Sql Server

Q. What is NewId()?

A. The following example uses NEWID() to assign a value to a variable declared as the uniqueidentifier data type. The value of the uniqueidentifier data type variable is printed before the value is tested.

```
-- Creating a local variable with DECLARE/SET syntax.
DECLARE @myiduniqueidentifier
SET @myid = NEWID()
PRINT 'Value of @myid is: '+ CONVERT(varchar(255), @myid)
```

Q. What is Scope_Identity()?

A. It returns the last IDENTITY value produced on a connection and by a statement in the same scope, regardless of the table that produced the value.
SCOPE_IDENTITY(), like @@IDENTITY, will return the last identity value created in the current session, but it will also limit it to your current scope as well. In other words, it will return the last identity value that you explicitly created, rather than any identity that was created by a trigger or a user defined function.

Q. What is difference between storeprocedures and functions?

A. Stored Procedure

- have to use EXEC or EXECUTE
- return output parameter
- can create table but won't return Table Variables
- you cannot join SP
- can be used to change server configuration
- can be used with XML FOR Clause
- can have transaction within SP

Functions

- can be used with Select statement
- Not returning output parameter but returns Table variables
- You can join UDF
- Cannot be used to change server configuration
- Cannot be used with XML FOR clause
- Cannot have transaction within function

Q. Can we call store procedure with in a function?

A. No, We cannot call store procedure with in a function.

Q. Can functions return table?

A. Yes a function can return table.
CREATE FUNCTION dbo.fnEmployeeList ()
RETURNS TABLE AS
RETURN (SELECT id, name, city FROM Employee) GO

Q. How to create temporary table in SQL?

A**.** CREATE TABLE #MyTempTable (cola INT PRIMARY KEY) INSERT INTO #MyTempTable VALUES (1) SELECT
* FROM #MyTempTable

Q. What are various types of joins?

A. **INNER JOIN:** An inner join (sometimes called a simple join) is a join of two or more tables that returns only those rows that satisfy the join condition.

1) selectbookstudent.BookId,Book.BookName,student.sname from Book **inner join**BookStudent on Book.BookId=BookStudent.BookId**inner join** student on student.sid=BookStudent.studentid

SELF JOIN: When we join a table to itself it is called Self Join.
Select p1.iPageId, p1.sPageName as Parent, p2.sPageName as Sub from PageMgmt p1 join PageMgmt p2 on p1.iPageId=p2.iParentid

OUTER JOIN: An outer join extends the result of a simple join. An outer join returns all rows that satisfy the join condition and also returns non matching rows based on Outer join type.

- **Left Outer Join:** It brings all the records from the table on left hand side and matching records from table on right hand side and return null where no match found.
Select * from table1 **left outer join** table2 on table1.Id=table2.Id

- **Right Outer Join:** It brings all the records from the table on right hand side and matching records from table on left hand side and return null where no match found. Select * from table1 **right outer join** table2 on table1.Id=table2.Id

- **Full Outer Join:** It brings all the records from both the table and return null where no match found.
Select * from table1 **full outer join** table2 on table1.Id=table2.Id

- **Cross Join:** A cross join that does not have a WHERE clause produces the Cartesian product of the tables involved in the join. The size of a Cartesian product result set is the number of rows in the first table multiplied by the number of rows in the second table.

Topics Covered:

SQL SERVER

- **Query Window**
  - The Foundation Statements of T-SQL
  - Started with SELECT Statement
  - Adding Data with the INSERT statement
  - The DELETE Statement
- **Joins**
  - INNER Joins
  - OUTER Joins
  - CROSS Joins
- **The UNION**
- **The CREATE statement**
  - CREATE Database
  - CREATE Table
- **The ALTER statement**
  - ALTER Database
  - Alter Table
- **The DROP Statement**
- **CONSTRAINTS**
  - Type of Constraints
  - Domain Constraints
  - Entity Constraints
  - Referential Integrity Constraints
  - Constraints Naming
  - Key Constraints
  - PRIMARY KEY Constraints
  - FOREIGN KET Constraints
  - UNIQUE Constraints
  - CHECK Constraints
  - DEFUALT Constraints
  - Disabling Constraints
- **Adding more to our Queries**
  - What is sub query?
  - Building a nested sub query
  - Correlated sub queries
  - How correlated sub query work
  - Correlated sub queries in the WHERE Clause
  - Dealing with NULL Data- the ISNULL function derived table

- The EXISTS Operator
- Using EXISTS in other ways
- Mixing data type: CAST and CONVERT

➢ **Understanding indexers**

- How data is accessed in Sql Server
- Creating, Altering and Dropping Indexes
- The CREATE INDEX Statement

➢ **Views**

- Simple Views
- Views as filters
- More complex views
- Editing Views with T-SQL
- Dropping Views

➢ Stored Procedures

- Basic Syntax
- Changing Stored Procedure with ALTER
- Dropping Sprocs
- Handling Errors

➢ **Triggers**

- What is a Trigger?
- With ENCYPTION 0
- The FOR/AFTER vs the INSTEAD of Clause
- With APPEND
- Not for replication
- Using Triggers for data integrity rules
- Dealing with requirements soured from other table
- Using Trigger to check the Delta of an Update
- Using Trigger for custom error message
- Other common uses for Trigger
- Other trigger issues
- Trigger can be nested
- Triggers can be recursive
- Dropping Triggers

WinNsoft
HAVE AN EXTRA EDGE

Crystal Report

- Crystal Report Overview
- Getting started with crystal report
- Creating basic report
- Working with sub report
- Integration with windows application
- Customizing appearance and layout of report viewer
- Creating XML report web services
- Working with .Net data
- Adding a database or table to a report
- Working with ADO.Net
- Formulas and logic
- Report formatting
- Integration with web application
- Working with Crystal Report Engine