1. What is the difference between the following objects in SQL Server
2. batch, script and transaction

|  |  |
| --- | --- |
| Batch | Is a series of one or more statements submitted and executed at the same time  Notes:-   * Create default, create function, create procedure, create rule, create trigger, and create view why they must be in single batch * You cannot bind a rule or default to a column and insert values into that column in the same batch * If there is error in some queries. the correct queries will be Executed normally , but wrong will not be executed * Example:-   Insert into ----  Update ----  Delete---  These will run statement by statement |
| Script | script is a set of SQL commands. A SQL script can one or more  batches separate by GO   * If there are error in some queries .the following queries will not be executed   Example:-  Create rule r AS @range>=$1000  Go  Sp\_bindrule r ,’student.st\_id’ |
| Transaction | A transaction is a set of operations performed so all operations are guaranteed to succeed or fail as one unit   * Type of transaction:   **Explicit Transactions** Explicit transactions are those in which you explicitly control when the transaction begins and when it ends. Prior to SQL Server 2000, explicit transactions were also called user-defined or user-specified transactions. T-SQL scripts for this mode use the BEGIN TRANSACTION, COMMIT TRANSACTION, and ROLLBACK TRANSACTION statements. Explicit transaction mode lasts only for the duration of the transaction. When the transaction ends, the connection returns to the transaction mode it was in before the explicit transaction was started.  **Implicit Transactions** the default transaction mode is implicit. Each statement starts its own transaction and is automatically committed. The benefit of an implicit transaction is that you do not need to specify START TRANSACTION, COMMIT or ROLLBACK statements. |

1. trigger and stored procedure

|  |  |
| --- | --- |
| **Trigger** | **stored procedure** |
| * SQL Server triggers are special [stored procedures](http://www.sqlservertutorial.net/sql-server-stored-procedures/) that are executed automatically * three type of triggers: * Data manipulation language (DML) triggers which are invoked automatically in response to [INSERT](http://www.sqlservertutorial.net/sql-server-basics/sql-server-insert/), [UPDATE](http://www.sqlservertutorial.net/sql-server-basics/sql-server-update/), and [DELETE](http://www.sqlservertutorial.net/sql-server-basics/sql-server-delete/) events against tables. * Data definition language (DDL) triggers which fire in response to [CREATE](http://www.sqlservertutorial.net/sql-server-basics/sql-server-create-table/), ALTER, and [DROP](http://www.sqlservertutorial.net/sql-server-basics/sql-server-drop-table/) statements. [DDL triggers](http://www.sqlservertutorial.net/sql-server-triggers/sql-server-ddl-trigger/) also fire in response to some system stored procedures that perform DDL-like operations. * Logon triggers which fire in response to LOGON events | * A stored procedure in SQL is a type of code in SQL that can be stored for later use and can be used many times. So, whenever you need to execute the query, instead of calling it you can just call the stored procedure. * We can write inside sp any   Sql statments |
| * we cannot directly call trigger | * We can call a stored procedure |
| * we can’t schedule a trigger. | * Stored procedures can be scheduled through a job to execute on a predefined time |
| * we can’t pass parameters as input to a trigger. | * Stored procedure can take input parameters, |
| * a trigger cannot return value | * Stored procedures can return value |
| * we can’t use print commands inside a trigger. | * We can use Print commands inside a stored procedure |

1. stored procedure and functions

|  |  |
| --- | --- |
| **stored procedure** | **Functions** |
| Stored Procedure it is optional return value or not.  Returned value must be integer. | Function must return a value.  Returned value can by any datatype. |
| Procedures can have input/output  parameters | Functions can have only input parameters |
| Procedure can call function inside it | Function can not called procedure |
| Procedure allows SELECT as well as DML(INSERT/UPDATE/DELETE) statement in it | 1. Function allows only SELECT statement in it. |
| We can not called stored procedure by select statement | 1. We can call function by 2. Select statement |
| Stored Procedures cannot be used in the SQL statements anywhere in the WHERE/H**AVIN**G/SELECT section | 1. Function can be. |
| The most important feature of stored procedures over function is to retention and reuse the execution plan | 1. function it will be compiled every time |
| Exception can be handled by try-catch | 1. try-catch block cannot be used in a Function. |

1. drop, truncate and delete statement

|  |  |  |
| --- | --- | --- |
| **Drop** | **Truncate** | **Delete** |
| DDL commands | TRUNCATE is a DDL command | DELETE is a DML command. |
|  | We cannot use WHERE clause with TRUNCATE. | We can use where clause with DELETE to filter |
| he DROP command removes a table from the database  (structure) | TRUNCATE removes all rows from a table | The DELETE command is used to remove rows from a table based on WHERE condition |
| All the tables' rows, indexes and privileges will also be removed | Identify column is reset to its seed value if table contains any identity column | Identity of column keep DELETE retains the identity. |
|  | To use Truncate on a table you need at least ALTER permission on the table. | To use Delete you need DELETE permission on the table |
|  | Minimal logging in transaction log | It maintain the log, |
|  | TRUNCATE is faster than DELETE. | it slower than TRUNCATE |
| The operation cannot be rolled back. | Can not get deleted data again | Can get |

1. select and select into statement

|  |  |
| --- | --- |
| select into | select |
| To copy the whole data from one table into another table using a single command.  Example :-  SELECT \* INTO Table1 from table2;  SELECT INTO statement could be used even if the target table doesn’t exist as it creates the target table if it doesn’t exist. | To represent columns from table  Or assign column to variable |

1. local and global variables

|  |  |
| --- | --- |
| Local | Global |
| See in it scope (function-procedure-view-batch-..etc)  Can declare local variable and assign value in it    Example :  Declare @n int  Set @n=0 | See in all place in sql  Can not declare global variable  There are many build in global variable  Example  @@identity  @@error |

1. convert and cast statements

|  |  |
| --- | --- |
| **Convert** | **Cast** |
| convert data from one type to another. | convert data from one type to another. |
| it accepts an optional style parameter which is used for formatting.  Like:-  CONVERT(VARCHAR,GETDATE(),101) | Not accept |

1. DDL,DML,DCL,DQL and TCL

|  |  |
| --- | --- |
| **DML** | **Data Manipulation Language**. It is used to retrieve, store, modify, delete, insert and update data in database.  Examples: SELECT, UPDATE, INSERT statements |
| **DDL** | **Data Definition Language.** It is used to create and modify the structure of database objects in database.  Examples: CREATE, ALTER, DROP statements |
| **DCL** | **Data Control Language**. It is used to create roles, permissions, and referential integrity as well it is used to control access to database by securing it.  Examples: GRANT, REVOKE statements |
| **TCL** | **Transactional Control Language**. It is used to manage different transactions occurring within a database.  Examples: COMMIT, ROLLBACK statements |

1. For xml raw and for xml auto
2. Table valued and multi statemcent function

|  |  |
| --- | --- |
| **Table valued** | **multi statemcent function** |
| returns a table of already existed table or more and have only one select statment inside | Multistatment: returns a variable as a table created and filled inside the function |

1. Varchar(50) and varchar(max)

|  |  |
| --- | --- |
| **Varchar(50)** | **varchar(max)** |
| Varchar(n) -Max length is fixed and that will be n | Indicate maximum length of characters , maximum of 8,000 characters. |

1. Datetime(3), datetime2(7) and datetimeoffset(7)

|  |  |  |
| --- | --- | --- |
| Datetime | **datetime2** | datetimeoffset |
| **datetime**  Defines a date that is combined with a time of day with fractional seconds that is based on a 24-hour clock. | **datetime2** [ (*fractional seconds precision*) ]  Defines a date that is combined with a time of day that is based on 24-hour clock. **datetime2** can be considered as an extension of the existing **datetime** type that has a larger date range, a larger default fractional precision, and optional user-specified precision.  YYYY-MM-DD hh:mm:ss[.fractional seconds] | ***datetimeoffset****[ (*fractional seconds precision*) ]*  Defines a date that is combined with a time of a day that has time zone awareness and is based on a 24-hour clock.  YYYY-MM-DD hh:mm:ss[.nnnnnnn] [+|-]hh:mm |

1. Default instance and named instance

you can install only one default instance and multiple named instance

1. SQL and windows Authentication

|  |  |
| --- | --- |
| windows Authentication | SQL |
| you can connect to server without username and password | you need a user name and password to login to sql server |

1. Clustered and non-clustered index

|  |  |
| --- | --- |
| Clustered | non-clustered index |
| clustered :used to improve performance of retrieval of data there is only one clustered index on table | while you can have more than one non clustered index |

1. Group by rollup and group by cube

|  |  |
| --- | --- |
| Group by rollup | group by cube |
| You won't see any difference since you're only rolling up a single column. Consider an example where we do  ROLLUP (YEAR, MONTH, DAY)  With a ROLLUP, it will have the following outputs:  YEAR, MONTH, DAY  YEAR, MONTH  YEAR  () | With CUBE, it will have the following:  YEAR, MONTH, DAY  YEAR, MONTH  YEAR, DAY  YEAR  MONTH, DAY  MONTH  DAY  ()CUBE essentially contains every possible rollup scenario for each node whereas ROLLUP will keep the hierarchy in tact (so it won't skip MONTH and show YEAR/DAY, whereas CUBE will. |

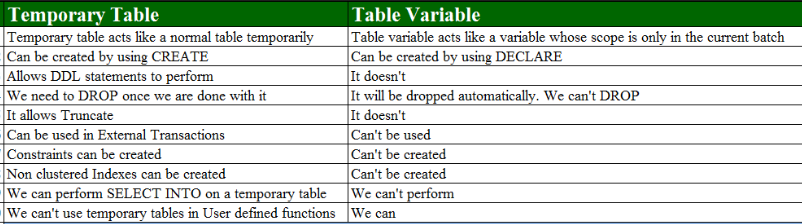
1. Sequence object and identity

|  |  |
| --- | --- |
| A sequence is created independently of the tables by using the CREATE SEQUENCE statement. Options enable you to control the increment, maximum and minimum values, starting point  Can be cycle  Shared between many table | is created dependently of the tables  belong to only one table |

1. Inline function and view

function can take parameters while view can not.

1. Table variable and temporary table



1. Row\_number() and dense\_Rank() function

|  |  |
| --- | --- |
| ROW\_NUMBER function simply assigns a new row number to each record irrespective of its value. | Dense rank function assign similar rows the same rank number |