SQL

1. Operators in SQL ?
Arithmetic, Bitwise, Comparison, Logical

2. Aggregate Functions in SQL ?
Min, Max, Count Avg, Sum

3. Types of join in SQL ?
Inner, Left, Right, Full

Constraints in SQL

NOT NULL - Ensures that a column cannot have a NULL value
UNIQUE - Ensures that all values in a column are different
PRIMARY KEY - A combination of a NOT NULL and UNIQUE. Uniquely identifies each row in a table
FOREIGN KEY - Uniquely identifies a row/record in another table
CHECK - Ensures that all values in a column satisfies a specific condition
DEFAULT - Sets a default value for a column when no value is specified
INDEX - Used to create and retrieve data from the database very quickly

4. Self Join
To query the data and get information for both people in one row, you could self join like this:

```
select e1.EmployeeID,

       e1.FirstName,

       e1.LastName,

       e1.SupervisorID,

       e2.FirstName as SupervisorFirstName,

       e2.LastName as SupervisorLastNamefrom Employee e1

       left outer join Employee e2 on e1.SupervisorID = e2.EmployeeID
```

5. Clauses in SQL

1. SELECT
2. FROM
3. WHERE
4. GROUP BY
5. HAVING
6. ORDER BY

6. What is store Procedure?
It is a set of SQL statement stored in database which can be reused and shared by multiple program.

8. Function
It is simple a program that   accept the parameter does the work with respect to parameter and returns the type of result.

- Scalar user defined functions
It may or may not have input parameter but will always return a single value is called scalar functions.

```
create function GetAge(@Dob Date)
returns int
as
begin
 declare @age int
 set @age = datediff(year,@Dob,getdate())
 return @age
end
```

Why function is required

```
select [TitleOfCourtesy], FirstName, LastName, dbo.GetAge(BirthDate) from
Employees
where dbo.GetAge(BirthDate) > 60
```

- inline table valued function

1) The return type is always table.
2) The function body is not enclosed between begin and end block.
3) Inline table values function can be used to achieve the functionality of Parametrized view.

```
Create function GetEmployeeList(@Country varchar(max))
returns table
as
return (select * from Employees where Country like '%' + @Country + '%')
```

- Multistatement Values Function

1) It is similar to inline table values function, the important difference is in inline we cannot specify the structure of table but in multiline we can specify the structure of table.
2) It can also have begin and end block

```
create function GetMultiStatementTableValuesFunction()
returns  @table table (Id int, Name varchar(max))
as
begin
    insert into @table
    select EmployeeID, LastName from Employees
    return
end
```

9. Views

View can be said as virtual table, it is simple saved sql query.

```
create view vEmpOrder
as
select a.FirstName, a.LastName, a.Title, b.ShipVia, b.ShipName, b.ShipAddress
from Employees a
inner join Orders b on a.EmployeeID = b.EmployeeID
```

Advantages of views.

1. It reduces the database complexity for non IT users because joins and other calculations can be quite difficult to them.
2. It increases Row and column level security, for e.g. We have to give access of IT tables to IT Dpt manager, it we give access of table than he can see all the tables and will be able to access all data, but if we create view this will not happen, only specified column will be shown.
3. Views can be used to present aggregated data and hide detailed data from the user.

Updating Views

View can be updated
New data can be inserted in view
Data can be deleted in the view
Above condition forms to be true only when view is formed by single table,

When views are formed by multiple table than update may not take place accurately. For this purpose triggers are used

10. Exception Handling
Try Catch is used for exception handling

What is transaction ?
It is unit of work that is done on database manually by user or automatically by programm (store procedure).

The following example shows how to use the TRY…CATCH construct to handle errors that occur inside a transaction. The XACT_STATE function determines whether the transaction should be committed or rolled back. In this example, SET XACT_ABORT is ON. This makes the transaction uncommittable when the constraint violation error occurs.

```
begin try
    SET XACT_ABORT ON;
    begin transaction
    SELECT 1/0; -- Create error on purpose
    commit transaction
end try
begin catch
    SELECT ERROR_NUMBER() AS ErrorNumber, ERROR_MESSAGE() AS ErrorMessage;

    IF (XACT_STATE()) = -1
   BEGIN
       PRINT
           N'The transaction is in an uncommittable state.' +
           'Rolling back transaction.'
       ROLLBACK TRANSACTION;
   END;

    IF (XACT_STATE()) = 1
   BEGIN
       PRINT
           N'The transaction is committable.' +
           'Committing transaction.'
       COMMIT TRANSACTION;
   END;

end catch
```

Retrieving error Information

ERROR_NUMBER() returns the number of the error.

ERROR_SEVERITY() returns the severity.

ERROR_STATE() returns the error state number.

ERROR_PROCEDURE() returns the name of the stored procedure or trigger where the error occurred.

ERROR_LINE() returns the line number inside the routine that caused the error.

ERROR_MESSAGE() returns the complete text of the error message. The text includes the values supplied for any substitutable parameters, such as lengths, object names, or times.

11. **When and why to use Store procedure, Views and functions ?**

Store Procedure :

It is always good to use store practice to use it for Insert, Update delete operation

Views :

It is good to use views when only using select operation.

Functions :

Functions are used by stored procedure to format data, calculations, data manipulation etc.

12. Rank

It returns the rank of each row within the partition of result set.

Difference between rank and dense rank is rank function skips rank if there is tie whereas dense rank does not.

Rank and dense rank

```
Select ShipVia, ShipName,Freight,
RANK() OVER (partition by ShipName ORDER BY Freight desc) as [Rank],
dense_RANK() OVER (partition by ShipName ORDER BY Freight desc) as [dense_Rank]
from Orders
```

Rank and dense rank can be used to find nth highest salary

ROW_NUMBER()

It returns sequentials number of rows within the specified partition.

```
select ROW_NUMBER() OVER (order by freight desc) AS [Ser.no], * from Orders

Select ROW_NUMBER() OVER (order by freight desc) AS [Ser.no],
ShipVia, ShipName,Freight,
RANK() OVER ( ORDER BY Freight desc) as [Rank],
dense_RANK() OVER (ORDER BY Freight desc) as [dense_Rank]
from Orders
```

NTILE

1. Distributes the rows in specified number of groups
2. If the number of rows is not divisible by number of groups than we may get group of different sizes
3. Largest group come before smaller group.

```
select NTILE(2) OVER (partition by ShipCountry order by Freight desc) AS [Ntile],
* from Orders

Select ROW_NUMBER() OVER (order by freight desc) AS [Ser.no],
NTILE(2) OVER (partition by ShipCountry order by Freight desc) AS [Ntile],
ShipVia, ShipName,Freight,
RANK() OVER ( ORDER BY Freight desc) as [Rank],
dense_RANK() OVER (ORDER BY Freight desc) as [dense_Rank]
from Orders
```