**Built In Functions(System Functions) IN SQL:**

SQL server provide number of built in functions like mathematical functions, character functions, date and time functions, aggregative functions, conversion functions etc. these can be used to perform certain operations and return a value.

* Syntax: SELECT <Function Name> [Expressions]

**Mathematical Functions:** These functions perform a calculation based on

input values provided as arguments, and return a numeric value.

**ABS ():** Returns the absolute, positive value of the given numeric expression.

* Ex: select ABS(-15)---- 15
* select ABS(45)----- 45

**CEILING ():** Returns the smallest integer greater than, or equal to, the given numeric expression.

* Ex: select ceiling(15.000)----15
* select ceiling(15.0001)----16
* select ceiling(-12.34)-----(-12)

**FLOOR ():** Returns the largest integer less than or equal to the given numeric expression.

* Ex: select floor(15.000)---15
* select floor(15.0001)----15
* select floor(-12.34)----(-13)

**SQUARE ():** Returns the square of the given expression.

* Ex: select SQUARE(5)---25

**SQRT ():** Returns the square root of the given expression.

* Ex: select SQUARE(25)---5

**SIGN ():** Returns the positive (+1), zero (0), or negative (-1) sign of the given expression.

* Ex: select SIGN(42)------------1
* select SIGN(0)-------------0
* select SIGN(-42)-----------(-1)

**PI ():** Returns the constant value of PI.

* Ex: select PI()---------3.14159265358979

**LOG ():** Returns the natural logarithm of the given expression.

* Ex: select LOG(2)------ 0.693147180559945

**LOG 10():** Returns the base-10 logarithm of the given expression.

* Ex: select LOG10(10)----1

**SIN ():** Returns the trigonometric sine of the given angle (in radians) in an approximate numeric expression.

* Ex: select SIN (0) -------0

**COS ():** A mathematic function that returns the trigonometric cosine of the given angle (in radians) in the given expression.

* Ex: select COS (0) -------1

**TAN ():** Returns the tangent of the input expression.

* Ex: select TAN (0) ---------0

**String Functions:** These functions perform an operation on a string input value

and return a string or numeric value.

**ASCII ():** Returns the ASCII code value of the leftmost character of a character expression.

* Ex: Select ASCII (‘Z’) -----90

**CHAR ():** A string function that converts an int ASCII code to a character.

* Ex: Select CHAR (90) -----Z

**CHARINDEX ():** Returns the starting position of the specified expression in a

character string.

* Ex: Select CHARINDEX (‘S’,’SUDHAKAR’) -------1

**LEFT ():** Returns the left part of a character string with the specified number of

characters.

* Ex: Select LEFT (‘SUDHAKAR’, 5) ----SUDHA

**RIGHT ():** Returns the right part of a character string with the specified number of characters.

* Ex: Select RIGHT (‘SUDHAKAR’, 3) ------KAR

**LEN ():** Returns the number of characters, rather than the number of bytes, of the given string expression.

* Ex: Select LEN (‘WELCOME’) ------------7

**LOWER ():** Returns a character expression after converting uppercase character data to lowercase.

* Ex: Select LOWER (‘SAI’) --------sai

**UPPER ():** Returns a character expression with lowercase character data converted to uppercase.

* Ex: Select UPPER (‘sai’) ------SAI

**LTRIM ():** Returns a character expression after removing leading blanks.

* Ex: Select LTRIM (‘ HELLO’) --------HELLO

**RTRIM ():** Returns a character string after truncating all trailing blanks.

* Ex: Select RTRIM (‘HELLO ‘) -------HELLO

**REPLACE ():** Replaces all occurrences of the second given string expression in the first string expression with a third expression.

* Ex: Select REPLACE (‘JACK AND JUE’, ‘J’, ‘BL’) ------BLACK AND BLUE

**REPLICATE ():** Repeats a character expression for a specified number of times.

* Ex: Select REPLICATE (‘SAI’, 3) -------SAISAISAI

**REVERSE ():** Returns the reverse of a character expression.

* Ex: Select REVERSE (‘HELLO’) --------OLLEH

**SPACE ():** Returns a string of repeated spaces.

* Ex: Select (‘SAI’+SPACE (50) +’SUDHAKAR’) -----SAI SUDHAKAR

**Date and Time Functions:** These functions perform an operation on a

date and time input value and return a string, numeric, or date and time value.

**GETDATE ():** Returns the current system date and time in the SQL Server standard internal format for date time values.

* Ex: Select GETDATE () ------- 2014-02-15 15:35:22.670

**DAY ():** Returns an integer representing the day date part of the specified date.

* Ex: Select DAY (get date ())

**MONTH ():** Returns an integer that represents the month part of a specified date.

* Ex: Select MONTH (get date ())

**YEAR ():** Returns an integer that represents the year part of a specified date.

* Ex: Select YEAR (get Date ())

**DATE NAME ():** Returns a character string representing the specified date part of the specified date.

* Ex: Select DATE NAME (DW, get date ())

**DATE PART ():** Returns an integer representing the specified date part of the

specified date.

* Ex: Select DATEPART (DD, get date ())

**DATE ADD ():** Returns a new date time value based on adding an interval to the specified date.

* Ex: Select DATEADD (DD, 5, get date ())

**DATE DIFF ():** Returns the difference between the start and end dates in the give date part format.

* Ex: Select DATEDIFF (MM, ‘2012-12-15’, get date ())

**Conversion Functions:** These functions are used to convert one data type

to another. We have two conversion functions are CAST and CONVERT both

provide similar functionality.

**CAST ():** Convert to one data type to another type.

* Syntax: CAST (Expression as data type [size])
* Ex: Select CAST (10.2587 as Int) -------------10

**CONVERT ():** Convert function can be used to display date time data in different format.

* Syntax: Convert (Data type [size], Expression, Style value)
* Ex: Select Convert (Varchar (24), get date (), 113)

**Aggregate functions/Group functions:** Aggregate functions perform

a calculation on a set of values and return a single value. Aggregate functions are

Often used with the GROUP BY clause of the SELECT statement.

**SUM ():** Returns the sum of all the values .Sum can be used with numeric columns

only. Null values are ignored.

* Ex: SELECT SUM (SALARY) FROM EMP

**AVG ():** Returns the average of the values in a group. Null values are ignored.

* Ex: SELECT AVG (SALARY) FROM EMP

**MAX ():** Returns the maximum value in the expression.

* Ex: SELECT MAX (SALARY) FROM EMP

**MIN ():** Returns the minimum value in the expression.

* Ex: SELECT MIN (SALARY) FROM EMP

**COUNT ():** Returns the number of records in a table. This function again use in three ways.

* COUNT (\*): It Returns total number of records in a table
* Ex: SELECT COUNT (\*) FROM EMP
* COUNT (Expression/Column name): It returns number of records
* including duplicate values but not null vales.
* Ex: SELECT COUNT (ENAME) FROM EMP
* COUNT (Distinct Column name): It returns number of records without
* null and duplicate values.
* Ex: SELECT COUNT (Distinct ENAME) FROM EMP

**Distinct Key:** If we use this key word on a column with in a query then it will

retrieve the values of the column without duplicates.

**OPERATORS IN SQL:** Operator is a symbol which performs some

specific operation on operands or expressions. These operands are classified into 6

types in SQL.

1. Assignment operator

2. Arithmetic operator

3. Comparison operator

4. Logical operator

5. Set operator

**Assignment operator:** Assignment operator contain only one operator is

knows as equal ‘=’ operator.

Ex1: Write a Query to display the employee details whose salary is equal to10000

* SELECT \* FROM EMP WHERE SAL=10000

Ex2: Write a query to change the deptno as ‘10’whose employee id is 101

* UPDATE EMP SET DEPTNO=10 WHERE EID=101

Ex3: Write a query to delete a record whose employee id is 107

* DELETE FROM EMP WHERE EID=107

**Arithmetic operator:** Arithmetic operators perform mathematical

operations on two expressions.

The lists of arithmetic operators are + (Add), -Subtraction,\* Multiplication. / (Divide) Division.% (Modulo) Returns the integer

remainder of a division. For example, 12 % 5 = 2 because the remainder of 12

divided by 5 is 2.

Ex1: Select 100+250

Select 245-400

Select 20\*20

Select 25/5

Select 37%6

Select 20/5+20/5

Select 35.50+20

Ex2: WAQ to find student TOTAL, AVERAGE AND CLASS OF a table

Step1: Create table student (Sid int, sname varchar (50), math’s int, phy int, che

int, total int, average int, class varchar (max))

Step2: Update student set total=maths+phy+che

Step3: Update student set average=total/3

Step4: Update student set class=

Case

When average>=60 then 'First class'

When average>=50 then 'second class'

When average>=40 then 'third class'

Else

'Fail'

End

**Comparison operators:** Comparison operators test whether two

expressions are the same. Comparison operators can be used on all expressions

except expressions of the text, ntext, or image data types. The following table lists

the Transact-SQL comparison operators are > (Greater Than),< (Less Than) ,>=

(Greater Than or Equal To) ,<= (Less Than or Equal To) ,!= (Not Equal To),!< (Not

Less Than),!> (Not Greater Than)

Examples:

* Select ename from EMP where salary<50000
* Update EMP set salary=1000 where salary>90000
* Update EMP set ename='joshitha' where salary<=25000
* Update EMP set salary=98000 where salary>=1000
* Select ename from Emp where salary !>98000
* Select ename from Emp where salary !<98000
* Select ename from Emp where salary !=98000

**Logical operator:** Logical operators test for the truth of some condition.

Logical operators, like comparison operators, return a Boolean data type with a

value of TRUE or FALSE. Logical operators are AND , OR , NOT, BETWEEN,

NOT BETWEEN, LIKE, NOT LIKE, IN, NOT IN, EXISTS,NOT EXISTS, ANY,

ALL, SOME.

Examples:

* Select \* from EMP where ename='siddhu' and salary=45000
* Select \* from EMP where ename='joshitha' or salary=98000
* Select \* from EMP where not ename='joshitha'
* Select \* from EMP where salary between 10000 and 50000
* Update EMP set ename='SAI' where eid=101 and salary=25000

Queries Using ‘Select’ with ‘where’ clause:

**Q.**- Write a Query to display the employee details whose salary is less than10000

**Ans**. SELECT \* FROM EMP WHERE SAL<10000

**Q.** Write a Query to display the employee details whose salary is greater than or equal to 9000 and less than 15000

**Ans**. SELECT \* FROM EMP WHERE SAL>=9000 AND SAL<=15000

(OR)

**Ans**. SELECT \* FROM EMP WHERE SAL BETWEEN 9000 AND 15000

**Q.** Write a Query to display the employee details whose salary is not between 9000 and 15000

**Ans**. SELECT \* FROM EMP WHERE SAL NOT BETWEEN 9000 AND15000

**Q.** Write a Query to display the employee details whose name starts with ‘r’

**Ans**. SELECT \* FROM EMP WHERE ENAME LIKE ‘r%’

**Q.** Write a Query to display the employee details whose name ends with ‘y’

**Ans**. SELECT \* FROM EMP WHERE ENAME LIKE ‘%Y’

**Q.** Write a Query to display the employee details whose name contains the letter ‘a’

**Ans**. SELECT \* FROM EMP WHERE ENAME LIKE ‘%A%’

**Q.** Write a Query to display the employee details whose names contains only three letters

**Ans**. SELECT \* FROM EMP WHERE ENAME LIKE ‘---‘

**Q.** Write a Query to display the employee details whose names contain ‘r’ and salary greater than 9000

**Ans**. SELECT \* FROM EMP WHERE ENAME LIKE ‘%R%’ AND SAL>9000

**Q.** Write a Query to display the employee details whose greater than ram

**Ans**. SELECT \* FROM EMP WHERE ENAME>’RAM’

**Q.** Write a Query to display the employee details whose employee id starts with 1 and ends with 1

**Ans**. SELECT \* FROM EMP WHERE EID LIKE ‘1%1’

Note :-(SQL commands are not case sensitive and also data available in SQL also not case

sensitive, in oracle Data available is case sensitive)

**Queries using ‘Update’ with ‘where’ clause:**

**Q.** Write a query to change the deptno as ‘10’whose employee id is 101, 103, 107

**Ans**. UPDATE EMPSET DEPTNO=10 WHERE EID=101 OR EID=103 OR EID=107

**Q.** Write a query to change the deptno as 20 who does not have deptno

**Ans**. UPDATE EMPSET DEPTNO=20 WHERE DEPTNO IS NULL

**Q.** Write a query to change the employee salaries as 12000 who are working under 10 dept and their names starts with ‘r’

**Ans**. UPDATE EMPSET SAL=12000 WHERE DEPTNO=10 AND ENAME LIKE ‘R%’

**Q.** Write a query to change the deptno as 30 whose second letter is ‘a’

**Ans**. UPDATE EMPSET DEPTNO=30 WHERE ENAME=’-A%’

**Q.** Write a query to change the employee salaries as 8500 who are working under 10 and 20 deptno

**Ans**. UPDATE EMPSET SAL=8500 WHERE DEPTNO=10 OR DEPTNO=20

(OR)

UPDATE EMPSET SAL=8500 WHERE DEPTNO IN(10,20)

**Q.** Write a query to change the employee salaries as 8500 who are not working under 10 and 20 deptno

**Ans**. UPDATE EMPSET SAL=8500 WHERE DEPTNO NOT IN (10,20)

**Q.** Write a query to change the employee salaries as 15000 and names ends with ‘m’ & working under 10 deptno

**Ans**. UPDATE EMPSET SAL=15000 WHERE ENAME=’%M’ AND DEPTNO=10

**Q.** Write a query to change the employee salaries as 5500 whose employee id ends with 4 and deptno starts with 2

**Ans**. UPDATE EMPSET SAL=5500 WHERE EID LIKE ‘%4’ AND DEPTNO LIKE ‘2%’

**Q.** Write a query to change the employee salaries as 25000 whose salary less than 10000 and the name contains letter ‘a’ and working under dept 20

**Ans**. UPDATE EMPSET SAL=25000 WHERE SAL<10000 AND ENAME LIKE ‘%A%’ AND DEPTNO IN (20)

**Q.** Write a query to change the employee salaries as 10000 whose salary is greater than or equal to 8500 and less than or equal to 9000

**Ans**. UPDATE EMPSET SAL=10000 WHERE SAL BETWEEN 8500 AND

9000