SQL

Structured Query Language

What is SQL?

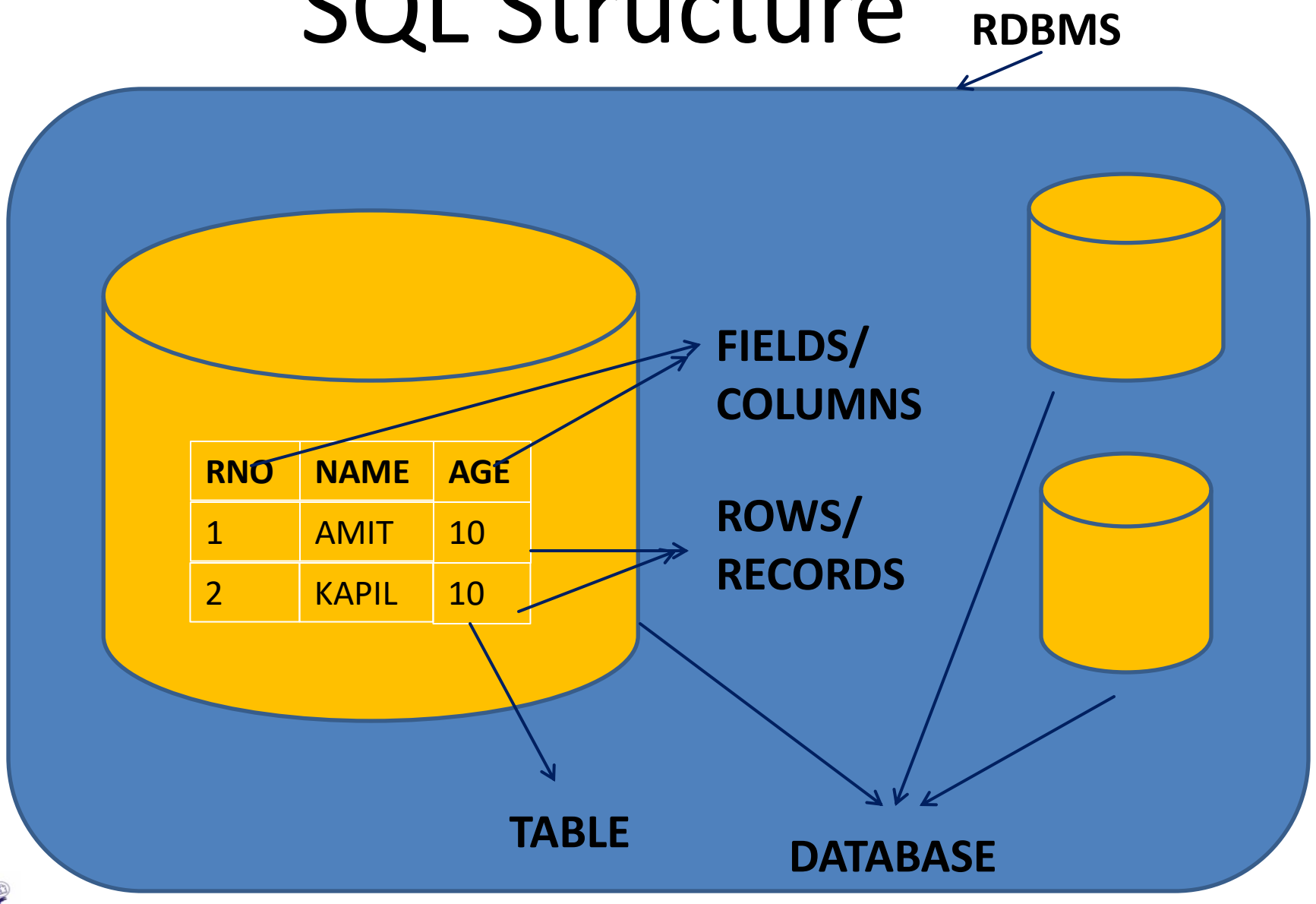
SQL is structured Query Language which is a computer language for storing, manipulating and retrieving data stored in relational database.

SQL is the standard language for Relation Database System. All relational database management systems like MySQL, MS Access, Oracle, SQL Server uses SQL as standard database language

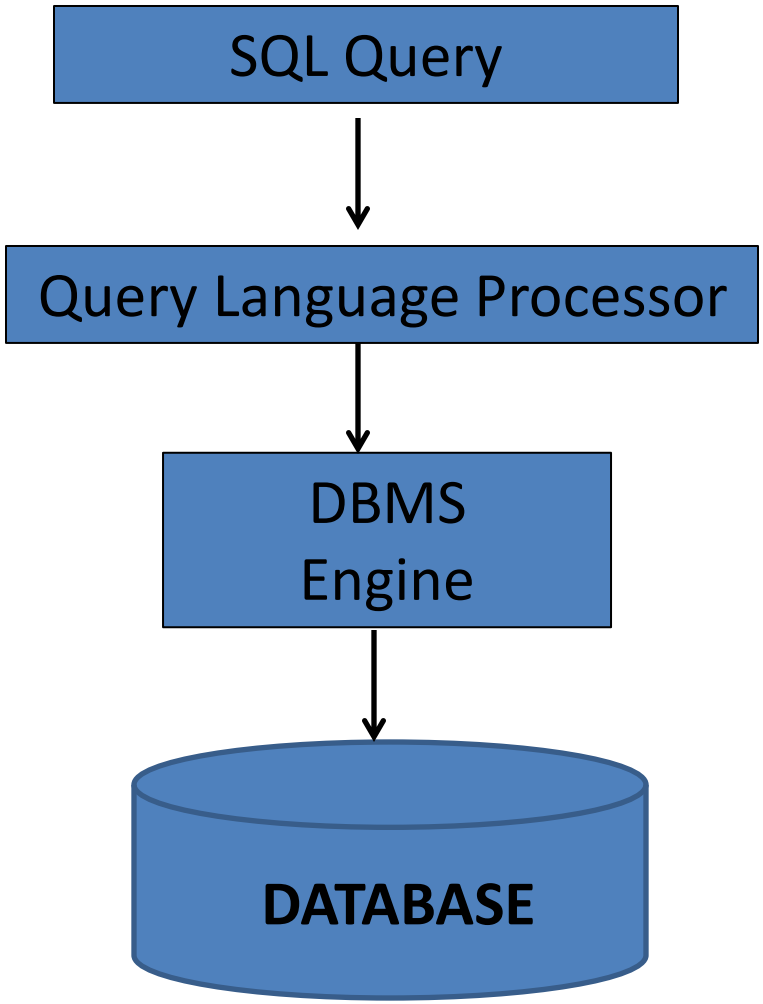
Why SQL?

- Allow users to access data in relational database management systems.
- Allow users to describe the data.
- Allow users to define the data in database and manipulate that data.
- Allow users to create and drop databases and tables.

SQL Structure



SQL Process



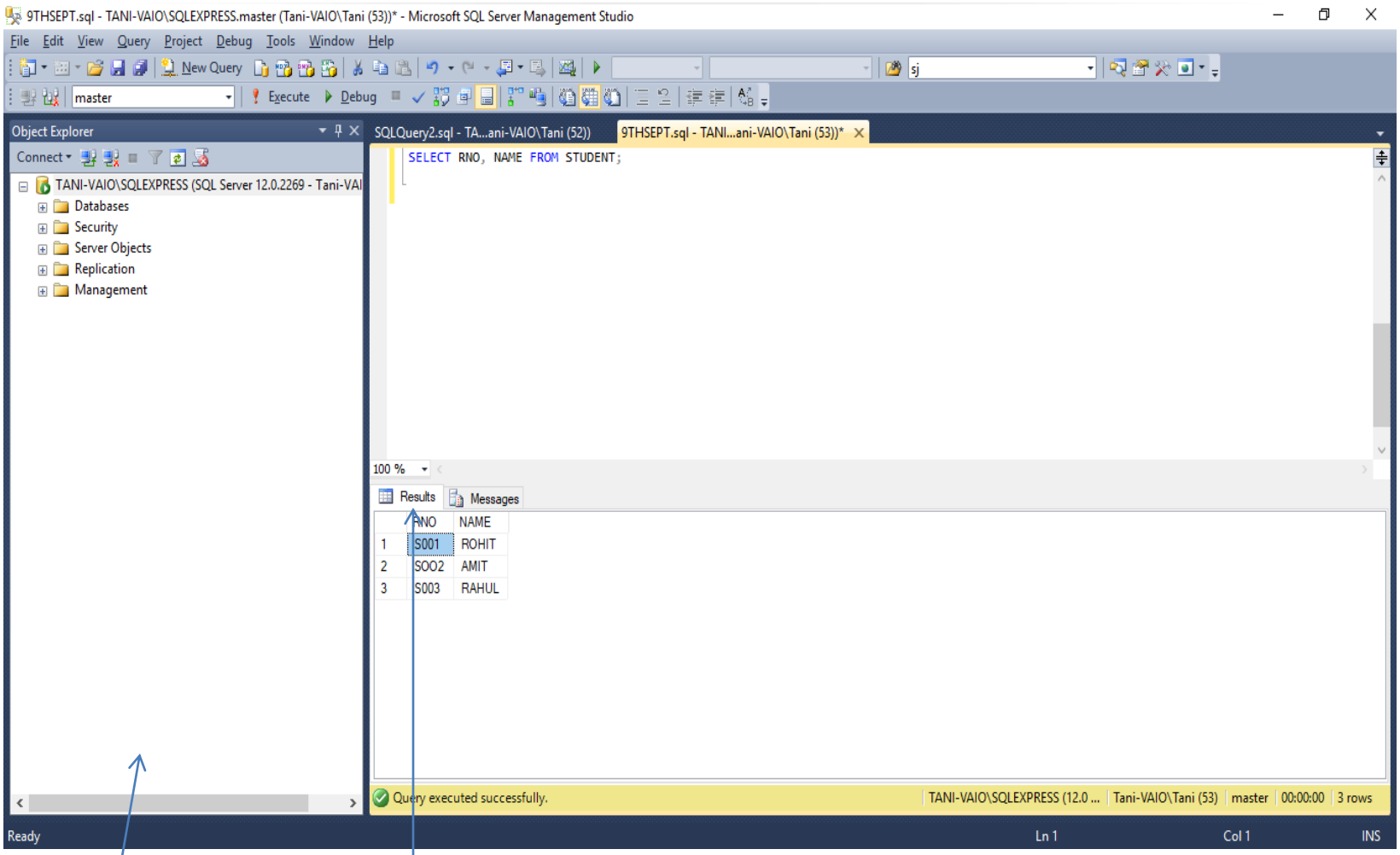
Introduction

- Data
- DataBase
- RDBMS
- SQL

SQL Environment

- Editor
- Result
- Message
- Object Explorer

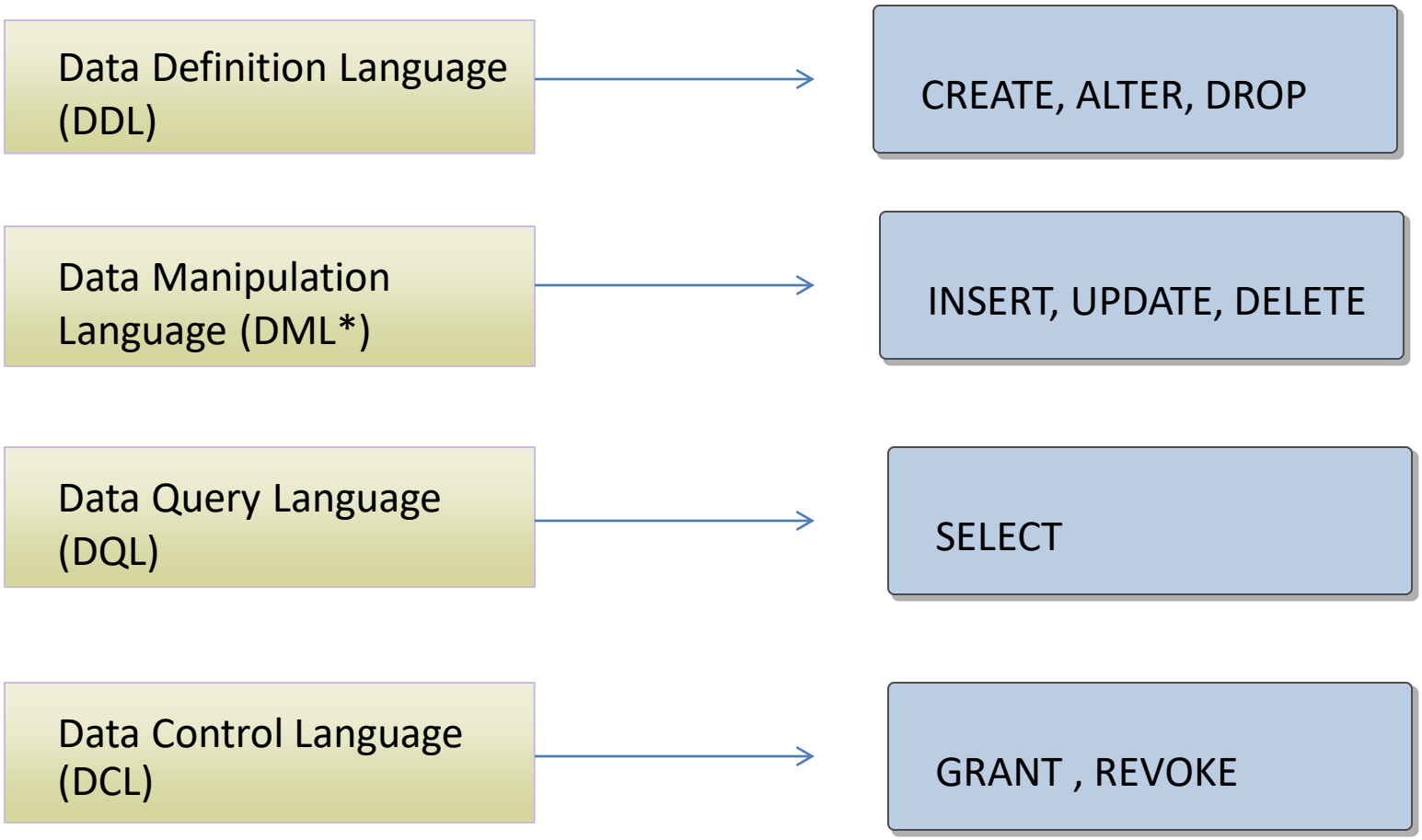
SQL Environment



Object Explorer

Result Tab

Categories of SQL Statements



RDBMS Concepts

ID	NAME	AGE	ADDRESS	SALARY
1	Ramesh	32	Ahmedabad	2000.00
2	Khilan	25	Delhi	1500.00
3	kaushik	23	Kota	2000.00
4	Chaitali	25	Mumbai	6500.00
5	Hardik	27	Bhopal	8500.00
6	Komal	22	MP	4500.00
7	Muffy	24	Indore	10000.00

Create Statement

CREATE DATABASE Statement

CREATE DATABASE <DATABASE EID>;

create database test;

use test;

Create Statement

CREATE TABLE Statement

CREATE TABLE table_EID(
column1 datatype (Size),
column2 datatype (Size),
column3 datatype,
.....
columnN datatype);

Create Statement

CREATE TABLE Statement

```
CREATE TABLE EMP (
    RNO char (5),
    EID CHAR (20),
    AGE INT ,
    CLASS CHAR(10),
);
```

INSERT Statement

- INSERT INTO table_EID (column1, column2....columnN) VALUES (value1, value2....valueN);

- INSERT INTO TABLE_EID VALUES (value1,value2,value3,...valueN);

SQL SELECT Statement

- SELECT column1, column2....columnN FROM table_EID

- SELECT * FROM table_EID;

SQL Data Types

Numeric Data Types:

DATA TYPE	FROM	TO
Int	-2,147,483,648	2,147,483,647
Small Int	-32,768	32,767
Decimal	-10^38 +1	10^38 +1
Money	-922,337,203,685,477.5808	+922,337,203,685,477.5807
Float	-1.79E + 308	1.79E + 308

Character Data Types:

DATA TYPE	FROM	TO
Char	Char	Maximum length of 8,000 characters.
Varchar	Varchar	Maximum length of 8,000 characters.
Text	text	maximum length of 2,147,483,647 characters.

SQL Data Types

Date & Time Data Types:

DATA TYPE	FROM	TO
Datetime	Jan 1, 1753	Dec 31, 9999
smalldatetime	Jan 1, 1900	Jun 6, 2079
Date	Stores a date like June 30, 1991	
Time	Stores a time of day like 12:30 P.M.	

Misc Data Types:

DATA TYPE	FROM	TO
Image	Maximum length of 2,147,483,647 bytes. (Variable length Binary Data)	

- Table creation
- Inserting data
- Verifying the data

ASSIGNMENT - 1

Create a database DEMO

Create table EMP with the following fields:

EID NAME ADDR CITY DOB PHONE EMAIL id should be like 'E0001'

Insert 10 appropriate records in the Emp table

Use SELECT command to view the contents of emp table

Installing SQL Server

Click SQL Server 2014 to download SQL Server 2014

Check the below two files from the list:

SERVER File: **Express 64BIT\SQLEXPR_x64_ENU.exe**

MANAGEMENT STUDIO File: **MgmtStudio 64BIT\SQLManagementStudio_x64_ENU.exe**

Click on NEXT the download will start automatically.
Once downloaded first install the Server file and then Management Studio file.
The installation wizard is simple you just need to follow on screen instructions.

Download link : https://www.microsoft.com/en-in/download/details.aspx?id=42299

SQL CLAUSES

SQL WHERE Clause:

SELECT column1, column2....columnN FROM table_EID WHERE CONDITION;

SQL LIKEClause:

SELECT column1, column2....columnN FROM table_EID WHERE column LIKE 'XXXX%‘

SELECT FROM table_EID WHERE column LIKE 'XXXX_'

There are two wildcards used in conjunction with the LIKE operator:

1. The percent sign (%)
2. The underscore (_)

SQL TOP Clause:

SELECT TOP number|percent column_EID(s) FROM table_EID WHERE [condition]

SQL UPDATE Statement:

```
UPDATE table_EID
SET column1 = value1, column2 = value2
    ....columnN=valueN
[ WHERE  CONDITION ];
```

SQL DELETE Statement:

- DELETE FROM table_EID WHERE {CONDITION};
- DELETE FROM table_EID
- DELETE table_EID

SQL ALTER TABLE Statement:

- ALTER TABLE table_EID
  ADD
  column_EID {data_type};

- ALTER TABLE table_EID
  DROP Column
  column_EID ;

- ALTER TABLE table_EID
  ALTER Column
  column_EID {data_type};

SQL DROP TABLE Statement:

- DROP TABLE table_EID;
- DROP DATABASE database_EID;

SQL TRUNCATE TABLE Statement :

TRUNCATE TABLE table_EID;

SQL COMMIT Statement:

```
COMMIT;
```

SQL ROLLBACK Statement :

```
ROLLBACK;
```

SQL OPERATORS

SQL Operators

- An operator is a reserved word or a character used primarily in an SQL statement's WHERE clause to perform operation(s), such as comparisons and arithmetic operations.
- Operators are used to specify conditions in an SQL statement

Type of Operators

- Arithmetic Operators (+, -, /, *, %)
- Comparison Operators (=, <>, !=, >, <, >= ,<=, !>, !<)
- Logical Operators (AND, OR, NOT)
- Other Operators (BETWEEN , IN, LIKE, IS NULL, DISTINCT, EXISTS

- Table creation
- Inserting data
- Verifying the data

ASSIGNMENT – 2

In the DEMO database create table EMP _SAL with the following fields:

- **EID DEPT DESI DOJ SALARY**
- **Insert 7 appropriate records in the EMP _SAL table**
- **Use SELECT command to view the contents of EMP _SAL table**

From the EMP table list all the employees with last name as Sharma.

Increase the salary of all Managers by 10%

NORMALIZATION

Normalization

Database normalization is the process of efficiently organizing data in a database. It is a set or rules/ guidelines / statements that we follow while storing the data.

There are two reasons of the normalization process:

- Eliminating redundant data, for example, storing the same data in more than one tables.
- Ensuring data dependencies make sense.

First Normal Form (1NF)

- Define the data items. This means looking at the data to be stored, organizing the data into columns, defining what type of data each column contains, and finally putting related columns into their own table.

- Ensure that there are no repeating groups of data

- Ensure that there is a primary key.

Second Normal Form (2NF)

- It should meet all the rules for 1NF
- There must be no partial dependences of any of the columns on the primary key

Third Normal Form (3NF)

- It should meet all the rules for 2NF
- Tables should have relationship.

CONSTRAINTS

SQL Constraints:

Constraints are the rules enforced on data columns on table. These are used to limit the type of data that can go into a table. This ensures the accuracy and reliability of the data in the database.

Following are commonly used constraints available in SQL:

- PRIMARY Key: Uniquely identified each rows/records in a database table.
- UNIQUE Constraint: Ensures that all values in a column are different.
- NOT NULL Constraint: Ensures that a column cannot have NULL value.
- DEFAULT Constraint : Provides a default value for a column when none is specified. CHECK Constraint: The CHECK constraint ensures that all values in a column satisfy certain conditions.
- FOREIGN Key: Uniquely identified a rows/records in any another database table.

NOT NULL Constraint:

By default, a column can hold NULL values. If we do not want a column to have a NULL value then we need to define such constraint on this column specifying that NULL is now not allowed for that column.

```
CREATE TABLE SALESS(
    ID   INT            NOT NULL,
    EID VARCHAR (20)     NOT NULL,
    AGE  INT            NOT NULL,
    ADDRESS  CHAR (25) ,
    SALARY   DECIMAL (18, 2)
);

ALTER TABLE SALESS
  ALTER COLUMN SALARY  DECIMAL (18, 2) NOT NULL;
```

DEFAULT Constraint:

The DEFAULT constraint provides a default value to a column when the INSERT INTO statement does not provide a specific value

```
CREATE TABLE SALESS(
    ID   INT            NOT NULL,
    EID VARCHAR (20)     NOT NULL,
    AGE  INT            NOT NULL,
    ADDRESS  CHAR (25) ,
    SALARY   DECIMAL (18, 2) DEFAULT 5000.00
);

ALTER TABLE SALESS
  ADD CONSTRAINT DSAL DEFAULT 5000.00 FOR SALARY;

ALTER TABLE SALESS
 DROP CONSTRAINT DSAL;
```

UNIQUE Constraint:

The UNIQUE constraint provides a unique value to a column.

```
CREATE TABLE SALESS(
    ID   INT            NOT NULL,
    EID VARCHAR (20)     NOT NULL,
    AGE  INT            NOT NULL UNIQUE,
    ADDRESS  CHAR (25) ,
    SALARY   DECIMAL (18, 2) DEFAULT 5000.00
);

ALTER TABLE SALESS
  ADD CONSTRAINT <CONSTRAINT EID > UNIQUE (AGE);

ALTER TABLE SALESS
  ADD CONSTRAINT myUniqueConstraint UNIQUE(AGE, SALARY);

ALTER TABLE SALESS
  DROP CONSTRAINT myUniqueConstraint;
```

CHECK Constraint:

The CHECK Constraint enables a condition to check the value being entered into a record. If the condition evaluates to false, the record violates the constraint and it s not entered into the table.

```
CREATE TABLE SALESS(
    ID   INT            NOT NULL,
    EID VARCHAR (20)     NOT NULL,
    AGE  INT            NOT NULL  CHECK (AGE > 18),
    ADDRESS  CHAR (25) ,
    SALARY   DECIMAL (18, 2) DEFAULT 5000.00
);

ALTER TABLE SALESS
  ADD CONSTRAINT ckAge CHECK (AGE > 18);

ALTER TABLE SALESS
  DROP CONSTRAINT ckAge;
```

PRIMARY KEY Constraint:

A primary key is a field in a table which uniquely identifies the each rows/records in a database table. Primary keys must contain unique values. A primary key column cannot have NULL values.

A table can have only one primary key which may consist of single or multiple fields. When multiple fields are used as a primary key, they are called a composite key.

```
CREATE TABLE SALESS(
    ID   INT            NOT NULL,
    EID VARCHAR (20)     NOT NULL,
    AGE  INT            NOT NULL,
    ADDRESS  CHAR (25) ,
    SALARY   DECIMAL (18, 2) ,
    PRIMARY KEY (ID)
);

ALTER TABLE SALESS
  ADD CONSTRAINT pkID  PRIMARY KEY (ID);

ALTER TABLE SALESS
  DROP CONSTRAINT pkID;
```

FOREIGN KEY Constraint:

A foreign key is a key used to link two tables together. This is sometimes called a referencing key.

```
CREATE TABLE SALESS(
     ID   INT            NOT NULL,
     EID VARCHAR (20)     NOT NULL,
     AGE  INT            NOT NULL,
     ADDRESS  CHAR (25) ,
     PRIMARY KEY (ID)
);

CREATE TABLE ORDERS(
     OID   INT            NOT NULL,
      CUST_ID INT REFERENCES SALESS (ID),
      ODATE  DATE,
      QTY INT,
      PRICE INT
);

ALTER TABLE ORDERS
ADD CONSTRAINTS  FKID  FOREIGN KEY (CUST_ID) REFERENCES  SALESS (ID);
```

ASSIGNMENT

ASSIGNMENT – 3

CREATE TWO TABLES EMP & EMP_SAL AS PER THE BELOW STRUCTURE:

EMP	
Field EID	Constraints
EMPID	Primary Key
NAME	NOT NULL
ADDR	No employee from UTTAM NAGAR
CITY	DEL, GGN, FBD, NOIDA
PHNO	UNIQUE
EMAIL	Should be on Gmail / Yahoo Domain
DOB	<= '1-Jan-1990'

EMP_SAL	
Field EID	Constraints
EMPID	Foreign Key
DEPT	HR, MIS, OPS , IT ADMIN, TEMP
DESI	ASSO, MGR, VP, DIR
BASIC	>=20000
DOJ	-

By default DEPT should be TEMP

CLAUSES

SQL CLAUSES

SQL BETWEEN Clause

SELECT column1, column2....columnN FROM table_EID WHERE column_EID BETWEEN val-1 AND val-2;

SQL IN Clause

SELECT column1, column2....columnN

FROM table_EID

WHERE column_EID IN (Val1, Val2... Valn);

SQL Like Clause

SELECT column1, column2....columnN FROM table_EID WHERE column_EID LIKE { PATTERN}

SQL COUNT Clause

SELECT COUNT(column_EID) FROM table_EID WHERE CONDITION;

SQL DISTINCT Clause

SELECT DISTINCT (column) FROM table_EID;

SQL CLAUSES

SQL ORDER BY Clause

```
SELECT column1, column2....columnN
FROM   table_EID
WHERE  CONDITION
ORDER BY column_EID {ASC|DESC};
```

SQL GROUP BY Clause

```
SELECT SUM(column_EID)
FROM   table_EID
WHERE  CONDITION
GROUP BY column_EID;
```

SQL HAVING Clause

```
SELECT SUM(column_EID)
FROM table_EID
WHERE  CONDITION GROUP BY column_EID
HAVING (arithematic function condition);
```

ASSIGNMENT – 4

In the EMP table display :

CITY WISE COUNT OF EMPLOYEES ARRANGED IN DESCENDING ORDFER

DETAILS OF THE EMPLOYEES WHO DOES NOT HAVE AN ACCOUNT ON YAHOO DOMAIN

From the Emp_Sal table display:

DESIGNATION WISE TOTAL COST AND NUMBER OF MEMBERS ARRANGED IN DESCENDING ORDER OF THE TOTAL COST