLAP query engines. The objects of analysis in a multidimensional data model are a set of numeric measurements. Sales, budget, revenue, inventory, and ROI (return on investment) are examples of such metrics. Each of the quantitative measurements is based on a collection of dimensions that serve as the measure's context. The locality, product name, and date of sale, for example, can all be parameters connected with a sale amount. The dimensions are supposed to determine the measure in a unique way.

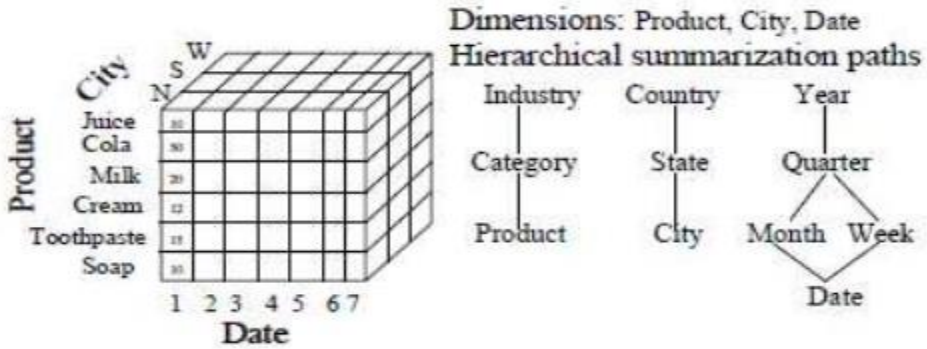

Figure 2. Multidimensional data

4. Front-end tools and conceptual model

Another distinguishing element of the OLAP conceptual model is its emphasis on measure aggregation by one or more dimensions as one of the essential features. computation and ranking of totals, for example each county's sales (or by each year). Other Comparing two measurements is a common procedure. (for example, sales and budget) gathered by the same person dimensions. Time is a dimension that has two dimensions. Particularly important for decision-making (e.g.,examination of trends). It is frequently advantageous to have built-in storage. calendars and other parts of the business dimension of time.

Tools for the Front End

The multidimensional data model evolved from the widespread view of company data popularised by PC spreadsheet tools widely used by business analysts. For OLAP, the spreadsheet remains the most engaging front-end application. The problem of providing an OLAP query environment may be summed up as that of effectively enabling spreadsheet operations across massive multi-gigabyte databases. Indeed, Arbor Corporation's Essbase solution employs Microsoft Excel as the front-end interface for its multidimensional engine.

5. Methodology for Database Design

MOLAP servers directly implement the multidimensional data model mentioned above. In the following part, we'll go through these in more detail.

When using a relational ROLAP server, however, the multidimensional model and operations must be translated into SQL queries and relations. In this part, we'll go through how to create relational database schemas that mirror multidimensional data views. In OLTP contexts, entity relationship diagrams and normalisation techniques are commonly utilised for database design. The database architectures advised by ER diagrams, on the other hand, are unsuited for decision support systems when querying and data loading performance (including incremental loads) are critical.

The multidimensional data model is often represented by a star schema in most data warehouses. A single fact table and a single table for each dimension make up the database. Each tuple in the fact table has a pointer (foreign key) to each of the dimensions that supply its multidimensional coordinates, as well as the numeric measurements for those coordinates. Each dimension table has columns that correspond to the dimension's characteristics. Figure 3 depicts a A star schema is an example.

Attribute hierarchies are not expressly supported by star schemas. As illustrated in Figure 4, snowflake schemas are a refinement of star schemas in which the dimensional hierarchy is

clearly reflected by normalising the dimension tables. This has advantages in terms of keeping the dimension tables up to date. The denormalized structure of dimensional tables in star schemas, on the other hand, may be better for exploring the dimensions.

6. Servers in the Warehouse

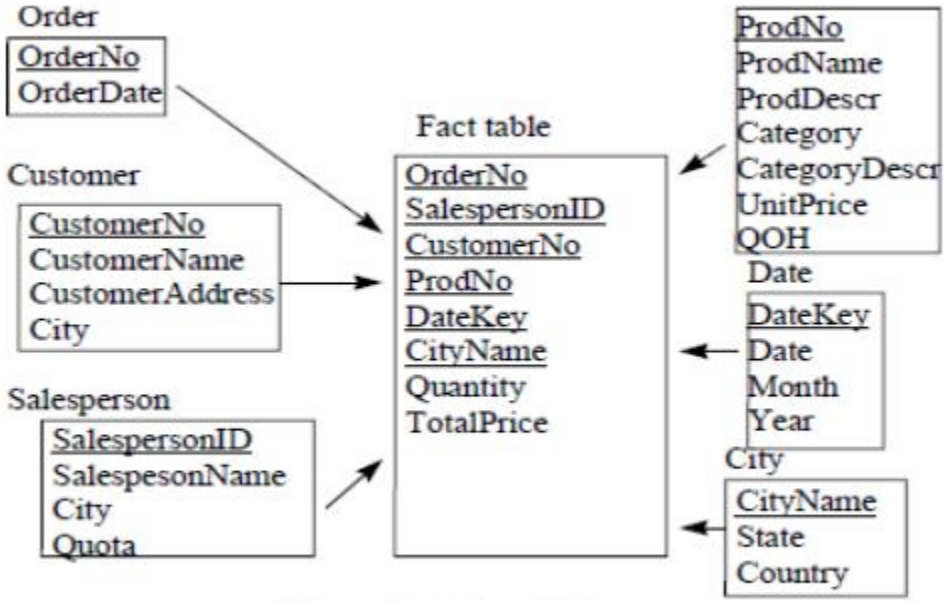

Figure 3. A Star Schema.

Large amounts of data may be stored in data warehouses. To respond to questions quickly, extremely efficient access mechanisms and query processing algorithms are required. A number of difficulties occur. Data warehouses, for starters, make use of redundant structures like indexes and materialised views. A critical physical design difficulty is deciding which indices to create and which perspectives to realise. The next step is to make optimal use of existing indexes and materialised views in order to respond to queries.

Another significant issue is the optimization of complicated queries. Furthermore, while fast index scans may be highly successful for data-selective searches, data-intensive queries require sequential scans. As a result, increasing scan efficiency is critical. Finally, parallelism must be taken advantage of in order to minimise query response times. It is impossible to go into detail on any of these topics in this short paper. As a result, we will simply touch on the highlights.

Index Structures and How to Use Them

A variety of indices-based query processing algorithms are helpful. Index intersection, for example, can be used to utilise the selectivities of many criteria. Union of indexes is another

helpful index operation. These index procedures can drastically minimise, and in some cases completely remove, the requirement to consult the underlying tables. Bit map indices, which provide efficient index operations (e.g., union, intersection), can be used by warehouse servers. Consider a domain value d that corresponds to a leaf page in an index structure.

A list of the record ids (RIDs) of records that contain the value is often included on such a leaf page.

d. Bit map indices, on the other hand, utilise a bit vector representation of the aforementioned RID list, with one bit set for each record when the domain value for that record is d. The bit map index is just a different representation of the RID list, rather than a new index structure.

Because the bit vector representation of RID lists may speed up index intersection, union, join, and aggregation11, the bit map index has gained in prominence. If we have a query like column1 = d & column2 = d', we can get the qualifying records by adding the two bit vectors together.

While such representations can be quite beneficial for low cardinality domains (e.g., gender), they can also be very useful for high cardinality domains (e.g., race).via effective for greater cardinality domains Bitmaps are compressed (e.g., run length encoding).

Model 204 was the first to employ bitmap indices. However, many products nowadays (e.g., Sybase) support them. IQ). The decision on which to use is a fascinating one. indexing characteristics In overall, this is a fantastic opportunity. a physical inquiry that must be answered

the procedure for creating a database In addition to indices on, there are indexes on single tables, star schemas' specialised nature makes join indices very appealing for decision-making support.

Consider Figure 3 as an example of a schema. A join index on City can keep track of the RIDs of the tuples in the fact table that relate to sales in that city for each city. As a result, a join index effectively precalculates a binary join. Precomputed n-way joins can be represented using multikey join indices. For example, a multidimensional join index from (Cityname, Productname) to the fact table may be built over the Sales database. As a result, the index entry for (Seattle, jacket) leads to RIDs in the Sales database for those tuples with the aforesaid combination. Using a multidimensional join index instead of the intersection of distinct indexes on Cityname and Productname can save time and money. Join indices can be used in a variety of ways. RID list formats for efficient join processing. The Use of Materialized Views Many data warehouse queries need As a result, aggregates are used to summarise data. As a result, in addition to indices, materialisation is important. Many typical tasks can be aided by summary data. queries. In the case of an investment, for example. a substantial majority of the searches are likely to be based on the most recent quarter's results in addition to the current fiscal year Having an overview of data These options can considerably improve query performance. processing.

Complex SQL Query Transformation

The challenge of devising effective strategies for It's been a priority to process complicated requests. query optimization piqued my curiosity.

In certain ways, decision support systems serve as a proving ground for previously researched concepts. Only a few of the most important contributions will be discussed.

When certain syntactic limitations are fulfilled, complicated SQL queries including nested subqueries can be "unnested" by converting them into single block SQL queries17 18 19 20. Another approach to optimise nested subqueries is to use semi-join-like methods to reduce the number of invocations and batch the execution of inner subqueries21 22.

Similarly, the difficulty of flattening queries with views has piqued people's curiosity. It is commonly established that participating views are SPJ queries. When one or more of the views incorporate aggregation, the situation becomes more complicated.

Parallel Processing

Parallel processing is very important in processing. databases of enormous size Teradata was a forerunner in several of the technologies that are now commonplace. a crucial technology Database providers from all across the world Data partitioning is now available in management systems. Technology technology for parallel query processing The Dewitt and Gray's essay gives an outline of this region28 One intriguing method that is related to The decision-making environment is read-only. The practise of piggybacking scans requested by other systems is known as there are several queries (used in Redbrick). Piggybacking Scanning minimises both the overall amount of labour and the time it takes to respond. by combining many contemporaneous scans requests.

Query Processing Server Architectures

Traditional relational servers were not designed to handle multidimensional views of data through the clever use of indices and other criteria. All relational DBMS providers, on the other hand, have moved quickly to satisfy these new standards. There are three types of servers designed expressly for decision support, in addition to typical relational servers.

Redbrick, for example, is an example of a specialised SQL server. In read-only environments, the goal is to offer sophisticated query language and query processing capabilities for SQL queries over star and snowflake schemas.

• ROLAP Servers: ROLAP Servers are intermediary servers that reside between a relational back end server (where the warehouse data is kept) and client front end tools. Microstrategy is an

example of a server of this type. They use specialised middleware to expand regular relational servers to effectively perform multidimensional OLAP queries, and they usually optimise for certain back end relational servers.

They determine which views should be materialised, rewrite user queries in terms of the materialised views, and create multi-statement SQL for the back end server. They also offer other services like query scheduling and resource assignment (for example, to avoid runaway queries). A recent tendency has been to tailor ROLAP servers for domain-specific ROLAP tools. The fundamental advantage of ROLAP servers is that they take use of relational systems' scalability and transactional characteristics. Inherent incompatibilities between OLAP-style querying and SQL (e.g., absence of sequential processing, column aggregation) might lead OLAP servers to have performance bottlenecks.

MOLAP Servers: These servers use a multidimensional storage engine to enable a multidimensional view of data. Through direct mapping, it is now able to implement front-end multidimensional queries on the storage layer.

Essbase is one example of such a server (Arbor). Although such a method has strong indexing qualities, it has low storage consumption, particularly when the data set is sparse.

Extensions for SQL

Several modifications to SQL have been suggested or implemented in extended relational servers to make the phrasing and processing of OLAP queries easier. Some of these add-ons are mentioned further down. Support for rank and percentile (e.g., all goods in the top 10% percentile or the top 10 products by total Sale) as well as a range of financial analysis functions are included in this extended family of aggregate functions (mean, mode, median).

• Reporting Characteristics: Aggregate features assessed across a time interval, such as a moving average, are frequently used in business analysis reports.In addition, being able to offer breakpoints and running totals is critical. These primitives are available in Redbrick's SQL extensions.

Comparisons: An article by Ralph Kimball and Kevin Strehlo provides an excellent overview of SQL's shortcomings in performing common business comparisons, such as comparing the difference between total projected and total actual sales by quarter, where projected and actual sales are columns of a table31. Multiple sequential scans may be required to execute such queries in a simple manner.

7. Warehouse and Metadata Management

Because a data warehouse mirrors an organization's business strategy, metadata management is an important part of the warehousing architecture.There must be many distinct types of metadata.

managed. Administrative metadata comprises all of the information required to set up and operate a warehouse, including: definitions of the warehouse schema, derived data, dimensions and hierarchies, predefined queries and reports; data mart locations and contents; physical organisation such as data partitions; data extraction, cleaning, and transformation rules; data refresh and purging policies; user profiles, user authorization, and access control policies Business metadata covers things like business words and definitions, data ownership, and billing procedures.

8. Issues in Research

The significant technological obstacles in building and deploying decision assistance systems have been discussed. Despite the abundance of commercial products and services, there are still a number of promising research areas. We'll simply touch on a handful of them in this article. Some of these topics are being studied by researchers, while others, particularly in regard to data warehousing, have gotten merely a passing glance.

References

1. Inmon, W.H., Building the Data Warehouse. John Wiley, 1992.
2. Codd, E.F., S.B. Codd, C.T. Salley,“Providing OLAP (On-Line Analytical Processing)
3. Kimball, R. The Data Warehouse Toolkit.John Wiley, 1996.
4. Barclay, T., R. Barnes, J. Gray, P.Sundaresan, “Loading Databases using Dataflow Parallelism.” SIGMOD Record,Vol.23, No. 4, Dec.1994.
5. Blakeley, J.A., N. Coburn, P. Larson.“Updating Derived Relations: Detecting Irrelevant and Autonomously Computable Updates.” ACM TODS, Vol.4, No. 3, 1989.
6. Gupta, A., I.S. Mumick, “Maintenance of Materialized Views: Problems, Techniques,and Applications.” Data Eng. Bulletin, Vol.18, No. 2, June 1995.
7. Zhuge, Y., H. Garcia-Molina, J. Hammer, J.Widom, “View Maintenance in a Warehousing Environment, Proc. Of SIGMOD Conf., 1995.
8. Roussopoulos, N., et al., “The Maryland ADMS Project: Views R Us.” Data Eng.Bulletin, Vol. 18, No.2, June 1995.
9. O’Neil P., Quass D. “Improved Query Performance with Variant Indices”, To appear in Proc. of SIGMOD Conf., 1997.
10. O’Neil P., Graefe G. “Multi-Table Joins through Bitmapped Join Indices” SIGMOD Record, Sep 1995.
11. Harinarayan V., Rajaraman A., Ullman J.D.“ Implementing Data Cubes Efficiently”Proc. of SIGMOD Conf., 1996.
12. Chaudhuri S., Krishnamurthy R.,Potamianos S., Shim K. “Optimizing Queries with aterialized Views” Intl.

Conference on Data Engineering, 1995.13. Levy A., Mendelzon A., Sagiv Y.“Answering Queries Using Views” Proc. Of PODS, 1995.
14. Yang H.Z., Larson P.A. “Query Transformations for PSJ Queries”, Proc. Of VLDB, 1987.
15. Kim W. “On Optimizing a SQL-like Nested Query” ACM TODS, Sep 1982.
16. Ganski,R., Wong H.K.T., “Optimization of Nested SQL Queries Revisited ” Proc. Of SIGMOD Conf., 1987.
17. Dayal, U., “Of Nests and Trees: A Unified Approach to Processing Queries that Contain Nested Subqueries, Aggregates and Quantifiers” Proc. VLDB Conf., 1987.