**SQL Server Interview Q & A**

**Deterministic vs Nondeterministic functions:**

Deterministic functions always return the same result any time they are called with a specific set of input values and given the same state of the database. Nondeterministic functions 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.

For example, the function AVG always returns the same result given the qualifications stated above, but the GETDATE function, which returns the current datetime value, always returns a different result.

All of the string built-in functions are deterministic.

For Ex.

AVG, ABS, DAY, DATEDIFF, ISNULL etc. are deterministic.

GETDATE, **Scalar functions** with information about current configuration option settings: @@OPTIONS, **@@SERVERNAME**, @@VERSION etc. are deterministic.

**SQL Server Basic Tools:**

**SQL Server Management Studio** is the principal tool for administering the Database Engine and writing Transact-SQL code. It is hosted in the Visual Studio shell. It is not included in SQL Server Express but is available as a separate download.

**SQL Server Configuration Manager** installs with both SQL Server and the client tools. It lets you enable server protocols, configure protocol options such as TCP ports, configure server services to start automatically, and configure client computers to connect in your preferred manner.

**While connecting to SQL Server Management Studio:**

If you are connecting to SQL Server 2005 on Windows Vista or Windows Server 2008 (or more recent), you may need to right-click Management Studio and then click **Run as Administrator** in order to connect using your Administrator credentials. Starting in SQL Server 2008, setup adds selected logins to SQL Server, so your Administrator credentials are not necessary.

For the **default** **instance** of SQL Server, the server name is the computer name. For a **named instance** of SQL Server, the server name is the *<computer\_name>, <port>***\***<instance\_name>.*

**Database Engine of SQL Server:**

The **Database Engine component** of SQL Server is the core service for storing, processing, and securing data. The Database Engine provides **controlled** access and rapid transaction processing to meet the requirements of the most demanding data consuming applications in your enterprise.

**SET Options on Stored Proc:**

**SET ANSI\_NULLS ON –** recommended (default)

The ANSI\_NULLS option specifies that how SQL Server handles the comparison operations with NULL values.

When it is set to **ON** any comparison with **NULL using = and <> will yield to false value**. And it is the ISO defined standard behavior. So to do the comparison with NULL values we need to use IS NULL and IS NOT NULL.

When it is set to **OFF** any comparison with **NULL using = and <> will work as usual i.e. NULL = NULL returns true and 1= NULL returns false.**

SET ANSI\_NULLS ON

IF NULL = NULL

PRINT 'same'

ELSE

PRINT 'different'

--result: different

SET ANSI\_NULLS OFF

IF NULL = NULL

PRINT 'same'

ELSE

PRINT 'different'

**SET QUOTED\_IDENTIFIER ON** – recommended (default)

It specifies how SQL Server treats the data that is defined in Single Quotes and Double Quotes.

When it is set to **ON** any character set that is defined in the **double quotes** “”is **treated as a T-SQL Identifier** (Table Name, Proc Name, Column Name….etc)

When any character set that is defined in the **single quotes ‘’** is **treated as a literal.**

SET QUOTED\_IDENTIFIER ON

CREATE TABLE "SELECT" ("TABLE" int) -- SUCCESS

GO

SET QUOTED\_IDENTIFIER ON

SELECT "sometext" AS Value -- FAIL because “sometext” is not a literal

When it is set to **OFF** any character set that is defined either in Single **Quotes or in Double Quotes** is treated as a **literal**.

SET QUOTED\_IDENTIFIER OFF

CREATE TABLE "SELECT"(“TABLE” int) -- FAIL

GO

SET QUOTED\_IDENTIFIER OFF

SELECT "sometext" AS Value -- SUCCESS as “sometext” is treated litral

**SET NOCOUNT ON** – recommended

When SET NOCOUNT is ON, the count (indicating the number of rows affected by a Transact-SQL statement) is not returned. When SET NOCOUNT is OFF, the count is returned. It is used with any SELECT, INSERT, UPDATE, DELETE statement.

SET NOCOUNT ON improves stored procedure (SP) performance.

Added to prevent extra result sets from interfering with SELECT statements.

**@@Error:**

Returns the error number for the last Transact-SQL statement executed.

Returns 0 if the previous Transact-SQL statement encountered no errors.

Returns an error number if the previous statement encountered an error. If the error was one of the errors in the sys.messages catalog view, then @@ERROR contains the value from the sys.messages.message\_id column for that error. You can view the text associated with an @@ERROR error number in sys.messages.

Because @@ERROR is cleared and reset on each statement executed, check it immediately following the statement being verified, or save it to a local variable that can be checked later.