EXPERIMENT NO 3

I: **Implementation of different types of SQL functions and SQL operators with suitable examples.**

SQL FUNCTIONS

 Number Function
 Aggregate Function
 Character Function
 Conversion Function
 String Function
 Date Function

1) Number Function:

Abs(n) :Select abs(-15) from dual;
    O/p: 15
Exp(n): Select exp(1) from dual;
    O/p: 2.71828183
Power(m,n): Select power(4,2) from dual;
    O/p:16
Mod(m,n): Select mod(10,3) from dual;
    O/p:1
Round(m,n): Select round(100.256,2) from dual;
    O/p:100.26
Trunc(m,n): ;Select trunc(100.256,2) from dual;
    O/p:100.25
Sqrt(m,n);Select sqrt(16) from dual;
    O/p:4

2) Aggregate Function:

Count
Sum
Avg
Max
Min

1. **Count:** COUNT following by a column name returns the count of tuple in that column. If DISTINCT keyword is used then it will return only the count of unique tuple in the column. Otherwise, it will return count of all the tuples (including duplicates)
count (*) indicates all the tuples of the column.
Syntax: COUNT (Column name)
Example: SELECT COUNT (Sal) FROM emp;
2. SUM: SUM followed by a column name returns the sum of all the values in that column.
Syntax: SUM (Column name)
Example: SELECT SUM (Sal) From emp;
3. AVG: AVG followed by a column name returns the average value of that column values.
Syntax: AVG (n1, n2...)

Example: Select AVG (10, 15, 30) FROM DUAL;
4. MAX: MAX followed by a column name returns the maximum value of that column.
Syntax: MAX (Column name)
Example: SELECT MAX (Sal) FROM emp;
5. MIN: MIN followed by column name returns the minimum value of that column.
Syntax: MIN (Column name)
Example: SELECT MIN (Sal) FROM emp;
SQL>select deptno,min(sal) from emp group by deptno having min(sal)>1000;

3) <u>Character Function</u>:
initcap(char) : select initcap("hello") from dual;
    O/p: Hello
lower (char): select lower ('HELLO') from dual;
    O/p:hello
upper (char) :select upper ('hello') from dual;
    O/p:HELLO
ltrim (char,[set]): select ltrim ('cseit', 'cse') from dual;
    O/p:it
rtrim (char,[set]): select rtrim ('cseit', 'it') from dual;
    O/p:cse
replace (char,search): select replace('jack and jue','j','bl') from dual;
    O/p:black and blue

4) <u>Conversion Functions</u>:

To_char: TO_CHAR (number) converts n to a value of VARCHAR2 data type.
SQL>select to_char(65,'RN')from dual;
LXV
To_number : TO_NUMBER converts expr to a value of NUMBER data type.
SQL>Select to_number ('1234.64') from Dual;
1234.64
To_date:TO_DATE converts char of CHAR, VARCHAR2, NCHAR, or NVARCHAR2 data type to a value of DATE data type.
SQL>SELECT TO_DATE('January 15, 1989, 11:00 A.M.')FROM DUAL;
TO_DATE
15-JAN-89

5) <u>String Functions</u>:

Concat: CONCAT returns char1 concatenated with char2. Both char1 and char2 can be any of the datatypes.
SQL>SELECT CONCAT('ORACLE','CORPORATION')FROM DUAL;
ORACLECORPORATION
Lpad: LPAD returns expr1, left-padded to length n characters with the sequence of characters in expr2.
SQL>SELECT LPAD('ORACLE',15,'*')FROM DUAL;
*********ORACLE
Rpad: RPAD returns expr1, right-padded to length n characters with expr2, replicated as many times as necessary.

SQL>SELECT RPAD (‘ORACLE’,15,’*’)FROM DUAL;
ORACLE*********
Ltrim: Returns a character expression after removing leading blanks.
SQL>SELECT LTRIM(‘SSMITHSS’,’S’)FROM DUAL;
MITHSS
Rtrim: Returns a character string after truncating all trailing blanks
SQL>SELECT RTRIM(‘SSMITHSS’,’S’)FROM DUAL;
SSMITH
Lower: Returns a character expression after converting uppercase character data to lowercase.
SQL>SELECT LOWER(‘DBMS’)FROM DUAL;
dbms
Upper: Returns a character expression with lowercase character data converted to uppercase
SQL>SELECT UPPER(‘dbms’)FROM DUAL;
DBMS
Length: Returns the number of characters, rather than the number of bytes, of the given string expression, excluding trailing blanks.
SQL>SELECT LENGTH(‘DATABASE’)FROM DUAL;
8
Substr: Returns part of a character, binary, text, or image expression.
SQL>SELECT SUBSTR(‘ABCDEFGHIJ’,3,4)FROM DUAL;
CDEF
Instr: The INSTR functions search string for substring. The function returns an integer indicating the position of the character in string that is the first character of this occurrence.
SQL>SELECT INSTR('CORPORATE FLOOR','OR',3,2)FROM DUAL;
1 4

6) <u>Date Function</u>s:
Sysdate:
SQL>SELECT SYSDATE FROM DUAL;
10-MAR-25
next_day:
SQL>SELECT NEXT_DAY(SYSDATE,’WED’)FROM DUAL;
12-MAR-25
add_months:
SQL>SELECT ADD_MONTHS(SYSDATE,2)FROM DUAL;
10-MAY-25
last_day:
SQL>SELECT LAST_DAY(SYSDATE)FROM DUAL;
31-MAR-25
months_between:
SQL>SELECT MONTHS_BETWEEN(SYSDATE,'10-MAY-2024')FROM DUAL;
10
Least:
SQL>SELECT LEAST('10-JAN-24','12-OCT-24')FROM DUAL;
10-JAN-24
Greatest:
SQL>SELECT GREATEST('10-JAN-24','12-OCT-24')FROM DUAL;

12-OCT-24
Trunc:
SQL>SELECT TRUNC(DATE,FMT) FROM DUAL;
Round:
SQL>SELECT ROUND(SYSDATE,'YEAR')FROM DUAL;
01-JAN-25
to_char:
SQL>select to_char(sysdate, 'dd\mm\yy') from dual;
09\03\25.
to_date:
SQL> select to_date (sysdate, ‘dd\mm\yy’) from dual;
24-mar-o5.

II: SQL OPERATORS.

Arithmetic Operator
Logical Operator
Comparison Operator
Special Operator
Set Operator

1) Arithmetic Operators:
(+) : Addition - Adds values on either side of the operator .
(-):Subtraction - Subtracts right hand operand from left hand operand .
(*):Multiplication - Multiplies values on either side of the operator .
(/):Division - Divides left hand operand by right hand operand .
(^):Power- raise to power of .
(%):Modulus - Divides left hand operand by right hand operand and returns remainder.

2) Logical Operators:

AND : The AND operator allows the existence of multiple conditions in an SQL statement's WHERE clause.
OR: The OR operator is used to combine multiple conditions in an SQL statement's WHERE clause.
NOT: The NOT operator reverses the meaning of the logical operator with which it is used.
Eg: NOT EXISTS, NOT BETWEEN, NOT IN, etc. **This is a negate operator.**

3) Comparison Operators:

(=): Checks if the values of two operands are equal or not, if yes then condition becomes true.
(!=): Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.
(< >): Checks if the values of two operands are equal or not, if values are not equal then condition becomes true.
(>): Checks if the value of left operand is greater than the value of right operand, if yes then condition becomes true

(<): Checks if the value of left operand is less than the value of right operand, if yes then condition becomes true.
(>=): Checks if the value of left operand is greater than or equal to the value of right operand, if yes then condition becomes true.
(<=): Checks if the value of left operand is less than or equal to the value of right operand, if yes then condition becomes true.

4) <u>Special operator</u>:

BETWEEN: The BETWEEN operator is used to search for values that are within a set of values, given the minimum value and the maximum value.
IS NULL: The NULL operator is used to compare a value with a NULL attribute value.
ALL: The ALL operator is used to compare a value to all values in another value set
ANY: The ANY operator is used to compare a value to any applicable value in the list according to the condition.
LIKE: The LIKE operator is used to compare a value to similar values using wildcard operators.It allows to use percent sign(%) and underscore (_) to match a given string pattern.
IN: The IN operator is used to compare a value to a list of literal values that have been specified.
EXIST: The EXISTS operator is used to search for the presence of a row in a specified table that meets certain criteria.

5) Set Operators:
The Set operator combines the result of 2 queries into a single result. The following are the operators:
□
Union
Union all
Intersect
Minus

Union: Returns all distinct rows selected by both the queries.It combines the both SELECT statement and removes duplicate rows between them
Union all: Returns all rows selected by either query including the duplicates.
Intersect: Returns rows selected that are common to both queries. It returns all of the rows.
Minus: Returns all distinct rows selected by the first query and are not by the second

LAB PRACTICE ASSIGNMENT:

Create a table EMPLOYEE with following schema:
(Emp_no, E_name, E_address, E_ph_no, Dept_no, Dept_name,Job_id, Designation , Salary)

```
CREATE TABLE EMPLOYEE (
    Emp_no     NUMBER(10) PRIMARY KEY,
    E_name     VARCHAR2(100) NOT NULL,
    E_address  VARCHAR2(255),
    E_ph_no    VARCHAR2(15),
    Dept_no    NUMBER(10),
    Dept_name  VARCHAR2(100),
    Job_id     NUMBER(10),
    Designation VARCHAR2(100),
    Salary     NUMBER(10,2),
    JOIN_DATE DATE;
);
```

INSERT 10 RECORDS TO THE EMPLOYEE TABLE:

```
INSERT ALL
   INTO EMPLOYEE VALUES (201, 'Amit Sharma', '123 MG Road, Mumbai',
'9876543210', 1, 'HR', 2001, 'HR Manager', 75000.00)
   INTO EMPLOYEE VALUES (202, 'Priya Verma', '456 Bannerghatta Rd,
Bangalore', '8765432109', 2, 'IT', 2002, 'Software Engineer', 85000.00)
    INTO EMPLOYEE VALUES (203, 'Rajesh Iyer', '789 T Nagar, Chennai',
'7654321098', 3, 'Finance', 2003, 'Accountant', 60000.00)
    INTO EMPLOYEE VALUES (204, 'Neha Gupta', '321 Park Street, Kolkata',
'6543210987', 1, 'HR', 2004, 'Recruiter', 55000.00)
    INTO EMPLOYEE VALUES (205, 'Vikram Choudhary', '654 Gachibowli,
Hyderabad', '5432109876', 2, 'IT', 2005, 'System Analyst', 90000.00)
   INTO EMPLOYEE VALUES (206, 'Ananya Reddy', '987 Banjara Hills,
Hyderabad', '4321098765', 3, 'Finance', 2006, 'Financial Analyst', 70000.00)
   INTO EMPLOYEE VALUES (207, 'Sandeep Malhotra', '159 Connaught Place,
Delhi', '3210987654', 4, 'Marketing', 2007, 'Marketing Manager', 80000.00)
   INTO EMPLOYEE VALUES (208, 'Ritu Mehta', '753 Karol Bagh, Delhi',
'2109876543', 4, 'Marketing', 2008, 'Sales Executive', 65000.00)

  INTO EMPLOYEE VALUES (209, 'Arjun Patel', '852 SG Highway, Ahmedabad',
'1098765432', 2, 'IT', 2009, 'Database Administrator', 88000.00)
  INTO EMPLOYEE VALUES (210, 'Kavita Joshi', '369 Civil Lines, Jaipur',
'9876543211', 3, 'Finance', 2010, 'Auditor', 62000.00)
SELECT 1 FROM DUAL;
```

Write SQL statements for the following query.

1. **List the E_no, E_name, Salary of all employees working for MANAGER.**

```
SELECT Emp_no, E_name, Salary
FROM EMPLOYEE
WHERE Designation LIKE '%Manager';
```

2. Display all the details of the employee whose salary is more than the Sal of any IT PROFF..
3. List the employees in the ascending order of Designations of those joined after 1981.
4. List the employees along with their Experience and Daily Salary.
5. List the employees who are either ‘CLERK’ or ‘ANALYST’ .
6. List the employees who joined on 1-MAY-81, 3-DEC-81, 17-DEC-81,19-JAN-80 .
7. List the employees who are working for the Deptno 10 or20.
8. List the Enames those are starting with ‘S’ .
9. Dislay the name as well as the first five characters of name(s) starting with ‘H’

10. List all the emps except ‘PRESIDENT’ & ‘MGR” in asc order of Salaries.

1.
Display all the dept numbers available with the dept and emp tables avoiding duplicates.
2. Display all the dept numbers available with the dept and emp tables.
3. Display all the dept numbers available in emp and not in dept tables and vice versa.