## Activity

## Data Retrieval Operations

**Plain SQL Queries**

1. Display the unique states from ZIPCODE\_INFO table.
2. **SELECT DISTINCT STATE FROM** ZIPCODE\_INFO;
3. Display the student\_id, name. Concatenate the first\_name and last name from STUDENT\_INFO table.
4. **SELECT** STUDENT\_ID, *CONCAT*(STUDENT\_FIRST\_NAME, STUDENT\_LAST\_NAME) **AS NAME FROM** STUDENT\_INFO;
5. From ZIPCODE\_INFO table display the zipcode, city and state as a single column. Separate the data with a comma. E.g 400050, Mumbai, MH. Give the column heading as Address.
6. **SELECT** ZIP\_CODE || ', ' || CITY || ', ' || **STATE AS** ADRESS **from** ZIPCODE\_INFO;
7. Display all the columns of the COURSE\_INFO table.
8. **SELECT** \* **FROM** COURSE\_INFO;

**SELECT Using WHERE**

1. Display the student\_id, section\_id and numeric\_grade of those students who have the grade code type as ‘FI’.
2. **SELECT** STUDENT\_ID, SECTION\_ID, NUMERIC\_GRADE **FROM** GRADE\_INFO  
   **WHERE** GRADE\_TYPE\_CODE = 'FI';
3. Display the section and the instructor id of the course 10 and 20. Sort the data in the ascending order of instructor id.
4. **SELECT** SECTION\_ID, INSTRUCTOR\_ID **FROM** SECTION\_INFO  
   **WHERE** COURSE\_NO = 10 **OR** COURSE\_NO = 20  
   **ORDER BY** INSTRUCTOR\_ID;
5. Display the student\_id, section\_id and numeric\_grade. Sort the section\_id in ascending order followed by numeric\_grade in descending order.
6. **SELECT** student\_id, section\_id, numeric\_grade **FROM** GRADE\_INFO  
   **ORDER BY** section\_id **ASC**, numeric\_grade **DESC**;
7. Display the course\_no, Course\_name and cost of the courses. The courses should have the word ‘Intro’.
8. **SELECT** COURSE\_NO, COURSE\_NAME, COST **FROM** COURSE\_INFO  
   **WHERE** COURSE\_NAME **LIKE** '%Intro%';
9. Display the course details where ‘a’ is the 3rd last letter
10. **SELECT** \* **FROM** COURSE\_INFO   
    **WHERE** COURSE\_NAME **LIKE** '%a\_\_';
11. Display the student names whose student\_id is in the range of 300 to 350.
12. **SELECT** *CONCAT*(STUDENT\_FIRST\_NAME, STUDENT\_LAST\_NAME) **AS** STUDENT\_NAME **FROM** STUDENT\_INFO   
    **WHERE** STUDENT\_ID **BETWEEN** 300 **AND** 350;
13. Write a select statement that displays the instructor's first name whose last name is not 'Schumer'.
14. **SELECT** INSTRUCTOR\_FIRST\_NAME **FROM** INSTRUCTOR\_INFO   
    **WHERE** INSTRUCTOR\_FIRST\_NAME != 'Schumer';
15. Display the course name and cost of those courses whose cost is 4000 or 4500.
16. **SELECT** COURSE\_NAME, COST **FROM** COURSE\_INFO   
    **WHERE** COST = 4000 **OR** COST = 4500;
17. Display the course name and cost of those courses whose cost is in range of 4000 to 4500 and the course name starts with 'I'.
18. **SELECT** COURSE\_NAME, COST **FROM** COURSE\_INFO   
    **WHERE** COST **BETWEEN** 4000 **AND** 4500 **AND** COURSE\_NAME= 'I%';
19. Write a select statement to list the last names of students living either in zip code 10048, 11102, or 11209.
20. **SELECT** \* **FROM** STUDENT\_INFO   
    **WHERE** ZIP\_CODE = 10048 **OR** ZIP\_CODE =11102 **OR** ZIP\_CODE =11209;
21. Write a select statement to list the first and last names of instructors with the letter 'i' in any case in the last name and living in zip code 10025.
22. **SELECT** INSTRUCTOR\_FIRST\_NAME,INSTRUCTOR\_LAST\_NAME **FROM** INSTRUCTOR\_INFO   
    **WHERE** INSTRUCTOR\_LAST\_NAME **LIKE** '%i%' **and** ZIP\_CODE = 10025;
23. Write a select statement to list the cost of courses that do not have a prerequisite. In the result the cost should not be repeated.
24. **SELECT** \* **FROM** COURSE\_INFO   
    **WHERE** COURSE\_PREREQUISITE **IS NULL**;
25. Display the course\_no and the course\_name not including a course\_prerequisite. Sort the data on the basis of course\_name.
26. **SELECT** COURSE\_NO, COURSE\_NAME **FROM** COURSE\_INFO   
    **ORDER BY** COURSE\_NAME;
27. Display the city and state as 1 column, from ZIPCODE\_INFO.
28. **SELECT** CITY || ', ' || **STATE AS** CITY\_AND\_STATE **FROM** ZIPCODE\_INFO;
29. Display the select statement that displays the cost, add 10 to cost, subtract 20 from cost, multiply cost by 30 and divide cost by 5.
30. **SELECT** (COST + 10 -20) \* 30 / 5 **AS** NEW\_COST **FROM** COURSE\_INFO;
31. Write a query to format the cost column. The cost displayed should have a leading $ sign followed by a comma to separate the thousands and should display 2 decimals. E.g $2,000.00
32. **SELECT** '$' || TO\_CHAR(COST, '9G999G999D99') **AS** COST **FROM** COURSE\_INFO;

## Aggregate Functions

1. Display the count of records in the course table.
2. **SELECT** *COUNT*(\*) **FROM** COURSE\_INFO;
3. Display the sum of numeric\_grade from grade\_info.
4. **SELECT** *SUM*(NUMERIC\_GRADE) **FROM** GRADE\_INFO;
5. Display the average, total, minimum, and maximum numeric grade.
6. **SELECT** *AVG*(NUMERIC\_GRADE) **AS** AVERAGE **FROM** GRADE\_INFO;  
   **SELECT** *SUM*(NUMERIC\_GRADE) **AS** TOTAL **FROM** GRADE\_INFO;  
   **SELECT** *MIN*(NUMERIC\_GRADE) **AS** MINIMUM **FROM** GRADE\_INFO;  
   **SELECT** *MAX*(NUMERIC\_GRADE) **AS** MAXIMUM **FROM** GRADE\_INFO;
7. Write a SELECT statement that displays the total number of courses not having a pre-requisite.
8. **SELECT** *COUNT*(COURSE\_NAME) **FROM** COURSE\_INFO  
   **WHERE** COURSE\_PREREQUISITE **IS NULL**;
9. Display the date of the student who was recently enrolled.
10. **SELECT** STUDENT\_FIRST\_NAME || STUDENT\_LAST\_NAME **AS** STUDENT\_NAME, ENROLLMENT\_DATE **FROM** STUDENT\_INFO,ENROLLMENT\_INFO  
    **WHERE** STUDENT\_INFO.STUDENT\_ID = ENROLLMENT\_INFO.STUDENT\_ID  
    **ORDER BY** ENROLLMENT\_DATE **DESC**;

## Group By and Having Clause

1. Display the count of cities for each state.
2. **select state**, *count*(city) **from** zipcode\_info  
   **group by state**;
3. Display the count of students enrolled in each section. Display only those sections where the number of students enrolled is more than 5.
4. **SELECT** SECTION\_ID, *COUNT*(STUDENT\_ID) **AS** STUDENT\_SUM **FROM** ENROLLMENT\_INFO  
   **GROUP BY** SECTION\_ID  
   **HAVING** *COUNT*(STUDENT\_ID) > 5;
5. Display the average numeric grade for each student and section. The average numeric grade should be more than 75 and also display the data for student\_id more than 280.
6. **SELECT** STUDENT\_ID, *AVG*(NUMERIC\_GRADE) **FROM** GRADE\_INFO  
   **WHERE** STUDENT\_ID > 280  
   **GROUP BY** STUDENT\_ID  
   **HAVING** *AVG*(NUMERIC\_GRADE) > 75  
   **UNION  
   SELECT** SECTION\_ID, *AVG*(NUMERIC\_GRADE) **FROM** GRADE\_INFO  
   **GROUP BY** SECTION\_ID  
   **HAVING** *AVG*(NUMERIC\_GRADE) > 75;

## Joins

1. Display the student number, section id, and grade type code, along with numeric grade and the month of enroll date for student id =220 and section id =119.

**SELECT** ENROLLMENT\_INFO.STUDENT\_ID, ENROLLMENT\_INFO.SECTION\_ID,   
GRADE\_TYPE\_CODE, NUMERIC\_GRADE,   
EXTRACT (**MONTH FROM** ENROLLMENT\_INFO.ENROLLMENT\_DATE) **AS MONTH  
FROM** GRADE\_INFO  
**INNER JOIN** ENROLLMENT\_INFO  
**ON** GRADE\_INFO.STUDENT\_ID = ENROLLMENT\_INFO.STUDENT\_ID  
**WHERE** ENROLLMENT\_INFO.STUDENT\_ID = 220 **AND** ENROLLMENT\_INFO.SECTION\_ID = 119;

1. Display the course number, description, and pre-requisite along with section id. Those courses not having sections defined should also be displayed.

**SELECT** COURSE\_INFO.COURSE\_NO, COURSE\_INFO.COURSE\_NAME, COURSE\_INFO.COURSE\_PREREQUISITE, SECTION\_ID  
**FROM** COURSE\_INFO  
**LEFT OUTER JOIN** SECTION\_INFO  
**ON** COURSE\_INFO.COURSE\_NO = SECTION\_INFO.COURSE\_NO;

1. The students with ids 102 and 301 display the student id, section id, numeric grade, and the grade type. Display the details even if the student is not enrolled or has not received any grade.

**SELECT** STUDENT\_INFO.STUDENT\_ID, GRADE\_INFO.SECTION\_ID, GRADE\_INFO.NUMERIC\_GRADE, GRADE\_INFO.GRADE\_TYPE\_CODE  
**FROM** STUDENT\_INFO   
**LEFT OUTER JOIN** GRADE\_INFO  
**ON** STUDENT\_INFO.STUDENT\_ID = GRADE\_INFO.STUDENT\_ID  
**WHERE** STUDENT\_INFO.STUDENT\_ID = 102 **OR** STUDENT\_INFO.STUDENT\_ID = 301;

1. Many courses have a pre-requisite. Display the course number and the description along with the pre-requisite and the description.  
   E.g.: 20 Intro to Java 10 Java Programming where 20 is the course number and 10 is the pre-requisite.

**SELECT** K1.COURSE\_NO || ' Intro to ' ||   
K1.COURSE\_NAME || ' ' ||  
K1.COURSE\_PREREQUISITE || ' ' ||  
K2.COURSE\_NAME || ' Programming'  
**FROM** COURSE\_INFO K1  
**INNER JOIN** COURSE\_INFO K2  
**ON** K1.COURSE\_NO = K2.COURSE\_NO;

1. Display the names of the students living at the same address and zip code.

**SELECT** K1.STUDENT\_FIRST\_NAME || K1.STUDENT\_LAST\_NAME **AS NAME   
FROM** STUDENT\_INFO K1  
**INNER JOIN** STUDENT\_INFO K2  
**ON** K1.STREET\_ADDRESS = K2.STREET\_ADDRESS **AND** K1.ZIP\_CODE = K2.ZIP\_CODE;