Date __/__/____

Database Management System (DBMS)

- Database Management system is a software which is used to managed the database. For example, MySQL, Oracle, etc are a very popular commercial database which is used in different applications.

- DBMS provides an interface to perform various operations like database creation, storing data in it, update data, creating a table in the database and a lot more.

- It provides protection and security to the database. In the case of multiple users, it also maintains data consistency.

* DBMS allows users the following tasks:-

- Data Definition - It is used for creation, modification and removal of definition that defines the organization of data in the database.

- Data Updation - It is used for the insertion, modification and deletion of the actual data in the database.

- Data Retrieval - It is used to retrieve the data from the database.

Date __/__/____

which can be used by applications for various purposes.

- User Administration - It is used for registering and monitoring users, maintain data integrity, enforcing data security, dealing with concurrency control, monitoring performance and recovering information corrupted by ~~expe~~ unexpected failure.

* Advantages of DBMS :-

- Controls database redundancy - It can control data redundancy because it stores all the data in one single database file and that recorded data is placed in the database.
- Data sharing - In DBMS, the ~~auto~~ authorized users of an organization can share the data among multiple users.
- Easily Maintenance - It can be easily maintainable due to the centralized nature of the database system.
- Reduce time - It reduces development time and maintenance need.

Date __/__/____

- Backup – It provides backup and recovery subsystems which create automatic backup of data from hardware and software failures and restores the data if required.

- Multiple user interface – It provides different types of user interfaces like graphical user interfaces, application program interfaces.

* Disadvantages of DBMS :-

- Cost of Hardware and Software – It requires a high speed of data processor and large memory size to run DBMS Software.

- Size – It occupies a large space of disks and large memory to run them efficiently.

- Complexity – Database System creates additional complexity and requirements.

- Higher impact of failure – Failure is highly impacted the database because in most of the organization, all the data stored in a single database and if the database is damaged due to electric failure or database corruption then the data may be lost forever.

Date __/__/____

1. DBMS basics

Data - Data is a collection of facts and figures that can be processed to produce information.

Data Bank - A data bank is a well-organized and maintained collection of data for easy consultation and use. This data repository is made accessible on local and remote servers, and can contain information about a single, dedicated subject or multiple subjects in a well-organized manner.

Database - The database is a collection of inter-related data which is used to retrieve, insert and delete the data ~~efficient~~ efficiently. It is also used to organize the data in the form a of a table, schema, views, and reports etc.

Types of DBMS -

- Relational database
- Object oriented database
- Hierarchical database
- Network database

Date __/__/____

1.) Relational Database - A relational database management System (RDBMS) is a system where data is organized in two-dimensional tables using rows and columns. It is based on SQL.

Relational database management system software is available for personal computers, workstation and large mainframe systems.

Example: Oracle Database, MySQL, Microsoft SQL Server etc.

Std ID	Name	City
101	Vishal	Ranchi
102	Ravindra	Ayodhya
103	Antanu	Ranchi

2.) Object Oriented Database - It is a system where information or data is represented in the form of ~~objects~~ objects which is used in OOP.

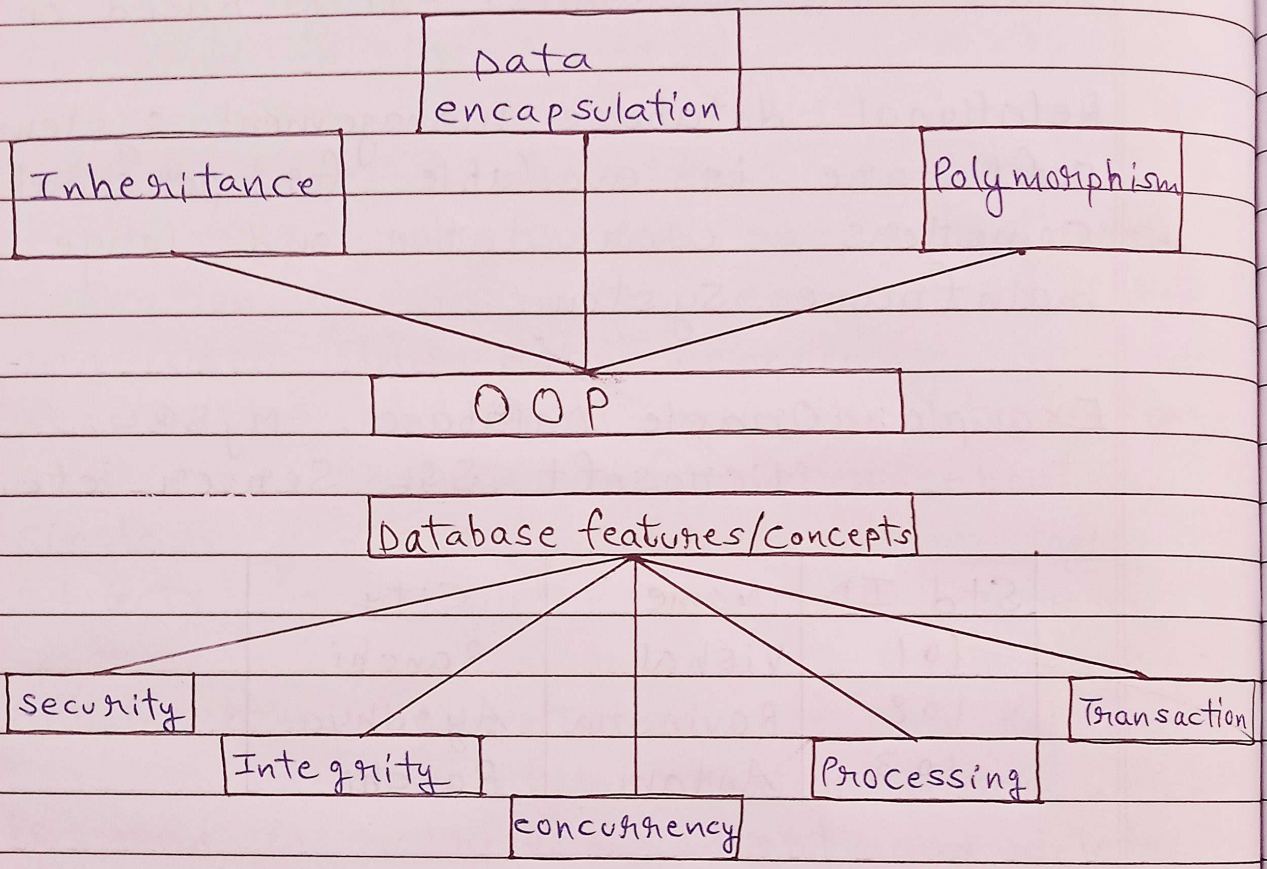
- It is a combination of relational database concepts and object-oriented principles.
- Relational database concepts are concurrency control, transactions, etc.
- OOPs principles are data encapsulation, inheritance and polymorphism.

Page No. ____

Date __/__/____

- It requires less code and is easy to maintain.

Example: Object DB software.

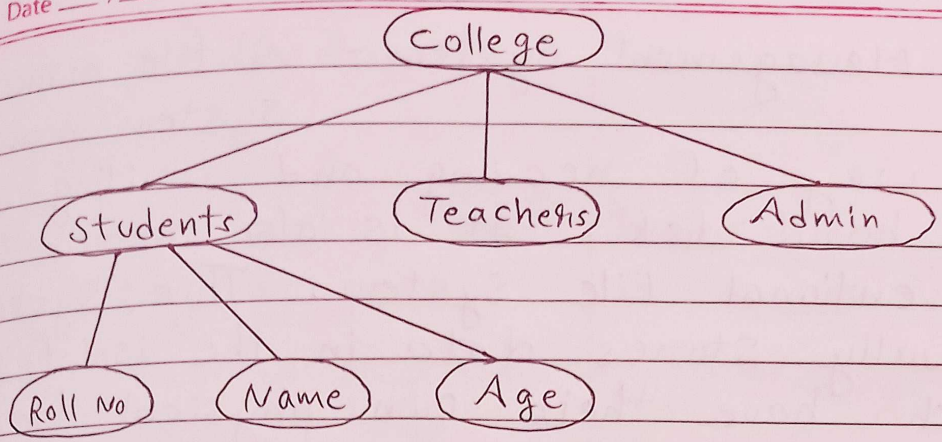


3.) Hierarchical Database – It is a system where the data elements have a one to many relationship (1:N). Here data is organized like a tree which is similar to a folder structure in your computer system.

- The hierarchy starts from the root node, connecting all the child nodes to the parent node.

Example: IMS (IBM), Windows registry (Microsoft)

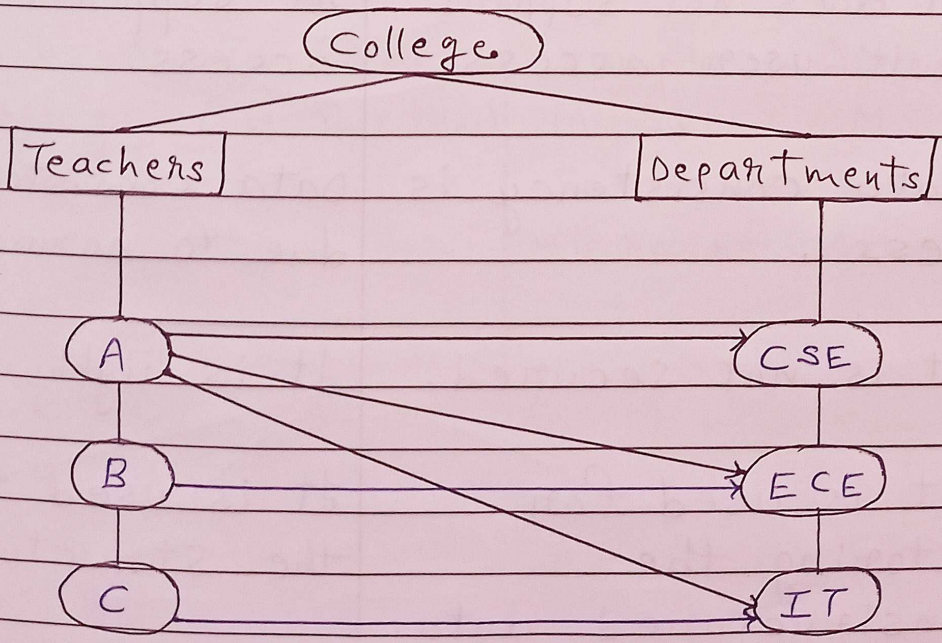
Date __/__/____



4.) Network Database - A Network database management system is a system where the data elements maintain one to one relationship (1:1) or many to many relationship (N:N).

It also has a hierarchical structure, but the data is organized like a graph and it is allowed to have more than one parent for one child record.

Example: Teachers can teach in multiple departments.



Date __/__/____

File Management System – A file management system manages the way of reading and writing data to the hard disk. It is also known as Conventional File System. This system actually stores data in the isolated files which have their own physical location on the drive and users manually go to these locations to access these files.

Difference between File System and DBMS –

File System	DBMS
* It is a software system that manages and controls the data files in a computer system.	It is a software system used for creating and managing the database. DBMS provides a systematic way to access, update and delete data.
* It does not support multi-user access.	It supports multi-user access.
* Data consistency is less.	Data consistency is more due to normalization.
* It is not secured.	It is highly secured.
* It is used for storing the unstructured data.	It is used for storing the structured data.

Date __/__/____

* Data redundancy is high.	Data redundancy is low.
* No data backup and recovery process.	There is a backup recovery for data.
* Handling of a file system is easy.	Handling a DBMS is complex.
* Cost of a file system is less.	Cost of a file system is high.
* If one application fails, it does not affect other application in a system.	If the database fails, it affects all application which depends on it.
* Data cannot be shared.	Data can be shared.
* It does not provide concurrency facility.	It provides concurrency facility.
* Example: NTFS (New Technology File System), EXT (Extended File System), etc.	Example: Oracle, MySQL, MS SQL Server, DB2, Microsof Access, etc.

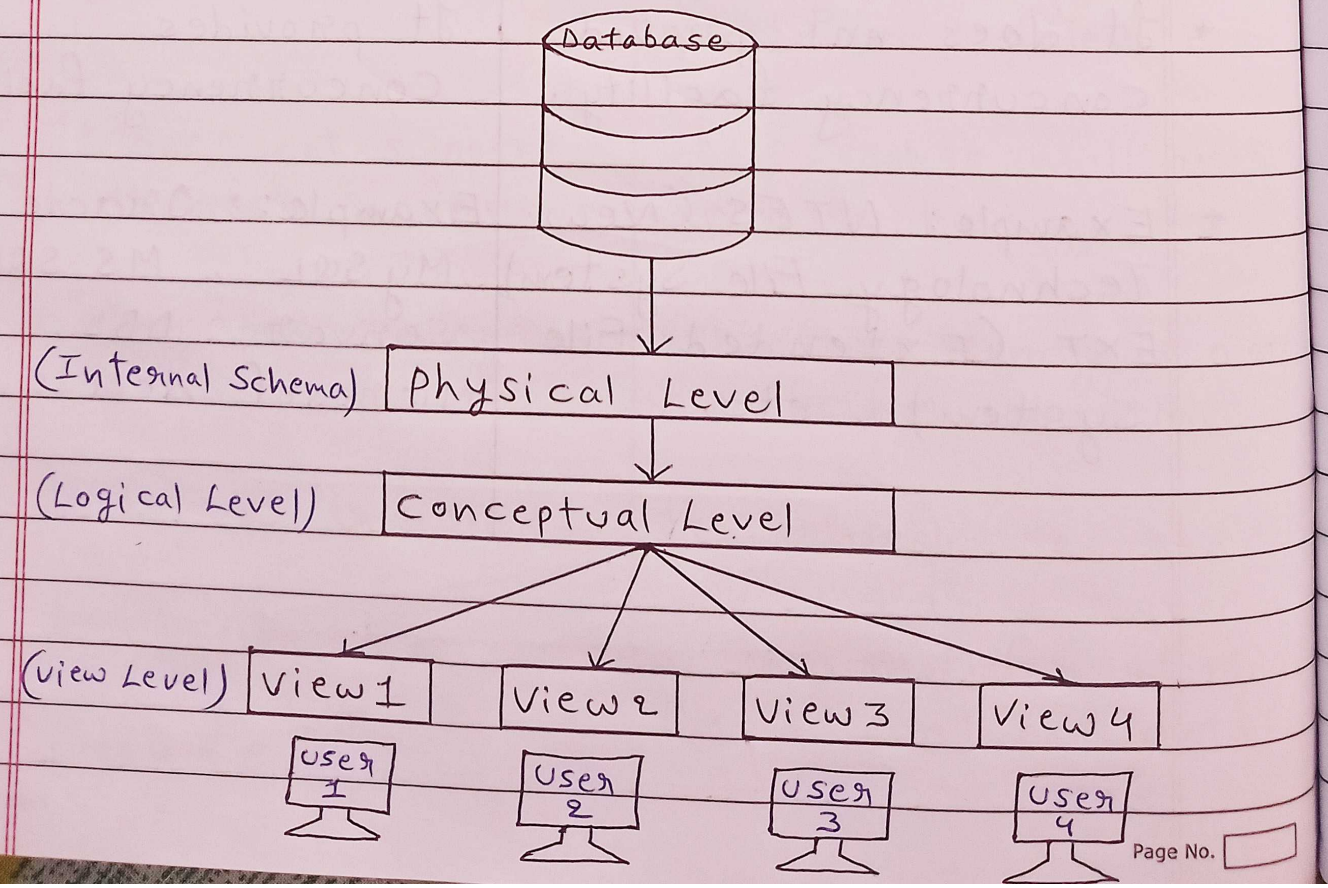
Date __/__/____

Data Abstraction - Data Abstraction is a process of hiding unwanted or irrelevant details from the end user. It provides a different view and helps in achieving data independence which is used to enhance the security of data.

* Levels of abstraction for DBMS -

Levels of abstraction simplyfy database design. There are three levels of abstraction for DBMS, which are as follows -

- Physical or Internal Level
- Logical or Conceptual Level
- View or External Level



Date __/__/____

1.) Physical or Internal Level -

It is the lowest level of abstraction for DBMS which defines how the data is actually stored. It defines data structure to store data and access methods used by the database. Actually, it is decided by developers or database application programmers how to store the data in the database.

2.) Logical or Conceptual Level -

It is the intermediate level or next higher level. It describes what data is stored in the database and what relationship exists among those data. It tries to describe the entire or whole data because it describes what tables to be created and what are the links among those tables that are created.

It is less complex than the physical level. Logical level is used by developers or database administrators (DBA).

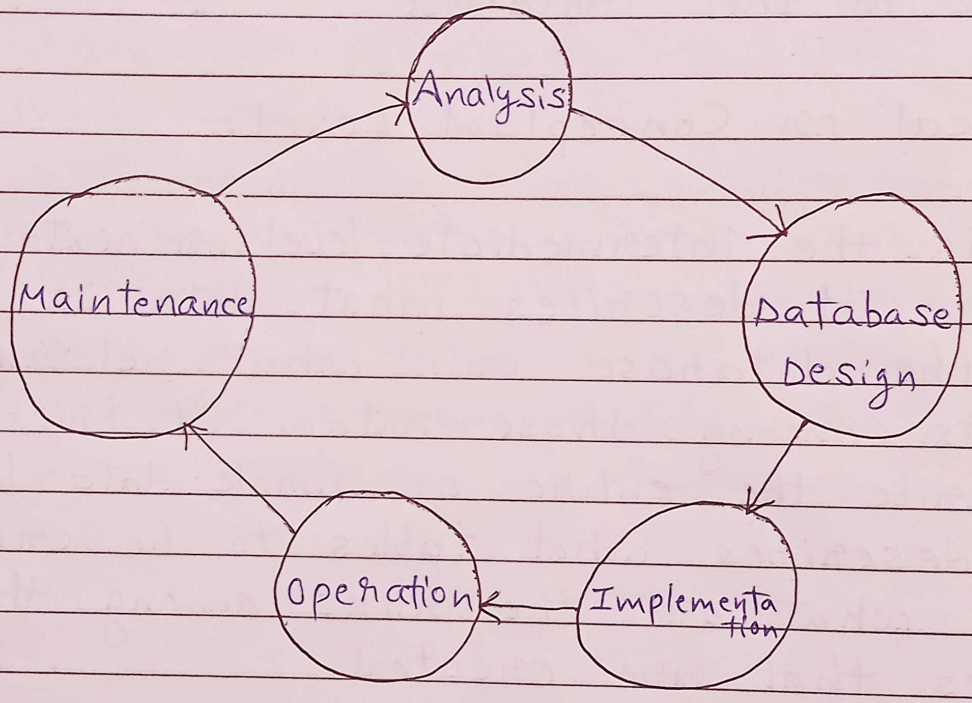
3.) View or External Level -

It is the highest level. In view level, there are different levels of views and every view only defines a part of the entire data. It also simplifies interaction with the

Date __/__/____

user. ~~and~~ View level can be used by all users (all level's users). This level is the least complex and easy to understand.

Database Life Cycle - Life Cycle of database begins with analysis and defining of the problems and objectives.



- Analysis - In the first phase, the current system's operation is analysed and problems are defined. Here the objective are also defined.
- Database Design - Here steps are taken for the final product to meet the user and system requirements.
- Implementation - Design specifications are

Date __/__/____

implemented here.

- Operation - Now the database is operational.
- Maintenance - DBA performs maintenance that includes backup and recovery.

2. RDBMS basics

Relation / Table - Everything in a relational database is stored in the form of relations. The RDBMS database uses tables to store data. A table is a collection of related data entries and contains rows and columns to store data.

- The organized collection of data into a relational table is known as the Logical view of the database.

Codd's Rule - Dr Edgar F. Codd, after his extensive research on the Relational Model of database systems, came up with twelve rules of his own, which a/c to him, a database must obey in order to be regarded as a true relational database.

These rules can be applied on any

Date __/__/____

database system that manages stored data using only its relational capabilities. This is a foundation rule, which acts as a base for all the other rules.

- Rule 1: Information Rule

The data stored in a database, may it be user data or metadata, must be a value of some table cell. Everything in a database must be stored in a table format.

- Rule 2: Guaranteed Access Rule

Every single data element (value) is guaranteed to be accessible logically with a combination of table-name, primary-key (row value), and attribute-name (column value). No other means, such as pointers, can be used to access data.

- Rule 3: Systematic Treatment of NULL Values

The NULL values in a database must be given a systematic and uniform treatment. This is a very important rule because a NULL can be interpreted as one of the following - data is missing, data is not known or data is not applicable.

Date __/__/____

- Rule 4: Active Online Catalog

The structure description of the entire database must be stored in an online catalog, known as data dictionary, which can be accessed by authorized users. Users can use the same query language to access the catalog which they use to access the database itself.

- Rule 5: Comprehensive Data Sub-Language Rule

A database can only be accessed using a language having linear syntax that supports data definition, data manipulation and transaction management operations. This language can be used directly or by means of some application. If the database allows access to data without any help of this language, then it is considered as a violation.

- Rule 6: View Updating Rule

All the views of a database, which can ~~theoree~~ theoretically be updated, must also be updatable by the system.

- Rule 7: High-Level Insert, Update and Delete Rule

A database must support high-level insertion, updation and deletion. This must not be

Page No. ____

Date __/__/____

limited to a single row, i.e. it must also support union, intersection and minus operations to yield sets of data records.

- Rule 8: Physical Data Independence

The data stored in a database must be independent of the applications that access the database. Any change in the physical structure of a database must not have any impact on how the data is being accessed by external applications.

- Rule 9: Logical Data Independence

The logical data in a database must be independent of it's user's view (application). Any change in logical data must not affect the applications using it. For example, if two tables are merged or one is split into two different tables, there should be no impact or change on the user application. This is one of the most difficult rule to apply.

- Rule 10: Integrity Independence

A database must be independent of the application that uses it. All its integrity constraints can be independently modified without the need of any change in

Date __/__/____

the application. This rule makes a database independent of the front-end application and its interface.

- Rule 11: Distribution Independence

The end-user must not be able to see that the data is distributed over various locations. Users should always get the impression that the data is located at one site only. This rule has been regarded as the foundation of distributed database systems.

- Rule 12: Non-Subversion Rule

If a system has an interface that provides access to low-level records, then the interface must not be able to subvert the system and bypass security and integrity constraints.

RDBMS — A relational database management system (RDBMS) is a database management system that is based on the relational model. It has the following major components: Table / Relation, Record / Tuple / Row, Field / column / Attribute, Domain, Instance, Schema, Keys.
Example of RDBMS: MySQL, Oracle, IBM DB2, and Microsoft SQL Server database. An RDBMS includes function that maintain the security, accuracy, integrity and

Date __/__/____

consistency of the data. This is different than the file storage used in a DBMS.

* Record / Tuple / Row - A record is also called as a row of data is each individual entry that exists in a table. A record is a horizontal entity.

* Field / Column / Attribute - Every table is broken up into smaller entities called Fields. A Field is a vertical entity.

* Domain - A domain is a set of permitted values for an attribute in table. A domain of dates can accept all possible valid dates etc. We specify domain of attribute while creating a table.

* Instance - The data stored in database at a particular moment of time is called instance of database.

* Schema - Design of a database is called the schema. Schema is of three types: Physical schema, Logical schema and view schema.

Date __/__/____

Differences b/w RDBMS and DBMS —

RDBMS	DBMS
* Data stored in in table format.	Data stored is in the file format.
* Multiple data elements are accessible together.	Individual access of data elements.
* Data in the form of a table are linked together.	No connection between data.
* Normalisation is not achievable.	There is normalisation.
* Support distributed database.	No support of distributed database.
* Data is stored in a large amount.	Data is stored in a small amount.
* Data redundancy is reduced with the help of key.	Data redundancy is common.
* It supports multiple users.	It supports single user.
* Multiple layers of security.	Low security.
* Software and hardware requirements are higher	Software and hardware requirements are low
* eg: Oracle, SQL Server	eg: XML, MS Access

Date __/__/____

Keys - Keys play an important role in the relational database. It is used to uniquely identify any record or row of data from the table. It is also used to establish and identify relationships between tables. It is an attribute or set of attributes that help to uniquely identify a tuple.

* Super key - It is an attribute set that can uniquely identify a tuple. A super key is a superset of a candidate key.

* Candidate key - It is an attribute or set of attributes that can uniquely identify a tuple.
Except for the primary key, the remaining attributes are considered a candidate key.

* Primary key - A primary key is a candidate key that the database designer selects while designing the database. Primary keys are unique and NOT NULL.

* Alternate key - Candidate keys that are left unimplemented or unused after implementing the primary key are called Alternate keys.

Date __/__/____

* Secondary key - It is the key that has not been selected to be the primary key. However, it is considered a candidate key for the primary key.

* Foreign key - Foreign keys are the column of the table used to point to the primary key of another table.

* Discriminator / Partial key - The set of attributes that are used to uniquely identify a weak entity set is called the Partial key. The partial key of the weak entity set is also known as a discriminator.

* Surrogate key - A surrogate key is a key with virtual or no actual reason and it is used for representing the existence for data analysis. It is a unique identifier that uniquely identifies an object or an entity in their respective fields.

Date __/__/____

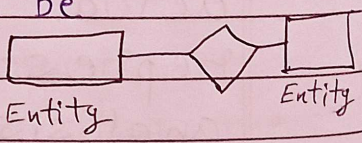
3. ER Model

ER Model stands for an 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.

It develops a conceptual design for the database. It also develops a very simple and easy to design view of data.

In ER modeling, the database structure is portrayed as a diagram called an entity-relationship diagram.

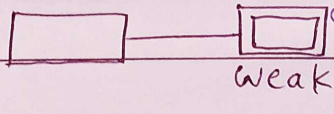
Entity - An entity may be any object, class, person or place. In the ER diagram, an entity can be represented as rectangles.



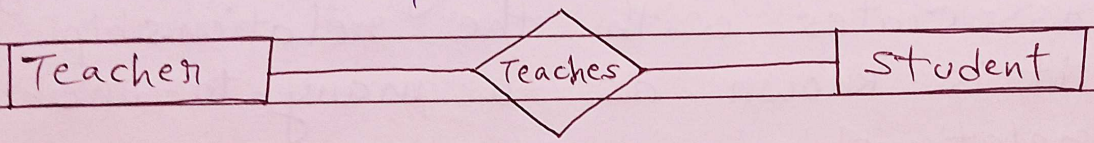
Entity set - An entity set is a group of similar entities and these entities can have attributes. In terms of DBMS, an entity is a table or attribute of a table in database, so by showing relationship among tables and their attributes, ER diagram shows the complete logical structure of a database.

Date __/__/____

Weak Entity - An entity that depends on another entity called a weak entity. The weak entity doesn't contain any key attribute of its own. The weak entity is represented by a double rectangle.

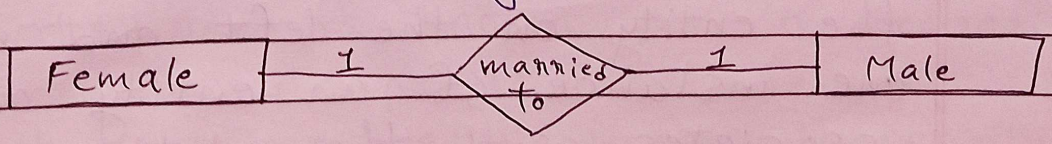


Relationships - A relationship is used to describe the relation between entities. Diamond or rhombus is used to represent the relationship.



* Types of Relationships :-

1.) One-to-one Relationship - when only one instance of an entity is associated with the relationship, then it is known as one-to-one relationship.
EX: A female can marry to one male and a male can marry to one female.

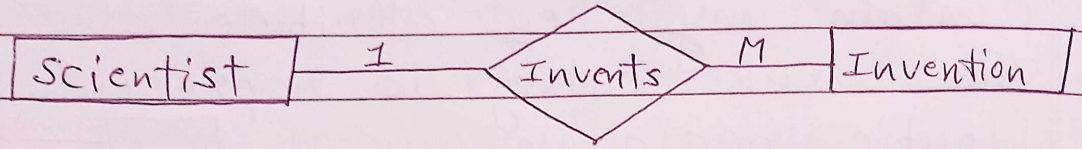


2.) One-to-many Relationship - when only one instance of the entity on the left and more than one instance of an entity on the right associates with the relationship then this is known as a one-to-many relationship.

Page No.

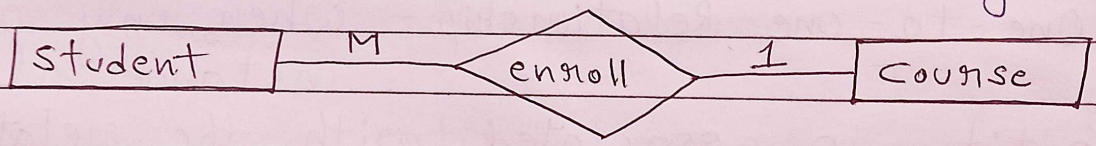
Date ___/___/_____

Ex: Scientist can invent many inventions, but the invention is done by the only specific scientist.



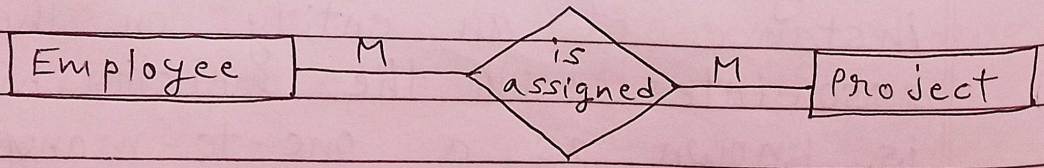
3.) Many-to-one Relationship - when more than one instance of the entity on the left and only one instance of an entity on the right associates with the relationship then it is known as a many-to-one relationship.

Ex: Student enrolls for only one course but a ~~course~~ course can have many students.



4.) Many-to-many Relationship - when more than one instance of the entity on the left and more than one instance of an entity on the right associates with the relationship then it is known as a many-to-many relationship.

Ex: Employee can assigned by many projects and project can have many employees.



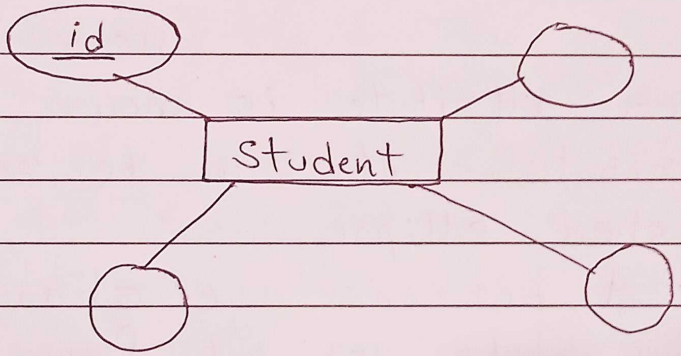
Page No. ____

Date

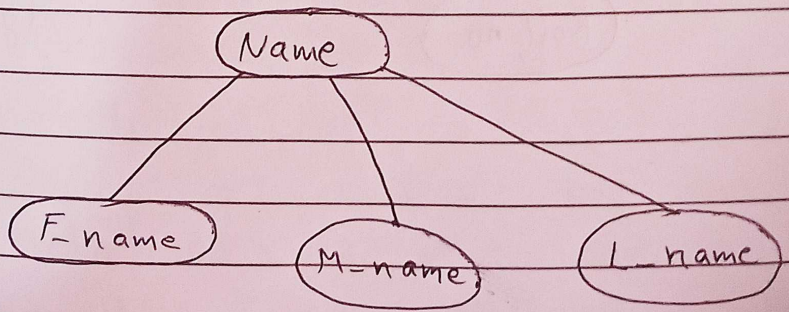
Attributes - The attribute is used to described the property of an entity. Eclipse is used to represent an attribute.

* Types of attribute :-

1) Key attribute - The key attribute is used to represent the main characteristic of an entity. It represents a primary key. The key attribute is represented by an ellipse with the text underline.



2) Composite Attribute - An attribute that composed of many other attributes is known as a composite attribute. The composite attribute is represented by an ellipse and those ellipse are connected with an ellipse.

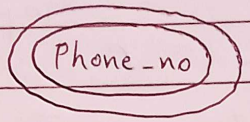


Page No.

Date __/__/____

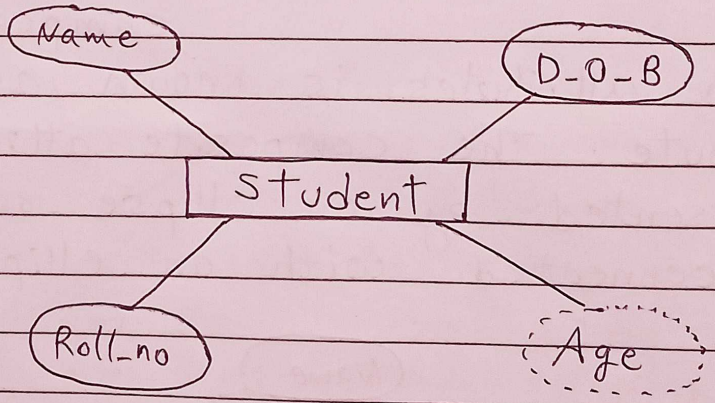
3.) Multivalued Attribute – An attribute can have more than one value. These attributes are known as a multivalued attribute. The double oval is used to represent multivalued attribute.

Ex: a student can have more than one phone no.



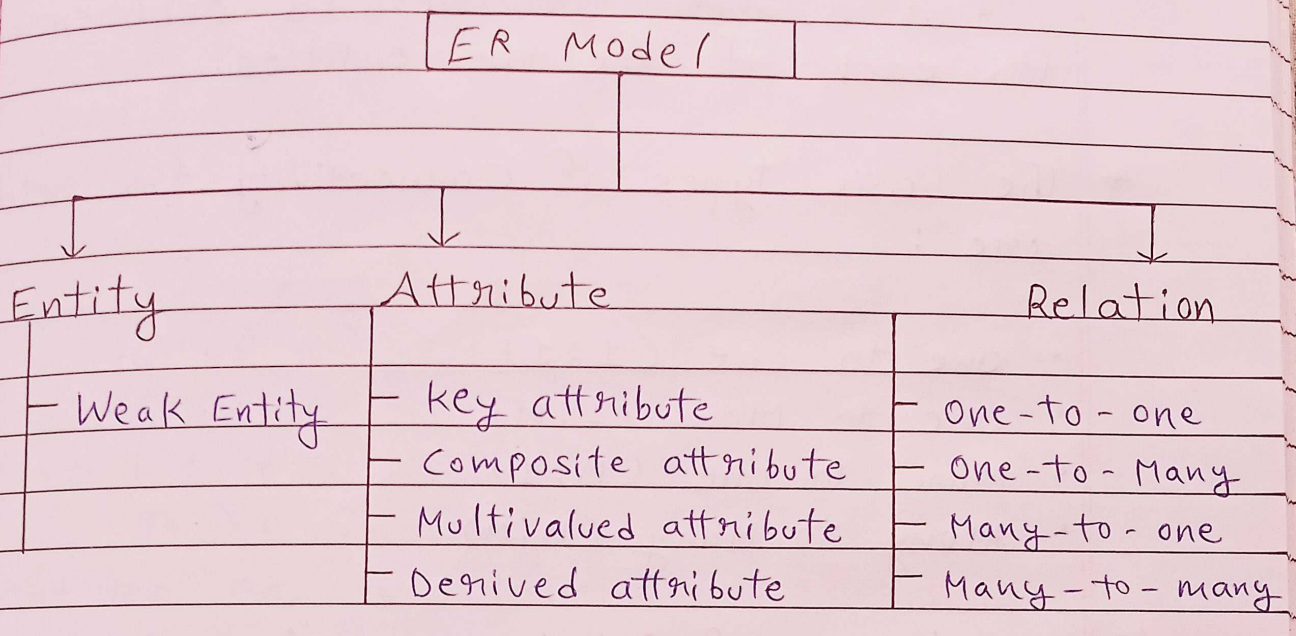
4.) Derived Attribute – An attribute that can ~~have~~ be derived from other attribute is known as a derived attribute. It can be represent by a dashed ellipse.

Ex: A person's age changes over time and can be derived from another attribute like date of birth.



Date __/__/____

Component of ER Diagram -



Degree of Relationship - The number of entity types which took part in the entity relationship is called the degree of relationships.

* There are three types of degree of relationships.

- Unary (degree 1)
- Binary (degree 2)
- Ternary (degree 3)
- N-ary (n degree)

Connectivity and Cardinality of Relationship -

The connectivity of a relationship describes the mapping of associated entity instances in the relationship. The values of connectivity are "one" or "many".

Date __/__/____

The cardinality of a relationship is the actual number of related occurences for each of the two entities.

* The basic types of connectivity for relations are :

- One to one (1:1)
- One to Many (1:M)
- Many to one (M:1)
- Many to Many (M:M)

Existence of Relationship - An entity's existence is dependent on the existence of the related entity. It is existence-dependent if it has a mandatory foreign key ~~(d-e-~~ (i.e., a foreign key attribute that can not be null). Ex: In the COMPANY database, a Spouse entity is existence-dependent on the Employee entity.

Attributes of relationship - Attribute relationships define how tables and columns are joined and used and which tables are related to other tables. without relationships, there is no interaction between data, and therefore no logical structure.

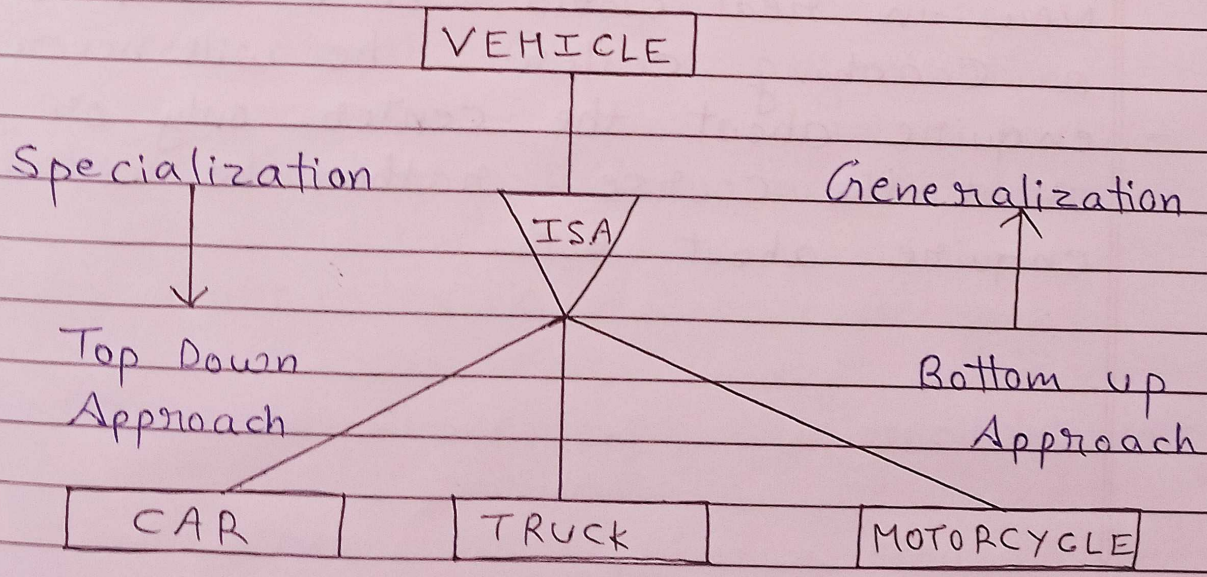
- one to one
- one to Many
- Many to Many

Date ___/___/_____

Generalization - Generalization is the process of extracting common properties from a set of entities and create a generalized entity from it. It is a bottom-up approach in which two or more entities can be generalized to a higher level entity if they have some attributes in common.

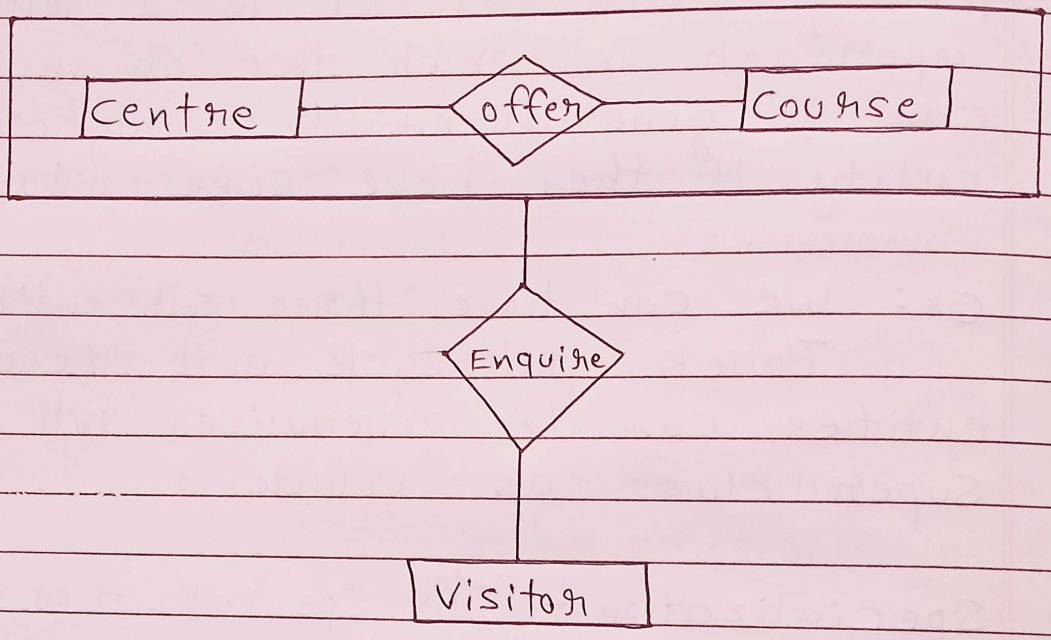
ex: We can have three sub entities as Car, Truck, Motorcycle and these three entities can be generalized into one general super class as Vehicle.

Specialization - In specialization, an entity is divided into sub-entities based on their characteristics. It is a top-down approach where higher level entity is specialized into two or more lower level entities.



Date __/__/____

Aggregation - Aggregation is a process when relation between two entities is treated as a single entity.



In the above diagram, the relationship between center and Course together, is acting as an Entity, which is in relationship with another entity Visitor. Now in real world, if a visitor visits a Coaching center, he will never enquire about the center only or just about the course, rather he will ask enquire about both.

Date __/__/____

4. Normalization

Normalization is the process of organizing the data in the database.

→ Normalization is used to minimize the redundancy from a relation or set of relations. It is also used to eliminate undesirable characteristics like insertion, update and Deletion Anomalies.

→ Normalization divides the larger table into smaller and links them using relationships.

→ The normal form is used to reduce redundancy from the database table.

Advantages of Normalization –

- It helps to minimize data redundancy.
- Greater overall database organization.
- Data consistency within the database.
- Much more flexible database design.
- Enforce the concept of relational integrity.

Disadvantages of Normalization -

- You can not start building the database before knowing what the user needs.
- The performance degrades when normalizing the relations to higher normal forms, i.e., 4NF, 5NF.
- It is very time-consuming and difficult to normalize relations of a higher degree.
- Careless decomposition may lead to a bad database design, leading to serious problems.

Types of Anomalies :-

Following are the types of anomalies that make the table inconsistency, loss of integrity and redundant data.

1.) Data redundancy - It occurs in a relational database when two or more rows or columns have the same value or repetitive value leading to unnecessary utilization of the memory.

2.) Insert Anomaly - It occurs in the relational database when some attributes or data items are

Date __/__/____

to be inserted into the database without existence of other attributes.
Ex: In the student table, if we want to insert a new roll_no, we need to wait until the student enrolled. In this way, It is difficult to insert new record in the table.

3.) Update Anomalies - It occurs when duplicate data is updated only in one place and not in all instances. Hence, it makes our data or table inconsistent state.
Ex: Suppose we have two table, a student table and a course table. If we want to update a course name of a student in student table then we have to update course also in course table, otherwise, data can be inconsistent.

4.) Delete Anomalies - It occurs in the database table when some records are lost or deleted from the database table due to the deletion of other records.
Ex: If we want to remove a student from the student table, it also removes his address, course and other details from the student Table. So, we need to avoid these types of anomalies from the tables and maintain the integrity, accuracy of the database table. Therefore, we use the normalization in DBMS.

Page No.

Date __/__/____

Functional Dependency (FD) -

It is a set of constraints between two attributes in a relation. Functional dependency says that if two tuples have same values for attributes ~~then those two tuples~~ $A_1, A_2, \ldots, A_n$, then those two tuples must have same ~~rate~~ values for attributes $B_1, B_2, \ldots, B_n$.

Functional dependency is represented by an arrow sign ($\rightarrow$) that is, $X \rightarrow Y$, when X functionally determines Y. The left-hand side attributes determines the values of attributes on the right-hand side.

* Types of function dependency :-

1.) Trivial - If a FD $X \rightarrow Y$ holds, where Y is a subset of X, then it is called a trivial FD.

2.) Non-Trivial - If an FD $X \rightarrow Y$ holds, where Y is not a subset of X, then it is called a non-trivial FD.

3.) Completely non-trivial - In a FD $X \rightarrow Y$ holds, where X intersect $Y = \emptyset$, it is said to be completely non-trivial FD.

Page No.

Date __/__/____

First Normal Form (1NF) -

- A relation will be in 1NF if it contains an atomic value.
- It states that an attribute of a table cannot hold multiple values. It must hold only single-valued attribute.
- 1NF disallows the multi-valued attribute, composite attribute and their combinations.

Second Normal Form (2NF) -

- In the 2NF, relation must be in 1NF.
- In the 2NF, all non-key attributes are fully functional dependent on the primary key.

Third Normal Form (3NF) -

- A relation will be in 3NF if it is in 2NF and not contain any transitive partial dependency.
- 3NF is used to reduce the data duplication. It is also used to achieve the data integrity.
- If there is no transitive dependency for non-prime attributes, then the relation must be in 3NF.

Page No.

Date __/__/____

Boyce - Codd Normal Form (BCNF) -

A relation is in BCNF if it is in Third normal form and for every FD, LHS is super key. i.e. A relational is in BCNF if in every non-trivial functional dependency $X \rightarrow Y$, X is a super key.

It is the advance version of 3NF.

Fourth Normal Form (4NF) -

- A relation will be in 4NF if it is in Boyce Codd normal form and has no ~~multivalu~~ multi-valued dependency.
- For a dependency $A \rightarrow B$, if for a single value of A, multiple values of B exists, then the relation will be a Multi-valued Dependency (MVD).

* Join Dependency (JD) - If a table can be recreated by joining multiple tables and each of this table have a subset of attributes of the table, then the table is JD. It is a generalization of Multivalued Dependency.

Date __/__/____

Fifth Normal Form (5NF) -

- A relation is in 5NF if it is in 4NF and not contains any join dependency and joining should be lossless.
- 5NF is satisfied when all the tables are broken into as many tables as possible in order to avoid redundancy.
- 5NF is also known as Project-join Normal Form (PJ/NF).

Domain-Key Normal Form (DK/NF) -

- A relation is DKNF when insertion or delete anomalies are not present in the database. DKNF is the highest form of Normalization. The constraints are verified by the domain an key constraints.
- A table is in DKNF only if it is in 4NF, 3NF and other normal forms. It is based on constraints -
  - Domain constraint
  - Key constraint
  - General constraint

Denormalization -

- It is a database optimization technique in which we add redundant data to one

Date __/__/____

or more tables. This can help us avoid costly joins in a relational database.

Denormalization does not mean not doing normalization. It is an optimization technique that is applied after doing normalization.

* Advantages of Denormalization -

- Retrieving data is faster since we do fewer joins.
- Queries to retrieve can be simpler (and therefore less likely to have bugs), since we need to look at fewer tables.

* Disadvantages of Denormalization -

- Updates and inserts are more expensive.
- Denormalization can make update and insert code harder to write.
- Data may be inconsistent. Which is the "correct" value for a piece of data?
- Data redundancy necessities more storage.

Page No.

Date __/__/____

5. Transaction Concept

Transaction – A transaction can be defined as a group of tasks. A single task is the minimum processing unit which cannot be divided further.

EX: Suppose a bank employee transfers Rs 500 from A's account to B's account. This very simple and small transaction involves several low-level tasks.

A's Account
Open_Account(A)
Old_Balance = A.balance
New_Balance = Old_Balance – 500
A.balance = New_Balance
Close_Account(A)

B's Account
Open_Account(B)
Old_Balance = B.balance
New_Balance = Old_Balance + 500
B.balance = New_Balance
Close_Account(B)

ACID Properties – A transaction in a database system must maintain Atomicity, Consistency, Isolation and Durability - commonly known as ACID properties – in order to ensure

Date __/__/____

accuracy, completeness and data integrity.

* Atomicity - This property states that a transaction must be treated as an atomic unit i.e., either all of its operations are executed or none.
There must be no state in database where a transaction is left partially completed.
States should be defined either before the execution of the transaction or after the execution/abortion/failure of the transaction.

* Consistency - The database must remain in a consistent state after any transaction. No transaction should have any adverse effect on the data residing in the database. If the database was in a consistent state before the execution of a transaction, it must remain consistent after the execution of the transaction as well.

* Durability - The database should be durable enough to hold all its latest updates even if the system fails or restarts. If a transaction updates a chunk of data in a database and commits, then the database will hold the

Page No.

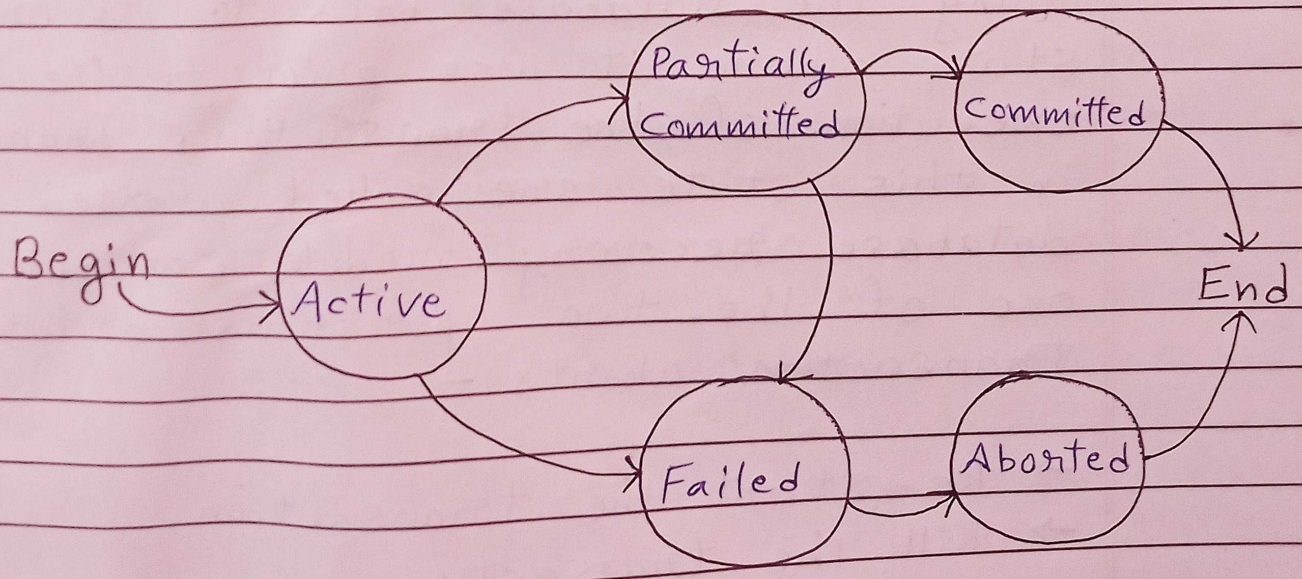
Date __/__/____

modified data. If a transaction commits but the system fails before the data could be written on to the disk, then that data will be updated once the system springs back into action.

* Isolation - In a database system where more than one transaction are being executed simultaneously and in parallel, the property of isolation states that all the transactions will be carried out and executed as if it is the only transaction in the system. No transaction will affect the existence of any other transaction.

Transaction States -

A transaction in a database can be in one of the following states -



Date __/__/____

- Active - In this state, the transaction is being executed. This is the initial state of every transaction.
- Partially Committed - When a transaction executes its final operation, it is said to be in a partially committed state.
- Failed - A transaction is said to be in a failed state if any of the checks made by the database recovery system fails. A failed transaction can no longer proceed further.
- Aborted - If any of the checks fails and the transaction has reached a failed state, then the recovery manager rolls back all its write operations on the database to bring the database back to its original state where it was prior to the execution of the transaction. Transaction in this state are called aborted. The database recovery module can select one of the two operations after a transaction aborts -

→ Re-start the transaction
→ Kill the transaction

Date __/__/____

- Committed – If a transaction executes all its operations successfully, it is said to be committed. All its effects are now permanently established on the database system.

Concurrent Executions – In the transaction process, a system usually allows executing more than one transaction simultaneously. This process is called a concurrent execution.

* Advantages of concurrent execution of a transaction –

- Decrease waiting time or turnaround time.
- Improve response time.
- Increased throughput or resource utilization.

Serializability – Serializability is the concept in a transaction that helps to identify which non-serial schedule is correct and will maintain the database Consistency. It relates to the isolation property of transaction in the database.

Serializability is a concurrency scheme where the execution of concurrent transactions is equivalent to the transactions which execute serially.

Date __/__/____

* Types of Serializability -

1. Conflict Serializability
2. View Serializability

* Example : Let's take two transactions T1 and T2, if both transactions are performed without interfering each other then it is called as serial schedule, It can be represented as follows,

T1	T2
READ 1 (A)	
WRITE 1 (A)	
READ 1 (B)	
C1	
	READ 2 (B)
	WRITE 2 (B)
	READ 2 (B)
	C2

* Non-Serial schedule - when a transaction is overlapped between the transaction T1 and T2,

T1	T2
READ 1 (A)	
WRITE 1 (A)	
	READ 2 (B)
	WRITE 2 (B)
READ 1 (B)	
WRITE 1 (B)	
READ 1 (B)	

Page No.

Date __/__/____

1) Conflict Serializability - It orders any conflicting operations in the same way as some serial execution. A pair of operation is said to be conflict if they operate on the same data item and one of them is a write operation.

That means

- Readi(x) readj(x) - non conflict read - read operation.
- Readi(x) writej(x) - conflict read - write operation.
- Writei(x) readj(x) - conflict write - read operation.
- Writei(x) writej(x) - conflict write - write operation.

2) View Serializability - A schedule is view-serializability if it is viewed equivalent to a serial schedule.

The rules it follows are as follows -

- T1 is reading the initial value of A, then T2 also reads the initial value of A.
- T1 is reading the value written by T2, then T2 also reads the value written by T1.
- T1 is writting the final value and then T2 also has the write operation as the final value.

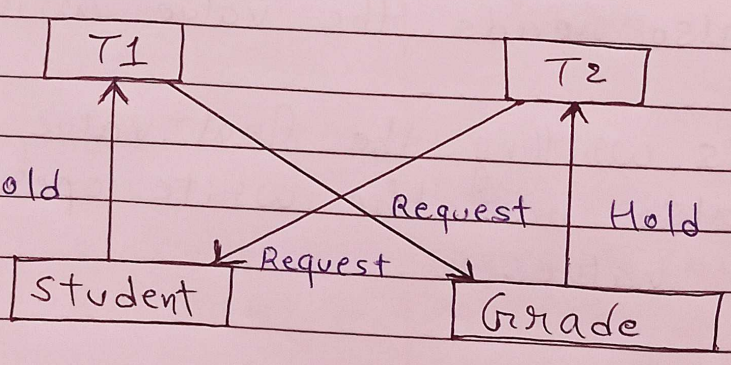
Page No.

Date __/__/____

Deadlock - A deadlock is a condition where two or more transactions are waiting indefinitely for one another to give up locks. Deadlock is said to be one of the most feared complications in DBMS as no task ever gets finished and is in waiting state forever.

Example: In the student table, transaction T1 holds a lock on some rows and needs to update some rows in the grade table. Simultaneously, transaction T2 holds locks on some rows in the grade table and needs to update the rows in the Student table held by Transaction T1.

Now, the main problem arises. Now T1 is waiting for T2 to release its lock and similarly, T2 is waiting for T1 to release its lock. All activities come to a halt state and remain at a standstill. It will remain in a standstill until the DBMS detects the deadlock and aborts one of the transactions.



Deadlock in DBMS

Date __/__/____

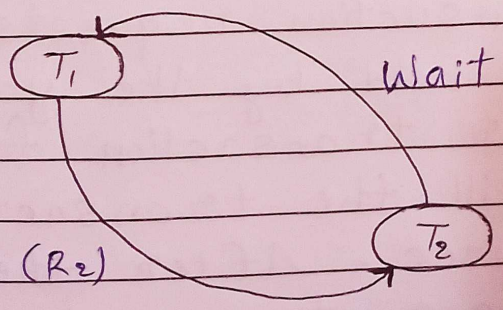
Deadlock Detection - In a database, when a transaction waits indefinitely to obtain a lock, then the DBMS should detect whether the transaction is involved in a deadlock or not. The lock manager maintains a wait for the graph to detect the deadlock cycle in the database.

* Wait for Graph -

- This is the suitable method for deadlock detection. In this method, a graph is created based on the transaction and their lock. If the created graph has a cycle or closed loop, then there is a deadlock.

- The wait for the graph is maintained by the system for every transaction which is waiting for some data held by the others. The system keeps checking the graph if there is any cycle in the graph.

* The wait for a graph for the above scenario is shown below:



Date __/__/____

Deadlock Prevention -

- Deadlock prevention method is suitable for a large database. If the resources are allocated in such a way that deadlock never occurs, then the deadlock can be prevented.

- The DBMS analyzes the ~~or~~ operations of the transaction whether they can create a deadlock situation or not. If they do, then the DBMS never allowed that transaction to be executed.

* Wait-Die scheme - In this scheme, if a transaction requests for a resource which is already held with a conflicting lock by another transaction then the DBMS simply checks the timestamp of both transactions. It allows the older transaction to wait until the resource is available for execution.

* Wound-wait scheme - In wound-wait scheme, if the older transaction requests for a resource which is held by the younger transaction, then older transaction forces younger one to kill the transaction and release the resource. After the minute delay, the younger transaction is restarted

Page No.

but with the same timestamp. If the older transaction has held a resource which is requested by the younger transaction, then the younger transaction is asked to wait until older releases it.

Deadlock Avoidance -

- When a database is stuck in a deadlock state, then it is better to avoid the database rather than aborting or restarting the database. This is a waste of time and resources.

- Deadlock avoidance mechanism is used to detect any deadlock situation in advance. A method like "wait for graph" is used for detecting the deadlock situation but this method is suitable only for the smaller database. For the larger database, deadlock prevention method can be used.

Date __/__/____

6. Relationship algebra & Calculus

Relational database systems are expected to be equipped with a query language that can assist its users to query the database instances. There are two kinds of query languages - relational algebra and relational calculus.

Relational Algebra -

- Relational algebra is a procedural query language, which takes instances of relations as input and ~~yet~~ yields instances of relations as output. It uses operators to perform queries.

- An operator can be either unary or binary. They accept relations as their input and yield relations as their output.

- Relational algebra is performed recursively on a relation and intermediate results are also considered relations.

The fundamental operations of relational algebra are as follows -

- Select
- Project

Page No.

Date __/__/____

- Union
- Set different
- Cartesian product
- Rename

* Select Operation (σ) –

It selects tuples that satisfy the given predicate from a relation.

Notation – $\sigma_p(r)$

where σ stands for selection predicate and r stands for relation. p is prepositional logic formula which may use connectors like and, or, not. These terms may use relational operators like $=, \neq, \geq, <, >, \leq$.

Example:
$\sigma_{subject = "database"}(Books)$
output – Selects tuples from books where subject is 'database'.

$\sigma_{subject = "database" \text{ and } price = "450"}(Books)$
Output – Selects tuples from books where Subject is 'database' and price is '450'.

* Project Operation (Π) –

It projects column(s) that satisfy a given predicate.

Date __/__/____

Notation - $\Pi_{A_1, A_2, A_n}(r)$

where A_1, A_2, A_n are attribute names of relation r.
Duplicate rows are automatically eliminated as relation is a set.

Example:
$\Pi_{subject, author}(Books)$
output - Selects and projects columns named as subject and author from the relation Books.

* Union operation (U) -

It performs binary union between two given relations and is defined as -

$r \cup s = \{t \mid t \in r \text{ or } t \in s\}$

Notation - $r \cup s$

where r and s are either database relations or relation result set (temporary relation).

→ For a union operation to be valid, the following conditions must hold -

- r and s must have the same number of attributes.
- Attribute domains must be compatible.

Page No.

Date __/__/____

- Duplicate tuples are automatically eliminated.

Example:
$\Pi_{author}(BOOKS) \cup \Pi_{author}(Articles)$
output - Projects the names of the authors who have either written a book or an article or both.

* Set difference (-) -

The result of set difference operation is tuples, which are present in one relation but are not in the second relation.

Notation - $r - s$
Finds all the tuples that are present in r but not in s.

Example:
$\Pi_{author}(BOOKS) - \Pi_{author}(Articles)$
output - Provides the name of authors who have written books but not articles.

* Cartesian Product (x) -

Combines information of two different relations into one.

Notation - $r \times s$
where r and s are relations and their output will be defined as -
$r \times s = \{qt \mid q \in r \text{ and } t \in s\}$

Page No.

Date __/__/____

Example:
$\sigma_{author = 'Vishal'}(BOOKS \times Articles)$
Output - Yields a relation, which shows all the books and articles written by vishal.

* Rename Operation (ρ) -

The results of relational algebra are also relations but without any name. The rename operation allows us to rename the output relation. 'rename' operation is denoted with small Greek letter rho (ρ).

Notation - $\rho_x(E)$
where the result of expression E is saved with name of x.

* Set Intersection (∩) -

Suppose there are two tuples R and S. The set intersection operation contains all tuples that are in both R & S.

Notation - $R \cap S$

Example:
$\Pi_{Customer\_Name}(BORROW) \cap \Pi_{Customer\_Name}(Depositor)$
Output -

Customer_Name
Smith
Jones

Page No.

Date __/__/____

* Join operations -

Join operation is essentially a cartesian product followed by a selection criterion. Join operation denoted by ⋈. Join operation also allows joining variously releated tuples from different relations.

→ Types of Join :-

* Inner joins -

- Theta join
- EQUI join
- Natural join

* Outer joins -

- Left Outer join
- Right outer join
- Full outer join

Inner Join - In an inner join, only those tuples that satisfy the matching criteria are included, while the rest are excluded.

* Theta join - The general case of JOIN operation is called a Theta join. It is denoted by symbol θ.
Ex: $A \bowtie_\theta B$

Page No.

Date ___/___/_____

Theta join can use any conditions in the selection criteria.

ex: $A \bowtie_{A.column\ 2 > B.column\ 2} (B)$

$A \bowtie_{A.Column\ 2 > B.Column\ 2} (B)$	
Column 1	Column 2
1	2

* EQUI Join - When a theta join uses only equivalence condition, it becomes a equi join.

Ex: $A \bowtie_{A.column\ 2 = B.column\ 2} (B)$

$A \bowtie_{A.Column\ 2 = B.Column\ 2} (B)$	
Column 1	Column 2
1	1

EQUI join is the most difficult operations to implement efficiently using SQL in an RDBMS and one reason why RDBMS have essential performance problems.

* Natural Join - Natural join can only be performed if there is a common ~~attrib~~ attribute (column) between relations. The name and the type of the attribute must be same.

Ex: consider the following two tables:

Date __/__/____

C	
Num	Square
2	4
3	9

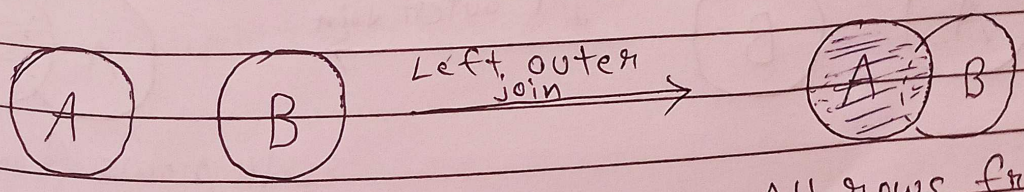
D	
Num	Cube
2	8
3	27

C ⋈ D

C ⋈ D		
Num	Square	Cube
2	4	8
3	9	27

Outer Join – In an outer join, along with tuples that satisfy the matching criteria, we also include some or all tuples that do not match the criteria.

* Left Outer Join (A ⋈ B) – In the left outer join, operation allows keeping all tuple in the left relation. However, if there is no matching tuple is found in right relation in the join result are filled with null values.



All rows from left table

Date __/__/____

Ex: Consider the following 2 tables.

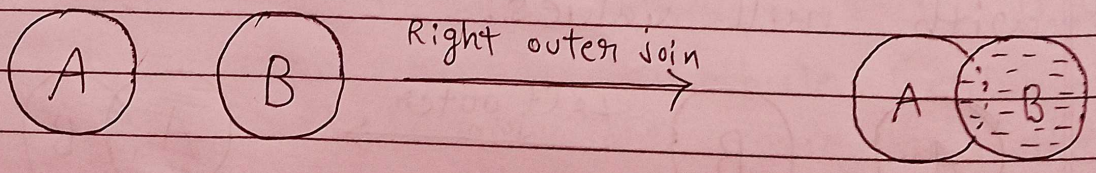
A	
Num	Square
2	4
3	9
4	16

B	
Num	Cube
2	8
3	27
5	125

A ⋈ B

A ⋈ B		
Num	Square	Cube
2	4	8
3	9	27
4	16	-

* Right Outer Join (A ⟖ B) - In the right outer join, operation allows keeping all tuples in the right relation. However, if there is no matching tuple is found in the left relation, then the attributes of the left relation in the join result are filled with null values.



All rows from right table

A ⟖ B

Date __/__/____

A ⋈ B		
Num	Cube	Square
2	8	4
3	27	9
5	125	—

* Full Outer Join (A ⟗ B) - In a full outer join, all tuples from both relations are included in the result, irrespective of the matching condition.

A ⟗ B

A ⋈ B		
Num	Cube	Square
2	8	4
3	27	9
4	—	16
5	125	—

Division operation - The division operator is used for queries which involve the 'all'.
R1 ÷ R2 = tuples of R1 associated with all tuples of R2.

Date __/__/____

Relational Calculus - In contrast to relational algebra, Relational calculus is a non-procedural query language, that is, it tells what to do but never explains how to do it.

→ Relational calculus exists in two forms:-

* Tuple Relational calculus (TRC) -

Filtering variable ranges over tuples.

Notation - {T | condition}
Returns all tuples T that satisfies a condition.
Ex: {T.name | Author(T) AND T.article = 'database'}

output - Returns tuples with 'name' from author who has written article on 'database'.

TRC can be quantified. We can use Existential (∃) and Universal Quantifiers (∀).

Ex: {R | ∃T ∈ Authors(T.article = 'database' AND R.name = T.name)}

output - The ~~query~~ above query will yield the same result as the previous one.

Page No.

Date __/__/____

* Domain Relational Calculus (DRC) -

In DRC, the filtering variable uses the domain of attributes instead of entire tuple values (as done in TRC).

Notation - $\{a_1, a_2, a_3, \ldots, a_n \mid P(a_1, a_2, \ldots, a_n)\}$
where a_1, a_2 are attributes and P stands for formulae built by inner attributes.

Ex: $\{<article, page, subject> \mid \in$ Vishal $\wedge$
subject = 'database'$\}$

Output - Yields Article, Page and Subject from the relation vishal, where subject is database.

- Just like TRC, DRC can also be written using existential and universal quantifiers. DRC also involves relational operators.

- The expression power of TRC and DRC is equivalent to Relational Algebra.

Date __/__/____

7. Oracle SQL

Oracle SQL - Structured Query Language (SQL) is the set of statements with which all programs and users access data in an oracle database. Application programs and oracle tools often allow users access to the database without using SQL directly, but these applications in turn must use SQL when executing the user's request.

- SQL - SQL is Structured Query Language, which is a computer language for storing, manipulating and retrieving data stored in relational database.

SQL is the standard language for Relational Database System. All the RDBMS like MySQL, MS Access, Oracle, Sybase, Informix, Postgres and SQL Server use SQL as their standard database language.

- Applications of SQL -

→ Allows users to access data in the RDBMS.
→ Allows users to describe the data.

Page No.

Date __/__/____

→ Allows users to define the data in a database and manipulate that data.

→ Allows to embed within other languages using SQL modules, libraries & pre-compilers.

→ Allows users to create and drop databases and tables.

→ Allows users to create view, stored procedure, functions in a database.

→ Allows users to set permissions on tables, procedures and views.

SQL * Plus – SQL * Plus is an interactive and batch query tool that is installed with every Oracle Database installation. It has a command-line user interface, a windows Graphical User Interface (GUI) and the iSQL * Plus Web-based user interface.

SQL * Plus has its own commands and environment and it provides access to the Oracle Database. It enables you to enter and execute SQL, PL/SQL, SQL* Plus and operating system commands to perform the following:

- Format, perform calculations on, store and

Date __/__/____

print from query results.

- Examine table and object definitions.
- Develop and run batch scripts
- Perform database administration

SQL Buffer - SQL* Plus keeps a copy of the most recently entered SQL statement or PL/SQL block in an internal memory area known as the SQL buffer, often referred to as the buffer. Command-line SQL* Plus needs a place to store your statement or block until you are finished entering it.

SQL* Plus provides you with the ability to edit the statement in the buffer.

Environment Variables - Environment variable to specify where SQL* Plus is installed. It is also used by SQL* Plus to specify where message files are located.
Environment variable to specify the locations of the NLS data and the user boot file in SQL* Plus 10.2.
The default location is $ORACLE_HOME/nls/data

Date __/__/____

Data Types - A data type specifies a particular type of data, such as integer, floating-point, Boolean etc.
A data type also specifies the possible values for that type, the operations that can be performed on that type and the way the values of that type are stored.

Oracle Built-in Data Types -

VARCHAR2, NVARCHAR2, NUMBER, FLOAT, LONG, DATE, BINARY_FLOAT, BINAY_DOUBLE, TIMESTAMP, INTERVAL YEAR, INTERVAL DAY, RAW, LONG RAW, ROWID, UROWID, CHAR, NCHAR, CLOB, NCLOB, BLOB, BFILE

Basic parts of Speech in SQL :

* SELECT - The SELECT statement is used to retrieve data from the data tables.
Syntax: SELECT Name, Surname FROM Students

* WHERE - WHERE clause is used to filter the data according to specified conditions.
Syntax: SELECT * FROM Students WHERE Age >= 20

Date __/__/____

* ORDER BY - ORDER BY statement helps us to sort the data either ascending or descending.
By defaut ORDER BY statement sorts data in ascending order.

Syntax: SELECT * FROM Student ORDER BY MARKS DESC

* FROM - The FROM clause is used to list the tables ~~any~~ and any joins required for the SQL statement.

* HAVING - The HAVING clause enables you to specify conditions that filter which group results appear in the results.

→ The WHERE clause places conditions on the selected columns, whereas the HAVING clause place conditions on groups created by the Group BY clause.

Syntax: SELECT student, percentage FROM Student GROUP BY student, percentage HAVING percentage > 95;

* GROUP BY - The GROUP BY clause is used in collaboration with the SELECT statement to arrange identical data into groups. This GROUP BY clause follows the WHERE clause in a SELECT statement and precedes the ORDER BY clause.

Page No.

Date __/__/____

Syntax: SELECT column1, function_name (column2)
FROM table_name WHERE condition
GROUP BY column1, column2
ORDER BY column1, column2

Operators :-

1.) Arithmetic Operators -

unary (+, -) Binary (+, -, *, /, %)

2.) Comparision Operators -

=, !=, <>, >, <, >=, <=

3.) Logical operators -

IN (TRUE if the operand is equal to one of a list of expression)

ALL (TRUE if all of the subquery values meet the condition)

AND (TRUE if all the conditions separated by AND is TRUE)

ANY (TRUE if any of the subquery values meet the condition)

BETWEEN (TRUE if the operand is within the range of comparisions)

Date __/__/____

EXISTS (TRUE if the subquery returns one or more records)

LIKE (TRUE if the operand matches a pattern)

NOT (Displays a record if the conditions is NOT TRUE)

OR (TRUE if any of the conditions separated by OR is TRUE)

SOME (TRUE if any of the subquery values meet the condition)

NOT IN (Used to replace a group of arguments using the <> (or !=) operator that are combined with an AND)

IS NULL (Used checking if the value of a column is null or not)

IS NOT NULL (Used to test for a non-NULL value)

4.) Set operators -

UNION, UNION ALL, INTERSECT, MINUS

Date __/__/____

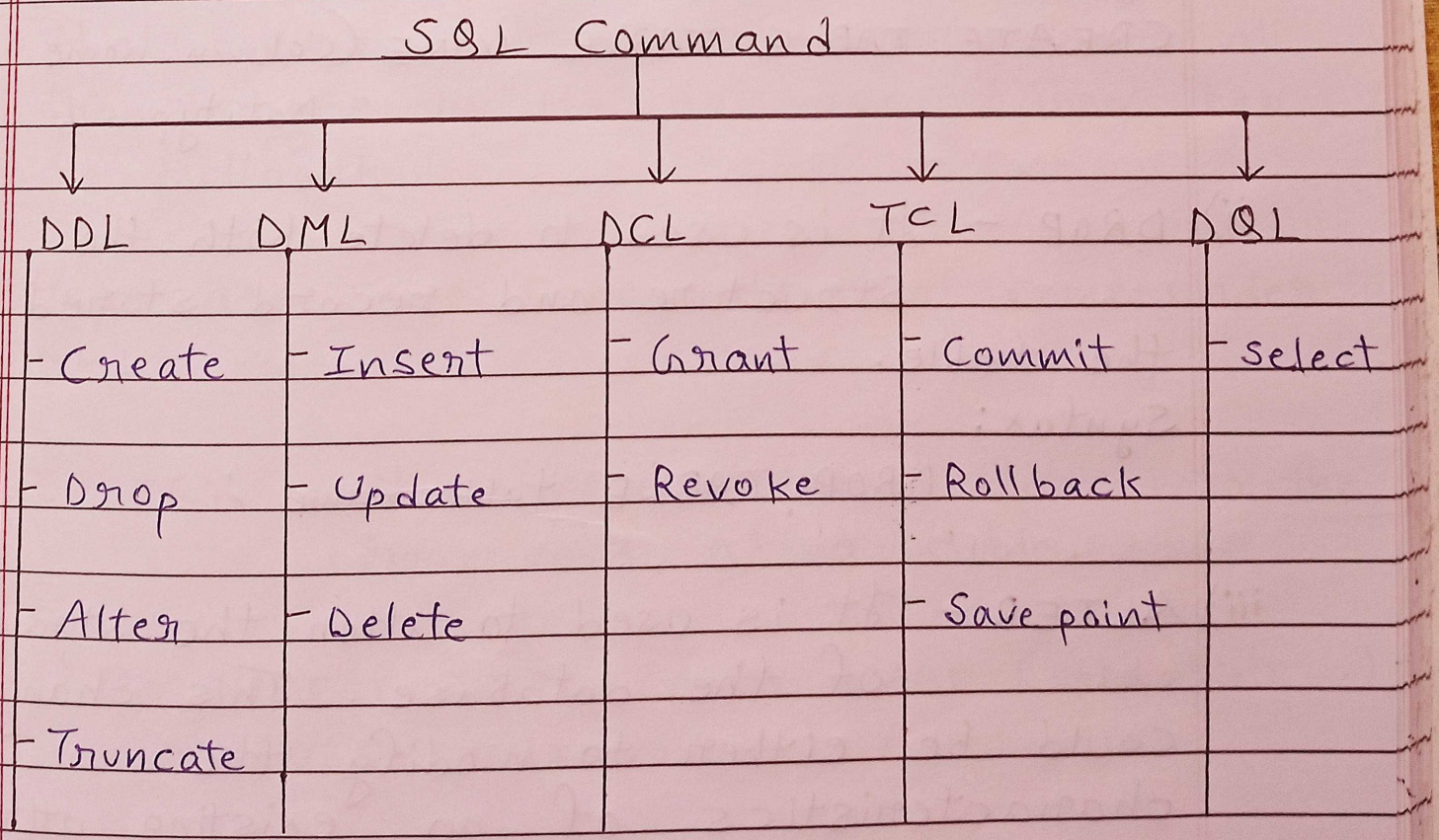
SQL Commands :-

- SQL commands are instructions. It is used to communicate with the database. It is also used to perform specific tasks, functions and queries of data.

- SQL can perform various tasks like create a table, add data to tables, drop the table, modify the table, set permission for users.

Types of SQL commands -

There are five types of SQL commands.



Date __/__/____

1) Data Definition Language (DDL) -

- DDL changes the structure of the table like creating a table, deleting a table, altering a table, etc.
- All the command of DDL are auto-committed that means it permanently save all the changes in the database.

→ Here are some commands that come under DDL :

i) CREATE - It is used to create a new table in the database.

Syntax:

```
CREATE TABLE TABLE_NAME (Column_name
                         Datatypes [...]);
```

ii) DROP - It is used to delete both the structure and record stored in the table.

Syntax:

```
DROP TABLE table_name ;
```

iii) ALTER - It is used to alter the structure of the database. This change could be either to modify the characteristics of an existing attribute or probably to add a new attribute.

Syntax:

```
ALTER TABLE table_name ADD colum_name COLUMN
definition ;
```

Page No.

Date __/__/____

ALTER TABLE table_name MODIFY (column-definitions...);

iv) TRUNCATE - It is used to ~~del~~ delete all the rows from the table and free the space containing the table.

Syntax:

TRUNCATE TABLE table_name ;

2.) Data Manipulation Language (DML) -

- DML commands are used to modify the database. It is responsible for all form of changes in the database.
- The command of DML is not auto-committed that means it can't permanently save all the changes in the database. They can be rollback.

→ Here are some commands that come under DML :

i) INSERT - It is used to insert data into the row of a table.

Syntax:

INSERT INTO TABLE_NAME (col1, col2...) VALUES (val1, val2...)

ii) UPDATE - It is used to update or modify the value of a column in the table.

Page No.

Date __/__/____

Syntax:
UPDATE table_name SET [column_name 1 = val1, ... column_name n = val n] [WHERE condition]

iii) DELETE - It is used to remove one or more row from a table.

Syntax:
DELETE FROM table_name [WHERE condition];

3.) Data Control Language (DCL) -

DCL commands are used to grant and take back authority from any database user.

→ Here are some commands that come under DCL:

i) GRANT - It is used to give user access privileges to a database.

ex:
GRANT SELECT, UPDATE ON MY_TABLE TO SOME_USER, ANOTHER_USER;

ii) REVOKE - It is used to take back permissions from user.

ex:
REVOKE SELECT, UPDATE ON MY_TABLE FROM USER1, USER2;

Date __/__/____

4) Transaction Control Language (TCL) -

- TCL commands can only use with DML Commands like INSERT, DELETE and UPDATE only.

- These operations are automatically committed in the database that's why they cannot be used while creating tables or dropping them.

→ Here are some commands that come under TCL :

i) COMMIT - It is used to save all the transactions to the database.

Syntax: COMMIT ;

ii) ROLLBACK - It is used to undo transactions that have not already been saved to the database.

Syntax: ROLLBACK ;

iii) SAVEPOINT - It is used to roll the transaction back to a certain point without rolling back the entire transaction.

Syntax: SAVEPOINT Savepoint-Name ;

Date __/__/____

5.) Data Query Language (DQL) -

DQL is used to fetch the data from the database.

→ It uses only one command :

- SELECT - This is the same as projection operation of relational algebra. It is used to select the attribute based on the condition described by WHERE clause.

syntax: SELECT expressions FROM TABLES WHERE conditions ;

Database Objects - A database object is any defined object in a database that is used to store or reference data. Anything which we make from create command is known as Database object. It can be used to hold and manipulate the data.

→ Some of the examples of database objects are:

- Table - Basic unit of storage ; composed rows and columns.

- View - Logically represents subsets of data from one or more tables.

Page No.

Date __/__/____

- Sequence - Generates primary key values
- Index - Improves the performance of some queries.
- Synonym - Alternative name for an object.

→ Different database objects :

1.) Table - This database object is used to create a table in database.

Syntax:

```
CREATE TABLE [Schema.] table (column datatype
        [DEFAULT expr][,...]);
```

2.) View - This database object is used to create a view in database. A view is a logical table based on a table or another view. A view contains no data of its own but is like a window through which data from tables can be viewed or changed. The table on which a view is based are called base tables. The view is stored as a SELECT statement in the data dictionary.

Syntax:

```
CREATE [OR REPLACE] [FORCE | NOFORCE] VIEW view
    [alias[, alias]...]]
    As subquery
    [WITH CHECK OPTION [CONSTRAINT constraint]]
    [WITH READ ONLY [CONSTRAINT constraint]];
```

Page No. ____

Date __/__/____

Simple queries - A simple query is a query that searches using just one parameter.

A simple query might use all of the fields in a table and search using just one parameter.

Nested sub-query - A Subquery or Inner query or a Nested query is a query within another SQL query and embedded within the WHERE ... clause.

A Subquery is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved.

Syntax:

SELECT column_name [, column_name] FROM table 1 [, table 2] WHERE Column_name OPERATOR (select column_name [, column_name] FROM table 1 [, table 2] [WHERE])

Self Join - A self join is a regular join, but the table is join with itself.

Syntax: SELECT column_name(s)
FROM table T1, table T2
WHERE condition;

Date __/__/____

Equi Join - It performs a JOIN against equality or matching column(s) values of the associated tables. An equal (=) is used as comparision operator in the WHERE clause to refer equality.

Syntax:

```
SELECT Column_list FROM table1, table2, ... WHERE table1.column_name = table2.column_name;
```

PL/SQL - PL/SQL stands for "Procedural Language ~~extentions~~ extensions to the structured Query Language." PL/SQL is Oracle corporation's procedural extension for SQL and the Oracle relational database. PL/SQL is available in Oracle Database.

* PL/SQL Procedure - A procedure is a PL/SQL block which performs one or more specific tasks.

The procedure contains a header and a body.

- Header: The header contains the name of the procedure and the parameters or variables passed to the procedure.

- Body: The body contains ~~a de the name of the procedure and the~~ a declaration section, execution section and exception section similar to a general PL/SQL block.

Page No.

Date __/__/____

Syntax for creating procedure:

```
CREATE [OR REPLACE] PROCEDURE
procedure_name [(Parameter [,Parameter])]
IS
   [declaration_section]
BEGIN
   executable_section
   [EXCEPTION exception_section]
END [Procedure_name];
```

PL/SQL Function – The PL/SQL Function is very similar to PL/SQL Procedure. The main difference between procedure and a function is, a function must always return a value, and on the other hand a procedure may or may not return a value. Except this, all the other things of PL/SQL procedure are true for PL/SQL function too.

Syntax to create a function:

```
CREATE [OR REPLACE] FUNCTION function_name
[Parameters] [(Parameter_name [IN | OUT | IN OUT]
  type [,...])]
RETURN return_datatype
{IS | AS}
BEGIN
     < function_body >
END [function_name];
```