RDBMS:OLAP & OLTP

-Aman Khandwe

April 8th, 2022

Abstract

A relational database is a digital database that uses the relational data model. A relational database management system is a system for maintaining relational databases (RDBMS). Many relational database systems allow you to query and maintain the database using SQL (Structured Query Language).It's a programme that lets us create, delete, and update relational databases. A relational database is a database system that stores and retrieves data in the form of rows and columns in a tabular format. The main DBMS, such as SQL, My-SQL, and ORACLE, are all based on relational DBMS concepts.The fact that the values of each table are connected to each other is the cornerstone of relational DBMS. It is capable of handling enormous amounts of data and easily simulating queries.Here in this paper we will look at the realtional model and will look at the integrity constraints and how to store objects in relational database, followed by advantages, disadvantages, and some applications of the same, and finally we will try to have a deeper look at the two major applications of RDBMS, OLAP (Online analytical processing) and OLTP (Online transaction processing).

Table of Contents

Introduction

RDBMS is a concept that is used to give the user with the ability to store and retrieve data that is connected with any other relation, also known as the Relational. A relation in a relational database is based on a

relational scheme and is made up of a number of properties. A relational database is composed of several relations and a relational database architecture. Users have the ability to access and alter the data stored in the database in a convenient and functional manner. Furthermore, the DBMS exerts centralised control over the database, prohibits unauthorised users from accessing the data, and secures data privacy.The fact that the values of one table are connected to others is the foundation of relational DBMS. It is capable of handling bigger amounts of data and quickly simulating queries. Relational Database Management Systems ensure data integrity by imitating the following characteristics:

1. Entity Integrity: No two records in the database table may be identical.
2. Referential Integrity: Only those rows of those tables that are not utilised by other tables can be erased. Otherwise, data discrepancy may occur.
3. User-defined Integrity: Rules based on confidentiality and access that are established by the users.
4. Domain integrity: The database table columns are contained inside certain specified limitations based on default values, data type, or ranges.

In RDBMS, data must be saved in a DB file in tabular form, that is, in the form of rows and columns. Each table row is referred to as a record/tuple. The cardinality of the table refers to the collection of such items. Each column in the table is referred to as an attribute/field. The arity of the table is a collection of similar columns. There can be no duplicate records in the DB table. By employing a candidate key, data duplication is eliminated. A Candidate Key is a collection of properties that must be present in order for each record to be uniquely identified. Tables are linked to one another via foreign keys.Database tables also support NULL values, which means that if the values of any of the table's elements are not filled in or are missing, the value becomes a NULL value, which is not equal to zero. RDBMS often include data dictionaries and metadata collections that aid in data management. These provide for the programmatic support of well-defined data structures and relationships. Data storage management is a typical RDBMS functionality, and it has been characterised by data objects ranging from binary large object strings to stored procedures. This type of data item extends the scope of conventional relational database operations and may be handled in a number of ways by different RDBMSes.For acessing data in RDBMs SQL is used. Data manipulation language and data definition language statements are its primary language components. RDBMS employs complicated algorithms that allow several concurrent users to access the database while preserving data integrity. Another overlay function provided by the RDBMS for the fundamental database when used in business settings is security management, which enforces policy-based access.

Relational Model

The relational model is based on the mathematical idea of a relation, which is represented practically as a table. Relations are utilized in the relational model to store information about the items that will be represented in the database. A relation is represented as a two-dimensional table, with rows corresponding to individual records and columns corresponding to attributes. Attributes can occur in any sequence, and the connection remains the same and so conveys the same meaning. Domains are a very important aspect of the relational model. A domain is used to specify each attribute in a relation. Domains might be unique for each characteristic, or they can be shared by two or more attributes. The domain idea is significant because it allows the user to specify the meaning and source of values that attributes can carry in a centralized location. As a result, the system has more information accessible to it when performing a relational action, and operations that are semantically inaccurate can be avoided. The rows or tuples in the table are the

elements of a relation. Tuples can be arranged in any sequence, and the connection will remain the same and hence communicate the same meaning. The structure of a relation, together with a definition of the domains and any other constraints on potential values, is referred to as its intension, which is normally fixed unless the meaning of the relation is updated to incorporate more characteristics. The tuples are referred to as a relation's extension (or state), which varies over time. A unary relation, also known as a one-tuple, is a relation with only one attribute and has degree one. Binary refers to a relationship with two qualities, ternary refers to a relationship with three attributes, and n-ary refers to a relationship with more than three attributes. A relation's degree is a characteristic of the relation's intension. The cardinality of the relation, refers to the number of tuples in the relation, which changes when tuples are added or removed. The cardinality of a connection is a feature of its extension that is defined by the specific instance of the relation at any given time.

A relation has the following properties:

- The relation has a unique name that differs from the names of all other relations in the relational schema.
- There is only one atomic (single) value in each cell of the relation.
- Each attribute is given a unique name.
- An attribute's values are all from the same domain.
- There are no duplicate tuples; each tuple is unique.
- The order of the attributes is irrelevant.
- The order of tuples has no theoretical importance.

Integrity Constraints

A data model contains two additional components: a manipulative part that defines the sorts of operations that may be performed on the data, and a set of integrity constraints that verify the data's accuracy. The relational integrity restrictions will be discussed in this section. Since each attribute has a domain, there are constraints (known as domain constraints) that limit the set of values that may be assigned to the attributes of relations. There are also two key integrity rules, which are limits or limitations that apply to all database instances. Entity integrity and referential integrity are the two main rules of the relational model. Multiplicity and general constraints are two further forms of integrity constraints.It's important to grasp the idea of nulls before we can establish entity and referential integrity. Null denotes the absence of a value for an attribute that is presently unknown or irrelevant to this tuple.A null is not the same as a numeric value of zero or a text string with spaces; all are values, but a null signifies the lack of a value. So, nulls must be handled differently from other values. Now let us talk about the relational integrity rules.

Entity Integrity The main keys of base relations are subject to the first integrity rule. No attribute of a main key can be null in a base relation. A primary key, by definition, is a single identifier that is used to uniquely identify tuples. This indicates that no subset of the main key is adequate to identify tuples uniquely. Allowing a null for any component of a primary key implies that not all attributes are required to differentiate between tuples, which is contrary to the primary key's definition.

Referential Integrity Foreign keys are subject to the second integrity rule. If a relation has a foreign key, the value of the foreign key must either match a candidate key value of some tuple in the home relation or be completely null.Referential integrity constraints are used to preserve consistency between tuples in two relations and are expressed between two relations or tables.

General Constraints These are the additional rules set by database users or database administrators that define or limit a particular area of the business.Additionally, users can set additional constraints that the data must meet. For example, if the number of people who can work at a branch office is limited to 20, the user must be able to define this general limitation and expect the DBMS to enforce it. If the number of staff presently allocated to a branch is 20, it should not be able to add a new member of staff to the Staff relation in this situation. However, the amount of support for generic restrictions varies considerably from one system to the next.

Storing Objects in a Relational Database

Using an RDBMS as the underlying storage engine is one way to achieve persistence with an object-oriented programming language like C++ or Java. This necessitates mapping class instances (i.e., objects) to one or more tuples distributed across one or more relations.Consider the inheritance structure depicted in Figure 1, which includes a superclass called Staff and three subclasses called Manager, SalesPersonnel, and Secretary.

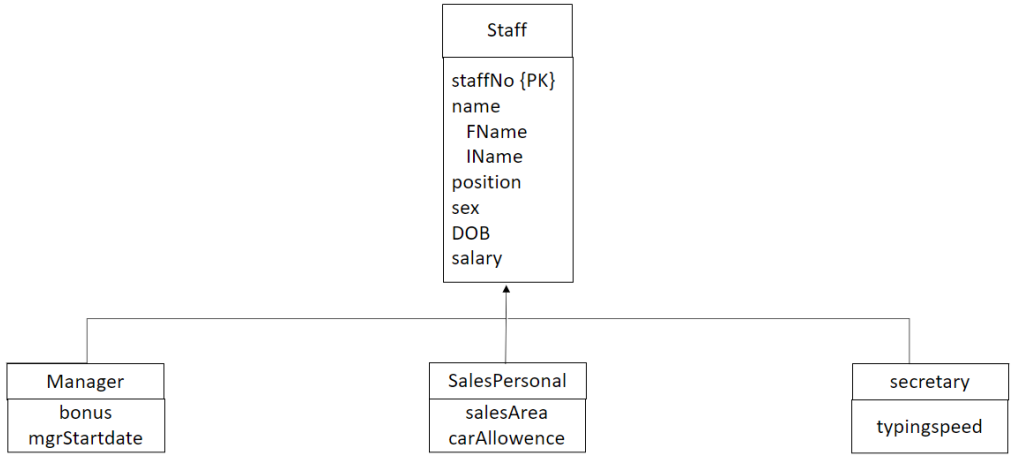

Figure 1: Inheritance hierarchy for Staff.

We have two main activities to undertake when dealing with this sort of class hierarchy:

1. Design the relations to represent the class hierarchy.
2. Create a plan for how objects will be accessible, which includes:

- developing code that decomposes objects into tuples and stores them in relations;
- writing code that reads tuples from relations and reconstructs the objects.

Now let us look at in depth about how to design the relations to represent the class hierarchy-

1.Mapping Classes to Relations

There are several strategies for mapping classes to relations, each of which results in a loss of semantic information. The code used to make objects persistent and read them back from the database is determined by the strategy used. We consider the following three options:

1. Map each class or subclass to a relation.
2. Map each subclass to a relation.
3. Map the hierarchy to a single relation.

1.1 Map each class or subclass to a relation

One method is to map each class or subclass with a relation.This would result in the following four relationships for the hierarchy shown in Figure 1:

1. Staff (staffNo, fName, lName, position, sex, DOB, salary).
2. Manager (staffNo, bonus, mgrStartDate).
3. SalesPersonnel (staffNo, salesArea, carAllowance).
4. Secretary (staffNo, typingSpeed).

Although we presume that the RDBMS supports the underlying data type of each attribute, this may not be the case, in that case we will need to create additional code to manage the translation from one data type to another. Unfortunately, we have lost semantic information with this relational schema: it is no longer evident which relation represents the superclass and which relation represents the subclasses. As a result, we'd have to embed this knowledge into each programme, which, as we've discussed before, can lead to code duplication and inconsistency.

1.2 Map each subclass to a relation

A second option is to map each subclass with a relation. This would result in the following three relations for the hierarchy shown in Figure 1:

1. Manager (staffNo, fName, lName, position, sex, DOB, salary, bonus, mgrStartDate)
2. SalesPersonnel (staffNo, fName, lName, position, sex, DOB, salary, salesArea, carAllowance)
3. Secretary (staffNo, fName, lName, position, sex, DOB, salary, typingSpeed)

Again, semantic information has been lost in this mapping: it is no longer evident whether these relations are subclasses of a single generic class. To get a list of all employees in this example, we'd have to choose the tuples from each relation and then combine the results.

1.3 Map the hierarchy to a single relation

A third option is to translate the complete inheritance hierarchy to a single relation, as shown below:

Staff (staffNo, fName, lName, position, sex, DOB, salary, bonus, mgrStartDate, salesArea, carAllowance, typingSpeed, typeFlag)

typeFlag is a discriminator that determines which type each tuple belongs to (for example, it may contain the value 1 for a Manager tuple, 2 for a SalesPersonnel tuple, and 3 for a Secretary tuple). In this mapping also w e've lost semantic information. This mapping will also result in a large number of nulls for attributes that do not apply to that tuple. The properties salesArea, carAllowance, and typingSpeed, will be null for a Manager tuple.

Advantages & Disadvantages of RDBMS

Advantages of RDBMs

A Relational Database system has numerous advantages over other types of databases. The following are a few important advantages:

1.Simple Model

A Relational Database system is the simplest basic type since it does not need any sophisticated structure or querying methods. It does not include time-consuming architecture operations such as hierarchical database structuring or definition. Because the structure is basic, it can be handled with simple SQL queries and does not necessitate the creation of sophisticated queries.

2.Data Accuracy

Multiple tables in a relational database system can be linked to one another using the primary key and foreign key principles. As a result, the data is non-repetitive. There is no possibility of data duplication. As a result, the accuracy of data in a relational database is greater than that of any other database system.

3.Easy Access to Data

There is no pattern or method for accessing data in the Relational Database System, whereas other types of databases can only be accessed by travelling via a tree or a hierarchical architecture. Anyone with data access can query any table in the relational database. Using join queries and conditional expressions, you may combine all or any number of linked tables to get the data you need. The resulting data may be updated

based on any column's values, on any number of columns, allowing the user to easily recover the relevant data as a consequence. It enables the user to select the desired columns to be included in the output, ensuring that only relevant data is presented.

4.Data Integrity

The integrity of data is a critical feature of the Relational Database system. Sturdy Data entries and validity validations guarantee that all data in the database is contained within appropriate arrangements and that the data required to create relationships is available. This relational consistency throughout the database tables helps to prevent records from being incomplete, isolated, or unconnected. Data integrity helps to ensure the relational database's other important features, such as ease of use, accuracy, and data stability.

5.Flexibility

A Relational Database system, on its own, contains attributes for scaling up and extending for greater lengths, since it is supplied with a malleable structure to fit continually changing requirements. This makes it easier to deal with the rising volume of data coming in, as well as to update and delete records as needed. This model accepts modifications made to a database setup as well, which may be done without crashing the data or other elements of the database. Individuals can quickly and simply insert, amend, or remove tables, columns, or individual data in the specified database system. A relational database is said to have no limit on the amount of rows, columns, or tables it may contain.The Relational Database Management System and the hardware provided by the servers limit development and transformation in any real application. As a result, these alterations may cause changes in various peripheral functional devices linked to the specific relational database system.

6.Normalization

The systematic approach is maintained to ensure that a relational database structure is free of any variations that might affect the integrity and correctness of the database's tables. A normalisation process establishes a set of rules, features, and goals for a relational database model's database structure and assessment.The goal of normalisation is to show several levels of data breakdow. Before going on to the following levels of normalisation, any level of normalisation is anticipated to be completed on the same level. Only after a relational database model meets the requisite requirements of the third normalisation form is it typically determined to be normalised. Normalization gives the idea that the database design is particularly sturdy and dependable.

7.High Security

Since the data is distributed across the tables of the relational database system, a few tables might be marked as confidential while others are not. Unlike other databases, a relational database management system makes this separation simple to build. When a individual attempts to log in with a username and password, the database can limit their degree of access by allowing them to access just the tables that they are authorised to work on, based on their access level.

8.Feasible for Future Modifications

Since the relational database system organises entries into tables depending on their categories, it's simple to add, remove, or update records that meet the most recent needs. This aspect of the relational database model allows the business to accommodate new requirements. By adhering to the fundamental properties of the relational database management system, any number of new or existing tables or columns of data may be introduced or updated based on the requirements supplied.

Disadvantages of RDBMs

1.Cost

The initial investment in a relational database is extremely high. A separate piece of software must be acquired in order to set up a relational database. In addition, the system should be maintained by a trained technician. All of this may be expensive, especially for small enterprises. Because relational databases have numerous appealing characteristics, they are rather expensive to utilise. As a result, obtaining such a database may seem difficult if the company has a limited budget.

2.Performance Issue

If there are a large number of rows and tables, the query will take longer to process the result because relational databases rely on rows and columns, performance might be sluggish. Furthermore, if there is a large amount of data in the system, it might slow down the working process, the presence of extra data not only slows down the system but also makes it more difficult to discover information. As a result, a relational database is recognised as a slower database. So we can say that the performance problem is a drawback of a relational database that most users may encounter when using it.

3.Physical Storage and Information Loss

Because it is made up of rows and columns, a relational database needs a large amount of physical memory. Each action is dependent on its own physical storage. Only by properly optimising the intended programmes can they be made to have the most physical memory.The RDBMS has limited space, and these storage devices cannot store new data if there is insufficient space. Because there is no longer any storage available, this data may be lost, causing issues in the future. As a result, relational databases are restricted and can cause a variety of problems if they are not adequately monitored.

4.Structure Limitations

A relational database has fields that have limits. In essence, limitations mean that it cannot handle additional information. It is vital to describe the data volume you wish to incorporate into any field while building the database. Because some of the search queries are or may be more exact than the original ones, data loss may occur. Even if additional information is supplied, data loss may occur. As a result, the exact amount of data volume that the field will be given must be defined.

Applications of RDBMs

In the recent decade, there have been tremendous developments in the computer sector. RDBMSs have gained significant popularity in database systems for classic commercial applications such as order processing, inventory control, banking, and airline bookings. However, standard RDBMSs have proven insufficient for applications with requirements that differ significantly from those of traditional corporate database applications.These applications include:

- Computer-aided manufacturing (CAM)- In addition to data pertaining to discrete production (such as automobiles on an assembly line), a CAM database includes data relevant to continuous production (such as chemical synthesis). There are applications that monitor information about the condition of the system, such as reactor vessel temperatures, flow rates, and yields, in chemical manufacture. There are additional applications that regulate various physical processes, such as opening valves, boosting the flow of cooling systems, and applying more heat to reactor vessels. These applications are frequently structured in a hierarchical structure, with a top-level application overseeing the whole factory and lower-level apps overseeing specific production processes.

- Computer-aided software engineering (CASE)- The stages of the software development lifecycle are stored in a CASE database: planning, requirements collecting and analysis, design, implementation, testing, maintenance, and documentation. Designs can be exceedingly huge, much like in CAD, and cooperative engineering is the norm. Software configuration management technologies, for example, enable simultaneous sharing of project design, code, and documentation. They also help with change management by tracking the dependencies between various components. Project management tools make it easier to coordinate a variety of project management duties, such as scheduling potentially extremely complicated interdependent tasks, estimating costs, and tracking progress.

- Network management systems- The delivery of communication services over a computer network is coordinated by network management systems. These systems handle network path management, issue management, and network planning, among other things. These systems, like the chemical production example we just reviewed, deal with complicated data and demand real-time performance and continuous operation. A telephone call, for example, may comprise a network switching device chain that routes a message from source to recipient, such as:

  Node Link Node Link Node Link Node

  Each Node represents a network device port, and each Link represents a bandwidth slice designated for that connection.A node, on the other hand, may be involved in several connections, and any database that is developed must maintain a complicated graph of interconnections. Network management systems must be able to move around this complicated structure in real time to route connections, identify faults, and balance loadings.

- Office information systems (OIS) and multimedia systems- Data linked to the computer control of information in a business, such as electronic mail, papers, invoices, and so on, is stored in an OIS database. We need to handle a larger range of data types other than names, addresses, dates, and money to give greater assistance in this area. Free-form text, photos, diagrams, and audio and video sequences are now supported by modern systems. Text, images, spreadsheets, and audio commentary, for example, can all be included in a multimedia document.

- Geographic information systems (GIS)- A geographic information system (GIS) database holds a variety of geographical and temporal data, such as those utilised in land management and undersea research. Much of the data in these systems comes from surveys and satellite pictures, and it's usually rather

extensive. Advanced pattern-recognition algorithms may be used to recognise characteristics based on form, colour, or texture.EOS (Earth Observing System), for example, is a constellation of satellites deployed by NASA in the 1990s to provide data to scientists interested in long-term changes in the earth's atmosphere, seas, and land. These satellites are expected to return more than a third of a petabyte of data per year. This information will be combined with information from other sources and saved in EOSDIS (EOS Data and Information System). Both scientists and nonscientists will be able to get information through EOSDIS. For example, students will be able to use EOSDIS to observe a simulation of global weather patterns.The massive size of this database, as well as the necessity to handle thousands of users with high amounts of information requests, will provide DBMSs with several hurdles.

- Some other examples of database applications are-

    1. Scientific and medical applications, which may contain complex data describing systems such as molecular models for synthetic chemical compounds and genetic material.
    2. For AI applications, expert systems can store knowledge and rule bases.
    3. Other applications using procedural data and complicated and interconnected objects.
    4. Apart from this OLAP and OLTP are two major applications of RDBMS on which we will be taking a deeper dig in upcoming sections.

OLAP & OLTP

The two names appear to be synonymous, however they relate to two different types of systems. Online transaction processing (OLTP) is a real-time data capture, storage, and processing system. Complex queries are used in online analytical processing (OLAP) to evaluate aggregated historical data from OLTP systems.

OLAP

On-Line Analytical Processing (OLAP) is an acronym for On-Line Analytical Processing. OLAP is a type of software that allows analysts, managers, and executives to get insight into data through quick, consistent, interactive access to a broad range of data views that have been transformed from raw data to represent the enterprise's true dimensionality as learnt by the client. Users may construct online descriptive or comparative summaries of data, as well as other analytics queries, using OLAP. It refers to a software and technology component that enables the gathering, storage, manipulation, and replication of multidimensional information for the purpose of analysis.It enables decision-makers to get insight into data by providing rapid, consistent, and interactive access to a wide range of data views that have been transformed from raw data to real-dimensionality characteristics. OLAP servers provide multidimensional data from the data warehouse or data marts to business users without having to worry about how or where the data is kept. Data storage difficulties should be considered in the physical construction and operation of OLAP servers.The dynamic multidimensional analysis of integrated corporate data is a hallmark of OLAP services. OLAP runs in a multiuser client/server mode and responds to queries consistently quickly, regardless of database size or complexity. It enables users to synthesise business data by allowing them to compare, customise, and analyse past and forecast data in a variety of data model scenarios.More drilling procedures are available in some OLAP systems. Drill-across, for example, implements queries containing (i.e., across) many fact tables. To drill through the bottom level of a data cube to its back-end relational tables, the drillthrough services need relational SQL procedures. OLAP operations can include calculating moving averages, growth values,

and interests, internal values of return, depreciation, currency conversions, and statistical services, as well as ranking the top N or bottom N items in lists and calculating moving averages, growth values, and interests, internal values of return, depreciation, currency conversions, and statistical services. The following are the three primary types of OLAP servers:

1. ROLAP- ROLAP is the abbreviation for Relational OLAP. It can store data using relational database management system (RDBMS) technology. Data and related aggregations are stored in an RDBMS in this case, and OLAP middleware is used to handle and explore data cubes. This design focuses on RDBMS backend optimization while also supporting extra tools and services, such as data cube navigation logic. The key benefit of ROLAP is scalability in managing enormous data volumes due to the usage of an RDBMS back end.
2. MOLAP- MOLAP is the abbreviation for Multidimensional OLAP. Tuples are used as the data storage unit. To manage data, the MOLAP provides a specialised n-dimensional array storage engine and OLAP middleware. As a result, OLAP queries are carried out by contacting linked multidimensional views directly (data cubes). This design emphasises the pre-calculation of transactional data into aggregations, resulting in quick query execution. MOLAP pre-calculates and saves aggregated measurements at each hierarchy level at load time, and then stores and indexes these data for instant retrieval.
3. HOLAP- HOLAP is an acronym for Hybrid OLAP. Some commercial OLAP servers are built on the HOLAP paradigm, which might result in a compromise between ROLAP's scalability and MOLAP's query implementation. In this method, the user selects which data to save in the MOLAP and which to save in the ROLAP. Low-level data, for example, is frequently saved in a relational database, whereas higher-level data, such as aggregations, is typically stored in a separate MOLAP.

Now we will look at how to use SQL to construct OLAP queries SQL included three enhancements to the GROUP BY statement to help with OLAP queries and data aggregation: the CUBE, ROLLUP, and GROUPING SETS operators.
On every subset of the given attribute types, the CUBE operator computes a union of GROUP BYs. Based on the source table, the output set depicts a multidimensional cube. Consider the SALES TABLE below.

PRODUCT	QUARTER	REGION	SALES
A	Q1	EAST	10
A	Q1	NORTH	20
A	Q2	EAST	20
A	Q2	SOUTH	50
A	Q3	WEST	20
A	Q4	NORTH	10
A	Q4	SOUTH	30
B	Q1	WEST	40
B	Q1	EAST	60
B	Q2	NORTH	20
B	Q2	NORTH	10
B	Q3	EAST	20
B	Q4	WEST	10
B	Q4	WEST	40

Figure 2: SALES TABLE.

SQL query:
SELECT QUARTER, REGION, SUM(SALES)
FROM SALESTABLE
GROUP BY CUBE (QUARTER, REGION)

Technically, this query computes the union of the SALESTABLE's 2^2= 4 groups, which are: (quarter,region), (quarter), (region), (), where () signifies an empty group list indicating the overall aggregate throughout the whole SALESTABLE. To put it another way, because quarter has four values and region has two, the final multiset will contain 4*2+4*1+1*2+1 or 15 tuples, as shown in Table 1. To signify the aggregate, NULL values have been added to the dimension columns Quarter and Region. If desired, these may be readily substituted with the more meaningful 'ALL'. More specifically, we may add the following two CASE clauses:

SELECT CASE WHEN grouping(QUARTER) = 1 THEN 'All' ELSE QUARTER END AS QUARTER, CASE WHEN grouping(REGION) = 1 THEN 'All' ELSE REGION END AS REGION, SUM(SALES)
FROM SALESTABLE
GROUP BY CUBE (QUARTER, REGION)

If a NULL value is created during the aggregate, the grouping() method returns 1; otherwise, it returns 0. This distinguishes between the created NULLs and the data's potential genuine NULLs. We won't include this in subsequent OLAP queries to avoid adding unnecessary complexity. Take note of the NULL value for Sales in the fifth row as well. This is an attribute combination that was not present in the original SALESTABLE since no items were allegedly sold in Europe in Q3. Other SQL aggregator functions, such as MIN(), MAX(), COUNT(), and AVG(), can be used in the SELECT query in addition to SUM().

QUARTER	REGION	SALES
Q1	Europe	50
Q1	America	80
Q2	Europe	40
Q2	America	60
Q3	Europe	NULL
Q3	America	40
Q4	Europe	20
Q4	America	80
Q1	NULL	130
Q2	NULL	100
Q3	NULL	40
Q4	NULL	90
NULL	Europe	110
NULL	America	250
NULL	NULL	360

Figure 3: Result from SQL query with Cube operator.

The ROLLUP operation computes the union on each prefix of the list of supplied attribute types, starting with the most detailed and working its way up to the grand total. It is especially handy for creating reports that include both subtotals and totals. The primary distinction between the ROLLUP and CUBE operators is that the former generates a result set containing aggregates for a hierarchy of values of the specified attribute types, whereas the latter produces a result set containing aggregates for all combinations of values of the selected attribute types. As a result, the sequence in which the attribute types are given is critical for the ROLLUP operator but not for the CUBE operator. Consider the following query:

SELECT QUARTER, REGION, SUM(SALES)
FROM SALESTABLE
GROUP BY ROLLUP (QUARTER, REGION)

This query returns the union of three groupings: (quarter,region), (quarter, (), where () denotes the whole aggregate once more. Table 2 displays the generated multiset, which has 4*2+4+1 or 13 rows. The region dimension is rolled up first, followed by the quarter dimension. When comparing the results of the CUBE operator in Table 1, notice the two rows that have been left out.

QUARTER	REGION	SALES
Q1	Europe	50
Q1	America	80
Q2	Europe	40
Q2	America	60
Q3	Europe	NULL
Q3	America	40
Q4	Europe	20
Q4	America	80
Q1	NULL	130
Q2	NULL	100
Q3	NULL	40
Q4	NULL	90
NULL	NULL	360

Figure 4: Result from SQL query with ROLLUP operator.

Whereas the preceding example employed the GROUP BY ROLLUP construct to two fully independent dimensions, it may also be applied to attribute types that reflect multiple aggregation levels (and consequently varying degrees of information) along the same dimension. Assume the SALESTABLE tuples provided more specific sales data at the city level and the database comprised three location-related columns: City, Country, and Region. We might then construct the following ROLLUP query, which would provide sales totals for each city, nation, region, and overall total:

SELECT REGION, COUNTRY, CITY, SUM(SALES)
FROM SALESTABLE
GROUP BY ROLLUP (REGION, COUNTRY, CITY)

Note that in that situation the SALESTABLE would comprise the attribute types City, Country and Region in a single table. Because the three attribute types reflect various levels of information in the same dimension, they are transitively reliant on one another, demonstrating that the data warehouse data is denormalized.

The GROUPING SETS operator produces a result set that is equal to the result set produced by a

UNION ALL of numerous simple GROUP BY clauses. Consider the following scenario:
SELECT QUARTER, REGION, SUM(SALES)
FROM SALESTABLE
GROUP BY GROUPING SETS ((QUARTER), (REGION))
This query is equivalent to:
SELECT QUARTER, NULL, SUM(SALES)
FROM SALESTABLE
GROUP BY QUARTER
UNION ALL
SELECT NULL, REGION, SUM(SALES)
FROM SALESTABLE
GROUP BY REGION

QUARTER	REGION	SALES
Q1	NULL	130
Q2	NULL	100
Q3	NULL	40
Q4	NULL	90
NULL	Europe	110
NULL	America	250

Figure 5: Result from SQL query with GROUPING SETS operator.

A single SQL query can contain several CUBE, ROLLUP, and GROUPING SETS statements. Equivalent result sets can be generated using various combinations of CUBE, ROLLUP, and GROUPING SETS. Consider the following query:
SELECT QUARTER, REGION, SUM(SALES)
FROM SALESTABLE
GROUP BY CUBE (QUARTER, REGION)
This query is equivalent to:
SELECT QUARTER, REGION, SUM(SALES)
FROM SALESTABLE
GROUP BY GROUPING SETS ((QUARTER, REGION), (QUARTER), (REGION), ())
Likewise, the following query:

SELECT QUARTER, REGION, SUM(SALES)
FROM SALESTABLE
GROUP BY ROLLUP (QUARTER, REGION)
is identical to:

SELECT QUARTER, REGION, SUM(SALES)
FROM SALESTABLE
GROUP BY GROUPING SETS ((QUARTER, REGION), (QUARTER),())

Given the volume of data to be aggregated and retrieved, OLAP SQL queries may become quite time intensive. Turning some of these OLAP queries into materialised views is one method to improve performance. For example, a SQL query containing a CUBE operator may be used to precompute aggregations on a set of dimensions, the results of which can then be saved as a materialised view. A downside of view materialisation is that more effort is required to routinely update these materialised views, albeit it should be noted that most businesses are comfortable with a close to current version of the data, so that synchronisation may be done overnight or at defined time intervals.

OLTP

OLTP, or online transactional processing, allows for the real-time execution of a large number of database transactions by a large number of individuals, generally through the internet. A database transaction is a data modification, insertion, deletion, or query in a database. Many of the financial transactions we perform every day are driven by OLTP systems (and the database operations they enable), such as online banking and ATM transactions, e-commerce and in-store purchases, and hotel and airline bookings, to mention a few. In each of these instances, the database transaction serves as a record of the related financial transaction. Non-financial database exchanges, such as password changes and text messages, can also be driven by OLTP.In OLTP, the defining feature of any database transaction is its atomicity (or indivisibility)—a transaction either succeeds or fails as a whole (or is canceled). It cannot remain pending or in an intermediate state. OLTP systems, in general, do the following:

1. Process a huge number of relatively basic operations, such as data insertions, updates, and removals, as well as simple data searches (for example, a balance check at an ATM).
2. Allow many users to access the same data while maintaining data integrity: OLTP systems rely on concurrency algorithms to ensure that no two users can alter the same data at the same time and that all transactions are completed in the correct sequence. This stops consumers who use online reservation systems from reserving the same accommodation twice and protects joint bank account users from unexpected overdrafts.
3. Priority is given to extremely fast processing, with reaction times measured in milliseconds: The total number of transactions that may be processed per second is used to determine the efficacy of an OLTP system.Make available indexed data sets: These are employed for quick searching, retrieval, and querying.
4. Are available 24 hours a day, seven days a week: Again, because OLTP systems handle a large number of concurrent transactions, any data loss or outage can have substantial and costly consequences. A comprehensive data backup must be available at all times. OLTP systems need frequent regular backups as well as continuous incremental backups.

Consider a supermarket's point-of-sale (POS) system. You've chosen a bar of chocolate and are waiting in line to get it billed. The cashier scans the bar code on the chocolate bar. Following the scanning of the bar code, several background actions occur.

- the database is consulted;
- the pricing and product details are obtained and shown on the computer screen;
- the cashier enters the amount purchased;
- the programme computes the total, prepares the bill, and prints it. You depart after paying the cash.

Your purchase has now been recorded in the application's database. This was an OLTP (On-Line Transaction Processing) system intended to allow online transactions and query processing.In other words, the supermarket's POS was an OLTP system. They are intended to cover the majority of an organization's day-to-day activities, such as purchasing, inventory, manufacturing, payroll, accounting, and so on. OLTP systems are distinguished by a significant number of brief on-line transactions such as INSERT (a record of a customer's final purchase was recorded to the database), UPDATE (a product's price was raised from Rs10 to Rs10.5), and DELETE (a product has gone out of demand and therefore the store removes it from the shelf as well as from its database).OLTP systems are used to record transactional data in almost all businesses today (including airlines, mail-order, supermarkets, and banking). OLTP systems often store the data they collect in commercial relational databases. For example, a grocery shop's database has the following tables to record information about its transactions, goods, workers, inventory supplies, and so on: Like Transactions, ProductMaster, EmployeeDetails, InventorySupplies, Suppliers, and so on. The two primary advantages of OLTP are concurrency and atomicity. They work together to bring order to real-time online transactions, and they have the following capabilities:

- The capacity to manage numerous users at the same time is referred to as concurrency. When one user performs a transaction at the same time as another, the OLTP programme guarantees that the transactions do not overlap and that each transaction is recorded.
- Compliance with atomicity, consistency, isolation, and durability (ACID) enables OLTP programmes to manage transactions correctly. Atomicity particularly assures that all components of the transaction must function in order for the transaction to be completed.

These characteristics result in a number of advantages for firms, including the following:

- Business knowledge. OLTP transaction data may be utilised to improve company analytics and intelligence. This study, as well as any from an OLAP programme, can improve an organization's OLTP and business plans.
- Usability. OLTP systems are straightforward, dependable, and user-friendly, which attracts new clients.
- Speed. These systems carry out commercial transactions in a timely and dependable manner.

Despite their benefits, OLTP systems have issues and flaws if they are not properly built and managed. Among them are the following:

- Overload of information. When there is an abundance of data, it can be challenging for firms to analyse valuable business insights in a back-end OLAP database or data warehouse.
- Silos of data. Because each programme has its own allocated database, data becomes siloed and impossible to transfer between apps.
- Analyses are limited. OLTP applications are designed for short, uncomplicated database operations and are ineffective for more in-depth research.
- Issues affecting small and medium-sized enterprises. OLTP solutions are often designed for large organisations and may not be suitable for SMBs.

The Architecture of OLTP looks like this-
Knowing the fundamental properties of an OLTP system, such as atomicity, concurrency, and integrity, as well as avoiding excessive usage of clusters and indexes, is required while designing an OLTP system. The following aspects should be addressed while designing an OLTP system-

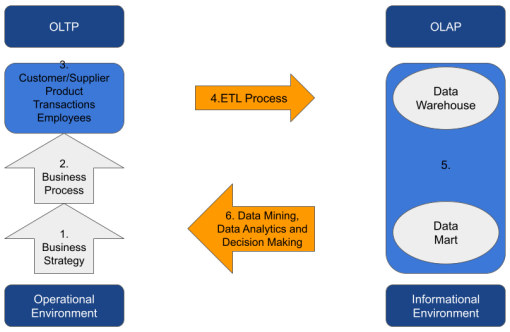

Figure 6: OLTP Architecture.

1. Rollback segments-Rollback segments are database components that record transactions in the event that one is rolled back. They are responsible for read consistency, database recovery, and transaction rollback.
2. Discrete transactions—they postpone data changes until the transaction is committed.
3. Clusters-A cluster is a structure that includes one or more tables that share columns. As a result, clustering tables in a database enhances the performance of join operations.
4. Data block size-The data block size should be a multiple of the operating system's block size, but it should not exceed the maximum limit. It eliminates needless I/O.
5. SQL queries should be optimised to leverage the database buffer cache to minimise consuming unnecessary resources.
6. Dynamic allocation refers to the allocation of space to tables and rollback segments on an ongoing basis.
7. Database partitioning-Partitioning the database improves the efficiency of sites with high transaction volumes while preserving security and availability.
8. Database tuning-OLTP may quickly and effectively improve performance.

OLAP vs OLAP

The Online Transaction Processing (OLTP) system feeds transactional data and offers assistance to the Online Analytical Processing (OLAP) system. The following are the significant distinctions between the two systems:

- In a three-tier design, OLTP systems served transaction-oriented applications, whereas OLAP systems analysed data stored in a data warehouse.
- OLTP manages an organization's everyday transactions, whereas OLAP provides a platform for business analysis that includes forecasting, analysis, planning, and budgeting.
- OLTP is defined by quick online transactions, whereas OLAP is characterised by vast amounts of data.
- OLTP makes use of a relational database to manage numerous concurrent transactions, whereas OLAP makes use of a data warehouse to aggregate multiple data sources in order to develop an integrated multidimensional database.
- OLTP systems are intended for use by front-line personnel such as cashiers, tellers, and so on, whereas OLAP systems are intended for use by business analysts and data scientists.

– OLTP systems change data, balance read and write operations, need limited storage, and require regular and concurrent backup. OLAP systems do not alter data, are read-intensive, need a large amount of storage, and may be backed up less regularly.

Conclusion

Here in this paper we have discussed about realtional model which is based on the mathematical concept of a relation, which is formally represented as a table. Relations are used to keep track of information about the things that will be represented in the database. A relation is shown as a two-dimensional table, with rows representing individual records and columns representing properties. Attributes can appear in any order, and the link stays unchanged and carries the same meaning and then we looked at the integrity constraints where we learned about the two integrity rules of relational model- entity integrity and referential integrity, which are limits or limitations that apply to all database instances and how to store objects in relational database where we discussed about mapping classes to relations in three different ways viz.map each class or subclass to a relation,map each subclass to a relation and map the hierarchy to a single relation, followed by advantages, disadvantages, and some applications of the same, and then finally we have had a deeper dig at the two major applications of RDBMS, OLAP and OLTP .

References

1. Ramakrishnan, Raghu, and Johannes Gehrke. Database Management Systems. 2002.
2. Connolly, Thomas, and Carolyn Begg. Database Systems. Addison Wesley, 2001.
3. Vaidya, Madhavi. RDBMS In-Depth Mastering SQL and PLSQL Concepts, Database Design, ACID Transactions, and Practice Real Implementation of RDBM. BPB Publications.2021.
4. Prasad, R. N., and Seema Acharya. Fundamentals of Business Analytics. John Wiley.2022.

⋆⋆⋆