Unit one Introduction

Data : information in raw or unorganized from (such as alphabet, number or symbol) that refer to or represent condition ideas or objects.

*data is limitless and present everywhere in the universe.

* data is any sort information which is stored in a computer memory

*in a database context data refers to all the single items are store in a database .

Information : Information is a group of data that collectively carries a logical meaning

*data that is specific and organized for a purpose.

Database: A database is a collection of related data

A database is a collection of related data

A database is a systamitic collection of data which support electronic storage and manipulation of data.

Database Management System : A DBMS is a software that allows creation, definition and manipulation of database .DBMS is also provide protection and security to database MySQL, SQLite, Oracle, IBM DB2 , file marker pro, Microsoft Access are example of DBMS

1.1 History, Database and its application

Telecome

Industry

Banking System

Education sector

Online Shopping

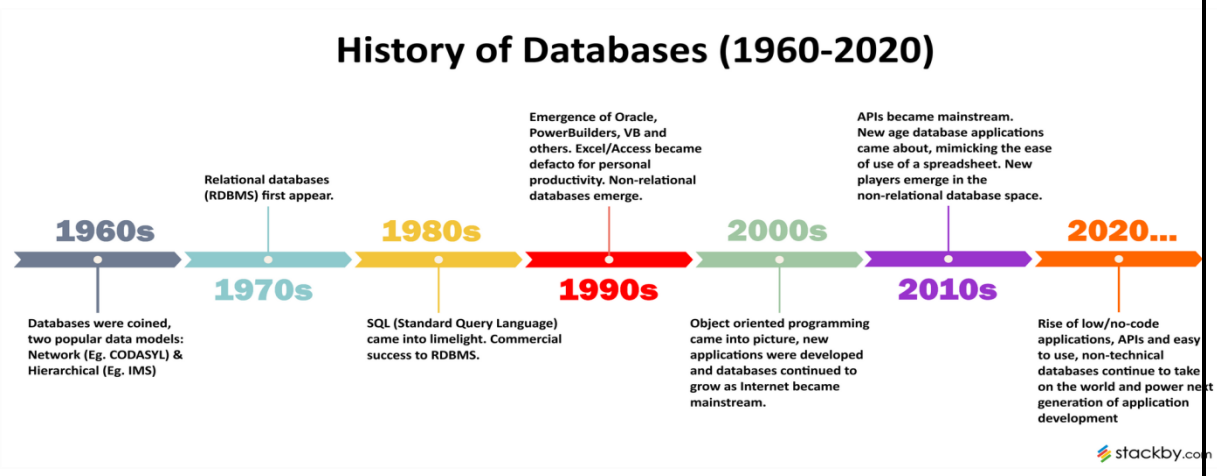
History of database

1960:- Traditional file system

1970:- Tree structure model

1970:- Network data model

1980:- relational data model



1.2 Characteristics of DBMS

**** Real-world entity :*** A modern DBMS is more realistic and use real world entities to design its architecture it use the behavior and attributes .

**** Relational based tables :*** DBMS allows entities and relation among them to form tables . A user can understand the architecture of a database just by looking at the table names.

**** less redundancy :*** DBMS follows the rules of normalization which splits a relation when any of its attributes is having redundancy in value

**** consistency :*** Consistency is a state where every relation in a database remain consistent. There exist method and technique, which can detect attempt of leaving database in inconsistent state.

**** Query language :*** DBMS is equipped with query language which makes it more efficient to retrieve and manipulate data. Traditionally it was not possible where file-processing system was use.

**** Multiuser and concurrent Access:*** DBMS offers multiple views for different users. A user who is the sales department will have a different view of database than a person working in the production department.

****Security:*** Features like multiple views offer security to some extent where users are unable to access data of other users and department.

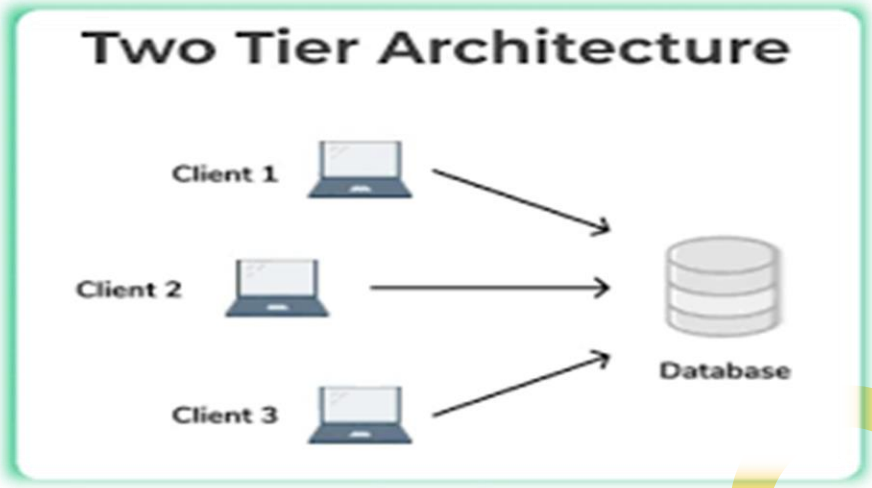
**** logical relationship between record and data :*** - A Data base give a logical relationship between its record and data . So a user can access various records depending upon the logical conditions by a single query from the database .

1.3 Architectures

:- Database architecture is representations of database design which helps to design developed and maintain the database management system.

-) Types of architecture

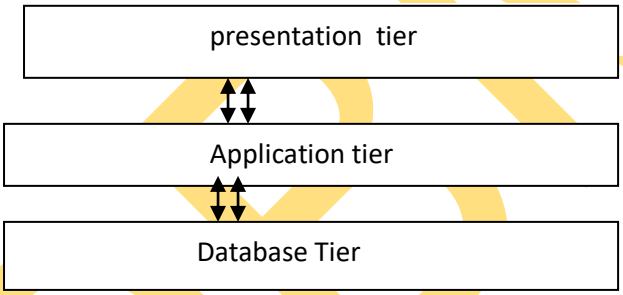
1 Two tier architecture :- In two tier architecture the direct communication takes place between client and server . There is no intermediate layer between client and server.



the main two tier of architecture is

data tier (data base)
Client tier (client application)

2 Three tier architecture :- Many web application use an architecture called the there-tier architecture which adds an intermediate layer between the client & the database server



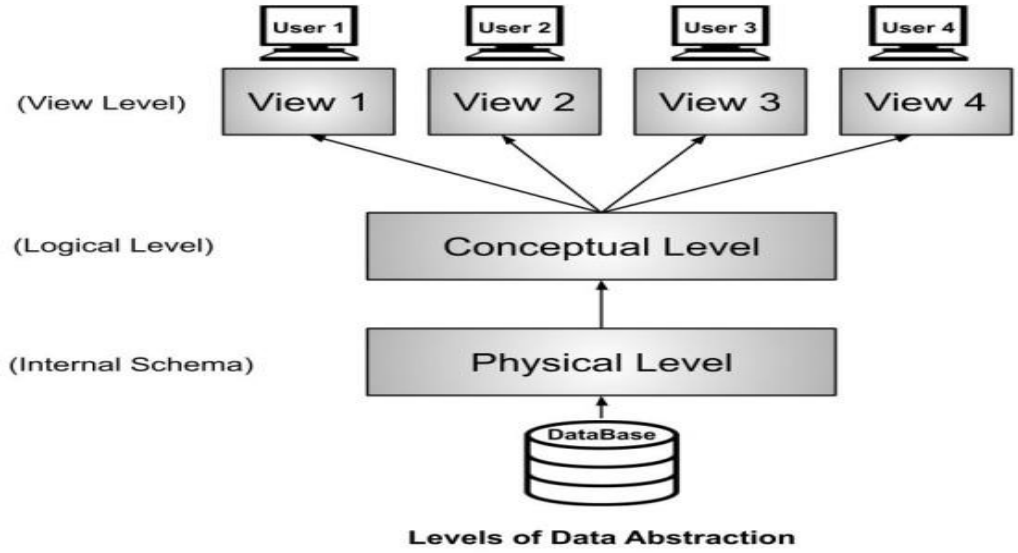
Presentation tier :- End users operate on this tier and they know nothing about the existence about database beyond this layer and it also provides user interface

Application Tier :- It is the intermediate layer which has the function for client layer (presentation layer) and it is used to make communication between client and database

Database Tier :- At this tier the database resides along with its query processing language normally database tier has the database

1.4 Data Abstraction and independence

**** Data Abstraction :*** DAta abstraction as in DBMS means hiding unnecenessary background details from the end users to make the accessing of data is and secure.



Levels of Data Abstraction

**** View level :-*** It is also called external level. This level represented only a part of the entire database which is internally **fetched** from database with the help of logical level and physical level mapping (addition).

**** Logical level ;-*** It is also called conceptual level. The level of abstraction describe what data are stored in the database and what relationships exits among those data it is the whole design of the database .

****Physical level:-*** It is also called internal level. The lowest level of abstraction describes how the data are actually stored. The physical level describes complex low level structure.

Data independence:- Data independence is a property of DBMS that help us to change the database schema at one level of database system without change the schema at next level.

Types data independence

Physical Data independence: It is the ability to change the internal schema without effecting conceptual or external schema of database Eg. using new storage device like hard disk

Logical independence :- it is the ability to change the conceptual schema to without effecting the external schema or application program

eg Addition of new data item in database

Removal or deletion or existing data items from database .

1.5 Schemas and Instances

Schema :- Schema is a logical representation of database .

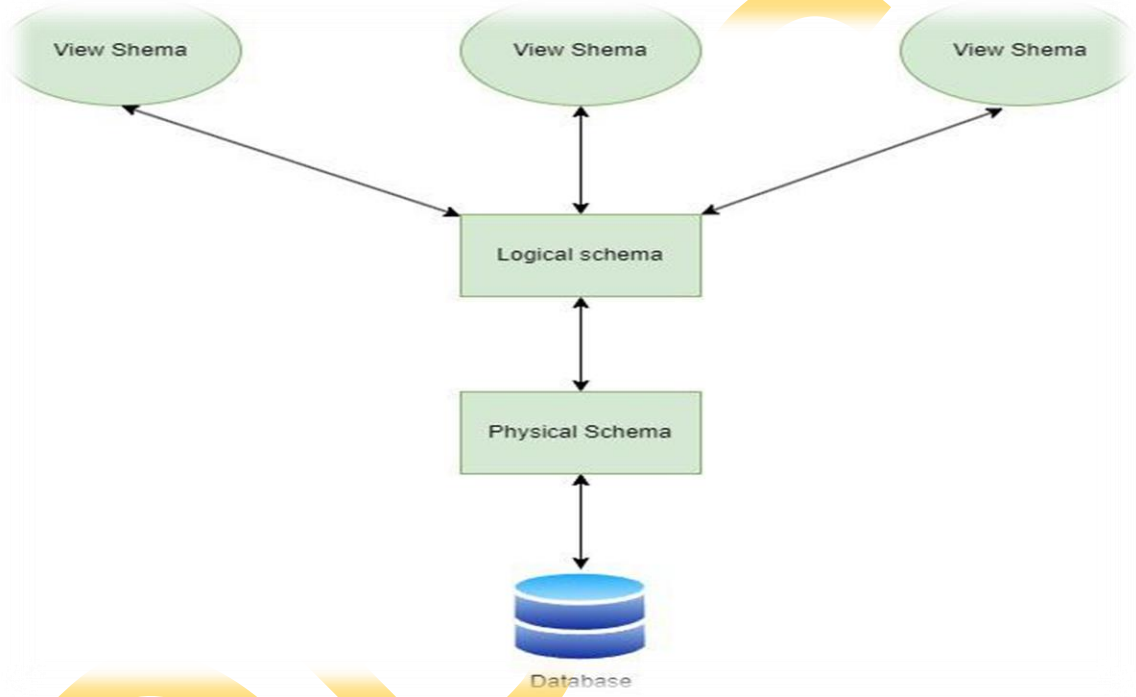
Refers to the organization of data how database is constructed and define its entities and the relationship among them

A database schema broadly divided into two categories :

logical DATABASE SCHEMA :- this schema define all the logical constraints that need to applied on the data store

-It defines tables, views etc

b> physical database schema :- this schema relates to the actual of the data and its form of data like file, image etc .It defines how the data will be stored in a secondary storage



***Instance** :- the data store in database at a particular moment of a time is called instance in short , at a particular moment the data store d in database known as instance that change overtime when we add or delete data from the database .

Example: in a student table , today the table has 100 records so today is the instance database has 100 records and tommarow we are going to add another 100 records is that table so, the instance of database tommarow will have 200 records in a table

1.6 Classification of DBMS

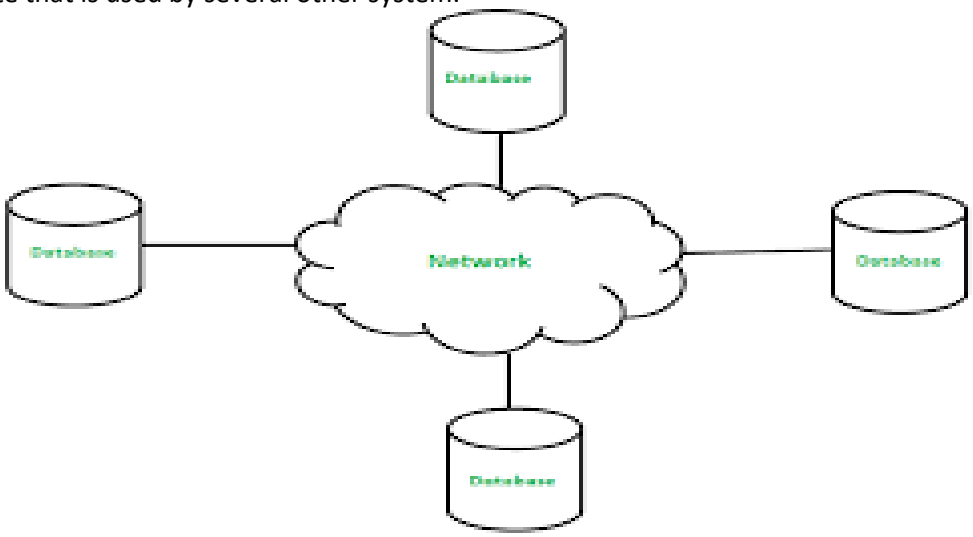
: classification of database management system can be classified based on several criteria area such as the data model, user number and database distributions all are describe below

1- classification based on data model:- the most popular data model in use today is the relational data model well known DBMS like oracle, mysql and ms sql support this model

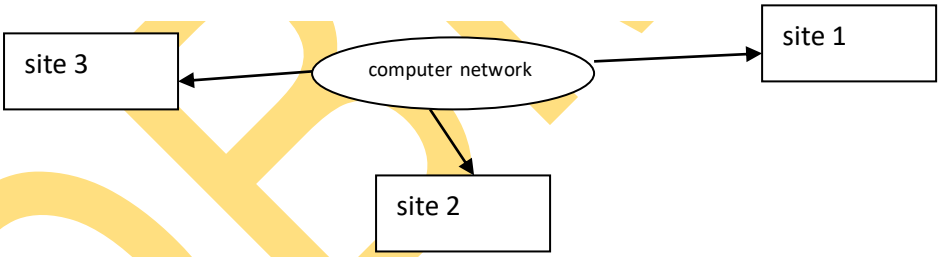
2- classification based on user number :- it can be a single user database system, which support multiple users concurrently .

3- classification based on database distribution :- there are main four distribution system

centralized system :- in centralized database system , the DBMS and database store at site that is used by several other system.



Distributed database system :- in distributed database system, the actual database and the DBMS software are distributed from various site that are connected by a computer network



Homogenous distribution database system:- Homogenous distributed database use the same DBMS software from multiple site which has the same operating system.

Hetrogenous distributed database system :- In distributed database system different site might use different DBMS software but there is additional common software but there is additional o common software to support data exchange between these sites

1.7 Introduction to DDL DML DCL

**** DDL :-*** it stands for data definition language which changes the structure of the table like creating table, deleting a table , altering a table commands

CREATE
DROP
ALTER
TRUNCATE

CREATE:- it is used to create a new database and table

Syntax : *syntax to create a database*

CREATE DATABASE data base name;

example : CREATE DATABASE school;

Syntax to create a new table

CREATE TABLE table name (column name1 data type[size],
column name1 data type[size],
column name1 data type[size],
);

```
CREATE TABLE GANESH(
    RN  into,
   Name  VARCHAR(20),
   Address VARCHAR(25),
            );
```

DROP :- it is used to delete both the structure and record store in table

```
syntax : DROP TABLE table name;
```

ALTER :- used to change the structure of table this change could be either to modify the existing attributes or probably to add a new attributes.

 # SYNTAX HAS TO ADD NEW COLUMN :- ALTER TABLE table name ADD column name data type(size);

example ALTER TABLE student ADD email varchar(30);

#SYNTAX TO DROP NEW ELEMENT :- ALTER TABLE table name DROP COLUMN columnname;

example :- ALTER TABLE student DROP COLUMN email;

#SYNTAX TO RENAME A COLUMN :- ALTER TABLE tablename RENAME COLUMN old name TO new name;

example :- ALTER TABLE student RENAME COLUMN R.N TO S.N;

TRUNCATE :- it is used to delete from the table without deleting the structure of table .

Syntax -TRUNCATE TABLE table name;

Example: - TRUNCATE TABLE student;

DML (DATA MANIPULATION LANGUAGE):- DML is used to modify the database by inserting, deleting and updating their data.

Command:-

INSERT
UPDATE
DELETE

Insert:- it is used to insert data into the row of a table

Syntax: - INSERT INTO table name (col1, col2, cool n) values (" value 1", "value 2" ..... "Valuen");

Example: - INSERT INTO student (RN, NAME, AGE, ADDRESS) VALUES ("1", "ram", "18", "patan");

UPDATE: - This command is used to update or modify the value of column in a table.

Syntax: UPDATE table name SET (columnname1='value1' ..... column name n='value n") where condition;

Example: - UPDATE student SET name =' sager ' where RN='1';

DELETE: - this command is used to remove one or more row from table.

Syntax: - DELETE FROM table name WHERE condition;

Example: - DELETE FROM student WHERE RN='1';

DCL (Data control language):- it controls user access in a database , related to the security issue.

- It allows to restricts the user from accessing a data in database .
- there are two command in database

a> ***GRANT :-*** It gives user access previlage (access right) to the database

Syntax :- GRANT previlage list ON table name TO user;

example GRANT INSERT ON employee TO RAM;

b> ***REVOKE :-*** It is used to cancel previously granted permission or restricts permission to the database

Syntax :- REVOKE previlagelist ON table name FROM user ;

example :- REVOKE INSERT ON employee FROM RAM;

UNIT 2 DATA MODEL

2.1 Introduction to ER model:

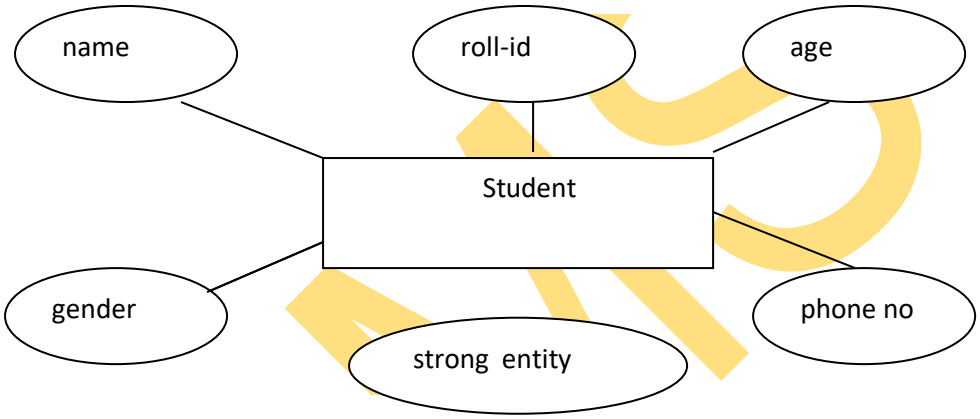
- ER-model for entity relationship model is a high level conceptual data model, which helps to systematically analyze data requirements to produce a well designed database.

2.2 Entities types : - An entity is a real world thing either living or non-living that is easily recognizable and non -recognizable

Egg. Person: - employee, student
Place: - store, library
Object: - car, machine
Concept: - account, course

Types of entities

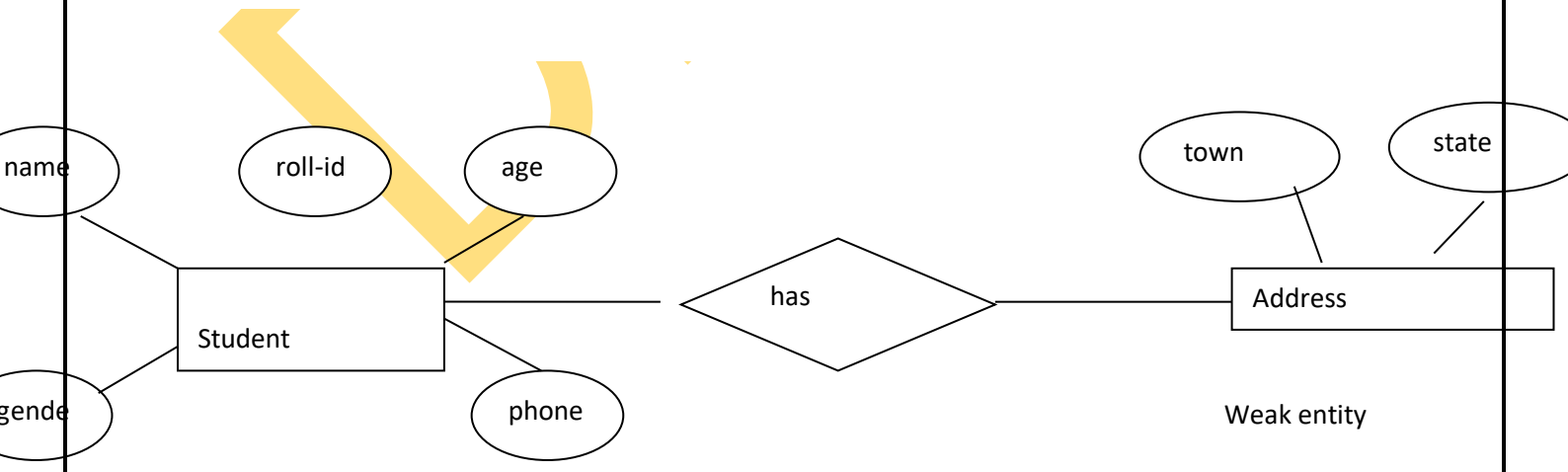
strong entity type :- strong entity are those entity types which has a key attribute the primary key helps in identifying each entity uniquely



weak Entity type :- weak entity type does not have a key attributes

-it can't be identified on its own

- depends upon some other strong entity for *i*ts identity



Strong entity

2.3 Entities set :- Entity set is a collection of entities of the same entity type

E1

E2

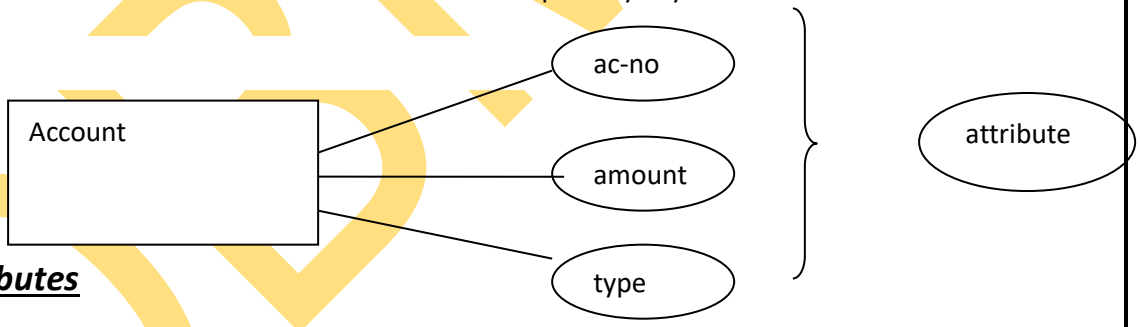
roll -no	student name	Age	Address
1	Gnash keypad	18	ganyapdhura
2	hari Prasad	16	ajayameru
3	keshav	17	amargadhi
4	khem	19	alital

E1 and E2 is a entity set

2.4 Attributes and keys

Attributes: - it is denoted by ellipse

An attribute is a property or characteristic of an entity. An entity may contain any number of attributes One of the attributes is considered as the primary key



types of attributes

Composite: - it is possible to break down composite attributes.

egg student's full name may be divided into first name ,last name.

simple attribute:- simple attributes can't be divided any further. example students contact number

Derived attributes :- This type of attribute does not present in the physical database however their values are derived from other attributes present in the database .

example Age should not be stored directly . it should be derived from the DOB.

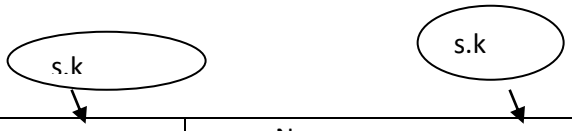
Multi-valued attributes:- Multi-valued attribute can have more than one values

eg student can have more than one mobile number

keys :- key in a DBMS is an attribute or set of attributes which helps to identify a tuple relation . It helps us to uniquely identify row in a table

Types of keys

super key:- A super key is a group of single or multiple keys which identifies rows in a table . A super key may have additional attributes that are not needed for unique identification



R.N	Name
1	LOK
2	GANESH KAPDI

in this R.N and Name are supper key

primary key :- primary key is a column or group of column in a table that uniquely identify every row in that table . the primary key field can not be null.

R.N	Name	Email
1	GANESH	ganeshkapadi@gmail.com
2	TEK	rajkumar@gmail.com

in this R.N is primary key which uniquely identify all the row in table.

Alternate key:- All the keys which are not primary key are called an alternate key.

A Table can have multiple choices for a primary key but only one can be set as the primary key.

candidate key :- candidate key is a super key with no repeated attributes

the primary key should be selected from the candidate key

every table must have at least a single candidate key.

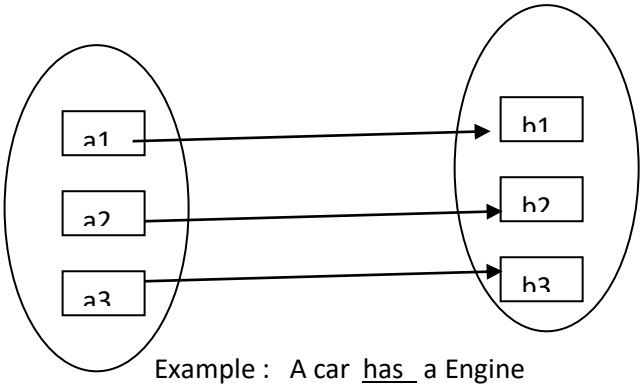
foreign key:- foreign key is a column tha creates a relationship between two tables

it acts as a cross -reference between two tables as it reference the primary keys of an table.

2.5 Relationship and its type :- Relationship or mapping cardinality express the number of entities to which another entity can be associated via relationship

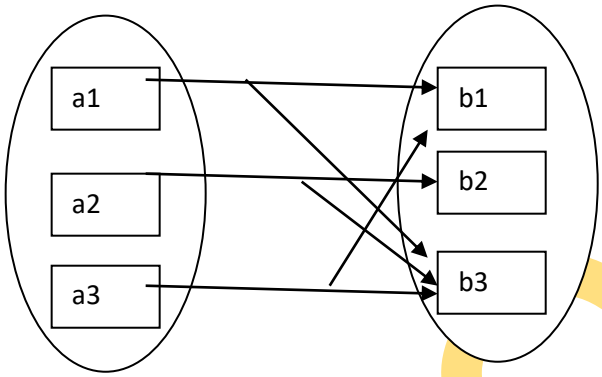
Types of relationship

one to one (1:1) :- an entity in A is associated with atmost one entity in B and an entity in B is also associated with atmost one entity in A is called one to one relationship .



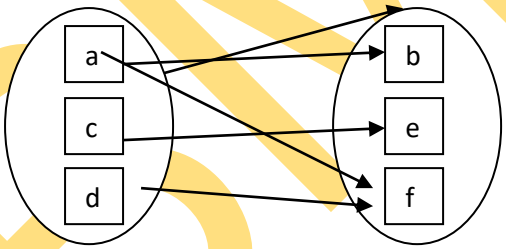
Example : A car has a Engine

one to many (1:m) :- An entity n A is associated with any number (Zero one more) of entity in B & an entity in B however can be associated with atmost one entity one entity in A is called one to many relationship >



example :- A father has a many Many children

Many to many(M:M) :- an entity in A is associated with any number zero or more of entities in B & an entity in B is however associated with any number (Zero or more) of entity in A is called many to many (M:M) relationship .



example : student s Choice many course

2.6 E-R diagram :-

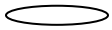
stands for entity relationship diagram also known as ERD, which displays the relationship of entity sets stored in a database .

ER diagram created based on three basic concept (entities , attributes and relatiosnship)

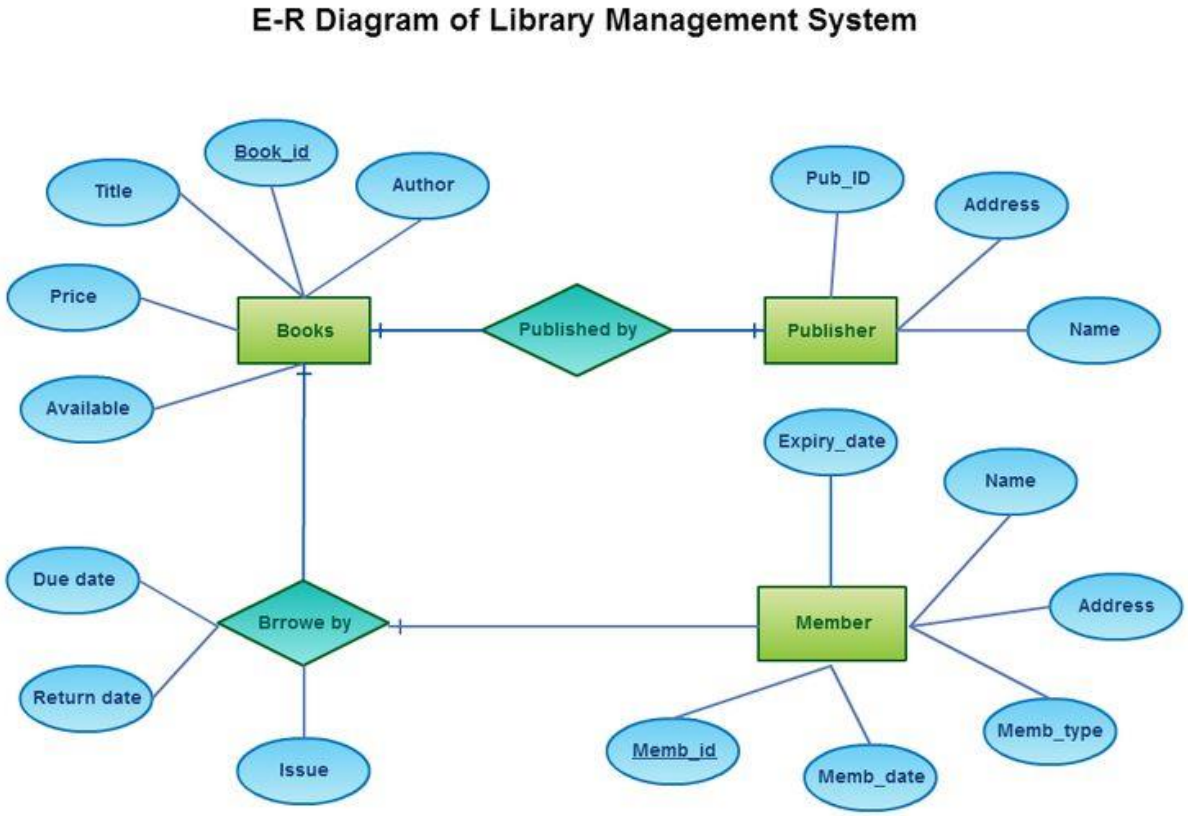
It contains different symbols that use rectangle to entity, ovals to define attributes and diamond to represent relationship.

Symbols used in ER diagram

S.N	Name	work	symbol
1	rectangle	entity	
2	ellipse	Attributes	
3	diamond	relationship	
4	lines	Link attributes to entity set	
5	double ellipse	multivalve attribute	

6	dashed ellipse	derived attribute	

example



Unit 3 NORMALIZATION

Normalization is a process of organizing the data in database to avoid redundancy (duplicate)

it is the process of minimizing redundancy from a relation .

3.1 Importance of normalization :- The main objective of database normalization to emiminate , minimize data modification error and simplified query process.

correct duplicate data and anomalies(error) :- when we apply normalization method to the company database we can identify and correct duplicate information quickly so if doesn't affect the rest of the database . normalization method can also help identify anomalies because it allows them to remain at the top or buttom of our database

Remove unwanted data connection :- data normalization method help remove data connection that don't relate to the main data at the organization where employeed ,this method can seprate data into different table that are no longer connected each other

Prevents data deletion:- using well established data normalization to method can help prevents us from deleting data that relates to the main key data .

Optimizes data store space :- data normalization can help remove unnecessary data from database . for example :- normalization method may reduce duplicate data in database . Remove large gigabyte of redundant data can allow us to have more storage for future data

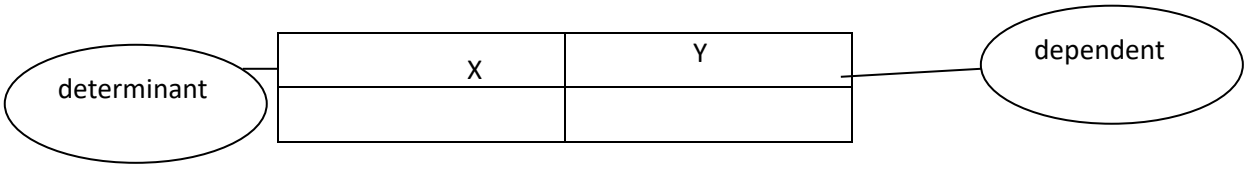
Increase data consistency:- it can help the reliability of the information in a database. It remove duplicate sets of data in different department and prevents outdate dataset from over writing accurate information .

3.2 functional dependence:-

functional dependence define how the attribute relation (table) are related to each other

it is denoted by an arrow (->) sign, x-> y where x functionally determine the y

if a given relation R with attributes 'X' and 'Y' then y is said to be functionally dependent on x. if given value of x uniquely determinant value of the attribute on y



Types of functional dependency

Trivial functional dependency :-

in trivial functional dependency a dependent is always subset of determinant .

a functional dependency is called trivial if the attributes on the right side are the subset of the attribute on the left side on the functional dependency .

example :- student

R.N	Name	Age
1	Ganesh	19
2	Ram	28
3	Arjun	30

here, [R.N, Name] -> [Name]
[Name]-> [Name]
[Age]-> [Age]

Non-trivial functional dependency :- it is the opposite of trivial functional dependency . in non trivial functional dependent is not of subset of determinant .

x->Y is called non-trivial functional dependency if y is not a subset of determinant .

Employee

E.ID	Name	Address
1	Bhagirath kunwar	Baitadi
2	Yogendra kunwar	doti
3	Ganesh kapadi	Dadeldhura

here , [E.ID]-> [Address]
[E.ID] -> [Name]

:- * ***Transitive functional dependency*** in transitive functional dependency , dependent is indirectly dependent on determinant .

if a->b and b->c , then here c indirectly dependent on a .

eg

S.N.0	NAME	SUBJECT NAME
1	HARI PRASAD KAPADI	GEOMETRIC ENGINEERING
2	KHEM RAJ KAPADI	HEALTH
3	GANESH KAPADI	DCOM

here, S.N-> NAME
NAME->SUBJECTNAME
hence according to transitive dependency

S.N->SUBJECT NAME is a valid functional dependency

3.3 Normal form (1NF , 2NF , 3NF, BCNF)

* FIRST NORMAL FORM (1NF):-

the relation is first normal form if it does not contain any multivalue attribute.

- A relation is in first normal form if every attribute.

STUID	STUNAME	STUPHONE	CITY
1	KESHAV JOSHI	9811543434 9863636362	DOTI NEPAL
2	GANESH KAPADI	9840047374	DADELDHURA NEPAL
3	UMESH BHAT	9811685628	BAITADI

AFTER NORMALIZATION

STUID	STUNAME	STUPHONE	CITY
1	KESHAV JOSHI	9811543434	DOTI NEPAL
1	KESHAV JOSHI	9863636362	DOTI NEPAL
2	GANESH KAPADI	9840047374	DADELDHURA NEPAL
3	UMESH BHAT	9811685628	BAITADI

* SECOND NORMAL FORM (2NF):-

Before we learn about second normal form, we need to understand to the following

a) prime attribute:-an attribute which is a part of primary key is known prime attribute

b) non-prime attribute :- an attribute which is not part of primary key is known as non prime attributes.

Rule :- table should be 1 NF

there should no any partial dependency .

example

Students

IDSt	LastName	IDProf	Prof	Grade
1	Mueller	3	Schmid	5
2	Meier	2	Borner	4
3	Tobler	1	Bernasconi	6

Startsituation

Result after normalisation

Students

ID	LastName
1	Mueller
2	Meier
3	Tobler

Professors

IDProf	Professor
1	Bernasconi
2	Borner
3	Schmid

Grades

IDStIDProf	Grade	
1	3	5
2	2	4
3	1	6

*THIRD NORMAL FORM (3NF) :- A relation is an 3NF where it is an 2NF and there is no transitive dependency

Rule :- table should be 2nf

No transitive dependency

Vendor

ID	Name	Account_No	Bank_Code_No	Bank

Startsituation

Result after normalisation

Vendor

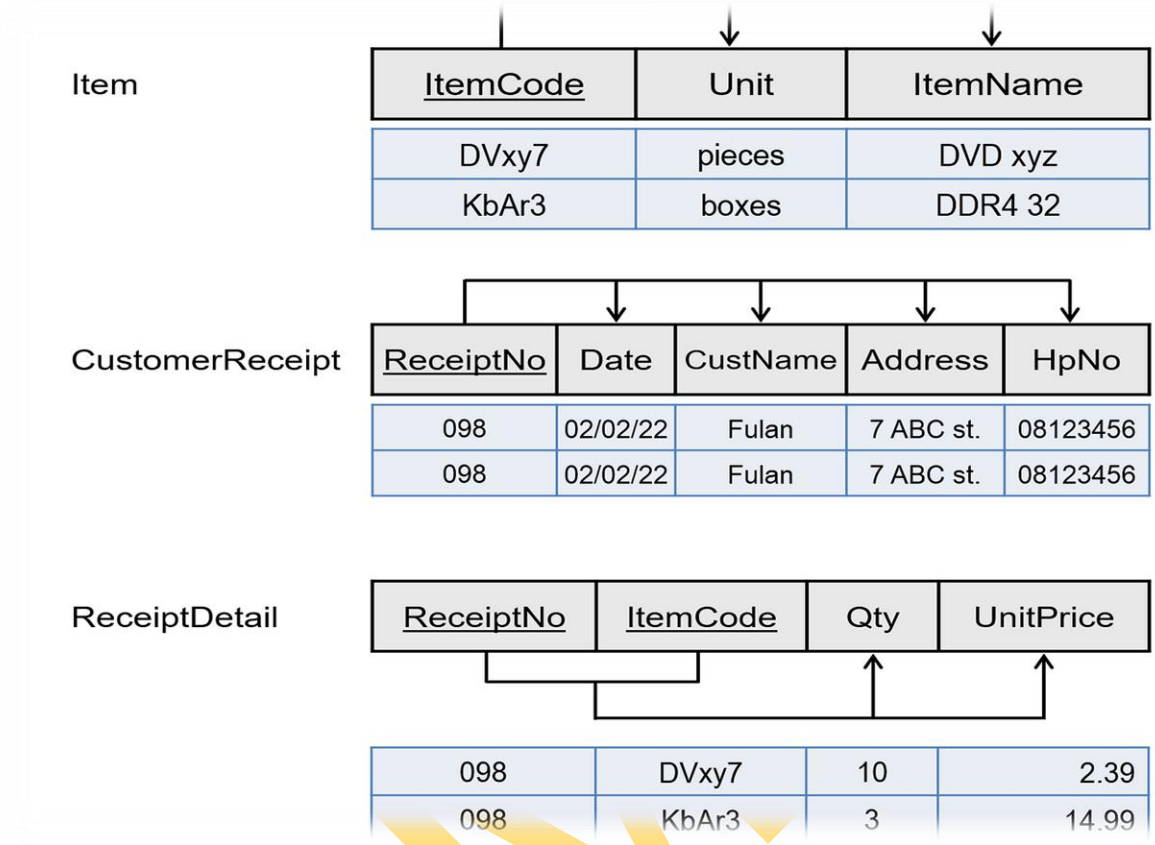
ID	Name	Account_No	Bank_Code_No

Bank

Bank_Code_No	Bank

* *BCNF(BOYCE CODD NORMAL FORM) :-* it is an advance version of 3NF which was introduced by RF Boyce and EF CODD IN 1970

- BCNF IS also known as 3.5 NF , a table is in BCNF if is determinant is a candidate key



Item

ItemCode	Unit	ItemName
DVxy7	pieces	DVD xyz
KbAr3	boxes	DDR4 32

CustomerReceipt

ReceiptNo	Date	CustName	Address	HpNo
098	02/02/22	Fulan	7 ABC st.	08123456
098	02/02/22	Fulan	7 ABC st.	08123456

ReceiptDetail

ReceiptNo	ItemCode	Qty	UnitPrice
098	DVxy7	10	2.39
098	KbAr3	3	14.99

3.4 Integrity and domain constraints :_ integrity constraints are a set of rules. it is used to maintain the quality of information .

integrity constraints ensure that the data insertion, updating, and other processes have to be performed in such a way that data integrity is not affected.

Types of Integrity Constraint

Domain constraint:- domain constraints can be defined as the definition of a valid set of values for an attribute.

The data type of domain include string , character , integer, time , date , currency, etc , the value of the attribute must be available in the corresponding domain

Entity integrity constraints :- The entity integrity constraint states that primary key value can't be null.

this is because the primary key value is used to identify individual rows in relation and if the primary key has a null value, then we can't identify those rows.

Referential Integrity constraints :- a referential integrity constraint is specified between two table. in the referential integrity constraints, if a foreign key in table 1 refers to the primary key of table 2 then every value of the foreign key in table 1 must be null or be available in table 2.

key constraints :- keys are the entity set that is used to identify an entity within its entity set uniquely. An entity set can have multiple keys, but ou of which one key will be the primary key. A primary key can contain a unique and null value in the relational table.

Unit 4 RELATIONAL LANGUAGE

4.1 INTRODUCTION TO SQL:- it stands for structure query language

used for storing and managing data in relational database management system (R-DBMS)

Enable users to create ,read , update,& delete relational database &table

SQL allows users to query , the database in a number of ways using English like statement.

4.2FEATURES OF SQL :-

FLEXIBILITY AND SCALABILITY :- with SQL it is easier to create new tables while dropping or deleting previously created table .

High Availability :- SQL is compatible with other database such as Microsoft SQL server, oracle, database msaccess & mores

HIGH SECURITY :- SQL also has high security. it is easy to give permissions on views at a table. so with SQL we get maximum security for our data .

open source :- SQL has the features of being open source programming language where everyone can use the SQL statement on the web.

EASY TO LEARN:- SQL Is easy to learn because its statement are written in English like statement.

4.3 BASIC RETRIEVEL QUERIES :- SQL has basic statement for retrieving information from a database the select statement.

- Basic form of the SQL select statement is called a mapping or a SELECT _FROM - WHERE block
- SELECT <attribute list>
- FROM <Table list / name>
- WHERE <condition >

EXAMPLE :-

CUSTOMER TABLE AS SHOW BELOW

CNO	LAST NAME	FIRST NAME
2121	KAPADI	GANESH
2156	JOSHI	BHUWAN
2157	AWASTHI	GAUTAM

TO SELECT ALL THE COLUMNS FROM A RELATION

SYNTAX :- SELECT* FROM table name;

ANS :- SELECT * FROM customer;

TO SELECT PARTICULAR COLUMN FROM THE RELATION

SYNTAX SELECT col1, col2 ….. , FROM table name;

ANS ;- SELECT CNO, FIRSTNAME FROM customer;

TO SELECT ALL THE RECORD FROM RELATION USING WHERE CLAUSE

SYNTAX :- SELECT * FROM table name WHERE condition;

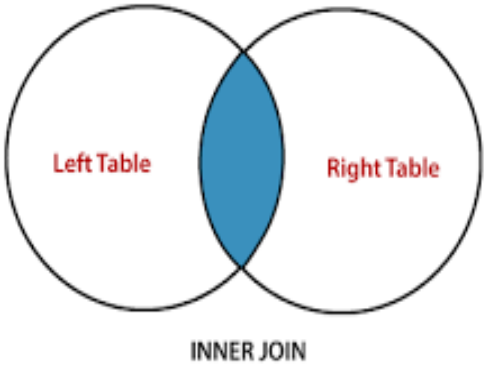
ANS :- SELECT * FROM customer WHERE CNO=2039;

JOIN:- SQL join use to fetch data from two or more table which joined to appear as single set of data it is command clause that combines record from two or more lables in a

database it is means of combining data in fields from two tables by using values common to each other TYPES OF JOIN:

INNER JOIN (EQUI JOIN):- Inner join combines two tables based on a shared key

inner join combine multiple tables by retrieving record that have matching values in both



INNER JOIN

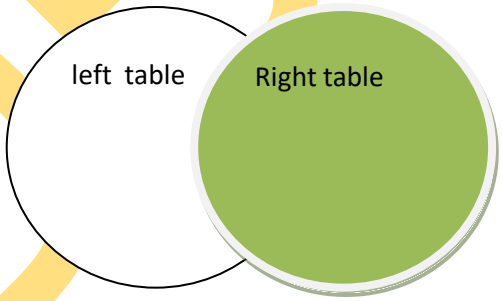
tables

SYNTAX :- SELECT column name FROM table name1 INNER JOIN table 2 ON table1.column name = table2 . column name;

OUTER JOIN :- outer join is a method of combining two or more tables so that the result includes unmatched rows of one of tables or both the tables

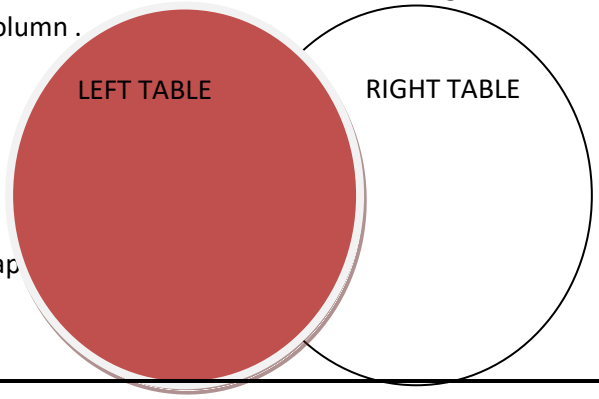
TYPES OF OUTER JOIN

RIGHT OUTER JOIN :- The right outer join returns a results table with the matched data of two tables then remaining rows of the right table and null for the left table column .



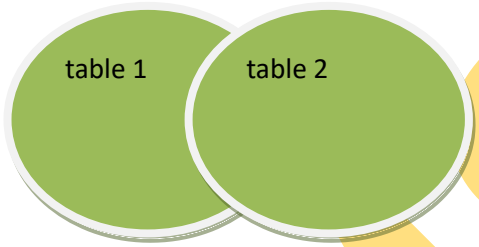
SYNTAX :- SELECT column name FROM table 1 RIGHT OUTER JOIN table 2 ON table 1. column name = table 2 column name;

LEFT OUTER JOIN :- The left outer join combines multiple tables by retrieving records that have matching values in the both table then remaining rows of the left table and NULL for the right tables column .



SYNTAX:- SELECT column names FROM table 1 LEFT OUTER JOIN table 2 ON table1.column name =table2.column name;

FULL OUTER JOIN :- The full outer join combines multiple tables by retrieving records that have matching values in the both table and null for the unmatched values in both value



SYNTAX:- SELECT column names FROM table 1 FULLOUTER JOIN table 2 ON table 1 . column name = table 2 . column name

SEMI JOIN :- it is a Inner join semi join matches the rows of two relation and then shows the matching rows of the relation whose name is mentioned to the left side of semi join operator

TEACHER

TID	NAME	SALARY
103	GANESH KAPADI	1000
204	LOKESH BIST	10000

STUDENT

SID	ROLL NO	MARKS
103	3	80
104	2	90

SYNTAX :- SELECT column FROM table 1 where EXISTS (SELECT VALUES FROM table 2 WHERE table 2 column = table 1 .column) ; the query will be

SELECT TEACHER.TID, TEACHER.NAME FROM TEACHER WHERE EXISTS (SELECT 103 FROM STUDENT WHERE STUDENT.SID = TEACHER . TID);

RESULTS

TID	NAME	SID
103	GANESH KAPADI	103

SUB QUERIES :- In SQL sub query can be simply defined as a query with in another query. In another words we can say that a sub query is a query that is embedded in WHERE clause of another SQL query

SYNTAX :- there is not any general syntax for sub queries . However sub queries are seen to be used most frequently with SELECT statement as shown below

SELECT column name FROM table name WHERE column name expression operator (SELECT column name FROM table name WHERE ……..); EXAMPLE :- CUSTOMER TABLE

ID	NAME	AGE	ADDRESS	SALARY
1	GANESH	23	DARCHULA	200
2	TEK RAJ	26	JHAPA	30000
3	JAYA RAJ	31	BAGLUNG	16000
4	DURAGA	30	DADELDHRUA	40000
5	KHEM RAJ	29	BAITADI	50000
6	BISHNU RAJ	24	ACCHAM	60000

TO SHOW THE SALARY GREATER THAN 40000

ANS :- SELECT * FROM CUSTOMER WHERE ID IN (SELECT ID FROM CUSTOMER WHERE SALARY >40000

4.6.VIEWS:- A views is a virtual table based on the result set of and SQL statement.

view contain rows and column just like real table

a view always shows up-to-date data. The database engine recreates the view every time a users quires it

SYNTAX :- CREATE VIEW VIEWNAME AS SELECT COL1, COL2, ……. FROM table name WHERE condition ;

EXAMPLE :- STUDENT TABLE

ID	NAME	AGE	ADDRESS
1	BHUWAN JOSHI	18	PATAN BAITADI
2	GANESH KAPADI	19	GANYAPDHURA DADELDHURA
3	GAUTAM AWASTHI	19	PATAN BAITADI

ANS:- CREATE VIEW GANYAPDHURA DADELDHURA STUDENT AS SELECT NAME AGE FROM STUDENT WHERE ADDRESS = GANYAPDHURA DADELDHURA ;

4.7 RELATIONAL ALGEBRA :-

Relational algebra is procedural query language , which takes relation as input and generate relation as output

it provides theoretical foundation of relational database and SQL

4.7.1 SELECT PROJECT :- IT HAS TWO CATIGORY

No.	Category	Selection	Projection
1.	Other Names	The selection operation is also known as horizontal partitioning.	The Project operation is also known as vertical partitioning.
2.	Use	It is used to choose the subset of tuples from the relation that satisfies the	It is used to select certain required attributes, while

No.	Category	Selection	Projection
		given condition mentioned in the syntax of selection.	discarding other attributes.
3.	Partitioning	It partitions the table horizontally.	It partitions the table vertically.
4.	Which used first	The selection operation is performed before projection (if they are to be used together).	The projection operation is performed after selection (if they are to be used together).
5.	Operator Used	Select operator is used in Selection Operation.	Project operator is used in Projection Operation.
6.	Operator Symbol	Select operator is denoted by Sigma symbol.	Project operator is denoted by Pi symbol.
7.	Commutative	Selection is commutative.	Projection is not commutative.
8.	Column Selection	Select is used to select all columns of a specific tuple.	Project is used to select specific columns.
9.	SQL Statements used	SELECT, FROM, WHERE	SELECT, FROM

4.7.2CARTEIAN PRODUCT :- Cross product done between two relation . It takes to relation as input and gives one relation the cross product as the output.

if there are x attributes in first relation and y in second relation then the final relation will have X+Y and the number of tupple in final relation will be X*Y

NAME	AGE	GENDER
GANESH	19	MALE
BHUWAN	17	MALE

ID	COURSE
1	DS
2	DBMS

RESULT :-

created by Ganesh Prasad kapadi

NAME	AGE	GENDER	ID	COURSE
GANESH	19	MALE	1	DS
BHUWAN	17	MALE	2	DBMS
GANESH	19	MALE	1	DS
BHUWAN	17	MALE	2	DBMS

4.7.3 SET DIFFERENCE :- The result of set difference operation is tuples, which are present in one relation but are notation : r-s

finds all the tuples that are present in r but not in s

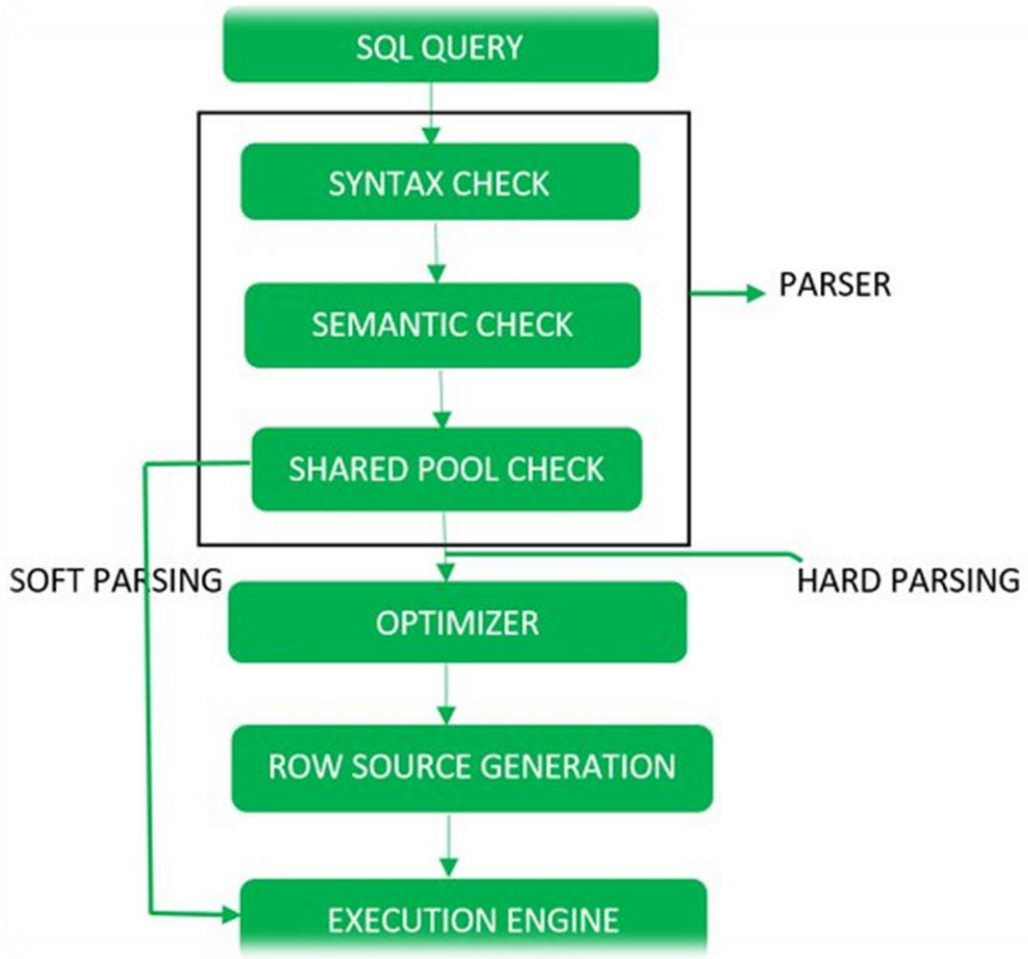
Example

$\prod$ author (books) -> $\prod$ author(article)

o/p :- provide the name of author who have written books not article

Unit 5 Query processing

5.1 Introduction to query processing :- Query processing includes translations on high level queries into low level expression that can be used at physical level of system .



- **Parser:** During parse call, the database performs the following checks- Syntax check, Semantic check and Shared pool check, after converting the query into relational algebra.Parser performs the following checks as (refer detailed diagram):
  - **Syntax check –** concludes SQL syntactic validity. Example: SELECT * FORM employee Here error of wrong spelling of FROM is given by this check.
  - **Semantic check –** determines whether the statement is meaningful or not. Example: query contains a table name which does not exist is checked by this check.
  - **Shared Pool check –** Every query possess a hash code during its execution. So, this check determines existence of written hash code in shared pool if code exists in shared pool then database will not take additional steps for optimization and execution.
  - **Hard Parse and Soft Parse –**
    If there is a fresh query and its hash code does not exist in shared pool then that query has to pass through from the additional steps known as hard parsing otherwise if hash code exists then query does not passes through additional steps. It just passes directly to execution engine (refer detailed diagram). This is known as soft parsing.
    Hard Parse includes following steps – Optimizer and Row source generation.

Step-2:

- **Optimizer:** During optimization stage, database must perform a hard parse at least for one unique DML statement and perform optimization during this parse. This database never optimizes DDL unless it includes a DML component such as sub query that require optimization. It is a process in which multiple query execution plan for satisfying a query are examined and most efficient query plan is satisfied for execution.
- **Row Source Generation:** The Row Source Generation is a software that receives a optimal execution plan from the optimizer and produces an iterative execution plan that is usable by the rest of the database. The iterative plan is the binary program that when executes by the SQL engine produces the result set.

Step-3:

Execution Engine: Finally runs the query and display the required result.

5.2 QUERY COST ESTIMATION :-

The query optimizer choose a query plan based on cost estimates. The cost estimation of a query evaluation plan is calculated in term of various resource that include

- no of disk accesses
- Execution time taken by CPU to execute a query
- no of tables rows
- communication costs

- to estimate the cost of a query evaluation plan ,we use the no. of blocks transferred from the disk, and the no of disk seeks.

DISK SEEK:- Seek time for a hard drive used to read/write data.
Relational latency :- Time required to position a specific under the read/write head

5.3 QUERY OPERATION , OPERATOR TREE

A Query Tree is a data structure used for the internal representation of a query in RDBMS. It is also known as the Query Evaluation/Execution Tree. The leaf nodes of the query tree represent the relations, and the internal nodes are the relational algebra operators like SELECT (**σ**), JOIN (⋈), etc. The root node gives the output of the query on execution.

Steps to Make a Query Tree

Step 1:

Execute the leaf nodes with their corresponding internal nodes having the relational algebra operator with the specified conditions to get the resulting tuples that we use for the execution of the next operation.

Step 2:

This process continues until we reach the root node, where we PROJECT (**π**) the required tuples as the output based on the given conditions.

Let's understand this using some examples:

Example 1:

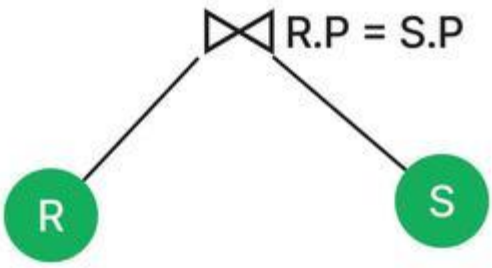
Consider a relational algebra expression –

$\pi_p (R \bowtie_{R.P = S.P} S)$

Step 1: Write the relations you want to execute as the tree's Leaf nodes. Here R and S are the relations.

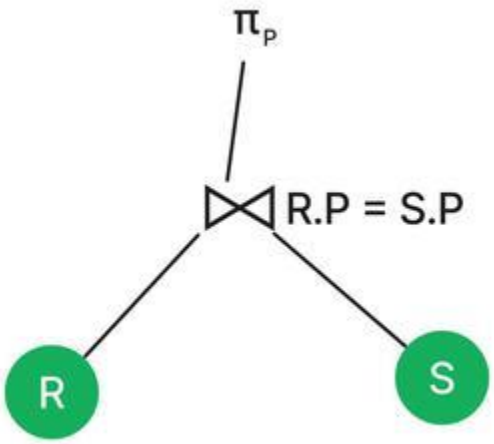
Two Relations R and S

Step 2: Add the condition (here R.P = S.P) with the relational algebra operator as an internal node (or parent node of these two leaf nodes).



OIN R and S where R.P = S.P

Step 3: Now add the root node that on execution gives the output of the query.



Execution Output

.

5.4 Evaluation of Expressions :-

For evaluating an expression that carries multiple operations in it, we can perform the computation of each operation one by one.

In the query processing system, we use two methods for evaluating an expression carrying multiple operations. These methods are:

1. Materialization
2. Pipelining

Materialization

In this method, the given expression evaluates one relational operation at a time. Also, each operation is evaluated in an appropriate sequence or order. After evaluating all the operations, the outputs are materialized in a temporary relation for their subsequent uses. It leads the materialization method to a disadvantage. The disadvantage is that it needs to construct those temporary relations for materializing the results of the evaluated operations, respectively. These temporary relations are written on the disks unless they are small in size.

Pipelining

-> Pipelining is an alternate method or approach to the materialization method.
-> In pipelining, it enables us to evaluate each relational operation of the expression simultaneously in a pipeline.
In this approach, after evaluating one operation, its output is passed on to the next operation, and the chain continues till all the relational operations are evaluated thoroughly. Thus, there is no requirement of storing a temporary relation in pipelining. Such an advantage of pipelining makes it a better approach as compared to the approach used in the materialization method. Even the costs of both approaches can have subsequent differences in-between. But, both approaches perform the best role in different cases. Thus, both ways are feasible at their place.

5.5 QUERY OPTIMIZATION

-> the process of selecting an efficient execution plan for processing a query is known as query optimization .
-> Query optimization is used to access and modify the database in the most efficient way possible
-> query optimization is formally described as the process of transforming a query into an equivalent form.

5.6 PERFORMANCE TUNNIG :-

-> Database performance tuning refers to the various ways database administrators can ensure databases are running as efficiently as possible.
-> Typically, this refers to tuning SQL Server or Oracle queries for enhanced performance.
-> The goal of database tuning is to reconfigure the operating systems according to how they're best used, including deploying clusters, and working toward optimal database performance to support system function and end-user experience.

-> Database performance tuning works initially by using historical data to establish a performance baseline. Baseline data should include:

- Application statistics (transaction volumes, response time)

- Database statistics
- Operating system statistics
- Disk I/O statistics
- Network statistics

Unit 6 TRANSACTION AND CONCURRENCY CONTROL

6.1 INTRODUCTION TO TRANSCATION :-

-> A transaction is a logical unit of a work which accesses and possibly modifies the content of a database.

-> The operation performed in transaction include one or more of database operation like insert, update , delete , or retrieve data .

-> Each high level operation can be divided into a number of low level of tasks or operation for example:- A data update operation can be divided into three task .

a> Read items() :- Read data item from storage to main memory.
b>modify item() :- change value of item in the main memory
c>write item():- write the modified value from main memory to storage

6.2 SERIALIZABILITY CONCEPT :-

In this article, we are going to explain the serializability concept and how this concept affects the DBMS deeply, we also understand the concept of serializability with some examples, and we will finally conclude this topic with an example of the importance of serializability. The DBMS form is the foundation of the most modern applications, and when we design the form properly, it provides high-performance and relative storage solutions to our application.

Example:

Transaction-1	Transaction-2
R(a)	
W(a)	
	R(b)

Transaction-1	Transaction-2
	W(b)
R(b)	
	R(a)
W(b)	
	W(a)

We can observe that Transaction-2 begins its execution before Transaction-1 is finished, and they are both working on the same data, i.e., “a” and “b”, interchangeably. Where “R”-Read, “W”-Writing .

Types of Serializability

There are two Types of serializablility

1. Conflict serializability

Conflict serializability refers to a subset of serializability that focuses on maintaining the consistency of a database while ensuring that identical data items are executed in an order. Here are a few of them.

1. Different transactions should be used for the two procedures.
2. The identical data item should be present in both transactions.
3. Between the two operations, there should be at least one write operation.

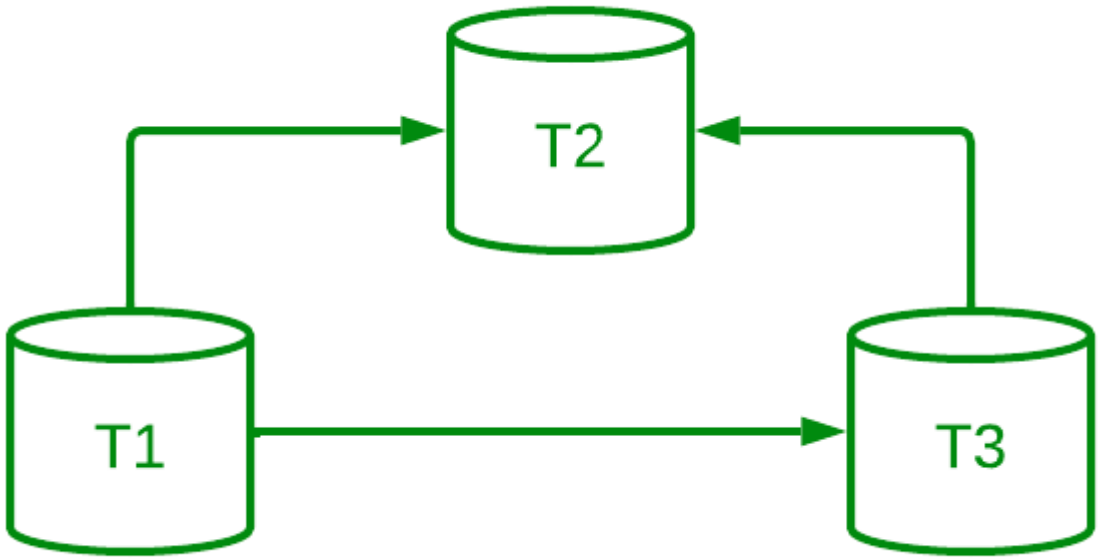
Example

Three transactions—t1, t2, and t3—are active on a schedule “S” at once. Let’s create a graph of precedence.

Transaction – 1 (t1)	Transaction – 2 (t2)	Transaction – 3 (t3)
R(a)		
	R(b)	
		R(b)
	W(b)	
W(a)		

Transaction – 1 (t1)	Transaction – 2 (t2)	Transaction – 3 (t3)
		W(a)
	R(a)	
	W(a)	

It is a conflict serializable schedule as well as a serial schedule because the graph (a DAG) has no loops. We can also determine the order of transactions because it is a serial schedule.



DAG of transactions t1 –> t3 –> t2

2. View Serializability

-> A schedule will be view serializable if it is view equivalent to a serial schedule.

6.3 CONCURRENT EXECUTION :-

Concurrency control concept comes under the **Transaction in database management system** (DBMS). It is a **procedure in DBMS** which helps us for the management of two simultaneous processes to execute without conflicts between each other, these conflicts occur in multi user systems.

Concurrency can simply be said to be executing multiple transactions at a time. It is required to increase time efficiency. If many transactions try to access the same data, then inconsistency arises. Concurrency control required to maintain consistency data.

For example, if we take ATM machines and do not use concurrency, multiple persons cannot draw money at a time in different places. This is where we need concurrency.

Advantages

The advantages of concurrency control are as follows −

- Waiting time will be decreased.
- Response time will decrease.
- Resource utilization will increase.
- System performance & Efficiency is increased.

6.4 LOCK BASED CONCURRENCY PROTOCOL

In this type of protocol, any transaction cannot read or write data until it acquires an appropriate lock on it. There are two types of lock:

1. Shared lock:

- It is also known as a Read-only lock. In a shared lock, the data item can only read by the transaction.
- It can be shared between the transactions because when the transaction holds a lock, then it can't update the data on the data item.

2. Exclusive lock:

- In the exclusive lock, the data item can be both reads as well as written by the transaction.
- This lock is exclusive, and in this lock, multiple transactions do not modify the same data simultaneously.

6.5 2PL AND STRICT 2PL :-

S.No.	Conservative 2-PL	Strict 2-PL

S.No.	Conservative 2-PL	Strict 2-PL
1.	In Conservative 2-PL, A transaction has to acquire locks on all the data items it requires before the transaction begins it execution.	In Strict 2-PL, A transaction can acquire locks on data items whenever it requires (only in growing phase) during its execution.
2.	It does not have growing phase.	It has growing phase.
3.	It has shrinking phase.	It has partial shrinking phase.
4.	It ensures that the schedule generated would be Serializable and Deadlock-Free.	It ensures that the schedule generated would be Serializable, Recoverable and Cascadeless.
5.	It does not ensures Recoverable and Cascadeless schedule.	It does not ensures Deadlock-Free schedule.
6.	It does not ensure Strict Schedule.	It ensures that the schedule generated would be Strict.
7.	It is less popular as compared to Strict 2-PL.	It is the most popular variation of 2-PL.
8.	It is not used in practise.	It is the most popular variation of 2-PL.
9.	In Conservative 2-PL, a transaction can read a value of uncommitted transaction.	In Strict 2-PL, a transaction only reads value of committed transaction.

2PL :- A transaction is said to follow the Two-Phase Locking protocol if Locking and Unlocking can be done in two phases.

- **Growing Phase:** New locks on data items may be acquired but none can be released.
- **Shrinking Phase:** Existing locks may be released but no new locks can be acquired.

6.6 TIME STAMP CONCEPT :- The Timestamp Ordering Protocol is used to order the transactions based on their Timestamps. The order of transaction is nothing but the ascending order of the transaction creation.

- The priority of the older transaction is higher that's why it executes first. To determine the timestamp of the transaction, this protocol uses system time or logical counter.

- The lock-based protocol is used to manage the order between conflicting pairs among transactions at the execution time. But Timestamp based protocols start working as soon as a transaction is created.
- Let's assume there are two transactions T1 and T2. Suppose the transaction T1 has entered the system at 007 times and transaction T2 has entered the system at 009 times. T1 has the higher priority, so it executes first as it is entered the system first.
- The timestamp ordering protocol also maintains the timestamp of last 'read' and 'write' operation on a data.

Suppose there are three transactions T1, T2, T3

T1 has entered the system at 0110

T2 has entered the system at 0020

T3 has entered the system at 0030

priority will be given to transaction T1, then T2 and lastly T3 .

UNIT 7 RECOVERY

7.1 FAILURE CLASSIFICATION :-

Recovery is the process of restoring a database to the correct state in the event of a failure. it ensures that the database is reliable and remains in consistent state in case of failure

failure in terms of database can be defined as its in to execute the specified transaction or loss data from the database .

TO find that where the problem has occurred , we generalize a failure into the following category

a> ***Tansaction failure :-*** The transaction failure occurs when it fails to execute or when it reaches a point from where it can't go any further .

reasons for transaction failure .

- logical errors :- If a transaction cannot complete due to some code error or an internal error condition , then the logical error occurs
- Syntax error :- it occurs when the DBMS itself terminates an active transaction because the database system is not able to execute. Example:- The system aborts an active transaction in case of dead lock.

b> ***System crash :-*** System failure can occur due to power failure or other hardware or software failure. example :- Operating system failure (Error)

c> ***Disk Failure*** :- It occurs when hard disk drive or storage drives used to fail frequently . It was a common problem in the early days of technology evolution .

7.2 RECOVERY AND ATOMICITY

- Atomicity in DBMS refers to the property of database transaction where all the actions with in the transaction are executed as a single , indivisible unit of work . It ensure that the transaction is either fully completed or rolled back to the state it was in before the transaction begin.

- Two bank accounts

A	B
5000	
Transfer R.S 2000 from A to B	
R(A)	R(B)
A=A-2000	B=B+2000
W(A)	=5000
=3000	

After successful transaction A+B= 8000 — consistent
Before transaction A+B=8000

7.3 IN PLACE AND OUT PLACE UPDATE

#IN PLACE UPDATE/UPGRADE :- In place updating the existing SQL server to a newer version without creating a new instance. In other words, the new version of SQL server is installed on the same server as the existing version and database are upgraded to the newer version.

- The disk version of the data item is overwritten by the cache version

 # OUT OF PLACE UPDATE :- Out of place update employed where the updated data is stored into a new location while its original copy is set as garbage and will not be used any further.

- The installer install the newer version in a separate oracle clusterware home.

7.4 LOG BASED RECOVERY :-

- A log is a file that records all the change made to the database.

- Log based recovery provides the facility to maintain or recover data if any failure may occur in the system

- A log is a series of records

- The logs for each transaction are kept in a log file to allow recovery in case of failure.

- The log is a sequence of log records recording all the update activities in the database

- Different types of log records are as follows –

- <Ti, Xi, V1, V2> – update log record, where Ti=transaction, Xi=data, V1=old data, V2=new value.
- <Ti, start> – Transaction Ti starts execution.
- <Ti, commit> – Transaction Ti is committed.
- <Ti, abort> – Transaction Ti is aborted

7.5 SHADOW PAGING

- An alternative to log based crash recovery technique is shadow paging

- The database is partitioned into same number of fixed length block, which are referred to as pages.

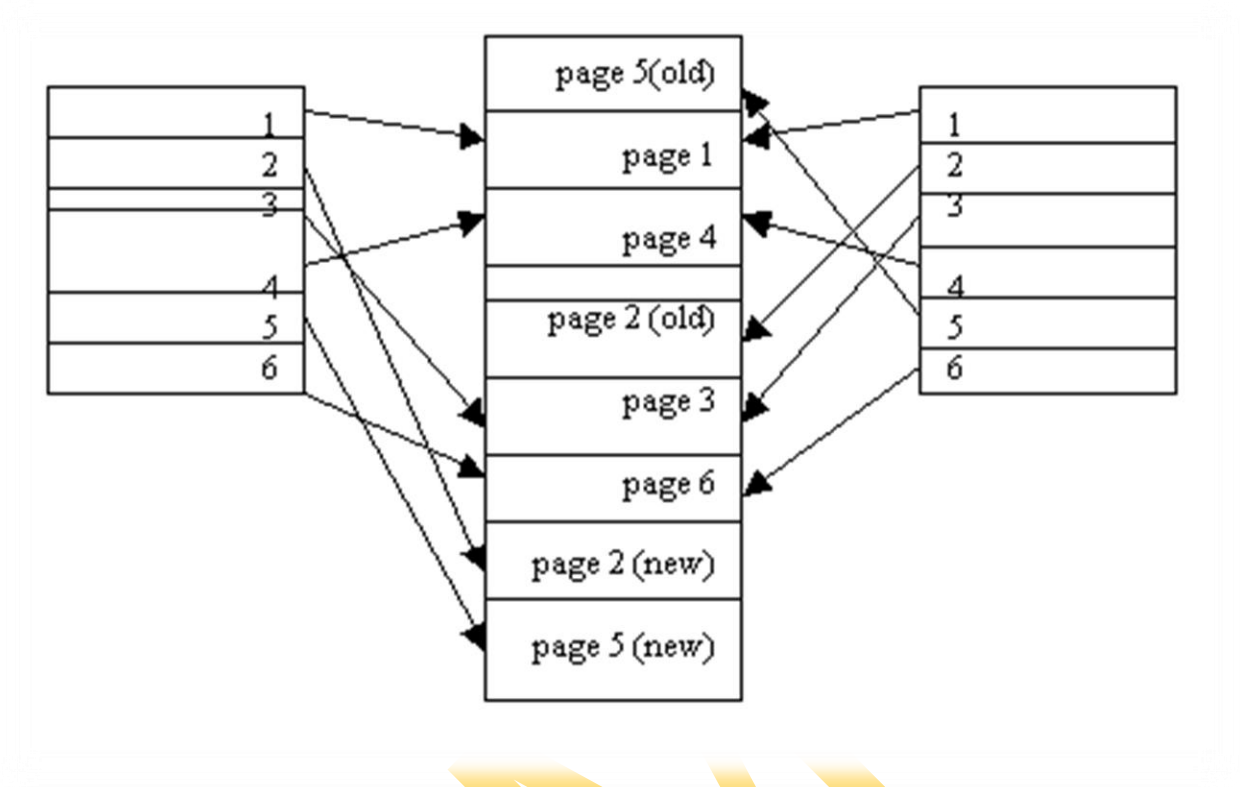
- During the process of shadow paging two page tables create such as current page table and shadow page table to access the pages of the disk (where data is stored). And if the system crash, our data will be saved by shadow page .

- Assume there are n pages, numbered from 1 to n, these pages are not stored in any particular order on the disk.

- For finding I^{th} page database for any given I . for this we use a page table.

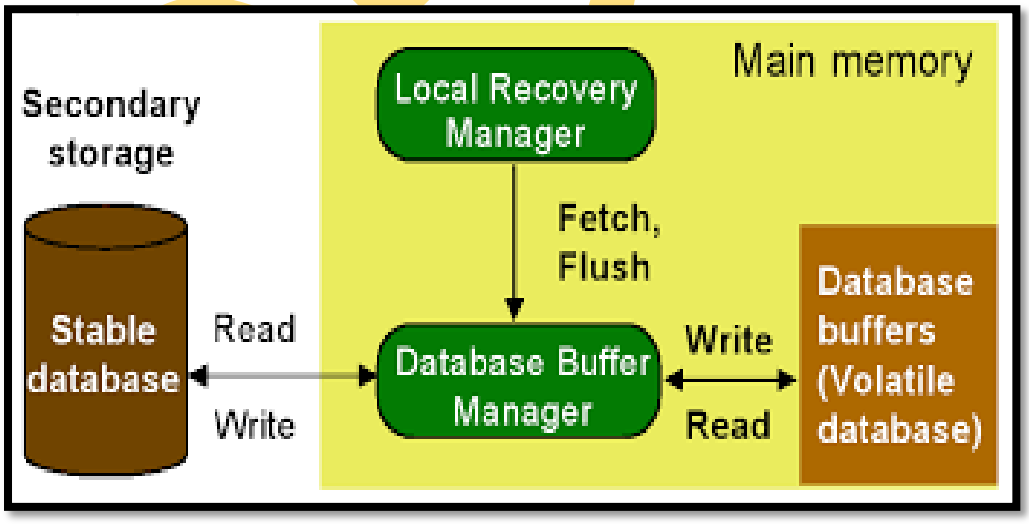
- Each entry contains a pointer to page on a disk.

- Shadow page table never changed. The current page table make all the changes.



7.6 LOCAL RECOVERY MANAGER :-

- Recovery manager is a utility that manages the processes of creating backups of all database files and restoring files from backup .



-Database buffer manager keeps some of the recently accessed data in main memory .

- The part of the database that is in the database that is in the database buffer is called the volatile database .

-LRM executes the operations on behalf of a transaction only one the volatile database , which at a later time is written back to the stable database .

- When LRM wants to read a page of data, it issues fetch command , indicating the page that it wants to read

- The buffer manager checks to see if that page is already in the buffer and if so, makes it available for transaction if not , it reads a page from the stable database .

7.7 UNDO AND REDO PROTOCOL

Redo and undo is database transaction log for handling transaction crash recovery. This algorithm stores all the values and changes made during transaction in a separate memory in case of failure or crash .

- This algorithm is a combination of two approaches

1. **Undo:**
   - stands for undone and restores the data value items that are updated by any transaction to their previous value.
   - using a log record sets the data item specified in log record to old value.
2. **Redo:**
   - using a log record sets the data item specified in log record to new value.
   - **stands for** Re-done and it set the value of all the data updated by the transaction to the new value.