SQL DATE FUNCTIONS

Below are the commonly used DATE Functions:

FUNCTION	SYNTAX
GETDATE	GETDATE()
DATEADD	DATEADD (datepart , number , date)
DATEDIFF	DATEDIFF (datepart , startdate , enddate)
DAY	DAY(DATE)
MONTH	MONTH(DATE)
YEAR	YEAR(DATE)
DATEPART	DATEPART(datepart, datecolumnEID)
CONVERT	CONVERT(datatype, expression, style)
FORMAT	FORMAT (getdate(), 'D') -> Wednesday, September 6, 2017

```
Select CONVERT (varchar(19),getdate()) -> Sep  6 2017 11:24PM
Select CONVERT (varchar(19),getdate(),10) -> 09-06-17
Select CONVERT (varchar(19),getdate(),110) -> 09-06-2017
```

ASSIGNMENT

ASSIGNMENT – 6

A-1: DEPARTMENT WISE TEAM SIZE AND AVERAGE SALARY OF ALL EMPLOYEES.

A-2 : COUNT OF MANAGERS IN THE COMPANY.

A-3: MAXIMUM & MINIMUM SALARY OF AN ASSOCIATE.

A-4: DEPARTMENT WISE TEAM SIZE AND AVERAGE SALARY OF DELHI EMPLOYEES.

A-5: GENERATE OFFICIAL EMAIL OF THE EMPLOYEE TAKING 1ST CHARATCET OF FIRST NAME , 1ST CHARATCER OF LAST NAME , LAST 3 DIGITS OF EID, FOLLED BY ‘RCG.COM’. EMAIL SHOULD BE IN A UPPER CASE.

A-6: NAME,CITY , PHNO & EMAIL OF THE EMPLOYEES WHOSE AGE >=40.

A-7 EID, NAME DOJ OF EMPLOYEES WHO HAVE COMPLETED 5 YEARS IN THE COMPANY

A-8: DETAILS OF THE MANAGERS HAVING BIRTHDAY IN THE CURRENT MONTH

A-9 : EID, DEPT , DESI , SALARY OF THE EMPLOYEE WHO IS GETTING THE MAXIMUM SALARY

A-10 : EID, NAME OF EMPLOYEE WHO HAS LONGEST NAME

USER DEFINED FUNCTIONS

User Defined functions can be used to perform a complex logic, can accept parameters and return data.

SQL Server supports two types of User Defined Functions as mentioned below

Scalar Functions – The function which returns a Scalar/Single value.

```
CREATE FUNCTION MYSUM (@A INT, @B INT )
RETURNS INT
AS
BEGIN
    DECALRE @C AS INT;
    SET @C=@A+@B;
    RETURN @C;
END;

SELECT DBO.MYSUM(10,20);
DROP FUNCTION MYSUM;
```

USER DEFINED FUNCTIONS

Table Valued Functions – The function which returns a row set of SQL server Table.

```
CREATE FUNCTION GETEMP (
    @DEP VARCHAR(50)
) RETURNS TABLE
AS

RETURN ( SELECT * FROM EMP WHERE DEPT = @DEP)
;

SELECT * FROM DBO.GETEMP('ADMIN')
```

ASSIGNMENT

ASSIGNMENT – 7

A-1: CREATE A FUNCTION CALC TO PERFORM THE SPECIFIED OPERATION ON THE GIVEN TWO NUMBERS .

A-2: FUNCTION TO GENERATE THE EMAIL ID BY ACCEPTING NAME & EID. EMAIL SHOULD CONTAIN 1ST CHARACTER OF 1ST NAME , 1ST CHARACTER OF LAST NAME, LAST 3 DIGITS OF EMP ID FOLLOWED BY @RCG.COM;

A-3: FUNCTION TO RETURN EID, NAME, DESI, DEPT ,SALARY OF THE EMPLOYEES OF A SPECIFIED DEPARTMENT.

A-4: FUNCTION TO DISPLAY THE NAME , DEPT . DESI , CITY OF THE EMPLOYEES WHO HAVE THE BIRTHDAY IN THE CURRENT MONTH.

A-5: FUNCTION TO DISPLAY THE NAME, DEPT & DOJ OF EMPLOYEES WHO HAVE COMPLETED 5 YEARS IN THE COMPANY.

SQL SUB QUERIES

Sub Queries

A Subquery or Inner query or Nested query is a query within another SQL query, and embedded within the WHERE clause.

A subquery is used to return data that will be used in the main query as a condition to further restrict the data to be retrieved

There are a few rules that subqueries must follow:

- Subqueries must be enclosed within parentheses.

- A subquery can have only one column in the SELECT clause.

- An ORDER BY cannot be used in a subquery, although the main query can use an ORDER BY.

- Subqueries that return more than one row can only be used with multiple value operators, such as the IN operator.

- The BETWEEN operator cannot be used with a subquery; however, the BETWEEN can be used within the subquery.

Simple Sub Queries

Subqueries are most frequently used with the SELECT statement. The basic syntax is as follows:

```
SELECT column_EID [, column_EID ]
FROM   table1 [, table2 ]
WHERE  column_EID OPERATOR
(SELECT column_EID [, column_EID ]
FROM table1 [, table2 ]
[WHERE])
```

Note: Although Subqueries are commonly used with Select statement, these can also be used with Insert, Update or Delete Statements

Correlated Sub Queries

There are ways to incorporate the outer query’s values into the subquery’s clauses. These types of queries are called correlated subqueries, since the results from the subquery are connected, in some form, to values in the outer query. Correlated queries are sometimes called synchronized queries.

Eg:

Corelatted sub query for average salary of those delhi employees whose salary is >100000

```
select avg(salary) from emp_sal

where eid in (select eid from emp where city='Delhi') and

eid in (select eid from emp_sal where salary > 100000);
```

SQL EXISTS Operator

- The EXISTS operator is used to test for the existence of any record in a subquery.
- The EXISTS operator returns true if the subquery returns one or more records.

```
SELECT column_EID(s)
FROM table_EID
WHERE EXISTS
(SELECT column_EID FROM table_EID WHERE condition);
```

ASSIGNMENT

ASSIGNMENT – 7

A-1 : EID, NAME, CITY OF GURGAON EMPLOYEES

A-2 : EID, NAME , DOJ ,DEPT, DESI & SALARY OF ALL MANAGERS

A-3: REDUCE THE SALARY OF ALL DELHI EMPLOYEES BY 10%.

A-4 : DISPLAY THE EID, NAME , CITY, DOJ ,DEPT, DESI & SALARY OF THE TEAM MEMBERS OF DAVID & RAMESH GUPTA.

A-5: CREATE A TRAINING TABLE CONTAINING EID, NAME, DEPT. INSERT THE DETAILS OF OPS TEAM MEMBERS IN THE TRAINING TABLE.

A-6: DETAILS OF DIRECTORS SHOULD BE DELETED FROM THE TRAINING TABLE.

A-7: DISPLAY THE SALARY DETAILS OFF ALL EMPLOYES IF ANY OF THE TEAM MEMBER HAS SALARY MORE THAN 200000.

STORED PROCEDURES

Stored Procedures

- A stored procedure is prepared SQL code that we save so we can reuse the code over and over again. So if we think about a query that we write over and over again, instead of having to write that query each time we would save it as a stored procedure and then just call the stored procedure to execute the SQL code that we saved as part of the stored procedure.
- In addition to running the same SQL code over and over again we also have the ability to pass parameters to the stored procedure.

SYNTAX

```
CREATE PROCEDURE <procedure_EID>
AS
BEGIN
<SQL Statement>
END

EXECUTE <procedure_EID>
EXEC  <procedure_EID>
 <procedure_EID>
```