

# SQL SERVER By Kannababu

## SQL Server Introduction

### Q) What is Data?

- Data is a Collection of Raw Facts.
- Data can exist in many forms like numeric, character, Audio file, Video file.

### Q) What is Database?

A database is a collection of information that is organized so that it can be easily accessed, managed and updated.

EX: MS SQL Server, Oracle, My SQL.

### Q) What is SQL?

A) SQL is a Query Language, which is used to interact with Database.

### Q) What is DBMS?

A) Database Management System is a software (Collection of programs) which enables the users to access (or) interact with the Database.

### Q) What is RDBMS?

A) Relational database management system (RDBMS) is a type of database management system (DBMS) that stores data in the form of related tables.

All modern database management systems like MYSQL, MS SQL Server, IBM DB2, ORACLE, ...

### Q) What is Data type?

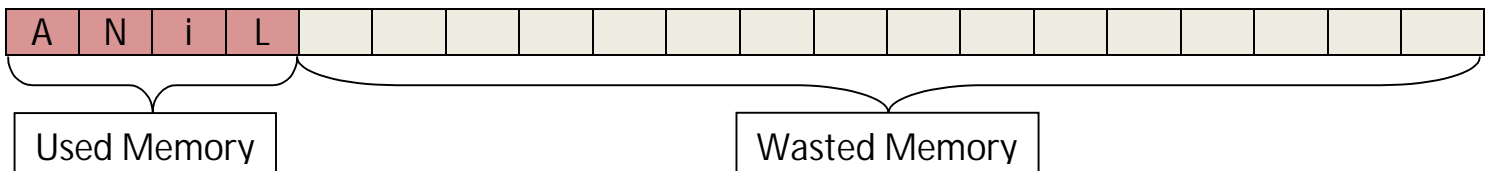
A) Data type specifies the type of data that we store in the memory.

### Note:

The default data type value of any SQL Server data type is Null.

A) Char(size):-

Ex:- if we declare `sname char(100)` then 100 characters of memory is allocated if we store Anil then only 4 characters of memory is used and the remaining 96 characters of memory is wasted.



- The maximum number of characters that we can store in char datatype is 4000 characters.
- Char does not support national characters data like French, Japanese etc.

nchar(size):-

- `nchar` allows national characters data.

varchar(size):-

The size of varchar is not fixed it is variable length.

- If we declare `varchar(100)` and if we store Anil then varchar will allocated only 4 characters of memory and remaining 96 characters of memory is not allocated(not wasted).
- The maximum number of characters that we store in varchar is 8000 characters.

- We cannot store national characters in varchar.

nvarchar(size):-

nvarchar is similar like varchar. But it supports national characters data.

**Note:** When we know the size of the data, better to choose "char" and if we don't know the range better to choose "varchar".

SQL Server is a database which is used to store data permanently.

### Q) What is SQL?

A) SQL is pronounced as Sequel Structured English query language.

- While installing SQL server software 2 tools will be installed automatically.
  1. Database Server
  2. Client Application (MS SQL Management Studio).

**Database Server:** It is a software which is used to store the data.

- Database server will automatically run by O.S
- Database server is an backend application. So, we cannot directly interact with Database server.
- Whenever the O.S is started then automatically SQL Server (Database Server) will start by O.S  
Go to → Start → Run → Services. MSc.  
SQLSERVER (MS-SQL)-Started.

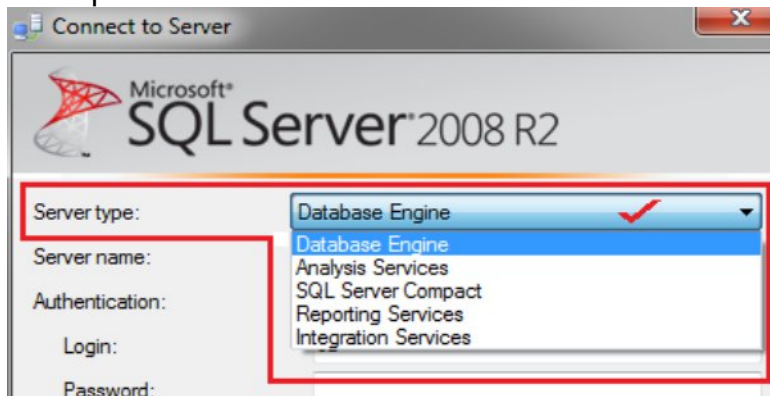
**Client Application:** Whenever we install SQL Server then SQL Server Management Studio was installed which is called as client application.

- Client Application is a frontend application.
- It is a software that was given by Microsoft in order to interact with SQL Server Database.

Version	Year	Release Name
1.0	1989	SQL Server 1.0
1.1	1991	SQL Server 1.1
4.2.1	1993	SQL Server 4.2.1
6.0	1995	SQL Server 6.0
7.0	1998	SQL Server 7.0
8.0	2000	SQL Server 2000
9.0	2005	SQL Server 2005
10.0	2008	SQL Server 2008
10.5	2010	SQL Server 2008R2
11.0	2012	SQL Server 2012
12.0	2014	SQL Server 2014
13.0	2016	SQL Server 2016

## Steps to Open SQL Server Management Studio:

- Go to → Start → Programs → Microsoft SQL Server Management Studio Then a dialog box opens.



- Server name = Computer name/IP address/local /./SQLEXPRESS



- Here we can use either SQL Server Authentication/ Windows Authentication

**Server type:** This Property is used to select the service that we want to work.

**SSIS:** SQL Server Integration services. It is used to integrate different data Sources, using a data warehouse tool. ETL() (Extract/Transfer/Load).

**SSAS:** SQL Server Analysis Services. It is used to analyze data.

**SSRS:** SQL Server Reporting Services. → To report.

**Server name:** It is used to mention the name of System where Database server is installed.

- Server name may be Computer name/ IP Address etc.

**Authentication:** It is a process of checking user credentials.

- User credentials means username and password.

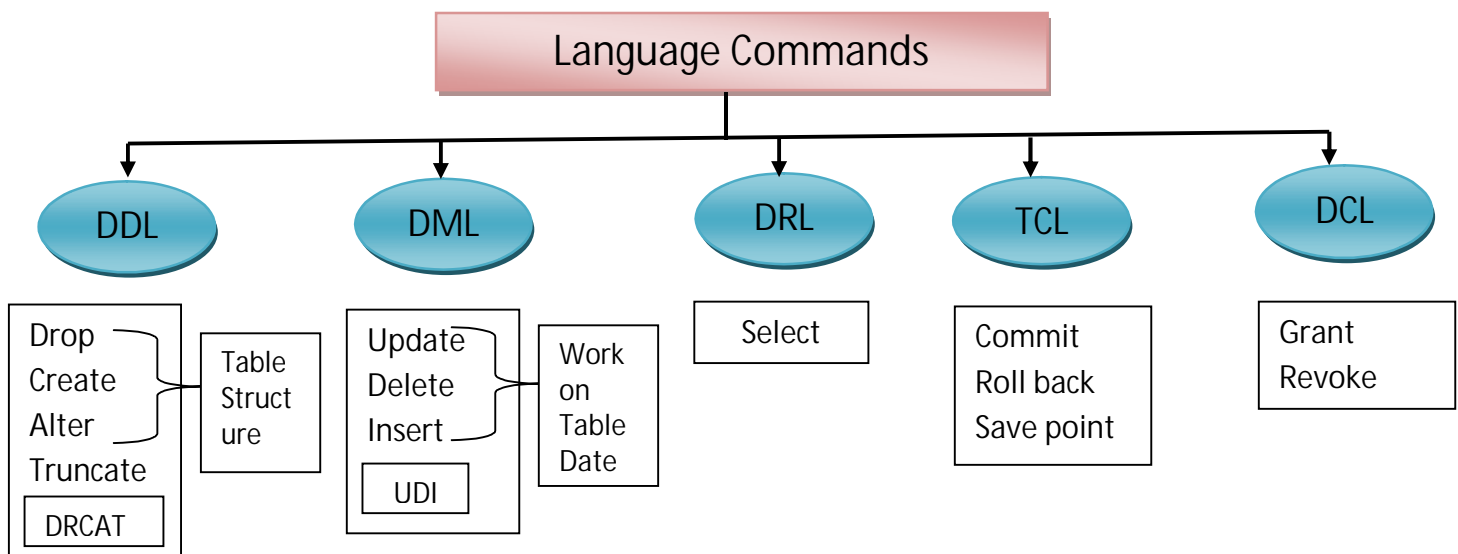
- Any user who is having username and password is called as Authenticated user.
- We can connect to SQL Server database in two ways.
  1. Windows Authentication.
  2. SQL Server Authentication.

**Windows Authentication:** When we connect to SQL Server database using windows Authentication it is not required to give username and password because by default we are the valid users of windows O.S.

**SQL Server Authentication:** When we connect to SQL server database using SQL Server Authentication then we need to mention username and password.

Default username: sa

Password in Sathya: abc.



**DDL:** Data Definition Language Commands.

**DML:** Data Manipulation Language Commands.

**DRL:** Data Retrieval Language Commands.

**TCL:** Transaction Control Language Commands.

**DCL:** Data Control Language Commands.

- Language Commands are used to perform operations on Database objects like Tables, Views, Synonyms, Stored Procedures and Triggers etc.

## Data Definition Language Commands (DDL):

These Commands are used to create (or) modify database (or) Database objects like Tables, Views etc.

### Q) What is Database?

A) Database is Collection of related data which is organizes in a proper manner.

Databases are of 2 types.

1. System defined database
2. User defined database.

1. **System defined database:** The database that was created automatically when we install SQL server management studio are called as System defined database.

- Different types of System defined databases

- a) Master
- b) Model
- c) Tempdb
- d) MSdb

a) **Master:** Master database is used to maintain the configuration settings that are required to run SQL Server Management Studio.

- Master database consists of user information like username, password, server name etc...

b) **Model:** Model database is a template for all the userdefined databases.

- Whenever we create a userdefined database, then the contents of the model database will be copied into userdefined database.

c) **Tempdb:** (Temporary) it is used to store temporary Calculations.

d) **MSdb:** Microsoft database

- It is used to schedule jobs and alters.

### Userdefined database:

The database that was created depending on the user requirement is called as userdefined database.

#### Note:

- DDL Commands are used to work on Table Structure.
- DML Commands are used to work on Table data.

#### Syntax to create database:

create database databasename

Ex: **createdatabase mydb**

#### Syntax to use the created database:

usedatabasename

Ex: **use mydb**

- If the database is not used then all the tables will be stored in master or some other db.

Q) Write a Query to view the information of db.

A) **sp\_helpdb** databasename

Ex: **sp\_helpdb mydb**

- When we execute the above Query the entire information of the database like databaseid, databasename, creationtime, dbsize will be displayed.

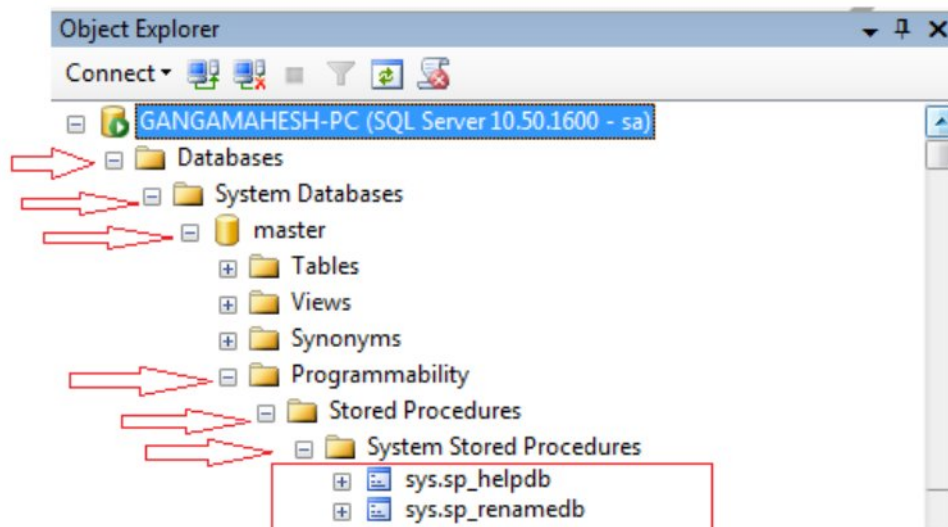
Q) Write a Query to rename the database?

A) **sp\_renamedb** old databasename, newdatabase name

Ex: **sp\_renamedb mydb, mydb1**

\*\*\* **sp\_helpdb, sp\_renamedb** are predefined stored procedures.

Go to → View → Object explorer



Q) Write a Query to drop the database?

A) **drop database** databasename

Ex: **drop database mydb1**

- Whenever we drop the database then the database will permanently removed from hard disk.

**Note:** We cannot drop the database if the database is currently in use.

**Table:** Table is a database object which is used to maintain the database in the form of rows and columns.

eno	ename	salary
101	Anil	20000.00
102	Sunil	25000.00

Column name

Row is also called as record

Cell



- Whenever we store the data in table we have to follow 2 rules

Rule1: Single cell Consists of Single value.

Rule2: Each record must be Unique.

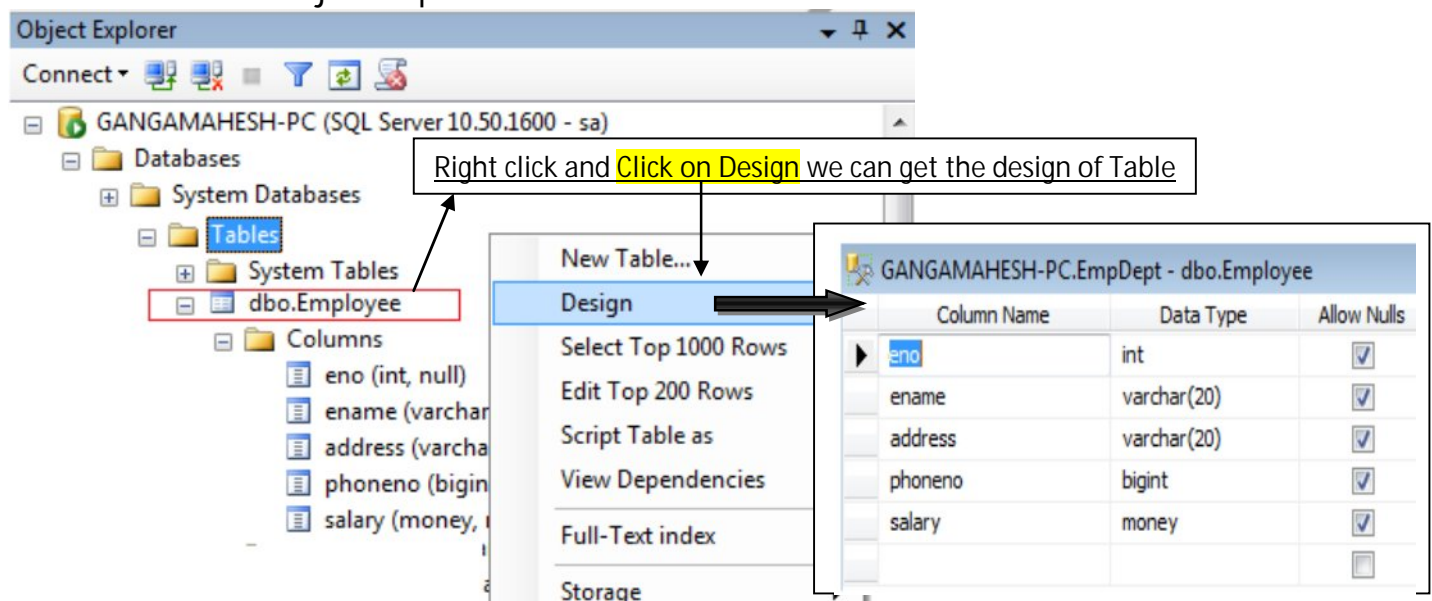
### Syntax to create a table:

Create table tablename(colnamedatatype, colnamedatatype)

Ex: **createtable Employee(enno int, ename varchar(20), addressvarchar(20), phoneno bigint, salary money)**

Ex:for Table(How to view Table)

Go to→ View→+Object Explorer



### Q) Write a Query to View the information of table?

A) syntax :sp\_helptablename

Ex: **sp\_help Employee**

- When we create the above query the entire information of the table will be displayed like table name, table creation time, column names, data types, size of the data type etc...

### Q) Write a Query to rename the tablename?

A) syntax: sp\_renameoldtablename, newtablename

Ex: **sp\_rename Employee, Emp**

**Alter Command:** This command is used to add a new column for the existing table.

- To remove the column from the table.
- To change the datatype of the column.



Q) Create a table with name Student with Columns sno, sname?

A) **createtable student(sno int, sname varchar(20))**

Q) Write a Query to add a new column with name address in Student table?

A) syntax: alter table tablename add columnnamedatatype

Ex: **altertable student addaddressvarchar(20)**

Q) Write a Query to change the datatype of address column from varchar(20) to varchar(50)?

A) syntax: alter table tablename alter column columnname newdatatype

Ex: **altertable student altercolumnaddressvarchar(50)**

Q) Write a Query to change the Column name of current table.

A) syntax: sp\_rename 'tablename.oldcolname', 'new colname'

Ex: **sp\_rename 'employee.eno', 'empno'**

**Drop Command:** It is used to remove the database or database objects like table or columns in tables permanently.

Q) Write a Query to remove or drop the address column in Student Table?

A) syntax: alter table tablename drop column columnname

Ex: **altertable student dropcolumnaddress**

Q) What is the difference between Truncate and drop?

A) Truncate will remove the table data but not the structure of the table.

- Drop will remove the table data along with the structure of the table.

**Note:** Once when we drop the table we cannot insert the record. But once when we truncate the table we can insert the record.

Syntax: truncate table tablename  
drop table tablename.

DML Commands: insert, update, delete

**Insert:** It is used to insert the record within the table.

Syntax1: insert into tablename values(value1, value2)

**Note:** The number of values that we insert must match the number of columns in a table.

- The order of the values that we pass must match with the order of columns.
- The type of the values that we pass must match with the type of columns.

Ex:

- **use** Sathya
- **createtable** employee(eno **int**,ename **varchar**(20), salary **money**)
- **insertinto** employee **values**(101,'Anil',2000)
- **select\*from** employee

Syntax2:

insert into tablename(columnlist) values (value list)

Ex: To insert specific values in employee table

- **insertinto** employee(eno,ename,salary)**values** (102,'Sunil',21000)
- **select\*from** employee

Syntax3: To insert multiple values in employee table

Insert into tablename(column list)

Select value list

Union all

Select value list

Ex:

```
insertinto employee(eno,ename,salary)
select 103,'Ajay',24000
unionall
select 104,'Vijay',30000
unionall
select 105,'Jhon',31000
```

syntax 4:

insert into table name(columnlist) values(valuelist1),(valuelist2),(valuelist3)

Ex:

- **insertinto** employee(eno,ename,salary)**values**(106,'kavitha',30000),(107,'Revathi',30000),(108,'Hari',31000)
- **select\*from** employee

**Delete Command:** This Command is used to delete a specific record or all the records from the table.

Syntax: to delete all records from the table

Delete from tablename

Ex:  
**deletefrom employee**

Syntax: to delete a specific record from the table.  
Delete from tablename where condition

Ex:  
**deletefrom employee where eno=101**

Q) What is the difference between delete and truncate?

A)

Truncate	Delete
1. It is DDL Command.	1. It is DML Command.
2. Truncate is used to delete all the records from the table.	2. Delete is used to delete all the records or specific Records from the table.
3. Truncate will delete all the records at a time.	3. Delete will delete the records page by page.
4. Truncate will work fast compare with delete.	4. Delete will work slow.

**Update Command:** It is used to update the table data based on where condition.

Q) Write a Query to update employee salary to 25000 whose employee number is 101?

A) syntax: update tablename set column = value where condition

Ex: **update employee set salary=25000 where eno=101**

Q) Write a Query to update employee name from Anil to Anilkumar?

A) **update employee set ename='Anilkumar' where ename='Anil'**

Q) Write a Query to update employee name and salary whose eno is 101?

A) **update employee set ename='anilkumar', salary=30000 where eno=101**

### DRL Commands:

**Select Command:** This Command is used to display all the records or specific records from the Table.

Select \* from tablename

- The above Query will display the all the records from the table.

**select\*from employee where 1=1**

**if(1=1)**

**select\*from employee**

Q) Write a Query to display ename and salary?

A) `select ename,salary from employee`

Q) Write a Query to display employee details whose eno is 101?

A) `select*from employee where eno=101`

Q) Write a Query to display salary of Anil?

A) `select salary from employee where ename='anil'`

## Operators:

Operators are used to perform operation on two (or) more operands.

Different types of Operators in SQL Server are:-

1. Arithmetic Operators: +, -, \*, /, %
2. Relational Operators: <, >, <=, >=, !=, =
3. Logical Operators: and, or
4. Ranging Operators: between, not between.
5. String Operators: Like, not like.

1. **Arithmetic Operators:** These Operators are used to perform operations like Addition, Subtraction, Multiplication, Division etc.

Q) What is an Expression?

A) An Expression is the combination of 2 or more Operators.

Ex:  $2+3-4*6+7$

- Whenever we solve an expression we have to solve based on the priority of the operator.
- 1<sup>st</sup> Priority: \*, /, %
- 2<sup>nd</sup> priority: +, -
- 3<sup>rd</sup> priority: =

Q) What is the difference between / and %?

A) '/' will give Quotient.

'%' will give us Remainder.

Ex:  $7/3 = 3$  7(2) → Quotient  
6  
1 → Remainder

**Note:** In SQL Server any datatype + string is "error".

- If the Numerator is less than Denominator then Quotient is "Zero" and remainder is "Numerator".

Ex:  $3/5=0$ ,  $7/9=0$ ,  $4\%5=4$ ,  $7\%9=7$

Ex1:  $3*5-7/3+7*3-4/6+5*3$   
 $=15-2+21-0+15$   
 $=15-2+21+15=49$

Ex2:  $6/7-7\%11+7*7-3/4-2$   
 $0-7+49-0-2=40$

Q) Create a table with name Student with columns sno, sname, m1, m2, m3?

A) **createtable** Student(sno **int**, sname **varchar**(50), m1 **int**, m2 **int**, m3 **int**)

sno	sname	m1	m2	m3
101	Anil	77	78	79
102	Sunil	70	80	75

Q) Write a Query to display Student details along with total marks and percentages of marks?

A) **select**\*, m1+m2+m3 **as** 'Total', (m1+m2+m3)/3 **as** 'Percentage' **from** Student

sno	sname	m1	m2	m3	Total	Percentage
101	Anil	77	78	79	234	78
102	Sunil	70	80	75	225	75

- In this above query '**as**' operator is a alias name means Dummy name.

Q) Create a table with name employee with columns eno, ename, bsal?

A) **createtable** employee(enno **int**, ename **varchar**(20), bsal **money**)

Q) Write a Query to display employee details along with da, hra and tsal?

A) bsal=20000  
 20% da on bsal  
 30% hra on bsal  
 $da = 0.2 * 20000$   
 $hra = 0.3 * 20000$

**select**\*, 0.2\* bsal **as** 'da', 0.3\* bsal **as** 'hra', (bsal+(0.2\*bsal)+(0.3\*bsal)) **as** 'total' **from** employee

eno	ename	bsal	da	hra	total
111	Shiva	20000.00	4000.00000	6000.00000	30000.00000
112	Vudai	22000.00	4400.00000	6600.00000	33000.00000

Relational operators / Conditional Operators:

- These operators are used to compare 2 or more Values
- Ex:  $-5 > 5$  true  
 $-5 < -7$  false

- A Condition will always return boolean value either true or false.

eno	ename	sal
1	Anil	21000
2	Sunil	22000
3	Ajay	30000

Q) Write a Query to display emp details whose sal>21000?

A) **select\*from emp where sal>21000**

eno	ename	sal
2	Sunil	22000.00
3	Ajay	30000.00

Q) Write a Query to display emp details whose eno=1?

A) **select\*from emp where eno=1**

eno	ename	sal
1	Anil	21000.00

Q) Write a Query to display deleteemp details who is working in IT dept?

A) **delete emp where dname='IT'**

Q) Write a Query to display emp name whose sal>21000?

A) **select ename from emp where sal>21000**

ename
Sunil
Ajay

**Logical Operators:** These Operators are used to Compare two or more Conditions.

C1	C2	and	Or
T	T	T	T
T	F	F	T
F	T	F	T
F	F	F	F

Ex: 10>=-5 and 10<=-5

T F (F)

-5<=-17 or 9<=5+4

F T (T)

Ex2:

eno	ename	sal	address
1	Anil	21000.00	Hyderabad
2	Sunil	22000.00	Vizag
3	Ajay	30000.00	Hyderabad

Q) Write a Query to display emp details whose sal>21000 and who is Staying in Hyderabad?

A) **select\*from emp where sal>21000 andaddress='Hyderabad'**

eno	ename	sal	address
3	Ajay	30000.00	Hyderabad

Q) Write a Query to display emp details whose sal>21000 or Who is Staying in Hyderabad?

A) **select\*from emp where sal>21000 oraddress='Hyderabad'**

eno	ename	sal	address
1	Anil	21000.00	Hyderabad
2	Sunil	22000.00	Vizag
3	Ajay	30000.00	Hyderabad

**Range Operators:** between, not between

- These Operators are used to display data between the range.

Q) Write a Query to display emp details whose salary range between 20000 and 25000?

A) **select\*from emp where sal between 20000 and 25000**

eno	ename	sal	address
1	Anil	21000.00	Hyderabad
2	Sunil	22000.00	Vizag

Q) Example to Write a Query to display empdetails whose age between 18 and 25?

A) **select\*from emp where age between 18 and 25**

Q) Example to Write a Query to display emp details whose age not between 18 and 25?

A) **select\*from emp where age notbetween 18 and 25**



**String Operators:** These Operators are used to display the data in a specific pattern Like, not Like.

**Like:** It is used to display the string in a specific pattern.

- a% starts with a
- %a ends with a
- %a% in between a

**Q) Write a Query to display emp details whose name starts with a?**

**A) `select*from emp where ename like 'a%'`**

eno	ename	sal	address
1	Anil	21000.00	Hyderabad
3	Ajay	30000.00	Hyderabad

**Q) Write a Query to display emp details whose sal>23000 and whose name not starts with s?**

**A) `select*from emp where sal>23000 and ename notlike 's%'`**

eno	ename	sal	address
3	Ajay	30000.00	Hyderabad

### Constraints:

- Constraints are used to apply Conditions or restrictions on the database objects like tables.
- Different types of Constraints are:
  1. Null Constraint
  2. Not null Constraint
  3. Primary key Constraint
  4. Unique key Constraint
  5. Unique + not null Constraint
  6. Foreign key Constraint
  7. Checked key Constraint
  8. Default Constraint
  9. Composite primary key Constraint

**Note:** We have to apply Constraints for the Columns in a table.

#### **1. Null Constraint:**

Null Constraints will allow null values.

Ex: `createtable emp(eno int, ename varchar(20)null)`

**Note:** By default every column in a table will allow null values.

#### **2. Not null Constraint:**

Not null Constraint will not allow null values.

- Syntax: `create table tablename(columnname datatype notnull, columnname datatype notnull)`

Ex: **createtable e1(eno intnotnull, ename varchar(50))**

- Syntax to apply not null constraint after creating the table.
- Syntax: alter table tablename alter column columnname datatype not null

Ex: **altertable e1 altercolumn ename varchar(20)notnull**

### 3. Primary key Constraint:

- Primary key Constraint will not allow duplicate values.
- Primary key Constraint will not allow null values.
- We cannot apply more than one primary key Constraint on a Single table.
- Primary key Constraint will have Constraint name.

#### Q) What is Constraint name?

A) The Constraint name is the name given for the Constraints which is used to identify the Constraints.

#### Q) What is the purpose of Constraint name?

A) The purpose of the Constraint name is to drop the Constraint.

#### Q) Why Primary key Constraint will not allow null values?

A) Whenever we apply Primary key Constraint then automatically not-null Constraint will be applied on Primary key Column and because of not-null Constraint Primary key will not allow null values.

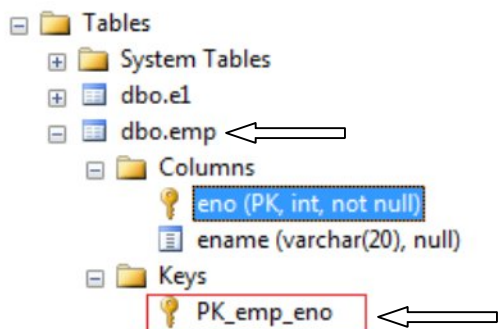
Syntax for applying Primary key Constraints at the time of creating the table

- Syntax: create table tablename(columnname datatype primary key, columnname datatype)

Ex: **createtable emp(eno intprimarykey, ename varchar(20))**

- Syntax2: For applying Primary key Constraint name:  
Create table tablename(column name datatype Constraint Constraintname primary key, Columnnamedatatype)

Ex: **createtable emp(eno intconstraint PK\_emp\_eno primarykey, ename varchar(20))**



- We can apply primary key constraint after creating the table.

- Syntax: alter table tablename add primary key(col name)

**Note:** We cannot apply primary key constraint on nullable column i.e, we cannot apply primary key constraint on the column will allow null values.

- So we have to make columns as not null and then apply primary key.

Ex:

- `altertable emp altercolumn eno intnotnull`
- `altertable emp addprimarykey(eno)`
- Syntax-2: To apply primary key Constraints after creating the table.
- Alter table tablename add constraint constraintname primary key(colname)
- Ex: `altertable emp addconstraint PK_emp_eno primarykey(eno)`
- Syntax-3: to drop the primary key constraint.  
Alter table tablename drop constraint constraintname.  
Ex: `altertable emp dropconstraint PK_emp_eno`

#### 4. Unique key Constrint:

- Unique key Constraint will not allow duplicate values .
- Unique key Constraint will allow null values.
- Unique key Constraint allows only one null value.
- We can apply more than one unique key Constraint on a single table.
- Unique key Constraint will have Constraint name.

**Note:** Unique key Constraint will not allow 2 null values because Unique key will not accept duplicate names.

- Syntax:  
create table tablename(columnname datatype unique)  
Ex: `createtable emp2(eno intunique,ename varchar(20))`
- Syntax-2:  
create table tablename(columnname datatype constraint constraintname)  
Ex: `createtable emp3(eno intconstraint UK_emp2_eno unique(eno),ename varchar(20))`
- Syntax-3: to apply unique key constraint after creating the table.  
Alter table tablename add unique (colname)  
Ex: `altertable emp2 addunique(eno)`
- Syntax-4:  
Alter table table name add constraint constraintname Unique(colname)
- Syntax: To drop Unique key Constraint.  
Alter table tablename drop constraint constraintname

Constraints	Null	Not null	PK	UK	UK+not null
Null Values	✓	✗	✗	✓	✗
Duplicate	✓	✓	✗	✗	✗
No. of Keys	>1	>1	Only One	>1	>1
Constraintname	✗	✗	✓	✓	✓

### 5. Unique+ not null Constraint:

- Does not allow duplicate values
- Does not allow null values
- Can use more than one Uk+not null
- Will have Constraint Name.
- Syntax: create table tablename(columnname datatype unique not null)  
Ex: **createtable emp4(eno intunique notnull,ename varchar(20))**

**Note:** We cannot apply Unique +not null Constraint after creating the table.

### 6. Checked Constraint:

It is used to apply some condition on the column.

- Syntax: create table tablename(colname datatype check(condition))

Q) Create table with name emp with columns eno, ename, salary and apply checked constraint on eno column saying that eno>100?

A) **createtable emp(eno intcheck(eno>100),ename varchar(20),salary money)**

- Syntax2: At the time of creating the table.

Create table tablename(columnname datatype constraint constraintname check(condition))

Q)create a table with name employee with columns eno, ename, age apply checked constraints for age column based on the condition age>=18 and age<=25?

A) **createtable employee(eno int, ename varchar(20),age intconstraint ck\_emp\_eno check(age>=18 and age<=25))**

Syntax to apply checked Constraint after creating the table:

Alter table tablename add constraint constraintnamecheck(condition)

Q) Write a query to apply checked constraint for salary column based on condition salary must be between 10000 to 15000?

A) **altertable emp addconstraint ck\_emp\_salary check(salary between 10000 and 15000)**

Syntax2: to drop cheked constraint

Alter table tablename drop constraint constraintname

Ex: **altertable emp dropconstraint ck\_emp\_salary**

**Default Constraint:** It is used to insert some default values instead of null values.

Syntax:

Create table tablename(colnamedatatype default(default value))

Ex: **createtable attendance(sidint, Datedate, Statusintdefault(0))**

**Composite primary key Constraint:**

It is used to apply primary key constraint for the combination of columns.

**Note:** Whenever we apply primary key constraint for the combination of columns then not-null constraints was applied for the individual columns and that not null constraint will not allow us to insert null values.

- Points to remember:
  - Composite primary key constraint will not allow duplicate values and null values.
  - We cannot apply more than one Composite Primary key Constraint on a single table.
  - But the individual columns in Composite PK Constraint allow duplicate values.
  - Composite PK Constraint will have constraint name.

Syntax:

Create table tablename(colname1 datatype, colname2 datatype Primary key(colname1,colname2))

Ex: **createtable emp(enno int, dno int, ename varchar(20), primarykey(enno, ename))**

Ex:

eno	dno	Ename	
101	10	Anil	
101	20	Sunil	
102	10	Ajay	(Invalid) Duplicate values
102	10	Vijay	
103	null	Ajith	(Invalid) Null Values
null	10	Akhil	

Syntax: To Apply Composite Primary Key after creating table

- Alter table tablename add primary key (col1,col2)
- Alter table tablename add constraint constraintname primary key(col1,col2)

### Normalization:

Normalization is the database design Techniques which is used to reduce Redundancy and Dependency of Data.

- Different normal forms that are available are:

Item Type	Items	Quantity	Price
Pizza	VP,CP	10,20	90,120
Burger	VB,CB	20,10	40,60
Cool drink	Sprite, TUp, Coke	20,30,50	22,25,23
Ice Cream	Bs, V	10,20	25,125
Puff	VP,EP,CP	20,30,50	10,20,30

- The above table is in unnormalized form.

#### 1. Normal Form(1<sup>st</sup>NF):

A table is said to in 1NF if it obeys 2 rules

- Single Cell consists of single value.
- Each record must be unique.

Item Type	Items	Quantity	Price
Pizza	VP	10	90
Pizza	CP	20	120
Burger	VB	20	40
Burger	CB	10	60
Cool drink	Sprite	20	22
Cool drink	TUp	30	25
Cool drink	Coke	50	23
Ice Cream	Bs	10	25
Ice Cream	V	20	125
Puff	VP	20	10
Puff	EP	30	20
Puff	CP	50	30

- Thus the above table in 1NF.

#### 2. 2<sup>nd</sup>NF: A table is said to be in 2<sup>nd</sup> Normal if it obeys the below rules.

- Table must be in 1<sup>st</sup>NF.
- We need to identify the key attributes and Non-key attributes.
- Every Non-key attribute must fully dependent on key attribute.

- iv. If any non-key attribute is not dependent on key attribute then remove the column(attribute) and place in a separate table.
- v. In 2<sup>nd</sup> Normal Form we need to establish the relationship b/w the tables. If relation is there then apply foreign key constraint.
- vi.

**Note:** Basing on Uniqueness and dependency of other columns on the important column, thus important table is identified.

### Q) How to identify key attribute?

- The maximum number of columns must depend on another column.
- In above table Quantity and Price fully dependent on items. So items are key attributes.
- Item type is not depending on items. Remove item types and place in a separate table.

Item Master

tid	type
t1	Pizza
t2	Burger
t3	Cool drink
t4	Ice Cream
t5	Puff

Item

ino	Items	Quantity	Price	tid
1	VP	10	90	t1
2	CP	20	120	t1
3	VB	20	40	t2
4	CB	10	60	t2
5	Sprite	20	22	t3
6	TUp	30	25	t3
7	Coke	50	23	t3
8	Bs	10	25	t4
9	V	20	125	t4
10	VP	20	10	t5
11	EP	30	20	t5
12	CP	50	30	t5

- In Second normal form we have to apply the primary keys and foreign keys.
- In Item Master table tid → primary key
- In Item table ino → primary key  
tid → foreign key
- The primary key of one table visiting to another table will become foreign key

### Q) What is foreign key?

A) Foreign key is used to establish relationship between two or more columns.

### Foreign key Constraint:

Foreign key constraint is used to establish the relationship between two or more tables.

- The Primary key of one table becomes foreign key in another table.
- Points to remember:



- Foreign key constraint will accept duplicate values.
- Foreign key constraint will accept null values.
- We can apply more than one foreign key on a single table.
- We cannot insert the value in foreign key column until the value is available in primary key column.
- We cannot delete the record from primary key column until the record is deleted in foreign key column.
- If we want to delete primary key value we have to delete foreign key value.
- If we want to drop primary key column first we need to remove foreign key constraint.

Syntax to create Foreign key constraint at the time of creating the table:

Create table tablename(colnamedatatype foreign key references  
p.keytablename(p.keycolumnname))

Ex:

- **createtable dept(dno intprimarykey,dname varchar(20))**
- **createtable student(sno int, sname varchar(20),dno  
intforeignkeyreferences dept(dno))**

Syntax2:

Create table tablename(colnamedatatype constraint constraintname foreign key references  
p.keytablename(p.keycolumnname))

Ex:

**createtable student(sno int,sname varchar(20),dno intconstraint  
FK\_PKdept\_dno foreignkeyreferences dept(dno))**

**Note:** The data type of primary key column and foreign key column both must be same

Syntax3:

Syntax to apply foreign key Constraint after creating the table.

Alter table tablename add foreign key (column name) references  
primarykeytableName(primarykeycol name)

Ex: **altertable dept addforeignkey(eno)references emp(eno)**

Identity Column:

It is used to apply Auto Generated number for the column.

Syntax: create table tablename(colnamedatatype identity(starting number, increment by),  
Column name datatype)

Ex: **createtable employee(eno intidentity(101,1),ename  
varchar(20),salary money)**

## Indexes:

An Index in SQL is similar like an index in a book. Indexes are used for fast accessing of data.

Indexes are of 2 types:

1. Clustered Index
2. Non Clustered Index

**1. Clustered Index:** Whenever we apply primary key constraint on a column in a table then automatically clustered index will be applied on primary key column. And that clustered index will arranged the data in ascending order.

- We cannot apply more than one clustered index on a Single table.
- Syntax: Create clustered index indexname on tablename(colname)

Ex: `create clustered index Index1 on employee(eno)`

**2. Non-clustered Index:** Whenever we apply unique key constraint on a column then automatically non-clustered index will be applied.

- We can apply more than one non-clustered index on a single table.
- Non-clustered index will not arrange the data in ascending order.
- Syntax: Create non clustered indexname on tablename(colname)

Ex: `create non clustered index NonIndex1 on employee(eno,ename)`

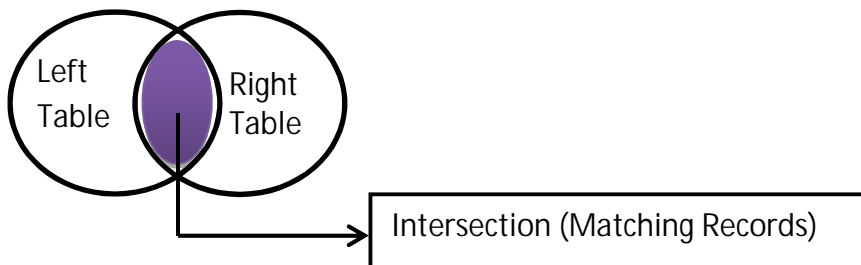
### (\*\*\*)Joins:

Joins are used to retrieve the data from more than one table.

Different types of Joins are:

1. Inner Join
2. Outer Join:
  - i. Left Outer Join
  - ii. Right Outer Join
  - iii. Full Outer Join
3. Cross Join
4. Self-Join
5. Equi Join
6. Non-Equi Join

1. InnerJoin: It is used to retrieve the matching records from both the tables.



Note: While working with Joins there are must be 2 minimum tables must exist and a matching column must exist between both the tables.

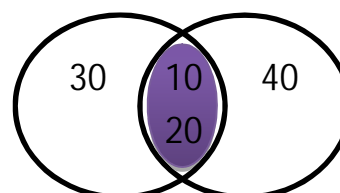
Employee			
Eno	Ename	Salary	Dno
101	Anil	20000	10
102	Sunil	23000	10
103	Ajay	25000	10
104	Kiran	30000	20
105	Vijay	40000	30

Dept	
Dno	Dname
10	IT
20	Maths
40	Physics

Syntax: Select tablename1.colname,tablename2.colname from table1 innerjoin table2 on Table1.common col=table2.common col

If (table1.common col=table2.Common col)

```
{
    Select tablename1.colname,
    Tablename2.colname
}
```



Ex: **select** e.eno,e.ename,e.salary,d.dname  
**from** employee e innerjoin dept d **on** e.eno=d.dno  
 (or)

**select** employee.eno,employee.ename,employee.salary,dept.dname  
**from** employee innerjoin dept **on** employee.eno=dept.dno

Q) Write a Query to display employee details who are working in IT dept ?

A) **select** emp.eno,emp.ename,emp.salary  
**from** employee emp innerjoin dept d **on** emp.dno=d.dno **where**  
 d.dname='IT'

	eno	ename	salary
1	101	Anil	20000.00
2	102	Sunil	23000.00
3	103	Ajay	25000.00

Q) Write a Query to display employee details who are working in IT dept and whose salary>20000?

A) **select** e.eno,e.ename,e.salary  
**from** employee e innerjoin dept d **on** e.dno=d.dno  
**where** d.dname='IT'and e.salary>20000

	eno	ename	salary
1	102	Sunil	23000.00
2	103	Ajay	25000.00

Q) Write a Query to display ename working in Mathsdept?

A) **select** e.ename  
**from** employee e innerjoin dept d **on** e.dno=d.dno  
**where** d.dname='Maths'

	ename
1	Kiran

Q) Write a Query to display employee details who are working in IT dept and whose name starts with a?

A) **select** e.eno,e.ename,e.salary  
**from** employee e innerjoin dept d **on** e.dno=d.dno  
**where** e.ename like 'a%'and d.dname='IT'

	eno	ename	salary
1	101	Anil	20000.00
2	103	Ajay	25000.00

Example:

Employee						Dept	
eno	ename	salary	dno	age	address	Dno	Dname
101	Anil	20000.00	10	20	Hyd	10	IT
102	Sunil	23000.00	10	22	Hyd	20	Maths
103	Ajay	25000.00	10	17	Chennai	40	Physics
104	Kiran	30000.00	20	27	Hyd		
105	Vijay	40000.00	30	26	Hyd		

Q) Write a Query to display employee details who are working in IT dept and whose age is between 18 and 22?

A) **select e.eno,e.ename,e.salary,e.address,e.age**  
**from employee e innerjoin dept d on e.dno=d.dno**  
**where d.dname='IT' and e.age between 18 and 25**

	eno	ename	salary	address	age
1	101	Anil	20000.00	Hyd	20
2	102	Sunil	23000.00	Hyd	22

**Note:** In above Queries we used alias names/dummy names are used (e and d) whereas for employee='e' and for dept='d'

Q) Write a Query to display employee details along with dno, dname who are staying inHyd?

A) **select e.eno,e.ename,e.salary,e.address,e.age,e.dno,d.dname**  
**from employee e innerjoin dept d on e.dno=d.dno**  
**where e.address='Hyd'**

(OR)

**select e.\*,d.dname**  
**from employee e innerjoin dept d on e.dno=d.dno**  
**where e.address='Hyd'**

	eno	ename	salary	address	age	dno	dname
1	101	Anil	20000.00	Hyd	20	10	IT
2	102	Sunil	23000.00	Hyd	22	10	IT
3	104	Kiran	30000.00	Hyd	27	20	Maths

**Note:** In above query '\*' represents to display all details of current table (or) employee.

State	
sid	sname
s1	Telangana
s2	AP
s3	TN
s4	Kerela
s5	Kamataka

City		
cid	cname	sid
1	Hyderabad	s1
2	Vijayavada	s2
3	Vizag	s2
4	Warangal	s1
5	Chennai	s3
6	Banglore	s5
7	Kochi	s4

Q) Write a Query to display State names?

A) **select sname from State**

	sname
1	Telangana
2	AP
3	TN
4	Kerela
5	Kamataka

Q) Write a Query to display City name belongs to Telangana?

A) **select c.cname  
from City c innerjoin State s on c.sid=s.sid  
where s.sname='Telangana'**

	cname
1	Hyderabad
2	Warangal

Employee.e					
eno	ename	age	address	salary	desig
101	Anil	22	Hyd	20000.00	Developer
102	Ajay	23	Hyd	25000.00	Developer
103	Viay	25	Chennai	30000.00	bde
104	Akhil	23	Banglore	40000.00	bdm

Dept.d	
dno	dname
10	IT
20	Marketing
30	Hr
40	Acouns

EmpDept.ed	
eno	dno
101	10
102	10
103	20
104	20



Q) Write a Query to display employee details?

A) `select * from employee`

eno	ename	age	address	salary	desig
101	Anil	22	Hyd	20000.00	Developer
102	Ajay	23	Hyd	25000.00	Developer
103	Viay	25	Chennai	30000.00	bde
104	Akhil	23	Banglore	40000.00	bdm

Q) Write a Query to display ename?

A) `select ename from employee`

	ename
1	Anil
2	Ajay
3	Viay
4	Akhil

Q) Write a Query to display employee details along with dname?

A) `select e.*,d.dname  
from employee e innerjoin empdept ed on e.eno=ed.eno innerjoin  
dept d on ed.dno=d.dno`

	eno	ename	age	address	salary	desig	dname
1	101	Anil	22	Hyd	20000.00	Developer	IT
2	102	Ajay	23	Hyd	25000.00	Developer	IT
3	103	Viay	25	Chennai	30000.00	bde	Marketing
4	104	Akhil	23	Banglore	40000.00	bdm	Marketing

Q) Write a Query to display employee names who are working in IT dept?

A) `select e.ename  
from employee e innerjoin empdept ed on e.eno=ed.eno innerjoin  
dept d on ed.dno=d.dno  
where d.dname='IT'`

	ename
1	Anil
2	Ajay

Q) Write a Query to display employee details along with dname whose age is greater than 22?

A) `select e.*,d.dname  
from employee e innerjoin empdept ed on e.eno=ed.eno innerjoin  
dept d on ed.dno=d.dno  
where e.age>22`



	eno	ename	age	address	salary	desig	dname
1	102	Ajay	23	Hyd	25000.00	Developer	IT
2	103	Viay	25	Chennai	30000.00	bde	Marketing
3	104	Akhil	23	Banglore	40000.00	bdm	Marketing

Q) Write a Query to display employee details who are working in IT dept and whose salary>22000 and whose age is greater than 22?

A) **select e.\*,d.dname**  
**from employee e innerjoin empdept ed on e.eno=ed.eno innerjoin**  
**dept d on ed.dno=d.dno**  
**where d.dname='IT ' and e.salary>22000 and e.age>22**

	eno	ename	age	address	salary	desig	dname
1	102	Ajay	23	Hyd	25000.00	Developer	IT

Q) Write a Query to display employee details along with dname who are staying in Hyd and whose name starts with a?

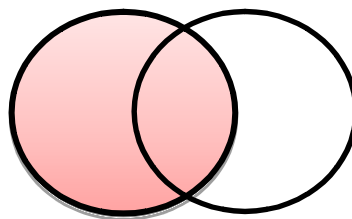
A) **select e.\*,d.dname**  
**from employee e innerjoin empdept ed on e.eno=ed.eno innerjoin**  
**dept d on ed.dno=d.dno**  
**where e.address='Hyd' and e.ename like 'a%'**

	eno	ename	age	address	salary	desig	dname
1	101	Anil	22	Hyd	20000.00	Developer	IT
2	102	Ajay	23	Hyd	25000.00	Developer	IT

## 2. Outer Join:

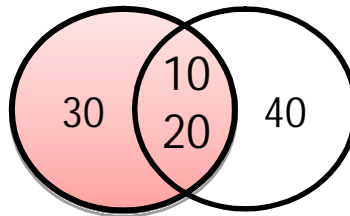
### i. Left Outer Join:

It is used to display all the records from left table and matching records from right table. If there are no matching records in right table then display with null values.



Employee			
eno	ename	salary	dno
101	Anil	20000.00	10
102	Sunil	23000.00	10
103	Ajay	25000.00	10
104	Kram	30000.00	20
105	Vijay	40000.00	30

Dept	
dno	dname
10	IT
20	Maths
40	Physics

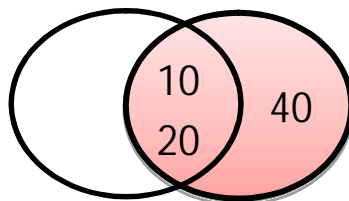
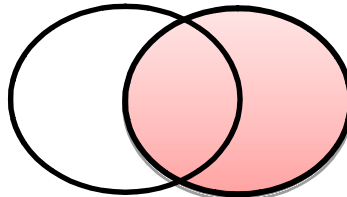


**Query:** `select e.*,d.* from employee e left join dept d on e.dno=d.dno`

Output:

	eno	ename	salary	dno	dno	dname
1	101	Anil	20000.00	10	10	IT
2	102	Sunil	23000.00	10	10	IT
3	103	Ajay	25000.00	10	10	IT
4	104	Kiram	30000.00	20	20	Maths
5	105	Vijay	40000.00	30	NULL	NULL

- ii. **Right Outer Join:** It is used to display all the records from right table and matching records from left table. If there are no matching records in left table then display with null values.



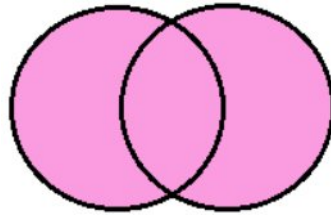
**Query:** `select e.*,d.* from employee e right join dept d on e.dno=d.dno`

Output:

	eno	ename	salary	dno	dno	dname
1	101	Anil	20000.00	10	10	IT
2	102	Sunil	23000.00	10	10	IT
3	103	Ajay	25000.00	10	10	IT
4	104	Kiram	30000.00	20	20	Maths
5	NULL	NULL	NULL	NULL	40	Physics

- iii. **Full Outer Join:** It is used to display all the records from both the tables.

- If there are no matching records in both the tables then display null values.



Query: `select e.*,d.*from employee e fulljoin dept d on e.dno=d.dno`

Output:

	eno	ename	salary	dno	dno	dname
1	101	Anil	20000.00	10	10	IT
2	102	Sunil	23000.00	10	10	IT
3	103	Ajay	25000.00	10	10	IT
4	104	Kiram	30000.00	20	20	Maths
5	105	Vijay	40000.00	30	NULL	NULL
6	NULL	NULL	NULL	NULL	40	Physics

Q) Write a Query for employee details whose salary is null?

A) `select*from employee where salary =null`

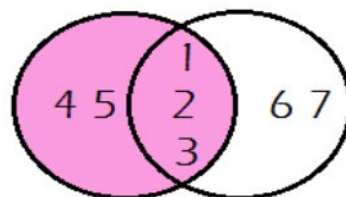
- The above Query will not display null values from employee table.
- If we want to display null values from table we have to use is operator.
- `select*from employee where salary isnull`

Q) Write a Query to display only left records from both the tables?

A)

Table1	
id	name
1	One
2	Two
3	Three
4	Four
5	Five

Table1	
id	name
1	One
2	Two
3	Three
6	Six
7	Seven



Only left records

Query: `select t1.*  
from Table1 t1 leftjoin Table2 t2 on t1.id=t2.id  
where t2.id isnull`

Output:

	id	name
1	4	Four
2	5	Five

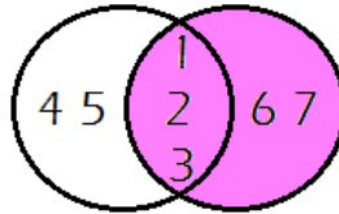
Q) Write a Query to display only right records?

A) **select** t2.\*

**from** table1 t1 **rightjoin** table2 t2 **on** t1.id=t2.id  
**where** t1.id isnull

Output:

	id	name
1	6	Six
2	7	Seven



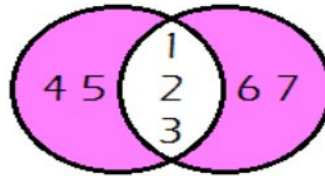
Q) Write a Query to display only left records and right records?

A) **select** t1.\*,t2.\*

**from** table1 t1 **fulljoin** table2 t2 **on** t1.id=t2.id  
**where** t1.id isnull or t2.id isnull

Output:

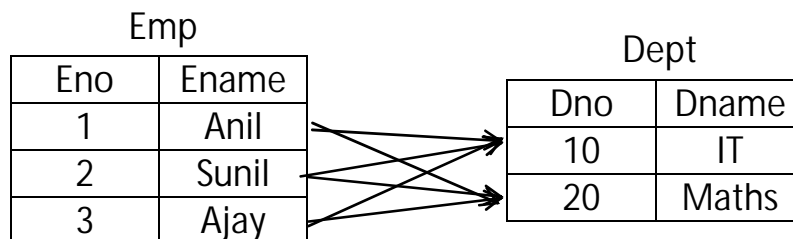
	id	name	id	name
1	4	Four	NULL	NULL
2	5	Five	NULL	NULL
3	NULL	NULL	6	Six
4	NULL	NULL	7	Seven



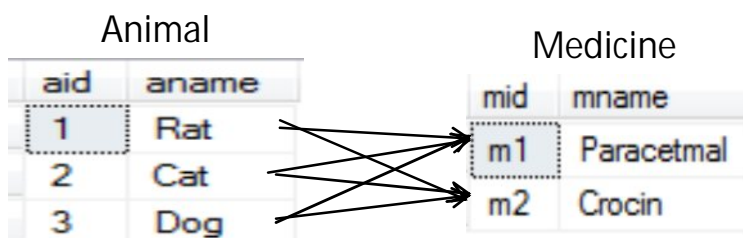
### 3. Cross Join:

Cross Join is the Cartesian product of no. of rows from left table with no. of rows from Right table.

- If there are m rows in left table and n rows in right table then the Cartesian Product is m\*n rows.
- i.e., if there are 3 rows in left table and 2 rows in right table then the Cartesian is 6 rows



Query: **select** e.\*,d.\***from** emp e **crossjoin** dept d



Query: `select * from animal, medicine (Or)`  
`select Animal.*, Medicine.*`  
`from Animal crossjoin Medicine`

Output:

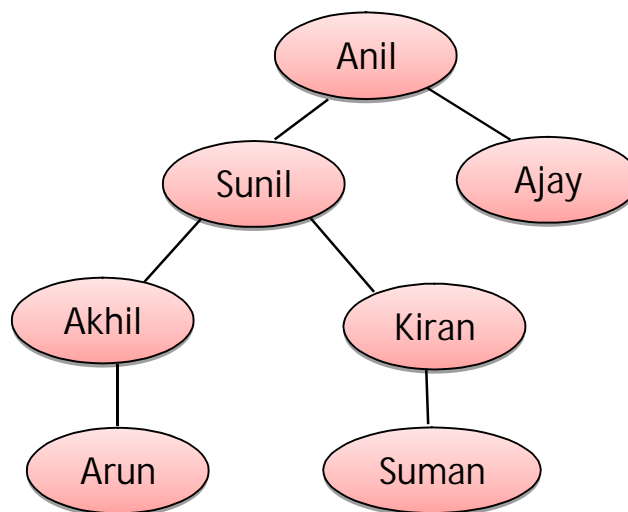
	aid	aname	mid	mname
1	1	Rat	m1	Paracetmal
2	2	Cat	m1	Paracetmal
3	3	Dog	m1	Paracetmal
4	1	Rat	m2	Crocin
5	2	Cat	m2	Crocin
6	3	Dog	m2	Crocin

#### 4. Self-Join:

Joining the table by itself is called Self-Join.

Emp

eno	ename	mid
1	Anil	NULL
2	Sunil	1
3	Ajay	1
4	Akhil	2
5	Kiran	2
6	Arun	4
7	Suman	5



Emp e

eno	ename	mid
1	Anil	NULL
2	Sunil	1
3	Ajay	1
4	Akhil	2
5	Kiran	2
6	Arun	4
7	Suman	5

Emp m

eno	ename	mid
1	Anil	NULL
2	Sunil	1
3	Ajay	1
4	Akhil	2
5	Kiran	2
6	Arun	4
7	Suman	5

**Query:** `select e.ename as 'Employee',m.ename as 'Manager' from emp e join emp m on e.mid=m.eno`

Output:

	Employee	Manager
1	Sunil	Anil
2	Ajay	Anil
3	Akhil	Sunil
4	Kiran	Sunil
5	Arun	Akhil
6	Suman	Kiran

### 5. Equi Join:

It is used to retrieve the data from multiple tables or more than one table by using equal (=) Operator.

- In Equi Join inner join is replaced with "," and on condition is replaced with where condition

Emp		
Eno	Ename	Dno
1	Anil	10
2	Sunil	20
3	Ajay	30

Dept	
Dno	Dname
10	ECE
20	CSE
30	EEE

**Query:** `select e.eno,e.ename,d.dname from emp e,dept d where e.dno=d.dno`

Output:

	eno	ename	dname
1	1	Anil	ECE
2	2	Sunil	CSE
3	3	Ajay	EEE



6. Non-Equi Join: It is used to display the data from more than one table without using "=" Operator.

Emp

Eno	Ename	Salary
101	Anil	20000
102	Sunil	21000
103	Ajay	22000
104	Akil	23000
105	Sunil	24000
106	Sagar	25000

Salary Grade

LowSal	HighSal	Grade
24000	25000	A
22000	23000	B
20000	22100	C
15000	19000	D

Q) Write a Query to display the Emp details along with his Grade?

A) **select e.\*,s.Grade from emp e,salGrade s  
where e.salary between s.LowSal and s.HighSal**

Output:

	Eno	Ename	Salary	Grade
1	105	Sunil	24000.00	A
2	106	Sagar	25000.00	A
3	103	Ajay	22000.00	B
4	104	Akil	23000.00	B

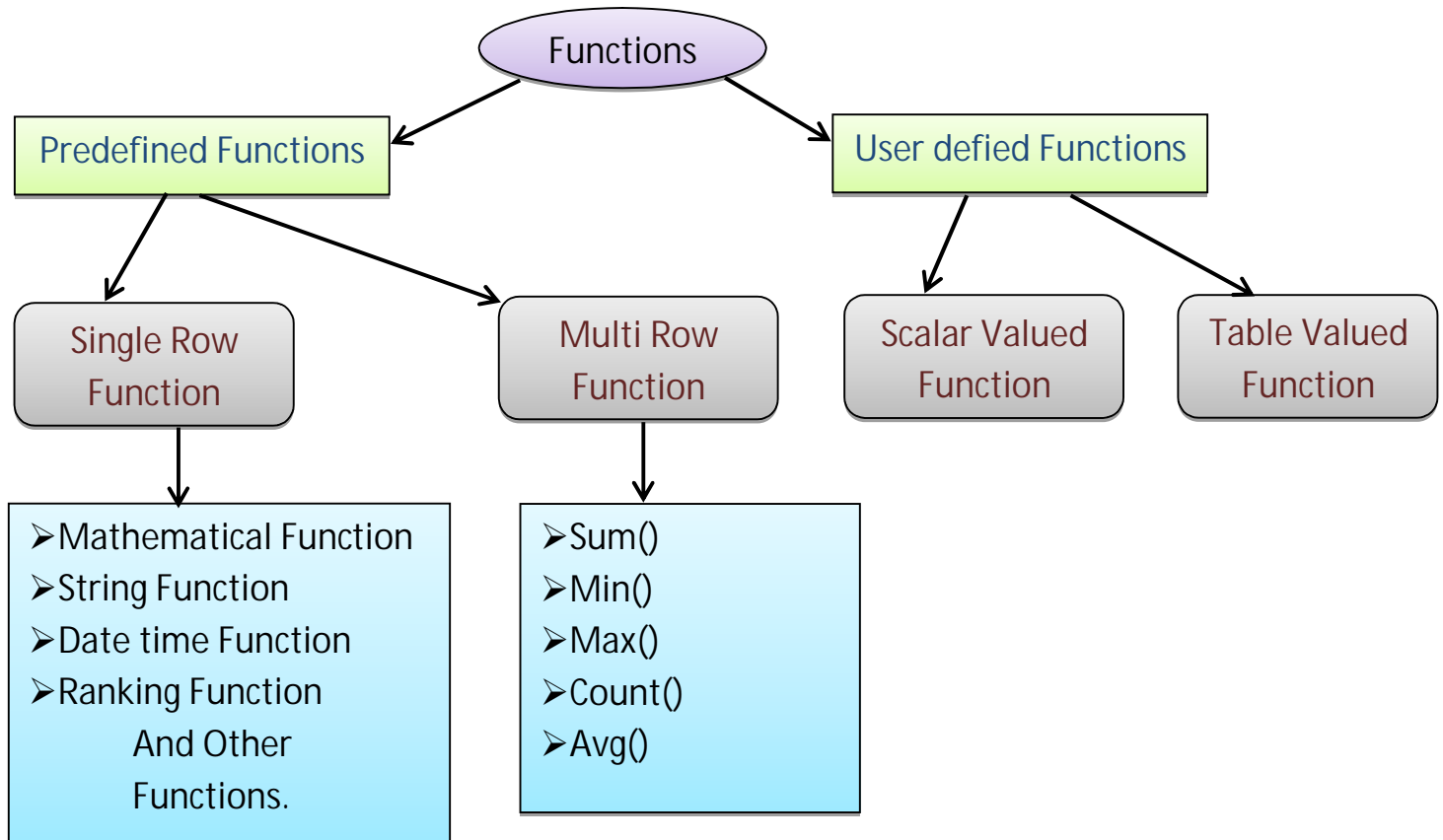


## Functions

### Functions in SQL:

Function is a Sub Program which is used to perform some Operation and return some value.

- Functions are of two types.



- Function will always take the input from the user and return only one value of any datatype. That may be int, varchar(), float, money etc.  
**Predefined Function:** The function that was given by Microsoft is called as Predefined Function.  
**User Defined Function:** The Function that was created by the programmer depending on the user requirement are called as User Defined Functions
- Predefined Functions are of two types.
  - Single row
  - Multi row

### Single Row Function:

The Function which will process on Single row at a time and return only one value is called as Single Row Function.

### Multi Row Function:

The function which will process on multiple rows at a time and return only one value is called as Multi Row Function.

### Mathematical Function:

The Function which will take number as input from the user and return a numeric value is called as Mathematical Function.

#### (i) Abs(number):

This Function is used to remove the sign of the given number.

Ex: `select Abs ( -5 )`

	(No column name)
1	5

#### (ii) Square():

This Function is used to find the square of the given number.

Ex: `select Square ( 5 )`

	(No column name)
1	25

#### (iii) Sqrt():

This Function is used to find the square root of the given number.

Ex: `select sqrt ( 25 )`

	(No column name)
1	5

#### (iv) Sign():

This function is used to take input as number and return 1 if the given number is positive.

Ex:	Input	Output				
	<code>select sign ( 25 )</code>	<table><tr><td></td><td>(No column name)</td></tr><tr><td>1</td><td>1</td></tr></table>		(No column name)	1	1
	(No column name)					
1	1					
	<code>select sign ( -25 )</code>	<table><tr><td></td><td>(No column name)</td></tr><tr><td>1</td><td>-1</td></tr></table>		(No column name)	1	-1
	(No column name)					
1	-1					
	<code>select sign ( 0 )</code>	<table><tr><td></td><td>(No column name)</td></tr><tr><td>1</td><td>0</td></tr></table>		(No column name)	1	0
	(No column name)					
1	0					

(v) **Power(base no.,exponent):**(Meaning = base no.<sup>exponent</sup>)

This function is used to get the power of the given number.

Ex: **select power ( 2 , 3 )**

(No column name)	
1	8

Note: Here  
power(2,3) means  $2^3$

(vi) **String function:**

This function input will take input as number and return an integer or varchar.

1. **Len(string):**This function is used to get the length of string.

Ex: **select len ( 'sathya' )**

(No column name)	
1	6

2. **Left(string,number):**This function is used to get the number of characters from left side of the string.

Ex: **select left ( 'AnilKumar' , 4 )**

(No column name)	
1	Anil

3. **Right(string,number):** This function is used to get the number of characters from right side of the string.

Ex: **select right ( '11C21A0484' , 4 )**

(No column name)	
1	0484

4. **Lower(string):** This function is used to Convert Upper case character to Lower case.

Ex: **select lower ( 'SATHYA' )**

(No column name)	
1	sathya

5. **Upper(string):** This function is used to Convert Lower case character to Upper case.

Ex: **select upper ( 'sathya' )**

(No column name)	
1	SATHYA

6. **ASCII(string)**: This function is used to get the ASCII values of the given Character

Ex: **select ASCII ( 'a' )**

	(No column name)
1	97

**Note:** a-z: 97-122  
A-Z: 65-90  
0-9: 48-57  
. : 46  
@ : 64  
\_ : 95  
Back space:

7. **Char(integer)**: This is used to convert integer to char based on ASCII.

Ex: **select char ( 97 )**

	(No column name)
1	a

8. **Ltrim(string)**: This function is used to remove the blank spaces before the given string.

Ex: **select ltrim ( '        aaaaa    ' )**

	(No column name)
1	aaaaa

9. **Rtrim(string)**: This function is used to remove the blank spaces after the given string.

Ex: **select rtrim ( '        aaaaa    ' )**

	(No column name)
1	aaaaaa

10. **SubString()**: This function is used to return substring from given string

Syntax: **select substring(column\_name,startposition,length)**

Ex: **select substring ( 'sathyatechnologies' , 7 , 4 )**

	(No column name)
1	tech

11. **Reverse()**: This function is used to reverse the given string.

Ex: **select reverse ( 'sathyam' )**

	(No column name)
1	mayhtas

(vii) **Datetime Function:**

These Functions are used to perform operations on date and time.

1. **getdate()**: This function is used to return the current system date along with time.

Ex: **select getdate ( )**

- ✓ datetimeformat in SQL  
yyyy-mm-ddhh:min:sec:msec

	(No column name)
1	2017-02-23 18:15:45.353

2. **day()**: This function is used to return the day from the given date.

Ex: **select day (getdate ( ) )** (or)

Ex: **select day ( '2017-02-23' )**

	(No column name)
1	23

3. **month()**: This function is used to return the month from the given date

Ex: **select month ( '2017-02-23' )** (or)

Ex: **select month (getdate ( ) )**

	(No column name)
1	2

4. **year()**: This function is used to return the year from the given date.

Ex: **select year ( '2017-02-23' )** (or)

Ex: **select year (getdate ( ) )**

	(No column name)
1	2017

Q) Write a Query to display the current system date in yyyy-mm-ddformat?

A) **select cast (day (getdate ( ) ) as varchar ( 20 ) ) + '-'**  
**' + cast (month (getdate ( ) ) as varchar ( 20 ) ) + '-'**  
**' + cast (year (getdate ( ) ) as varchar ( 20 ) )**

	(No column name)
1	23-2-2017

5. **DateAdd(datepart,increment,date)**: This function is used to add the number of days, months, years to the given date. Datepart is the part of the date.  
yyyy (or) yy (or) year

mm - month  
dd - day  
hh - hour  
mi - minutes  
ss - seconds  
ms - milliseconds  
dy - date of the year

dw - date of the week  
wk - week  
qq - quarter of the year

Q) Write a Query to display date after 5 years?

A) `select dateadd(yy, 5, getdate())`

	(No column name)
1	2022-02-23 19:09:38.440

Q) Write a Query to display date after 3 months?

A) `select dateadd(mm, 3, getdate())`

	(No column name)
1	2017-05-23 19:12:15.313

Q) Write a Query to display date before 3 days?

A) `select dateadd(dd, -3, getdate())`

	(No column name)
1	2017-02-20 19:17:04.233

#### 6. Datediff(datepart, startdate, enddate):

Datediff function is used to find the difference between the dates.

Ex:

Note: From based on this Emp table Example:

eno	ename	doj
101	Anil	2012-03-10
102	Sunil	2014-01-15
104	Ganesh	2011-05-20
106	Surya	2017-01-20

Query:

`select *, datediff(yy, doj, getdate())  
as 'Exp' from emp`

Output:

	eno	ename	doj	Exp
1	101	Anil	2012-03-10	5
2	102	Sunil	2014-01-15	3
3	104	Ganesh	2011-05-20	6
4	106	Surya	2017-01-20	0

Q) Write a Query to display Emp details and who is having more than 3 years of Experience.

A) `select*,datediff(yy,dof,getDate())`  
`as'Exp'from emp1`  
`wheredatediff(yy,dof,getDate())>3`

	eno	ename	dof	Exp
1	101	Anil	2012-03-10	5
2	104	Ganesh	2011-05-20	6

Q) Write a Query to display emp details along with age?

	eno	ename	dob
	101	Anil	1996-03-07
	102	Ajay	1994-05-27
	103	Arun	1995-04-07
	104	John	2000-04-04

A)`select*,datediff(yy,dob,getDate())as'Age'`  
`from emp1`

	eno	ename	dob	Age
1	101	Anil	1996-03-07	21
2	102	Ajay	1994-05-27	23
3	103	Arun	1995-04-07	22
4	104	John	2000-04-04	17

Q) Write a Query to display Employee details whose age is greater than 18?

A) `select*,datediff(yy,dob,getDate())as'Age'`  
`from emp1 wheredatediff(yy,dob,getDate())>18`

	eno	ename	dob	Age
1	101	Anil	1996-03-07	21
2	102	Ajay	1994-05-27	23
3	103	Arun	1995-04-07	22

Q) What is the difference between cast() function and Convert() function?

- Cast(): It is used to Convert from one data type to another data type
- Convert(): It is mostly designed to Convert date time to varchar.

Converting datetime to varchar by using cast function:

Ex:

- `selectgetDate()`
- `selectcast(getdate()asvarchar(11))`
- `selectcast(getdate()asvarchar(20))`

Outputs:



	(No column name)
1	2017-02-23 21:56:02.873

	(No column name)
1	Feb 23 2017

	(No column name)
1	Feb 23 2017 9:56PM

### Convert(datetype,date):

It is used to Convert date time to varchar and it is used to display the dates in different formats.

Ex:

- 1) `select convert (varchar(11),getdate())`
- 2) `select convert (varchar(19),getdate(),101)`
- 3) `select convert (varchar(19),getdate(),103)`
- 4) `select convert (varchar(19),getdate(),104)`
- 5) `select convert (varchar(19),getdate(),105)`
- 6) `select convert (varchar(19),getdate(),106)`
- 7) `select convert (varchar(19),getdate(),107)`
- 8) `select convert (varchar(19),getdate(),102)`

Outputs:

1.	1	(No column name)	Feb 23 2017
2.	1	(No column name)	02/23/2017
3.	1	(No column name)	23/02/2017
4.	1	(No column name)	23.02.2017
5.	1	(No column name)	23-02-2017
6.	1	(No column name)	23 Feb 2017
7.	1	(No column name)	Feb 23, 2017
8.	1	(No column name)	2017.02.23

### Datename(datepart,date):

This function is used to return the datename.

Ex1. : `select datename (mm, getdate())`

	(No column name)
1	February

Ex2. : `select date_name(dw, getdate())`

	(No column name)
1	Thursday

### Multi row Function (or) Aggregate Function:

The Function which will process on multiple rows at a time and return only one value is called as Multi row function.

1. sum()
2. Min()
3. Max()
4. Count()
5. Avg()

Q) Write a Query to Count the number of Employees working in the Company from given table (emp)?

Eno	Ename	Salary
101	Anil	20000.00
102	Sunil	21000.00
103	Ajay	22000.00
104	Akil	23000.00
105	Sunil	24000.00
106	Sagar	25000.00

A) `select count(*) from emp`

O/P:

	(No column name)
1	6

Q) Write a Query to Count the number of males in the company from given table(employee)?

eno	ename	gender	salary
101	Anil	male	20000.00
102	Sunitha	female	21000.00
103	Archana	female	21000.00
104	Sai	male	23000.00
105	Goutham	male	25000.00
106	Swapna	female	24000.00
107	Abhi	male	25000.00

A) `select count(*) from employee where gender='male'`

O/P:

	(No column name)
1	4

Q) Write a Query to display total salary of all Employees?

A) **select sum(salary) from employee**

O/P:

	(No column name)
1	159000.00

Q) Write a Query to Display average salary of an Employee?

A) **select avg(salary) from employee**

O/P:

	(No column name)
1	22714.2857

Q) Write a Query to display minimum salary of an Employee?

A) **select min(salary) from employee**

O/P:

	(No column name)
1	20000.00

Q) Write a Query to display maximum salary of an Employee?

A) **select max(salary) from employee**

O/P:

	(No column name)
1	25000.00

## Clauses in SQL

### Clauses in SQL:

1. Order by Clause
2. Group by Clause
3. Having Clause
4. From Clause

1. Order by Clause: It is used to display the data either in ascending order (or) in descending order.

- By default Order by Clause arrange the data in ascending order.

Q) Write a Query to display the employee details in ascending order based on ename?

A) `select * from employee order by ename asc`

	eno	ename	gender	salary
1	107	Abhi	male	25000.00
2	101	Anil	male	20000.00
3	103	Archana	female	21000.00
4	105	Goutham	male	25000.00
5	104	Sai	male	23000.00
6	102	Sunitha	female	21000.00
7	106	Swapna	female	24000.00

Q) Write a Query to display employee details in descending order based on ename?

A) `select * from employee2 order by ename desc`

	eno	ename	gender	salary
1	106	Swapna	female	24000.00
2	102	Sunitha	female	21000.00
3	104	Sai	male	23000.00
4	105	Goutham	male	25000.00
5	103	Archana	female	21000.00
6	101	Anil	male	20000.00
7	107	Abhi	male	25000.00

**Note:** desc is mandatory to write, but asc is not mandatory.

Q) Write a Query to display emp details in ascending order who are staying in Hyderabad based on ename?

Note: From based on this table.

eno	ename	city	salary
101	Anil	Vizag	20000.00
102	Sunitha	Hyderabad	21000.00
103	Archana	Vijayavada	21000.00
104	Sai	Hyderabad	23000.00
105	Goutham	Hyderabad	25000.00
106	Swapna	Vizag	24000.00
107	Abhi	Hyderabad	25000.00

A) `select * from employee2 where city='Hyderabad' order by ename`

	eno	ename	city	salary
1	107	Abhi	Hyderabad	25000.00
2	105	Goutham	Hyderabad	25000.00
3	104	Sai	Hyderabad	23000.00
4	102	Sunitha	Hyderabad	21000.00

Q) Write a Query to display emp details along with dept name from emp and dept tables order by ename?

Dept		Emp		
Dno	Dname	Eno	Ename	Dno
10	ECE	101	Anil	10
20	CSE	102	Sunil	10
30	EEE	103	Ajay	30
		104	Akil	20
		105	Sagar	20

A) `select e.*,d.dname from emp e inner join dept d on e.dno=d.dno order by e.ename`

	Eno	Ename	Dno	dname
1	103	Ajay	30	EEE
2	104	Akil	20	CSE
3	101	Anil	10	ECE
4	105	Sagar	20	CSE
5	102	Sunil	10	ECE

## 2. Group by Clause:

Q) Write a Query to display the total salary of males and females?

Emp2			
Eno	Ename	Gender	Salary
101	Anil	male	20000.00
102	Sunil	male	22000.00
103	Swathi	female	23000.00
104	Anitha	female	21000.00
105	Rakesh	male	18000.00
106	Nirisha	female	22000.00

A) `select Gender, sum(salary) as 'Total Sal'`  
`from emp2 groupby gender`

	Gender	Total Sal
1	female	66000.00
2	male	60000.00

**Group by Clause:** It will group the common set of values as Single group in a Single Column.

- Group by Clause will always work with aggregate functions.
- Aggregate functions are applied on the individual groups created by group by clause.
- Whenever we execute the above query group by clause will group the data based on gender Column. i.e., male group and female group of sum(salary).
- Aggregate function will be applied on grouped data i.e., on male group and femaleseparately.

Q) Write a Query to Count the number of employees working in each department?

A) `select d.dname, Count(*) as 'No of Emp'`  
`from emp e innerjoin dept d on e.dno=d.dno groupby d.dname`

	dname	No of Emp
1	CSE	2
2	ECE	2
3	EEE	1

Q) Write a Query to display the min sal and max sal of an Emp2 based on gender ?

A) `select Gender, Min(salary) as 'min sal', Max(salary) as 'max sal'`  
`from emp2 groupby Gender`

	Gender	min sal	max sal
1	female	21000.00	23000.00
2	male	18000.00	22000.00

Q) Write a Query to display number of employees in each city order by City name?



Employee2			
eno	ename	city	salary
101	Anil	Vizag	20000.00
102	Sunitha	Hyderabad	21000.00
103	Archana	Vijayavada	21000.00
104	Sai	Hyderabad	23000.00
105	Goutham	Hyderabad	25000.00
106	Swapna	Vizag	24000.00
107	Abhi	Hyderabad	25000.00

A) `select city, Count(*) as 'No of Emp' from employee2 groupby city orderby city`

	city	No of Emp
1	Hyderabad	4
2	Vijayavada	1
3	Vizag	2

Note: Here 'orderby city' is Optional.

- Always write the order by after group by if we want to use both Clauses.

Note: We cannot apply where condition on group by clause, if we want to apply where Condition on group by clause we have to use having Clause.

- Having Clause will always work with aggregate function.

Q) Write a Query to display number of employees who are working in each department where number of employees > 1?

A) `select d.dname, Count(*) as 'No of Emp'`

`from emp e innerjoin dept d on e.dno=d.dno groupby d.dname having Count(*) > 1`

	dname	No of Emp
1	CSE	2
2	ECE	2

Q) Write a Query to display the total salary that was assigned for each department where totalsal > 20000?

A) `select d.dname, sum(salary) as 'total sal'`

`from employee e innerjoin dept d on e.dno=d.dno groupby dname having sum(salary) > 20000`



	dname	total sal
1	CSE	30000.00
2	ECE	68000.00
3	EEE	40000.00

**SubQuery:** SubQuery is a query inside another query.

- A select Command inside another select command.
- We can write maximum of 32 select commands in a single subquery .

Syntax: OuterQuery(innerQuery)

()→SubQuery Operator.

1. In SubQuery always the inner query will gets executed and based on the result of the inner query the highest query will gets executed.

**Q) Write a Query to display the max salary of an Employee?**

A) `select max(salary) from employee`

	(No column name)
1	40000.00

**Q) Write a Query to display the second max salary of an Employee?**

A) `select Max(salary) from employee where Salary < (select max(salary) from employee)`

	(No column name)
1	30000.00

**Q) Write a Query to display the third max salary of an Employee?**

A) `select Max(salary) from employee where salary < (select max(salary) from employee where salary < (select max(salary) from employee))`

	(No column name)
1	25000.00

**In Operator:** In Operator will work like or Operator.

**Q) Write a Query to display employee details whose employee numbers are 101,102?**

A) `select * from employee where eno in(101,102)`

	eno	ename	salary	dno
1	101	Anil	20000.00	10
2	102	Sunil	23000.00	10

Student			
Sno	Sname	Age	Address
101	Anil	20	Hyd
102	Sunil	23	Hyd
103	Ajay	22	Chennai
104	Vijay	20	Banglore

Department	
Did	Dname
10	ECE
20	CSE
30	EEE

StudentDept	
Sno	Did
101	10
102	10
103	20
104	30

Q) Write a Query to display the Student details along with department name?

A) **select s.\*,d.dname**  
**from student s innerjoin Studentdept sd on s.sno=sd.sno**  
**innerjoin department d on sd.did=d.did**

	Sno	Sname	Age	Address	dname
1	101	Anil	20	Hyd	ECE
2	102	Sunil	23	Hyd	ECE
3	103	Ajay	22	Chennai	CSE
4	104	Vijay	20	Banglore	EEE

Q) Write a Query to display Student details who are Studying in department 10?

A) **select\*from Student where sno in**  
**(select s.sno**  
**from student s innerjoin studentdept sd on s.sno=sd.sno**  
**innerjoin department d on sd.did=d.did where d.did=10)**

	Sno	Sname	Age	Address
1	101	Anil	20	Hyd
2	102	Sunil	23	Hyd

Q) Write a Query to display Student details who are working in EEE department?

A) **select\*from Student where sno in**  
**(select s.sno**  
**from student s innerjoin studentdept sd on s.sno=sd.sno**  
**innerjoin department d on sd.did=d.did where d.dname='EEE')**

	Sno	Sname	Age	Address
1	104	Vijay	20	Banglore

Q)Write a Query to display Student details who are working in ECE department and whose age is greater than 22?

A)**select\*from Student where sno in**  
**(select s.sno**

```
from student s innerjoin studentdept sd on s.sno=sd.sno
innerjoin department d on sd.did=d.did where d.dname='ECE' and
s.age>22)
```

(or)

```
select*from Student where sno in
(select sno from studentdept where did in
(select did from department where dname='ECE' and age>22))
```

	Sno	Sname	Age	Address
1	102	Sunil	23	Hyd

Q) Write a Query to display Student details who are working in ECE dept and whose name starts with a?

A) 

```
select*from Student Where sno in(
select sno from StudentDept where Did in(
select Did from Department where Dname='ECE' and sname like 'a%'))
```

	Sno	Sname	Age	Address
1	101	Anil	20	Hyd

Q) Write a Query to display 5<sup>th</sup> max salary of an Employee?

A) Formula to display N<sup>th</sup> Max Salary:-  
 Select e.colname from table1 e where n=  
 (select count (f.colname) from table1 f where condition)

Employee e		
eno	ename	salary
101	Anil	20000.00
102	Sunil	23000.00
103	Ajay	25000.00
104	Kram	30000.00
105	Vijay	40000.00

Employee f		
eno	ename	salary
101	Anil	20000.00
102	Sunil	23000.00
103	Ajay	25000.00
104	Kram	30000.00
105	Vijay	40000.00

```
select e.salary from employee e where
5=(selectcount(f.salary)from employee f where
e.salary<=f.salary)
```

	salary
1	20000.00

### Explanation for Query:

If(e.salary <= f.salary)

```
{
    Select count(f.salary) from employee f
}
```

e.salary	f.salary	Count			
20000	<= 20000	1	5	= 5 (True)	execute
20000	<= 23000	1	23000	<= 20000	0
20000	<= 25000	1	4	= 5(False)	Does not Execute
20000	<= 30000	1	---		
20000	<= 40000	1	---		
		<u>5</u>	---		

### Query if the column has the duplicate records:

Note: The above query will not work with duplicate values so distinct keyword is used in the above query.

```
select distinct(e.salary) from employee e where 1=
(select count(distinct f.salary) from employee f where
e.salary <= f.salary)
```

**Note:** distinct keyword will consider duplicate values in a column as a single value.

Example:

e.salary	f.salary	Count
22000	<= 20000	0
22000	<= 22000	1
22000	<= 23000	1
		<u>2</u>

### 3. From Clause:

It is used to pass query as table name.

Ex:

```
select * from emp
select * from (select * from emp) e
```

**Top:** It is used to display the top records from the table.

Q) Write a Query to display the top 3 records from the table?

A) **select top 3 \* from employee**

	eno	ename	salary
1	101	Anil	20000.00
2	102	Sunil	23000.00
3	103	Ajay	25000.00

Q) Write a Query to display the 1<sup>st</sup> record from the table?

A) `select top 1 * from employee`

	eno	ename	salary
1	101	Anil	20000.00

Q) Write a Query to display the max salary of an employee without using max function?

A) `select top 1 salary from employee order by salary desc`

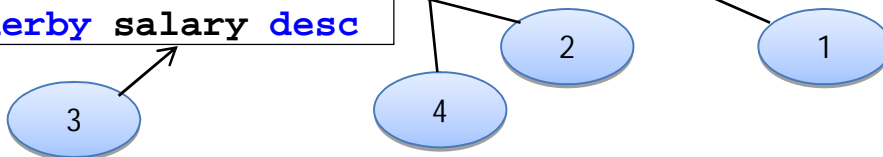
	salary
1	40000.00

**Note:** In the above Query salary will be displayed from employee table and Order by clause will arrange the salary in descending order and top function will display top 1 record from ordered data.

Q) Write a Query to display top the maximum 2 salaries from employee table?

A) `select distinct top 2 salary from employee`

`order by salary desc`



(Or)

`select distinct top 2 salary from employee order by salary desc`

Execution Steps:

1. Salary from employee
2. Distinct will consider the duplicate values as single value.
3. Order by clause will arrange the salary in desc order.
4. Top 2 will display top 2 salaries.

Q) Write a Query to display 5<sup>th</sup> max salary of an Employee?

A) `select distinct top 1 salary from (select distinct top 5 salary from employee order by salary desc) e order by salary asc`

salary
1 20000.00

For Example on let us

Considering Salary based on above Query:

Salary	Order by desc	Top 5	Order by asc
15000	40000	40000	20000
18000	30000	30000	23000
20000	25000	25000	25000
23000	23000	23000	30000
23000	20000	20000	40000
25000	18000		
25000	15000		
30000			
40000			
Applied <b>distinct</b> keyword here	Here duplicate records removed in order by desc		<b>top 1 salary</b>

- Whenever we execute the above query, the inner query will gets executed first and then the result of the inner query will be stored in e.
- Here the inner Query result is max 5 salaries.
- Whenever we want to display only 5<sup>th</sup> max salary from employee, then the Outer Query will display 5<sup>th</sup> max salary.

Q) Write a Query to display the last record?

A) `select top 1 * from (select top (select count(*) from employee) * from employee order by eno desc) e`

eno	ename	salary
1 105	Vijay	40000.00

Execution process:(From inner Query to Outer Query)

Step1: `select count(*) from employee`

101=	1	} Count=5
102=	1	
103=	1	
104=	1	
105=	1	



Step2: `selecttop (5)*from employee orderby eno desc`

	eno	ename	salary
1	105	Vijay	40000.00
2	104	Kram	30000.00
3	103	Ajay	25000.00
4	102	Sunil	23000.00
5	101	Anil	20000.00

Step3: `selecttop 1*from(X)e`

1	105	Vijay	40000.00
---	-----	-------	----------

Q) Write a Query to display last 3 records?

A) `selecttop 3*from(selectcount(*)from employee)*from employee orderby eno desc)e`

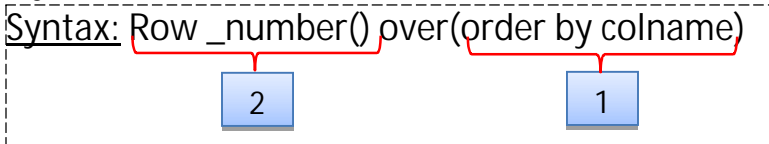
	eno	ename	salary
1	105	Vijay	40000.00
2	104	Kram	30000.00
3	103	Ajay	25000.00

Q) How to delete duplicate Records?

Row-Number(): Row number function is used to assign a unique row id for every row.

- Row\_Number() will always work with over clause and over clause will always with order by clause.

- Syntax: `Row_number() over (order by colname)`



Q) Write a Query to display employee details along with Rowid?

A) `select*,ROW_NUMBER()over (orderby eno)as'rowid'from employee`

	eno	ename	salary	rowid
1	101	Anil	20000.00	1
2	102	Sunil	23000.00	2
3	103	Ajay	25000.00	3
4	104	Kram	30000.00	4
5	105	Vijay	40000.00	5

- When we execute the above query order by clause will arrange the table in ascending order based on eno column and Row\_number() will assign a unique rowid for the ordered data.

Q) Write a Query to display the even records from the table?

A) `select*from(select*,ROW_NUMBER()over (orderby (eno))as'rowid'from employee)e where rowid%2=0`



	eno	ename	salary	rowid
1	102	Sunil	23000.00	2
2	104	Kram	30000.00	4

Partition by Clause: It is used to assign row number for the partitioned data.

Syntax: `Row_Number() over (Partition by colname order by (colname))`

3
1
2

Emp1				
eno	ename	dno	dname	City
101	Anil	10	IT	Hyd
102	Sunil	20	Maths	Hyd
103	Ajay	10	IT	Chennai
104	Vijay	30	Physics	Banglore
105	Arun	20	maths	Chennai

- Partition by clause will group the common set of values in a single columns as a Partition.
- Then order by clause will arrange each partitioned data in ascending order based on column.
- Row\_Number() function will give the rowid for the each row in each partition.

Q) Write a Query to display Employee deatails along with rowidpartition bydno order by city?

A) `select*,Row_Number()over (partitionby dno orderby (city))as'rowid'from emp1`

	eno	ename	dno	dname	City	rowid
1	103	Ajay	10	IT	Chennai	1
2	101	Anil	10	IT	Hyd	2
3	105	Arun	20	maths	Chennai	1
4	102	Sunil	20	Maths	Hyd	2
5	104	Vijay	30	Physics	Banglore	1

- When we execute above query partition by clause the partition ofdata based on dno.
- Order by clause will arrange each partition in ascending order based on city and then rowid is arranged by Row\_Number.

With Clause:

## Q) What is CTE?

A) CTE is Common Table Expression.

- CTE is a temporary database, whatever the modifications we are doing on CTE will be auto committed i.e. the modifications will affect to the original database.

Ex:

```
with emplcte
as
(
select * from emp1 where dname='IT'
)
deletefrom emplcte
```



(2 row(s) affected)

## Q) How to deleteduplicate Records? (\*\*\*\*\*)

A) with emplcte(ename,rowid)

as

```
(
select ename,row_number()over(partitionby ename
orderby(ename))from emp1
)
deletefrom emplcte where rowid>1
```

**TSQL:** Transact Structured Query Language.

TSQL is a Programming Language in SQL

Structure of TSQL Program:

Declare

Begin

End

- Declare block is used to declare the variable.
- Begin block is used to set the value for the variable.
- End block is used to end TSQL Program.
- Variable is an Identifier which is used to identify the value.

Syntax to declare variable is:

@variablename datatype

Syntax to set the value for the variable:

Set @variablename = value

Q) Write a program to declare first name and last name and display full name?

A) **declare**

@fname **varchar**(20),

@lname **varchar**(20),

@fullname **varchar**(20)

**begin**

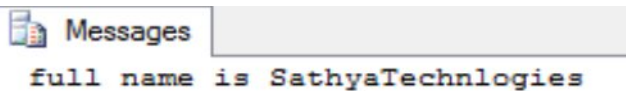
**set** @fname = 'Sathya'

**set** @lname = 'Technologies'

**set** @fullname = @fname ++ @lname

**print** 'full name is ' + @fullname

**end**



Messages

full name is SathyaTechnologies

Q) Write a Program to perform addition of two numbers?

A) **declare**

@fno **int**,

@sno **int**,

@add **int**

**begin**

**set** @fno = 10

**set** @sno = 20

**set** @add = @fno + @sno

**print** 'Sum is ' + **cast**(@add **as varchar**(20))

**end**

Messages

Sum is 30

Q) Write a program to declare sno, sname, m1, m2, m3 calculate total and % marks and display them

A) declare

```
@sno int,@sname varchar(20),@m1 int, @m2 int,@m3 int,@total
int,@per float
begin
set @sno=101
set @sname='Anil'
set @m1=90
set @m2=90
set @m3=90
set @total=@m1+@m2+@m3
set @per=@total/3.0
set @per=Round(@per,2)
print'Details of Student are:-'
print'Student Number is: '+cast(@sno asvarchar(20))+ ' and Name
is: '+@sname
print'total is '+cast(@total asvarchar(20))
print'Percentage is '+cast(@per asvarchar(20))
end
```

Messages

Details of Student are:-  
Student Number is: 101 and Name is: Anil  
total is 270  
Percentage is 90

Q) Write a Program to declare two numbers and check which number is greater?

A)declare

```
@a int,@b int
begin
set @a=10
set @b=20
if(@a>@b)
printcast(@a asvarchar(20))+ ' is big'
else
printcast(@b asvarchar(20))+ ' is big'
end
```

Messages

20 is big

Student							
sno	sname	m1	m2	m3	total	per	grade
101	Anil	79	78	77	NULL	NULL	NULL
102	Ajay	82	87	89	NULL	NULL	NULL
103	Vijay	85	87	76	NULL	NULL	NULL

Q) Write a TSQL program to calculate total marks, % of marks @grade of 101 and update total, percentage, grade to Student table.

A)declare

```
@sno int,@sname varchar(20),@m1 int,@m2 int,@m3 int,@total
int,@per float,@grade char(1)
begin
set @sno=101
set @sname=(select sname from Student where sno=@sno)
set @m1=(select m1 from Student where sno=@sno)
set @m2=(select m2 from Student where sno=@sno)
set @m3=(select m3 from Student where sno=@sno)
set @total=@m1+@m2+@m3
set @per=@total/3
if(@per>=75 and @per<100)
set @grade='A'
elseif(@per>=60 and @per<75)
set @grade='B'
else
set @grade='C'
update Student set total=@total,per=@per,grade=@grade where
sno=@sno
print'Record is updated'
end
```



Messages

```
(1 row(s) affected)
Record is updated
```

Q) Write a Query to view the result of updated Student table?

A)select\*from student

	sno	sname	m1	m2	m3	total	per	grade
1	101	Anil	79	78	77	234	78	A
2	102	Ajay	82	87	89	NULL	NULL	NULL

Q) Write a program to calculate and update?

Employee					
eno	ename	bsal	da	hra	Tsal
101	Anil	20000.00	NULL	NULL	NULL
102	Sunil	30000.00	NULL	NULL	NULL

A)declare

```
@eno int,@ename varchar(20),@bsal money,@da money,@hra
money,@Tsal money
begin
set @eno=101
set @ename=(select ename from employee where eno=@eno)
set @bsal=(select bsal from employee where eno=@eno)
set @da=0.2*@bsal
set @hra=0.4*@bsal
set @Tsal=@bsal+@da+@hra
update employee set da=@da,hra=@hra,Tsal=@Tsal where eno=@eno
print'Record is Updated'
end
```



Messages

(1 row(s) affected)  
Record is Updated

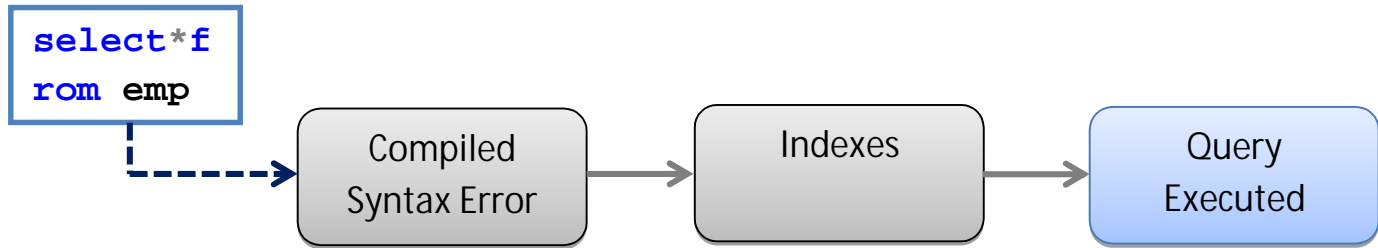
Q) Write a Query to view the result of updated Employee table?

A)select\*from Employee

	eno	ename	bsal	da	hra	Tsal
1	101	Anil	20000.00	4000.00	8000.00	32000.00
2	102	Sunil	30000.00	NULL	NULL	NULL

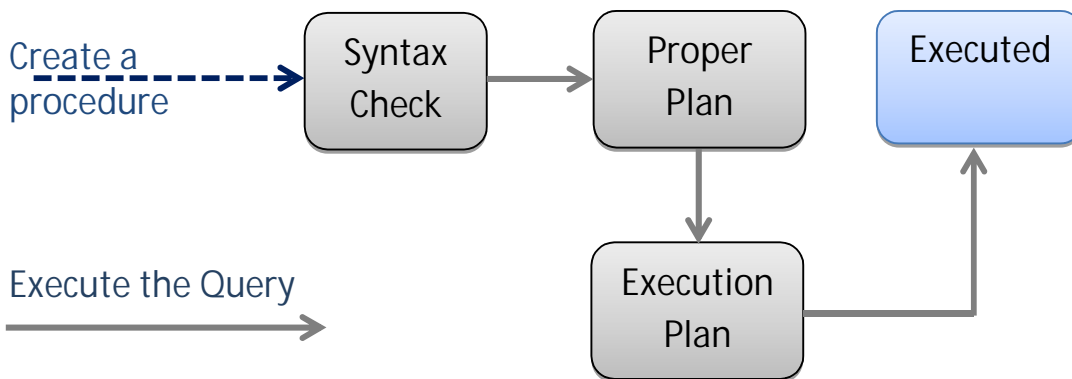
## Stored Procedure

Stored procedure is set of precompiled SQL statements which will gets executed when we call it.



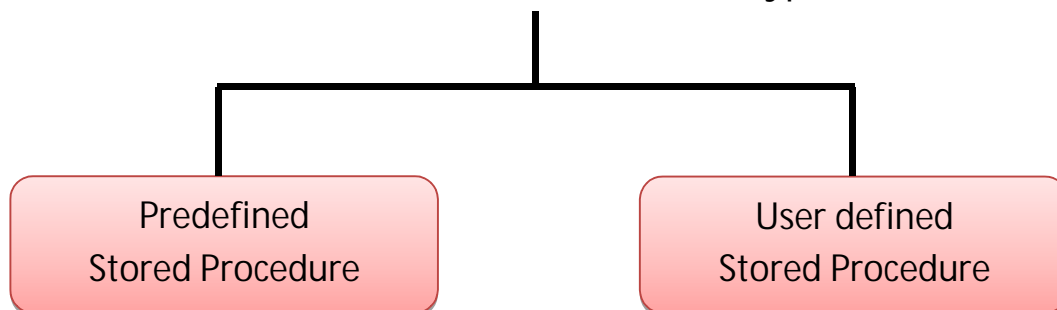
- Whenever we pass any SQL Query 3 types of Operations will be done.
  1. Syntax Checking.
  2. Proper plan is selected like any indexes are applied or not.
  3. Query execution will be done.

These 3 steps will be done every time when we pass the Query.



- Whenever we create a procedure, the syntax checking will be done and proper plan is selected like any indexes and this proper plan is stored in execution plan.
- Whenever we execute the procedure, the procedure will gets executed from execution plan.

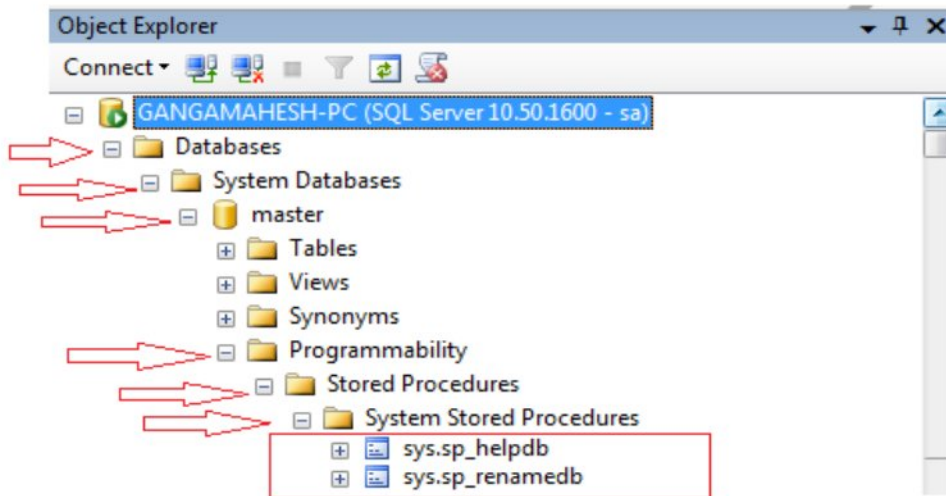
Stored Procedures are of 2 types.





### Predefined Stored Procedure:

- Predefined Stored Procedures are created by Microsoft.
- All the System defined Stored Procedures are available under  
Go to→View→Object explorer

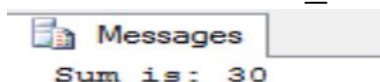


### User defined Stored Procedure:

- The Stored Procedure that was created depending on the user requirement is called as User defined Stored Procedure.
- Syntax to create Stored Procedure:  
Create procedure procedurename (parameters)  
As begin  
----- Sql Query----  
End
- Syntax to create the Procedure:  
Exec ProcedureName values
- At the time of executing the procedure we have to pass the values.
- The number of values that we pass must match with number of parameters.
- The order of the values that we pass must match with order of the parameters.
- The type of the values that we pass must match with type of parameters.

### Q) Create a procedure to perform addition operation

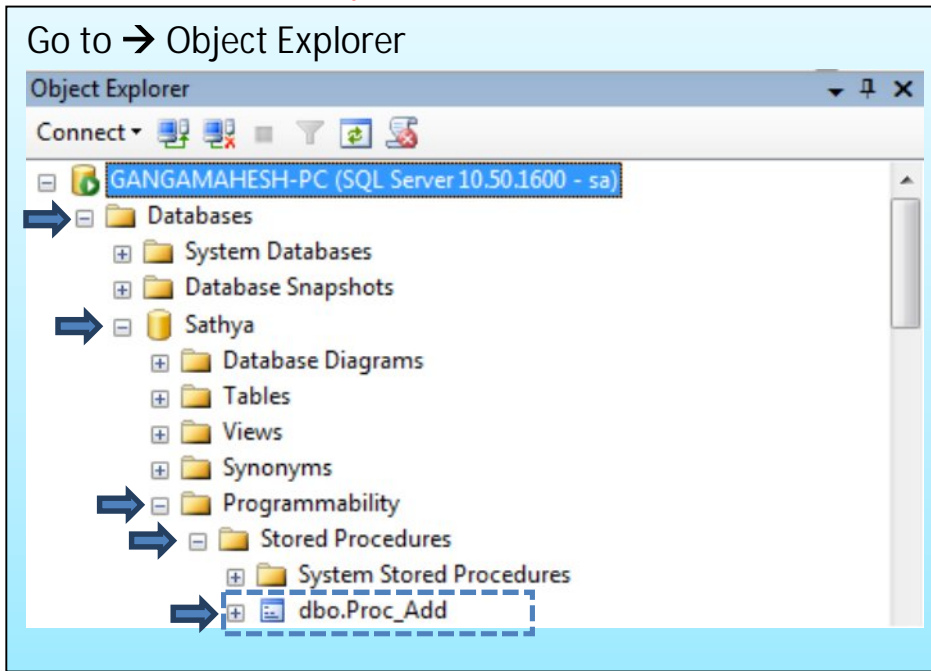
- **createprocedure** Proc\_Add(@a int,@b int)  
**asbegin**  
**declare** @c int  
**set** @c=@a+@b  
**print**'Sum is: '+cast(@c asvarchar(20))  
**end**
- **Exec** Proc\_Add10,20



Q) How to view the saved procedure?

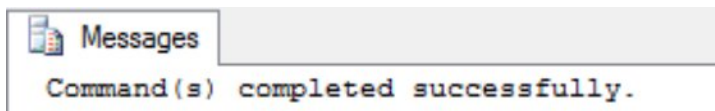
A)

Go to → Object Explorer

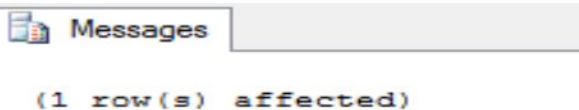


Q) Create Procedure to insert the record in Employee table

- `createprocedure Proc_insertEmployee(@eno int,@ename varchar(20),@salary money)  
asbegin  
insertinto Employee values(@eno,@ename,@salary)  
end`

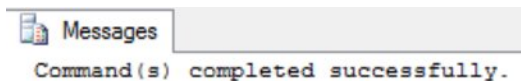


- `exec Proc_insertEmployee101,'Anil',23000`



Q) Create a Procedure to display the Employee details?

- `createprocedure Proc_Display  
asbegin  
select*from Employee  
end`



- `exec Proc_Display`

	eno	ename	salary
1	101	Anil	23000.00

Q) Create a procedure to delete the Employee details?

- **createprocedure** Proc\_DeleteEmp(@eno **int**)  
**asbegin**  
**deletefrom** Employee **where** eno=@eno  
**end**

Messages  
Command(s) completed successfully.

- **Proc\_DeleteEmp** 101

Messages  
(1 row(s) affected)

**Note:** Here Procedure name is mandatory, **exec** is Optional

Q) Create Procedure to display Employee details based on Eno?

- **createprocedure** Proc\_DisplayEmp(@eno **int**)  
**asbegin**  
**select\*** **from** Employee **where** eno=@eno  
**end**

Messages  
Command(s) completed successfully.

- **Proc\_DisplayEmp** 102

	eno	ename	salary
1	102	Ajay	23000.00

Q) Create a Procedure to display Employee details along with dname from Employee and Dept tables?

- **createprocedure** Proc\_DisplayEmpDept  
**asbegin**  
**select** e.\*,d.dname  
**from** Employee e **innerjoin** dept d **on** e.dno=d.dno  
**end**

Messages  
Command(s) completed successfully.

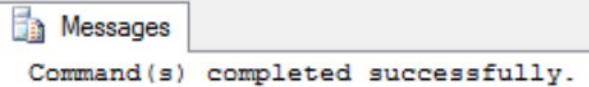
- **exec** Proc\_DisplayEmpDept

	eno	ename	salary	Dno	dname
1	101	Anil	20000.00	20	CSE
2	102	Ajay	23000.00	10	ECE
3	103	Surya	25000.00	10	ECE
4	104	Vinod	30000.00	30	EEE
5	105	Vijay	27000.00	20	CSE

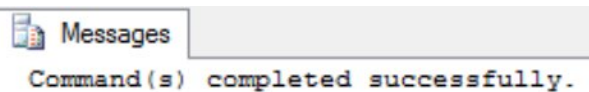
Q) Can we call a stored procedure in another Stored Procedure?

A) Yes

- **createprocedure P1**  
**asbegin**  
**select\*from Employee**  
**end**



- **createprocedure P2**  
**asbegin**  
**exec P1**  
**select\*from Dept**  
**end**



- **exec P2**

	eno	ename	salary	Dno
1	101	Anil	20000.00	20
2	102	Ajay	23000.00	10
3	103	Surya	25000.00	10
4	104	Vinod	30000.00	30
5	105	Vijay	27000.00	20

	Dno	Dname
1	10	ECE
2	20	CSE
3	30	EEE

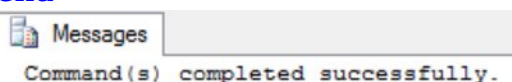
Stored Procedure will have 2 parameters

1. Input Parameter
2. Output Parameter

- Output Parameters must declare with Output keyword.
- Output Parameters are used to return the value from the Procedure.

Ex: Write a program to perform addition of two numbers by using Output Parameters?

- **createprocedure Proc\_Addition(@a int,@b int,@c intoutput)**  
**asbegin**  
**set @c=@a+@b**  
**end**




- `declare @sum int`  
`exec Proc_Addition10,20,@sum output`  
`print 'Sum is: ' + cast(@sum as varchar(20))`

 Messages   
 Sum is: 30

Q) Create a Procedure to display ename based on eno?

- `createprocedure Proc_Displayename(@eno int,@ename`  
`varchar(20)output)`  
`asbegin`  
`set @ename=(select ename from Employee where eno=@eno)`  
`end`


 Messages   
 Command(s) completed successfully.

- `declare @x varchar(20)`  
`exec Proc_Displayename101,@x output`  
`print 'Ename is: ' + @x`


 Messages   
 Ename is: Anil

Q) Create a Procedure to declare sno, sname, m1, m2, m3 calculate total & % and Display them?

- `createprocedure Proc_Student(@sno int,@sname varchar(20),`  
`@m1 int, @m2 int, @m3 int, @total intoutput, @percentage`  
`floatoutput)`  
`asbegin`  
`set @total=@m1+@m2+@m3`  
`set @percentage=@total/3`  
`end`

 Messages   
 Command(s) completed successfully.

- `declare @tot int`  
`declare @per float`  
`exec Proc_Student101,'Arjun',60,50,75,@tot output,@per`  
`output`  
`print 'Total Marks are: ' + cast(@tot as varchar(20))`  
`print 'Percentage is: ' + cast(@per as varchar(20))`

 Messages   
 Total Marks are: 185  
 Percentage is: 61

## TCL Commands (Transmission Control Language Commands):

Transaction is a portion of work that has been done.

1. Commit ctrl+s
2. Rollback ctrl +z

Commit: It is used to save the changes permanently.

Rollback: It is used to undo the changes.

Savepoint: Savepoint is used to save the portion of transactions.

- Savepoint is used to rollback the portion of transactions.

Once Committed Cannot rollback  
Once Savepoint can rollback.

- In SQL Server Whenever we pass any Query the Operations will perform on buffer database but not on permanent database.

Buffer db

Eno	Ename	Sal
101	Anil	20000
102	Sunil	30000

Permanent db

Sathyadb		
Emp		
Eno	Ename	Sal
101	Anil	20000
102	Sunil	30000

Ex: **begintran t1**  
**deletefrom emp where eno=102**

 Messages

(1 row(s) affected)

- Whenever we execute the above query 102 record is deleted from buffer database but not from Permanent database.

➤ **select \* from emp**

Buffer db

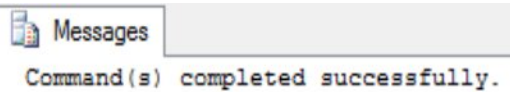
Eno	Ename	Sal
101	Anil	20000

Permanent db

Sathyadb		
Emp		
Eno	Ename	Sal
101	Anil	20000
102	Sunil	30000

- When the above query is executed, records are displayed that are in buffer db

➤ **begintran t2**  
**rollback**

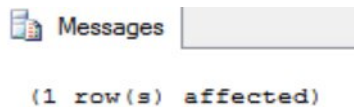


- When the above query is executed then again the record is inserted into buffer db from permanent db.

➤ **select \* from emp**

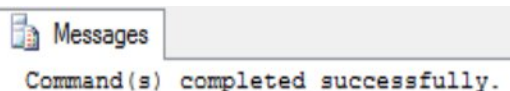
- All the records will be displayed.

➤ **begintran t1**  
**delete from emp where eno=104**  
**commit**



- Whenever we execute the above query, the record is deleted from the buffer db and the changes are permanently saved to permanent database.

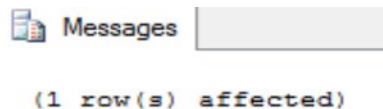
➤ **begintran t2**  
**rollback**



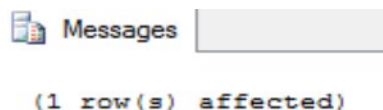
- We cannot rollback the record, once committed transaction.
- Savepoint is used to rollback the portion of transactions.

EX:

➤ **begintran t1**  
**delete from employee where eno=101**  
**save tran s1**



➤ **begintran t2**  
**delete from employee where eno=102**  
**save tran s2**



➤ **begintran t3**  
**delete from employee where eno=103**  
**save tran s3**



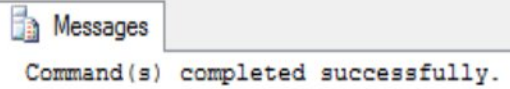
➤ **select \* from employee**



Output:

	eno	ename	salary
1	104	Vinod	30000.00
2	105	Vijay	27000.00

- **begintran t4**  
**rollbacktran s1**



- **select\*from employee**

Output:

	eno	ename	salary
1	102	Ajay	23000.00
2	103	Surya	25000.00
3	104	Vinod	30000.00
4	105	Vijay	27000.00

- rollback s1 means s2,s3... will come but s1 will not come.
- rollback s2 means s3, s4... will come but s2 will not come. ...so on.

## Exception Handling

Whenever we write any Stored Procedure, the procedure will be compiled at the time of compilation of the procedure.

- The compiler will check for Syntax errors and display to the programmer.
- Programmer will read the error message and rectify it.

Q) When compile time errors will occur?

A) At the Compilation of the query or Procedure.

Q) Why Compile time error will occur?

A) Because of the wrong syntax given by programmer.

Q) Can we rectify compile time error?

A) Yes.

Q) Who will identify Compile time errors?

A) Compiler.

Q) What is an Exception?

A) An Exception is runtime error.

Q) When Exception will occur?

A) At the time of Executing the query or Procedure.

Q) Can we Rectify runtime errors?

A) No, we cannot rectify but we can handle.

Q) How to handle Runtime Errors?

A) By using Exception handling Mechanism.


- We can handle runtime errors in 2 ways.
  1. Logical Implementation.
  2. Try-catch Implementation.

Logical Implementation: The Programmer must analyze that if any runtime errors will occur he has to identify and handle the runtime error by writing some logic.


Ex:

```
➤ createprocedure Proc_div(@a int,@b int)
  asbegin
  declare @c int
  if (@b=0)
```


```
print'Denominator Must not be zero'
else
set @c=@a/@b
print'Quotient is: '+cast(@c asvarchar(20))
end
```

 Messages   
Command(s) completed successfully.

➤ **exec** proc\_div10,0

 Messages   
Denominator Must not be zero

➤ **exec** proc\_div10,2

 Messages   
Quotient is: 5

- It is difficult for the Programmer to write the logic each and every time.
- So to reduce the burden on the Programmer MS has given try-catch implementation.

Syntax:

Begin try


End try

Begin catch


End catch

Ex: With try catch implementation

```
➤ alterprocedure P1(@a int,@b int)
asbegin
begintry
declare @c int
set @c=@a/@b
print'Quotient is'+cast(@c asvarchar(20))
endtry
begincatch
print'Denominator must not be zero'
endcatch
end
```

 Messages   
Command(s) completed successfully.

➤ **exec** p110,0

 Messages   
Denominator must not be zero

## Triggers

Trigger is a Special type of stored procedure which will gets invoked immediately after performing DML Operations.

Triggers are of three types:

1. DDL Triggers
2. DML Triggers
3. Instead of Triggers

### 1. DDL Triggers:

These Triggers are used to create (or) modify (or) drop the Triggers.

### 2. DML Triggers:

These Triggers will invoke immediately after performing DML Operations like insert, update and delete.

Syntax to create a Trigger:

Create trigger tiggername on tablename

After insert/delete

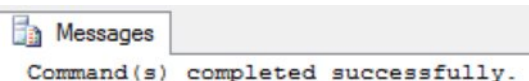
As

Begin

end

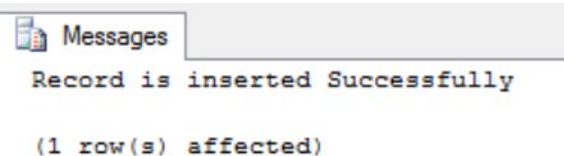
Q) Create trigger to display record is inserted successfully?

```
➤ createtrigger t1 on Employee
afterinsert
asbegin
print'Record is inserted Successfully'
end
```



Messages  
Command(s) completed successfully.

```
➤ insertinto employee values(106,'Swathi',23000)
```



Messages  
Record is inserted Successfully  
  
(1 row(s) affected)

- Whenever we execute the above inserted query first the record is inserted in Employee table and then the trigger will gets invoked Automatically.

Q) Create a trigger to display Employee details after inserting the record.

```
➤ createtrigger t2 on Employee
afterinsert
asbegin
select*from Employee
end
```

Messages  
Command(s) completed successfully.

```
➤ insertinto employee values(107,'Harish',26000)
```

	eno	ename	salary
1	101	Anil	20000.00
2	102	Ajay	23000.00
3	103	Surya	25000.00
4	104	Vinod	30000.00
5	105	Vijay	27000.00
6	106	Swathi	23000.00
7	107	Harish	26000.00

- Whenever we are working with Triggers internally, two tables are created.
  1. Insert table
  2. Delete table
 These two tables are called as magic tables.
- We cannot see the above two magic tables.
- Insert table is created after performing inserting operation and deleted table is created after performing deleting operation.

Q) Create a trigger to convert lower case into upper case and insert uppercase character

```
➤ createtrigger t3 on emp
afterinsert
asbegin
declare @eno int, @ename varchar(20),@desig varchar(50)
set @eno=(select eno from inserted)
set @ename=(select ename from inserted)
set @desig=(select desig from inserted)
update emp set
ename=upper(@ename),desig=upper(@desig)where eno=@eno
select*from emp
end
```

Messages  
Command(s) completed successfully.

```
➤ insertinto emp values(101,'anil','developer')
```

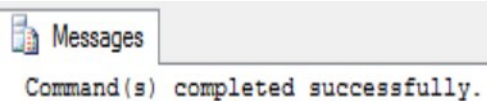
	eno	ename	Desig
1	101	ANIL	DEVELOPER

- Whenever we execute the above query first the record is inserted in emp table and then into inserted table.
- We have to get the data from insertedtable and store in TSQL variables and we have to convert the lower case to upper case characters and update in employee table.

Emp			Dept	
eno	ename	Sal	eno	dno
101	Anil	20000	101	

- Whenever we insert record in emp table then automatically insert enoin dept table  
Ex:

```
➤ createtrigger t4 on emp
afterinsert
asbegin
declare @eno int
set @eno=(select eno from inserted)
insertinto dept values(@eno,null)
select*from emp
select*from dept
end
```



```
➤ insertinto emp values(101,'Anil','Designer')
```

Results				Messages	
	eno	ename	Desig		
1	101	ANIL	DESIGNER		

	eno	dno
1	101	NULL

- Whenever we insert the record in E1 table then Automatically insert eno and ename in Empbackup table.
- Whenever we delete the record from E1 table then automatically insert the deleted values in eno and ename of Empbackup tables.

eno	ename
101	Anil

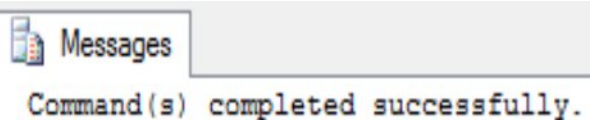
eno	ename	nenno	nename
101	Anil	NULL	NULL
NULL	NULL	101	Anil

Ex:

```

➤ createtrigger t5 on e1
afterinsert,delete
asbegin
declare @oeno int,@oename varchar(20),@nenno int,@nename
varchar(20)
set @oeno=(select eno from inserted)
set @oename=(select ename from inserted)
set @nenno=(select eno from deleted)
set @nename=(select ename from deleted)
insertinto empbackup
values(@oeno,@oename,@nenno,@nename)
select*from e1
select*from empbackup
end

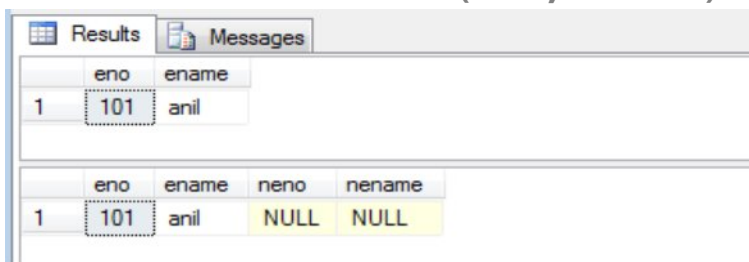
```



```

➤ insertinto e1 values(101,'anil')

```



	eno	ename
1	101	anil

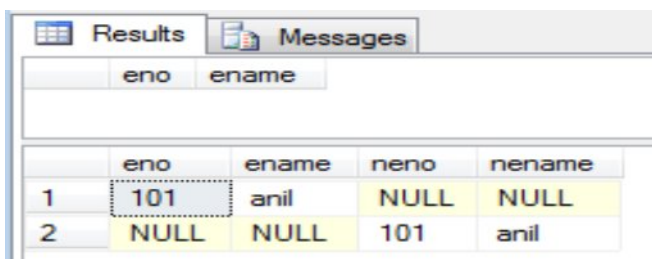
  

	eno	ename	nenno	nename
1	101	anil	NULL	NULL

```

➤ deletefrom e1 where eno=101

```



	eno	ename

	eno	ename	nenno	nename
1	101	anil	NULL	NULL
2	NULL	NULL	101	anil



## Views

**Views:** View is an imaginary table or virtual table.

- View is used to hide the confidential data from the table.
- View is a stored select query.
- Views are of 2 types:
  1. Simple View
  2. Complex View

**1. Simple View:** The View that was created on single table is called as Simple View.

**2. Complex View:** The View that was created on multiple tables is called as Complex View.

Syntax for creating View:

Create view viewname

As

Selected query

- View is used to provide security for the table data.


Employee

Eno	Ename	Design	Dno	Dname
101	ANIL	DEVELOPER	10	IT
102	AJAY	TEAMLEADER	10	IT
103	KIRAN	BDE	20	MARKETING
104	ARUN	BDM	20	MARKETING

- The Operations done on view table will affect the original table.

**Q) Create a view to display the Employee details who are working in IT department**

```
➤ createview itdept
as
select * from Employee where dname='IT'
```

 Messages   
Command(s) completed successfully.

```
➤ select * from itdept
```

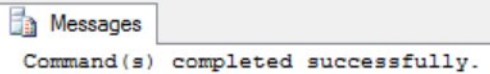

	Eno	Ename	Desig	Dno	Dname
1	101	ANIL	DEVELOPER	10	IT
2	102	AJAY	TEAMLEADER	10	IT

- We can perform DML, DRL Operations on Views

**Q) How to restrict uninserting Operation on view?**

A) With check Option

Ex:

- `createview itdeptcheck`  
`as`  
`select*from Employee where dname='IT'`  
`withcheckoption`
- 
- `select*from itdeptcheck`
- 

	Eno	Ename	Desig	Dno	Dname
1	101	ANIL	DEVELOPER	10	IT
2	102	AJAY	TEAMLEADER	10	IT
- `insertinto itdeptcheck`  
`values(106,'RAVI','DEVELOPER',10,'MARKETING')`

Error Message:

Msg 550, Level 16, State 1, Line 1  
The attempted insert or update failed because the target view either specifies WITH CHECK OPTION or spans a view that specifies WITH CHECK OPTION and one or more rows resulting from the operation did not qualify under the CHECK OPTION constraint.  
The statement has been terminated.

- With check option will not allow to insert other department details except IT dept.

Ex:

`insertinto itdeptcheck values(106,'RAVI','DEVELOPER',10,'IT')`



Q) How to View the Stored Query?

A) `sp_helptextviewname`

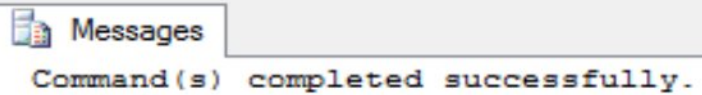
Ex: `sp_helptext itdeptcheck`

	Results	Messages
	Text	
1	create view itdeptcheck	
2	as	
3	select *from Employee where dname='IT'	
4	with check option	

- Whenever the user gives the above query, then he get the entire view code that was written in which the original tablename exists. So It must be hidden.

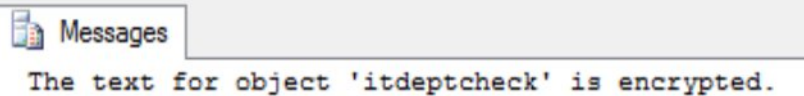
Q) How to hide the stored query?

- **alterview itdeptcheck withencryption**  
**as**  
**select\*from employee where dname='IT'**  
**withcheckoption**



**Note:** Here we have already created the view with viewname as 'itdeptcheck'. So for that we cannot create again the view with same name. To make changes on created view we have to use alter keyword.

- **sp\_helptext itdeptcheck**

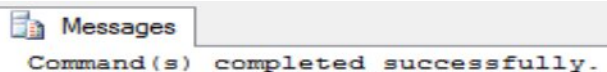


**Complex Views:** The view that was created on multiple tables is called as complex views.

Dept		Emp			
Dno	Dname	Eno	Ename	Salary	Dno
10	IT	101	Anil	20000	10
20	Designer	102	Sunil	23000	20
30	Marketing	103	Vijay	25000	10
		104	Surya	230000	30
		105	Ajay	27000	20

**Q) Create a view to display Emp details along with Dno and Dname?**

- **createview empdept**  
**as**  
**select e.\*,d.dname**  
**from emp e innerjoin dept d on e.dno=d.dno**



- **select\*from empdept**

	eno	ename	salary	dno	dname
1	101	Anil	20000.00	10	IT
2	102	Sunil	23000.00	20	Designer
3	103	Vijay	25000.00	10	IT
4	104	Surya	23000.00	30	Marketing
5	105	Ajay	27000.00	20	Designer

- **insertinto empdept values(106,'Kumar',30000,30,'Marketing')**

Error Messages:

**Msg 4405, Level 16, State 1, Line 1**  
**View or function 'empdept' is not updatable because the modification affects multiple base tables.**

- Whenever we execute the above insert query an Error message is displayed saying that we cannot insert the record in complex views as it affects multiple base tables.

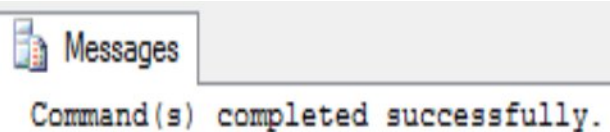
- If we want to insert the records in complex views we have to use Instead of Triggers.

### Instead of Triggers:

The Triggers that was created on Views are called as Instead of Triggers.

Ex:

- ```
createtrigger trig_empdept on empdept
insteadofinsert
asbegin
declare @eno int,@ename varchar(50),@salary money,@dno
int,@dname varchar(20)
set @eno=(select eno from inserted)
set @ename=(select ename from inserted)
set @salary=(select salary from inserted)
set @dno=(select dno from inserted)
set @dname=(select dname from inserted)
insertinto emp values(@eno,@ename,@salary,@dno)
insertinto dept values(@dno,@dname)
end
```

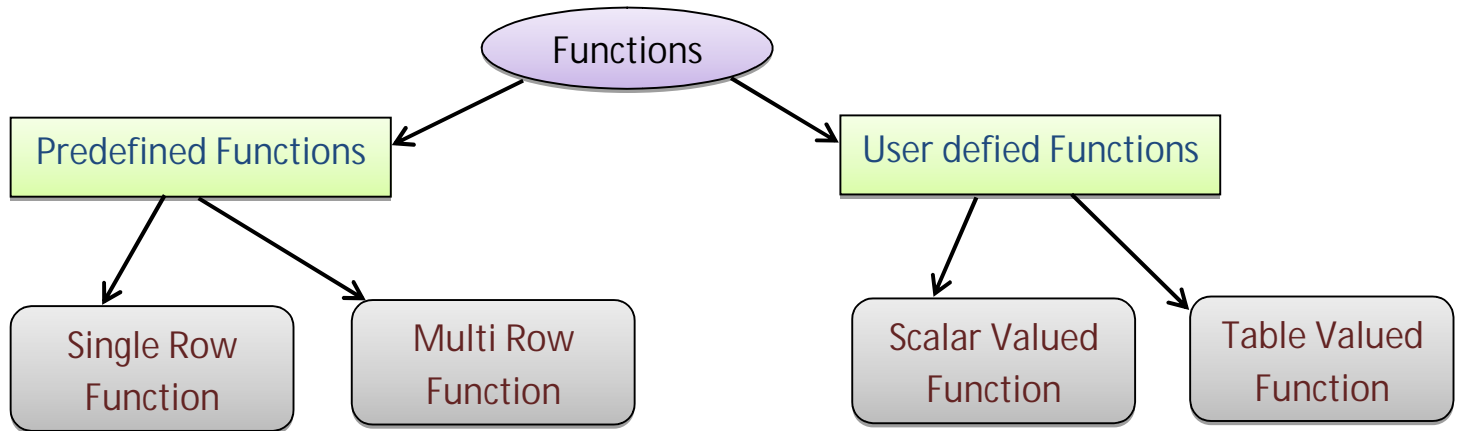


- ```
insertinto empdept values(106,'Kumar',30000,30,'Marketing')
```



### Functions (Remaining part)

**Functions:** Function is a Subprogram which is used to perform some operation and return some value.



**User defined Functions:** The function which was created by the programmer depending on the user requirement is called as User defined Function.

Two types of user defined functions are

1. Scalar Valued Functions
2. Table Valued Functions

1. **Scalar Valued Functions:** The Function which will take the input from the user and return only one value of any datatype is called as Scalar Valued Function.

Syntax to declare Scalar Valued function:

Create function functionname(Parameters)

Returns datatype

As begin

Declaration

Execution

Return value/variable/expression

End

**Q) Create a function to accept two numbers from user and return the sum?**

```
➤ createfunction f1(@a int,@b int)
returnsint
asbegin
declare @c int
set @c=@a+@b
return @c
end
```

Messages  
Command(s) completed successfully.

```
➤ select dbo.f1(10,20)
```

	(No column name)
1	30

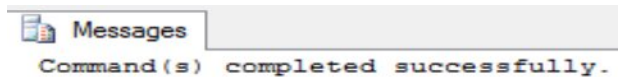
**Note:** Function will get executed when we call it.

Syntax to call the function:

Select dbo.functionname(values)

**Q) Create a function to display empname based on eno?**

```
➤ createfunction f2(@eno int)
returnsvarchar(20)
asbegin
declare @ename varchar(20)
set @ename =(select ename from emp where eno=@eno)
return @ename
end
```



```
➤ select dbo.f2(101)
```

	(No column name)
1	Anil

**2. Table Valued Functions:** The function which will take the input from the table is called as Table Valued Function.

- The return type of table valued functions is table.

Syntax for Table Valued Function:

Create function functionname(Parameters)

Returns table

As

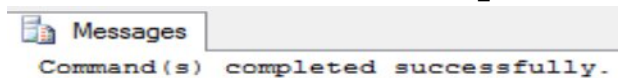
Return(select query)

Syntax to call table valued Function:

Select \* from dbo.functionname(values)

**Q) Create a function to display emp details based on empno.**

```
➤ createfunction f3(@eno int)
returnstable
as
return(select*from emp where eno=@eno)
```



```
➤ select*from dbo.f3(101)
```

	eno	ename	salary	dno
1	101	Anil	20000.00	10

### Q) Differences between Scalar valued and Table Valued Functions?

Scalar	Table
1. Scalar Valued Function will process on single row and return only one value	1. Table Valued Function will process on multiple rows and return multiple rows from the table.
2. The return type of <u>Scalar valued Function</u> is datatype.	2. The return type of <u>Table Valued Function</u> is Table.
3. <u>Scalar Valued Function</u> will always return a value/variable/Expression	3. <u>Table Valued Function</u> will always return select query.
4. <u>Scalar Valued Function</u> will have As begin End	4. <u>Table Valued Function</u> will not have As Begin End
5. Syntax: Select dbo.functionname(values)	5. Syntax: Select * from dbo.functionname(values)

### Q) Differences between Stored Procedure and Function

Stored Procedure	Function
1. Stored Procedure is a set of precompiled SQL Statements which will gets executed when we call it.	1. Function is a Sub program which is used to perform some Operation and return some value.
2. Stored Procedure will compile only once	2. Function will compile every time.
3. Stored Procedure <u>will have</u> execution plan	3. Function <u>will not have</u> execution plan.
4. Stored Procedure may or may not have input parameters.	4. Function must have at least one input parameter.
5. Stored Procedure <u>will support</u> exception handling	5. Function <u>does not support</u> exception handling.
6. Stored Procedure <u>will support</u> DML Commands Operations.	6. Function <u>does not support</u> DML Command Operations.
7. Stored procedure <u>will support</u> TCL Commands Operations like commit, rollback, savepoint.	7. Function <u>does not support</u> TCL Commands.
8. Stored Procedure <u>may (or) may not</u> return value.	8. Function <u>must return</u> some value.
9. Stored Procedure will gets executed by using exec command.	9. Function will gets executed by using select command.
10. Stored procedure will accept both input and output parameters.	10. Function will accept only input parameters.
11. We can call one Stored Procedure in	11. We can call one function in another

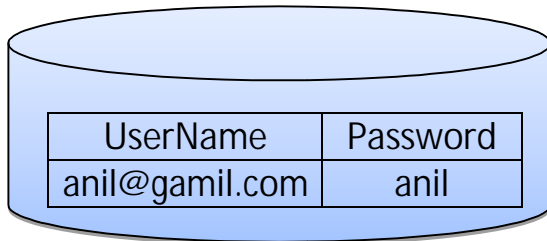


another Stored Procedure.	function.
12.We can call function in Stored Procedure.	12.We cannot call Stored Procedure in Function.

### Q) What is SQL Injection?

A) SQL Injection is a mechanism which is used to back the data from the database (or) which is used to inject the unsecure (or) unnecessary data into the database.

- It is a technique where malicious users(Hackers) can inject SQL Commands into an SQL Statement, via Web page input.



User Name

Password

Sign In

Version Control tools

To update details of server in company

**select\*from emp where eno=101 or 1=1**

### Q) How to prevent SQL Injection attacks?

1. By using parameterized queries.

Ex:

```
stringcommandtext = "select * from Login where username=@username
and password=@password";
SqlCommandcmd = newSqlCommand(commandtext, con);
cmd.Parameters.AddWithValue("@username", Login1.username);
cmd.Parameters.AddWithValue("@password", Login1.password);
```

2.Stored Procedures.

3. Encrypt sensitive data like password.

4. Perform validations for input fields.

5. Do not display predefined error message to the user.

6. Frequently we have to do code reviews.

### Pivot and Unpivot:

Ex:

**select\*from product**

Product

cname	Pname	amt
Anil	Shoes	300.00
Anil	Shirts	200.00
Ajay	Shoes	200.00
Ajay	Shirts	400.00
Anil	Shoes	600.00

Ex:

```
select cname,[Shoes] as 'Shoes',[Shirts] as 'Shirts'
from
(
select cname,pname,amt from Product) p
pivot
(sum(amt) for pname in([Shoes],[Shirts]))as
pivoting
```

Output:

	cname	Shoes	Shirts
1	Ajay	200.00	400.00
2	Anil	900.00	200.00

**Pivoting:** Changing the row data into columns

**Unpivoting:** Changing the column name to row data.

**Pivot:** Pivoting is a mechanism where we can interchange rows into columns.

**Unpivot:** Unpivot is a mechanism which is used to interchange column into rows.

**Q) What is the difference between union and Union all?**

A) Both the Operations are generally used to manage more than one table.

- By using these 2 operations we can get the data from more than one table into a single resultant.
- Union will not display duplicate records from both the tables.
- Union all will display duplicate records from both the tables.

IndianCustomers

id	name	email
1	Anil	anil@gmail.com
2	Sunil	sunil@gmail.com

UsCustomers

id	name	email
1	Anil	anil@gmail.com
2	Sunil	sunil@gmail.com

Ex1:

```
select * from IndianCustomers
union
select * from UsCustomers
```

Output:

	id	name	email
1	1	Anil	anil@gmail.com
2	2	Sunil	sunil@gmail.com
3	3	Ajay	ajay@gmail.com
4	3	Vijay	vijay@gmail.com

E2:

```
select * from IndianCustomers
union all
select * from UsCustomers
```

Output:

	id	name	email
1	1	Anil	anil@gmail.com
2	2	Sunil	sunil@gmail.com
3	3	Ajay	ajay@gmail.com
4	1	Anil	anil@gmail.com
5	2	Sunil	sunil@gmail.com
6	3	Vijay	vijay@gmail.com

**Cursor:** Cursor is a temporary SQL memory area which is used to fetch the data from the database and we can perform the operations from the Cursor.

Steps to work with Cursor:

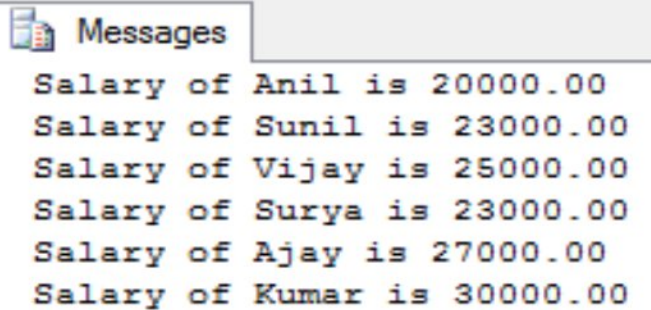
1. Declare the Cursor.
2. Open the Cursor.
3. Fetch the data from the Cursor.
4. Deallocate the Cursor.

Ex:

```
declare empcur cursor for
select ename,salary from emp
```

```
declare @ename varchar(20),@salary money
open empcur
fetchnextfrom empcur into @ename,@salary
while@@FETCH_STATUS=0
begin
print'Salary of '+@ename+' is '+cast(@salary asvarchar(20))
fetchnextfrom empcur into @ename,@salary
end
close empcur
```

Output:



The screenshot shows a 'Messages' window with the following output:

```
Salary of Anil is 20000.00
Salary of Sunil is 23000.00
Salary of Vijay is 25000.00
Salary of Surya is 23000.00
Salary of Ajay is 27000.00
Salary of Kumar is 30000.00
```