**DBMS LAb :**

28/02/24

Table Students :

CREATE TABLE Software (

    PNAME VARCHAR2(20) NOT NULL,

    TITLE VARCHAR2(25) NOT NULL,

    DEV VARCHAR2(10) NOT NULL,

    SCOST NUMBER(10,2) NOT NULL,

    DCOST NUMBER(10,2) NOT NULL,

    SOLD NUMBER(4),

    CONSTRAINT pk\_Software PRIMARY KEY (PNAME, TITLE)

);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Anand', 'Pharachutes', 'Basic', 399.95, 6000, 43);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Anand', 'Video Titling Pack', '.NET', 7500.00, 16000, 9);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Juliana', 'Inventory Control', 'Cobol', 3000, 3500, 0);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Tulasi', 'Payroll software', 'Dbase', 9000, 20000, 7);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Mary', 'Financial Acc S/W', 'Oracle', 18000, 85000, 4);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Mary', 'Code Generation', 'C', 4500, 20000, 23);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Patrick', 'Read Me', 'C++', 300, 1200, 84);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Qadir', 'Bombs Away', 'Assembly', 750, 5000, 11);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Qadir', 'Vaccines', 'C', 1900, 3400, 21);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Ramesh', 'Hotel Management', 'Dbase', 12000, 35000, 4);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Ramesh', 'Dead Lee', '.NET', 599.95, 4500, 73);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Remitha', 'PC Utilities', 'C', 725, 5000, 51);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Remitha', 'TSR Help software', 'Assembly', 2500, 6000, 6);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Revathi', 'Hospital Management', '.NET', 1100, 75000, 2);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Revathi', 'Quiz Master', 'Basic', 3200, 2100, 15);

INSERT INTO Software (PNAME, TITLE, DEV, SCOST, DCOST, SOLD) VALUES ('Vijaya', 'ISR Editor', 'C', 900, 700, 6);

Table Programmer:

CREATE TABLE Programmer (

    PNAME VARCHAR2(20) NOT NULL,

    DOB DATE NOT NULL,

    DOJ DATE NOT NULL,

    GENDER CHAR(1) NOT NULL,

    PROF1 VARCHAR2(10),

    PROF2 VARCHAR2(10),

    SALARY NUMBER(7,2) NOT NULL,

    CONSTRAINT pk\_Programmer PRIMARY KEY (PNAME)

);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Anand', ’21-apr-66’,’21-apr-92', 'M', '.NET', 'Basic', 3200);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Altaf',   ’02-jul-64',   ’13-nov-90', 'M', 'Fortran', 'Cobol', 2800);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Juliana',   '31-jan-68', '21-apr-90',  'F', 'Cobol', 'Dbase', 3000);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Tulasi',   '30-oct-68', '02-jan-92',  'F', 'C', 'Dbase', 2900);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Mary',   '24-jun-70', '01-feb-91', 'F', 'C++', 'Oracle', 4500);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Nelson',   '14-sep-65', '11-oct-89',  'M', 'Cobol', 'Dbase', 2500);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Patrick',   '11-nov-65',   '21-apr-90',  'M', '.NET', 'Fortran', 2800);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Qadir',   '30-aug-65',   '21-apr-91', 'M', 'Assembly', 'C', 3000);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Ramesh',   '08-may-67', '28-feb-91', 'M', '.NET', 'Dbase', 3200);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Rebacca',   '04-jan-67', '01-dec-90',  'F', 'Basic', 'Cobol', 2500);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Remitha',   '19-apr-70',   '20-apr-93',  'F', 'C', 'Assembly', 3600);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Revathi',   '02-dec-69',   '02-jan-92', 'F', '.NET', 'Basic', 3700);

INSERT INTO Programmer (PNAME, DOB, DOJ, GENDER, PROF1, PROF2, SALARY)

VALUES ('Vijaya',   '11-dec-65',  '02-may-92',  'F', 'Foxpro', 'C', 3500);

Table Studies:

CREATE TABLE Studies (

    PNAME VARCHAR2(20) NOT NULL,

    SPLACE VARCHAR2(20) NOT NULL,

    COURSE VARCHAR2(20) NOT NULL,

    COST NUMBER(5),

    CONSTRAINT pk\_Studies PRIMARY KEY (PNAME, SPLACE, COURSE)

);

-- Inserting data for Anand

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Anand', 'Sabhari', 'PGDCA', 4500);

-- Inserting data for Altaf

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Altaf', 'CBIT', 'DCA', 7200);

-- Inserting data for Juliana

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Juliana', 'BITS', 'MCA', 22000);

-- Inserting data for Tulasi

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Tulasi', 'Pragathi', 'DCP', 5000);

-- Inserting data for Mary

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Mary', 'Sabhari', 'PGDCA', 4500);

-- Inserting data for Nelson

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Nelson', 'Pragathi', 'DAP', 6200);

-- Inserting data for Patrick

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Patrick', 'Pragathi', 'DCAP', 5200);

-- Inserting data for Qadir

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Qadir', 'Apple', 'HDCP', 14000);

-- Inserting data for Ramesh

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Ramesh', 'Sabhari', 'PGDCA', 4500);

-- Inserting data for Rebacca

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Rebacca', 'Brilliant', 'DCA&P', 11000);

-- Inserting data for Remitha

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Remitha', 'BDPS', 'DCS', 5000);

-- Inserting data for Revathi

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Revathi', 'Sabhari', 'DAP', 5000);

-- Inserting data for Vijaya

INSERT INTO Studies (PNAME, SPLACE, COURSE, COST) VALUES ('Vijaya', 'BDPS', 'DCA', 48000);

—1. Find out the selling cost for the software(s) developed in .NET

SELECT PNAME, TITLE, SCOST

FROM Software

WHERE DEV = '.NET';

—9. Display the details of software developed by Ramesh

SELECT \*

FROM Software

WHERE PNAME = 'Ramesh';

—3. Display the names of programmers who have done the DAP course

SELECT DISTINCT P.PNAME

FROM Programmer P

JOIN Studies S ON P.PNAME = S.PNAME

WHERE S.COURSE = 'DAP';

—11. Display the details of softwares whose sales have been crossed 2000 mark

SELECT \*

FROM Software

WHERE SCOST > 2000;

—19. Display the details of programmers knowing C

SELECT \*

FROM Programmer

WHERE PROF1 = 'C' OR PROF2 = 'C';

—29. How many people draw 2000-4000

SELECT COUNT(\*) AS NumberOfPeople

FROM Programmer

WHERE SALARY BETWEEN 2000 AND 4000;

—30.Display the details of programmers who don’t know Fortran, Cobol or .NET

SELECT \*

FROM Programmer

WHERE PROF1 NOT IN ('Fortran', 'Cobol', '.NET') AND PROF2 NOT IN ('Fortran', 'Cobol', '.NET');

—35.Identify and include the keys (Primary Keys & Foreign Keys) in the tables

**Programmer Table:**

* Primary Key (PNAME):

ALTER TABLE Programmer

ADD CONSTRAINT PK\_Programmer PRIMARY KEY (PNAME);

**Studies Table:**

* Primary Key (PNAME, COURSE):
* Foreign Key (PNAME referencing Programmer table):

ALTER TABLE Studies

ADD CONSTRAINT PK\_Studies PRIMARY KEY (PNAME, COURSE);

ALTER TABLE Studies

ADD CONSTRAINT FK\_Studies\_Programmer

FOREIGN KEY (PNAME) REFERENCES Programmer(PNAME);

**Software Table:**

* Primary Key (PNAME, TITLE):
* Foreign Key (PNAME referencing Programmer table):

ALTER TABLE Software

ADD CONSTRAINT PK\_Software PRIMARY KEY (PNAME, TITLE);

ALTER TABLE Software

ADD CONSTRAINT FK\_Software\_Programmer

FOREIGN KEY (PNAME) REFERENCES Programmer(PNAME);

PART-2

24.Display the names of male and female programmers

SELECT PNAME, GENDER

FROM Programmer

ORDER BY GENDER, PNAME;

25.Display the programmers name and their softwares

SELECT P.PNAME, S.TITLE

FROM Programmer P

JOIN Software S ON P.PNAME = S.PNAME

ORDER BY P.PNAME, S.TITLE;

PART-3

36.List the softwares which have not been sold so far

SELECT TITLE

FROM Software

WHERE SOLD = 0;

40. Display the names of the programmers whose names contain 2 occurrences of the letter ‘a’

SELECT PNAME

FROM Programmer

WHERE LENGTH(PNAME) - LENGTH(REPLACE(PNAME, 'a', '')) = 2;

49. Display the names of the softwares whose name contain more than one word

SELECT TITLE

FROM Software

WHERE CHARINDEX(' ', TITLE) > 0;

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Version:0.9 StartHTML:0000000105 EndHTML:0000124636 StartFragment:0000000141 EndFragment:0000124596

Queries on Operators

1. Find out the selling cost for the software(s) developed in .NET

2. Display the names and ages of programmers

3. Display the names of programmers who have done the DAP course

4. Display the details of software developed by Ramesh

5. Display the details of softwareâ€™s whose sales have been crossed 2000 mark

6. How many softwareâ€™s were developed in dbase

7. Display the details of programmers knowing C

8. List the people draw 2000-4000

9. Display the details of programmers who donâ€™t know Fortran, Cobol or .NET

10. Identify and include the keys (Primary Keys & Foreign Keys) in the tables

11. Display the names of male and female programmers

12. Display the programmers name and their softwareâ€™s

13. List the softwareâ€™s which have not been sold so far

14. Find out the cost of the software developed by Mary

15. Display the institute name from studies table without duplicates

16. List the different courses are mentioned in the studies table

17. Display the names of the programmers whose names contain 2 occurrences of the letter â€˜aâ€™

18. Display the names of the softwareâ€™s whose name contain more than one word

ANSWERS :

Here are the  SQL queries for the provided questions using appropriate operations:

1. Find out the selling cost for the software(s) developed in .NET

SELECT TITLE, SCOST

FROM Software

WHERE DEV\_D = '.NET';

2. Display the names and ages of programmers

Assuming age is calculated from the DOB:

SELECT PNAME, TRUNC(MONTHS\_BETWEEN(SYSDATE, DOB) / 12) AS AGE

FROM Programmer;

3. Display the names of programmers who have done the DAP course

SELECT PNAME

FROM Studies

WHERE COURSE = 'DAP';

4. Display the details of software developed by Ramesh

SELECT \*

FROM Software

WHERE PNAME = 'Ramesh';

5. Display the details of softwareâ€™s whose sales have been crossed 2000 mark

SELECT \*FROM Software

WHERE SOLD > 2000;

6. How many softwareâ€™s were developed in Dbase

SELECT COUNT(\*)

FROM Software

WHERE DEV\_D = 'Dbase';

7. Display the details of programmers knowing C

SELECT \*

FROM Programmer

WHERE PNAME IN (SELECT PNAME FROM Studies WHERE COURSE = 'C');

8. List the people draw 2000-4000

SELECT PNAME

FROM Software

WHERE SCOST BETWEEN 2000 AND 4000;

9. Display the details of programmers who donâ€™t know Fortran, Cobol or .NET

SELECT \*

FROM Programmer

WHERE PNAME NOT IN (

SELECT PNAME

FROM Studies

WHERE COURSE IN ('Fortran', 'Cobol', '.NET')

);

10. Identify and include the keys (Primary Keys & Foreign Keys) in the tables

Primary Key:

- `Programmer`: PNAME

- `Studies`: PNAME (Foreign Key to Programmer), Combination of PNAME and COURSE (Composite Primary Key)

- `Software`: No primary key provided in the table schema

Foreign Key:

- `Studies`: PNAME (References Programmer.PNAME)

- `Software`: PNAME (References Programmer.PNAME)

11. Display the names of male and female programmers

SELECT PNAME, GENDER

FROM Programmer;

12. Display the programmers name and their softwareâ€™s

SELECT p.PNAME, s.TITLE

FROM Programmer p

JOIN Software s ON p.PNAME = s.PNAME;

13. List the softwareâ€™s which have not been sold so far

SELECT \*

FROM Software

WHERE SOLD IS NULL;

14. Find out the cost of the software developed by Mary

SELECT TITLE, SCOST

FROM Software

WHERE PNAME = 'Mary';

15. Display the institute name from studies table without duplicates

SELECT DISTINCT SPLACE

FROM Studies;

16. List the different courses mentioned in the studies table

SELECT DISTINCT COURSE

FROM Studies;

17. Display the names of the programmers whose names contain 2 occurrences of the letter â€˜aâ€™

SELECT PNAME

FROM Programmer

WHERE REGEXP\_COUNT(PNAME, 'a') = 2;

18. Display the names of the softwareâ€™s whose name contains more than one word

SELECT TITLE

FROM Software

WHERE REGEXP\_LIKE(TITLE, '(\S+\s){1,}\S+');

These queries should help you to retrieve the required information from the tables. Adjust them as needed based on your specific database schema and data.

Queries on Aggregate Functions:

Built-In functions

(Aggregate Functions) (any 10 queries)

1. What is the highest number of copies sold by a software(aggregate)

2. Display the lowest course fee (aggregate)

3. How many programmers have done the PGDCA course (aggregate)4. How much revenue has been earned through the sale of software developed in C (aggregate)

5. How many programmers have studied at Sabhari (aggregate)

6. What is the price of the costlier software developed in Basic

7. How many programmers studied in Pragathi (aggregate)

8. How many programmers paid 5000-10000 for their course (aggregate)

9. What is the average course fee (aggregate)

10. How may programmers know either Cobol or .NET (aggregate)

ANSWERS:

Here are the  queries for the provided questions using aggregate functions:

1. What is the highest number of copies sold by a software?

SELECT MAX(SOLD) AS "Highest Copies Sold"

FROM Software;

2. Display the lowest course fee.

SELECT MIN(COST) AS "Lowest Course Fee"

FROM Studies;

3. How many programmers have done the PGDCA course?

SELECT COUNT(\*) AS "Programmers Count"

FROM Studies

WHERE COURSE = 'PGDCA';

4. How much revenue has been earned through the sale of software developed in C?

SELECT SUM(SCOST \* SOLD) AS "Total Revenue"

FROM Software

WHERE DEV\_D = 'C';

5. How many programmers have studied at Sabhari?

SELECT COUNT(DISTINCT PNAME) AS "Programmers Count"

FROM Studies

WHERE SPLACE = 'Sabhari';

6. What is the price of the costlier software developed in Basic?

SELECT MAX(SCOST) AS "Costliest Software Price"

FROM Software

WHERE DEV\_D = 'Basic';

7. How many programmers studied in Pragathi?

SELECT COUNT(DISTINCT PNAME) AS "Programmers Count"

FROM Studies

WHERE SPLACE = 'Pragathi';

8. How many programmers paid 5000-10000 for their course?

SELECT COUNT(\*) AS "Programmers Count"

FROM Studies

WHERE COST BETWEEN 5000 AND 10000;

9. What is the average course fee?

SELECT AVG(COST) AS "Average Course Fee"

FROM Studies;

10. How many programmers know either Cobol or .NET?

SELECT COUNT(\*) AS "Programmers Count"

FROM Studies

WHERE COURSE IN ('Cobol', '.NET');

These queries should provide you with the required aggregate information from the tables. Adjust them as needed based on your specific database schema and data.

String functions (All queries)

1. Display the number of people born in each year (conversion and aggregation)

2. Display the number of people joined in each year (conversion and aggregation)

3. Display the number of people born in each month (conversion and aggregation, string)

4. Display the number of people joined in each month (conversion and aggregation, string)

5. Display the names of programmers whose names contains up to 5 characters

6. What is the length of the shortest name in the programmers table

ANSWERS:

Here are the  queries for the provided questions:

1. Display the number of people born in each year:

SELECT EXTRACT(YEAR FROM DOB) AS Birth\_Year, COUNT(\*) AS Num\_People

FROM Programmer

GROUP BY EXTRACT(YEAR FROM DOB)

ORDER BY Birth\_Year;

2. Display the number of people joined in each year:

SELECT EXTRACT(YEAR FROM DOJ) AS Join\_Year, COUNT(\*) AS Num\_People

FROM Programmer

GROUP BY EXTRACT(YEAR FROM DOJ)

ORDER BY Join\_Year;

3. Display the number of people born in each month (conversion and aggregation, string):

SELECT TO\_CHAR(DOB, 'Month') AS Birth\_Month, COUNT(\*) AS Num\_People

FROM Programmer

GROUP BY TO\_CHAR(DOB, 'Month')

ORDER BY TO\_DATE(Birth\_Month, 'Month');

4. Display the number of people joined in each month (conversion and aggregation, string):

SELECT TO\_CHAR(DOJ, 'Month') AS Join\_Month, COUNT(\*) AS Num\_People

FROM Programmer

GROUP BY TO\_CHAR(DOJ, 'Month')

ORDER BY TO\_DATE(Join\_Month, 'Month');

5. Display the names of programmers whose names contain up to 5 characters:

SELECT PNAME

FROM Programmer

WHERE LENGTH(PNAME) <= 5;

6. What is the length of the shortest name in the programmers table:

SELECT MIN(LENGTH(PNAME)) AS Shortest\_Name\_Length

FROM Programmer;

These queries should provide you with the desired information based on the Programmer table. Adjust them as needed based on your specific database schema and data.

Date functions (All queries)

1. How many female programmers knowing Cobol have more than 2 years experience

2. What is the average development cost of a software developed in Cobol

3. Display the name, gender, dob (dd/ mmm/ yy format), doj (dd/ mmm/ yy format) for all programmers without using the conversion function

4. Who are the programmers who were born on the last day of the month

5. Display the name, job, dob, doj of those months of birth and month of joining are same

ANSWERS:

Here are the  queries for the provided questions:

1. How many female programmers knowing Cobol have more than 2 years of experience:

SELECT COUNT(\*) AS Female\_Cobol\_Programmers

FROM Programmer

WHERE GENDER = 'F'

AND PNAME IN (

SELECT PNAME

FROM Studies

WHERE COURSE = 'Cobol'

)

AND MONTHS\_BETWEEN(SYSDATE, DOJ) > 24;

2. What is the average development cost of a software developed in Cobol:

SELECT AVG(DCOST) AS Avg\_Development\_Cost

FROM Software

WHERE DEV\_D = 'Cobol';

3. Display the name, gender, dob (dd/ mmm/ yy format), doj (dd/ mmm/ yy format) for all programmers without using the conversion function:

SELECT PNAME, GENDER, TO\_CHAR(DOB, 'DD/MON/YY') AS DOB\_Format, TO\_CHAR(DOJ, 'DD/MON/YY') AS DOJ\_Format

FROM Programmer;

4. Who are the programmers who were born on the last day of the month:

SELECT PNAME

FROM Programmer

WHERE EXTRACT(DAY FROM DOB) = (

SELECT EXTRACT(DAY FROM LAST\_DAY(DOB))

FROM DUAL

);

5. Display the name, job, dob, doj of those months of birth and month of joining are the same:

SELECT PNAME, PJOB, DOB, DOJ

FROM Programmer

WHERE EXTRACT(MONTH FROM DOB) = EXTRACT(MONTH FROM DOJ);

These queries should provide you with the desired information based on the Programmer table. Adjust them as needed based on your specific database schema and data.

Numeric (All queries)

1.Find out the number of copies, which should be sold in-order to recover the development cost of each software(Math function)

2. Display the details of the softwareâ€™s for which development cost have been recovered

3.Display the names and ages of programmers (numeric and date)

4. How many female programmers knowing COBOL have more than 2 years experience?

ANSWERS :

Here are the  queries for the provided questions:

1. Find out the number of copies, which should be sold in order to recover the development cost of each software (Math function):

SELECT TITLE, CEIL(DCOST / SCOST) AS Copies\_For\_Recovery

FROM Software;

2. Display the details of the softwareâ€™s for which development cost has been recovered:

SELECT \*

FROM Software

WHERE SOLD >= CEIL(DCOST / SCOST);

3. Display the names and ages of programmers (numeric and date):

SELECT PNAME, TRUNC(MONTHS\_BETWEEN(SYSDATE, DOB) / 12) AS AGE

FROM Programmer;

4. How many female programmers knowing COBOL have more than 2 years of experience?

SELECT COUNT(\*) AS Female\_Cobol\_Programmers

FROM Programmer

WHERE GENDER = 'F'

AND PNAME IN (

SELECT PNAME

FROM Studies

WHERE COURSE = 'COBOL'

)

AND MONTHS\_BETWEEN(SYSDATE, DOJ) > 24;

These queries should provide you with the desired information based on the provided tables. Adjust them as needed based on your specific database schema and data.

Conversion (All queries)

Display the names and date of birth of all programmers born in January (conversion)

Who are the programmers who celebrate their birthdays in the current month (conversion)

ANSWERS:

Here are the  queries for the provided questions:

1. Display the names and date of birth of all programmers born in January (conversion):

SELECT PNAME, TO\_CHAR(DOB, 'DD-MON-YYYY') AS Date\_of\_Birth

FROM Programmer

WHERE TO\_CHAR(DOB, 'MM') = '01';

2. Who are the programmers who celebrate their birthdays in the current month (conversion):

SELECT PNAME

FROM Programmer

WHERE TO\_CHAR(DOB, 'MM') = TO\_CHAR(SYSDATE, 'MM');

These queries should provide you with the desired information based on the Programmer table. Adjust them as needed based on your specific database schema and data.

Queries on Order by and group by(any 15 queries)

1. Calculate the experience in years for each programmer and display along with their names in descending order (order by)

2. Display language wise count of prof1 ( aggregate, groupBy)

3. Display language wise count of prof2 ( aggregate, groupBy)

4. Display the number of people in each salary group ( aggregate, groupBy)

5. Display the number of people studied in each institute ( aggregate, groupBy)

6. Display the number of people studied in each course ( aggregate,groupBy)

7. Display the total development cost of softwareâ€™s developed in each language ( (aggregate,groupBy)

8. Display the selling cost of softwareâ€™s developed in each language ( aggregate,groupBy)

9. Display the costs of softwareâ€™s developed by each programmer language wise(aggregate,groupBy)

10. Display the number of softwares sold by each programmer aggregate,groupBy)11. Display the sales cost of the softwareâ€™s developed by each programmer (aggregate,groupBy)

12. Display the sales cost of the softwareâ€™s developed by each programmer language wise (aggregate,groupBy)

13. Display the language name with average development cost, average selling cost and average price per copy (aggregate,groupBy)

14. Display the programmers name, costliest software, cheapest software developed by him/ her (aggregate,groupBy)

15. Display each institute name with number of courses, average cost per course (aggregate,groupBy)

ANSWERS:

Here are the  queries for the provided questions:

1. Calculate the experience in years for each programmer and display along with their names in descending order (order by):

SELECT PNAME, TRUNC(MONTHS\_BETWEEN(SYSDATE, DOJ) / 12) AS Experience\_Years

FROM Programmer

ORDER BY Experience\_Years DESC;

2. Display language-wise count of prof1 (aggregate, group by):

SELECT PROF1, COUNT(\*) AS Language\_Count

FROM Programmer

GROUP BY PROF1;

3. Display language-wise count of prof2 (aggregate, group by):

SELECT PROF2, COUNT(\*) AS Language\_Count

FROM Programmer

GROUP BY PROF2;

4. Display the number of people in each salary group (aggregate, group by):

SELECT FLOOR(SALARY / 1000) \* 1000 AS Salary\_Group, COUNT(\*) AS People\_Count

FROM Programmer

GROUP BY FLOOR(SALARY / 1000) \* 1000;

5. Display the number of people studied in each institute (aggregate, group by):

SELECT SPLACE, COUNT(\*) AS People\_Count

FROM Studies

GROUP BY SPLACE;

6. Display the number of people studied in each course (aggregate, group by):

SELECT COURSE, COUNT(\*) AS People\_Count

FROM Studies

GROUP BY COURSE;

7. Display the total development cost of softwareâ€™s developed in each language (aggregate, group by):

SELECT DEV\_D, SUM(DCOST) AS Total\_Development\_Cost

FROM Software

GROUP BY DEV\_D;

8. Display the selling cost of softwareâ€™s developed in each language (aggregate, group by):

SELECT DEV\_D, SUM(SCOST) AS Total\_Selling\_Cost

FROM Software

GROUP BY DEV\_D;

9. Display the costs of softwareâ€™s developed by each programmer language-wise (aggregate, group by):

SELECT PNAME, DEV\_D, SUM(DCOST) AS Total\_Development\_Cost

FROM Software

GROUP BY PNAME, DEV\_D;

10. Display the number of softwares sold by each programmer (aggregate, group by):

SELECT PNAME, COUNT(\*) AS Software\_Count

FROM Software

GROUP BY PNAME;

11. Display the sales cost of the softwareâ€™s developed by each programmer (aggregate, group by):

SELECT PNAME, SUM(SCOST) AS Total\_Selling\_Cost

FROM Software

GROUP BY PNAME;

12. Display the sales cost of the softwareâ€™s developed by each programmer language-wise (aggregate, group by):

SELECT PNAME, DEV\_D, SUM(SCOST) AS Total\_Selling\_Cost

FROM Software

GROUP BY PNAME, DEV\_D;

13. Display the language name with average development cost, average selling cost, and average price per copy (aggregate, group by):

SELECT DEV\_D, AVG(DCOST) AS Avg\_Development\_Cost, AVG(SCOST) AS Avg\_Selling\_Cost, AVG(SCOST / SOLD) AS Avg\_Price\_Per\_Copy

FROM Software

GROUP BY DEV\_D;

14. Display the programmer's name, costliest software, cheapest software developed by him/her (aggregate, group by):

SELECT P.PNAME, MAX(S.SCOST) AS Costliest\_Software, MIN(S.SCOST) AS Cheapest\_Software

FROM Programmer P

JOIN Software S ON P.PNAME = S.PNAMEGROUP BY P.PNAME;

15. Display each institute name with the number of courses, average cost per course (aggregate, group by):

SELECT SPLACE, COUNT(\*) AS Num\_Courses, AVG(COST) AS Avg\_Cost\_Per\_Course

FROM Studies

GROUP BY SPLACE;

These queries should provide you with the desired information based on the provided tables. Adjust them as needed based on your specific database schema and data.

Queries on Nested queries(do any 15 queries)

1. Who is the highest paid C programmer

2. Who is the highest paid female Cobol programmer

3. Display the name of the highest paid programmers for each language (prof1)

4. Who is the least experienced programmer

5. Who is the most experienced male programmer knowing .NET

6. Which language does only one programmer know

7. Who is the above programmer

8. Who is the youngest programmer knowing dbase

9. Which institute has the most number of students

10. Which course has been done by most of the students

11. Display the names of the institute and course which has below average course fee

12. Which is the costliest course

13. Which institute conducts the costliest course

14. Which course has the below average number of students

15. Which institute conducts the above courses

ANSWERS:

Here are the  queries for the provided questions using nested queries:

1. Who is the highest paid C programmer:

SELECT PNAME

FROM Programmer

WHERE PJOB = 'C'

AND SALARY = (

SELECT MAX(SALARY)

FROM Programmer

WHERE PJOB = 'C'

);

2. Who is the highest paid female Cobol programmer:

SELECT PNAME

FROM Programmer

WHERE GENDER = 'F'

AND PJOB = 'Cobol'

AND SALARY = (

SELECT MAX(SALARY)

FROM Programmer

WHERE GENDER = 'F'

AND PJOB = 'Cobol'

);

3. Display the name of the highest paid programmers for each language (prof1):

SELECT PROF1, PNAME

FROM Programmer

WHERE (PROF1, SALARY) IN (

SELECT PROF1, MAX(SALARY)

FROM Programmer

GROUP BY PROF1

);

4. Who is the least experienced programmer:

SELECT PNAME

FROM Programmer

WHERE DOB = (

SELECT MIN(DOB)

FROM Programmer

);

5. Who is the most experienced male programmer knowing .NET:

SELECT PNAME

FROM Programmer

WHERE GENDER = 'M'

AND PROF1 = '.NET'

AND DOB = (

SELECT MIN(DOB)

FROM Programmer

WHERE GENDER = 'M'

AND PROF1 = '.NET'

);

6. Which language does only one programmer know:

SELECT PROF1 AS Language

FROM (

SELECT PROF1, COUNT(\*) AS Programmer\_Count

FROM Programmer

GROUP BY PROF1

) AS Language\_Count

WHERE Programmer\_Count = 1;

7. Who is the above programmer:

SELECT PNAME

FROM Programmer

WHERE PROF1 = (

SELECT PROF1

FROM (

SELECT PROF1, COUNT(\*) AS Programmer\_Count FROM Programmer

GROUP BY PROF1

) AS Language\_Count

WHERE Programmer\_Count = 1

);

8. Who is the youngest programmer knowing dbase:

SELECT PNAME

FROM Programmer

WHERE PJOB = 'dbase'

AND DOB = (

SELECT MAX(DOB)

FROM Programmer

WHERE PJOB = 'dbase'

);

9. Which institute has the most number of students:

SELECT SPLACE

FROM Studies

GROUP BY SPLACE

HAVING COUNT(\*) = (

SELECT MAX(Student\_Count)

FROM (

SELECT COUNT(\*) AS Student\_Count

FROM Studies

GROUP BY SPLACE

) AS Institute\_Student\_Count

);

10. Which course has been done by most of the students:

SELECT COURSE

FROM Studies

GROUP BY COURSE

HAVING COUNT(\*) = (

SELECT MAX(Student\_Count)

FROM (

SELECT COUNT(\*) AS Student\_Count

FROM Studies

GROUP BY COURSE

) AS Course\_Student\_Count

);

11. Display the names of the institute and course which has below average course fee:

SELECT SPLACE, COURSE

FROM Studies

WHERE COST < (

SELECT AVG(COST)

FROM Studies);

12. Which is the costliest course:

SELECT COURSE

FROM Studies

WHERE COST = (

SELECT MAX(COST)

FROM Studies

);

13. Which institute conducts the costliest course:

SELECT SPLACE

FROM Studies

WHERE COST = (

SELECT MAX(COST)

FROM Studies

);

14. Which course has the below average number of students:

SELECT COURSE

FROM Studies

GROUP BY COURSE

HAVING COUNT(\*) < (

SELECT AVG(Student\_Count)

FROM (

SELECT COUNT(\*) AS Student\_Count

FROM Studies

GROUP BY COURSE

) AS Course\_Student\_Count

);

15. Which institute conducts the above courses:

SELECT DISTINCT SPLACE

FROM Studies

WHERE COURSE IN (

SELECT COURSE

FROM Studies

GROUP BY COURSE

HAVING COUNT(\*) < (

SELECT AVG(Student\_Count)

FROM (

SELECT COUNT(\*) AS Student\_Count

FROM Studies

GROUP BY COURSE

) AS Course\_Student\_Count

)

);

These queries should provide you with the desired information based on the provided tables. Adjust them as needed based on your specific database schema and data.

Joins

1. Display the details of those who are drawing the same salary

2. Display the details of the software developed by the male programmers earning more than 3000

3. Display the details of the softwareâ€™s developed in .NET by female programmers

4. Display the details of software developed in C by female programmers of Pragathi

5. Display the number of softwareâ€™s, number of copies sold and sales value of each programmer institute wise

6. Display the details of the software developed in dbase by male programmers who being to the institute in which most number of programmer studied

7. Display the details of the software developed by male programmers who salary less 2500 and female programmers whose salary greater than 2500

8. Display the details of the software that was developed in the language that is not programmers first proficiency

9. Display the details of the software that was developed in the language that is neither the first nor the second proficiency of the programmer

10. Display the details of the software that was developed by the male students of Sabhari

11. Display the names of programmers who have not developed any software

12. What is the total cost of the software developed by the programmer in apple

ANSWERS:

Here are the  queries for the provided questions using joins:

1. Display the details of those who are drawing the same salary:

SELECT \*

FROM Programmer p1

JOIN Programmer p2 ON p1.SALARY = p2.SALARY AND p1.PNAME != p2.PNAME;

2. Display the details of the software developed by the male programmers earning more than 3000:

SELECT s.\*

FROM Software s

JOIN Programmer p ON s.PNAME = p.PNAME

WHERE p.GENDER = 'M' AND p.SALARY > 3000;

3. Display the details of the softwareâ€™s developed in .NET by female programmers:

SELECT s.\*

FROM Software s

JOIN Programmer p ON s.PNAME = p.PNAME

WHERE p.GENDER = 'F' AND p.PROF1 = '.NET';

4. Display the details of software developed in C by female programmers of Pragathi:

SELECT s.\*

FROM Software s

JOIN Programmer p ON s.PNAME = p.PNAME

JOIN Studies st ON p.PNAME = st.PNAME

WHERE p.GENDER = 'F' AND p.PROF1 = 'C' AND st.SPLACE = 'Pragathi';

5. Display the number of softwareâ€™s, number of copies sold, and sales value of each programmer institute wise:

SELECT p.PNAME, st.SPLACE, COUNT(\*) AS Num\_Softwares, SUM(SOLD) AS Total\_Copies\_Sold, SUM(SCOST \* SOLD) AS Total\_Sales\_ValueFROM Programmer p

JOIN Software s ON p.PNAME = s.PNAME

JOIN Studies st ON p.PNAME = st.PNAME

GROUP BY p.PNAME, st.SPLACE;

6. Display the details of the software developed in dbase by male programmers who belong to the institute in which most number of programmers studied:

SELECT s.\*

FROM Software s

JOIN Programmer p ON s.PNAME = p.PNAME

JOIN Studies st ON p.PNAME = st.PNAME

WHERE p.GENDER = 'M' AND st.SPLACE = (

SELECT SPLACE

FROM Studies

GROUP BY SPLACE

ORDER BY COUNT(\*) DESC

LIMIT 1

) AND s.DEV\_D = 'dbase';

7. Display the details of the software developed by male programmers who salary less than 2500 and female programmers whose salary greater than 2500:

SELECT s.\*

FROM Software s

JOIN Programmer p ON s.PNAME = p.PNAME

WHERE (p.GENDER = 'M' AND p.SALARY < 2500) OR (p.GENDER = 'F' AND p.SALARY > 2500);

8. Display the details of the software that was developed in the language that is not the programmer's first proficiency:

SELECT s.\*

FROM Software s

JOIN Programmer p ON s.PNAME = p.PNAME

WHERE s.DEV\_D NOT IN (p.PROF1, p.PROF2);

9. Display the details of the software that was developed in the language that is neither the first nor the second proficiency of the programmer:

SELECT s.\*

FROM Software s

JOIN Programmer p ON s.PNAME = p.PNAME

WHERE s.DEV\_D NOT IN (p.PROF1, p.PROF2)

AND s.DEV\_D NOT IN (p.PROF2, p.PROF1);

10. Display the details of the software that was developed by the male students of Sabhari:

SELECT s.\*

FROM Software s

JOIN Programmer p ON s.PNAME = p.PNAME

JOIN Studies st ON p.PNAME = st.PNAME

WHERE p.GENDER = 'M' AND st.SPLACE = 'Sabhari';

11. Display the names of programmers who have not developed any software:

SELECT PNAME

FROM Programmer

WHERE PNAME NOT IN (SELECT DISTINCT PNAME FROM Software);

12. What is the total cost of the software developed by the programmer in Apple:

SELECT SUM(DCOST) AS Total\_Cost

FROM Software

WHERE PNAME IN (

SELECT PNAME

FROM Programmer

WHERE PNAME IN (

SELECT PNAME

FROM Studies

WHERE SPLACE = 'Apple'

)

);

These queries should provide you with the desired information based on the provided tables. Adjust them as needed based on your specific database schema and data.

1.for EXP-3 in PL/ do the following

â€¢ Write a PL/ program to check whether a date falls on weekend i.e. SATURDAY or SUNDAY.

â€¢ Write a PL/ program to check whether a given character is letter or digit.

â€¢ Write a PL/ program to display the description against a grade.

â€¢ Write a PL/ program to display the description against a grade using CASE statement.

â€¢ Write a program in PL/ using nested loop with EXIT WHEN statement

â€¢ Write a program in PL/ to print the prime numbers between 1 to 50.

ANSWERS:

Sure, here are the PL/ programs for each of the given tasks:

1. PL/ program to check whether a date falls on a weekend (Saturday or Sunday):

DECLARE

v\_date DATE := TO\_DATE('2024-04-13', 'YYYY-MM-DD'); -- Example date

v\_day VARCHAR2(9);

BEGIN

-- Get the day of the week

SELECT TO\_CHAR(v\_date, 'DAY') INTO v\_day FROM DUAL;

-- Check if it's Saturday or Sunday

IF v\_day IN ('SATURDAY', 'SUNDAY') THEN

DBMS\_OUTPUT.PUT\_LINE('The date falls on a weekend.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('The date does not fall on a weekend.');

END IF;

END;

/

2. PL/ program to check whether a given character is a letter or digit:

DECLARE

v\_char CHAR(1) := 'A'; -- Example character

BEGIN

-- Check if the character is a letter or digit

IF (ASCII(v\_char) BETWEEN 48 AND 57) OR (ASCII(v\_char) BETWEEN 65 AND 90) OR (ASCII(v\_char) BETWEEN 97 AND 122) THEN

DBMS\_OUTPUT.PUT\_LINE('The character is a letter.');

ELSE

DBMS\_OUTPUT.PUT\_LINE('The character is a digit.');

END IF;

END;

/

3. PL/ program to display the description against a grade:

DECLARE

v\_grade CHAR(1) := 'A'; -- Example grade

BEGIN

-- Display description based on grade

CASE v\_grade

WHEN 'A' THEN DBMS\_OUTPUT.PUT\_LINE('Excellent');

WHEN 'B' THEN DBMS\_OUTPUT.PUT\_LINE('Good');

WHEN 'C' THEN DBMS\_OUTPUT.PUT\_LINE('Satisfactory');

WHEN 'D' THEN DBMS\_OUTPUT.PUT\_LINE('Needs improvement');

ELSE DBMS\_OUTPUT.PUT\_LINE('Invalid grade');

END CASE;

END;

/

4. PL/ program to display the description against a grade using the CASE statement:

DECLARE

v\_grade CHAR(1) := 'A'; -- Example grade

BEGIN

-- Display description based on grade

DBMS\_OUTPUT.PUT\_LINE(

CASE v\_grade

WHEN 'A' THEN 'Excellent'

WHEN 'B' THEN 'Good'

WHEN 'C' THEN 'Satisfactory'

WHEN 'D' THEN 'Needs improvement'

ELSE 'Invalid grade'

END

);

END;

/

5. Program in PL/ using nested loop with EXIT WHEN statement:

DECLARE

i INTEGER := 1;

j INTEGER := 1;

BEGIN

LOOP

LOOP DBMS\_OUTPUT.PUT(i || '\*' || j || '=' || i \* j || ' ');

j := j + 1;

EXIT WHEN j > 10; -- Exit loop when j exceeds 10

END LOOP;

DBMS\_OUTPUT.NEW\_LINE;

i := i + 1;

j := 1; -- Reset j for the outer loop

EXIT WHEN i > 10; -- Exit outer loop when i exceeds 10

END LOOP;

END;

/

6. Program in PL/ to print the prime numbers between 1 to 50:

DECLARE

i INTEGER := 2;

j INTEGER;

is\_prime BOOLEAN;

BEGIN

WHILE i <= 50 LOOP

is\_prime := TRUE;

j := 2;

WHILE j <= SQRT(i) LOOP

IF MOD(i, j) = 0 THEN

is\_prime := FALSE;

EXIT;

END IF;

j := j + 1;

END LOOP;

IF is\_prime THEN

DBMS\_OUTPUT.PUT\_LINE(i);

END IF;

i := i + 1;

END LOOP;

END;

/

These PL/ programs should help you accomplish the tasks outlined. Adjust them as needed based on your specific requirements.