DATABASE MANAGEMENT SYSTEMS (DBMS) SYNOPSIS

Unit-1: Database System Architecture and Data Models:

Data Abstraction, Data Independence, Data Definition Language (DDL), Data Manipulation Language (DML), Entity-relationship model, network model, relational and object oriented data models, integrity constraints, data manipulation operations.

Unit-2: Relational Query Languages and Relational Database Design:

Relational algebra, Tuple and domain relational calculus, SQL3, DDL and DML constructs, Open source and Commercial DBMS - MYSQL, ORACLE, DB2, SQL server.

Unit-3: Query Processing and Optimization and Storage Strategies:

Evaluation of relational algebra expressions, Query equivalence, Join strategies, Query optimization algorithms, Indices, B-trees, hashing.

Unit-4: Transaction Processing and Database Security:

Concurrency control, ACID property, Serializability of scheduling, Locking and timestamp based schedulers, Multi-version and optimistic Concurrency Control schemes, Database recovery Authentication, Authorization and access control.

Unit-5: SQL and PL/SQL Concepts:

Basics of SQL, DDL,DML,DCL, structure – creation, alteration, defining constraints – Primary key, foreign key, unique, not null, check, IN operator, aggregate functions, Built-in functions –numeric, date, string functions, set operations, sub-queries, correlated sub-queries, join, Exist, Any, All , view and its types., transaction control commands

----

Unit-1: Database System Architecture and Data Models:

Data Abstraction, Data Independence, Data Definition Language (DDL), Data Manipulation Language (DML), Entity-relationship model, network model, relational and object oriented data models, integrity constraints, data manipulation operations.

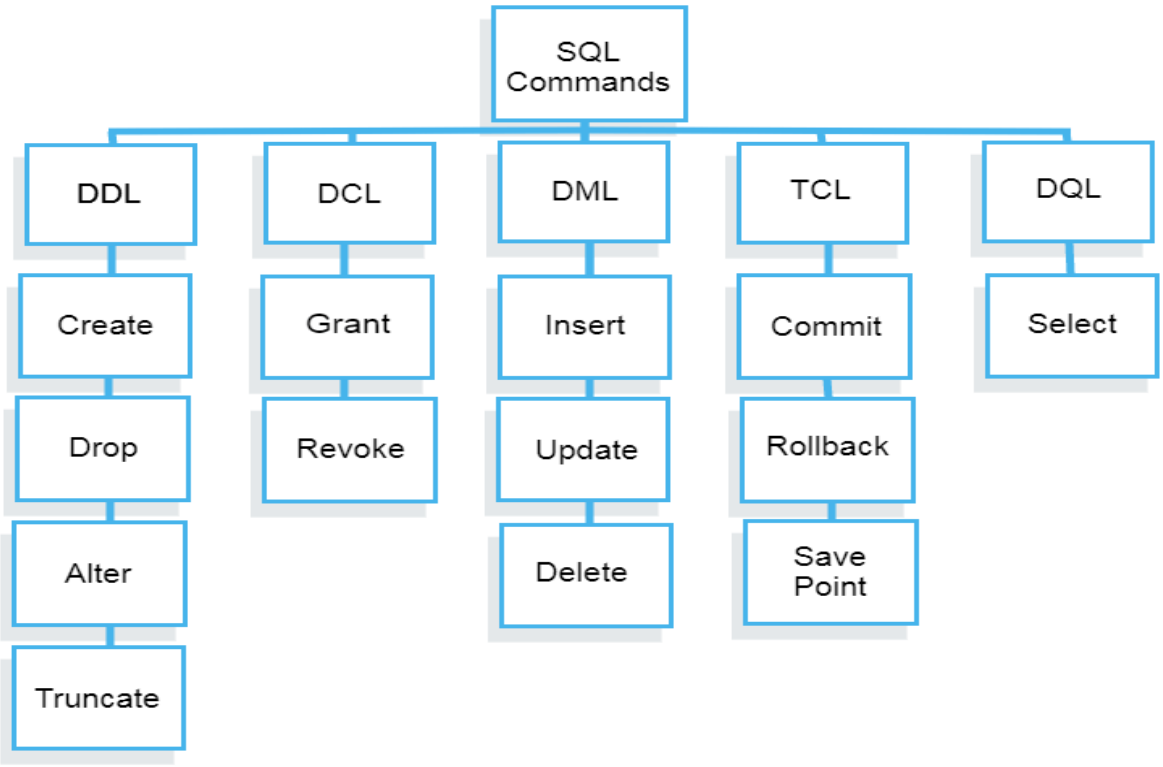
- **A database management system (or DBMS) is essentially nothing more than a computerized data-keeping system. Users of the system are given facilities to perform several kinds of operations on such a system for either manipulation of the data in the database or the management of the database structure itself.**
- **The DBMS manages incoming data, organizes it, and provides ways for the data to be modified or extracted by users or other programs. Some DBMS examples include MySQL, PostgreSQL, Microsoft Access, SQL Server, FileMaker, Oracle, RDBMS, dBASE, Clipper, and FoxPro.**
- **FORMS allow you to both add data to tables and view data that already exists. REPORTS present data from tables and also from QUERIES, which then search for and analyze data within these same tables.**

Database System Applications, Purpose of Database Systems, Railway Reservation System; Library Management System; Banking; Education Sector; Credit card exchanges; Social Media Sites; Pinterest, Quora, Facebook, Twitter, and Linked in; Broadcast communications; Account; Online Shopping; Human Resource Management; Manufacturing; Airline Reservation System; etc.

- **A representation of DBMS design- It helps to design, develop, implement, and maintain the database management system.**
- **One Tier Architecture (Single Tier Architecture)**
- **Two Tier Architecture**
- **Three Tier Architecture**
- **The following features are desirable in a database system used in transaction processing systems: Good data placement: The database should be designed to access patterns of data from many simultaneous users.**
- **Short transactions: Short transactions enables quick processing. This avoids concurrency and paces the systems.**
- **Real-time backup: Backup should be scheduled between low times of activity to prevent lag of the server.**
- **High normalization: This lowers redundant information to increase the speed and improve concurrency, this also improves backups.**

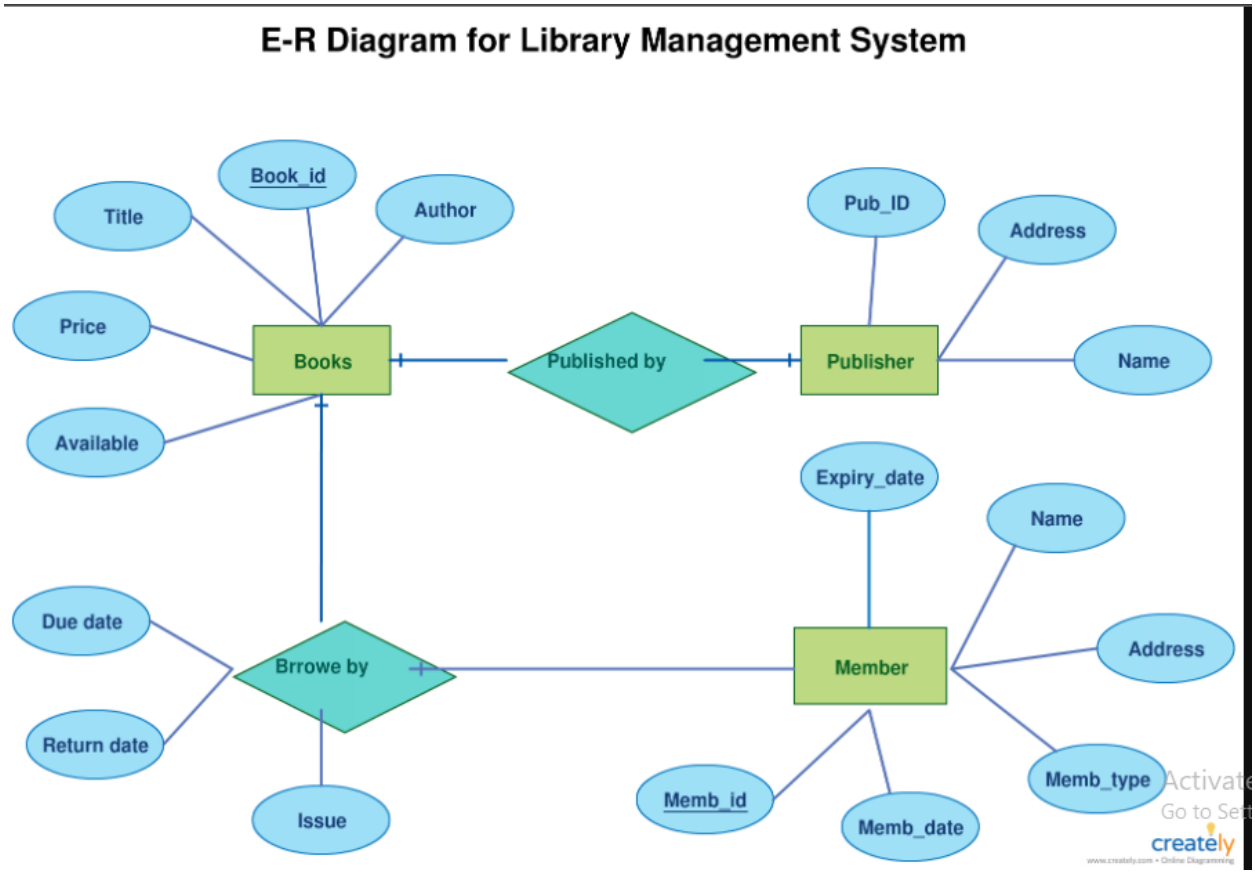
- **Archiving of historical data: Uncommonly used data are moved into other databases or backed up tables. This keeps tables small and also improves backup times.**
- **Hashing and indexing for quick searches.**
- **Data abstraction is the method of hiding the unimportant details that are present in the database from the end users to make the accessing of data easy and secure.**
- **The data abstraction in DBMS is implemented in 3 layers:**
    - **Physical or Internal Level**
    - **Logical or Conceptual Level**
    - **View or External Level**
- **Physical Level or Internal Level**
- **This is the layer of data abstraction where the raw data is physically stored as files.**
- **Example: When we access data we may get a single data or a table of data. Moreover, by the term "relational database" we visualize a table of rows and columns. But at a physical level, these tables are stored in hard drives which are located at a very secure data center.**
- **Logical Level or Conceptual Level**
- **After taking the raw data from the physical or internal level, the structure of the data is defined at the logical or conceptual level.**
- **Example: We have data of a few products like product id, product name, and manufacturing date, and we have another set of data of customers containing customer id, customer name, and customer address. Now, we need to frame this data in proper tables of products and customers. After that, we can even frame a join to show which product has been ordered by which customer.**
- **View Level or External Level: This is what an end-user gets to see. He/she does not get the entire database, but depending on the queries made from the front-end the user gets to see the data.**
- **EXAMPLE: let us say a customer wants to view the order history, he gets to see only the orders he had made in the past.**
- **Data independence is the ability to modify one level of a DBMS without affecting the next higher level's data structure or access methods. It's of three types, PHYSICAL, LOGICAL and VIEW.**

- **Physical Data Independence: This is defined as the ability to modify the physical schema of the database without the modification causing any changes in the logical/conceptual or view/external level.**
- **Examples of Physical Data Independence:**
- **Changing from one data structure to another.**
- **Making use of new storage technology, such as a hard drive or magnetic tapes**
- **Change the location of the database from one drive to another.**
- **Changing the database's file organization.**
- **Logical Data Independence**
- **Logical data independence is the ability to modify logical schema without causing any unwanted modifications to the external schema or the application programs to be rewritten.**
- **Logical data is database data, which means it stores information about how data is managed within the database. Logical data independence is a method that makes sure that if we make modifications to the table format, the data should not be affected.**
- **Examples of Logical Data Independence:**
- **Without rewriting current application scripts, you can add, modify, or delete a new attribute, entity, or relationship.**
- **To divide an existing record into two or more records.**
- **Merging two records into a single one.**
- **A DATA DEFINITION LANGUAGE (DDL) is a computer language used to create and modify the structure of database objects in a database. These database objects include views, schemas, tables, indexes, etc. aka DATA DESCRIPTION LANGUAGE**
- **It is considered to be a subset of SQL (STRUCTURED QUERY LANGUAGE).**
- **Common examples of DDL statements include CREATE, ALTER, and DROP.**
- **A DATA MANIPULATION LANGUAGE (DML) is a computer programming language used for adding (inserting), deleting, and modifying (updating) data in a database. A DML is often a sublanguage of a broader database language such as SQL, with the DML comprising some of the operators in the language.**



DDL- Data definition language, DCL= Data control language, DML = Data manipulation language, TCL- transaction control language, DQL – Data query language

- **ER model (Entity-Relationship model) It is a high-level data model. This model is used to define the data elements and relationship for a specified system.**
- **Four types of relationships exist in relational database design:**
- **ONE TO ONE - where one table record relates to another record in another table**
- **ONE TO MANY - where one table record relates to multiple records in another table**
- **MANY TO ONE - where more than one table record relates to another table record**
- **MANY TO MANY - where multiple records relate to more than one record in another table**



- **Extended or Enhanced ER model in DBMSSPECIALIZATION. The process of designing sub groupings within an entity set is called specialization. It is a top-down process.**
- **GENERALIZATION. It is the reverse process of specialization. It is a bottom-up approach. ...**
- **AGGREGATION. It is an abstraction in which relationship sets are treated as higher level entity sets and can participate in relationships.**

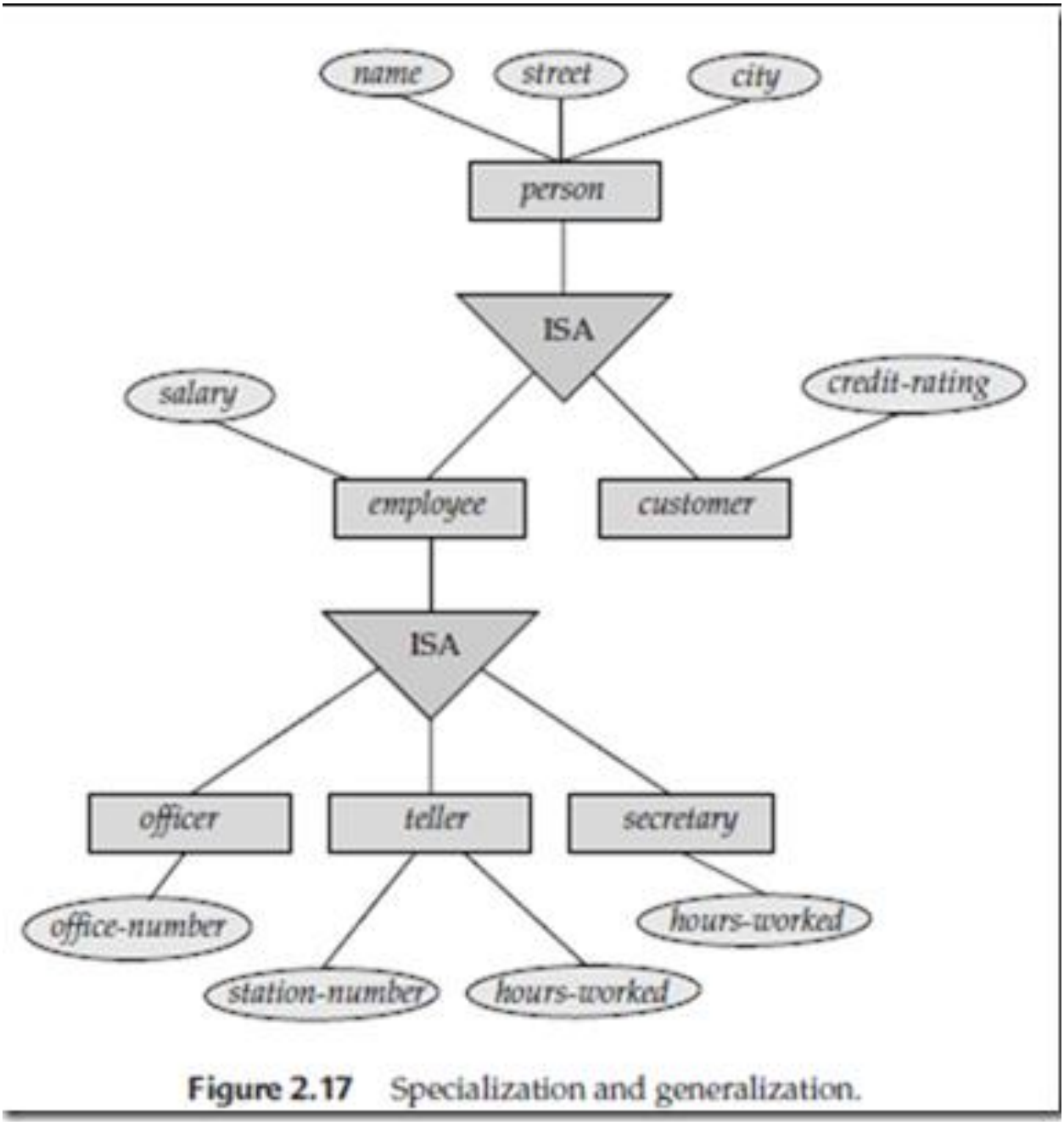


Figure 2.17 Specialization and generalization.

- **weak entity setsThe entity sets which do not have sufficient attributes to form a primary key are known as weak entity sets and the entity sets which have a primary key are known as strong entity sets. As the weak entities do not have any primary key, they cannot be identified on their own, so they depend on some other entity (known as owner entity). The weak entities have total participation constraint (existence dependency) in its identifying relationship with owner identity.**

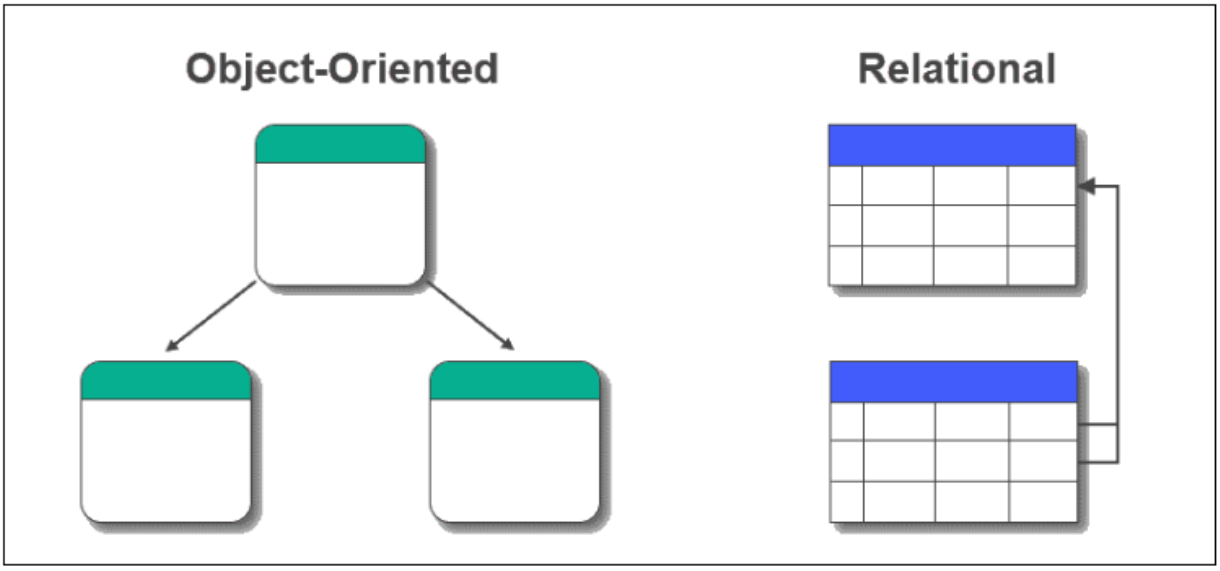
- **A network model of DBMS is a way of organizing data in which multiple many-to-many relationships between records are represented using a graph-like structure. In this model, records are represented as nodes in the graph, and relationships between records are**

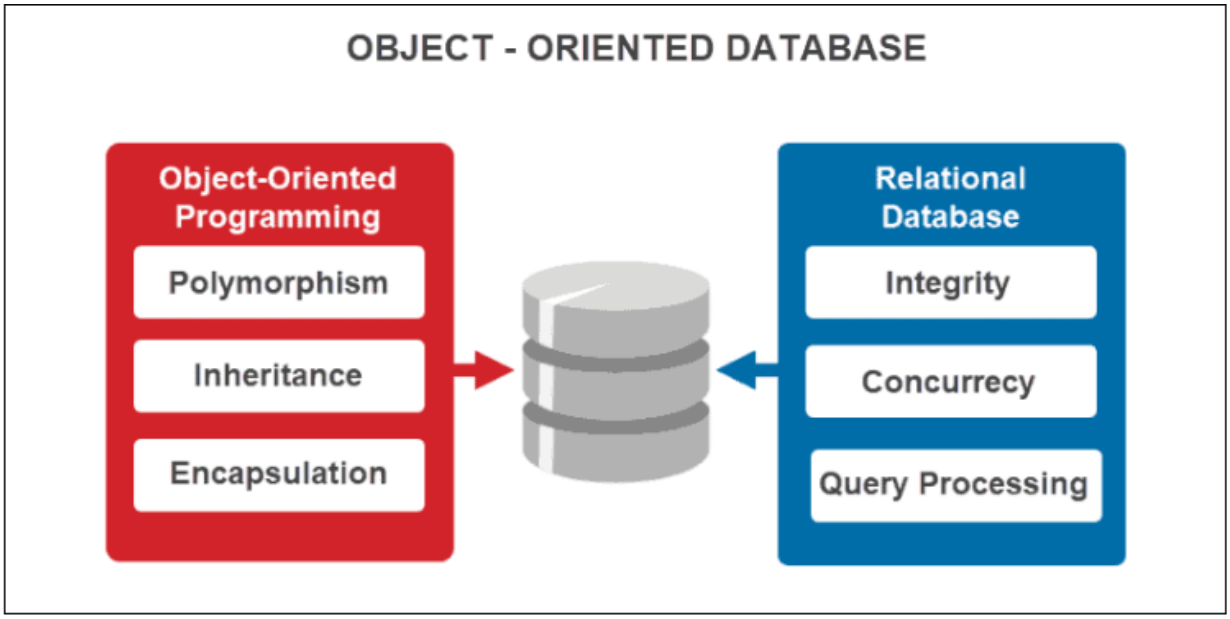
represented as edges connecting the nodes. A network model of DBMS can handle complex data structures and support multiple paths to access the same data. A network model of DBMS was formalized by the Database Task group in the 1960s https://www.geeksforgeeks.org/network-model-in-dbms/ and was one of the most popular models before the introduction of the relational model.

- **Some of the advantages of a network model of DBMS are:**
- **• It is simple and easy to design like the hierarchical model.**
- **• It can represent 1:1, 1:M, and M:N relationships among entities.**
- **• It can avoid data redundancy problems by supporting multiple links to the same record.**
- **• It can provide fast and efficient data access by using pointers or direct links.**
- **Some of the disadvantages of a network model of DBMS are:**
- **• It is difficult to maintain and modify the database schema as any change in the structure affects all the related records.**
- **• It requires complex programming and navigation logic to traverse the database graph.**
- **• It is not compatible with standard query languages like SQL.**
- **Some examples of network model of DBMS are:**
- **• IDMS (Integrated Database Management System) by CA Technologies**
- **• RDM (Relational Data Model) by Raima Inc.**
- **• TurboIMAGE by Hewlett-Packard**
- **What is semi-structured data in database?**
- **Semi-structured data refers to data that is not captured or formatted in conventional ways. Semi-structured data does not follow the format of a tabular data model or relational databases because it does not have a fixed schema.**
- **An object database is a database management system in which information is represented in the form of objects as used in object-oriented programming. Object databases are different from relational databases which are table-oriented.**
- **Object oriented data models are a way of organizing data in which both data and their relationships are contained in a single structure called an object. An object is an abstraction of a real-world entity that has attributes (properties) and methods (behaviors). Object oriented data models are used to represent complex and diverse data types, such as images, audio, video, and other multimedia. Object oriented data models also support features such as**

inheritance, polymorphism, and encapsulation, which allow for code reuse and data abstraction.

- **Some of the advantages of object oriented data models are: They can model real-world entities more naturally and accurately than relational or hierarchical models.**
- **They can handle complex data structures and support multiple paths to access the same data.**
- **They can avoid data redundancy and inconsistency by using pointers or direct links between objects.**
- **They can provide fast and efficient data access by using indexes and hashing techniques.**
- **Some of the disadvantages of object oriented data models are: • They are not well standardized and have different implementations and query languages.**
- **• They are difficult to maintain and modify as any change in the structure affects all the related objects.**
- **• They are not compatible with traditional query languages like SQL and require complex programming and navigation logic.**





Polymorphism is the ability of an object to take on many forms. The most common use of polymorphism in OOP occurs when a parent class reference is used to refer to a child class object. Object-oriented programming -ability to process objects differently depending on their class or data type.

- **Object-Oriented Database featuresMost contain the following features:**
- **Query Language - Language to find objects and retrieve data from the database.**
- **Transparent Persistence-Ability to use an object-oriented programming language for data manipulation.**
- **ACID Transactions- Atomicity, Consistency, integrity and durability transactions guarantee all transactions are complete without conflicting changes.**
- **Database Caching-Creates a partial replica of the database. Allows access to a database from program memory instead of a disk.**
- **Disaster recovery - in case of application or system failure.**
- **EncapsulationThe provision of an interface for a piece of software or hardware to allow or simplify access for the user.**
- **We can create a fully encapsulated class in Java by making all the data members of the class private. Now we can use setter and getter methods to set and get the data in it.**
- **Inheritance is an important pillar of OOP(Object-Oriented Programming). It is the mechanism in java by which one class is allowed to inherit the features(fields and methods) of another class.**

- **Super Class: The class whose features are inherited is known as superclass(or a base class or a parent class).**
- **Sub Class: The class that inherits the other class is known as a subclass(or a derived class, extended class, or child class). The subclass can add its own fields and methods in addition to the superclass fields and methods.**
- **Reusability: Inheritance supports the concept of "reusability",**
- **Some prominent ODBMSMongoDB-NoSQL- data objects are stored as documents**
- **Oracle, Microsoft SQL, IBM DB2 and PostGreSQL support objects**
- **LISP (List programming), Small Talk**
- **Applications of ODBMSEngineering design, CAD, STAAD, VEDA, 3D**
- **Architecture, building information modeling**
- **Astronomy**
- **Real time systems for manufacturing, traffic management, ATC etc.**
- **Telecommunications,**
- **Molecular sciences**
- **Hi performance, calculations and fast results**
- **Integrity constraintsIntegrity constraints are rules that help to maintain the accuracy and consistency of data in a database. They can be used to enforce business rules or to ensure that data is entered correctly. There are different types of integrity constraints in DBMS, such as:**
- **• Domain constraints: These specify the valid values for an attribute or a column in a table. For example, if the domain of an attribute is integer, then only integer values can be inserted into that column.**
- **• Key constraints: These ensure that each row in a table is uniquely identified by a primary key or a candidate key. A primary key is a minimal set of attributes that can uniquely identify a row, and a candidate key is any other set of attributes that can also serve as a primary key For example, if the primary key of a table is employee ID, then no two rows can have the same employee ID value**
- **Integrity constraints are important for maintaining the quality and reliability of data in a database. They can prevent data anomalies, such as insertion, deletion, and update anomalies, that can cause inconsistency and redundancy in the database. They can also**

facilitate data manipulation and query processing by providing information about the structure and semantics of the data.

- **• Entity integrity constraints: These ensure that the primary key value of a row is not null. This is because the primary key value is used to identify individual rows in a table and if the primary key has a null value, then we cannot identify those rows. For example, if the primary key of a table is employee ID, then every row must have a non-null employee ID value**
- **• Referential integrity constraints: These ensure that the foreign key value of a row in one table matches the primary key value of a row in another table or is null. A foreign key is an attribute or a set of attributes that refers to the primary key of another table. For example, if the foreign key of a table is department ID, then every department ID value must exist as a primary key value in the department table or be null**
- **Data manipulation operations are the actions that can be performed on data to transform, organize, or analyze it. Some of the common data manipulation operations are:**
- **• Aggregation: This operation involves combining or summarizing data into groups or categories based on some criteria. For example, you can aggregate sales data by month, product, or region to get the total sales for each group**
- **• Classification: This operation involves assigning labels or categories to data based on some rules or algorithms. For example, you can classify emails into spam or non-spam based on their content or sender**
- **• Mathematical formulas: This operation involves applying mathematical functions or calculations to data to derive new values or metrics. For example, you can calculate the average, median, standard deviation, or percentage change of a numerical data set**
- **Regression analysis: This operation involves finding the relationship between one or more independent variables and a dependent variable using a mathematical model. For example, you can use regression analysis to predict the sales of a product based on its price, advertising, and customer satisfaction**
- **• Row and column filtering: This operation involves selecting or excluding specific rows or columns of data based on some conditions. For example, you can filter data by date range, keyword, value range, or missing values.**
- **• String concatenation: This operation involves joining two or more strings of text together to form a new string. For example, you can concatenate the first name and last name of a person to get their full name**
- **Data manipulation operations are essential for preparing and processing data for various purposes, such as reporting, visualization, modeling, and decision making. Data manipulation can be done using various tools and languages, such as SQL, Excel, Python, R, and others.**

•

Unit-2: Relational Query Languages and Relational Database Design:

Relational algebra, Tuple and domain relational calculus, SQL3, DDL and DML constructs, Open source and Commercial DBMS - MYSQL, ORACLE, DB2, SQL server.

Relational Algebra is procedural query language, which takes Relation as input and generate relation as output. Relational algebra mainly provides theoretical foundation for relational databases and SQL.

- **Operators in Relational Algebra**
- **Projection (π)**
  Projection is used to project required column data from a relation.

By Default projection removes duplicate data.

Selection

-
- **(σ)**
  Selection is used to select required tuples of the relations.
- **for the above relation**
  σ (c>3)R
  will select the tuples which have c more than 3.
- **Note: selection operator only selects the required tuples but does not display them. For displaying, data projection operator is used.**
- **For the above selected tuples, to display we need to use projection also.**
- **Union (U)**
  Union operation in relational algebra is same as union operation in set theory, only constraint is for union of two relation both relation must have same set of Attributes.
- **Set Difference (-)**
  Set Difference in relational algebra is same set difference operation as in set theory with the constraint that both relation should have same set of attributes.
- **Rename (ρ)**
  Rename is a unary operation used for renaming attributes of a relation.
  ρ (a/b)R will rename the attribute 'b' of relation by 'a'.
- **Cross Product (X)**

- **Cross product between two relations let say A and B, so cross product between A X B will results all the attributes of A followed by each attribute of B. Each record of A will pairs with every record of B.**

Natural Join (⋈)
Natural join is a binary operator. Natural join between two or more relations will result set of all combination of tuples where they have equal common attribute. Conditional join works similar to natural join. In natural join, by default condition is equal between common attribute while in conditional join we can specify the any condition such as greater than, less than, not equalSelect(σ)The SELECT operation is used for selecting a subset of the tuples according to a given selection condition

Projection(π)The projection eliminates all attributes of the input relation but those mentioned in the projection list.

Union Operation(∪)UNION is symbolized by symbol. It includes all tuples that are in tables A or in B.

Set Difference(-)– Symbol denotes it. The result of A – B, is a relation which includes all tuples that are in A but not in B.

- **Intersection(∩) defines a relation consisting of a set of all tuple that are in both A and B.**
- **Cartesian Product(X) operation is helpful to merge columns from two relations.**
- **Inner join, includes only those tuples that satisfy the matching criteria.**
- **Theta Join(θ)The general case of JOIN operation is called a Theta join. It is denoted by symbol θ.**
- **EQUI Join-When a theta join uses only equivalence condition, it becomes a equi join.**
- **Natural Join(⋈) can only be performed if there is a common attribute (column) between the relations.**
- **Left Outer Join(=|X|)In the left outer join, operation allows keeping all tuple in the left relation.**
- **Right Outer join(|x|=)In the right outer join, operation allows keeping all tuple in the right relation.**
- **Full Outer Join(=|X|=)In a full outer join, all tuples from both relations are included in the result irrespective of the matching condition.**
- **Insertion.**
- **insert into course (course_id, title, dept_name, credits)**

- **values ('CS-437', 'Database Systems', 'Comp. Sci.', 4);**
- **Updates.**
- **update instructor**
- **set salary = salary * 1.05**
- **where salary < 70000;**
- **Join in DBMS is a binary operation which allows you to combine data from two or more tables in one single statement. The tables in DBMS are associated using the primary key and foreign keys.**
- **There are two types of joins in DBMS:**
- **Inner Joins: Theta, Natural, EQUI**
- **Outer Join: Left, Right, Full**
- **Inner Join is used to return rows from both tables which satisfy the given condition. It is the most widely used join operation and can be considered as a default join-type**
- **An Inner join or equijoin is a comparator-based join which uses equality comparisons in the join-predicate. However, if you use other comparison operators like ">" it can't be called equijoin.**
- **Inner Join further divided into three subtypes:**
- **Theta Join allows you to merge two tables based on the condition represented by theta. Theta joins work for all comparison operators. It is denoted by symbol θ. The general case of JOIN operation is called a Theta join.**
- **Syntax:**
- **$A \bowtie_\theta B$**
- **A ⋈ A.column 2 > B.column 2 (B)**
- **Consider the following tables.**
- **Table A** **Table B**

column 1	column 2	column 1	column2
1	10	1	10
2	20	2	30

A ⋈ A.column 2 < B.column 2 (B)

column 1 column 2

2 2

- **EQUI Join is done when a Theta join uses only the equivalence condition. EQUI join is the most difficult operation to implement efficiently in an RDBMS, and one reason why RDBMS have essential performance problems.**
- **For example:**
- **A ⋈ A.column 2 = B.column 2 (B)**
- **Natural Join (⋈)**
- **Natural Join does not utilize any of the comparison operators. In this type of join, the attributes should have the same name and domain. In Natural Join, there should be at least one common attribute between two relations.**
- **It performs selection forming equality on those attributes which appear in both relations and eliminates the duplicate attributes.**
- **An Outer Join doesn't require each record in the two join tables to have a matching record. In this type of join, the table retains each record even if no other matching record exists.**
- **Three types of Outer Joins are:**
- **Left Outer Join**
- **Right Outer Join**
- **Full Outer Join**
- **Tuple Relational Calculus (TRC) and Domain Relational Calculus (DRC) Tuple Relational Calculus (TRC) and Domain Relational Calculus (DRC) are two non-procedural query languages used in relational database management systems (RDBMS) to retrieve data from tables. TRC is used to select tuples from a relation based on a condition, while DRC is used to select individual values from a relation based on a condition.**
- **In TRC, the variables represent the tuples from specified relations, while in DRC, the variables represent the value drawn from a specified domain. TRC uses tuple variables to represent tuples, while DRC uses individual value variables.**
- **Here are some examples of how to use TRC and DRC: TRC: {T | EMPLOYEE (T) AND T.DEPT_ID = 10} selects all the tuples of employee names who work for Department 10.**

- **DRC: { | < EMPLOYEE > DEPT_ID = 10 } selects EMP_ID and EMP_NAME of employees who work for department 10.**

- **SQL3 is the name of the next version of the ANSI/ISO SQL standard, which is expected to introduce new features and enhancements to the SQL language. SQL3 is also known as SQL:2023, as it is planned to be published in 2023. Some of the main features of SQL3 are:**

- **Object-relational extensions: SQL3 will support object-oriented concepts, such as user-defined types, inheritance, polymorphism, and methods. This will allow SQL to handle complex and diverse data types, such as multimedia, spatial, and temporal data**

- **Persistent stored modules: SQL3 will support the creation and execution of stored procedures and functions, which are blocks of SQL code that can be invoked from within the database or from external applications. This will improve the modularity, reusability, and security of SQL code**

- **Triggers: SQL3 will support the definition and activation of triggers, which are special types of stored procedures that automatically execute when certain events occur in the database, such as insert, update, or delete operations. This will enable SQL to perform complex actions and enforce business rules based on data changeshttps://www.w3schools.com/sql/default.asp.**

- **Dynamic SQL: SQL3 will support the generation and execution of SQL statements at run time, based on user input or program logic. This will allow SQL to adapt to different situations and data sources dynamically**

- **Recursive queries: SQL3 will support the use of recursive queries, which are queries that refer to themselves or to other queries within the same statement. This will allow SQL to handle hierarchical and network data structures more efficiently**

- **SQL3 is not yet widely implemented by database systems, but some of its features are already supported by some popular databases, such as SQLite, MySQL, SQL Server, Oracle, PostgreSQL**

Unit-3: Query Processing and Optimization and Storage Strategies:

Evaluation of relational algebra expressions, Query equivalence, Join strategies, Query optimization algorithms, Indices, B-trees, hashing.

- **• DDL is used to create, modify, and delete the structure of database objects, such as tables, indexes, views, and constraints. DML is used to insert, update, delete, and query the data within those database objects**
- **DDL commands are CREATE, ALTER, DROP, TRUNCATE, etc. whereas DML commands are INSERT, UPDATE, DELETE, SELECT, etc**
- **DDL statements operate on the entire table whereas the DML statements operate on rows. For example, DROP TABLE deletes the whole table whereas DELETE FROM deletes specific rows from the table**
- **• DDL statements do not have a WHERE clause to filter the data whereas the DML statements use WHERE clause to filter the data. For example, ALTER TABLE changes the table definition without any condition whereas UPDATE SET changes the data values based on some condition**
- **• DDL statements are typically executed less frequently than DML statements as they affect the database schema. DML statements are frequently executed to manipulate and query data as they affect the database contents**
- **• DDL statements are typically executed by database administrators who have the privilege to create and modify the database schema. DML statements are typically executed by application developers or end-users who have the permission to access and manipulate the data**
- **Types of databasesHierarchical, network, relational, Object oriented, graph or document databases**
- **Object has fields, properties and methods**
- **Open source and Commercial DBMS - MYSQL, ORACLE, DB2, SQL server.**
- **Evaluation of relational algebra expressionsEvaluation of relational algebra expressions is the process of finding the most efficient way to execute a query that involves one or more operations on relations, such as selection, projection, join, union, etc. There are different methods and techniques for evaluating relational algebra expressions, such as:**
- **• Materialization: This method evaluates one operation at a time in a bottom-up manner and stores the intermediate results in temporary files. This method is simple and easy to implement, but it has a high cost of writing and reading data from disk**

- • **Pipelining: This method evaluates several operations simultaneously in a pipeline, where the output of one operation is passed directly to the next operation without storing it in a temporary file. This method reduces the disk I/O cost and improves the performance, but it requires more memory and synchronization**
- **Evaluation of relational algebra expressions-2Heuristic optimization: This method applies some rules or heuristics to transform a relational algebra expression into an equivalent one that has a lower cost. For example, some heuristics are: push selections down as far as possible, push projections down as far as possible, combine selections and projections into one operation, etc**
- • **Cost-based optimization: This method estimates the cost of different evaluation plans for a relational algebra expression and chooses the one with the lowest cost. The cost can be measured by various factors, such as the number of disk accesses, CPU time, memory usage, etc. To estimate the cost, this method uses statistics and information about the relations, such as the size, cardinality, distribution, indexes, etc**
- **Query equivalenceQuery equivalence and join strategies are two important concepts in query processing and optimization, which are the steps of transforming and executing a query in a database system.**
- **Query equivalence means that two queries produce the same result for any database instance, regardless of the syntax or the order of operations. For example, the queries SELECT * FROM R WHERE A = 10 and SELECT * FROM (SELECT * FROM R WHERE A = 10) are equivalent, as they both return the same set of records from relation R that have A equal to 10. Query equivalence can be used to rewrite a query into a simpler or more efficient form, using some rules or heuristics, such as pushing selections and projections down, combining operations, etc**
- **Join strategies are the methods of implementing the join operation, which combines records from two or more relations based on a common attribute. There are different types of join strategies, such as nested-loop join, single-loop join, hash join, merge join, etc. Each join strategy has its own advantages and disadvantages, depending on the size, distribution, and indexing of the relations involved. The choice of the best join strategy for a query depends on various factors, such as the cost of disk I/O, CPU time, memory usage, etc**
- **Query optimization algorithms are methods of finding the most efficient way to execute a query in a database system. Query optimization algorithms can be classified into two main categories: heuristic-based and cost-based**
- **Heuristic-based algorithms use some rules or guidelines to transform a query into an equivalent form that may be evaluated more efficiently. For example, some heuristics are: push selections and projections down as far as possible, combine operations, use indexes, etc.**

Heuristic-based algorithms are simple and fast, but they may not always find the optimal solution.

- **Cost-based algorithms use statistics and information about the database to estimate the cost of different execution plans for a query and choose the one with the lowest cost. The cost can be measured by various factors, such as the number of disk accesses, CPU time, memory usage, etc. Cost-based algorithms are more accurate and flexible, but they require more computation and maintenance.**
- **• Exhaustive search: This algorithm generates all possible execution plans for a query and selects the one with the lowest cost. This algorithm is guaranteed to find the optimal solution, but it is very expensive and impractical for large queries**
- **• Dynamic programming: This algorithm uses a bottom-up approach to divide a query into subqueries and find the optimal plan for each subquery. This algorithm reduces the search space by reusing the optimal plans for subqueries, but it still has a high complexity and memory requirement**
- **• Greedy algorithm: This algorithm uses a top-down approach to select one operation at a time for a query based on some heuristic or cost function. This algorithm is fast and simple, but it may get stuck in a local optimum and miss the global optimum**
- **• Genetic algorithm: This algorithm uses an evolutionary approach to generate and improve a population of execution plans for a query based on some fitness function. This algorithm can explore a large and diverse search space, but it may take a long time to converge to a good solution**
- **Indices in DBMS are data structures that help to improve the performance of a database by reducing the number of disk accesses required to process a query. Indices are created on one or more columns of a table and store the values of those columns in a sorted order along with pointers to the corresponding records in the table. Indices can speed up data retrieval, especially for queries that involve selection, projection, or join operations**
- **There are different types of indices in DBMS, such as:**
- **• Primary index: This is an index created on the primary key of a table, which is a unique identifier for each record. A primary index can be dense or sparse, depending on whether it contains an entry for every record or only for some records in the table**
- **• Secondary index: This is an index created on a non-primary key column or a combination of columns that may not be unique for each record. A secondary index is always dense and can be used to access records based on other attributes than the primary key**
- **Clustered index: This is an index that determines the physical order of records in a table. A table can have only one clustered index, which can be based on the primary key or any other**

column. A clustered index can improve the performance of range queries and sorting operations

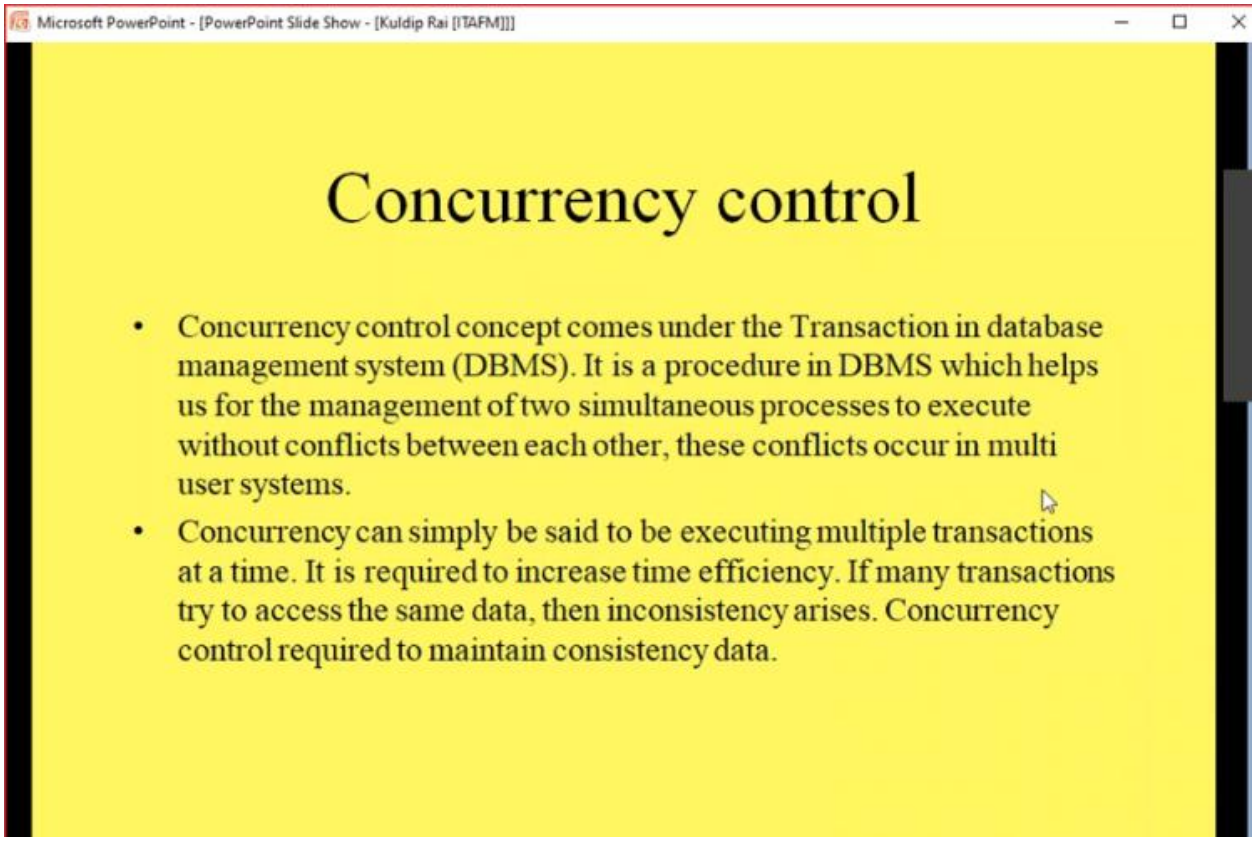
- **• Non-clustered index: This is an index that does not affect the physical order of records in a table. A table can have multiple non-clustered indices, which can be based on any column or combination of columns. A non-clustered index can improve the performance of queries that involve specific values or conditionsB-trees are a type of data structure that can store and manipulate large amounts of data efficiently. They are often used in database systems, file systems, and other applications that require fast and reliable access to disk-based data. B-trees have some advantages over other data structures, such as:**
- **• They can handle variable-length records and keys, unlike arrays or hash tables.**
- **• They can support multiple operations, such as search, insert, delete, and range queries, in logarithmic time, unlike linked lists or binary search trees.**
- **• They can reduce the number of disk accesses by storing multiple keys and pointers in each node, unlike binary search trees that store only one key and two pointers per node.**
- **They can maintain balance and order by splitting and merging nodes when necessary, unlike binary search trees that may become unbalanced or skewed.**
- **B-trees are defined by a parameter called the order or the degree, which is the maximum number of children that a node can have. A B-tree of order m has the following properties:**
- **• Every node has at most m children and at most m-1 keys.**
- **• Every node except the root has at least m/2 children and at least m/2 - 1 keys.**
- **• The root has at least 2 children (unless it is a leaf) and at least 1 key.**
- **• All the leaf nodes are at the same level and have no children.**
- **• All the keys in a node are sorted in ascending order, and all the keys in a subtree are between the keys that separate it from other subtrees.**
- **Hashing in DBMS is a technique to directly search the location of desired data on the disk without using index structure. Hashing method is used to index and retrieve items in a database as it is faster to search that specific item using the shorter hashed key instead of using its original value**
- **Hashing in DBMS works by applying a hash function to the search key of a record, which is usually an attribute or a combination of attributes that can uniquely identify the record. The hash function maps the search key to a numeric value, called the hash value or the hash index, which indicates the address of the data block where the record is stored. The data block where the records are stored is also known as the data bucket or the data page**

- **There are different types of hashing techniques in DBMS, such as static hashing and dynamic hashing.**
- **Static hashing uses a fixed number of buckets and assigns records to them based on their hash values. Static hashing is simple and fast, but it may suffer from problems such as overflow, underflow, and collision.**
- **Dynamic hashing uses a variable number of buckets and adjusts them according to the changes in the database size. Dynamic hashing is more flexible and scalable, but it may incur more overhead and complexity**
- **Some of the advantages of hashing in DBMS are:**
- **• It can provide direct and fast access to records without scanning through intermediate levels of index structures.**
- **• It can handle variable-length records and keys, unlike arrays or hash tables.**
- **• It can support multiple operations, such as search, insert, delete, and range queries, in logarithmic time or constant time, depending on the hashing technique**
- **There are different types of hashing techniques in DBMS, such as static hashing and dynamic hashing.**
- **Static hashing uses a fixed number of buckets and assigns records to them based on their hash values. Static hashing is simple and fast, but it may suffer from problems such as overflow, underflow, and collision.**
- **Dynamic hashing uses a variable number of buckets and adjusts them according to the changes in the database size. Dynamic hashing is more flexible and scalable, but it may incur more overhead and complexity**
- **Some of the advantages of hashing in DBMS are:**
- **• It can provide direct and fast access to records without scanning through intermediate levels of index structures.**
- **• It can handle variable-length records and keys, unlike arrays or hash tables.**
- **• It can support multiple operations, such as search, insert, delete, and range queries, in logarithmic time or constant time, depending on the hashing technique**
- **• It may cause data redundancy and inconsistency by storing multiple copies of the same record in different buckets due to collision or overflow.**
- **• It may require complex programming and navigation logic to handle collision resolution and bucket reorganization.**

- **• It may not be compatible with traditional query languages like SQL and require special syntax or functions**
- **Transaction managementA transaction is a set of logically related operations.**
- **Lets say your account is A and your friend's account is B, you are transferring 10000 from A to B, the steps of the transaction are:**
- **1. R(A);**
- **2. A = A - 10000;**
- **3. W(A);**
- **4. R(B);**
- **5. B = B + 10000;**
- **6. W(B);**
- **In the above transaction R refers to the Read operation and W refers to the write operation.**

Unit-4: Transaction Processing and Database Security:

Concurrency control, ACID property, Serializability of scheduling, Locking and timestamp based schedulers, Multi-version and optimistic Concurrency Control schemes, Database recovery Authentication, Authorization and access control.



- **LockingLock guaranties exclusive use of data items to a current transaction. It first accesses the data items by acquiring a lock, after completion of the transaction it releases the lock.**
- **The types of locks are as follows –**
- **Shared Lock [Transaction can read only the data item values]**
- **Exclusive Lock [Used for both read and write data item values]**
- **Time stamp is a unique identifier created by DBMS that indicates relative starting time of a transaction. Whatever transaction we are doing it stores the starting time of the transaction and denotes a specific time.**
- **This can be generated using a system clock or logical counter. This can be started whenever a transaction is started. Here, the logical counter is incremented after a new timestamp has been assigned.**

- **Transaction processing system featuresPerformance: Fast performance with a rapid response time (number of transactions/second)**
- **Continuous availability**
- **Data integrity- without corrupting data. Multiple users must be prevented from attempting to change the same piece of data at the same time, for example two operators cannot sell the same seat on an airplane. (a record under editing gets locked)**
- **Ease of use- protect users from data-entry errors as much as possible, and allow them to easily correct their errors.**
- **Modular growth at incremental costs, rather than requiring a complete replacement. It should be possible to add, replace, or update hardware and software components without shutting down the system.**
- **Databases and files- A database is an organized collection of data. Databases offer fast retrieval times for non-structured requests as in a typical transaction processing application.**
- **ACID in DBMS stands for Atomicity, Consistency, Isolation, and Durability. These are four properties that ensure reliable and consistent processing of transactions in a database system. Transactions are sequences of database operations that are executed as a single unit of work, and the ACID properties ensure that transactions are processed correctly even in the event of failures or concurrency**
- **• Atomicity means that a transaction is either executed completely or not at all. If any part of the transaction fails, the entire transaction is aborted and the database is restored to its previous state before the transaction started**
- **Consistency means that a transaction preserves the integrity and validity of the data in the database. A transaction must follow the rules and constraints defined by the database schema, such as primary keys, foreign keys, data types, etc. A transaction must also ensure that the database state is consistent before and after the transaction**
- **• Isolation means that a transaction is executed independently from other concurrent transactions. A transaction must not interfere with or be affected by other transactions that are running at the same time. A transaction must also ensure that its intermediate results are not visible to other transactions until it is committed**
- **• Durability means that a transaction's effects are permanent and persistent in the database. Once a transaction is committed, its changes to the data are written to disk and cannot be lost or undone even if a system failure occurs**

- **ACID properties are important for maintaining the quality and reliability of data in a database system. They can prevent data anomalies, such as lost updates, dirty reads, unrepeatable reads, and phantom reads, that can cause inconsistency and corruption in the database**
- **They can also facilitate data manipulation and query processing by providing information about the structure and semantics of the datahtt**
- **Serializability of scheduling is a concept that is used to ensure the correctness and consistency of transactions in a database system. Serializability of scheduling means that a non-serial schedule, which is a schedule that allows transactions to execute concurrently with interleaved operations, is equivalent to a serial schedule, which is a schedule that executes transactions one after another without any overlap**
- **Serializability of scheduling can be checked by different methods, such as conflict serializability and view serializability.**
- **Conflict serializability means that a non-serial schedule is equivalent to a serial schedule if it has the same order of conflicting operations, which are operations that access the same data item and at least one of them is a write operation**
- **View serializability means that a non-serial schedule is equivalent to a serial schedule if it has the same initial read, final write, and read-write dependencies for each data item**
- **Serializability of scheduling is important for maintaining the quality and reliability of data in a database system. It can prevent data anomalies, such as lost updates, dirty reads, unrepeatable reads, and phantom reads, that can cause inconsistency and corruption in the database. It can also facilitate data manipulation and query processing by providing information about the structure and semantics of the data.**
- **Multi-version and optimistic concurrency control schemes are two techniques to improve the performance and consistency of transactions in a database system. They both allow transactions to execute concurrently without locking the data, but they use different methods to detect and resolve conflicts.**
- **Multi-version concurrency control (MVCC) is a technique that maintains multiple versions of each data item and assigns them timestamps to indicate their validity. When a transaction reads a data item, it gets the latest version that is compatible with its timestamp. When a transaction writes a data item, it creates a new version with a higher timestamp. Conflicts are detected when two transactions try to write the same data item, and the one with the lower timestamp is aborted**
- **Optimistic concurrency control (OCC) is a technique that assumes that conflicts are rare and validates transactions at the end of their execution. When a transaction reads or writes a data item, it records its version number in a local buffer. When a transaction commits, it checks**

whether any of the data items it accessed have been modified by other transactions. If so, it aborts and restarts. Otherwise, it writes its changes to the database

- **Both MVCC and OCC can reduce the overhead of locking and increase the concurrency level of transactions, but they have different trade-offs. MVCC requires more storage space and maintenance for keeping multiple versions of data items, but it can avoid aborting read-only transactions. OCC requires less storage space and maintenance, but it can incur more aborts and restarts for conflicting**
- **Database recovery, authentication, authorization, and access control are some of the important topics in database security. Database security is the process of protecting the data and the database system from unauthorized access, modification, or destruction. Here is a brief overview of each topic:**
- **• Database recovery is the process of restoring the database to a consistent state after a failure or an error. Database recovery can involve techniques such as backup and restore, transaction logging and rollback, checkpointing, shadow paging, etc**
- **• Authentication is the process of verifying the identity of a user or a system that wants to access the database. Authentication can involve methods such as passwords, tokens, biometrics, certificates, etc**
- **• Authorization is the process of granting or denying permissions to a user or a system to perform certain operations on the database. Authorization can involve mechanisms such as access control lists, roles, privileges, views, etc**
- **• Access control is the process of enforcing the authorization rules and policies on the database. Access control can involve models such as discretionary access control, mandatory access control, role-based access control, etc**

Unit-5: SQL and PL/SQL Concepts:

Basics of SQL, DDL,DML,DCL, structure – creation, alteration, defining constraints – Primary key, foreign key, unique, not null, check, IN operator, aggregate functions, Built-in functions –numeric, date, string functions, set operations, sub-queries, correlated sub-queries, join, Exist, Any, All , view and its types., transaction control commands

- **Database languages (The Sub-languages of SQL) Data Definition Language (DDL)**
- **Data Manipulation language (DML)**
- **Data Query Language (DQL)**
- **Data Control Language (DCL)**
- **Transactional Control Statements.**
- **Session Control Statements.**
- **Database Administration Statements.**
- **Prepared/Procedural/Embedded SQL Statements.**
- **Modification of the DatabaseDeletion. Delete all tuples in the instructor relation pertaining to instructors in the Finance department.**
- **delete from instructor**
- **where dept_name= 'Finance';**
- **Delete all instructors with a salary between $13,000 and $15,000.+**
- **delete from instructor**
- **where salary between 13000 and 15000**
- **Constraints in SQLNOT NULL - Ensures that a column cannot have a NULL value**
- **UNIQUE - Ensures that all values in a column are different**
- **PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table**
- **FOREIGN KEY - Prevents actions that would destroy links between tables**
- **CHECK - Ensures that the values in a column satisfies a specific condition**
- **DEFAULT - Sets a default value for a column if no value is specified**
- **CREATE INDEX - Used to create and retrieve data from the database very quickly**

- **The IN operator is a logical operator in database management systems that allows you to specify multiple values in a WHERE clause. It is a shorthand for multiple OR conditions.**
- **Here's an example of how to use the IN operator in SQL:**
- **SELECT * FROM Customers WHERE Country IN ('Germany', 'France', 'UK'); This query returns all customers from Germany, France, or UK.**
- **You can also use the NOT keyword in front of the IN operator to return all records that are NOT any of the values in the list. Here's an example:**
- **SELECT * FROM Customers WHERE Country NOT IN ('Germany', 'France', 'UK'); This query returns all customers that are NOT from Germany, France, or UK.**
- **You can also use the IN operator with a subquery in the WHERE clause. With a subquery, you can return all records from the main query that are present in the result of the subquery. Here's an example:**
- **SELECT * FROM Customers WHERE CustomerID IN (SELECT CustomerID FROM Orders); This query returns all customers that have an order in the Orders table.**
- **Aggregate functions in DBMS are used to perform calculations on sets of data. They take a set of values as input and return a single value as output. These functions are often used to generate summary statistics on large datasets, such as the average, minimum, maximum, and sum of a set of values [1].**
- **Some examples of aggregate functions in DBMS are:**
- **COUNT: This function returns the number of rows in a table or the number of rows that satisfy a condition. For example, SELECT COUNT(*) FROM Customers returns the total number of customers in the table.**
- **SUM: This function returns the sum of all values in a column or the sum of values that satisfy a condition. For example, SELECT SUM(Salary) FROM Employees returns the total salary of all employees in the table.**
- **AVG: This function returns the average of all values in a column or the average of values that satisfy a condition. For example, SELECT AVG(Age) FROM Students returns the average age of all students in the table.**
- **MIN: This function returns the minimum value in a column or the minimum value among values that satisfy a condition. For example, SELECT MIN(Price) FROM Products returns the lowest price of any product in the table.**

- **MAX: This function returns the maximum value in a column or the maximum value among values that satisfy a condition. For example, SELECT MAX(Quantity) FROM Orders returns the highest quantity of any order in the table.**
- **You can also use aggregate functions with other clauses such as GROUP BY, HAVING, and ORDER BY to further manipulate and analyze data. For example, you can use SELECT Product, SUM(Quantity) FROM Orders GROUP BY Product ORDER BY SUM(Quantity) DESC to get the total quantity ordered for each product and sort them in descending order.**
- **Creation, alteration, and defining constraints in DBMS are some of the tasks that can be performed using SQL commands.**
- **• Creation in DBMS means creating new database objects, such as tables, views, indexes, etc. This can be done using the CREATE statement in SQL. For example, to create a table named student with four columns, you can use the following SQL command:**
- **CREATE TABLE student (**
- **StudentID INT,**
- **StudentName VARCHAR(20),**
- **StudentAge INT,**
- **StudentAddress VARCHAR(50));**
- **• Alteration in DBMS means modifying the structure or definition of existing database objects, such as adding, deleting, or renaming columns, changing data types, etc. This can be done using the ALTER statement in SQL. For example, to add a new column named StudentEmail to the student table, you can use the following SQL command:**
- **ALTER TABLE student ADD StudentEmail VARCHAR(30);**
- **Numeric functions: These functions operate on numeric data types and return a numeric value. Some examples of numeric functions are SUM, AVG, MAX, MIN, COUNT, etc.**
- **Date and time functions: These functions operate on date and time data types and return a date or time value. Some examples of date and time functions are NOW, CURDATE, CURTIME, YEAR, MONTH, DAY, etc.**
- **String functions: These functions operate on string data types and return a string value. Some examples of string functions are CONCAT, SUBSTRING, LENGTH, LOWER, UPPER, etc .**
- **Set operations: These operations are used to combine the results of two or more SELECT statements into a single result set. The most commonly used set operations are UNION, UNION ALL, INTERSECT, and EXCEPT .**

- **JOIN operator: The JOIN operator is used to combine two or more tables based on a specified common field between them [1][2]. There are different types of JOINs, such as INNER JOIN, LEFT JOIN, RIGHT JOIN, and FULL JOIN [2]. For example, you can use SELECT * FROM Customers INNER JOIN Orders ON Customers.CustomerID = Orders.CustomerID to combine the Customers and Orders tables based on the CustomerID field .**

- **Exist operator: The EXISTS operator is used to check if a subquery returns any rows. If the subquery returns at least one row, the EXISTS operator returns true [3][4]. For example, you can use SELECT * FROM Customers WHERE EXISTS (SELECT * FROM Orders WHERE Customers.CustomerID = Orders.CustomerID) to return all customers who have placed an order .**

- **Any operator: The ANY operator is used to compare a value to each value in a list or results from a query and evaluates to true if the result of an inner query contains at least one row. ANY must be preceded by comparison operators [3]. For example, you can use SELECT * FROM Products WHERE ProductID = ANY (SELECT ProductID FROM OrderDetails WHERE Quantity = 9) to return all products that have an order with a quantity of 9.**

- **All operator: The ALL operator is used to compare a value to every value in another value set or result from a subquery. The ALL operator returns TRUE if all of the subqueries values meet the condition. The ALL must be preceded by comparison operators and evaluates true if all of the subqueries values meet the condition. ALL is used with SELECT, WHERE, HAVING statement [3]. For example, you can use SELECT * FROM Products WHERE ProductID = ALL (SELECT ProductId FROM OrderDetails WHERE Quantity = 6 OR Quantity = 2) to return all products that have orders with quantities of 6 or 2**

- **ALTER TABLE student ADD StudentEmail VARCHAR(30);**

- **• Defining constraints in DBMS means specifying rules or restrictions that apply to the data in a table or a column. Constraints are used to ensure the accuracy and consistency of the data and prevent invalid or duplicate values. Some of the common constraints in SQL are NOT NULL, UNIQUE, PRIMARY KEY, FOREIGN KEY, CHECK, etc. Constraints can be defined when the table is created using the CREATE statement or after the table is created using the ALTER statement. For example, to define a primary key constraint on the StudentID column of the student table, you can use either of these SQL commands:**

- **-- Using CREATE statement**

- **CREATE TABLE student (**

- **StudentID INT PRIMARY KEY,**

- **StudentName VARCHAR(20),**

- **StudentAge INT,**

- **StudentAddress VARCHAR(50),**
- **StudentEmail VARCHAR(30));**
- **-- Using ALTER statement**
- **ALTER TABLE student ADD PRIMARY KEY (StudentID);**
- **Views in SQL are considered as a virtual table. A view also contains rows and columns.**
- **To create the view, we can select the fields from one or more tables present in the database.**
- **A view can either have specific rows based on certain condition or all the rows of a table.**
- **There are 2 types of Views in SQL: Simple views can only contain a single base table. Complex views can be constructed on more than one base table. To create the view, we can select the fields from one or more tables present in the database. A view can either have specific rows based on certain condition or all the rows of a table.**
- **Syntax:**
- **CREATE VIEW view_name AS.**
- **SELECT column1, column2.....**
- **FROM table_name.**
- **WHERE condition;+ A nested query is a query that has another query embedded within it. The embedded query is called a subquery. A subquery typically appears within the WHERE clause of a query. It can sometimes appear in the FROM clause or HAVING clause.**
- **The SELECT DISTINCT statement is used to return only distinct (different) values.**
- **Inside a table, a column often contains many duplicate values; and sometimes you only want to list the different (distinct) values.**
- **SELECT DISTINCT column1, column2, ...**
- **FROM table_name;**
- **Complex query exampleBack up proceduresIt involves the backup, journal, checkpoint, and recovery manager:**
- **JOURNAL: A journal maintains an audit trail of transactions and database changes. Transaction logs and Database change logs are used, a transaction log records all the essential data for each transactions, including data values, time of transaction and terminal number. A database change log contains before and after copies of records that have been modified by transactions.**

- **CHECKPOINT: The purpose of checkpointing is to provide a snapshot of the data within the database. A checkpoint, in general, is any identifier or other reference that identifies the state of the database at a point in time. Modifications to database pages are performed in memory and are not necessarily written to disk after every update. Therefore, periodically, the database system must perform a checkpoint to write these updates which are held in-memory to the storage disk. Writing these updates to storage disk creates a point in time in which the database system can apply changes contained in a transaction log during recovery after an unexpected shut down or crash of the database system.**
- **RECOVERY MANAGERA recovery manager is a program which restores the database to a correct condition which can restart the transaction processing.**
- **<u>Types of recovery:</u>**
- **BACKWARD RECOVERY: Roll back used to undo unwanted changes to the database. It reverses the changes made by transactions which have been aborted.**
- **FORWARD RECOVERY: it starts with a backup copy of the database. The transaction will then reprocess according to the transaction journal that occurred between the time the backup was made and the present time.**
- **Types of back-up proceduresGrandfather-father-son**
- **This procedure refers to at least three generations of backup master files. thus, the most recent backup is the son, the oldest backup is the grandfather. It is commonly used for a batch transaction processing system. If the system fails during a batch run, the master file is recreated by using the son backup and then restarting the batch. However if the son backup fails, is corrupted or destroyed, then the previous generation of backup (the father) is used. Likewise, if that fails, then the generation of backup previous to the father (i.e. the grandfather) is required. Of course the older the generation, the more the data may be out of date. Organizations can have many generations of backup.**
-