1. List all students with a total credit count of 0.

select * from student where tot_cred>0 and rownum<10;

```
SQL> select * from student where tot_cred>0 and rownum<10;

ID    NAME                 DEBT_NAME             TOT_CRED
----- -------------------- -------------------- ----------
61065 Jovicic              Civil Eng.                   31
107   Shabuno              Math                         19
11453 Yamashita            Astronomy                   109
53805 Ludwig               Cybernetics                  30
39241 Solar                Mech. Eng.                   64
32886 Damas                Psychology                   58
40080 Llam                 Civil Eng.                    6
22142 Gerstend             History                      22
94257 Unger                Languages                    12
```

2. Find the names of all courses offered by the "Physics" department.

Select * from student where tot_cred>0 and rownum<10;

```
SQL> select title from course where Debt_name='Physics' and rownum<10;

TITLE
--------------------------------------------------
Mobile Computing
Cost Accounting
Bacteriology
Hydraulics
Stream Processing
The Music of Donovan
Differential Geometry
The Music of the Ramones
The Music of Dave Edmunds

9 rows selected.
```

3. Retrieve the names of all instructors who earn less than ₹50,000.

4. List all classrooms with a capacity of exactly 100.

```
SQL> select name from instructor where salary<'50000' and rownum<10;

NAME
--------------------
Konstantinides
Queiroz
Hau
Lembr
Vicentino
Desyl
Ullman
Morris
Yin

9 rows selected.
```

```
SQL> select *  from classroom where capacity<100 and rownum<10;

BUILDING        ROOM_NU   CAPACITY
--------------- ------- ----------
Lamberton       134             10
Chandler        375             10
Fairchild       145             27
Nassau          45              92
Grace           40              34
Lamberton       143             10
Painter         86              97
Alumni          547             26
Alumni          143             47

9 rows selected.
```

5. Find the total number of courses offered in the "Summer" semester.

```
SQL> select count(Distinct course_id) as total_count from section where semester
='Spring' group by semester;

TOTAL_COUNT
-----------
         45
```

6. List all students who have taken exactly 30 total credits.

```
SQL> select * from student where tot_cred=30;

ID    NAME                 DEBT_NAME             TOT_CRED
----- -------------------- -------------------- ----------
53805 Ludwig               Cybernetics                  30
76291 Dellwo               Physics                      30
7390  Stone                Accounting                   30
5925  Maw                  Languages                    30
36845 Okaf                 Math                         30
81638 Chiu                 Statistics                   30
38899 Murphy               Marketing                    30
28952 Kennedy              Accounting                   30
61127 Tuki                 Physics                      30
68712 Hill                 Civil Eng.                   30
52866 Loull                Math                         30

ID    NAME                 DEBT_NAME             TOT_CRED
----- -------------------- -------------------- ----------
5381  Diana                Languages                    30
83003 Nam                  Psychology                   30
72177 Eller                Mech. Eng.                   30

14 rows selected.
```

7. Find the names of all courses offered by the "Math" department.

```
SQL> select title, course_id from course where debt_name='Math';

TITLE                                              COURSE_I
-------------------------------------------------- --------
Environmental Law                                  843
The Beatles                                        679
Physical Chemistry                                 461
International Trade                                235
Sailing                                            858
Computability Theory                               919
Geology                                            659
Colloid and Surface Chemistry                      258
Optics                                             694
Music of the 90s                                   270

10 rows selected.
```

8. Retrieve the names of all instructors who earn more than ₹100,000.

```
SQL> select name from instructor where salary>100000;

NAME
--------------------
Mird
Shuming
Voronina
Arias
Mingoz
Kenje
Jaekel
Bondi
Lent
Sakurai
Bietzk

NAME
--------------------
Wieland

12 rows selected.
```

9. List all classrooms located in the "Taylor" building.

```
SQL> select * from classroom where building='Taylor'
  2  ;

BUILDING        ROOM_NU   CAPACITY
--------------- ------- ----------
Taylor          183             71
Taylor          812            115
```

10. Find the total number of courses offered in the "Winter" semester.

```
SQL> select count(distinct course_id) as total_Fall from section  where semester='Fall' group by s
emester;

TOTAL_FALL
----------
        46
```

11. List all students who have taken courses with a grade of "C" or lower.

```
SQL> SELECT DISTINCT student.ID, student.name
  2  FROM student
  3  JOIN takes ON student.ID = takes.ID
  4  WHERE takes.grade IN ('C ', 'C-') and rownum<10;

ID    NAME
----- --------------------
31560 Neld
10727 Allard
32217 Argar
5414  Aiken
61920 Marcol
32376 Nakajima
88887 Wodn
96710 Katehakis
52866 Loull

9 rows selected.
```

12. Retrieve the names of instructors who have taught in the "Fall" semester of 2022.

```
SQL> select i.name from instructor i join teaches t on i.id =t.id where semester='Fall' and year=2
005;

NAME
--------------------
Mingoz
DAgostino
Sakurai
Kean
```

13. Find the average capacity of classrooms in each building.

```
SQL> select building, avg(capacity) from classroom
  2  group by building;

BUILDING        AVG(CAPACITY)
--------------- -------------
Painter                    97
Stabler                   113
Nassau                     92
Chandler                10.5
Whitman                    76
Alumni                  36.5
Main                       26
Power                      11
Garfield                   59
Taylor                     93
Saucon              49.3333333

BUILDING        AVG(CAPACITY)
--------------- -------------
Gates                   37.5
Fairchild                  27
Polya                      28
Grace                      34
Bronfman                   12
Lambeau                    51
Lamberton                  10
Drown                      18
Rathbone                   60

20 rows selected.
```

14. List all courses that have more than 3 credits.

```
SQL> select * from course where credits>3 and rownum<10;

COURSE_I TITLE                                              DEBT_NAME              CREDITS
-------- -------------------------------------------------- -------------------- ----------
787      C  Programming                                     Mech. Eng.                    4
278      Greek Tragedy                                      Statistics                    4
972      Greek Tragedy                                      Psychology                    4
400      Visual BASIC                                       Psychology                    4
762      The Monkeys                                        History                       4
482      FOCAL Programming                                  Psychology                    4
581      Calculus                                           Pol. Sci.                     4
843      Environmental Law                                  Math                          4
704      Marine Mammals                                     Geology                       4

9 rows selected.
```

15. Retrieve the names of students who have taken courses in the "Watson" building.

```
select st.name from student st
join takes t on st.id=t.id
 join section se on se.course_id=t.course_id
  4  where se.building ='Whitman' and rownum<10;

NAME
--------------------
Zeng
Towsey
Peeri
Conti
Grant
Grant
Pomy
Pomy
Nirenbu

9 rows selected.
```

16. Find the total number of students advised by each instructor.

```
SQL> select i.name, i.id ,count(a.s_id)
  2  from instructor i
  3  join advisor a on a.I_id=i.id
  4  group by i.name ,i.id;

NAME                 ID    COUNT(A.S_ID)
-------------------- ----- -------------
Atanassov            28400            44
Desyl                59795            31
Sullivan             73623            46
Romero               43779            34
Voronina             74420            31
Hau                  57180            45
McKinnon             63395            31
Mahmoud              77346            54
Moreira              31955            35
Levine               79653            46
Choll                90643            24

NAME                 ID    COUNT(A.S_ID)
-------------------- ----- -------------
Arias                37687            50
Lembr                14365            39
Sarkar               48570            49
Bietzk               90376            39
Gustafsson           3199             38
Murata               4034             45
Bondi                34175            33
Gutierrez            64871            36
Bourrier             3335             32
Wieland              19368            33
DAgostino            22591            40

NAME                 ID    COUNT(A.S_ID)
-------------------- ----- -------------
Bertolino            97302            50
Mingoz               6569             46
Bawa                 15347            38
Valtchev             81991            40
Dale                 99052            33
Yazdi                16807            48
Sakurai              95709            41
Tung                 41930            41
Ullman               79081            40
Mird                 96895            38
Luo                  4233             44
```

```
NAME                 ID    COUNT(A.S_ID)
-------------------- ----- -------------
Pimenta              65931            38
Arinb                95030            41
Pingr                78699            33
Lent                 48507            36
Morris               36897            42
Soisalon-Soininen    35579            54
Konstantinides       50885            28
Vicentino            42782            41
Kean                 28097            37
Yin                  72553            48
Dusserre             58558            40

NAME                 ID    COUNT(A.S_ID)
-------------------- ----- -------------
Liley                25946            38
Bancilhon            52647            48
Kenje                74426            39
Jaekel               63287            41
Queiroz              80759            43
Shuming              50330            39

50 rows selected.

SQL> group by i.name ,i.id;
```

17. List all sections that do not have a classroom assigned.

```
SQL> set linesize 200;
SQL> select * from section where room_number is NULL;

COURSE_I SEC_ID   SEMEST       YEAR BUILDING        ROOM_NU TIME
-------- -------- ------ ---------- --------------- ------- ----
313      1        Spring       2025

SQL>
```

18. Retrieve the names of students who have taken courses with the highest number of credits.

```
SQL> select s.name from student s
  2  join takes t on s.id=t.id
  3  join course c on t.course_id=c.course_id
  4  where c.credits=(select MAX(credits) from course) and rownum<10;

NAME
--------------------
Hendrickson
Hendrickson
Hendrickson
Hendrickson
Hendrickson
Zeng
Zeng
Zeng
Zeng

9 rows selected.

SQL>
```

19. Find the departments with the highest average instructor salary.

```
SELECT debt_name, AVG(salary) AS highest_salary
FROM instructor
GROUP BY debt_name
HAVING AVG(salary) = (
    SELECT MAX(AVG(salary))
    FROM instructor
    GROUP BY debt_name
  8  );

DEBT_NAME            HIGHEST_SALARY
-------------------- --------------
Physics                    114576.9
```

20. List all courses that have been taught by more than one instructor.

```
SQL> select course_id ,count(distinct Id) as instructor_count
  2  from teaches
  3  group by course_id having count(distinct id)>1;

COURSE_I INSTRUCTOR_COUNT
-------- ----------------
867                     2
158                     2
200                     2
960                     2
468                     2
```

21. List all students who have taken courses with a grade of "A" in the "Fall" semester of 2022.
22. Retrieve the names of instructors who have taught in the "Spring" semester of 2023.

23. Find the average number of students enrolled in each course.
24. List all courses that have exactly 4 credits.
25. Retrieve the names of students who have taken courses in the "Packard" building.

26. Find the total number of sections taught by each instructor.

```
SELECT i.name, COUNT(*)
FROM instructor i
JOIN teaches t ON i.ID = t.ID
GROUP BY i.name;
```

27. List all sections that are held in rooms with a capacity of less than 50.

```
SELECT sec.course_id, sec.sec_id
FROM section sec
JOIN classroom c ON sec.building = c.building AND sec.room_number = c.room_number
WHERE c.capacity < 50;
```

28. Retrieve the names of students who have taken courses with the lowest number of credits.

```
select s.name from student s join takes t on t.id = s.id
join course c on t.course_id=c .course_id
where c.credits=(select min(credits) from course);
```

29. Find the departments with the lowest average instructor salary.

```
SELECT dept_name
FROM instructor
GROUP BY dept_name
ORDER BY AVG(salary)
FETCH FIRST 1 ROW WITH TIES;
```

Or limit

30. List all courses that have been taught by only one instructor.

31. Find the students who have taken courses with every instructor in the university.

```
SELECT s.id, s.name
FROM student s
WHERE NOT EXISTS (
   SELECT i.id
   FROM instructor i
   MINUS
   SELECT t.instructor_id
   FROM takes tk
   JOIN teaches t ON tk.course_id = t.course_id
   WHERE tk.id = s.id
);
```

32. Retrieve the names of instructors who have taught courses in all available semesters. *

```
SELECT i.id, i.name
FROM instructor i
WHERE NOT EXISTS (
   SELECT DISTINCT semester FROM section
   MINUS
   SELECT DISTINCT s.semester
   FROM teaches t
   JOIN section s ON t.course_id = s.course_id AND t.sec_id = s.sec_id
   WHERE t.instructor_id = i.id
);
```

33. List all courses that have been taught in every building.

```
SELECT c.course_id, c.title
FROM course c
WHERE NOT EXISTS (
   SELECT DISTINCT building FROM section
   MINUS
   SELECT DISTINCT s.building
   FROM section s
   WHERE s.course_id = c.course_id
);
```

```
SELECT c.title
FROM course c
JOIN section s ON c.course_id = s.course_id
GROUP BY c.title
HAVING COUNT(DISTINCT s.building) = (SELECT COUNT(DISTINCT building) FROM
classroom);
```

34. Find the students who have taken the most number of courses in a single year.

```
SELECT tk.id, s.name, tk.year, COUNT(*) AS course_count
FROM takes tk
JOIN student s ON tk.id = s.id
GROUP BY tk.id, s.name, tk.year
HAVING COUNT(*) = (
   SELECT MAX(course_count)
   FROM (SELECT id, year, COUNT(*) AS course_count
         FROM takes
         GROUP BY id, year)
);
```

```
SELECT s.name, t.year, COUNT(*)
FROM student s
JOIN takes t ON s.ID = t.ID
GROUP BY s.name, t.year
ORDER BY COUNT(*) DESC
FETCH FIRST 1 ROW WITH TIES;
```

35. Retrieve the names of students who have taken courses with overlapping time slots in the same semester.

```
SELECT DISTINCT s.id, s.name
FROM student s
JOIN takes t1 ON s.id = t1.id
JOIN section sec1 ON t1.course_id = sec1.course_id AND t1.sec_id = sec1.sec_id
JOIN time_slot ts1 ON sec1.time_slot_id = ts1.time_slot_id
JOIN takes t2 ON s.id = t2.id AND t1.course_id <> t2.course_id
JOIN section sec2 ON t2.course_id = sec2.course_id AND t2.sec_id = sec2.sec_id
JOIN time_slot ts2 ON sec2.time_slot_id = ts2.time_slot_id
WHERE t1.semester = t2.semester AND ts1.time_slot_id = ts2.time_slot_id;
```

```
SELECT DISTINCT s.name
FROM takes t1
JOIN takes t2 ON t1.ID = t2.ID AND t1.course_id != t2.course_id
JOIN section s1 ON t1.course_id = s1.course_id
JOIN section s2 ON t2.course_id = s2.course_id
JOIN time_slot ts1 ON s1.time_slot_id = ts1.time_slot_id
JOIN time_slot ts2 ON s2.time_slot_id = ts2.time_slot_id
WHERE t1.semester = t2.semester
AND ts1.day = ts2.day
AND (
  (ts1.start_hr < ts2.end_hr OR (ts1.start_hr = ts2.end_hr AND ts1.start_min < ts2.end_min))
  AND
  (ts2.start_hr < ts1.end_hr OR (ts2.start_hr = ts1.end_hr AND ts2.start_min < ts1.end_min))
);
```

36. List all courses that have been taught by the same instructor in consecutive semesters.

```
SELECT DISTINCT t1.course_id, t1.instructor_id
FROM teaches t1
JOIN teaches t2 ON t1.instructor_id = t2.instructor_id
AND t1.course_id = t2.course_id
AND t1.year = t2.year - 1
AND (t1.semester = 'Fall' AND t2.semester = 'Spring');
```

```
SELECT DISTINCT t1.course_id, i.name
FROM teaches t1
JOIN teaches t2 ON t1.ID = t2.ID AND t1.course_id = t2.course_id
JOIN instructor i ON t1.ID = i.ID
WHERE (t1.year = t2.year -1 AND t1.semester = 'Fall' AND t2.semester = 'Spring')
OR (t1.semester = 'Winter' AND t2.semester = 'Spring');
```

37. Find the departments where the total student credits are less than the department budget.

```
SELECT d.dept_name
FROM department d
JOIN student s ON d.dept_name = s.debt_name
```

```
GROUP BY d.dept_name, d.budget
HAVING SUM(s.tot_cred) < d.budget;

SELECT d.dept_name
FROM department d
JOIN student s ON d.dept_name = s.dept_name
GROUP BY d.dept_name, d.budget
HAVING SUM(s.tot_cred) < d.budget;
```

38. Retrieve the names of students who have taken courses with every time slot.

```
SELECT s.id, s.name
FROM student s
WHERE NOT EXISTS (
   SELECT time_slot_id FROM time_slot
   MINUS
   SELECT DISTINCT sec.time_slot_id
   FROM takes t
   JOIN section sec ON t.course_id = sec.course_id AND t.sec_id = sec.sec_id
   WHERE t.id = s.id
);

SELECT s.name
FROM student s
WHERE NOT EXISTS (
  SELECT time_slot_id FROM time_slot
  MINUS
  SELECT DISTINCT time_slot_id
  FROM takes t
  JOIN section sec ON t.course_id = sec.course_id
  WHERE t.ID = s.ID
);
```

39. List all courses that have been taught in every year.

```
SELECT c.course_id, c.title
FROM course c
WHERE NOT EXISTS (
   SELECT DISTINCT year FROM section
   MINUS
   SELECT DISTINCT s.year
   FROM section s
   WHERE s.course_id = c.course_id
);

SELECT course_id
FROM section
GROUP BY course_id
HAVING COUNT(DISTINCT year) = (SELECT COUNT(DISTINCT year) FROM section);
```

40. Find the students who have taken the most number of courses with the same instructor.

```
SELECT t.id, s.name, t.instructor_id, COUNT(*) AS course_count
FROM takes t
JOIN student s ON t.id = s.id
GROUP BY t.id, s.name, t.instructor_id
HAVING COUNT(*) = (
   SELECT MAX(course_count)
   FROM (SELECT id, instructor_id, COUNT(*) AS course_count
        FROM takes
        GROUP BY id, instructor_id)
);
```

```
SELECT s.name, te.ID, COUNT(*)
FROM student s
JOIN takes t ON s.ID = t.ID
JOIN teaches te ON t.course_id = te.course_id
GROUP BY s.name, te.ID
ORDER BY COUNT(*) DESC
FETCH FIRST 1 ROW WITH TIES;
```

41. Find the students who have taken courses from at least 3 different departments.

```
SELECT t.id, s.name
FROM takes t
JOIN section sec ON t.course_id = sec.course_id
JOIN course c ON sec.course_id = c.course_id
GROUP BY t.id, s.name
HAVING COUNT(DISTINCT c.dept_name) >= 3;
```

```
SELECT s.name
FROM student s
JOIN takes t ON s.ID = t.ID
JOIN course c ON t.course_id = c.course_id
GROUP BY s.name
HAVING COUNT(DISTINCT c.dept_name) >= 3;
```

42. Retrieve the names of instructors who have taught the same course in consecutive years.

```
SELECT DISTINCT t1.instructor_id, i.name
FROM teaches t1
JOIN teaches t2 ON t1.course_id = t2.course_id
AND t1.instructor_id = t2.instructor_id
AND t1.year = t2.year - 1
JOIN instructor i ON t1.instructor_id = i.id;
```

```
SELECT i.name, t1.course_id
FROM teaches t1
JOIN teaches t2 ON t1.ID = t2.ID AND t1.course_id = t2.course_id
JOIN instructor i ON t1.ID = i.ID
WHERE t2.year = t1.year + 1;
```

43. List all courses that are prerequisites for exactly 2 other courses.

```
SELECT p.course_id
FROM prereq p
GROUP BY p.course_id
HAVING COUNT(DISTINCT p.prereq_id) = 2;

SELECT prereq_id
FROM prereq
GROUP BY prereq_id
HAVING COUNT(*) = 2;
```

44. Find the departments where the total student credits are equal to the department budget.

```
SELECT d.dept_name
FROM department d
JOIN student s ON d.dept_name = s.debt_name
GROUP BY d.dept_name, d.budget
HAVING SUM(s.tot_cred) = d.budget;
```

45. Retrieve the names of students who have taken courses with every time slot in a single semester.

```
SELECT s.id, s.name
FROM student s
WHERE NOT EXISTS (
   SELECT time_slot_id FROM time_slot
   MINUS
   SELECT DISTINCT sec.time_slot_id
   FROM takes t
   JOIN section sec ON t.course_id = sec.course_id AND t.sec_id = sec.sec_id
   WHERE t.id = s.id AND t.semester = 'Fall'
);

SELECT s.name, t.semester, t.year
FROM student s
JOIN takes t ON s.ID = t.ID
JOIN section sec ON t.course_id = sec.course_id
GROUP BY s.name, t.semester, t.year
HAVING COUNT(DISTINCT sec.time_slot_id) = (SELECT COUNT(DISTINCT time_slot_id)
FROM time_slot);
```

46. List all courses that have been taught in every building except one.

```
SELECT c.course_id, c.title
FROM course c
WHERE (
   SELECT COUNT(DISTINCT s.building)
   FROM section s
   WHERE s.course_id = c.course_id
```

```
) = (SELECT COUNT(DISTINCT building) FROM section) - 1;
```

47. Find the students who have taken the most number of courses in a single department.

```
SELECT t.id, s.name, c.dept_name, COUNT(*) AS course_count
FROM takes t
JOIN section sec ON t.course_id = sec.course_id
JOIN course c ON sec.course_id = c.course_id
JOIN student s ON t.id = s.id
GROUP BY t.id, s.name, c.dept_name
HAVING COUNT(*) = (
   SELECT MAX(course_count)
   FROM (SELECT id, dept_name, COUNT(*) AS course_count
        FROM takes
        JOIN section USING(course_id)
        JOIN course USING(course_id)
        GROUP BY id, dept_name)
);
```

```
SELECT s.name, c.dept_name, COUNT(*)
FROM student s
JOIN takes t ON s.ID = t.ID
JOIN course c ON t.course_id = c.course_id
GROUP BY s.name, c.dept_name
ORDER BY COUNT(*) DESC
FETCH FIRST 1 ROW WITH TIES;
```

48. Retrieve the names of instructors who have taught courses in all available time slots.

```
SELECT i.id, i.name
FROM instructor i
WHERE NOT EXISTS (
   SELECT time_slot_id FROM time_slot
   MINUS
   SELECT DISTINCT s.time_slot_id
   FROM teaches t
   JOIN section s ON t.course_id = s.course_id AND t.sec_id = s.sec_id
   WHERE t.instructor_id = i.id
);
```

```
SELECT i.name
FROM instructor i
JOIN teaches t ON i.ID = t.ID
JOIN section sec ON t.course_id = sec.course_id
GROUP BY i.name
HAVING COUNT(DISTINCT sec.time_slot_id) = (SELECT COUNT(DISTINCT time_slot_id)
FROM time_slot);
```

49. List all students who have taken courses with overlapping time slots in different semesters.

```
SELECT DISTINCT s.id, s.name
```

```
FROM student s
JOIN takes t1 ON s.id = t1.id
JOIN section sec1 ON t1.course_id = sec1.course_id AND t1.sec_id = sec1.sec_id
JOIN time_slot ts1 ON sec1.time_slot_id = ts1.time_slot_id
JOIN takes t2 ON s.id = t2.id AND t1.semester <> t2.semester
JOIN section sec2 ON t2.course_id = sec2.course_id AND t2.sec_id = sec2.sec_id
JOIN time_slot ts2 ON sec2.time_slot_id = ts2.time_slot_id
WHERE ts1.time_slot_id = ts2.time_slot_id;

SELECT DISTINCT s.name
FROM takes t1
JOIN takes t2 ON t1.ID = t2.ID AND t1.course_id != t2.course_id
JOIN section s1 ON t1.course_id = s1.course_id
JOIN section s2 ON t2.course_id = s2.course_id
JOIN time_slot ts1 ON s1.time_slot_id = ts1.time_slot_id
JOIN time_slot ts2 ON s2.time_slot_id = ts2.time_slot_id
WHERE t1.semester != t2.semester
AND ts1.day = ts2.day
AND (
  (ts1.start_hr < ts2.end_hr OR (ts1.start_hr = ts2.end_hr AND ts1.start_min < ts2.end_min))
  AND
  (ts2.start_hr < ts1.end_hr OR (ts2.start_hr = ts1.end_hr AND ts2.start_min < ts1.end_min))
);
```

50. Find the courses that have the lowest enrollment across all semesters.

```
SELECT t.course_id, c.title, COUNT(*) AS enrollment
FROM takes t
JOIN course c ON t.course_id = c.course_id
GROUP BY t.course_id, c.title
HAVING COUNT(*) = (
    SELECT MIN(course_count)
    FROM (SELECT course_id, COUNT(*) AS course_count
          FROM takes
          GROUP BY course_id)
);

SELECT course_id
FROM takes
GROUP BY course_id
ORDER BY COUNT(*)
FETCH FIRST 1 ROW WITH TIES;
```