SQL (Structured Query Language)

Database – Database is a collection of interrelated data that helps in the efficient retrieval insertion & deletion of data from the database and organise data in the form of table, views, schema, reports etc.

→ **DBMS** – It stands for database management system, it is a software system designed to create, manage and manipulate databases. It allow users to efficiently store retrieve update and manage data in a structured way. Ex. MySQL

→ **Key Features of DBMS :**

1. Data Manage
2. Data Retrieval
3. Data Manipulation
4. Data Security
5. Data Integrity

→ **Data Abstraction** : It means hiding the complex internal details of data and showing only the essential features to the user.

→ **Data Independence** : It means the ability to change the data storage structure or schema at one level of a database system without affecting the higher levels.

Types :

1. **logical data Independence** – changing the logical schema (like adding/removing fields, merging tables, changing relationships) should not affect the external views or applications.

2. Physical Data Independence – Changing the physical storage of data (like how it's indexed, stored on disk, file structure) should not affect the logical schema.

Ex. Moving a table from one hard drive to another hard drive, changing indexing method from B-tree to Hash Index.

DDL – Data definition language, it is a set of SQL commands used to define, create, modify and delete database structures like tables, schemas, indexes and views.

MY SQL

→ Create Table

```
CREATE TABLE students (
    roll_no INT PRIMARY KEY,
    name VARCHAR(50),
    age INT
);
```

→ Alter Table

```
ALTER TABLE students ADD
phone_number VARCHAR(15);
```

→ Drop Table

```
DROP TABLE students;
```

→ Truncate

```
TRUNCATE TABLE students;
```

→ Rename

```
RENAME TABLE students TO
Learners;
```

Rename Postgre SQL

→ ALTER TABLE Students Rename to Learners.

Postgre SQL

→ Create Table

```
CREATE TABLE students (
    roll_no SERIAL PRIMARY KEY, //auto incre-ment
    name VARCHAR(50) NOT NULL,
    age INT CHECK(age>0),
    phone_number VARCHAR(15),
    extra_data JSONB //
);
```

→ Alter Table

```
ALTER TABLE students ADD
email VARCHAR(100) UNIQUE;
```

→ Drop Table

```
DROP TABLE IF EXISTS students
CASCADE
```

// CASCADE = drop dependent object also

→ Truncate

```
TRUNCATE TABLE students RESTART
IDENTITY CASCADE
```

#Restart Identity = reset auto increment
CASCADE = truncate dependent tables too

DML (Data Manipulation Language)

MY SQL | POSTGRE SQL

1. Insert (Add data into table)

```
INSERT INTO Students (roll_no, name, age)
VALUES (1, 'Shivam', 21);
```

```
INSERT INTO Students (name, age)
VALUES ('Shivam', 21)
Returning roll_no; // returns auto generated id
```

2. Update (Modify data)

```
UPDATE Students
SET age = 22
WHERE roll_no = 1;
```

```
UPDATE Students
SET age = 22
WHERE roll_no = 1
RETURNING *;
```

3. DELETE (Remove Data)

```
DELETE FROM Students WHERE roll_no = 1;
```

```
DELETE FROM Students
WHERE roll_no = 1
Returning;
```

DQL (Data Query Language)

1. SELECT (Retrieve data)

```
SELECT * FROM Students WHERE age > 18;
```

```
SELECT name, age
FROM Students
WHERE age > 18
ORDER By age DESC
```

DCL (Data Control Language)

1. Grant
2. Revoke

```
// Grants permission to select and insert to user
GRANT SELECT, INSERT ON students
TO 'shivam'@'localhost';
// Permission revoke
REVOKE INSERT ON students FROM
'shivam'@'localhost';
```

```
GRANT SELECT, INSERT ON Students
TO shivam;

REVOKE INSERT ON Students FROM
Shivam;
```

TCL (Transaction Control language)

→ It is used to control the transactions.

Transaction = Set of query executed as unit.

Common TCL Commands:

1. COMMIT – Changes permanently save
2. ROLLBACK – Undo changes
3. SAVEPOINT – Create immediate point in transaction
4. SET TRANSACTION – To set transaction properties.

MySQL

```
START TRANSACTION;
UPDATE Students SET age=22
WHERE roll_no = 1;
SAVEPOINT before_delete;
DELETE FROM students WHERE
roll_no = 2;
ROLLBACK TO before_delete;
COMMIT;
```

Postgre SQL

```
BEGIN;
UPDATE students SET age=22
WHERE roll_no=1;
SAVEPOINT before_delete;
DELETE FROM students WHERE
roll_no = 2;
ROLLBACK TO SAVEPOINT before_delete;
COMMIT;
```

Data Models

Data Models – It can be defined as an integrated collection of concept for describing and manipulating data, relationship b/w data and constraint on the data in an organization.

Types of data models: 1. Object Based
2. Record Based

Object Based :- 1. Object oriented
2. ER Model

Record Based :- 1. Heirarchical Model
2. Network Model
3. Relational Model

Record Based

→ Relational Model :

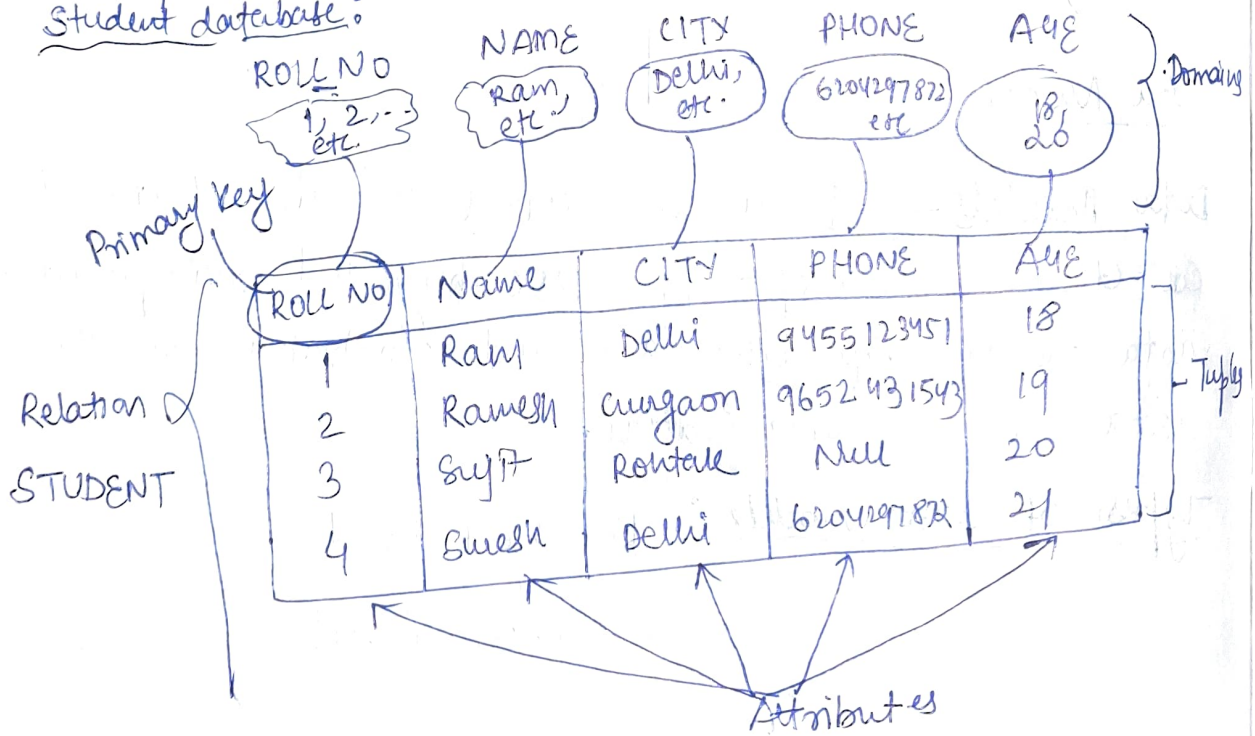
→ Proposed in 1970s by Codd of IBM.

→ It represents the database as collection of relations and each relation represent a table of values, each row representing a collection of related values.

→ It consist of set of relation and domain.

→ The table is a relation in which each row is called tuple, Column B attribute.

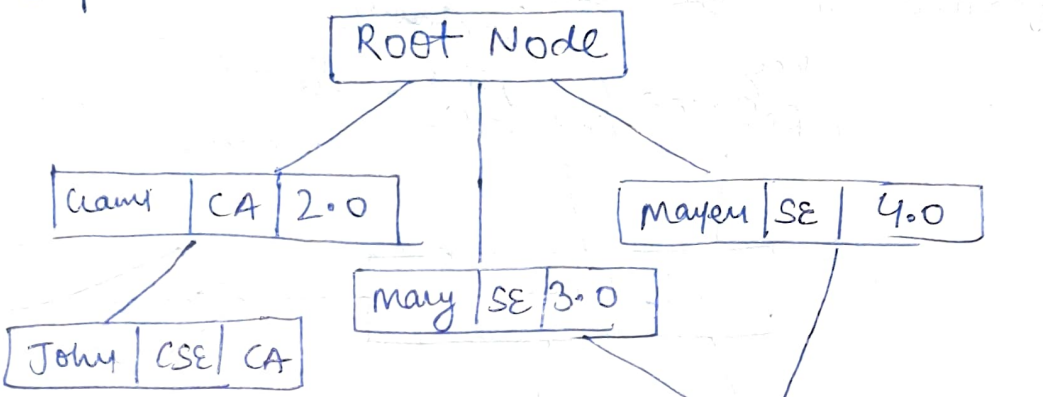
Student database:



→ Hierarchical Model

→ This model is used to describe those record structure in which the various physical records which make up the logical record are tied together in a sequence which looks like an inverted tree.

→ At the top of the structure is a single record, that is your parent node.



Faculty Table

Name	Dep	Course Taught
John	CSE	CA
Jake	CSE	S E
Royal	CSE	DBMS

Student Table

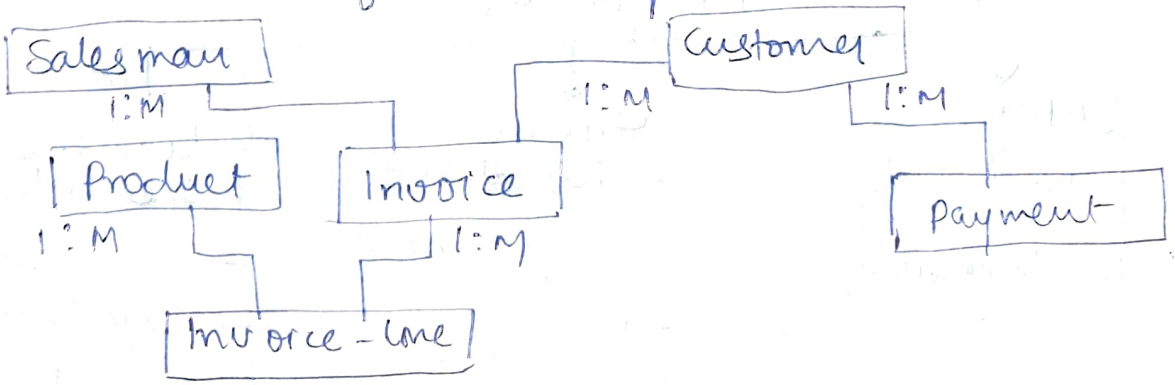
Name	Course-enroll	Grade
Liam	CA	2.0
Mary	SE	3.0
Mayer	SE	4.0

→ Network Model

→ It replaces the hierarchical tree with a graph thus allow more general connection among the nodes.

→ The popularity of this model coincide with popularity of heirarchical data model.

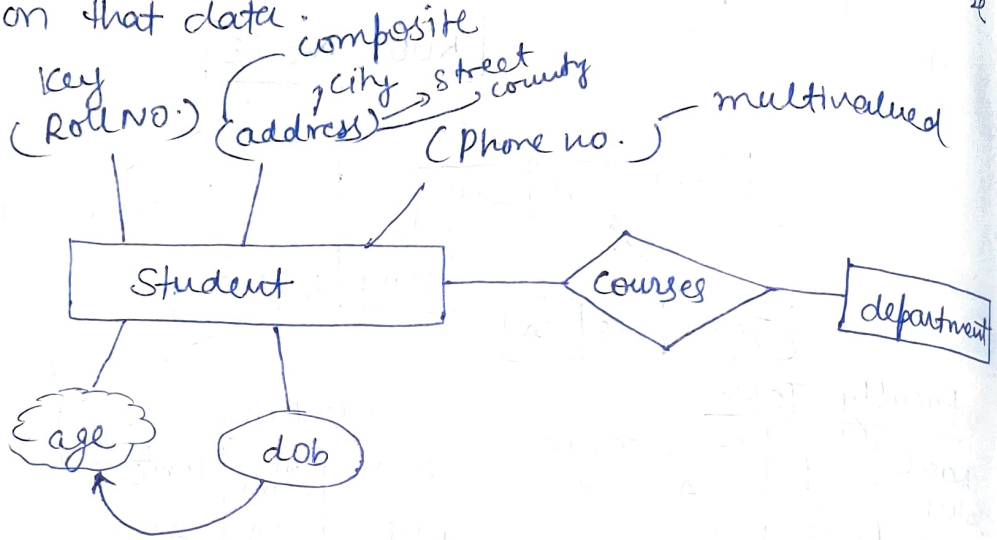
Network model of finance department :



→ Objected Oriented Data Model

→ It is a data model that organizes data based on the principle of oops.

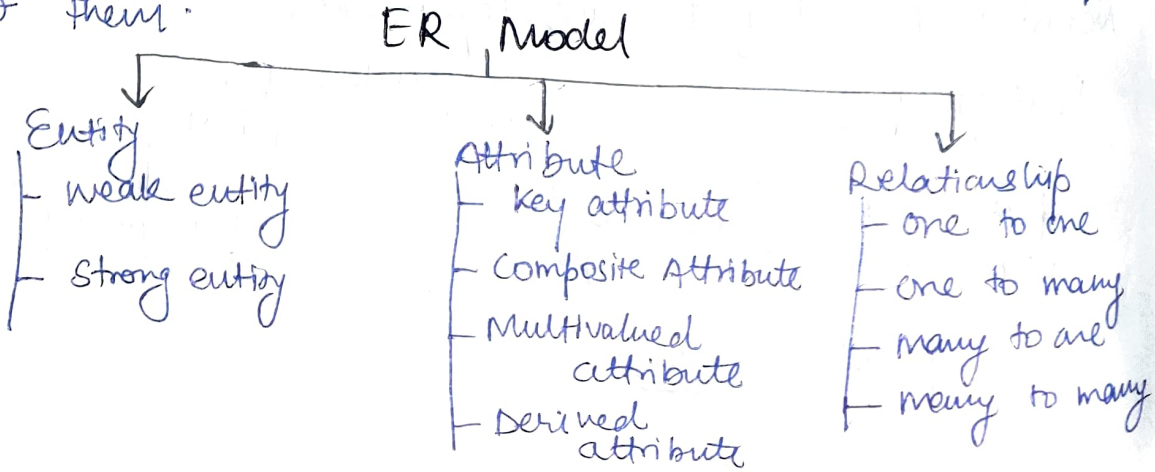
→ In this model data is represented as object which encapsulate both data and the operation that can be performed on that data.



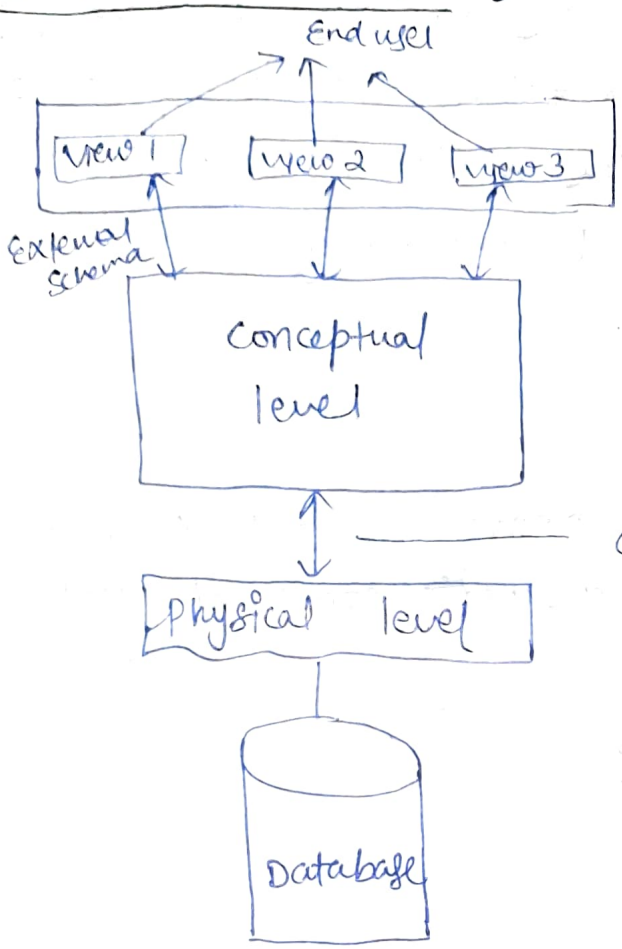
→E-R Model

→ E-R represents entity relationship model, it is the conceptual model for designing database.

→ This model represent the logical structure of database including entities, their attributes and relationship b/w them.



Level of Abstraction :



3-Level architecture of DBMS

→ Physical level is also known as internal level or it is the lowest level of data abstraction. It describes how the data is actualy stored in database.

→ Conceptual level is the middle level. It describes what data is stored in database and what relationship exist in those data.

→ View level or external, it is the highest level of data abstraction, it describes only part of the entire database and it describes user interaction with database system by application program that hide the data type.

→ Database schema defines the variable declaration along with type declaration in table that belong to a particular database & value of these variables at a moment of time is called instance of that database.

Integrity Constraints

→ Integrity constraints in a DBMS are rules that help keep the data in a db accurate, consistent and reliable.

Type of Integrity constraints :

1. Domain constraints
2. Entity Integrity constraints
3. Key constraints
4. Referential Integrity constraints
5. Assertion
6. Triggers.

→ **Domain Constraints** – It is a type of integrity constraint that ensure the values stored in a column of a db are valid and within a specific range or domain.

→ **Entity Integrity Constraints** – It states that primary key can never contain null value because primary key is used to determine individual rows in a relation uniquely, If primary key contains null value then we cannot identify those rows.

- **Key constraints** – It ensures that certain columns or combinations of columns in a table uniquely identify each row.
  - → It prevents duplicated
  - → maintain relationships
  - → Enforce data integrity

- → **Refrential Integrity Constraints** – These are the rules that ensure relationships ~~They~~ b/w table remain consistent.
- → They enforce that a foreign key in one table must either match a value in the refrenced primary key of another table or be NULL.
- → This guarantees the logical connection b/w related tables in a relational database.
- → It is important : maintain consistency
  Prevents orphan records
  Enforce logical relationships.

- → **Assertion** – It is a declarative mechanism in a db that ensures a specific condition rule is always satisfied across the entire database.

  Create assertion <assertion-name> check <predicate>

- → **Triggers** – It is a procedural statement in a db that is automatically executed in response to certain events such as INSERT, UPDATE, or DELETE.