1. What is DBMS?

A Database Management System (DBMS) is a program that controls creation, maintenance and use of a database. DBMS can be termed as File Manager that manages data in a database rather than saving it in file systems.

2. What is RDBMS?

RDBMS stands for Relational Database Management System. RDBMS store the data into the collection of tables, which is related by common fields between the columns of the table. It also provides relational operators to manipulate the data stored into the tables.

Example: SQL Server.

3. What is SQL?

SQL stands for Structured Query Language , and it is used to communicate with the Database. This is a standard language used to perform tasks such as retrieval, updation, insertion and deletion of data from a database.

Standard SQL Commands are Select.

4. What is a Database?

Database is nothing but an organized form of data for easy access, storing, retrieval and managing of data. This is also known as structured form of data which can be accessed in many ways.

Example: School Management Database, Bank Management Database.

5. What are tables and Fields?

A table is a set of data that are organized in a model with Columns and Rows. Columns can be categorized as vertical, and Rows are horizontal. A table has specified number of column called fields but can have any number of rows which is called record.

Example:.

Table: Employee.

Field: Emp ID, Emp Name, Date of Birth.

Data: 201456, David, 11/15/1960.

6. What is a primary key?

A primary key is a combination of fields which uniquely specify a row. This is a special kind of unique key, and it has implicit NOT NULL constraint. It means, Primary key values cannot be NULL.

7. What is a unique key?

A Unique key constraint uniquely identified each record in the database. This provides uniqueness for the column or set of columns.

A Primary key constraint has automatic unique constraint defined on it. But not, in the case of Unique Key.

There can be many unique constraint defined per table, but only one Primary key constraint defined per table.

8. What is a foreign key?

A foreign key is one table which can be related to the primary key of another table. Relationship needs to be created between two tables by referencing foreign key with the primary key of another table.

9. What is a join?

This is a keyword used to query data from more tables based on the relationship between the fields of the tables. Keys play a major role when JOINs are used.

11. What is normalization?

Normalization is the process of minimizing redundancy and dependency by organizing fields and table of a database. The main aim of Normalization is to add, delete or modify field that can be made in a single table.

12. What is Denormalization.

DeNormalization is a technique used to access the data from higher to lower normal forms of database. It is also process of introducing redundancy into a table by incorporating data from the related tables.

13. What are all the different normalizations?

The normal forms can be divided into 5 forms, and they are explained below -.

First Normal Form (1NF):.
This should remove all the duplicate columns from the table. Creation of tables for the related data and identification of unique columns.

Second Normal Form (2NF):.
Meeting all requirements of the first normal form. Placing the subsets of data in separate tables and Creation of relationships between the tables using primary keys.

Third Normal Form (3NF):.
This should meet all requirements of 2NF. Removing the columns which are not dependent on primary key constraints.

Fourth Normal Form (4NF):.
Meeting all the requirements of third normal form and it should not have multi- valued dependencies.

17. What is a Cursor?

A database Cursor is a control which enables traversal over the rows or records in the table. This can be viewed as a pointer to one row in a set of rows. Cursor is very much useful for traversing such as retrieval, addition and removal of database records.

18. What is a relationship and what are they?

Database Relationship is defined as the connection between the tables in a database. There are various data basing relationships, and they are as follows:.

One to One Relationship.
One to Many Relationship.
Many to One Relationship.
Self-Referencing Relationship.

19. What is a query?

A DB query is a code written in order to get the information back from the database. Query can be designed in such a way that it matched with our expectation of the result set. Simply, a question to the Database.

20. What is subquery?

A subquery is a query within another query. The outer query is called as main query, and inner query is called subquery. SubQuery is always executed first, and the result of subquery is passed on to the main query.

21. What are the types of subquery?

There are two types of subquery – Correlated and Non-Correlated.

A correlated subquery cannot be considered as independent query, but it can refer the column in a table listed in the FROM the list of the main query.

A Non-Correlated sub query can be considered as independent query and the output of subquery are substituted in the main query.

23. What is a trigger?

A DB trigger is a code or programs that automatically execute with response to some event on a table or view in a database. Mainly, trigger helps to maintain the integrity of the database.

Example: When a new student is added to the student database, new records should be created in the related tables like Exam, Score and Attendance tables.

24. What is the difference between DELETE and TRUNCATE commands?

DELETE command is used to remove rows from the table, and WHERE clause can be used for conditional set of parameters. Commit and Rollback can be performed after delete statement.

TRUNCATE removes all rows from the table. Truncate operation cannot be rolled back.

25. What are local and global variables and their differences?

Local variables are the variables which can be used or exist inside the function. They are not known to the other functions and those variables cannot be referred or used. Variables can be created whenever that function is called.

Global variables are the variables which can be used or exist throughout the program. Same variable declared in global cannot be used in functions. Global variables cannot be created whenever that function is called.

26. What is a constraint?

Constraint can be used to specify the limit on the data type of table. Constraint can be specified while creating or altering the table statement. Sample of constraint are.

NOT NULL.
CHECK.
DEFAULT.
UNIQUE.
PRIMARY KEY.
FOREIGN KEY.

27. What is data Integrity?

Data Integrity defines the accuracy and consistency of data stored in a database. It can also define integrity constraints to enforce business rules on the data when it is entered into the application or database.

28. What is Auto Increment?

Auto increment keyword allows the user to create a unique number to be generated when a new record is inserted into the table. AUTO INCREMENT keyword can be used in Oracle and IDENTITY keyword can be used in SQL SERVER.

Mostly this keyword can be used whenever PRIMARY KEY is used.

29. What is the difference between Cluster and Non-Cluster Index?

Clustered index is used for easy retrieval of data from the database by altering the way that the records are stored. Database sorts out rows by the column which is set to be clustered index.

A nonclustered index does not alter the way it was stored but creates a complete separate object within the table. It point back to the original table rows after searching.

30. What is Datawarehouse?

Datawarehouse is a central repository of data from multiple sources of information. Those data are consolidated, transformed and made available for the mining and online processing. Warehouse data have a subset of data called Data Marts.

31. What is Self-Join?

Self-join is set to be query used to compare to itself. This is used to compare values in a column with other values in the same column in the same table. ALIAS ES can be used for the same table comparison.

32. What is Cross-Join?

Cross join defines as Cartesian product where number of rows in the first table multiplied by number of rows in the second table. If suppose, WHERE clause is used in cross join then the query will work like an INNER JOIN.

33. What is user defined functions?

Three types of user defined functions are.

Scalar Functions.
Inline Table valued functions.
Multi statement valued functions.
Scalar returns unit, variant defined the return clause. Other two types return table as a return.

34. What is collation?

Collation is defined as set of rules that determine how character data can be sorted and compared. This can be used to compare A and, other language characters and also depends on the width of the characters.

ASCII value can be used to compare these character data.

35. What are all different types of collation sensitivity?

Following are different types of collation sensitivity -.

Case Sensitivity – A and a and B and b.
Accent Sensitivity.
Kana Sensitivity – Japanese Kana characters.
Width Sensitivity – Single byte character and double byte character.

36. Advantages and Disadvantages of Stored Procedure?

Stored procedure can be used as a modular programming – means create once, store and call for several times whenever required. This supports faster execution instead of executing multiple queries. This reduces network traffic and provides better security to the data.

Disadvantage is that it can be executed only in the Database and utilizes more memory in the database server.

37. What is Online Transaction Processing (OLTP)?

Online Transaction Processing (OLTP) manages transaction based applications which can be used for data entry, data retrieval and data processing. OLTP makes data management simple and efficient. Unlike OLAP systems goal of OLTP systems is serving real-time transactions.

Example – Bank Transactions on a daily basis.

38. What is CLAUSE?

SQL clause is defined to limit the result set by providing condition to the query. This usually filters some rows from the whole set of records.

Example – Query that has WHERE condition

Query that has HAVING condition.

39. What is recursive stored procedure?

A stored procedure which calls by itself until it reaches some boundary condition. This recursive function or procedure helps programmers to use the same set of code any number of times.

40. What is Union, minus and Interact commands?

UNION operator is used to combine the results of two tables, and it eliminates duplicate rows from the tables.

MINUS operator is used to return rows from the first query but not from the second query. Matching records of first and second query and other rows from the first query will be displayed as a result set.

INTERSECT operator is used to return rows returned by both the queries.

41. What is an ALIAS command?

ALIAS name can be given to a table or column. This alias name can be referred in WHERE clause to identify the table or column.

Example-.

Select st.StudentID, Ex.Result from student st, Exam as Ex where st.studentID = Ex. StudentID
Here, st refers to alias name for student table and Ex refers to alias name for exam table.

42. What is the difference between TRUNCATE and DROP statements?

TRUNCATE removes all the rows from the table, and it cannot be rolled back. DROP command removes a table from the database and operation cannot be rolled back.

43. What are aggregate and scalar functions?

Aggregate functions are used to evaluate mathematical calculation and return single values. This can be calculated from the columns in a table. Scalar functions return a single value based on the input value.

Example -.

Aggregate – max(), count - Calculated with respect to numeric.

Scalar – UCASE(), NOW() – Calculated with respect to strings.

44. How to fetch common records from two tables?

Common records result set can be achieved by -.

Select studentID from student INTERSECT Select StudentID from Exam

45. Which operator is used in query for pattern matching?

LIKE operator is used for pattern matching, and it can be used as -.

% - Matches zero or more characters.
_(Underscore) – Matching exactly one character.
Example -.

Select * from Student where studentname like 'a%'
Select * from Student where studentname like 'ami_'

46. What are Entities and Relationships?

Entity: An entity can be a real-world object, either tangible or intangible, that can be easily identifiable. For example, in a college database, students, professors, workers, departments, and projects can be referred to as entities. Each entity has some associated properties that provide it an identity.

Relationships: Relations or links between entities that have something to do with each other. For example - The employees table in a company's database can be associated with the salary table in the same database.

47. List the different types of relationships in SQL.

One-to-One - This can be defined as the relationship between two tables where each record in one table is associated with the maximum of one record in the other table.
One-to-Many & Many-to-One - This is the most commonly used relationship where a record in a table is associated with multiple records in the other table.
Many-to-Many - This is used in cases when multiple instances on both sides are needed for defining a relationship.
Self Referencing Relationships - This is used when a table needs to define a relationship with itself.

48. What is OLTP?

OLTP stands for Online Transaction Processing, is a class of software applications capable of supporting transaction-oriented programs. An essential attribute of an OLTP system is its ability to maintain concurrency. To avoid single points of failure, OLTP systems are often decentralized. These systems are usually designed for a large number of users who conduct short transactions. Database queries are usually simple, require sub-second response times and return relatively few records. Here is an insight into the working of an OLTP system [Note - The figure is not important for interviews] -

49. What are the subsets of SQL?

There is three significant subset of the SQL:

Data definition language (DDL):DDL is used to define the data structure it consists of the commands like CREATE, ALTER, DROP, etc.
Data manipulation language (DML):DML is used to manipulate already existing data in the database. The commands in this category are SELECT, UPDATE, INSERT, etc.
Data control language (DCL):DCL is used to control access to data in the database and includes commands such as GRANT, REVOKE.

50. What is a Data Definition Language?

Data definition language (DDL) is the subset of the database which defines the data structure of the database in the initial stage when the database is about to be created. It consists of the following commands: CREATE, ALTER and DELETE database objects such as schema, tables, view, sequence, etc.

51. What is a Data Manipulation Language?

Data manipulation language makes the user able to retrieve and manipulate data. It is used to perform the following operations.

Insert data into database through INSERT command.
Retrieve data from the database through SELECT command.
Update data in the database through UPDATE command.
Delete data from the database through DELETE command.

52. What is Data Control Language?

Data control language allows you to control access to the database. DCL is the only subset of the database which decides that what part of the database should be accessed by which user at what point of time. It includes two commands GRANT and REVOKE.

GRANT: to grant the specific user to perform a particular task

REVOKE: to cancel previously denied or granted permissions.

53. What is the difference between SQL and PL/SQL?

SQL or Structured Query Language is a language which is used to communicate with a relational database. It provides a way to manipulate and create databases. On the other hand, PL/SQL is a dialect of SQL which is used to enhance the capabilities of SQL. It was developed by Oracle Corporation in the early 90's. It adds procedural features of programming languages in SQL.

In SQL single query is being executed at once whereas in PL/SQL a whole block of code is executed at once.

SQL is like the source of data that we need to display on the other hand PL/SQL provides a platform where the SQL the SQL data will be shown.

SQL statement can be embedded in PL/SQL, but PL/SQL statement cannot be embedded in SQL as SQL do not support any programming language and keywords.

54. What is a "TRIGGER" in SQL?

A trigger allows you to execute a batch of SQL code when an insert, update or delete command is run against a specific table as TRIGGER is said to be the set of actions that are performed whenever commands like insert, update or delete are given through queries.
The trigger is said to be activated when these commands are given to the system.
Triggers are the particular type of stored procedures that are defined to execute automatically in place or after data modifications.
Triggers are generated using CREATE TRIGGER statement.

55. What is self-join and what is the requirement of self-join?

A self-join is often very useful to convert a hierarchical structure to a flat structure. It is used to join a table to itself as like if that is the second table.

56. What is the difference between BETWEEN and IN condition operators?

The BETWEEN operator is used to display rows based on a range of values. The values can be numbers, text, and dates as well. BETWEEN operator gives us the count of all the values occurs between a particular range.

The IN condition operator is used to check for values contained in a specific set of values. IN operator is used when we have more than one value to choose.

57. What is ACID property in a database?

ACID property is used to ensure that the data transactions are processed reliably in a database system.

A single logical operation of a data is called transaction.

ACID is an acronym for Atomicity, Consistency, Isolation, Durability.

Atomicity: it requires that each transaction is all or nothing. It means if one part of the transaction fails, the entire transaction fails and the database state is left unchanged.

Consistency: the consistency property ensure that the data must meet all validation rules. In simple words you can say that your transaction never leaves your database without completing its state.

Isolation: this property ensure that the concurrent property of execution should not be met. The main goal of providing isolation is concurrency control.

Durability: durability simply means that once a transaction has been committed, it will remain so, come what may even power loss, crashes or errors.

58. What is the difference between NULL value, zero and blank space?

A NULL value is not the same as zero or a blank space. A NULL value is a value which is 'unavailable, unassigned, unknown or not applicable.' On the other hand, zero is a number, and a blank space is treated as a character.

The NULL value can be treated as unknown and missing value as well, but zero and blank spaces are different from the NULL value.

59. What is the usage of SQL functions?

Functions are the measured values and cannot create permanent environment changes to SQL server. SQL functions are used for the following purpose:

To perform calculations on data
To modify individual data items
To manipulate the output
To format dates and numbers
To convert data types

60. What are the ways to get the identity column value in sql server?

Identity Column values are auto generated.
There are several ways in Sql server, to retrieve the last identity value that is generated. The most common way to use SCOPE_IDENTITY() built in function.
You can also use @@IDENTITY and IDENT_CURRENT("TABLE_NAME").
Difference:
SCOPE_IDENTITY() – Same Session and Same Scope
@@IDENTITY – Same Session and Across any Scope
IDENT_CURRENT("TABLE_NAME") – Specific table across any session and any scope.

61. What is the difference between Primary key and Unique Key constraint.

1. A table can have only one primary key, but more than one unique key.
2. Primary key does not allow nulls, where as unique key allow only one null.
3. Primary key is bidefault Clustered Index, but Unique key is bidefault NonClusterd Index.

62. What is Difference between WHERE and HAVING

1. Where Clause can be used with select, insert and update statements, where as having clause can be only used with the select statement.
2. Where filters rows before aggregation (Grouping),where as having filter groups, after the aggregation are performed.
3. Aggregation functions cannot be used in the where clause, unless it is in a sub query contained in a having clause, where as aggregate functions can be used in having clause.

63. What are the type of join in sql server?

Joins in SQL Server are used to retrieve data from 2 or more related tables. In general , tables are related to each other using foreign key constraints.
In Sql Server, there are different of Joins.
1.Inner Join 2. Outer Join 3. Cross Join
Outer Join are again devided into 1. Left Join 2. Right Join 3. Full Join.
Inner Join : it returns only the matching rows between both the tables. Non Matching rows are eliminated.
Left Join: It returns all the matching rows + Non matching rows from the left table.
Right Join : It returns all the matching rows + Non matching rows from the Right Table.
Full Join: It returns all rows from both left and right table, including the non-matching rows.
Cross Join -It produces the Cartesian product of the 2 tables involed in the join.

64. What are the ways to replace NULL values in sql server?

There are 3 ways to replace null values – ISNULL (), CASE STATEMENT, COALESEE FUNCTION.
COALESE FUNCTION RETURNS THE FIRST NON NULL VALUE.

65. What is the DIFFERENCE BETWEEN UNION AND UNION ALL

UNION AND UNION ALL OPERATOR IN SQL SERVER ARE USED TO COMBINE THE RESULTSET OF TWO OR MORE SELECT QUERIES. 1. UNION REMOVES DUPLICATE ROWS, WHERE AS UNION ALL DOES NOT.
2. UNION HAS TO PERFORM DISTINCT SORT TO REMOVE DUPLICATES, WHICH MAKES IT LESS FASTER THAN UNION ALL.
ORDER BY CLAUSE SHOULD BE USED ONLY ON THE LAST SELECT STATEMENT IN THE UNION QUERY

66. What is DIFFERENCE BETWEEN UNION AND JOIN

UNION COMBINES THE RESULT SET OF TWO OR MORE SELECT QUERY INTO A SINGLE RESULT SET WHICH INCLUDES ALL THE ROWS FROM ALL THE QUERY IN THE UNION, WHERE AS JOINS, RETRIVE DATA FROM TWO OR MORE TABLE BASED ON THE LOGICAL RELATIONSHIPS BETWEEN THE TABLES. IN SHORT, UNION COMBINES ROWS FROM 2 OR MORE TABLE, WHERE AS JOIN COMBINES COLUMNS FROM 2 OR MORE TABLES.
FOR UNION AND UNION ALL TO WORK, THE NUMBER, DATA TYPE AND ORDER OF THE COLUMN IN THE SELECT STATEMENT SHOULD BE SAME.

67. What is Stored procedure ?

A STORED PROCEDURE IS A GROUP OF T-SQL STATEMENTS, IF YOU HAVE A SITUATION WHERE YOU WRITE THE SAME QUERY OVER AND OVER AGAIN, YOU CAN SAVE THAT SPECIFIC QUERY AS A STORED PROCEDURE AND CALL IT JUST BY THE NAME.
Use CREATE PROCEDURE SPNAME / CREATE PROC SPNAME
TO EXECUTE THE STORED PROCEDURE :
spGetEmployees
EXEC spGetEmployees
Execute spGetEmployees
Parameters and variables have an prefix @ in their name
To view the text of the stored procedure : sp_helptext procedure_name
ALTER PROCEDURE PROC_NAME
DROP PROCEDURE PROC_NAME
TO ENCRYPT THE TEXT OF THE STORED PROCEDURE, USE WITH ENCRYPTION OPTION, IT IS NOT POSSIBLE TO VIEW THE TEXT OF AN ENCRYPTED SP.

68. What are the advantage of stored procedure ?

1. EXECUTION PLAN RETENTION AND REUSABILITY. (STORED PROCEDURE FIRST CHECK FOR SYNTAX ERROR , THEN COMPILED THE QUERY AND THEN GENERATE EXECUTION PLAN)
2. REDUCES NETWORK TRAFFIC
3. CODE REUSABILITY AND BETTER MAINTAINBILITY.
4. BETTER SECURITY
5. AVOIDS SQL INJECTION ATTACK

69. What are the built in string functions?

select ASCII('A') --Return the ASCII code value of a character

select CHAR(65) --Convert an ASCII value to a character

SELECT CHARINDEX('M','MAHESH') --Search for a substring inside a string starting from a specified location and return the position of the substring.

SELECT CONCAT('MAHESH',' KUMAR') --Join two or more strings into one string

SELECT DIFFERENCE('MAHESH','MAHISH') --Compare the SOUNDEX() values of two strings

SELECT FORMAT(GETDATE(),'dd-MM-yyyy hh:mm:ss tt') --Return a value formatted with the specified format and optional culture

select LEFT('MAHESH',4) --Extract a given a number of characters from a character string starting from the left

SELECT LEN('MAHESH KUMAR') --Return a number of characters of a character string

SELECT LOWER('MAHESH KUMAR') --Convert a string to lowercase

SELECT UPPER('mahesh kumar') -- convert a string to UPPER CASE

SELECT LTRIM('   MAHESH KUMAR') --Return a new string from a specified string after removing all leading blanks IN LEFT SIDE

SELECT RTRIM('MAHESH KUMAR   ') -- Return a new string from a specified string after removing all leading blanks IN RIGHT SIDE
SELECT PATINDEX('%st%', 'interesting data') --Returns the starting position of the first occurrence of a pattern in a specified expression, or zeros if the pattern is not found, on all valid text and character data types.

select REVERSE('UNIVERSE') -- Returns the string in the reverse order

SELECT QUOTENAME('MAHESH KUMAR',']') --Returns a Unicode string with the delimiters added to make the input string a valid delimited identifier

SELECT REPLACE('It is a good tea at the famous tea store.', 'tea', 'coffee') --Replace all occurrences of a substring, within a string, with another substring

SELECT REPLICATE('**',5) --Return a string repeated a specified number of times

select SPACE(100) --Returns a string of repeated spaces.

select STUFF('MAHESH',2,3,'***') --Delete a part of a string and then insert
another substring into the string starting at a specified position.

SELECT SUBSTRING('MAHESHKP',1,3) --Extract a substring within a string starting from a specified location with a specified length

70. What are date functions in sql server?

SELECT GETDATE()
Select datepart(day, getdate())
SELECT DATENAME(WEEKDAY,GETDATE())
SELECT DATEADD(DAY,10,GETDATE())
SELECT DATEDIFF(YEAR,'5-11-1991',GETDATE())
SELECT DATEFROMPARTS(2019,11,11)

71. What is the difference between cast and convert function?

CAST is based on ANSI Standard and Convert is specific to SQL Server. So IF PORTABILITY IS A CONCERN, and if you want to use script with other database applications, use CAST()
CONVERT provides more flexibility than cast. For example- its possible to control how you want datetime datatypes to be converted using styles with convert function.
Note- The general guideline is to use CAST function., unless you want to take advantage of the style functionality in Convert function.

72. What are the mathmatical functions in sql server?

ABS - ABS stands for absolute and returns the absolute positive number

CEILING – IT TAKES ONE PARAMETER. RETURNS THE VALUE WHICH IS GREATER THAN OR EQUAL TO THE PARAMETER.
FLOOR – IT TAKES ONE PARAMETER, RETURNS THE VALUE WHICH IS SMALLER THAN OR EQUAL TO THE PARAMETER.
POWER - RETURNS THE POWER VALUE OF THE SPECIFIED EXPREESSION,TO THE SPECIFIED POWER.
RAND - RETURNS A RANDOM FLOAT NUMBER BETWEEN 0 AND 1. RAND FUNCTION TAKES A OPTIONAL SEED PARAMETER. WHEN SEED VALUE IS SUPPLIED, THEN RAND FUNCTION ALWAYS RETURNS THE SAME VALUE FOR THE SAME SEED.
SQUARE - RETURNS THE SQUARE OF A GIVEN NUMBER.
SQRT - RETURNS THE SQUARE ROOT OF A GIVEN NUMBER
ROUND - ROUNDS THE GIVEN NUMERIC EXPRESSION BASED ON THE GIVEN LENGTH, THIS FUNCTION TAKES 3 PARAMETERS.
1.Numeric Expression : It is the number that we want to round.
2.Length Parameter specifies the no. of the digits that we want to round to , if the length is a positive number , then the rounding is applied for the decimal part, where as if the length is negative, then the rounding is applied to the number before the decimal.
3. The Optional Function Parameter – It is used to indicate rounding or truncation operations. 0 Indicates rounding. Non Zero Indicates truncation. Default , If not specified is 0.

```
SELECT ABS(-1212)
SELECT CEILING(11.156)
SELECT FLOOR(11.156)
SELECT POWER(11,2)
SELECT RAND() --RETURNS A VALUE BETWEEN 0 AND 1
SELECT SQUARE(9) -- RETURNS THE SQUARE VALUE OF THE GIVEN NUMBER
SELECT SQRT(121) --RETURNS THE SQUARE ROOT VALUE OF THE GIVEN NUMBER
SELECT ROUND(121.44,1)
```

73. How many type of User defined functions are there ?

IN SQL SERVER, THERE ARE 3 TYPE OF USER DEFINED FUNCTIONS.
1. SCALAR FUNCTION
2. INLINE TABLE-VALUED FUNCTION
3. MULTI-STATEMENT TABLE-VALUED FUNCTION

74. What is Scalar function in sql server?

SCALAR FUNCTION MAY OR MAY NOT HAVE PARAMETERS , BUT ALWAYS RETURN A SINGLE (SCALAR) VALUE. THE RETURNED VALUE CAN BE OF ANY DATA TYPE EXCEPT TEXT, NTEXT, IMAGE, CURSOR AND TIMESTAMP.
WHEN CALLING A SCALAR USER DEFINED FUNCTIONS, YOU MUST SUPPLY A TWO PART NAME.
OWNER NAME . FUNCTION_NAME -- DBO STANDS FOR DATABASE OWNER.
YOU CAN ALSO CALL IT USING THE COMPLETE 3 PART NAME –
DATABASE_NAME.OWNER_NAME.FUNCTION_N
SCALAR USER DEFINED FUNCTION CAN BE USED IN THE SELECT CLAUSE AND SELECT WHERE CLAUSE
SCALAR FUNCTION CAN BE USED IN SELECT STATEMENT , BUT YOU CAN NOT USE STORED PROCEDURE IN A SELECT OR WHERE CLAUSE

75. What is inline table valued function?

INLINE TABLE VALUED FUNCTION RETURNS A TABLE.
2. WE SPECIFY TABLE AS THE RETURN TYPE, INSTEAD OF ANY SCALAR DATA TYPE.
3. THE FUNCTION BODY IS NOT ENCLOSED BETWEEN BEGIN AND END BLOCK.
4. THE STRUCTURE OF THE TABLE THAT GETS RETURNED, IS DETERMINED BY THE SELECT STATEMENT WITH IN THE FUNCTION.

76. WHERE CAN WE USE INLINE TABLE VALUED FUNCTION?

INLINE TABLE VALUED FUNCTION CAN BE USED TO ACHIVE THE FUNCTIONALITY OF PARAMETERIZED VIEWS
WE CAN USE WHERE CLAUSE IN INLINE TABLE VALUED FUNCTION CALLING TIME.
WE CAN UPDATE A TABLE USING INLINE TABLE VALUED FUNCTION CALLING TIME .
THE TABLE RETURNED BY THE TABLE VALUED FUNCTION , CAN ALSO BE USED IN JOINS WITH OTHER TABLE.

```
CREATE FUNCTION fnGetDataByDock()
returns table
as return
(
 select top 10 * from mstcustomer
)
-- select * from dbo.fnGetDataByDock() //calling table valued function
```

77. What is the Multi statement table valued function?

MULTI-STATEMENT TABLE VALUED FUNCTIONS ARE VERY SIMILAR TO INLINE TABLE VALUED FUNCTION, WITH A FEW DIFFERENCES.

```
create function fnGetDataByMSTV()
returns @table table (name nvarchar(50), age int)
as
begin
 insert into @table values('Mahesh',28)
 return

end

select * from dbo.fnGetDataByMSTV()  --calling multi statement table valued function
```

78. What is the difference between Inline table valued function and multi statement table valued function?

1.IN AN TABLE VALUED FUNCTION, THE RETURN CLAUSE CANNOT CONTAIN THE STRUCTURE OF THE TABLE, THE FUNCTION RETURNS. WHERE AS , WITH THE MULTI-STATEMENT TABLE VALUED FUNCTION, WE SPECIFY THE STRUCTURE OF THE TABLE THAT GETS RETURNED.
2. INLINE TABLE VALUED FUNCTION CANNOT HAVE BEGIN AND END BLOCK, WHERE AS THE MULTI STATEMENT TABLE VALUED FUNCTION CAN HAVE.
3. INLINE TABLE VALUED FUNCTION ARE BETTER FOR PERFORMANCE, THAN MULTI STATEMENT TABLE VALUED FUNCTION. IF THE GIVEN TASK , CAN BE ACHIVED USING AN INLINE TABLE VALUED FUNCTION, ALWAYS PREFER TO USE THEM, OVER MULTI STATEMENT TABLE VALUED FUNCTION.
4. ITS POSSIBLE TO UPDATE THE UNDERLYING TABLE, USING AN INLINE TABLE VALUED FUNCTION , BUT NOT POSSIBLE USING MULTI-STATEMENT TABLE VALUED FUNCTION.
5. REASON FOR IMPROVED PERFORMANCE OF AN INLINE TABLE VALUED FUNCTION: INTERNALLY, SQL SERVER TREATS AN INLINE TABLE VALUED FUNCTION MUCH LIKE IT WOULD A VIEW AND TREATS A MULTI STATEMENT TABLE VALUED FUNCTION SIMILAR TO HOW IT WOULD A STORED PROCEDURE.

79. What is DETERMINISTIC AND NON-DETERMINISTIC FUNCTION

DETERMINISTIC FUNCTION ALWAYS RETURN THE SAME RESULT ANY TIME THEY ARE CALLED WITH A SPECIFIC SET OF INPUT VALUES AND GIVEN THE SAME STATE OF DATABASE.
EXAMPLES: SQUARE() , POWER(), SUM() , AVG(), AND COUNT()
NOTE – ALL AGGREGATE FUNCTION ARE DETERMINISTIC FUNCTIONS.
NON-DETERMINISTIC FUNCTION MAY RETURN DIFFERENT RESULTS EACH TIME THEY ARE CALLED WITH A SPECIFIC SET OF INPUT VALUES EVEN IF THE DATABASE STATE THAT THEY ACCESS REMAINS THE SAME.

EXAMPLE- GETDATE() AND CURRENT_TIMESTAMP
RAND FUNCTION IS A NON-DETERMINISTIC FUNCTION, BUT IF YOU PROVIDE THE SEED VALUE , THE FUNCTON BECOMES DETERMINISTIC, AS THE SAME VALUE GETS RETURNED FOR THE SAME SEED VALUE.

80. What is WITH ENCRYPTTION AND SCHEMABINDING?

ENCRYPTING A FUNCTION DEFINITION USING WITH ENCRPTION OPTION.
YOU CAN ALSO ENCRYPT THE FUNCTION TEXT. ONCE ENCRYPTED, YOU CANNOT ENCRYPT A FUNCTION TEXT, USING SP_HELPTEXT SYSTEM STORED PROCEDURE.IF YOU TRY TO , YOU WILL GET A ERROR MESSAGE STATING, "THE TEXT FOR OBJECT IS ENCRYPTED"
CREATING A FUNCTION WITH SCHEMABINDING OPTION:
SCHEMABINDING SPECIFIES THAT THE FUNCTION IS BOUND TO THE DATABASE OBJECTS THAT IT REFERENCES. WHEN SCHEMABINDING IS SPECIFIED, THE BASE OBJECTS CANNOT BE MODIFIED IN ANY WAY THAT WOULD AFFECT THE FUNCTION DEFINITION. THE FUNCTION DEFINITION ITSELF MUST FIRST BE MODIFIED OR DROPPED TO REMOVE DEPENDENCIES ON THE OBJECT THAT IT IS TO BE MODIFIED.

81. What is Temporary Table ?

TEMPORARY TABLES, ARE VERY SIMILAR TO PERMANENT TABLES. PERMANENT TABLES GET CREATED IN THE DATABASE YOU SPECIFIY. AND REMAIN IN THE DATABASE PERMANENTLY UNTIL YOU DELETE (DROP) THEM. ON THE OTHER HAND, TEMPORARY TABLES GET CREATED IN THE TEMPDB AND ARE AUTOMATICALLY DELETED , WHEN THEY ARE NO LONGER USED.
2 TYPE OF TEMPORARY TABLE ARE THERE .
1. LOCAL TEMPORARY TABLE
2. GLOBAL TEMPORARY TABLE

82. WHAT IS LOCAL TEMPORARY TABLE?

CREATE TABLE #tblPerson (id int, name nvarchar(50))
CHECK IF THE LOCAL TEMPORARY TABLE IS CREATED:
TEMPORARY TABLES ARE CREATED IN THE TEMPDB, QUERY THE SYSOBJECTS SYSTEM TABLE IN TEMPDB. THE NAME OF THE TABLE, IS SUFFIEXED WITH LOT OF UNDERSCORE AND A RANDOM NUMBER. FOR THIS REASON YOU HAVE TO USE THE LIKE OPERATOR IN THE QUERY.
select name from tempdb..sysobjects where name like '%#tblPerson%'
A LOCAL TEMPORARY TABLE IS AVAILABLE , ONLY FOR THE CONNECTION THAT HAS CREATED THE TABLE.
A LOCAL TEMPORARY TABLE IS AUTOMATICALLY DROPPED, WHEN THE CONNECTION THAT HAS CREATED IT, IS CLOSED.

```
if object_id('tempdb..#tblTempData') is not null drop table #tblTempData
create table #tblTempData (Name nvarchar(50), Age int, Address nvarchar(50), Contact
nvarchar(50))

select * from #tblTempData
```

IF THE USER WANTS TO EXPLICITLY DROP THE TEMPORARY TABLE, THEN HE CAN DO SO USING :
DROP TABLE #tblPerson
IF THE TEMPORAR TABLE , IS CREATED INSIDE THE STORED PROCEDURE, IT GETS DROPPED AUTOMATICALLY UPON THE COMPLETION OF STORED PROCEDURE EXECUTION.
IT IS ALSO POSSIBLE FOR DIFFERENT CONNECTIONS, TO CREATE A LOCAL TEMPORARY TABLE WITH THE SAME NAME. FOR EXAMPLE USER1 AND USER2 , BOTH CAN CREATE A LOCAL TEMPORARY TABLE WITH THE SAME NAME #tblPERSONS.

83. WHAT IS GLOBAL TEMPORARY TABLE?

TO CREATE A GLOBAL TEMPORARY TABLE, PREFIX THE NAME OF THE TABLE WITH 2 POUND (##) SYMBOLS
CREATE TABLE ##tblPerson (id int , name nvarchar(50))

GLOBAL TEMPORARY TABLES ARE VISIBLE TO ALL THE CONNECTIONS OF THE SQL SERVER, AND ARE ONLY DESTROYED WHEN THE LAST CONNECTION REFERENCING THE TABLE IS CLOSED.
MULTIPLE USERS, ACROSS MULTIPLE CONNECTIONS CAN HAVE LOCAL TEMPORARY TABLES WITH THE SAME NAME, BUT, A GLOBAL TEMPORARY TABLE NAME HAS TO BE UNIQUE, AND IF YOU INSPECT THE NAME OF THE GLOBAL TEMP TABLE, IN THE OBJECT EXPLORER, THERE WILL BE NO RANDOM NUMBERS SUFFIXED AT THE END OF THE TABLE NAME.

84. What is DIFFERENCE BETWEEN LOCAL AND GLOBAL TEMPORARY TABLE

1. LOCAL TEMP TABLES ARE PREFIXED WITH SINGLE POUND (#) SYMBOL.WHERE AS GLOBAL TEMPORARY TABLE ARE PREFIEXED WITH 2 POUND (##) SYMBOLS.

2. SQL SERVER APPENDS SOME RANDOM NUMBERS AT THE END OF THE LOCAL TEMP TABLE NAME, WHERE THIS IS NOT DONE FOR GLOBAL TEMP TABLE NAMES.

3. LOCAL TEMPORARY TABLE ARE ONLY VISIBLE TO THAT SESSION OF THE SQL SERVER WHICH HAS CREATED IT, WHERE AS GLOBAL TEMPORARY TABLES ARE VISIBLE TO ALL THE SQL SERVER SESSIONS.

4. LOCAL TEMPORARY TABLES ARE AUTOMATICALLY DROPPED, WHEN THE SESSION THAT CREATED THE TEMPORARY TABLES IS CLOSED, WHERE AS GLOBAL TEMPORARY TABLES ARE DESTORYED WHEN THE LAST CONNECTION THAT IS REFERENCING THE GLOBAL TEMP TABLE IS CLOSED.

85. Why index is used in sql server?

INDEXS ARE USED BY QUERIES TO FIND DATA FROM TABLES QUICKLY. INDEXES ARE CREATED ON TABLES AND VIEWS. INDEX ON A TABLE OR VIEW, IS VERY SIMILAR TO THE INDEX THAT WE FIND IN A BOOK.
IF YOU DON'T HAVE AN INDEX , AND I ASK YOU TO LOCATE A SPECIFIC CHAPTER IN THE BOOK, YOU WILL HAVE TO LOOK AT EVERY PAGE STARTING FROM THE FIRST PAGE OF THE BOOK.
ON THE OTHER HAND, IF YOU HAVE THE INDEX, YOU LOOK UP THE PAGE NUMBER OF THE CHAPTER IN THE INDEX. AND THEN DIRECTLY GO TO THE PAGE NUMBER TO LOCATE THE CHAPTER. OBVIOUSLY, THE BOOK INDEX IS HELPING TO DRASTICALLY REDUCE THE TIME IT TAKES TO FIND THE CHAPTER.
IN A SIMILAR WAY, TABLE AND VIEW INDEXES, CAN HELP THE QUERY TO FIND DATA QUICKLY.
IN FACT, THE EXISTANCE OF THE RIGHT INDEXES, CAN DRASTICALLY IMPROVE THE PERFORMANCE OF THE QUERY. IF THERE IS NO INDEX TO HELP THE QUERY, THEN THE QUERY ENGINE, CHECKS EVERY ROW IN THE TABLE FROM THE BEGINNING TO THE END. THIS IS CALLED TABLE SCAN. TABLE SCAN IS BAD FOR PERFORMANCE.

86. What is Clustered Index ?

A CLUSTERED INDEX DETERMINES THE PHYSICAL ORDER OF DATA IN A TABLE. FOR THIS REASON, A TABLE CAN HAVE ONLY ONE CLUSTERED INDEX.
PRIMARY KEY CONSTRAINT CREATE CLUSTERED INDEXS AUTOMATICALLY IF NO CLUSTERED INDEX ALREADY EXISTS ON TABLE.
A CLUSTERED INDEX IS ANALOGOUS TO A TELEPHONE DIRECTORY, WHERE THE DATA IS ARRANGED BY THE LAST NAME. WE JUST LEARNED THAT A TABLE CAN HAVE ONLY ONE CLUSTERED INDEX. HOWEVER AN INDEX CAN CONTAIN MULTIPLE COLUMNS (A COMPOSITE INDEX), LIKE THE WAY A TELEPHONE DIRECTORY IS ORGANIZED BY THE LAST NAME AND THE FIRST NAME.

87. What is Non-Clustered Index?

A NON CLUSTERED INDEX IS ANALOGOUS TO AN INDEX OF A TEXTBOOK. THE DATA IS STORED IN ONE PLACE, THE INDEX IN ANOTHER PLACE. THE INDEX WILL HAVE POINTERS TO THE STORAGE LOCATION OF THE DATA.

SINCE THE NONCLUSTERED INDEX IS STORED SEPARATELY FROM THE ACTUAL DATA, A TABLE CAN HAVE MORE THAN 1 NON CLUSTERED INDEX, JUST LIKE HOW A BOOK CAN HAVE AN INDEX BY CHAPTERS AT THE BEGINNING AND ANOTHER INDEX BY COLUMN TERMS BY THE END.
IN THE INDEX ITSELF, THE DATA IS STORED IN AN ASNENDING OR DESCENDING ORDER OF THE INDEX KEY, WHICH DOES NOT IN ANY WAY INFLUENCE THE STORAGE OF DATA IN THE TABLE.

88. What is the difference between Cluster and Non-Cluster Index?

1. ONLY ONE CLUSTERED INDEX PER TABLE, WHERE AS YOU CAN HAVE MORE THAN ONE NON CLUSTERED INDEX.
2. CLUSTERED INDEX IS FASTER THAN A NON CLUSTED INDEX, BECAUSE THE CLUSTERED INDEX HAS TO REFER BACK TO THE TABLE, IF THE SELECTED COLUMN IS NOT PRESENT IN THE INDEX.
3. CLUSTERED INDEX DITERMINES THE STORAGE ORDER OF ROWS IN THE TABLE, AND HENCE DOES NOT REQUIRE ADDITIONAL DISK SPACE, BUT WHERE AS A NONCLSTERED INDEX IS STORED SEPARATELY FROM THE TABLE.ADDITION SPACE IS REQUIRED.

89. What is Unique Index?

UNIQUE INDEX IS USED TO ENFORCE UNIQUENESS OF KEY VALUES IN THE INDEX.
NOTE – BY DEFAULT , PRIMARY KEY CONSTRAINT , CREATES A UNIQUE CLUSTERED INDEX.
UNIQUENESS IS A PROPERTY OF AN INDEX, AND BOTH CLUSTERED AND NON CLUSTERED INDEX CAN BE UNIQUE.

```
CREATE UNIQUE NONCLUSTERED INDEX
UIX_TBLEMPLOYEE_FIRSTNAME_LASTNAME
ON TBLEMPLOYEE (FIRST_NAME, LAST_NAME)
```

90. What is the difference between Unique Constaint and Unique Index?

THERE ARE NO MAJOR DIFFERENCE BETWEEN A UNIQUE CONSTRAINT AND A UNIQUE INDEX. IN FACT, WHEN YOU ADD A UNIQUE CONSTRAINT, A UNIQUE INDEX GETS CREATED BEHIND THE SCENES.
WHILE CREATING UNIQUE CONSTRINT, A NON CLUSTERED UNIQUE INDEX GETS CREATED BI DEFAULT.

91. Points to remember for Primary key and unique constraint

1. BY DEFAULT, A PRIMARY KEY CONSTRAINT, CREATES A UNIQUE CLUSTERED INDEX , WHERE AS A UNIQUE CONSTRAINT CREATES A UNIQUE NON-CLUSTERED INDEX, THESE DEFAULTS CAN BE CHANGED IF YOU WISH TO.
2. A UNIQUE CONSTRAINT OR A UNIQUE INDEX CAN NOT BE CREATED ON AN EXISTING TABLE, IF THE TABLE CONTAINS DUPLICATE VALUES IN THE KEY COLUMNS. OBVIOUSLY TO SOLVE THIS, REMOVE THE KEY COLUMNS FROM THE INDEX DEFINITION OR DELETE OR UPDATE THE DUPLICATE VALUES.

92. What are the disadvantages of Index?

1. ADDITIONAL DISK SPACE : CLUSTERED INDEX DOES NOT REQUIRE ANY ADDITIONAL STORAGE. EVERY NON CLUSTERED INDEX REQUIERES ADDITIONAL SPACE AS IT IS STORED SEPARATELY FROM THE TABLE. THE AMOUNT OF SPACE REQUIRED WILL DEPEND ON THE SIZE OF THE TABLE, AND THE NUMBER AND TYPE OF COLUMNS USED IN THE INDEX.
2. INSERT , UPDATE AND DELETE STATEMENTS CAN BECOME SLOW : WHEN DML (DATA MANIPULATION LANGUAGE) STATEMENTS (INSERT , UPDATE, DELETE) MODIFIES THE DATA IN A TABLE, THE DATA IN ALL THE INDEXES ALSO NEEDS TO BE UPDATED. INDEXES CAN HELP , TO SEARCH, AND LOCATE THE ROWS, THAT WE WANT TO DELETE, BUT TOO MANY INDEXES TO UPDATE CAN ACTUALLY HURT THE PERFORMANCE OF THE DATA MODIFICATIONS.
3. WHAT IS A COVERING QUERY : IF ALL THE COLUMNS THAT YOU HAVE REQUESTED IN THE SELECT CLAUSE OF QUERY , ARE PRESENT IN THE INDEX , THEN THERE IS NO NEED TO LOOKUP IN THE TABLE AGAIN. THE REQUESTED COLUMNS DATA CAN SIMPLY BE RETURNED FROM THE INDEX.
4. A CLUSTERED INDEX, ALWAYS COVERS A QUERY : SINCE IT CONTAINS ALL OF THE DATA IN A TABLE. A COMPOSITE INDEX IS AN INDEX ON TWO OR MORE COLUMNS. BOTH CLUSTERED AND

NON CLUSTERED INDEXES CAN BE COMPOSITE INDEX. TO A CERTAIN EXTENT, A COMPOSITE INDEX , CAN COVER A QUERY.

93. What is View?

A VIEW IS NOTHING MORE THAN A SAVED SQL QUERY. A VIEW CAN ALSO BE CONSIDERED AS A VIRTUAL TABLE.

```
CREATE VIEW vwEmployeesByDepartment
as
SELECT E.Name, E.Gender, E.Salary, D.DeptName FROM tblEmployee E
LEFT JOIN tblDept D ON D.ID=E.DeptID

--view the data
select * from vwEmployeesByDepartment
```

ADVANTAGES OF VIEW:

1. VIEWS CAN BE USED TO REDUCE THE COMPEXITY OF THE DATABASE SCHEMA.
2. VIEWS CAN BE USED AS A MECHANISM TO IMPLEMENT ROW AND COLUMN LEVEL SECURITY.
3. VIEWS CAN BE USED TO PRESENT AGGREGATED DATA AND HIDE DETAILED DATA.
4. VIEWS DOES NOT STORE DATA.

TO MODIFY A VIEW – ALTER VIEW STATEMENT
TO DROP A VIEW – DROP VIEW STATEMENT

94. What is UPDATABLE VIEWS?

WE CAN INSERT, UPDATE OR DELETE IN THE BASE TABLE USING VIEW.

```
--update using view
UPDATE vwEmployeesByDepartment set salary=11000 where name='ABC'

--delete from view
delete from vwEmployeesByDepartment where Name='XYZ'
```

IF A VIEW IS BASED ON MULTIPLE TABLES, AND IF YOU UPDATE THE VIEW, IT MAY NOT UPDATE THE UNDERLYING BASE TABLES CORRECTLY. TO CORRECTLY UPDATE A VIEW, THAT IS BASED ON MULTIPLE TABLES, INSTEAD OF TRIGGERS ARE USED.

95. What is INDEXED VIEW?

A STANDARD OR NON-INDEXED VIEW, IS JUST A STORED SQL QUERY. WHEN WE TRY TO RETRIVE DATA FROM THE VIEW, THE DATA IS ACTUALLY RETRIVED FROM THE UNDERLYING BASE TABLES. SO A VIEW IS JUST A VIRTUAL TABLE. IT DOES NOT STORE ANY DATA. BY DEFUALT.
HOWEVER WHEN YOU CREATE AN INDEX, ON A VIEW, THE VIEW GETS MATERIAIIAZED , THIS MEANS, THE VIEW IS NOW , CAPABLE OF STORING DATA.
IN SQL SERVER, WE CALL THEM INDEXED VIEWS AND IN ORACLE- MATERIALIZED VIEWS.
GUIDELINES FOR CREATING INDEXED VIEWS:

1. THE VIEW SHOULD BE CREATED WITH SCHEMABINDING OPTION.
2. IF AN AGREEGATE FUNCTION IN THE SELECT LIST, REFERENCES AN EXPRESSION, AND IF THERE IS A POSSIBILITY FOR THAT EXPRESSION TO BECOME NULL, THEN A REPLACEMENT VALUE SHOULD BE SPECIFIED.
3. IF GROUP BY IS SPECIFIED, THE VIEW SELECT LIST MUST CONTAIN A COUNT_BIG(*) EXPRESSION.
4. THE BASE TABLES IN THE VIEW, SHOULD BE REFERENCED WITH 2 PART NAME.

```
create view vwTotalSalesByProduct
with schemabinding
```

```
as
select Name, sum(isnull((QuantitySold * Unitprice),0)) as TotalSales, COUNT_BIG(*) as
TotalTransactions from dbo.tblProductSales
Join dbo.tblProduct on dbo.TBLPRODUCT.PRODUCTID=DBO.TBLPRODUCTSALES.PRODUCTID
GROUP BY NAME

SELECT * FROM vwTotalSalesByProduct

CREATE UNIQUE CLUSTERED INDEX UIX_vwTotalSalesByProduct_Name
on vwTotalSalesByProduct (Name)
```

96. What is View Limitations?

1. YOU CANNOT PASS PARAMETERS TO A VIEW. INLINE TABLE VALUED FUNCTIONS ARE AN EXCELLENT REPLACEMENT FOR PARAMETERIZED VIEWS.
2. RULES AND DEFAULTS CAN NOT BE ASSOCIATED WITH VIEWS.
3. THE ORDER BY CLAUSE IS INVALID IN VIEWS UNLESS TOP or FOR XML IS ALSO SPECIFIED.
4. Views cannot be based on temporary tables.

97. What are type of triggers?

In SQL Server, there are 3 type of Triggers.
1. DML Triggers
2. DDL Triggers
3. Logon Triggers

DML Triggers are fired automatically in response to the DML events (Insert, Update & Delete)
DML Triggers can be again classified into 2 types.
1. After Triggers (Sometimes called a FOR Triggers)
2. Instead of Triggers

After Triggers, fires after the triggering action, The Insert, Update, Delete statements , causes an after trigger to fire after the respective statements complete execution.
Instead of Triggers, fires instead of triggering action. The Insert, Update, Delete statements, causes an instead of trigger to fire instead of respective statement execution.

98. Query to create insert and delete trigger?

```
--AFTER INSERT TRIGGER

CREATE TRIGGER TR_TBLEMPLOYEE_FORINSERT
ON DBO.TBLEMPLOYEE
AFTER INSERT
AS
BEGIN

 DECLARE @ID INT
 SELECT @ID=EMPLOYEE_ID FROM inserted
 INSERT INTO tblemployeeAudit values('New Employee with ID='+CAST(@ID AS NVARCHAR(5)) + ' IS
ADDED AT '+CAST(GETDATE() AS NVARCHAR(20)));

END

--AFTER DELETE TRIGGER

CREATE TRIGGER TR_TBLEMPLOYEE_FORDELETE
```

```
ON DBO.TBLEMPLOYEE
AFTER DELETE
AS
BEGIN

 DECLARE @ID INT
 SELECT @ID=EMPLOYEE_ID FROM deleted
 INSERT INTO tblemployeeAudit values('An Existing Employee with ID='+CAST(@ID AS
NVARCHAR(5)) + ' is deleted at '+CAST(GETDATE() AS NVARCHAR(20)));

END
```

99. What is Triggers?

TRIGGERS ARE THE SPECIAL KIND OF STORED PROCEDURES THAT EXECUTES AUTOMATICALLY IN RESPONSE TO DML EVENTS.
The After trigger for Update Event , makes use of both inserted and deleted table. The inserted table contains the updated data and the deleted table contains the old data.

100. What is instead of trigger?

Instead of Trigger , fires instead of the triggering action.
It is of 3 types. 1. Instead of Insert trigger 2. Instead of Update Trigger 3. Instead of Delete Trigger
Example – it can insert data to a table using trigger on response to a View.

101. What is table variable?

Just Like Temptables, a table variable is also created in tempdb database. The scope of a table variable is the batch, stored procedure , or statement block in which it is declared. They can be passed as parameters between procedures.

```
--Using Table Variable
declare  @tblEmolyeeCount table (DeptName nvarchar(20),deptID int, EmployeeCount int)
insert @tblEmolyeeCount
select mstDept.Name, mstDept.id, count(*) FROM tblEmployee
join mstDept on mstDept.id=tblEmployee.DeptID
group by mstDept.Name , mstDept.id

select * from @tblEmolyeeCount where EmployeeCount>1
```

102. What is derived table ?

Derived tables are available only in the context of the current query.

```
--using Derived tables
select deptname, totalemployee from
(
select Name as DeptName, DeptID, count(*) as totalEmployee from tblEmployee
join mstDept on tblEmployee.DeptID=mstDept.id
group by Name, Deptid
)as EmployeeCount  -- here EmployeeCount is the derived table
where totalEmployee>1
```

103. What is CTE ?

A CTE can be thought of as a temporary result set that is defined within the execution scope of a single SELECT, INSERT, UPDATE, DELETE or CREATE VIEW statement. A CTE is similar to a derived table in that it is not stored as an object and lasts only for the duration of the query.

```
--Using CTE   - Common Table Expression
with EmployeeCount
as
```

```
(
 select Name as DeptName, DeptID, count(*) as totalEmployee from tblEmployee
join mstDept on tblEmployee.DeptID=mstDept.id
group by Name, Deptid

)
select DeptName, TotalEmployee from EmployeeCount where totalEmployee>1
```

104. What is CTE with example ?

Common table expression(CTE) is introduced in SQL Server 2005. A CTE is a temporary result set, that can be referenced with in a SELECT, INSERT, UPDATE, DELETE statement, that immediately follows the CTE

```
with EmployeeCount(DeptID, TotalEmployees)
as
(
 select DeptID, count(*) as TotalEmployee from tblEmployee
 group by DeptID
)
select Name as DeptName, TotalEmployees from mstDept
join EmployeeCount
on mstDept.id=EmployeeCount.DeptID
order by TotalEmployees



--creating MULTIPLE CTE'S USING A SINGLE WITH CLAUSE
with EmployeeCountByType1(DeptID, TotalEmployees)
as
(
 select dept.Name as DeptName, count(*) as TotalEmployee from tblEmployee tmp
 left join mstDept dept on dept.id=tmp.DeptID
 WHERE tmp.DeptID in (1,2)
 group by dept.Name
),
EmployeeCountByType2(DeptID, TotalEmployees)
as
(

 select dept.Name as DeptName, count(*) as TotalEmployee from tblEmployee tmp
 left join mstDept dept on dept.id=tmp.DeptID
 WHERE tmp.DeptID in (3,4)
 group by dept.Name
)
select * from EmployeeCountByType1
UNION
SELECT * FROM EmployeeCountByType2
order by TotalEmployees
```

105. What is updatable CTE ?

If a CTE is created on one base table, then it is possible to update the CTE, which in turn will update the underlying base table

```
WITH  Employee_Name_Gender

as

(

 select  Employee_ID, Employee_Name,Age, Gender from tblEmployee

)

UPDATE Employee_Name_Gender SET AGE=24 WHERE EMPLOYEE_ID=1


-- updatable CTE on 2 base tables, update affecting only one base table.


With Employee_Dept_Gender

as

(

 select emp.Employee_ID, emp.Employee_Name, emp.Gender, dept.Name as DeptName from
tblEmployee emp

 left join mstDept dept on dept.id=emp.DeptID

)

update Employee_Dept_Gender set Gender='Male' where Employee_ID=1


--If a CTE is based on Multiple tables, and if the update statement affects more than 1 base table,
then the update is not allowed.

With Employee_Dept_Gender

as

(

 select emp.Employee_ID, emp.Employee_Name, emp.Gender, dept.Name  from tblEmployee emp

 left join mstDept dept on dept.id=emp.DeptID

)

update Employee_Dept_Gender set Gender='Male', Name='IT' where Employee_ID=1

--Error - View or function 'Employee_Dept_Gender' is not updatable because the modification
affects multiple base tables.
```

1. If a CTE is based on a single base table, then the update succeeds and works as expected.

2. If a CTE is based on more than one base table, and if the update affects multiple base tables, the update is not allowed and the statement terminates with an error.

3. If a CTE is based on more than one base table, and if the UPDATE affects only one base table, the Update succeeds (but not as expected always)

106. What is Recursive CTE?

Along with employee and their Manager Name, we also want to display their level in the organization.

```
WITH EmployeeCTE (EmpID, Name, ManagerID, [Level])
as
(
 select Emp_id , Employee_Name , manager_id , 1 from mstemployee where Manager_id is null
 Union All
 select mstEmployee.Emp_id, mstemployee.Employee_Name , mstemployee.Manager_id,
EmployeeCTE.[Level] +1 FROM MSTEMPLOYEE
 JOIN EmployeeCTE ON MSTEMPLOYEE.MANAGER_ID=EMPLOYEECTE.EmpID

)

select Empcte.Name as Employee, isnull(mgrcte.Name,'Super Boss') aS Manager, emPCTE.[LEVEL] from
EmployeeCTE EMPCTE
 left JOIN EmployeeCTE MGRCTE ON EMPCTE.Managerid=mgrcte.Empid
```

107. What is Normalization?

Database normalization is the process of organizing data to minimize data redundancy (data duplication), which in turn ensures data consistency.

Problems of Data Redundancy:
1. Disk Space Usage
2. Data Inconsistency
3. DML Query can become slow.

Database Normalization is a step by step process. There are 6 Normal forms, First Normal Form to Sixth Normal Form. Most Databases are in Third Normal Form(3NF). There are certain rules, that each normal form should follow.

108. What is First Normal Form (1NF)?

A table is said to be in 1NF, if
1. The Data in each column should be atomic. No multiple values, separated by commas in a column.
2. The Table does not contain any repeating column groups.
3. Identify each record uniquely using Primary Key.

Problems of Non-atomic Column : it is not possible to insert, update, delete and select just one employee.

Problems of Repeating Column Groups: More than 3 employees, table structure change required.
Less than 3 employees, wasted disk space.

109. What is Second Normal Form?

A table is said to be in 2NF, if
1. The table meets all the conditions of 1NF.
2. Move redundant data to a separate table.
3. Create relationship between these tables using foreign keys.

110. What is Third Normal Form?

A table is said to be in 3NF, if
1. The table meets all the conditions of 1NF and 2NF.
2. Does not contain columns (attributes) that are not fully dependent upon the primary key. For ex – computed column – annual salary / or dept name , dept head.

111. What is PIVOT Operator?

PIVOT is a sql server operator that can be used to turn unique values from one coumn, into multiple columns in the output, there by effectively rotating a table.

```
--USING PIVOT Operator   -- THIS QUERY DON'T GIVE PROPER OUTPUT AS EXPECTED DUE TO THE ID COLUMN IN THE TABLE..
SELECT SalesAgent, INDIA, CHINA , USA from sales
PIVOT
(
 SUM(SaleAmount)
 FOR SalesCountry in ([INDIA],[CHINA],[USA])
)
AS PIVOT_TABLE

--USING PIVOT OPERATOR – IT WILL SHOW PROPER OUTPUT
SELECT SALESAGENT, INDIA, CHINA, USA
FROM
(
 SELECT SALESAGENT, SALESCOUNTRY, SALEAMOUNT FROM SALES
)AS SOURCE
PIVOT
(
 SUM(SALEAMOUNT)
 FOR SALESCOUNTRY IN ([INDIA],[CHINA],[USA])
) AS PIVOT_TABLE
order by SALESAGENT desc
```

Output :

112. What is Error Handling in sql server ?

With the introduction of Try/Catch blocks in sql server 2005, error handling in sql server, is now similar to programming languages like c# or Java.
Error handling in SQL Server 2000 - @@ERROR
Error handling in SQL Server 2005 – Try..Catch
Note – Sometimes, System functions that begin with two at signs (@@), are called as global variables, They are not variables, and don't have the same behavior as variables , instead they are very similar to functions.
RAISERROR ("Error Message", ErrorSeverity, ErrorState) – create and returns custom errors.
Severity level – 16 (indicates general errors that can be corrected by the user)
State – Number between 1 and 255. Raiserror only generates errors with state from 1 to 127.

113. What is @@ERROR ?

it returns a Non-Zero value, if there is an error, otherwise ZERO, indicating that the previous sql statement encountered no errors.
Note - @@Error is cleared and reset on each statement execution. Check if immediately following the statement being verified, or save it to a local variable that can be checked later.

```
insert into tblData values(1,'Mahesh');   -- if it raise a error, then @@error print..
```

```
if(@@ERROR <> 0)
 print 'Error Occured'   -- Error occurred will print
else
 print 'No Error'


insert into tblData values(1,'Mahesh');
select * from tbldata    -- here No error will print..
if(@@ERROR <> 0)
 print 'Error Occured'
else
 print 'No Error'
```

114. what is Error handling using try..catch?

```
BEGIN TRY

INSERT INTO TBLDATA VALUES (1,'MAHESH');   //any state of SQL Statements

END TRY

BEGIN CATCH



 SELECT  ERROR_NUMBER() AS Error_Number,

 ERROR_MESSAGE() as Error_Message,

 ERROR_PROCEDURE() as Error_Procedure,

 ERROR_STATE() as Error_State,

 ERROR_SEVERITY() as Error_Severity,

 ERROR_LINE() as Error_Line

END CATCH
```

Any State of SQL Statements , that can possibly throw an exception are wrapped between BEGIN TRY and END TRY blocks. If there is an exception in the try Block, the control immediately jumps to the catch block. If there is no exception, CATCH block will be skipped, and the statements, after the CATCH block are executed.

Error trapped by a CATCH block are not returned to the calling Application : if any part of the error information must be returned to the application, the code in the catch block must do so by using RAISERROR Function.

In the scope of the CATCH block, there are several system functions, that are used to retrieve more information about the error that occurred. These functions return null if they are executed outside the scope of the catch block. Try / Catch cannot be used in a user-defined functions.

115. What is Transactions in SQL Server?

A transaction is a group of commands that change the data stored in a database. A Transaction is treated as a single unit. A Transaction ensures that, either all of the commands succeed, or none of them. If one of the commands in the transaction fails, all of the commands fail, and any data that was modified in the database is rolled back. In this way, transactions maintain the integrity of data in a database.

```
BEGIN TRY
 BEGIN TRANSACTION
  update mstEmployee set  Employee_Name='Mahesh kumar' where Emp_ID=1;

  update mstEmployee set  Manager_id=2 where Emp_ID=5;

 COMMIT TRANSACTION
 print 'transaction commited'
END TRY
BEGIN CATCH
 ROLLBACK TRANSACTION
 PRINT 'Trasaction Rollback'
END CATCH
```

116. What is Transactions ACID Test?

A Transaction is a group of Database Commands that are treated as a single unit. A Successful transaction must pass the "ACID" test, that is it must –Atomic, Consistent, Isolated, and Durable
Atomic: All statements in the transaction either completed successfully or they were all rolled back. The task that the set of operations represents is either accomplished or not, but in any case not left half alone.
Consistent: All data touched by the transaction is left in a logically consistent state. For Ex – if stock available numbers are decremented from tblProductTable, then there has to be a related entry in tblProductSale table. The Inventory cannot just disappear.
Isolated: The transaction must affect data without interfering with other concurrent transactions , or being interfered with them. This prevents transactions from making changes to data based on uncommitted information, for example – changes to a record that are subsequently rolled back. Most databases use locking to maintain transaction isolation.
Durable: Once a change is made, it is permanent. If a system error or power failure occurs before a set of commands is complete, those commands are undone and the data is restored to its original state once the system begins running again.

117. What is Sub Query?

A Subquery is simply a select statement, that returns a single value and can be nested inside a select, update, insert, or delete statement. It also possible to nest a subquery inside another subquery. According to MSDN, subquery can be nested upto 32 Levels.
Subqueries are always enclosed in paranthesis and are also called as inner queries, and the query containing the subquery is called as outer query. The columns from a table that is present only inside a subquery, cannot be used in the select list of the outer query.

```
--sub query with where clause
SELECT ProductId, Name FROM tblProduct
WHERE ProductId NOT IN (SELECT PRODUCTID FROM tblProductSales)

--sub query with select statment
select productid, name, (select sum(QuantitySold) from tblProductSales where
ProductId=tblProduct.ProductId) as QtySold from tblProduct
```

118. What is Correlated Subquery?

If the Subquery depends on the outer query for its value, then that sub query is called as correlated subquery.
In the where clause of the subquery below, “ProductId” column get its value from the tblproduct table that is present in the outer query.

```
--CORELATED SUBQUERY
select Name, (select sum(QuantitySold) from tblProductSales WHERE
ProductId=tblProduct.ProductId) as QtySold from tblProduct
```

So here the Subquery is dependent on the outer query for it’s value, hence this subquery is a correlated subquery.
Correlated Subquery gets executed , once for every row that is selected by the outer query.
Correlated subquery , cannot be executed independently of the outer query.

119. Which performance is better – Subqueries or Join ?

According to MSDN, in most cases, there is usually no performance difference between queries that uses sub-queries and equivalent queries using Joins.
According to MSDN, in some cases where existence must be checked, a join produces better performance. Otherwise, the nested query must be processed for each result of the outer query. In such cases, a join approach would yield better results.
In general Join works faster than subqueries, but in reality it all depends on the execution plan that is generated by SQL Server. Id does not matter how we have written the query, SQL Server will always transform it on an execution plan. If it is smart enough to generate the same plan from both queries, you will get the same result.

120. What is Cursors in SQL Server?

Relational Database Management System , including SQL Server are very good at handling data in SETS. For Example, the following "UPDATE" query, updates a set of rows that matches the condition in the "where" clause at the same time. However, if there is ever need to process the rows, on a row-by-row basis, the cursors are your choice. Cursors are very bad for performance, and should be avoided always. Most of the time , cursors can be very easily replaced using joins.
There are different type of cursors in sql server as listed below.
1. Forward Only 2. Static 3. Keyset 4. Dyanamic

```
declare @product_id int
declare @product_Name nvarchar(50)
declare @price decimal(10,2)

Declare ProductCursor  CURSOR FOR
select id, Product_Name, Price from tblProduct where id<=1000

Open ProductCursor

FETCH NEXT FROM ProductCursor INTO @product_id, @product_Name, @price

WHILE (@@FETCH_STATUS =0)
BEGIN
 PRINT @product_name
 FETCH NEXT FROM ProductCursor INTO @product_id, @product_Name, @price

END
CLOSE ProductCursor
DEALLOCATE ProductCursor
```

121. How to List of all tables in a sql server database using a query?

Object explorer with in sql server management studio can be used to get the list of tables in a specific database. However, if we have to write a query to achieve the same, there are 3 system views that we can use.

- SYSOBJECTS - SQL SERVER 2000, 2005, 2008
- SYS.TABLES - SQL SERVER 2005 , 2008
- INFORMATION_SCHEMA.TABLES - SQL SERVER 2005, 2008

```
--GET THE LIST OF TABLES ONLY
select * from SYSOBJECTS WHERE XTYPE='U'

--GET THE LIST OF TABLES ONLY
select * from sys.tables

-- to get all views
select * from sys.views

--to get all procedures
select * from sys.procedures


--GET THE LIST OF TABLES AND VIEWS
select * from INFORMATION_SCHEMA.TABLES

select * from INFORMATION_SCHEMA.VIEWS

select * from INFORMATION_SCHEMA.ROUTINES

--To get the list of different object types (XTYPE) in a database
select distinct xtype from sysobjects
```

122. What is a re-runnable SQL Script?

A Re-runnable script is a script , that when run more than once, will not throw errors.

```
CREATE TABLE tblTest
(
 id int identity(1,1) primary key not null,
 Product nvarchar(50)
)

-- --TO MAKE THIS ABOVE SCRIPT RE-RUNNABLE

use CMS
IF NOT EXISTS (select * from INFORMATION_SCHEMA.TABLES where TABLE_NAME='tblTest')
BEGIN
 CREATE TABLE tblTest
(
 id int identity(1,1) primary key not null,
 Product nvarchar(50)
)
print 'Table - tblTest has been created sucessfully'
end
else
BEGIN
```

```
 print 'Table Name - tblTest is already present in database'
end
```

To Make a SCRIPT - Re-Runnable.

- check for the existence of the table
- create the table if it does not exist.
- Else print a message stating, the table already exists.

Sql server built-in function OBJECT_ID(), can also be used to check for the existence.

```
--USING OBJECT_ID()

if OBJECT_ID('tblTest') is null
begin
 print 'TABLE HAS BEEN CREATED'
end
ELSE
BEGIN
 PRINT 'TABLE IS ALREADY PRESENT IN DATABASE'
END

ALTER TABLE tblTest
add Product nvarchar(50)

--in the above query, we will get error, as the column is already present in the table.

use CMS
if not exists (select * from information_schema.columns where COLUMN_NAME='Product' and
TABLE_NAME='tblTest' and  TABLE_SCHEMA='DBO')
begin
 ALTER TABLE tblTest
 add Product nvarchar(50)

END
else
BEGIN
 PRINT 'Column Name is already exists in the table.'
end

--using Col_Length() function
if col_length('tblTest','Product') is not null
begin
 print 'Column Name is already exists'
end
else
begin
 print 'Column Name is not present in the Table'
end
```

123. What is Optional Stored Procedures Parameters?

Paramerters of a SQL Server Stored Procedure can be made optional by specifying default values.

```
create proc spSearchProducts
@product nvarchar(50) = NULL,
@price decimal = null
```

```
as
begin
 select * from tblProduct where (product_name = @product or @product is null) and (price
=@price or @price is null)
end

EXEC spSearchProducts
EXEC spSearchProducts  'Product -3'

EXEC spSearchProducts  @price=2
```

124. What is the Use of MERGE statement in SQL Server ?

Merge Statement introduced in SQL Server 2008 allows us to perform insert , update and deletes in one statement. This means we no longer have to use multiple statements for performing insert, update and delete statement.

With MERGE Statement, we require 2 tables

1. Source Table - contains the changes that needs to be applied to the target table.

2. Target Table - The table that require changes (insert , update , delete)

Merge Statements joins the target table to the source table by using a common column in both the tables. Based on how the rows match up, we can perform insert, update, and delete on the target table.

```
MERGE TBLTARGET AS T

USING TBLSOURCE AS S

ON T.ID=S.ID

WHEN MATCHED THEN

 UPDATE SET T.PRODUCT=S.PRODUCT

WHEN NOT MATCHED  THEN

 INSERT (PRODUCT) VALUES(S.PRODUCT)

WHEN NOT MATCHED BY SOURCE THEN

DELETE;
```

125. What is a transaction ?

A Transaction is a group of commands that change the data stored in a database. A Transaction, is treated as a single unit of work.
Common Concurrency Problems:
- Dirty Reads
- Lost Updates
- Nonrepeatable Reads

• Phantom Reads
SQL Server Transaction Isolation levels:
• Read Uncommitted
• Read Committed
• Repeatable Reads
• Snapshot
• Serializable

126. What is SQL Server Dirty Read ?

A Dirty Read happens when one transaction is permitted to read data that has been modified by another transaction that has not yet been committed. In most cases, this would not cause a problem. However, if the first transaction is rolled back after the second reads the data, the second transaction has dirty data that does not exist anymore.

```
--SQL SERVER DIRTY READ EXAMPLE

BEGIN TRANSACTION
UPDATE tblProduct set Price=19 where id=1

WAITFOR delay '00:00:15'
ROLLBACK TRANSACTION


SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED
SELECT * FROM TBLPRODUCT WHERE ID=1
```

Read Uncommitted transaction isolation level is the only isolation level that has dirty read side effect. This is the least restrictive of all the isolation levels. When this transaction isolation level is set, it is possible to read uncommitted or dirty data. Another option to read dirty data is by using NOLOCK table hint. This query below is equivalent to the query in transaction 2 .

```
select * from tblProduct (NOLOCK) Where id=1
```

127. What is SQL Server Lost Update ?

Lost update problem happens when 2 transactions read and update the same data.
Both Read Uncommitted and Read Committed transaction isolation levels have the lost update side effect.
Repeatable Read , snapshot and Serializable isolation levels does not have this side effect.
The Repeatable read isolation level uses additional locking on rows that are read by the current transaction , and prevents them from being updated or deleted elsewhere. This solves the lost update problem.

128. What is Non Repeatable Read ?

Non Repeatable Read happens when one transaction reads the same data twice and another transaction updates the data in between the first and second read of transaction one.
Repeatable Read or any other higher isolation level should solve the non-repeatable read problem.
To fix the non-repeatable read problem, set transaction isolation level of transaction 1 to repeatable read. This will ensure that the data that transaction 1 has read, will be prevented from being updated or deleted elsewhere. This solves the non-repeatable read problem.

129. What is Phantom Read ?

Phantom read happens when one transaction executes a query twice and it gets a different number of rows in the result set each time. This happens when a second transaction inserts a new row that matches the where clause of the query executed by the first transaction.
To fix the phantom read problem, set transaction isolation level of transaction 1 to serializable. This will place a range lock on the rows between 1 and 3, which prevents any other transaction from inserting new rows with in that range.

130. What is Repeatable Read v/s Serializable ?

Repeatable read prevents only non-repeatable read. Repeatable read isolation level ensures that the data that one transaction has read, will be prevented from being updated or deleted by any other transaction, but it does not prevents new rows from being inserted by other transactions resulting in phantom read concurrency problem.
Serializable prevents both non-repeatable read and phantom read problems. Serializable isolation level ensures that the data that one transaction has read, will be prevented from being updated or deleted by any other transaction. It also prevents new rows from being inserted by other transactions, so this isolation level prevents both non-repeatable read and phantom read problems.

131. What is Snapshot Isolation level?

Just like serializable isolation level, snapshot isolation level does not have any concurrency side effects.
Difference between Serializable and Snapshot Isolation Level:
Serializable isolation is implemented by acquiring locks which means the resources are locked for duration of the current transaction. This isolation level does not have any concurrency side effects but at the cost of significant reduction in concurrency.
Snapshot isolation does not acquire locks, it maintains versioning in tempdb. Since, snapshot isolation does not lock resources, it can significantly increase the no. of concurrent transactions while providing the same level of data consistency as serializable isolation does.

132. What is Read Committed Snapshot isolation level in SQL Server?

Read committed snapshot isolation level is not a different isolation level. It is a different way of implementing read committed isolation level. One problem we have with read committed isolation level is that, it blocks the transaction if it is trying to read the data, that another transaction is updating at the same time.

133. What is SQL Server Deadlock ?

In a database, a deadlock occurs when two or more processes have a resource locked, and each processes requests a lock on the resource that another process has already locked. Neither of the transactions here can move forward, as each one is waiting for the other to release the lock.
When deadlock occur, sql server will choose one of the processes as the deadlock victim and rollback that processes, so other processes can move forward.

134. How SQL Server detects deadlocks?

Lock Monitor thread in sql server, runs every 5 seconds by default to detect if there are any deadlocks. If the lock monitor thread finds deadlocks, the deadlock detection interval will drop from 5 seconds to as low as 100 miliseconds depending on the frequency of deadlocks. If the lock monitor thread stops finding deadlocks, the database engine increases the interval between searches to 5 seconds.

135. What happens when a deadlock is detected?

When a deadlock is detected, the database engine ends the deadlock by choosing one of the threads as the deadlock victim. The deadlock victim's transaction is then rolled back and returns a 1205 error to the application. Rolling back the transaction of the deadlock victim releases all the locks held by the transaction. This allows the other transactions to become unblocked and move forward.

136. What is deadlock priority?

By default, SQL Server chooses a transaction as the deadlock victim that is least expensive to roll back. However, a user can specify the priority of sessions in a deadlock situation using the SET DEADLOCK_PRIORITY statement. The session with the lowest deadlock priority is choosen as the deadlock victim.
Example – SET DEADLOCK_PRIORITY NORMAL

137. What is Deadlock priority?

• The default is NORMAL.

• Can be set to LOW, NORMAL, or HIGH.
• Can also be set to a integer value in the range of -10 to 10
• LOW : -5, NORMAL : 0, HIGH : 5

138. What are deadlock victim selection criteria?

1. If the DEADLOCK_PRIORITY is different, the session with lowest priority is selected as the victim.
2. If both the sessions have the same priority, the transactions that is least expensive to rollback is selected as the victim.
3. If both the sessions have the same deadlock priority and the same cost, a victim is chosen randomly.

139. What is Logging Deadlock in SQL Server?

When deadlock occur, sql server chooses one of the transactions as the deadlock victim and rolls it back. There are several ways in SQL Server to track down the queries that are causing deadlocks. One of the options is to use SQL Server trace flag 1222 to write the deadlock information to the sql server error log.

```
--set SQL SERVER TRACE FLAG 1222
DBCC TRACEON(1222,-1)

--Check the status of the Trace Flag
DBCC TraceStatus(1222,-1)

--turn off the trace flag
DBCC traceoff(1222,-1)

--to read the error log
EXECUTE sys.sp_readerrorlog
```

140. How to find blocking queries in sql server?

Blocking occurs if there are open transactions.
DBCC Opentran will display only the oldest active transaction. it is not going to show you all open transactions.
sql server processes can be killed using
1. SQL Server Activity Monitor
2. Using SQL Command - KILL process-id
What happens when you kill a session?
All the work that the transaction has done will be rolled back. The database must be put back in the state it was in, before the transaction started.

141. What is SQL server Except Operator?

Except operator returns unique rows from the left query that are not in the right query's result.
It is introduced in SQL Server 2005. The number, and the order of the columns must be the same in both the queries. The data type must be same or compatible. This is similar to minus operator in oracle.

```
--except operator in 2 tables
select id, Name, gender  from tableA
EXCEPT
select id, Name, gender from tableB
```

You can also use Except Operator on a single table.

```
--You can also use Except Operator on a single table.
select id, Name, Gender, Salary from tblEmployee where salary >=50000
except
select id, Name, Gender, Salary from tblEmployee where salary >=60000
```

Order by clause should be used only once after the right query.

```
--using Order by Clause
select id, Name, Gender, Salary from tblEmployee where salary >=50000
except
select id, Name, Gender, Salary from tblEmployee where salary >=60000
order by salary asc
```

142. What is Difference between Except and Not in Operator?

Except operator returns all the rows from the left query that are not in the right query's result. Not in operator also does the same.

```
--except operator in 2 tables
select id, Name, gender  from tableA
EXCEPT
select id, Name, gender from tableB
```

Except filters duplicates and returns only distinct rows from the left query that are not in the right query's result, where as Not in does not filter the duplicates.

```
--You can also use Except Operator on a single table.
select id, Name, Gender, Salary from tblEmployee where salary >=50000
except
select id, Name, Gender, Salary from tblEmployee where salary >=60000
```

Except operator expects the same number of columns in both the queries, where as NOT IN, compares a single column from the outer query with a single column from the sub-query.

```
--using Order by Clause
select id, Name, Gender, Salary from tblEmployee where salary >=50000
except
select id, Name, Gender, Salary from tblEmployee where salary >=60000
order by salary asc
```

143. What is Intersect Operator in SQL Server?

Intersect operator retrieves the common records from both the left and right query of the intersect operator.

It is introduced in sql server 2005. The number and order of the columns must be same in both the queries. The data types must be same or at least compatible.

```
--USING INTERSECT OPERTOR
select ID, NAME, GENDER from tableA --left query
INTERSECT
SELECT ID, NAME, GENDER FROM tableB --right query
--USING JOin the same output
SELECT TABLEA.ID,TABLEA.NAME, TABLEA.GENDER FROM TABLEA
INNER JOIN TABLEB ON TABLEA.ID=TABLEB.ID
```

144. What is the difference between Intersect Vs Inner Join?

Intersect filters duplicates and return only distinct rows that are common between the left and right query, where as inner join does not filter the duplicates.

To make INNER Join behave like INTERSECT Operator, use the distinct operator.

Inner join treats two NULLs as two different values, so if you are joining two tables based on a nullable column and if both tables have NULL in that joining column then, INNER Join will not include those rows in the result-set, where as INTERSECT treats two nulls

145. What is the Difference between UNION INTERSECT and EXCEPT in SQL Server

Union operator returns all the unique rows from both the left and the right query. UNION ALL includes the duplicates as well.

Intersect operator retrieves the common unique rows from both the left and the right query.

Except operator returns unique rows from the left query that are not in the right query's results.

146. What is Cross Apply and Outer Apply in SQL Server?

The apply operator introduced in sql server 2005, is used to join a table to a table-valued function.

The Table valued function on the right hand side of the apply operator gets called for each row from the left (also called outer table) table.
Cross apply returns only matching rows (semantically equivalent to inner join)
Outer apply returns matching + non-matching rows(semantically equivalent to left outer join). The unmatched columns of the table valued function will be set to NULL.

```
--USING JOINGING A TABLE-VALUED FUNCTION WITH A TABLUE
SELECT D.DEPTNAME, E.Name,E.Gender, E.Salary from Department D
CROSS APPLY fn_GetEmployeeByDeptID(d.id) E --same like inner join


--USING JOINGING A TABLE-VALUED FUNCTION WITH A TABLUE
SELECT D.DEPTNAME, E.Name,E.Gender, E.Salary from Department D
OUTER APPLY fn_GetEmployeeByDeptID(d.id) E     --same like left outer join
CREATE FUNCTION fn_GetEmployeeByDeptID(@DP int)
returns table
(
 select * from mstEmployee where deptID=@dp;
)
```

147. What is DDL Trigger in SQL Server?

DDL Triggers fire in response to DDL Events - CREATE, ALTER, and DROP (Table, Function, Index, Stored Procedures)
Certain System stored procedures that perform DDL-like operations can also fire DDL Triggers. Ex- SP_RENAME system Stored procedure.

148. What is the Use of DDL Triggers?

If you want to execute some code in response to a specific DDL event. To prevent certain changes to your database schema. Audit the changes that the users are making to the database structure
DDL Trigger Syntax:

```
CREATE TRIGGER  [Trigger Name]
ON [Scope (Server | Database)]
FOR [EventType1, EventType2, EventType3]
AS
BEGIN
 --Trigger Body
END
```

DDL Triggers scope : DDL Triggers can be created in a specific database or at the server level.
DDL Trigger Example :

```
--Trigger that fires in response to a single DDL Event
Create trigger trMyFirstTrigger
ON Database
FOR Create_Table
as
begin
 print 'New table Created'
end


create trigger trMyMultipleTrigger
on database
for create_table, alter_table, drop_table
as
begin
```

```
 print 'you just created, modifed, deleted a table'
end

--disable a trigger
Disable trigger trMyFirstTrigger on database

--enable a trigger
Disable trigger trMyFirstTrigger on database

--drop a trigger
drop trigger trMyFirstTrigger on database
```

Certain System stored procedures that perform DDL-like operations can also fire DDL Triggers.

```
create trigger trRenameTrigger
on DATABASE
for rename
AS
BEGIN
 print 'You just rename something'
end
```

The trigger will be fired whenever you rename a database object using sp_rename system stored procedure.

Ex - sp_rename 'OLD_TABLE','NEW_TABLE' -- Rename table, fires the trigger

Ex - sp_rename 'OLD_TABLE.id','new_id','column '-- Rename table column, fires the trigger

149. What is Server-Scoped DDL Triggers?

```
CREATE TRIGGER TR_ServerScopeTrigger
on all server
for create_table, alter_table, drop_table
as
begin
 Print 'You have created, altered or deleted a table'
end
```

150. What is Logon Trigger in SQL Server?

As the name implies logon triggers in response to a LOGON Event, logon triggers fire after the authentication phase of logging in finishes, but before the user session is actually established.

Logon trigger can be used for

- Tracking Login Activity
- Restricting logins to SQL Server
- Limiting the number of sessions for a specific login.

```
create trigger tr_AuditLogin
on all server
for logon
as
begin
 declare @loginName nvarchar(100)
 set @loginName=ORIGINAL_LOGIN()

 if(SELECT COUNT(*) from sys.dm_exec_sessions where is_user_process=1 and
original_login_name=@loginName)>100
 BEGIN
 print 'Fourth Connection attempt by'+@loginName+' blocked'
 ROLLBACK;
```

```
 end
 ELSE
 BEGIN
  SELECT COUNT(*) from sys.dm_exec_sessions where is_user_process=1 and
original_login_name=@loginName
 END

end
```

151. What is SELECT INTO in SQL Server?

The SELECT INTO statement in SQL Server, selects data from one table and inserts it into a new table.
Select into statement in sql server can do the following
1. Copy all rows and columns from an existing table into a new table. This is extremely useful when you want to make a backup copy of the existing table.

```
--Copy all rows and columns from an existing table into a new table.
Select * into EmployeeBackUp from mstEmployee

--Copy all rows and columns from an existing table into a new table in an external database.
Select * into vdic.dbo.EmployeeBackUp from mstEmployee

--Copy only selected columns into a new table
Select Emp_ID, Employee_Name into EmployeeBackUp from mstEmployee

--Copy only selected rows into a new table
Select * into EmployeeBackUp from mstEmployee where Emp_ID=1

--Copy columns from 2 or more table into a new table
Select * into EmployeeBackUp
from mstEmployee mp
left join mstDept md on md.id=mp.Manager_id

--Copy columns from 2 or more table into a new table
Select mp.Emp_ID, mp.Employee_Name, md.Name as DeptName into EmployeeBackUp
from mstEmployee mp
left join mstDept md on md.id=mp.Manager_id

--create a new table whose columns and datatypes match with an existing table but no data will be copied
select * into EmployeeBackup from mstEmployee where 1<>1

--You can not use select into statment to select data into an existing table
select * into EmployeeBackup from mstEmployee

--Use insert into statement to select data into an existing table
insert into EmployeeBackUp
select * from mstEmployee

insert into EmployeeBackUp (Emp_ID, Employee_Name) select Emp_ID, Employee_Name from mstEmployee
```

```
--Copy all rows and columns from an existing table into a new table on a different
--sql server instance . for this , create a linked server and use the 4 part naming convention
select * into TargetTable from [SourceServer].[SourceDB].[DBO].[SourceTable]
```

152. What is Table Valued Parameters in SQL Server?

Table valued Parameter is a new feature introduced in sql server 2008. Table valued parameter allows a table to be passed as a parameter to a stored procedure from T-SQL code or from an application. Prior to SQL Server 2008,it is not possible to pass a table to a stored procedure.
Please Note – Table valued parameters must be passed as read-only to stored procedures, functions etc. This means you cannot perform DML operations like insert, Update, delete on a table-valued parameter in the body of a function, stored procedure etc.
3-steps to pass multiple rows to a stored procedure using table valued parameter.
1. Create user defined table type
2. Use the user-defined table type as a parameter in the stored procedure.
3. Declare a table variable, insert the data rows and then pass the table variable as a parameter to the stored procedure.

```
create type ttStudent as table  // table type
(
 id int primary key,
 Name nvarchar(50),
 Gender nvarchar(20)
)
 create proc spInsertEmployee  //procedure takes parameter – table type
@EmpType ttStudent readonly
as
begin
 insert into Employee
 select * from @EmpType
end

declare @EmpType ttStudent
insert into  @EmpType values(1,'Mahesh -1 ','Male');

Execute spInsertEmployee @EmpType //execute proc.
```

153. What is Grouping Sets in SQL Server?

Grouping Sets is a new feature introduced in sql server 2008.

```
--Group by Query to calculate sum of salary by Country and Gender
select Country, Gender, Sum(Salary)  as Total_Salary from tblemployee group by country, gender

union ALL
--Along with sum of salary by country and gender, we also want sum of salary just by country.
select Country, NULL, Sum(Salary)  as Total_Salary
from tblemployee
group by country

union ALL
--Along with sum of salary by country and gender, we also want sum of salary just by gender.
select NULL, GENDER, Sum(Salary)  as Total_Salary
from tblemployee
group by GENDER
```

```
UNION ALL
--Along with sum of salary by country and gender, we also want sum of salary just by nothing.
select NULL, null, Sum(Salary)  as Total_Salary
from tblemployee

--WITH GROUPING SETS Feature introduced in sql server 2008, the amount of T-SQL code that you
have to write will be greatly reduced.
select country, gender, sum(salary) as Total_Salary from tblemployee
Group by
grouping sets
(
 (Country, Gender), -- SUM of Salary by Country and Gender
 (Country), -- SUM of salary by Country
 (Gender), -- SUM of salary by Gender
 () -- Grand Total
)
ORDER BY GROUPING(COUNTRY), GROUPING(GENDER), GENDER
```

154. What is Rollup in SQL Server?

Rollup is used to do aggregate operation on multiple levels in a hierarchy

```
select Country, SUM(SALARY) as Total_Salary from tblEmployee
group by COUNTRY WITH ROLLUP
```

155. What is CUBE in SQL Server?

CUBE in sql server produces the result set by generating all combinations of columns specified in GROUP BY CUBE()

write a query to retrieve sum of salary grouped by all combinations of the 2 columns (Country & Gender) as well as grand total.

```
select Country, Gender, SUM(SALARY) AS TOTAL_SALARY FROM TBLEMPLOYEE
GROUP BY CUBE(COUNTRY, GENDER)

--OR

select Country, Gender, SUM(SALARY) AS TOTAL_SALARY FROM TBLEMPLOYEE
GROUP BY COUNTRY, GENDER WITH CUBE
```

156. What is Difference between CUBE and ROLLUP in SQL Server?

Cube generates a result set that shows aggregates from all combinations of values in the selected columns, where as rollup generates a result set that shows aggregates for a hierarchy of values in the selected columns.

You won't see any difference when you use rollup and cube on a single column.

157. What is GROUPING function in SQL Server?

Grouping (Column) indicates whether the column in a group by list is aggregated or not. Grouping returns 1 for aggregated or 0 for not aggregated in the result set.

```
--Grouping Function

select Continent, Country, City, sum(SalesAmount) as Sales_Amount,

 GROUPING(CONTINENT) AS GP_Continent,

 GROUPING(Country) AS GP_Country,
```

```
GROUPING(City) AS GP_City

from tblContinet group by Rollup(continent, country, city)
```

158. What is the use of grouping Function in real world?

When a column is aggregated in a result set , the column will have a NULL value, if you want to replace NULL with all then this GROUPING function is very handy.

```
select  (case when grouping(continent)=1 then 'ALL' else ISNULL (Continent, 'Unknown') end) as Continent,
(case when grouping(Country)=1 then 'ALL' else ISNULL (Country, 'Unknown') end) as  Country,
(case when grouping(City)=1 then 'ALL' else ISNULL (City, 'Unknown') end) as  City,
Sum(SalesAmount) as Sales_Amount
from
tblContinet
group by ROLLUP(continent, country, city)
```

159. What is Over Clause in SQL Server?

The OVER Clause combined with PARTITION BY is used to break up data into partitions. The specified function operates for each partition.

Syntax - function (...) over (partition by Col1, Col2,....)

For Example - Count(Gender) Over (Partition by Gender) will partition the data by Gender. I.e there will 2 partitions (Male and Female) and then count function is applied over each partition.

Any of the Following Functions can be used ..

Count, AVG, SUM, MIN, MAX, ROW_NUMBER(), RANK(), DENSE_RANK() etc.

```
SELECT Name, Gender, Salary,

count(gender) over (partition by gender)  as GenderTotal,

AVG(SALARY) over (partition by gender)  as AVG_Salary,

MIN(SALARY) over (partition by gender)  as Min_Salary,

MAX(SALARY) over (partition by gender)  as Max_Salary

FROM tblEmployee
```

160. What is ROW NUMBER Function in SQL Server?

- It is introduced in SQL Server 2005
- Returns the sequential number of row starting at 1

• Order by clause is required
• PARTITION BY Clause is optional
• When the data is partitioned, row number is reset to 1 when the partition changes.
• Syntax - ROW_NUMBER() over (ORDER BY Col1, Col2)

```
--ROW_NUMBER () Function
select Name, Gender, Salary, ROW_NUMBER() OVER (ORDER BY GENDER) AS ROW_NO from
tblEmployee
```

161. What is RANK and DENSE_RANK in SQL Server?

• introduced in sql server 2005
• Returns a rank starting at 1 based on the ordering of rows imposed by the ORDER BY Clause.
• ORDER BY Clause is required.
• PARTITION BY Clause is optional
• when the data is partitioned, rank is reset to 1 when the partition changes.

162. What is Difference between RANK and DENSE_RANK functions?

Rank Function skips ranking(s) if there is a tie where as DENSE_RANK will not.

For Ex - if you have 2 rows at rank 1 and you have 5 rows in total.

RANK() returns - 1,1,3,4,5

DENSE_RANK() returns - 1,1,2,3,4

Syntax - RANK() over (order by col1, col2,...)

DENSE_RANK() over (order by col1, col2, ..)

```
--using of RANK AND DENSE_RANK Function

select Name, Gender, SALARY,

RANK() over (order by salary desc) as [RANK],

DENSE_RANK() over (order by salary desc) as [Dense_RANK]

from tblemployee

--using of RANK AND DENSE_RANK Function with partition by

select Name, Gender, SALARY,

RANK() over (PARTITION BY GENDER order by salary desc) as [RANK],

DENSE_RANK() over (PARTITION BY GENDER order by salary desc) as [Dense_RANK]

from tblemployee
```

163. Find the nth highest salary?

```
WITH RESULT AS
(
```

```
 SELECT SALARY, RANK() OVER (ORDER BY SALARY DESC)  AS SALARY_RANK
 FROM tblEmployee
)
select top 1 salary from result where salary_rank=9
```

164. Find the nth highest salary with gender?

You can also use rank() and dense_rank function to find the nth highest salary among Male or Female employee groups

```
--Find the Nth Highest salary for Male or Female Employee Groups.
WITH RESULT AS
(
 SELECT GENDER, SALARY, dense_RANK() OVER (PARTITION BY GENDER ORDER BY SALARY DESC)
AS SALARY_RANK
 FROM tblEmployee
)
select top 1 salary from result where salary_rank=1 AND Gender='Male'
```

165. What is Similarities between Rank, Dense_Rank & Row_Number Functions?

• Returns an increasing integer value starting at 1 based on the ordering of rows imposed by the ORDER BY clause (if there are no ties)

• ORDER BY clause is required.

• PARTITION BY clause is optional

• When the data is partitioned, the integer value is reset to 1 when the partition changes.

```
--using RANK, DENSE_RANK AND ROW_NUMBER FUNCTION

select Name, Gender, Salary ,

ROW_NUMBER() OVER (ORDER BY SALARY DESC) AS ROW_NO,

RANK() OVER (ORDER BY SALARY DESC) AS RANK_NO,

DENSE_RANK() OVER (ORDER BY SALARY DESC) AS DENSE_RANK_NO

from tblEMployee
```

166. What is Difference between Rank, Dense_Rank & Row_Number Functions:?

• ROW_NUMBER: Returns an increasing unique number for each row starting at 1 even if there are duplicates
• RANK: Returns an increasing unique number for each row starting at 1. When there are duplicates, same rank is assigned to all the duplicate rows, but the next row after the duplicate rows will have the rank it would have been assigned if there had been no duplicates. So RANK function skips rankings if there are duplicates.
• DENSE_RANK: Returns an increasing unique number for each row starting at 1. When there are duplicates, same rank is assigned to all the duplicate rows but the DESNSE_RANK function will not

skip any ranks. This means the next row after the duplicate rows will have the next rank in the sequence.

167. What is NTILE Function in SQL Server?

• Introduced in SQL Server 2005
• ORDER BY clause is required
• PARTITION BY is optional
• Distributes the Rows into a specified no. of groups.
• if the number of rows is not divisible by no. of groups, you may have groups of two different sizes.
• Larger groups come before smaller groups.
• For Example - NTILE(2) of 10 rows, devides the rows in 2 groups (5 in each group)
• NTILE(3) of 10 rows devides the rows in 3 groups (4 in first group, 3 in 2nd group, & 3 in 3rd group)
• syntax- NTILE(Number_of_groups) over (order by Col1, col2,...)

168. What is NTILE Function without partition by clause?

```
--NTILE function without Partition By Clause
select Name, Gender, Salary,
NTILE(3) OVER (ORDER BY SALARY) AS [NTILE] from tblEmployee
```

169. What is Lead & Lag function in SQL Server 2012?

• introduced in sql server 2012
• Lead function is used to access subsequent row data along with current row data
• Lag function is used to access previous row data along with current row data
• ORDER BY clause is required.
• PARTITION BY Clause is optional.

Syntax:

```
LEAD(Column_Name, Offset,Default_Value) over (ORDER BY Col1, Col2, ...)
LAG(Column_Name, Offset,Default_Value) over (ORDER BY Col1, Col2, ...)
```

OffSet- No. of rows to lead or lag.
Default_Value -The default value to return if the number of rows to lead or lag goes beyond first row or last row in a table or partition. if default value is not specified NULL is returned.

```
--USE OF LEAD FUNCTION
SELECT NAME, GENDER, SALARY,
lead(salary) over (order by salary) as lead
FROM tblEmployee
```

170. What is FIRST VALUE function in SQL Server?

• Introduced in SQL Server 2012
• Retrieves the first value from the specified column.
• ORDER BY Clause is required.
• PARTITION BY is optional.
• syntax - FIRST_VALUE(Column_Name) OVER (Order by Col1, Col2, ...)

```
-FIRST VALUE -
select Name, Gender, Salary,
FIRST_VALUE(NAME) OVER (ORDER BY Salary) as First_Value
 from tblEmployee
```

171. What is Window function in SQL Server?

Different Categories of Window functions
• Aggregate Function – AVG, SUM, COUNT, MIN, MAX etc
• Ranking Function – RANK,DENSE_RANK, ROW_NUMBER
• Analytic Function – LEAD, LAG, FIRST_VALUE, LAST_VALUE

Over clause defines the partitioning and ordering of rows (a window) for the above functions to operate on. Hence these functions are called window functions. The OVER clause accepts the following three arguments to define a window for these functions to operate on
• ORDER BY: Defines the logical order of the rows.
• PARTITION BY: Devides the query result set into partitions. The window function is applied to each partition separately.
• Rows or Range Clause: Further limits the rows within the partition by specifying start and end points within the partition.

The Default for ROWS or RANGE clause is
RANGE BETWEEN UNBOUNDED PRECEDING AND CURRENT ROW

```
SELECT NAME, GENDER , SALARY,
avg(salary) over (order by salary ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) as AVGERAGE,
COUNT(salary) over (order by salary ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) as COUNTS,
SUM(salary) over (order by salary ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) as SUMS
 FROM Employees
```

172. What is LAST_VALUE Function in SQL Server?

• Introduced in SQL Server 2012

• Retrieves the last value from the specified column.

• ORDER BY clause is required

• PARTITION BY is optional

• ROWS and RANGE clause is optional, but for it to work correctly you may have to explicitly specify a value.

• Syntax – LAST_VALUE(Column_Name) over (order by Col1, Col2,…)

```
select Name, Gender, Salary,

LAST_VALUE(NAME) OVER (ORDER BY SALARY) AS LAST_VALUE

 from Employees
```

```
select Name, Gender, Salary,

LAST_VALUE(NAME) OVER (ORDER BY SALARY ROWS BETWEEN UNBOUNDED PRECEDING AND UNBOUNDED FOLLOWING) AS LAST_VALUE

from Employees
```

```
select Name, Gender, Salary,

LAST_VALUE(NAME) OVER (PARTITION BY ORDER BY SALARY ROWS BETWEEN UNBOUNDED
PRECEDING AND UNBOUNDED FOLLOWING) AS LAST_VALUE

from Employees
```

173. What is UNPIVOT Operator in SQL Server?

PIVOT Operator turns rows into Columns, where as UNPIVOT turns COLUMNS to ROWS.

```
--UNPIVOT EXAMPLE
SELECT SALESAGENT, COUNTRY, SALESAMOUNT
FROM tblProductSales1
UNPIVOT
(
 SalesAmount
 For Country in (India, US, UK)
) AS UNpivotExample
```

174. What is Choose Function in SQL Server?

- Introduced in SQL Server 2012
- Returns the item at the specified index from the list of available values
- The Index position starts at 1 and Not 0.
- Syntax – CHOOSE (INDEX,VAL_1, VAL_2, …)
- Example – Returns the item at the index position 2
- SELECT CHOOSE (2, ‘india’,’us’,’uk’) AS Country – it returns US

```
--INDEX START FROM 1.
SELECT CHOOSE(2,'A','B','C') -- O/P - B

SELECT
CHOOSE(DATEPART(MM,GETDATE()),'JAN','FEB','MAR','APR','MAY','JUN','JULY','AUGUST','SEPTEMB
ER','OCTOBER','NOVEMBER','DECEMBER')  -IT WILL RETURN MAY.
```

175. What is IIF Function in SQL Server?

- Introduced in sql server 2012
- Returns one of the two values, depending on whether the Boolean expression evaluates to true or false.
- IIF is a shorthand way for writing a CASE expression.
- Syntax – IIF(Boolean_Expression, true_value, false_value)

```
--IIF FUNCTION
DECLARE @GENDER INT
SET @GENDER=1
SELECT IIF(@GENDER=2,'Male','Female') as Gender  -- O/P – Female
```

176. What is TRY_PARSE Function in SQL Server?

- introduced in sql server 2012
- converts a string to date/time or Numeric Type
- Returns NULL if the provided string cannot be converted to the specified data type
- Requires .NET Framework Common Language Runtime (CLR)
- Syntax - TRY_PARSE(string_value as data_type)

Example - As the string can be converted to INT, the result will be 99.

select TRY_PARSE('99' AS INT) as Result
Example - As the string cannot be converted to INT, TRY_PARSE returns NULL.
select TRY_PARSE('ABC' AS INT)
Use case statement or IIF function to provide a meaningful error message when then conversion fails.
Example - Using case statement to provide a meaningful error message when the conversion fails.

```
--returns a meaningful error message when conversion failed
select
case when TRY_PARSE('ABC' as int) is NULL
  then 'Conversion Failed'
  else 'Conversion Sucessfull'
END as Result


--using IIF Function to return a meaningful error message
select IIF(TRY_PARSE('ABC' AS INT) IS NULL,'Conversion Failed','Conversion Sucessfull') AS RESULT
```

177. What is the difference between PARSE and TRY_PARSE?

PARSE will result an error if conversion fails, where as TRY_PARSE will return NULL instead of an error.

178. What is TRY_CONVERT Function in SQL Server?

- Introduced in sql server 2012
- Converts a value to a specified data type.
- Returns NULL if the provided value cannot be Converted to the specified data type
- If you request a conversion that is explicitly not permitted, then TRY_CONVERT fails with an error.
- Syntax - TRY_CONVERT(Data_type, value, [style])
- Style parameter is optional. The Range of acceptable values is determined by the target data_type
- Example - as the string can be converted to INT, the result will be 99.
- select TRY_CONVERT(INT,'99') AS result.

```
select TRY_CONVERT(INT,'99') AS RESULT --99

select TRY_CONVERT(INT,'ABC') AS RESULT --null

select TRY_CONVERT(XML,10) AS RESULT --error

SELECT
Case when TRY_CONVERT(int,'abc') is NULL
  then 'Conversion Failed'
  else 'Conversion sucessful'
end as Result

select IIF(TRY_CONVERT(int,'ABC') IS NULL ,'Conversion Failed','Conversion Sucessfull') AS RESULT
```

179. what is the difference between CONVERT and TRY_CONVERT?

CONVERT will result in an error if the conversion fails, where as TRY_CONVERT will return NULL instead of an error.

180. what is the difference between TRY_PARSE and TRY_CONVERT?

TRY_PARSE can only be used for converting from string to date/time or Number data types where as TRY_CONVERT can be used to any general type conversions. Try_parse relies on the presence of .net framework CLR where as TRY_CONVERT does not .

181. What is EOMONTH Function in SQL Server 2012?

• Introduced in SQL Server 2012
• returns the last day of month of the specified date.
• Syntax - EOMONTH(start_date[,month_to_add])
• start_date - the date for which to return the last day of the month.
• month_to_add - Optional. Number of months to add to the start_Date. EOMONTH adds the specified number of months to start_Date,and then returns the last day of the month for the resulting date.
• Example - Returns last day of November Month - select EOMONTH('11/20/2015') AS LastDay

SELECT EOMONTH('11/20/2014') as Last_Day - 11-31-2014

--month_to_add parameter is optional.
SELECT EOMONTH('11/20/2014',2) as Last_Day - 1-31-2015

SELECT EOMONTH('11/20/2014',-1) as Last_Day - 10-31-2014

182. What is DATEFROMPARTS Function in SQL Server 2012?

• introduced in sql server 2012
• returns a date type value for the specified year,month, and day.
• The data type of all the 3 parameters (year, month, day) is integer.
• if invalid arguments values are specified, the function returns an error.
• if any of the arguments are NULL, the function return NULL.

Syntax - DATEFROMPARTS(year, month, day)
Example - All the arguments have valid values, so DATEFROMPARTS function returns the expected date.

--RETURNS 2019-04-22
select DATEFROMPARTS(2019,4,22)

--returns datetime
select SMALLDATETIMEFROMPARTS(2019,5,18,11,33) -- 2019-05-18 11:33:00

183. What is DateTime2FromParts function in SQL Server

• introduced in sql server 2012
• Returns Datetime2
• The data type of all the parameters is integer.
• if invalid argument values are specified, the function return an error.
• if any of the required arguments are NULL , the function returns NULL.
• if the precision argument is NULL, the function returns an error.

Syntax - DATETIME2FROMPARTS(year, month, day, hour, minute, seconds, fractions, preceisions)
Example - all the function arguments have valid values . so datetime2fromparts returns datetime2 values as expected.
TIMEFROMPARTS : returns time values
syntax - TIMEFROMPARTS (hour, minute, second, fraction, precision)
select TIMEFROMPARTS(23,14,18,0,0) - 23:14:18

184. What is Offset fetch in SQL Server 2012?

SQL Server 2012 OFFSET FETCH clause makes it very easy to implement paging.
OFFSET Clause:
• introduced in sql server 2012
• returns a page of results from the result set.

• ORDER BY clause is required.

OFFSET Syntax :
select * from TABLE_NAME
ORDER BY Column_List
OFFSET row_to_skip ROWS
FETCH NEXT row_to_fetch ROWS ONLY
Example - it will leave the first 20 rows, and fetch the next 20 rows.
SELECT * FROM tblProduct order by id
OFFSET 20 ROWS
FETCH NEXT 20 ROWS ONLY

185. How to find dependencies using sql server managment studio?

Use view Dependencies option in sql server management studio to find the object dependencies.
For example - to find the dependencies on the Employee table, right click on it and select View Dependencies from the context menu.
Identifing object dependencies is important, especially when you intend to modify or delete an object upon which other objects depend. Otherwise you may risk breaking the functionality.

186. What is SEQUENCE Object in SQL Server?

• introduced in sql server 2012
• Generates sequence of numeric values in an ascending or descending order

```
--Creating a Incrementing Sequence
CREATE SEQUENCE [DBO].[SequenceObject]
as int
start with 1
increment by 1


--generating the next sequence value
SELECT NEXT VALUE FOR [SequenceObject]



--Retriving the CURRENT SEQUENCE VALUE
SELECT CURRENT_VALUE FROM SYS.SEQUENCES WHERE NAME='SequenceObject'

SELECT * FROM SYS.SEQUENCES WHERE NAME='SequenceObject'


--RESET THE SEQUENCE VALUE
ALTER SEQUENCE [SequenceObject] RESTART WITH 1


--USING SEQUENCE VALUE IN AN INSERT QUERY
insert into MSTEmployee(id) values(NEXT VALUE FOR SequenceObject)


--CREATING A DECREMENTING SEQUENCE
CREATE SEQUENCE [DecrementSequnce]
as int
start with 100
increment by -1
```

```
CREATE SEQUENCE [DecrementSequnce2]
as int
start with 100000
increment by -10
minvalue 1000
maxvalue 9900000
```

Recycling Sequence Values : When the sequence object has reached it's maximum value, and if you want to restart from the minimum value, set cycle option.

```
--recycling sequence values
ALTER SEQUENCE [SequenceObject]
increment by 10
MINVALUE 10
MAXVALUE 150
CYCLE
```

To improve performance , the sequence object values can be cached using the CACHE option.

```
ALTER SEQUENCE [SequenceObject]
increment by 10
MINVALUE 10
MAXVALUE 150
CYCLE
CACHE 10
```

187. What is a GUID in sql server?

A GUID is a 16 byte binary data that is globally unique. GUID Stands for Global Unique Identifier. The terms GUID and UNIQUEIDENTIFIER are used interchangebly.

```
declare @id UNIQUEIDENTIFIER
```

How to create a GUID?

To create a GUID in sql server, use NEWID() Function.

For ex - select NEWID() creates a GUID that is guaranteed to be unique across tables, databases, and servers. Like - ' 2F19CA2E-2CAA-457F-AFCE-5D2F16F13F6B

```
create table Customers
(
 id uniqueidentifier primary key default newid(),
 Name nvarchar(50)
)
go
insert into Customers values(default, 'Mahesh')
```

Advantages:

- A GUID is unique across table, databases, and servers.
- Useful if you are consolidating records from multiple sql servers into a single table

Disadvantages:

- Size is 16 bytes, where as INT is only 4 bytes.
- One of the largest datatypes in sql server.
- An index built on a GUID is larger and slower.
- Hard to read compared to INT

Summary: Only use a GUID when you really need a global unique identifier. in all other cases, it is better to use an int data type.

188. How to check GUID is NULL or Empty in SQL Server

Use ISNULL keywords to check if a GUID is NULL.

```
Declare @MyGuid uniqueidentifier
set @MyGuid=NEWID()
if(@MyGuid is null)
begin
 print 'GUID IS NULL'
end
else
begin
 PRINT 'GUID is not null'
end
```

What is an empty GUID?

An empty GUID is a GUID with all zeros.

```
-- TO CREATE AN EMPTY GUID
SELECT CAST(CAST(0 AS BINARY) AS UNIQUEIDENTIFIER)
--00000000-0000-0000-0000-000000000000    -- 36 characters
-or select cast(0X0 as UNIQUEIDENTIFIER)
```

189. What is dyanamic sql ?

Dyanamic SQL is a SQL built from strings at runtime.

```
DECLARE @SQL NVARCHAR(1000)
DECLARE @params nvarchar(1000)

SET @SQL='SELECT * FROM tblProduct where price=@price'
set @params='@Product_Name nvarchar(50), @Price int'
EXECUTE sys.sp_executesql @SQL, @params, @Product_Name='Product -2', @Price=1
```

To execute the Dyanamic SQL use system stored procedure sp_executesql. it takes two pre-defined parameters and any no. of user defined parameters.

@statement - This is the first parameter which is mandatory,and contains the sql statements to execute.

@params - This is the second parameter and is optional. This is used to declare parameters specified in @statement.

The rest of the parameters are the parameters that you declared in @params, and you pass them the same way as you pass parameters to a stored procedure.

190. What is Exec vs Execute?

Two options to execute dyanamic sql

- Exec/ Execute
- sp_executesql

many articles on the web says using exec over sp_executesql will have 2 problems.

- it open doors for sql injection attacks (QUOTENAME function can prevent this)
- cached query plans may not be reused and leads to poor performance (with auto-parameterization capability this may not be an issue)

What is Exec() in sql server?

Exec() or Execute() function is used to execute dyanamic sql and has only one parameter that is dyanamic sql statement you want to execute.

Summary

• if you use QUOTENAME() function, you can prevent sql injection while using Exec()

• Cached query plan reusability is also not an issue while using Exec() as sql server automatically parameterizes queries.

• It is better to use sp_executesql over exec() as we can explicitly parameterise queries instead of relying on sql server auto-parameterisation feature or QUOTENAME() function. Use Exec() only in throw away scripts rather than in production code.

191. What is QUOTENAME Function in sql server?

• it takes 2 parameter - the first is a string, and the second is a delimeter that you want sql server to use to wrap the string in
• The delimeter can be a left or right bracket ([]), a single quotation mark (,), or a double quotation mark("")
• The default for the second parameter is []

```
--quotename function
declare @sql nvarchar(max)
declare @tableName nvarchar(100)
set @tableName='INDIA tblProducts'
set @sql='select * from '+QUOTENAME(@tableName)
print @sql

--O/P - select * from [INDIA tblProducts]
```

192. What is Dyanamic SQL vs Stored Procedure in sql server?

Advantage and disadvantages of dyanamic SQL and Stored Procedure :

• Separating Database Logic from Business Logic

• Network traffic

• SQL Injection attacks

• Cached query plans reuse

• Maintainence

• Implementing Flexible logic

193. How to use Temp table in Dyanamic SQL?

• creating , inserting, selecting from temp table all happens in the dyanamic sql code block, so we are able to access the temp table without any issue.
• creating and inserting into temp table happen in the dyanamic sql code block, but selecting data is outside of the dyanamic sql code block, so we get this error - invalid object name '#test'
• This proves the point temp tables created by dyanamic sql are not accesible from the calling procedure. they are dropped when the dyanamic sql block in the stored procedure completes execution.
• Temp tables created by dyanamic SQL are not accessible from the calling procedure

• They are dropped when the dyanamic sql block in the stored procedure completes execution

194. SQL query to get organization hierarchy

195. How does a recursive CTE work

196. Delete duplicate rows in SQL

197. SQL query to find employees hired in last n months

198. Transform rows into columns in sql server

199. SQL query to find rows that contain only numerical data

200. SQL Query to find department with highest number of employees

201. Difference between inner join and left join

202. Join 3 tables in sql server

203. Real time example for right join

204. Can we join two tables without primary foreign key relation

205. Difference between blocking and deadlocking

206. Sql query to select all names that start with a given letter without like operator

207. SQL script to insert into many to many table

208. How and why a sql inner left right full and even cross join returns the same row count