4.3 SQL Functions

1.Use numeric functions like ROUND, MOD, POWER on dummy values.

CODE:

SELECT ROUND(126.2587447, 2) FROM dual;

SELECT MOD(10, 3) FROM dual;

SELECT POWER(2, 2) FROM dual;

OUTPUT:

```
SQL> SELECT ROUND(126.2587447, 2) FROM dual;

ROUND(126.2587447,2)
--------------------
              126.26

SQL> SELECT MOD(10, 3) FROM dual;

 MOD(10,3)
----------
         1

SQL> SELECT POWER(2, 2) FROM dual;

POWER(2,2)
----------
         4
```

2. Use string functions like LENGTH, SUBSTR, INSTR, UPPER, LOWER on names in Professors and Students.

CODE:

For Professors:

SELECT Prof_Name,

LENGTH(Prof_Name) AS Length,

SUBSTR(Prof_Name, 1, 5) AS Substr,

INSTR(Prof_Name, 'a') AS Pos,

UPPER(Prof_Name) AS Uppercase,

LOWER(Prof_Name) AS Lowercase

FROM Professors;

For Students:

SELECT Student_Name,

LENGTH(Student_Name) AS Length,

SUBSTR(Student_Name, 1, 5) AS Substr,

INSTR(Student_Name, 'a') AS Pos,

UPPER(Student_Name) AS Uppercase,

LOWER(Student_Name) AS Lowercase

FROM Students;

OUTPUT:

```
SQL> SELECT Prof_Name,
  2         LENGTH(Prof_Name) AS Length,
  3         SUBSTR(Prof_Name, 1, 5) AS Substr,
  4         INSTR(Prof_Name, 'a') AS Pos,
  5         UPPER(Prof_Name) AS Uppercase,
  6         LOWER(Prof_Name) AS Lowercase
  7  FROM Professors;

PROF_NAME                 LENGTH SUBSTR      POS UPPERCASE                 LOWERCASE
------------------------- ------ ---------- ---- ------------------------- ------------------
Dr. Meera Nair                14 Dr. M         9 DR. MEERA NAIR            dr. meera nair
Dr. Arjun Rao                 13 Dr. A        12 DR. ARJUN RAO             dr. arjun rao
Dr. Kavita Singh              16 Dr. K         6 DR. KAVITA SINGH          dr. kavita singh
Dr. Raj Malhotra              16 Dr. R         6 DR. RAJ MALHOTRA          dr. raj malhotra
```

```
SQL> SELECT Student_Name,
  2  LENGTH(Student_Name) AS Length,
  3  SUBSTR(Student_Name, 1, 5) AS Substr,
  4  INSTR(Student_Name, 'a') AS Pos,
  5  UPPER(Student_Name) AS Uppercase,
  6  LOWER(Student_Name) AS Lowercase
  7  FROM Students;

STUDENT_NAME              LENGTH SUBSTR      POS UPPERCASE                 LOWERCASE
------------------------- ------ ---------- ---- ------------------------- ---------------
Anjali Sharma                 13 Anjal         4 ANJALI SHARMA             anjali sharma
 Ravi Kumar                   11  Ravi         3  RAVI KUMAR                ravi kumar
 Nisha Verma                  12  Nish         6  NISHA VERMA               nisha verma
Aman Sheikh                   11 Aman          3 AMAN SHEIKH               aman sheikh
```

3. Use conversion functions on DOB.

CODE:

SELECT Student_ID, Student_Name, DOB,

TO_CHAR(DOB, 'DD-MM-YYYY') AS TO_CHAR,

TO_CHAR(DOB, 'Month') AS Month_Name

FROM Students;

SELECT TO_DATE('17-08-2004', 'DD-MM-YYYY') AS TO_DATE FROM DUAL;

OUTPUT:

```
SQL> SELECT Student_ID, Student_Name, DOB,
  2  TO_CHAR(DOB, 'DD-MM-YYYY') AS TO_CHAR,
  3  TO_CHAR(DOB, 'Month') AS Month_Name
  4  FROM Students;

STUDEN STUDENT_NAME              DOB       TO_CHAR    MONTH_NAME
------ ------------------------- --------- ---------- -------------------------
S0001  Anjali Sharma             22-AUG-03 22-08-2003 August
S0002   Ravi Kumar               28-FEB-03 28-02-2003 February
S0003   Nisha Verma              13-MAY-03 13-05-2003 May
S0004  Aman Sheikh               02-NOV-02 02-11-2002 November
```

```
SQL> SELECT TO_DATE('17-08-2004', 'DD-MM-YYYY') AS TO_DATE FROM DUAL;

TO_DATE
-------------------------
17-AUG-04

SQL> |
```

4. Count total number of students.

CODE:

SELECT COUNT(*) AS Total_Students FROM Students;

OUTPUT:

```
SQL> SELECT COUNT(*) AS Total_Students FROM Students;

TOTAL_STUDENTS
--------------
             4
```

5. Find max and min marks in Enrollments as max_marks, min_marks.

CODE:

SELECT MAX(Marks) AS Max_Marks,

MIN(Marks) AS Min_Marks

FROM Enrollments;

OUTPUT:

```
SQL> SELECT MAX(Marks) AS Max_Marks,
  2  MIN(Marks) AS Min_Marks
  3  FROM Enrollments;

 MAX_MARKS  MIN_MARKS
---------- ----------
        93       68.5
```

6. Count number of students with marks over 75.

CODE:

SELECT COUNT(*) AS Students_Above_75

FROM Enrollments

WHERE Marks > 75;

OUTPUT:

```
SQL> SELECT COUNT(*) AS Students_Above_75
  2  FROM Enrollments
  3  WHERE Marks > 75;

STUDENTS_ABOVE_75
-----------------
                4
```

4.4 Date Functions

1. List student names and their day of birth.

CODE:

SELECT Student_Name,

TO_CHAR(DOB, 'Day') AS Day_Of_Birth

FROM Students;

OUTPUT:

```
SQL> SELECT Student_Name,
  2  TO_CHAR(DOB, 'Day') AS Day_Of_Birth
  3  FROM Students;

STUDENT_NAME                 DAY_OF_BIRTH
------------------------- -------------------------
Anjali Sharma                Wednesday
 Ravi Kumar                  Wednesday
 Nisha Verma                 Sunday
Aman Sheikh                  Thursday
```

2. Format DOBs in 'DD-Month-YYYY' format.

CODE:

SELECT Student_Name,

TO_CHAR(DOB, 'DD-Month-YYYY') AS DOB

FROM Students;

OUTPUT:

```
SQL> SELECT Student_Name,
  2  TO_CHAR(DOB, 'DD-Month-YYYY') AS DOB
  3  FROM Students;

STUDENT_NAME                 DOB
------------------------- -------------------------
Anjali Sharma                14-May      -2003
 Ravi Kumar                  20-November -2002
 Nisha Verma                 02-February -2003
Aman Sheikh                  25-July     -2002
```

3. Show DOBs in 'DD-MM-YY' format.

CODE:

SELECT Student_Name,

TO_CHAR(DOB, 'DD-MM-YY') AS DOB

FROM Students;

OUTPUT:

```
SQL> SELECT Student_Name,
  2  TO_CHAR(DOB, 'DD-MM-YY') AS DOB
  3  FROM Students;

STUDENT_NAME                 DOB
------------------------- --------
Anjali Sharma                14-05-03
 Ravi Kumar                  20-11-02
 Nisha Verma                 02-02-03
Aman Sheikh                  25-07-02
```

4. Add 100 days to all DOBs.

CODE:

UPDATE Students SET DOB = DOB + 100;

OUTPUT:

```
SQL> UPDATE Students SET DOB = DOB + 100;

4 rows updated.

SQL> SELECT * FROM Students;

STUDEN STUDENT_NAME                 DEPT DOB
------ ------------------------- ---- ---------
S0001  Anjali Sharma               D01  22-AUG-03
S0002   Ravi Kumar                 D02  28-FEB-03
S0003   Nisha Verma                D03  13-MAY-03
S0004  Aman Sheikh                 D01  02-NOV-02
```

5. List students born in May.

CODE:

SELECT Student_Name, DOB

FROM Students

WHERE TO_CHAR(DOB, 'Mon') = 'May';

OUTPUT:

```
SQL> SELECT Student_Name, DOB
  2  FROM Students
  3  WHERE TO_CHAR(DOB, 'Mon') = 'May';

STUDENT_NAME                   DOB
------------------------- ---------
 Nisha Verma                13-MAY-03
```

6. List students born between 2002 and 2003.

CODE:

SELECT Student_Name, DOB

FROM Students

WHERE TO_CHAR(DOB, 'yyyy') BETWEEN '2002' AND '2003';

OUTPUT:

```
SQL> SELECT Student_Name, DOB
  2  FROM Students
  3  WHERE TO_CHAR(DOB, 'yyyy') BETWEEN '2002' AND '2003';

STUDENT_NAME                   DOB
------------------------- ---------
Anjali Sharma              22-AUG-03
 Ravi Kumar                28-FEB-03
 Nisha Verma               13-MAY-03
Aman Sheikh                02-NOV-02
```

4.5 Set Operators

Create Top_Courses table:

```
CREATE TABLE Top_Courses (
   Course_Name VARCHAR2(100),
   Dept_Name VARCHAR2(50)
);
```

Insert some course-department pairs.

```
INSERT INTO Top_Courses VALUES ('Data Structures', 'Computer Science');
INSERT INTO Top_Courses VALUES ('Circuit Theory', 'Electrical Engg.');
INSERT INTO Top_Courses VALUES ('Thermodynamics', 'Mechanical Engg.');
INSERT INTO Top_Courses VALUES ('Artificial Intelligence', 'Computer Science');
INSERT INTO Top_Courses VALUES (' Concrete Technology', 'Civil Engg');
```

OUTPUT:

```
SQL> SELECT * FROM Top_Courses;

COURSE_NAME                    DEPT_NAME
------------------------------ ------------------
Data Structures                Computer Science
Circuit Theory                 Electrical Engg.
Thermodynamics                 Mechanical Engg.
Artificial Intelligence        Computer Science
Concrete Technology            Civil Engg
```

1. Show unique course names from both Courses and Top_Courses.

CODE:

```
SELECT Course_Name FROM Courses
UNION
SELECT Course_Name FROM Top_Courses;
```

OUTPUT

```
SQL> SELECT Course_Name FROM Courses
  2  UNION
  3  SELECT Course_Name FROM Top_Courses;

COURSE_NAME
------------------------
Structural Analysis
Data Structures
Operating Systems
Circuit Theory
Thermodynamics
Artificial Intelligence
Concrete Technology

7 rows selected.
```

2. Show common courses between Courses and Top_Courses.

CODE:

SELECT Course_Name FROM Courses

INTERSECT

SELECT Course_Name FROM Top_Courses;

OUTPUT

```
SQL> SELECT Course_Name FROM Courses
  2  INTERSECT
  3  SELECT Course_Name FROM Top_Courses;

COURSE_NAME
------------------------
Data Structures
Circuit Theory
Thermodynamics
```

3. Show top courses that are not in current courses.

CODE:

SELECT Course_Name FROM Top_Courses

MINUS

SELECT Course_Name FROM Courses;

OUTPUT

```
SQL> SELECT Course_Name FROM Top_Courses
  2  MINUS
  3  SELECT Course_Name FROM Courses;

COURSE_NAME
-------------------------
Artificial Intelligence
Concrete Technology
```

4.Show union all of both.

CODE:

SELECT Course_Name FROM Courses

UNION ALL

SELECT Course_Name FROM Top_Courses;

OUTPUT:

```
SQL> SELECT Course_Name FROM Courses
  2  UNION ALL
  3  SELECT Course_Name FROM Top_Courses;

COURSE_NAME
-------------------------
Structural Analysis
Data Structures
Operating Systems
Circuit Theory
Thermodynamics
Data Structures
Circuit Theory
Thermodynamics
Artificial Intelligence
Concrete Technology

10 rows selected.
```

4.6 Sorting

1. Sort professors by descending experience.

CODE:

SELECT * FROM Professors

ORDER BY Experience_Years DESC;

OUTPUT:

```
SQL> SELECT * FROM Professors
  2  ORDER BY Experience_Years DESC;

PROF_ PROF_NAME                 DEPT EXPERIENCE_YEARS
----- ------------------------- ---- ----------------
P1004 Dr. Raj Malhotra          D01                15
P1001 Dr. Meera Nair            D01                12
P1002 Dr. Arjun Rao             D01                 9
P1003 Dr. Kavita Singh          D01                 7
```

4.7 Group By, Having

1. Number of students per department (only those >1).

CODE:

SELECT Dept_Id, COUNT(Student_ID)

FROM Students

GROUP BY Dept_ID

HAVING COUNT(Student_ID) > 1;

OUTPUT:

```
SQL> SELECT Dept_Id, COUNT(Student_ID)
  2   FROM Students
  3  GROUP BY Dept_ID
  4  HAVING COUNT(Student_ID) > 1;

DEPT COUNT(STUDENT_ID)
---- -----------------
D01                  2
```

2. Departments with avg classroom count >5.

CODE:

SELECT Dept_Name, AVG(Number_of_Classrooms) AS AVG_CLASSROOMS

FROM Departments

GROUP BY Dept_Name

HAVING AVG(Number_of_Classrooms) > 5;

OUTPUT:

```
SQL> SELECT Dept_Name, AVG(Number_of_Classrooms) AS AVG_CLASSROOMS
  2  FROM Departments
  3  GROUP BY Dept_Name
  4  HAVING AVG(Number_of_Classrooms) > 5;

DEPT_NAME                                          AVG_CLASSROOMS
-------------------------------------------------- --------------
Computer Science                                               10
Electrical Engg.                                                8
Mechanical Engg.                                                6
```

3. Courses taught by professors with more than 1 course.

CODE:

```
SELECT c.Course_Name, p.Prof_Name
FROM Courses c, Professors p
WHERE c.Prof_ID = p.Prof_ID
AND c.Prof_ID IN (
SELECT Prof_ID FROM Courses
GROUP BY Prof_ID
HAVING COUNT(Course_ID) > 1 );
```

OUTPUT:

```
SQL> SELECT c.Course_Name, p.Prof_Name
  2  FROM Courses c, Professors p
  3  WHERE c.Prof_ID = p.Prof_ID
  4  AND c.Prof_ID IN (
  5  SELECT Prof_ID FROM Courses
  6  GROUP BY Prof_ID
  7  HAVING COUNT(Course_ID) > 1 );

COURSE_NAME                    PROF_NAME
------------------------------ ------------------------------
Structural Analysis            Dr. Raj Malhotra
Thermodynamics                 Dr. Raj Malhotra
```