SQL

Ques :- What are Databases?

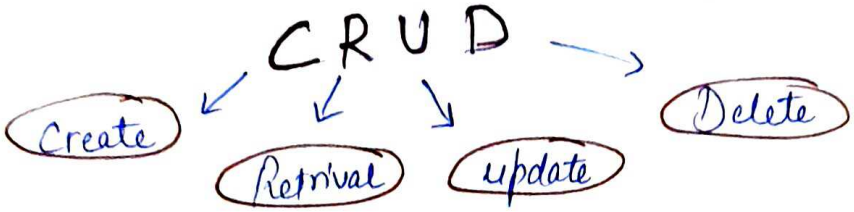
A Database is a shared collection of logically related data description designed to meet the information needs of an organization.

Data storage :- A database is used to store large amounts of structured data, making it easily accessible, searchable, and retrievable.

Data Analysis :- A database can be used to perform complex data analysis, generate reports, and provide insights into the data.

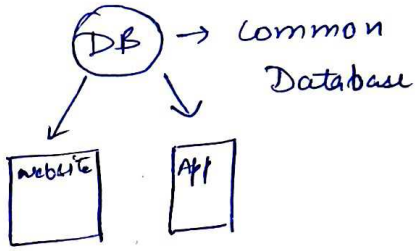
Record Keeping :- A database is often used to keep track of important records, such as financial transaction, customer information and inventory levels.

Web Application :- Databases are an essential component of many web applications, providing dynamic content and user management.



Properties of an Ideal Database

1. Integrity → accurate + consistency
2. Availability → 24X7 (0 downtime)
3. Security
4. Independent of Application →



5. concurrency → parallel usage

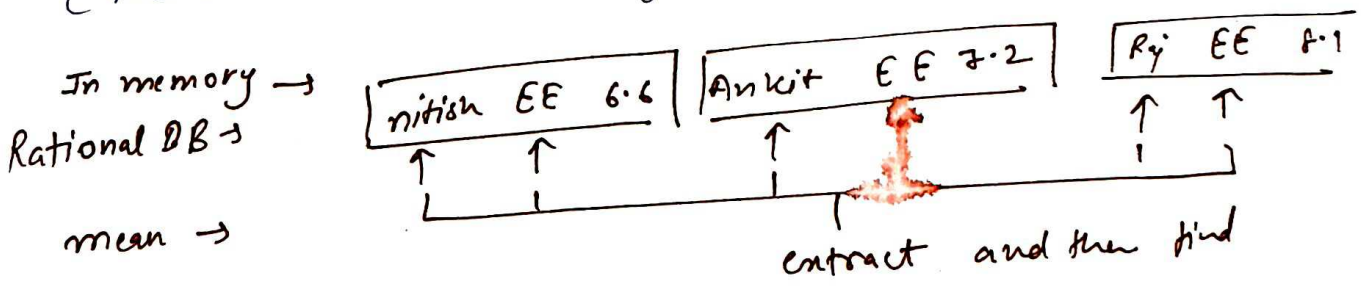
Types of Databases

1. Relational Databases (Rows) – Also known as SQL database these databases use a ~~sati~~ relational model to organize data into tables with rows and columns.
2. NoSQL Database :- These databases are designed to handle large amount of unstructured or semi-structured data, such as documentation image, or videos (MongoDB).

3. Column Database (Col): These databases store data in columns rather than rows, making them well-suited for data warehousing and analytical applications. (Amazon, Redshift, Google Big Query)

4. Graph Databases: The databases are used to store and query graph-structured data, such as social network connection or recommendation systems. (Amazone Neptune)

5. Key-value Database: These database store data as collection of keys and values, making them well-suited for caching and simple data storage needs (Redis and Amazon Dynamo DB)

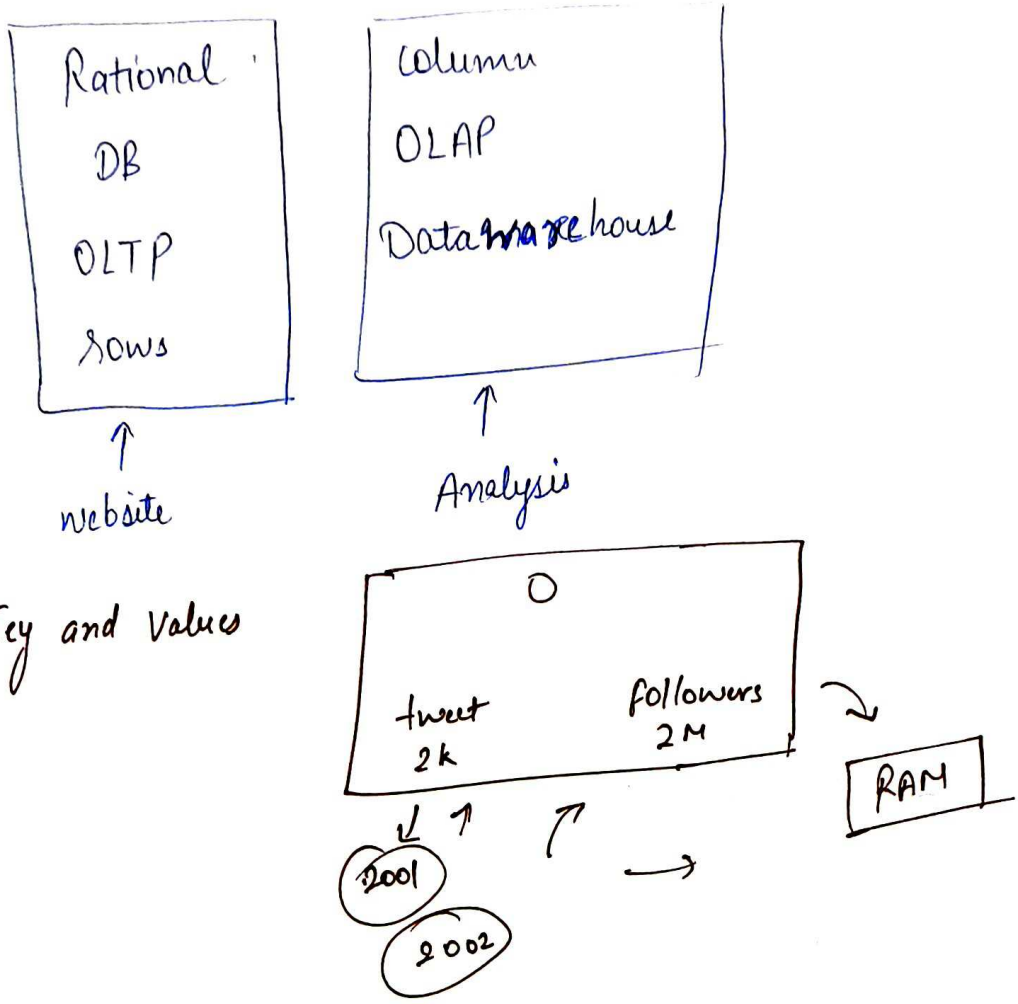


col DB

Student id	name	Student id	Branch	Student Cgpa
1	nitish		EE	8.2
2	Ankit		EE	8.1
3	Raj		EE	6.6
4	!		!	!
	!		!	!

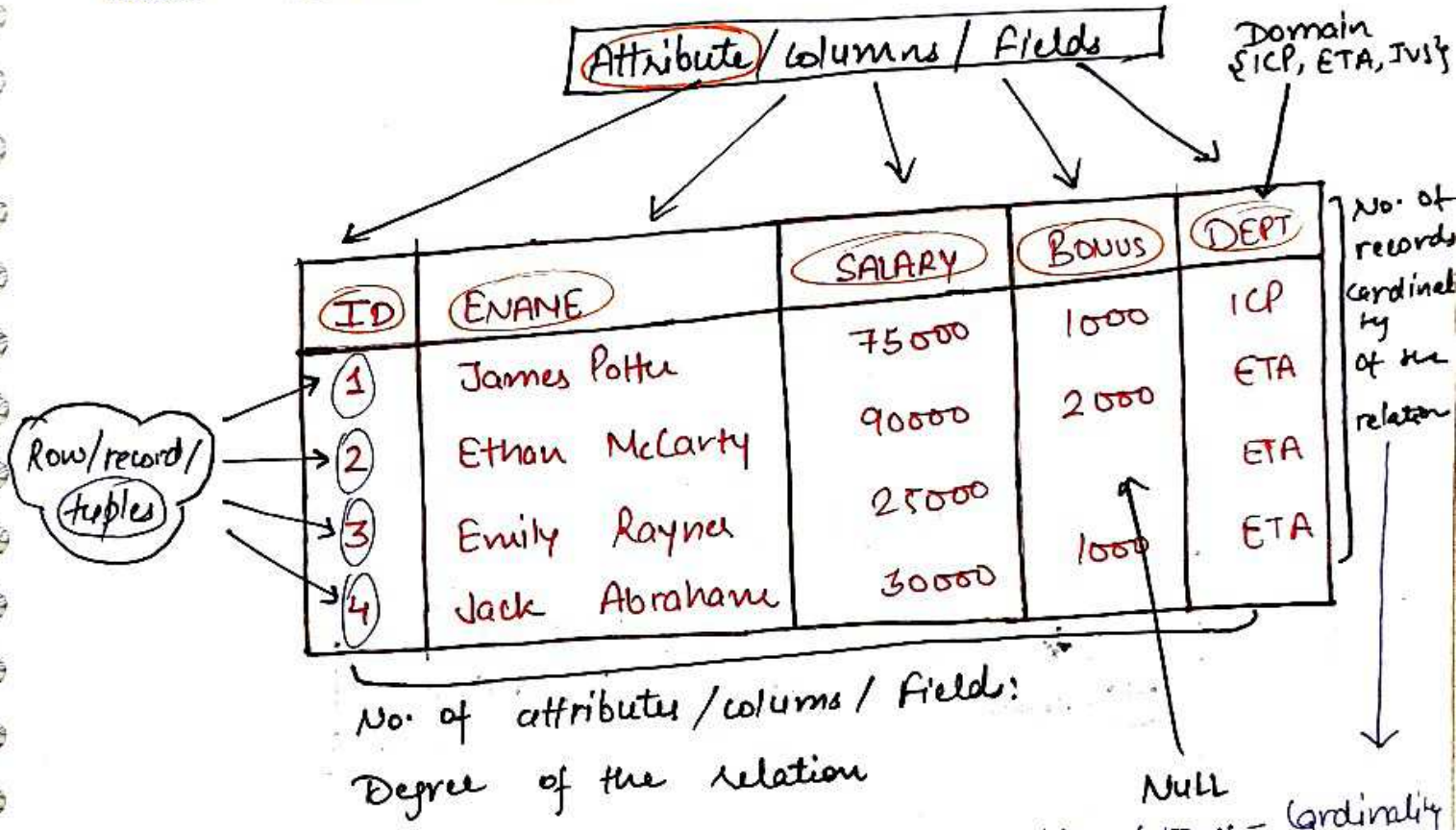
In Memory: nitish, Ankit Raj - -, EE, - - - - | 8.2, 8.1 - - -

Load data and find mean


Rational
DB
OLTP
Rows
Column
OLAP
Data warehouse
website
Analysis
Key and Values
tweet
2k
followers
2M
RAM
2001
2002

Relational Databases

Also known as SQL databases, these databases use a rational model to organize data into tables with rows and columns.



* No. of rows = Cardinality of relation

  4 = Cardinality

* Domain
  - ↳ Type of item stored in column

* Null = 0 value

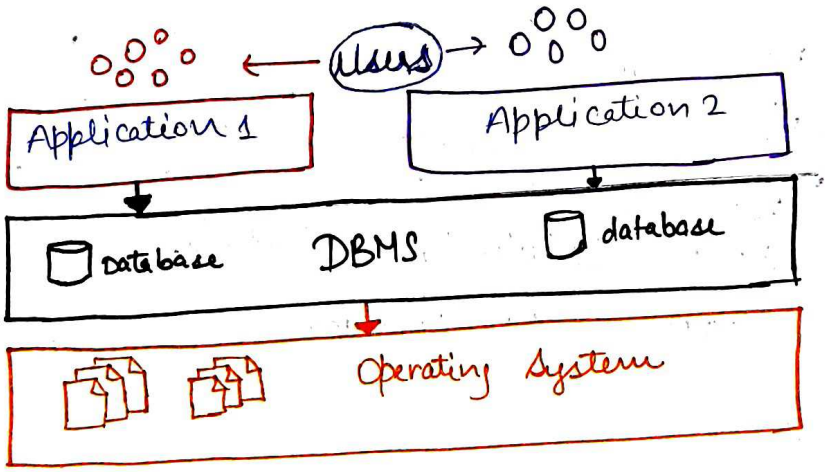
* tuples ⇒ row

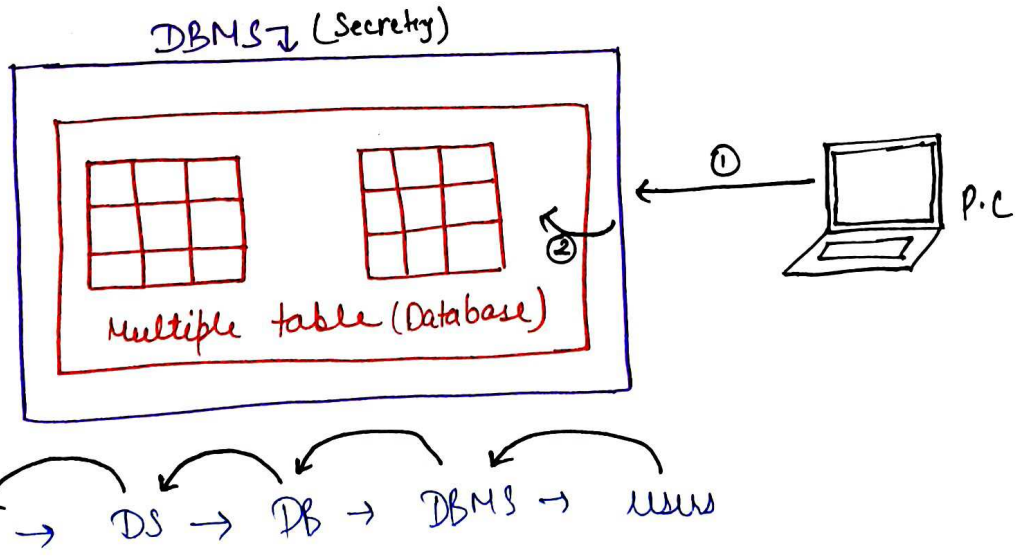
* 5 columns = Degree of the relation

* Attribute ⇒ column

What is DBMS?

A database management system (DBMS) is a software system that provides the interface and tool needed to store, organize, and manage data in a database. A Database Management system act as an intermediary between the database and the application or users that access the data stored in the database.





Function of DBMS

Data Management :- Store, retrieve and modify data

CRUD

Integrity :- Maintain accuracy of data.

Concurrency :- Simultaneous data access for multiple users

Transaction :- Modification to database must either be successful or must not happen at all.

Security :- Access to authorized users only

Utilities :- Data import / export user management, backup, logging

Database Keys

A key in a database is an attribute (column) or set of attributes (multiple col.) that uniquely identifies a tuple (row) in a table. Keys a crucial role in ensuring the integrity and establishing relationships between tables.

Roll no is unique for every student, so, Roll no will be (key)

multiple same name not unique (no key)

Email is unique of every student (Key)

Roll no	Name	Branch	Email
1	Nitish Singh	CSE	nitish@gmail.co
2	Ankit Sharma	EEE	ankit@gmail.co
3	Neha Verma	ME	neha@gmail.co

multiple same branch (No key)

1. **Super Key** :- A super key is combination of columns that uniquely identifies any row within in a relation database management system (RDBMS) table.

Set of keys

- Roll no
- Email
- Roll no. + Name
- Roll no. + Branch
- Roll no. + Email
- Name + Branch + Email
- Roll no. + Branch + Name
- Roll no. + Branch + Name + Email

Candidate Key :- A Candidate Key is a minimal super key, meaning it has no redundant attributes. In other words, it's the smallest set of attributes that can be used to uniquely identify a tuple (rows) in the table.

- → Roll no. ✓ → No Redundant
- → Email ✓ → No Redundant

⇒ Candidate Key

Candidate Key

- → Roll no. + Name ✗ → Redundant
- → Roll no. + Branch ✗ → Redundant
- → Roll no. + Email → Roll no. is key so, we don't need one more candidate
- → Name + Branch + Email ✗ → Candidate
- → Roll no. + Branch + Name ✗ (Redundant)
- → Roll no. + Branch + Name + Email ✗

Criteria of Candidate

- → Not Null
- → Not Repeat → Not duplicate
- →

good to have criteria

- → Numerical
- → small
- → constant

} Primary Key

Primary Key :- A primary key is a unique identifier for each ~~column~~ tuple in a table. There can only be one primary key in a table, and it cannot contain null values.

Criteria of Candidate

1. Not Null
2. Not Repeat → Not duplicate

Good to have Criteria

1. Numerical
2. Small
3. Constant

→ 2 Candidate Key :- Roll no. , Email

① Roll no. → not null ✓
→ Not repeat ✓
→ Numerical ✓
↓ ↓ ↓
Constant ✓ Small ✓ ↳ Sometimes alpha numerical

Roll no. is primary key
↳ 4 Criteria

② Email → Not Null ✓
→ Not repeat ✓
→ Numerical ✗
↓ ↓ ↓
Constant ✗ Small → ✗
↳ as compare to roll no.

Email is not primary key
↳ 2 criteria

↳ sometimes student change their email of the college

Alternate Key :- An Alternate key is a candidate key that is not used as the primary key.

→ Roll no. is primary key. So, Email is a Alternate key

Candidate Key – Primary Key ⇒ Alternate Key

composite key :- A composite key is a primary key that is made up ~~with~~ of two or more attributes. Composite keys are used when a single attribute is not sufficient to uniquely identify a tuple in a table.

Student

Std. id	Name	email	Phone

course

course id	course Name	Price	instructor

* Composite key *

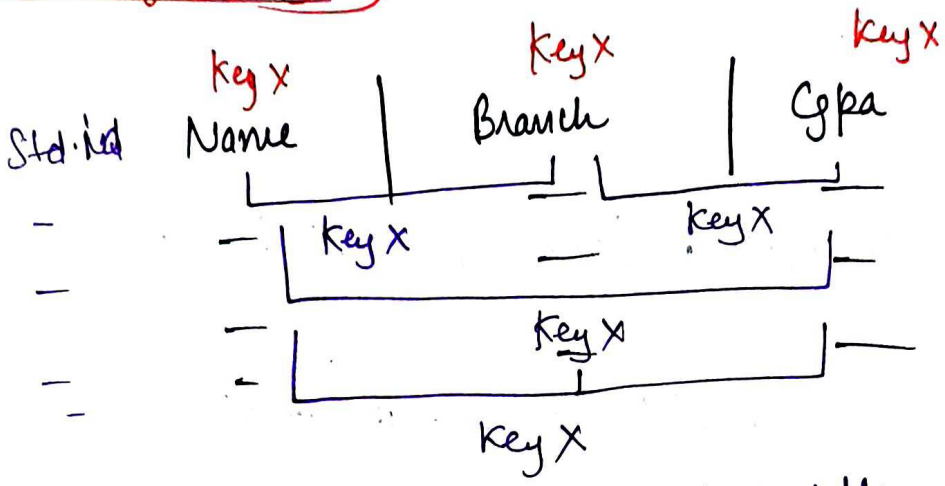
Single student can not enrolled Same course multiple time

enrollment

Std. id	course id	date	Payment method

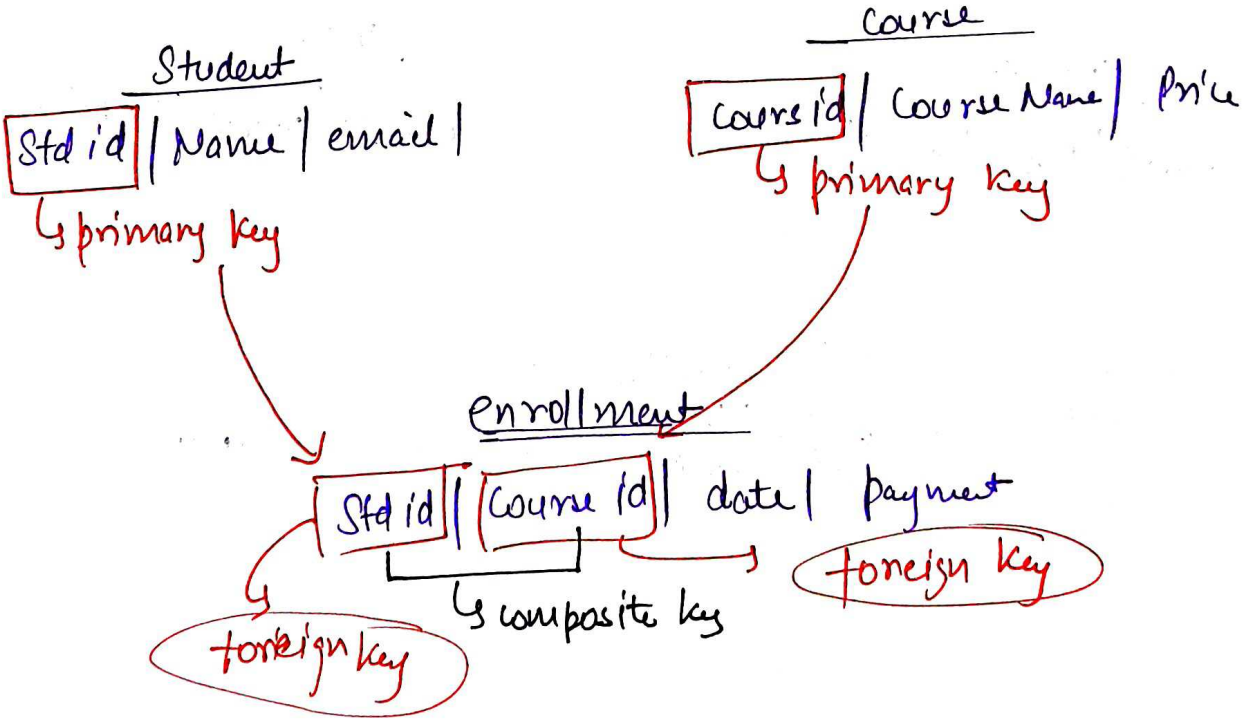
- Std. id: Student enrolled multiple course; key ×
- course id: key × ↳ multiple student enrolled same course
- date: key × ↳ multiple course enrolled Same date
- Payment method: key × ↳ multiple student pay with same method

Serrogate Key



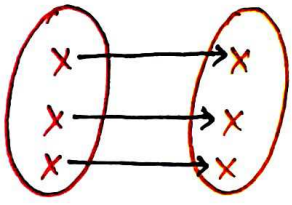
So there are no key in this table. We can add own column like student id or Roll no.

Foreign Key :- A foreign key is a primary key from one table that is used to establish a relationship with another table.

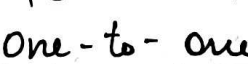


Cardinality Of Relationships

Cardinality in databases relationship refers to the number of occurrences of an entity in a relationship with another entity. Cardinality defines the number of instances of one entity that can be associated with a single instance of the related.

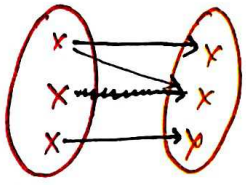


1 : 1

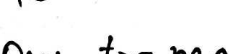


One-to-one relationship

↳ 1 table

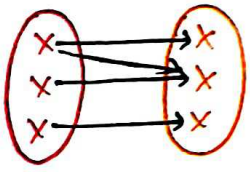


1 : N



One-to-many relationship

↳ 2 table



M : N

Many-to-many relationship

3 table

→ Jiska hum table bna skte hai

entity → table e.g → student, Restaurant

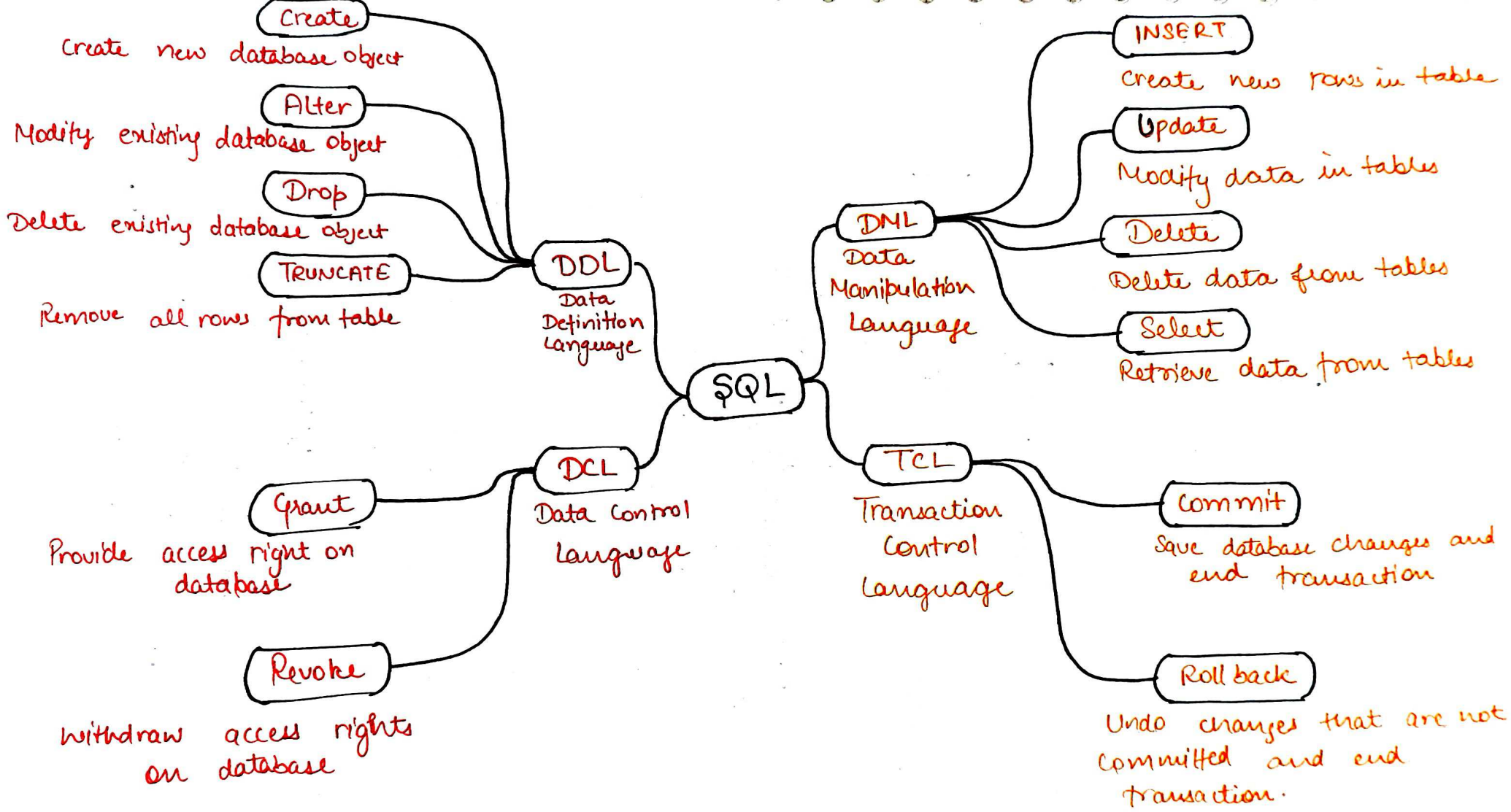
Example :-

1. Person → Driving License Number
2. Student → College branch
3. Restaurant → orders
4. Restaurant → menu
5. students → courses

Flexibility :- The structure of a database is often rigid and inflexible, making it difficult to adapt to changing requirement or to accommodate new types of data.

What is SQL?

SQL (Structured Query Language) is a programming language used for manage and manipulating data in relational database. It allows you to insert, update, retrieve, and delete data in a database. It is widely used for data management in many application, website, and business. In simple terms, SQL is used to communicate with a relational database.



Types of SQL commands

DDL commands for Database

1. Create

```
CREATE DATABASE Database name
```

```
CREATE DATABASE IF NOT EXISTS database Name
```

2. DROP

```
DROP DATABASE databasename
```

```
DROP DATABASE IF EXISTS database name
```

DDL commands for Tables

1. Create 2. Truncate 3. Drop

user

user id	Name	Email	Password
int	varchar (255)	varchar (255)	Varchar (255)

create ↓

```
CREATE TABLE table name (
    column_name  data-type,
    Column_ name  data_type (size),
    Column_ name  data_type,
)
```

Truncate ⇒ empty table

```
TRUNCATE TABLE table name
```

Drop ⇒ Delete table

```
DROP TABLE IF EXISTS table_name
```

Data Integrity ⇒ Consistency + accuracy
(all story) (not change)

Data integrity in database refers to the accuracy, completeness and consistence of the data stored in a database. It is a measure of the reliability and trustworthiness of the data and ensures that the data in a database is protected from errors, corruption, or unauthorized changes.

There are various methods used to ensure data integrity, Including:

Constraints: Constrains in databases are rules or conditions that must be met for data to be inserted, updated, or deleted in a database table. They are used to enforce the integrity of the data stored in a database and to prevent data from becoming inconsistent or corrupted.

Transaction :- a sequence of database operation that are treated as a single unit of work.

Normalization :- a design technique that minimize data redundancy and ensures data consistency by organizing data into seperate tables.

Constraints in MySQL

Constraints in database are rules or conditions that must be met for data to inserted, updated, or deleted in a database table. They are used to enforce the integrity of the data stored in a database and to prevent data from becoming inconsistent or corrupted.

1. Not Null → Null is not allowed eg:- IRCT → ticket → (PNR) (always written)
2. Unique (combo) → email → everyone have unique email
   → Another way of creating constraint
3. Primary Key → Not Null + Unique ⇒ Any one is primary key and multiple column is possible.
4. Auto increament

Users

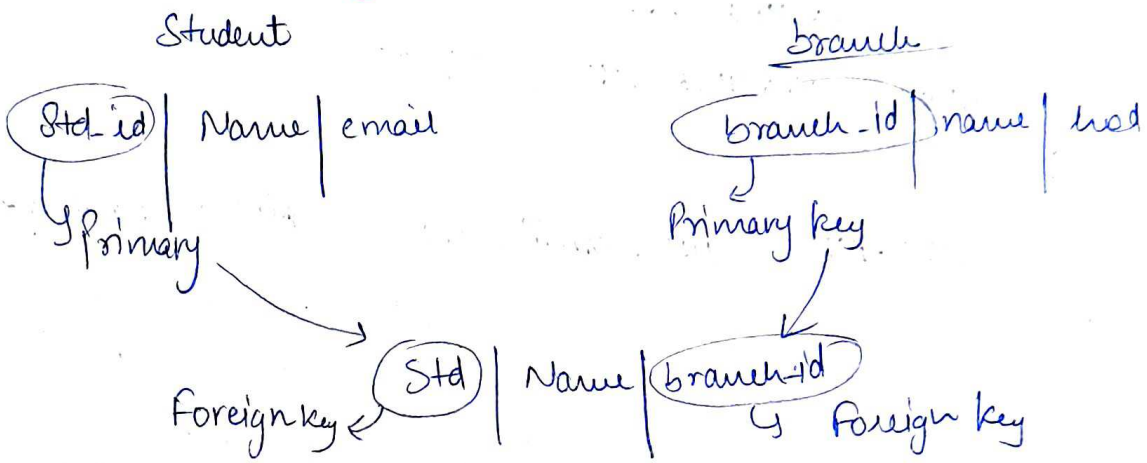
User_id	Name	email

increament +1 → 70 → 71

5. Check ⇒ Check Age should be greater than 13

6. Default ⇒ assign default value

7. Foreign key



Commands

1. Not Null

```
CREATE TABLE table_name (
    column_name INTEGER NOT NULL,
    column_name VARCHAR(255) NOT NULL,
    column_name VARCHAR(255) NOT NULL,
    column_name VARCHAR(255),
)
```

→ some value insert in Null

2. Unique

```
CREATE TABLE table_name (
    column_name INTEGER NOT NULL UNIQUE
)
```

another method

```
CREATE TABLE users (
    user_id INTEGER NOT NULL,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL,
    password VARCHAR(255) NOT NULL,
}
CONSTRAINT user_email_unique UNIQUE (name, email)
)
```

↑ you can give the name of constraint

Combination of email and name not repeat

eg:-

Name
Ankit
Ankit
Suraj

This is not repeat

email
Ankit@gmail.com
Ankit@gmail.com
Ankit@gmail.com

email can repeat

* if you want to delete constraint in future. so, there no need to delete entire table.

* You can remove or delete unique and add same name and email combination after deleting constraint. no need to delete entire table.

* combination of name and email is not repeat

3. Primary key

```
CREATE TABLE users (
    user_id INTEGER NOT NULL PRIMARY KEY,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL,
    password VARCHAR(255) NOT NULL,
)
```

another method

```
CREATE TABLE users (
    user_id INTEGER NOT NULL,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL,
    password VARCHAR(255) NOT NULL,

    CONSTRAINT users_pk PRIMARY KEY (user_id)
)
```

PRIMARY KEY (user_id) → combination of column become Primary key.

```
CONSTRAINT user_email_unique UNIQUE (email)
```

4 Auto Increament

```
create TABLE users (
    user_id INTEGER PRIMARY KEY AUTO_INCREAMENT,
    name VARCHAR(255) NOTNULL,
)
```

AUTO_INCREAMENT → if we don't give any value then automatically increament value from 1 then 2 then 3 . . .

5. Check

```
CREATE TABLE students (
    student_id INTEGER PRIMARY KEY AUTO_INCREAMENT,
    name VARCHAR(50) NOT NULL,

    ~~CONSTRAINT student_age_check~~
    age INTEGER CHECK (age > 6 AND age < 25)
)
```

OR

Use Constraint → CONSTRAINT student_age_check CHECK (age > 6 AND age < 25)

* If we Insert age less than 6 or more than 25 then produce error.

6. Default → If we not give any input then any default value insert at that place. eg:- Gender → (i) male and (ii) female and (iii) other. If male or female will not selected then default insert other.

```
CREATE TABLE ticket (
    city VARCHAR(255) DEFAULT 'Sadness'
)
```

7. Foreign key

Customer

Customerid | name | email

Orders

order_id | customer_id | date

```
CREATE TABLE Customers (
    cid INTEGER PRIMARY KEY AUTO_INCREMENT,
    name VARCHAR(255) NOT NULL,
    email VARCHAR(255) NOT NULL UNIQUE,
)
```

```
CREATE TABLE Order (
    Order_id INTEGER PRIMARY KEY AUTO_INCREMENT,
    Cid INTEGER NOT NULL,
    order_date DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    CONSTRAINT Orders_fk FOREIGN KEY (Cid) REFRENCE
        customers (Cid).
)
```

* If we want to delete customers table it produce error because order table depend on customers table of Cid column (Foreign key).

Customer

Cid	Name	email
1	Sam	Sam...
2	Dan	Dan...

↳ Primary key

Order

Order_id	Cid	date
1	1	...
2	2	...

↳ Foreign key

* If I delete primary key (Cid) without deleting Foreign key (order) (Cid) then produce error.

* 1. Delete Foreign key
  2. Delete Primary key.

Referential Actions → If two tables are related via a foreign key. So, In a one table you can delete or change then how second table will response is called Referential Actions.

1. Restrict :- Means if foreign key present and trying to delete from Primary key without deleting foreign key that's create Restriction and produce error.

2. Cascade : Means if change or delete in Primary key then automatically change or delete in foreign key

3. Set Null : If delete in Primary key then automatically data change into NULL in Foreign key.

4. Set Default : If I set default value 0. and I delete data in Primary key then data change into 0 not null in Foreign key.

Cascade :- * Customers table created by same command

```
CREATE TABLE orders (
    Order_id INTEGER PRIMARY KEY,
    c_id INTEGER NOT NULL,
    Order_date DATETIME NOT NULL DEFAULT
        CURRENT_TIMESTAMP,
```

```
CONSTRAINT order-fk FOREIGN KEY (cid) REFFENCE Customer(cid)
ON DELETE CASCADE
ON UPDATE CASCADE
)
```

Set default :- * Customer table created by same command

```
CREATE TABLE orders (
    Order-id INTEGER PRIMARY KEY,
    cid INTEGER,
    Order_date DATETIME NOT NULL DEFAULT CURRENT_TIMESTAMP,
    CONSTRAINT order-fk FOREIGN KEY (cid) REFRENCE
                                    customer (cid)
    ON DELETE SET NULL,
)
```

don't write NOT NULL because cid value change into NULL after deleting.

ALTER TABLE COMMAND

The Alter Table statement in SQL is used to modify the structure of an existing table. Some of the things that can be done using the Alter TABLE statement include

1. Add columns
2. Delete columns
3. Modify columns