SQL ASSIGNMENT

QUERIES I

1. SELECT AVG(scost) AS selling_cost_average FROM Software WHERE dev_in LIKE '%pascal%';

2. SELECT name, FLOOR(TIMESTAMPDIFF(YEAR, dob, CURDATE())) AS age

FROM Programmer;

3. SELECT p.name, FLOOR(TIMESTAMPDIFF(YEAR, p.dob, CURDATE())) AS age

FROM Programmer p

JOIN Studies s ON p.name = s.name

WHERE s.course = 'DCS';

4. SELECT MAX(sold) AS highest_sold FROM Software;

5. SELECT name, dob FROM Programmer WHERE EXTRACT(MONTH FROM dob) = 1;

6. SELECT MIN(ccost) AS low_course_fee FROM Studies;

7. SELECT COUNT(*) AS pgdca_programmers FROM Studies WHERE course = 'pgdca';

8. SELECT SUM(sold * scost) AS revenue FROM Software WHERE dev_in LIKE '%c%';

9. SELECT * FROM Software WHERE name = 'ramesh';

10. SELECT COUNT(*) AS sabhari_programmers FROM Studies WHERE splace = 'sabhari';

11. SELECT * FROM Software WHERE sold > 20000;

12. SELECT name, CEIL(dcost / scost) AS copies_sell FROM Software;

13. SELECT MAX(scost) AS costliest_software FROM Software WHERE dev_in = 'basic';

14. SELECT * FROM Software WHERE sold * scost >= dcost;

15. SELECT COUNT(*) AS dbase_packages FROM Software WHERE dev_in = 'dbase';

16. SELECT COUNT(*) AS paragathi_programmers FROM Studies WHERE splace = 'paragathi';

17. SELECT COUNT(*) AS programmers_paid_between_5000_and_10000 FROM Studies

WHERE ccost BETWEEN 5000 AND 10000;

18. SELECT AVG(ccost) AS average_course_fee FROM Studies;

19. SELECT * FROM Programmer WHERE prof1 = 'c' OR prof2 = 'c';

20. SELECT COUNT(*) AS cobol_or_pascal FROM Programmer WHERE prof1 IN ('cobol', 'pascal') OR prof2 IN ('cobol', 'pascal');

21. SELECT COUNT(*) AS programmers_not_knowing_pascal_and_c

FROM Programmer

WHERE prof1 NOT IN ('pascal', 'c') AND prof2 NOT IN ('pascal', 'c');

22. SELECT MAX(EXTRACT(YEAR FROM CURRENT_DATE) - EXTRACT(YEAR FROM dob)) AS oldest_male_age FROM Programmer WHERE sex = 'male';

23. SELECT AVG(EXTRACT(YEAR FROM CURRENT_DATE) - EXTRACT(YEAR FROM dob)) AS average_female_age FROM Programmer WHERE sex = 'female';

24. SELECT name, TIMESTAMPDIFF(YEAR, doj, CURDATE()) AS experience_in_years

FROM Programmer ORDER BY experience_in_years DESC;

25. SELECT name, dob FROM Programmer

WHERE EXTRACT(MONTH FROM dob) = EXTRACT(MONTH FROM SYSDATE);

26. SELECT COUNT(*) AS female_programmers FROM Programmer WHERE sex = 'female';

27. SELECT DISTINCT prof1 AS language FROM Programmer WHERE sex = 'male'

UNION

SELECT DISTINCT prof2 FROM Programmer WHERE sex = 'male';

28. SELECT AVG(salary) AS average_salary FROM Programmer;

29. SELECT COUNT(*) AS count_salary FROM Programmer WHERE salary BETWEEN 2000 AND 4000;

30. SELECT * FROM Programmer WHERE prof1 NOT IN ('clipper', 'cobol', 'pascal') AND prof2 NOT IN ('clipper', 'cobol', 'pascal');

31. SELECT COUNT(*) AS female_programmers FROM Programmer

WHERE sex = 'female' AND (prof1 = 'c' OR prof2 = 'c') AND FLOOR(TIMESTAMPDIFF(YEAR, dob, CURDATE())) > 24;

32 SELECT name, dob FROM Programmer WHERE MONTH(dob) = MONTH(CURDATE())

AND

DAYOFMONTH(dob) BETWEEN DAYOFMONTH(CURDATE()) AND DAYOFMONTH(CURDATE() + INTERVAL 7 DAY);

[note: compare the month and compare day of month with ahead of 7 days]

33. SELECT * FROM programmer WHERE TIMESTAMPDIFF(YEAR, doj, CURDATE()) < 1;

34. SELECT *FROM programmer WHERE TIMESTAMPDIFF(YEAR, doj, CURDATE()) = 2

AND MONTH(doj) <= MONTH(CURDATE())

AND DAYOFMONTH(doj) <= DAYOFMONTH(CURDATE());

35. SELECT name, (dcost - (sold * scost)) AS amount_to_be_recovered

FROM Software WHERE sold * scost < dcost;

36. SELECT * FROM Software WHERE sold = 0;

37. SELECT SUM(scost) AS total_cost FROM Software WHERE name = 'mary';

38. SELECT DISTINCT splace AS institute_name FROM Studies;

39. SELECT COUNT(DISTINCT course) AS courses FROM Studies;

40. SELECT name FROM Programmer WHERE LENGTH(name) - LENGTH(REPLACE(name, 'a', '')) = 2;

41. SELECT name FROM Programmer WHERE LENGTH(name) <= 5;

42. SELECT COUNT(*) AS femaleProgrammers FROM Programmer

WHERE sex = 'female' AND (prof1 = 'cobol' OR prof2 = 'cobol') AND
TIMESTAMPDIFF(YEAR,doj,CURDATE())> 2;

43. SELECT MIN(LENGTH(name)) AS shortest_name FROM Programmer;

44. SELECT AVG(dcost) AS averagedevelopment_cost FROM Software

WHERE dev_in = 'cobol';

45. SELECT name, sex, DATE_FORMAT(dob, '%d/%m/%y') AS dob_ddmmyy, DATE_FORMAT(doj, '%d/%m/%y') AS doj_ddmmyy FROM Programmer;

46. SELECT name, dob FROM Programmer WHERE DAY(LAST_DAY(dob)) = DAY(dob);

47. SELECT SUM(salary) AS total_salary_paid FROM Programmer WHERE sex = 'male' AND (prof1 IN 'cobol' OR prof2 IN 'cobol');

48. SELECT title, scost, dcost, (scost - dcost) AS difference FROM Software

ORDER BY difference DESC;

49. SELECT name, dob, doj FROM Programmer WHERE MONTH(dob) = MONTH(doj);

50. SELECT * FROM Software WHERE title LIKE '% %';

QUERIES II

1. SELECT dev_in AS language, COUNT(*) AS package_count FROM Software GROUP BY dev_in;

2. SELECT name, COUNT(*) AS package_count FROM Software GROUP BY name;

3. SELECT sex, COUNT(*) AS programmer_count FROM Programmer GROUP BY sex;

4. SELECT dev_in AS language, MAX(dcost) AS costliest_package, MAX(sold) AS highest_selling

FROM Software GROUP BY dev_in;

5. SELECT YEAR(dob) AS birth_year, COUNT(*) AS people_count FROM Programmer

GROUP BY YEAR(dob);

6. SELECT YEAR(doj) AS joinYear, COUNT(*) AS peopleCount

FROM Programmer GROUP BY YEAR(doj);

7 .SELECT MONTH(dob) AS birthMonth, COUNT(*) AS peopleCount FROM Programmer GROUP BY MONTH(dob);

8.. SELECT MONTH(doj) AS join_month, COUNT(*) AS people_count

FROM Programmer GROUP BY MONTH(doj);

9. SELECT prof1 AS language, COUNT(*) AS count_prof1 FROM Programmer GROUP BY prof1;

10. SELECT prof2 AS language, COUNT(*) AS count_prof2 FROM Programmer GROUP BY prof2;

11. SELECT FLOOR(salary / 1000) AS salary_group, COUNT(*) AS count FROM Programmer GROUP BY salary_group;

12. SELECT splace AS institute_name, COUNT(*) AS people_count FROM Studies

GROUP BY splace;

13. SELECT course, COUNT(*) AS people_count FROM Studies GROUP BY course;

14. SELECT dev_in AS language, SUM(dcost) AS total_dcost

FROM Software GROUP BY dev_in;

15. SELECT dev_in AS language, SUM(scost) AS total_scost FROM Software GROUP BY dev_in;

16. SELECT name, SUM(dcost) AS total_dcost FROM Software GROUP BY name;

17. SELECT name, SUM(sold * scost) AS total_sales_value FROM Software GROUP BY name;

18. SELECT name, COUNT(*) AS package_count FROM Software

GROUP BY name;

19. SELECT name, dev_in AS language, SUM(sold * scost) AS total_sales_cost FROM Software GROUP BY name, language;

20. SELECT name, MAX(scost) AS costliest_package, MIN(scost) AS cheapest_package

FROM Software GROUP BY name;

21. SELECT dev_in AS language, AVG(dcost) AS average_dcost,AVG(scost) AS average_cost,

SUM(scost) AS total_scost, AVG(scost) / AVG(sold) AS average_price_per_copy FROM Software

GROUP BY dev_in;

22. SELECT splace AS institute_name, COUNT(DISTINCT course) AS number_of_courses,

AVG(ccost) AS average_ccourse FROM Studies

GROUP BY splace;

23. SELECT splace AS institute_name, COUNT(*) AS number_of_students

FROM Studies GROUP BY splace;

24. SELECT name, sex FROM Programmer

WHERE sex IN ('male', 'female');

25. SELECT name AS programmer_name, title AS package_name from Software;

26. SELECT dev_in AS language, COUNT(*) AS number_of_packages

FROM Software GROUP BY dev_in; **[Doubt]**

27. SELECT dev_in AS language, COUNT(*) AS number_of_packages

FROM Software WHERE dcost < 1000 GROUP BY dev_in;

28. SELECT dev_in AS language, AVG(scost - dcost) AS average_difference

FROM Software GROUP BY dev_in;

29. SELECT name, SUM(scost) AS total_scost, SUM(dcost) AS total_dcost, SUM(scost - dcost) AS amount_to_be_recovered FROM Software GROUP BY name HAVING SUM(scost - dcost) > 0; [**doubt]**

30. SELECT MAX(salary) AS high_salary, MIN(salary) AS low_salary, AVG(salary) AS average_salary FROM Programmer WHERE salary > 2000;

QUERIES III

1. SELECT name FROM Programmer WHERE prof1 = 'C' OR prof2 = 'C' ORDER BY salary DESC LIMIT 1;
2. SELECT name FROM Programmer WHERE sex = 'female' AND (prof1 = 'COBOL' OR prof2 = 'COBOL') ORDER BY salary DESC LIMIT 1;
3. SELECT prof1 AS language, name FROM Programmer WHERE (prof1 IS NOT NULL) GROUP BY prof1 HAVING MAX(salary);
4. SELECT name FROM Programmer ORDER BY dob LIMIT 1;
5. SELECT name FROM Programmer ORDER BY dob DESC LIMIT 1;
6.
```
SELECT language FROM (
   SELECT prof1 AS language FROM Programmer
   UNION ALL
   SELECT prof2 AS language FROM Programmer
) AS combined_languages
GROUP BY language
HAVING COUNT(*) = 1;
```
7. SELECT name FROM Programmer WHERE prof1 = 'DBASE' OR prof2 = 'DBASE' ORDER BY dob DESC LIMIT 1;
8. SELECT splace AS institute_name, COUNT(*) AS num_students FROM Studies GROUP BY splace ORDER BY num_students DESC1;

9. SELECT name
FROM Programmer
WHERE sex = 'female' AND salary > 3000
AND prof1 NOT IN ('C', 'C++', 'Oracle', 'DBASE') AND prof2 NOT IN ('C', 'C++', 'Oracle', 'DBASE');

10. SELECT course FROM Studies ORDER BY ccost DESC LIMIT 1;

11. SELECT course FROM Studies GROUP BY course
ORDER BY COUNT(*) DESC LIMIT 1;

12. SELECT splace AS institute_name, course FROM Studies
GROUP BY splace, course
HAVING AVG(ccost) < (SELECT AVG(ccost) FROM Studies);

13. SELECT splace AS institute_name FROM Studies
WHERE ccost = (SELECT MAX(ccost) FROM Studies);

14. SELECT courseFROM Studies
GROUP BY course HAVING COUNT(*) < (SELECT AVG(num_students) FROM (SELECT COUNT(*) AS num_students FROM Studies GROUP BY course) AS avg_students);

15. SELECT splace AS institute_name FROM Studies
WHERE course = (SELECT course FROM Studies GROUP BY course HAVING COUNT(*) < (SELECT AVG(num_students) FROM (SELECT COUNT(*) AS num_students FROM Studies GROUP BY course) AS avg_students));

16. SELECT course FROM Studies
GROUP BY course HAVING ccost BETWEEN (SELECT AVG(ccost) - 1000 FROM Studies) AND (SELECT AVG(ccost) + 1000 FROM Studies);

17. SELECT title FROM Software ORDER BY dcost DESC LIMIT 1;

18. SELECT title FROM Software ORDER BY scost ASC LIMIT 1;

19. SELECT name FROM Software ORDER BY sold LIMIT 1;

20. SELECT dev_in FROM Software ORDER BY sold * scost DESC LIMIT 1;

21. SELECT sold FROM Software ORDER BY ABS(dcost - scost) LIMIT 1;

22. SELECT title FROM Software WHERE dev_in = 'Pascal' ORDER BY scost DESC LIMIT 1;

24. SELECT dev_in FROM Software GROUP BY dev_in ORDER BY COUNT(*) DESC LIMIT 1;

25. SELECT name FROM Software GROUP BY name ORDER BY COUNT(*) DESC LIMIT 1;

26. SELECT name FROM Software ORDER BY scost DESC LIMIT 1;

27. SELECT title FROM Software GROUP BY title HAVING sold < (SELECT AVG(sold) FROM Software);

28. SELECT name FROM Programmer WHERE sex = 'female' AND salary > (SELECT MAX(salary) FROM Programmer WHERE sex = 'male');

29. SELECT prof1 FROM Programmer GROUP BY prof1 ORDER BY COUNT(*) DESC LIMIT 1;

30. SELECT name FROM Software GROUP BY name HAVING SUM(scost) > 2 * SUM(dcost);

31. SELECT p.name AS programmer_name, s.title AS cheapest_package

FROM Programmer p

INNER JOIN Software s ON p.name = s.name

INNER JOIN (

SELECT dev_in, MIN(scost) AS min_scost

```
    FROM Software
    GROUP BY dev_in
) cheapest ON s.dev_in = cheapest.dev_in AND s.scost = cheapest.min_scost;
```

32. SELECT name FROM Programmer WHERE sex = 'male' AND YEAR(dob) = 1965 ORDER BY dob DESC LIMIT 1;

33. SELECT name,

```
    (SELECT dev_in FROM Software WHERE name = p.name ORDER BY sold DESC LIMIT 1)
AS highest_selling_language,
    (SELECT dev_in FROM Software WHERE name = p.name ORDER BY sold ASC LIMIT 1)
AS lowest_selling_language
FROM Programmer p;
```

34. SELECT name FROM Programmer WHERE sex = 'female' AND YEAR(doj) = 1992 ORDER BY dob ASC LIMIT 1;

35. SELECT YEAR(dob) AS birth_year, COUNT(*) AS num_programmers_born

```
FROM Programmer
GROUP BY birth_year
ORDER BY num_programmers_born DESC
LIMIT 1;
```

36. SELECT MONTH(doj) AS join_month, COUNT(*) AS num_programmers_joined

```
FROM Programmer
GROUP BY join_month
ORDER BY num_programmers_joined DESC
LIMIT 1;
```

37. SELECT prof1 AS language, COUNT(*) AS num_programmers

```
FROM Programmer
GROUP BY prof1
UNION ALL
SELECT prof2 AS language, COUNT(*) AS num_programmers
FROM Programmer
GROUP BY prof2
ORDER BY num_programmers DESC
LIMIT 1;
```

38. SELECT name FROM Programmer WHERE sex = 'male' AND salary < (SELECT AVG(salary) FROM Programmer WHERE sex = 'female');

QUERIES-IV

1. SELECT * FROM Programmer WHERE salary IN (SELECT salary FROM Programmer GROUP BY salary HAVING COUNT(*) > 1);

2. SELECT * FROM Software WHERE name IN (SELECT name FROM Programmer WHERE sex = 'male' AND salary > 3000);

3. SELECT * FROM Software WHERE dev_in = 'Pascal' AND name IN (SELECT name FROM Programmer WHERE sex = 'female');

4. SELECT * FROM Programmer WHERE YEAR(doj) < 1990;

5. SELECT * FROM Software WHERE dev_in = 'C' AND name IN (SELECT name FROM Programmer WHERE sex = 'female' AND splace = 'PRAGATHI');

6. SELECT p.name AS programmer_name, p.splace AS institute_name, COUNT(*) AS number_of_packages, SUM(sold) AS total_copies_sold, SUM(sold * scost) AS total_sales_value FROM Programmer p LEFT JOIN Software s ON p.name = s.name GROUP BY p.name, p.splace;

7. SELECT * FROM Software WHERE dev_in = 'DBASE' AND name IN (SELECT name FROM Programmer WHERE sex = 'male' AND splace = (SELECT splace FROM Programmer GROUP BY splace ORDER BY COUNT(*) DESC LIMIT 1));

8. SELECT * FROM Software WHERE name IN (SELECT name FROM Programmer WHERE (sex = 'male' AND YEAR(dob) < 1965) OR (sex = 'female' AND YEAR(dob) > 1975));

9. SELECT * FROM Software WHERE dev_in NOT IN (SELECT prof1 FROM Programmer UNION SELECT prof2 FROM Programmer);

10. SELECT * FROM Software WHERE dev_in NOT IN (SELECT prof1 FROM Programmer UNION SELECT prof2 FROM Programmer);

11. SELECT * FROM Software WHERE name IN (SELECT name FROM Programmer WHERE sex = 'male' AND splace = 'SABHARI');

12. SELECT name FROM Programmer WHERE name NOT IN (SELECT DISTINCT name FROM Software);

13. SELECT SUM(scost) AS total_cost FROM Software WHERE name IN (SELECT name FROM Programmer WHERE splace = 'APPLE');

14. SELECT name FROM Programmer GROUP BY doj HAVING COUNT(*) > 1;

15. SELECT name FROM Programmer GROUP BY prof2 HAVING COUNT(*) > 1;