# SQL Server

Wednesday, February 10, 2021 10:09 AM

**Database**

| | | |
|---|---|---|
| 1. | Modify Name | Alter database <old_name> modify name=<new_Name> **or** exec sp_rename oldname,newname |
| 2. | To Get Data Dictionary | In Query Area Select Table Name and press alt + F1 / sp_help |
| 3. | To Allow Identity Column to supply value manually | SET Identity_Insert tablename ON;<br><br>Note : Have to Supply Values like regular Insertion |
| 4. | Check | Add constraint <nameofc> default(value) for column_name |
| 5. | Default | Add constraint <nameofc> CHECK(expression) |
| 6. | Options For Foreign Key Column | Set Null, Set Default, Cascade. |
| 7. | To Reset Identity | DBCC CHECKIDENT(tblName,RESEED,seed value); |
| 8. | To Get Current Identity Value | • Scope_Indentity() : Of Particular Scope in Same Scope<br>• @@identity () : Same Session but Across Every Scope<br>• Ident_Current('tableName') : OF Particular Table.<br>• Generally All With Select To Display Value |
| 9. | To Get Distinct Value form Column | Select distinct column Name/s from tableName |
| 10. | To Provide Condition (Filter or short) | Select columns from tableName where condition<br>• For Not Equal to <> **or** !=<br>• Other Like =, >=, <=, =, >, <<br>• **IN** : where column IN (Lists of Elements) - True for all Value Form List<br>• **NOT :** where column NOT IN (Lists of Elements) - True for all apart from Value Form List **or** satisfy condition like NOT Between and NOT LIKE<br>• **BETWEEN :**where column between value1 and value2 - for Ranging<br>• **LIKE:** where column LIKE 'pattern'<br>○ % : Like * in REGEX means zero or more occurrence<br>○ _: Like . In Regex means only one char<br>○ [] : Like [] in regex any form this<br>○ [^]: Like [^] in Regex not any from this<br>• **AND & OR :** For Logical<br>• **IS NULL or IS NOT NULL : Compare Null Values** |
| 11. | For Sorting | Select * from column_name order by (column_name ASC /DESC)*<br>• Default ASC |
| 12. | To Retrieve few records of choice | Select top number . . . |
| 13. | To Aggregate | Select . . . . tableName group by columnName/es (Optional: Having=value)<br>• Can Select or Operate Aggregate Functions Only or columnName/es By Which Aggregate.<br>• Using Having We Can Get Particular Values only like Gender is Male<br>• If Condition is there can use Where Having is Helpful to filter based on Value |
| 14. | Types of Join | • Inner Join : Common of A and B<br>• Left Join : Complete A + Null Where Not Matched With B<br>• Right Join : Null Where Not Matched With A+ Complete B<br>• Full Join : Complete A and B (Null Where Not Matched in A or B)<br>• Cross Join : A * B = Each of A is Join With Each of B<br>• Self Join: It is not any different type of join just a concept that if needed then can join same with itself |
| 15. | To Replace Null Value | • ISNULL() function<br>• Case Statement |

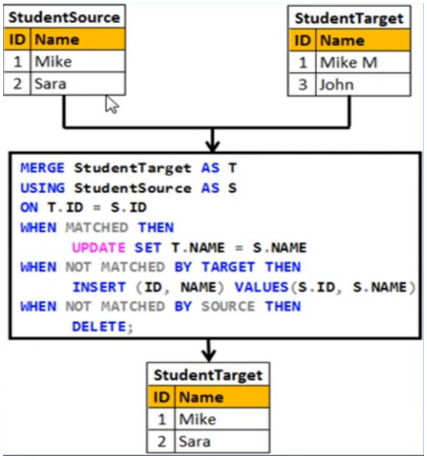
| | | |
|---|---|---|
| | | • Case when exp then value else value as ColnameToReprsent<br>• COALESCE() function<br>• Generally It Returns First No Null Values from Provided List of values |
| 16. | To Combine the resultant data (Same Structure including data type) | • UNION<br>• Remove Duplication after union data and before displaying<br>• UNION ALL<br>• Gather all data and Display. |
| 17. | Stored Procedure | • With Encryption before definition of stored procedure to stop retrieving definition.<br>• sp_helptext to retrieve definition of stored procedure |
| 18. | Basic String Functions | • **ASCII(char value) :** Gives ASCII Value of character<br>• **Char (int value) :** Character corsponde the ascii value<br>• **LTRIM(string), RTRIM(string), TRIM(string) :** To Trim The Spaces for Left and Right side<br>• **LEN(string) :** To get Length of String<br>• **LEFT/RIGHT(string, int):** To Get Specified Char from string from left/right side<br>• **SubString(string, int, int) :** Substring from string<br>• **Replicate(string, int times) :** To Repeat String times<br>• **Spcae(int) :** To Print Space int times<br>• **Lower(string) :** String To Lower<br>• **Upper (string) :** String To Upper<br>• **Reverse(string):** To Reverse the String<br>• **CharIndex(char, string) :** Returns First Occurrence of char in string, 0 if no<br>• **PatIndex (pattern, string) :** Gives Index where First Occurrence of Pattern is matched, 0 if no<br>• **Replace (string, stringToReplace, ReplacementString) :** To Replace Value<br>• **Stuff (string, startpos, length,replacement string) :** Similar to Replace just way is different |
| 19. | Date Time | • **Data Types:**<br>• Time - 3 to 5<br>• Date -3<br>• SmallDateTime - 4<br>• DateTime - 8<br>• DateTime2 - 6 to 8<br>• DateTimeoffset 8 to 10 ( Includes Time Zone +/- hh:mm )<br>• Difference is in Accuracy and size of storing 1day/ minute to, 0.0033s - 100 ns<br>• **Functions**:<br>1. GETDATE / URRENT TIMESTAMP: Moreover Same<br>2. SYSDATETIME / SYSDATETIMEOFFSET : Difference is just accuracy<br>3. GETUTCDATE / SYSUTCDATETIME : To Get UTC based Time<br>4. ISDATE() - if yes then 1 else 0 (Not Applicable for DateTime2 and offset so max nnn)<br>5. DAY/ Month/ Year (Valid Date) : Return Particualar Value<br>6. DateName/ DatePart: ( part, date): To Gate More Details About Date in Name or Integer |

| DatePart | Abbreviation |
|---|---|
| year | yy, yyyy |
| quarter | qq, q |
| month | mm, m |
| dayofyear | dy, y |
| day | dd, d |
| week | wk, ww |
| weekday | dw |
| hour | hh |
| minute | mi, n |
| second | ss, s |
| millisecond | ms |
| microsecond | mcs |
| nanosecond | ns |
| TZoffset | tz |

7. DateAdd (part,value,date)
8. DateTDiff (part,date1,date2)
9. EOMONTH(date) : End of Month
10. DATEFROMPARTS(year,month,date)
11. DateTIMEFROMPARTS: year, month, day, hour, minutes, s,milis
12. SmallDATETIMEFROMPARTS: year, month, day, hour, minutes
13. DateTIME2FROMPARTS: year, month, day, hour, minutes, s, fractions, precision
14. TIMEfromParts: hh, mm, s, fraction, preciosn
15. DateTIME2(precion)

| | | |
|---|---|---|
| 20. | Cast and Convert | • Both are Use to Convert One Data Type to Another<br>  • CAST(column or value as datatype)<br>  • Convert(datatype, column or value [style/format])<br>  ○ So Just Difference is Can Style Using Convert not By Cast<br>  ○ Can Get Styles from MSDN styles for date<br>  ○ Cast is More Preferable and Acceptable |
| 21. | Mathematical Functions | • There are  Many functions and unlike String Can That from **Programmability/Functions/System Functions/String or Mathematical Functions.**<br>• RAND() : Return float between 0 and 1 and if provide seed always get same value, To Increase Range can increase multiplier.<br>• In ROUND function Third Para Tells Truncate or Round, Minus Places Tells Operation before Point |
| 22. | Functions | • Scaler Function Can Not Return text, Cursor, Timestamp<br>• Inline Tabled Value Function Returns Table and Not Have Begin and End otherwise It Will throw and Error (Returns Table).<br>• In Multi value Table function have to create table variable of Type Table with structure Definition ex: returns @TableName Table (col1 datatype. Col2 datatype.....), and it also need begin and end block, Just Have To write Return<br>• As Returns Table so have to treat like table.<br>• WITH SCHEMABINDING is helpful so that used table can not delectable until Function Is Referring That. |
| 23. | Temporary Table | • Local Temp Table's Scope and Life is till Particular Connection or In Case Of Stored Procedure till execution of Sp only, Can Possible Same Table From Other Connection.<br>• # for Local and ## for Global.<br>• Global Temp Table is till all connection not referring and Unique Across All Users |
| 24. | Indexing | • It is Very helpful as it increase scanning performance, as default table Scanning means row by row every place need to check or handle<br>• Sp_HelpIndex tablename : List Indices<br>• Types:<br>  • Clustered : Only One To Store Data  (Default PK)<br>  • Non Clustered : As Many but store additional like index page<br>  • Unique : To Make Unique Ex: Unique Constraint<br>  • Non Unique : if Not Unique Then its is Not-Unique |
| 25. | View | • Just a Saved SQL Query or Can Say Virtual Table<br>• As View is not storing Data so we can Update Base Table form View<br>• Can Added Index to view But Main Thing is It is Generally Helpful in case of OLAP as not frequent changes are there in tables so.<br>• Can Not Have Order By Clause<br>• Not Applicable on Temp Tables |
| 26. | Triggers | • Type:<br>  • DML : Fires on DML Events like Insert, Update and Delete.<br>  ○ After: After Event Occurred |

| | | |
|---|---|---|
| | | ▪ Maintain Inserted(Inserted or After Update), Deleted(Deleted or Before Update) Table in trigger context only<br>○ Instead:<br>▪ Specifically to solve operation on view in case of Multiple Tables.<br>▪ Update(Column Name) to Check User Updating That Column)<br>• DDL<br>○ Only For , On Database or On ALL Server<br>○ Enable / Disable trigger trDisableCreation on all server<br>○ Create_Table, ALter_Table,Drop_Table only<br>• Logon<br>○ Not A different but Like DDL and Event is LOGON<br>`Create trigger tr_AuditLogin`<br>`ON ALL SERVER`<br>`FOR LOGON`<br>`AS`<br>`BEGIN`<br>`Declare @LoginName nvarchar(100)`<br>`Set @LoginName = Original_Login`<br>`Select is_user_process, original_login_name, *`<br>`from sys.dm_exec_sessions order by login_time desc`<br>`END`<br>• CLR |
| 27. | Others | • Select * into tablename from tablename, Insert into tblname select * tablename;<br>• While(Exists())<br>Begin<br>End<br>• colname from tablename<br>• Delete from tblname join tblname2 on : Which Delete All Matched Rows<br>• Can Declare Table Variable and use it like table or temp table<br>• TOP<br>• Object_ID('name') function to check whether Object Exists Or Not<br>• To Get All data of Store Proc Can Set Name=@Name or @Name id NULL<br>`EXEC sp_settriggerorder`<br>`@triggername = 'tr_DatabaseScopeTrigger1',`<br>`@order = 'first',`<br>`@stmttype = 'CREATE_TABLE',`<br>`@namespace = 'DATABASE'`<br>`GO`<br>•<br>`Select is_user_process, original_login_name, *`<br>• `from sys.dm_exec_sessions order`<br>• Original_Login() to get name of current User<br>• Sp_readerrorlog<br>• DATALENGTH for Get Size of DataType |
| 28. | Derived Table | • Select id name from () as tblname; |
| 29. | Common Table Expression | • With tblname(column list) as (select) query, tblname(column list) as select query, . ,....<br>• Is Useful Which immediate Follow by use of that.<br>• Can Change Base Table but if not multiple base then only can change CTE. |
| 30. | PIVOTING | • To Rotate Data one Dimension (Aggregation is Required)<br>• PIVOT<br>(<br>AggregateFunction<br>FOR Column IN Values()<br>) |
| 31. | Error Handling | • @@ERROR()<br>• Return 0 if not else Any Error is there<br>• Rollback and commit to commit Transaction |

| | | |
|---|---|---|
| | | • Main Thing is it Clear and Reset after Every Statement so May Problematic to Use Directly, need to use local Varibale<br>• Try Cacth<br>• Transaction is in Try Block (Begin Try and End TRY)<br>• Catch Block is After Try Block (Begin Catch and End Catch) |
| 32. | Transaction | • Batch of Commands To Maintain Integrity<br>• Begin Tran or Begin Transaction then Commit Tran or Rollback Tran |
| 33. | Sub Queries | Can Independent or Corelated to Outer Query |
| 34. | Cursor | Declare name cursor FOR select Query<br>Open name<br>Fetch Next From name into @var1 @car2<br>While(@@fetch_status=0)<br>Fetch Next From name into @var1 @car2<br>Close name // Release the result Set<br>Deallocate name; |
| 35. | To Get Objects of Database | Object type. Can be one of the following object types:<br>AF = Aggregate function (CLR)<br>C = CHECK constraint<br>D = Default or DEFAULT constraint<br>F = FOREIGN KEY constraint<br>L = Log<br>FN = Scalar function<br>FS = Assembly (CLR) scalar-function<br>FT = Assembly (CLR) table-valued function<br>IF = In-lined table-function<br>IT = Internal table<br>P = Stored procedure<br>PC = Assembly (CLR) stored-procedure<br>PK = PRIMARY KEY constraint (type is K)<br>RF = Replication filter stored procedure<br>S = System table<br>SN = Synonym<br>SQ = Service queue<br>TA = Assembly (CLR) DML trigger<br>TF = Table function<br>TR = SQL DML Trigger<br>TT = Table type<br>U = User table<br>UQ = UNIQUE constraint (type is K)<br>V = View<br>X = Extended stored procedure<br>Select * from SysObjects where XTYPE=' from above list'<br>Select * from sys.objects type<br>Select * from Information_Schema |
| 36. | Re Runnable Query | • Just check Whether Possible or Not before execute Query |
| 37. | Merge | Merge table1 as t1<br>Using table2 as t2<br>On Condition Like Join<br>When MATCHED then<br>Code<br>When NOT MATCHED By Target then<br>Code<br>When Not Matched By Source Then<br>Code<br>  <br>All Are Not Compulsory means We Can Choose From That |

| | | |
|---|---|---|
| 38. | Except Operator | Just Like Union, It Return Pure Data From A Which Not Contain Shared Data Between Two, It Remove Duplicate But Not In Don't |
| 39. | Intersect Operator | It Return Common Between 2 Tables, Same Duplication |
| 40. | Cross Apply and Outer Apply | Like Inner Join and Left Join, Basically to Join with Table and Table Value Function |
| 41. | More On Select | • Select * Into baktblname from tblname<br>  • Baktblname must not exists. |
| 42. | Creation Of User Type | Create type Name as Table<br>(Table Structure of Which we want to create Type) |
| 43. | Passing of table as Para | • Need To Create Type Of That Table<br>• Have To Declare Para of That Type With ReadOnly as passed Table Must be ReadOnly No DML Accepts<br>`-- Declare the Table Variable`<br>`DECLARE @EmployeeTableType EmpTableType`<br><br>`-- Insert rows into the Table variable that you want to send to the SP`<br>`INSERT INTO @EmployeeTableType VALUES (1, 'Mark', 'Male')`<br>`INSERT INTO @EmployeeTableType VALUES (2, 'Mary', 'Female')`<br>`INSERT INTO @EmployeeTableType VALUES (3, 'John', 'Male')`<br>`INSERT INTO @EmployeeTableType VALUES (4, 'Sara', 'Female')`<br>`INSERT INTO @EmployeeTableType VALUES (5, 'Rob', 'Male')`<br><br>`-- Pass the table variable as a parameter to the stored procedure`<br>`EXECUTE spInsertEmployees @EmployeeTableType`<br>• Just Helpful When Want to Use Like Structure Variable Group of Data Types Thats It.<br>• To Pass Just have to Create SqlParameter of that Type and Value Is Created Datawith That StructureTable |
| 44. | Grouping Sets | Group By<br>Grouping Sets<br>(<br>  (col1,col2,...),<br>  (column List)<br>  (May Possible No Columns)<br>)<br>Also<br>  • Order by GROUPING(Column from Group1),GROUPING(Column from Group2)<br>  • If We Want Then Can use Rollup or cube as per need To Avoid Manual List of Column<br>  • GROUPING Function (column) : if Aggregated then 1 else 0<br>  • Grouping ID(List Of Columns) Returns Binary Data Of Individual Grouping |
| 45. | Over Clause | • Aggregate Function(column Name) Over (<br>PARTITION by column name)<br>• By Column Name)<br>• ROW_NUMBER over (order by column name) |
| 46. | Choose Function | • Choose (Index,Values) Like Enum |
| 47. | IIF() | • IIF(exp,Val1,Val2) like Ternary Operator |
| 48. | Sequence Object | • Main Use To Create Automatic Counter<br>• Signature Is Like<br>Create Sequence dbo.Name<br>As datatype number<br>Start with val number<br>Increment by number<br>Minvalue number<br>MaxValue number<br>Cycle<br>Cache number<br><br>Next value for dbo.sequence name<br>• To Get Current Sequence current Value from sys.sequences where name=<name of sequence><br>• To Reset Alter sequence name reeset value<br>• Main Thing that it can be shared |

| | | |
|---|---|---|
| 49. | GUID | • Unique Identifier (Data Type)<br>• NEW ID() to get value<br>• 32 char size (16 byte size)<br>• Identity is not allowed But Using Default<br>• To Create Empty GUID<br>`Select CAST(CAST(0 as Binary) as uniqueidentifier)`<br>`Select Cast(0x0 as uniqueidentifier)` |
| 50. | Dynamic Sql | `Declare @sql nvarchar(1000)`<br>`Declare @params nvarchar(1000)`<br>`Set @sql = 'Select * from Employees' + 'Where FirstName=@FirstName and LastName=@LastName'`<br>`Set @params = '@FirstName nvarchar(100), @LastName nvarchar(100)'`<br>`--Execute sp_executesql @sql, @params, @FirstName='Mark', @LastName='Hastings'` |
| 51. | Plan cache | `SELECT cp.usecounts, cp.cacheobjtype, cp.objtype, st.text, qp.query_plan`<br>`FROM sys.dm_exec_cached_plans AS cp`<br>`CROSS APPLY sys.dm_exec_sql_text(plan_handle) AS st`<br>`CROSS APPLY sys.dm_exec_query_plan(plan_handle) AS qp`<br>`ORDER BY cp.usecounts DESC`<br>`DBCC FREEPROCCACHE` |
| 52. | Exec()<br>sp_executesql | • EXEC Accept only Ona para<br>• In Sp_executeSQL Can Pass Paras |
| 53. | QuoteName() | • It Put Values in Quote and Table Name in [] to avoid Sql Injection<br>• Quote Name ' <Type Of Quotation Like [ or ] or ' ' > '<br>• Undone Effect by PARSE |

**Note** :

- If we directly execute query then all queries got execute but By Selecting and Execute then we can execute selected Query from all list of query in Single Query Window.
- Best way to declare table with tbl prefix and stored procedure with sp prefix not by 'sp_'.
- As to give alias and if Column Name contain Space then [Column Name].
- Main DB Rule is Do Thing As Early as Possible.
- Union Combine the rows of table where as Join combine columns based on logical relationship.
- Inline Table Value Function is batter in performance and sql treat as view, where as Multi Value is treat like Stored Procedure, as Value function can update table which getting from Function.
- Primary Key Uses Clustered and Unique Indexing To Become Unique.
- By Indexing Can Create Constraint.
- As SP, Functions and Views are From Table so signature is like
  - Create or alter proc/function/view name
    Params if (In case of Functions, sp ) and Return Type (In Case of function)
    As
    Begin // If Not Inline Table Value Function
    // code
    // return (In Case Of Function)
    End
- In Case of Index and Trigger as are on Table so Signature is Like
  - Create {Type} INDEX / TRIGGER name
    On <tblname or viename> {Colname}
    {These Are for Index Only}
    ########## For Trigger Only #########
    FOR <insert/delete/update>
    As
    Begin
    End
- Raiserror to Throw Error in DB
- Scope of Variable also stored in DB not in memory

- View, Table value function
- Inner Join Two Null as Different Values
- TableName.* To get All From Any Tabel.

**Observations:**

- In Case of identity when Insert Statement executes then it increment the Identity counter first, so if in case Insert statement is throw an error then also identity counter no get decremented, and next successful will not stored on last stored record's identity + 1 instead of that identity + 1 + in between failed insert Statement. (Not Applicable For Compiling Query)
- If Default is Not Set And Set Default option as Insert Update Specification in FK then it gives NULL.
- As ASCII is till 256 so if provide more then that to char() it return NULL
- Length of Null String is Null same for LEFT and RIGHT, TRIMING functions, Replicate,
- If Date is Not Valid Format then Day, Month and Other will Not work.
- If Multiple Tables are involved in View or inline table value function then Insertion, Updating & deletion may throw error if multiple table get affected.
- String_AGG() function to aggregate strings and in case of pivot if want data which is varchar then min or max is fine.
- May Possible One DDL Trigger affects on Other. Like If We set can to allow create, alter trigger then if we try to create second DDL trigger then not allowing Us to do that.

**Confusions and Solutions:**

Some **Differences Between VARCHAR** and **TEXT**

The VAR in **VARCHAR** means that you can set the max size to anything **between** 1 and 65,535. **TEXT** fields have a fixed max size of 65,535 characters. ... Meanwhile, **TEXT** is stored off table with the table having a pointer to the location of the actual storage. 19-Feb-2020

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**.
15-Sep-2012

```
SELECT
    c1, c2, aggregate_function(c3)
FROM
    table
GROUP BY ROLLUP (c1, c2);
```

The `ROLLUP` assumes a hierarchy among the input columns. For example, if the input column is (c1,c2), the hierarchy `c1 > c2`. The `ROLLUP` generates all grouping sets that make sense considering this hierarchy. This is why we often use `ROLLUP` to generate the subtotals and the grand total for reporting purposes.

In the syntax above, `ROLLUP(c1,c2)` generates three following grouping sets:

```
(c1,c2)
(c1)
()
```

**Useful Links:**

- Date and Time Style For Convert : https://www.mssqltips.com/sqlservertip/1145/date-and-time-conversions-using-sql-server/
- Events for DDL triggers in SQL: https://docs.microsoft.com/en-us/sql/relational-databases/triggers/ddl-events?view=sql-server-ver15

---

**Main Thing is Database =>** Stores Data In Form of Tables and Also Store Other Things Like Stored Procedure, Functions, Views, Indexes, Constraints

**Data Storage =>** Mainly For Data Storing Two Option

1. Temporary Storage (Local Scope Table and Global Scope Table)
2. Permanent Storage ( Data Tables)

**Tables** => Have Columns and Rows Store the Data, Columns May Have Its Own Constraints like
1. Unique (For NULL too, Multiple Nulls are Not Allowed)
2. Check (For Condition Checking)
3. Default value
4. Primary Key
5. Foreign Key
6. Null or Not Null
7. Identity (Auto Increment)

=> Can Add Constraint Inline While Defining Columns
=> Can Add Constraint After Creation of Table too. Using Alter Statement
=> As Table Contains Data So Insert, Update and Delete Statement are There To Handle Data Manipulation
=> Select command is there to display data.
=> Using Alter, Drop, Create can Manipulate Table
=> Using Indexing Can Retrieve smoothly.

**Data Types** => There are Many Data Types like String, Numbers, Binary, Bit, Date, Time, DateTime
=> To Cast The Data We Also Have CAST(value as dbtype), Convert(datatype, value[style])
=> Predefine Function like For Numeric Round, Floor and Many More, For String Left(), Right() and Many More and For DateTime GETDATE(), YEAR() and Many more

**Functions** => Can Create Different Functions like Scaler Function, Inline Table Value and Multi-Line Table Value Function
=> Inline Value Function is Treated Like View whereas Multi-Line is Like Stored Procedure.
=> As (INLINE) table Value Function Returning Value So Can Modify Data which is direct replicate to base table, But Main Is Function is Storing Value they just stores query

**Stored Procedure =>** Can Say it is special Kind Of Function Which Have both Input and Output Para, But it is also Storing Queries Which is pre compiled so Not Need To Compile Every time When It Called.

**Views** => Unlike Function it is also storing Queries but using Triggers we can Change Data Of Base Tables too not matter base tables are more then one and changing affects Multiple Tables (If Designed only then if Not Want to all then Different Case).
=> It is also Called Virtual Table and if data of View not rapidly Got Updated Then Can Create Indexed View Too For Batter Performance. In This case View Is Capable To Storing Data.

**Triggers** => Triggers are also similar to function and Stored Procedure just main Difference is it execute automatically When Event like Insertion Update and Delete Occurs. Can Not Call Manually.

**Aggregation** => To Aggregate the data we have group by clause and along with having To Filter Data, Also Aggregate Function Like SUM, Count.

**Sorting and Filtering** => To Sort The Data Have Order By Clause and For Filter Have Where clause
=> Special Operators like LIKE, IN, NOT, BETWEEN and Other Logical and Conditional Opearators
=> Distinct for Remove Duplication, TOP number to select that much records only.

**Joins and Set Operations** => Basically Just Small Difference Between Set Operations and Joins is Set Operation Require Same Type of Structure of tables and Joins are Used to Different tables via Foreign Key Primary Key Relation ship to Get More Idea as Data Base is split for normalization
=> Set Operation Include UNION, UNION All, INTERSECT, EXCEPT
=> Different Types of Joins are Inner, Outer (Left, right, Full), Cross Join and Self Join (Basically Not Different Type).
=> Along With This CROSS APPLY and OUTER APLLY is there which are similar to joins but useful in case of Join data of table and data coming from Table Value Function.

**Error and Handling of Error** => Unlike Programming language we can raise and Handle Error
=> Raiserror() to Raise the error
=> @@ERROR to check error is there or not but before every line scan in get reset so Need To take Casre
=> Try and catch to Handle Error

**Derived Table** => We Can Derive From table too in case of intermediation storage
=> One Of the Famous way is CTE (Common Expression Table) basically work like view which stores the query but helpful.

**Transaction** => As Database Must be in Consistent state after every transaction so for consistency we are using transaction so that if any error raise then all transaction get rollback or get execute.

**PIVOTING** => It is way by which we can rotate Whole Data based on one dimension For Example If Data is Like

| Gender | Population |
|---|---|

| | |
|---|---|
| Male | 20000 |
| Female | 10244 |

Then PIVOTING ON GENDER

| | Male | Female |
|---|---|---|
| Population | 20000 | 10244 |

**Handling of Null Values** => Can Replace Null Values with no null as part of pre process data using Functions like ISNULL(), COALESCC() and CASE statement.