**SQLServer**

**Lab**

1. Create a cursor for Employee table that increases Employee salary by 10% if Salary <3000 and increases it by 20% if Salary >=3000. Use company DB
2. Display Department name with its manager name using cursor. Use ITI DB
3. Try to display all students first name in one cell separated by comma. Using Cursor
4. Create full, differential Backup for SD DB.
5. Use import export wizard to display students data (ITI DB) in excel sheet
6. Try to generate script from DB ITI that describes all tables and views in this DB
7. Create a sequence object that allow values from 1 to 10 without cycling in a specific column and test it.

Part2: What is the difference between the following objects in SQL Server

1. batch, script and transaction

* Transaction: A transaction is a single unit of work. If a transaction is successful, all of the data modifications made during the transaction are committed and become a permanent part of the database. If a transaction encounters errors and must be canceled or rolled back, then all of the data modifications are erased.
* Batch: A batch of Transact-SQL statements contains two or more statements, separated by a semicolon (;), built into a single string passed to SQLExecDirect or SQLPrepare Function.

1. trigger and stored procedure

| Triggers | Procedures |
| --- | --- |
| A Trigger is implicitly invoked whenever any event such as INSERT, DELETE, UPDATE occurs in a TABLE. | A Procedure is explicitly called by user/application using statements or commands such as exec, EXECUTE, or simply procedure\_name |
| Only nesting of triggers can be achieved in a table. We cannot define/call a trigger inside another trigger. | We can define/call procedures inside another procedure. |
| In a database, syntax to define a trigger: CREATE TRIGGER TRIGGER\_NAME | In a database, syntax to define a procedure: CREATE PROCEDURE PROCEDURE\_NAME |
| Transaction statements such as COMMIT, ROLLBACK, SAVEPOINT are not allowed in triggers. | All transaction statements such as COMMIT, ROLLBACK are allowed in procedures. |
| Triggers are used to maintain referencial integrity by keeping a record of activities performed on the table. | Procedures are used to perform tasks defined or specified by the users. |
| We cannot return values in a trigger. Also, as an input, we cannot pass values as a parameter. | We can return 0 to n values. However, we can pass values as parameters. |

1. stored procedure and functions

* The function must return a value but in Stored Procedure it is optional. Even a procedure can return zero or n values.
* Functions can have only input parameters for it whereas Procedures can have input or output parameters.
* Functions can be called from Procedure whereas Procedures cannot be called from a Function.

1. drop, truncate and delete statement

* Delete: basically, it is a Data Manipulation Language Command (DML). It is used to delete one or more tuples of a table. With the help of the “DELETE” command, we can either delete all the rows in one go or can delete rows one by one. i.e., we can use it as per the requirement or the condition using the Where clause. It is comparatively slower than the TRUNCATE command. The TRUNCATE command does not remove the structure of the table.
* Drop: it is a Data Definition Language Command (DDL). It is used to drop the whole table. With the help of the “DROP” command we can drop (delete) the whole structure in one go i.e. it removes the named elements of the schema. By using this command the existence of the whole table is finished or say lost.
* Truncate: it is also a Data Definition Language Command (DDL). It is used to delete all the rows of a relation (table) in one go. With the help of the “TRUNCATE” command, we can’t delete the single row as here WHERE clause is not used. By using this command the existence of all the rows of the table is lost. It is comparatively faster than the delete command as it deletes all the rows fastly.

1. select and select into statement

* select display the result of the query
* select into creates new table with the with the result of the query

1. local and global variables

* Global variables are pre-defined system functions. Their names begin with an @@ prefix. The server maintains the values in these variables. Global variables return various pieces of information about the current user environment for SQL Server. Global Variable are automatically updated and interact with the system.
* Local variables are user defined and start with @ prefix

1. convert and cast statements

* CAST and CONVERT are two SQL functions used by programmers to convert one data type to another.
* The CAST function is ANSI standard and is compatible to use in other databases while the CONVERT function is a specific function of the SQL server.
* Since the CAST function is compatible with other databases, it is also described as portable though it has fewer features compared to the CONVERT function. The CONVERT function, meanwhile, can do some things that the CAST function cannot.
* The CAST function is used to convert a data type without a specific format. The CONVERT function does converting and formatting data types at the same time.
* In terms of syntax, both functions have the optional parameter of length. In the CONVERT function, there is an additional parameter called style which specifies the format of the data type after conversion.
* The CAST function is often used to preserve decimal values and places while converting them into integers. [The function can also truncate the decimal](http://www.differencebetween.net/technology/software-technology/difference-between-truncate-and-delete/) value if needed. The CONVERT function cannot perform this task.

1. DDL,DML,DCL,DQL and TCL

* Data Definition Language actually consists of the SQL commands that can be used to define the database schema. It simply deals with descriptions of the database schema and is used to create and modify the structure of database objects in the database.DDL is a set of SQL commands used to create, modify, and delete database structures but not data. These commands are normally not used by a general user, who should be accessing the database via an application.
* Data Query Language statements are used for performing queries on the data within schema objects. The purpose of the DQL Command is to get some schema relation based on the query passed to it. We can define DQL as follows it is a component of SQL statement that allows getting data from the database and imposing order upon it. It includes the SELECT statement. This command allows getting the data out of the database to perform operations with it. When a SELECT is fired against a table or tables the result is compiled into a further temporary table, which is displayed or perhaps received by the program i.e. a front-end.
* Data Manipulation Language commands that deals with the manipulation of data present in the database belong to DML or Data Manipulation Language and this includes most of the SQL statements. It is the component of the SQL statement that controls access to data and to the database. Basically, DCL statements are grouped with DML statements.
* Data Control Language includes commands such as GRANT and REVOKE which mainly deal with the rights, permissions, and other controls of the database system.
* Transaction Control Language commands deal with the transaction within the database. Such as commit, rollback, save point, set transaction.

1. For xml raw and for xml auto

* The RAW mode generates a single XML element for each row in the result set returned by the query.
* The AUTO mode generates the XML by using heuristics based on how the SELECT statement is defined.

1. Table valued and multi statement function

|  | **ITVF** [inline table-valued function](https://database.guide/introduction-to-inline-table-valued-functions-itvf-in-sql-server/) | **MSTVF**  [**multi-statement table-valued function**](https://database.guide/introduction-to-multi-statement-table-valued-functions-mstvf-in-sql-server/) |
| --- | --- | --- |
| **The RETURNS Syntax** | You simply state RETURNS TABLE and the return table’s definition will be based on the function’s SELECT statement. No need to specify the structure of the return table. | Your RETURNS syntax explicitly specifies the structure of the return table. This is done by declaring a TABLE variable that will be used to store and accumulate the rows that are returned as the value of the function. |
| **The BEGIN/END Syntax** | ITVFs do not use the BEGIN/END syntax. | MSTVFs do use the BEGIN/END syntax. |
| **Performance** | Generally faster than MTSVFs. | Generally slower than ITVFs. |
| **Data Updates** | In some cases it’s possible to update data in the underlying tables using an ITFV. | You cannot update data in the underlying tables using a MSTVF. |

1. Varchar(50) and varchar(max)

The maximum number of characters for this data type can hold unlimited by defining as MAX indicates that the maximum storage size is (2147483647) 2^31-1 bytes (2 GB). The storage size is the actual length of the data entered + 2 bytes.

Varchar(50) is 50 bytes

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

The datetime2 is an expansion of the existing DateTime type with a longer date range, higher default fractional precision, and a new feature of user-specified precision.

 the datetimeoffset value includes the time zone offset and the datetime2 value doesn’t.

1. Default instance and named instance

there can be only one default instance in SQL Server while there can be multiple named instances in the SQL Server.

1. SQL and windows Authentication

* Windows Authentication uses AD ( Active Directory) to manage user account and passwords. The account can be part of an AD group. SQL Server uses AD to validate the account is active and then checks what permissions that account has in the SQL Server.
* SQL Server Authentication manages the created account and password. This information is stored in the Master Database.

1. Clustered and non-clustered index

| CLUSTERED INDEX | NON-CLUSTERED INDEX |
| --- | --- |
| Clustered index is faster. | Non-clustered index is slower. |
| Clustered index requires less memory for operations. | Non-Clustered index requires more memory for operations. |
| In clustered index, index is the main data. | In Non-Clustered index, index is the copy of data. |
| A table can have only one clustered index. | A table can have multiple non-clustered index. |
| Clustered index has inherent ability of storing data on the disk. | Non-Clustered index does not have inherent ability of storing data on the disk. |
| Clustered index store pointers to block not data. | Non-Clustered index store both value and a pointer to actual row that holds data. |
| In Clustered index leaf nodes are actual data itself. | In Non-Clustered index leaf nodes are not the actual data itself rather they only contains included columns. |
| In Clustered index, Clustered key defines order of data within table. | In Non-Clustered index, index key defines order of data within index. |
| A Clustered index is a type of index in which table records are physically reordered to match the index. | A Non-Clustered index is a special type of index in which logical order of index does not match physical stored order of the rows on disk. |

1. Group by rollup and group by cube

ROLLUP operator generates aggregated results for the selected columns in a hierarchical way. On the other hand, CUBE generates a aggregated result that contains all the possible combinations for the selected columns.

1. Sequence object and identity

The Identity property is a column property meaning it is tied to the table, whereas the sequence is a user-defined database object and it is not tied to any specific table meaning its value can be shared by multiple tables.

1. Inline function and view

* View can be materialized (indexed view) and hence performs better. But Inline Table Valued functions cannot be indexed and performance decreases when number of rows increases.
* Views can have triggers since they can be used to change underlying tables (INSTEAD OF triggers) but not Inline Table Valued functions.

1. Table variable and temporary table

* Temporary Tables are physically created in the tempdb database. These tables act as the normal table and also can have constraints, index like normal tables.
* Table Variable acts like a variable and exists for a particular batch of query execution. It gets dropped once it comes out of batch. It is created in the memory database but may be pushed out to tempdb.

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

* Dense\_Rank() Function is similar to Rank with only difference, this will not leave gaps between groups.