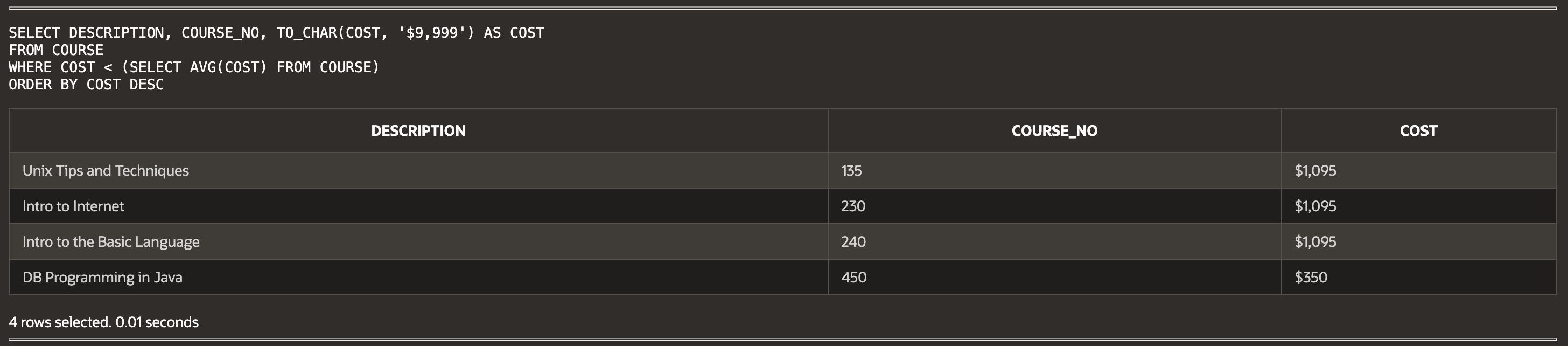
**SDC250 Final Exam**

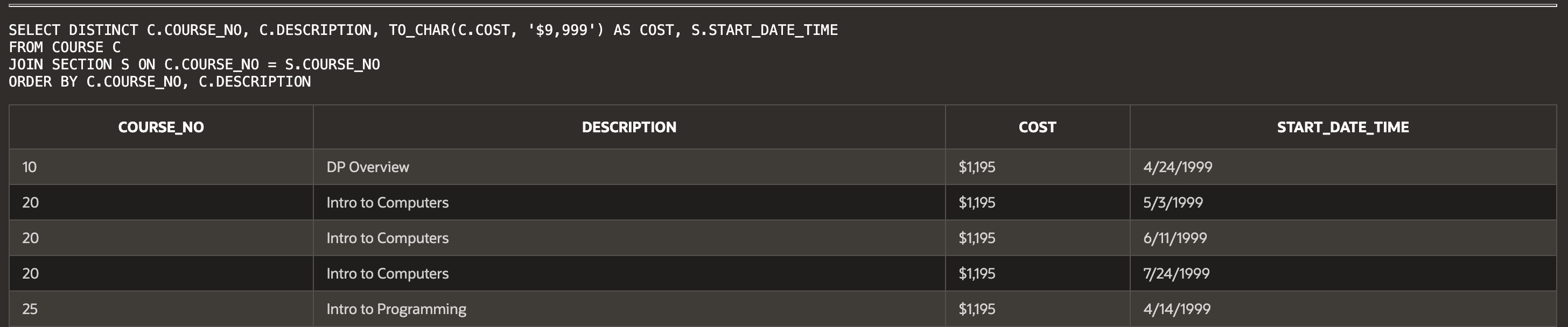
For each question, fill in one SQL statement, and a screen shot of the output. Please paste the answers below the questions in this document.

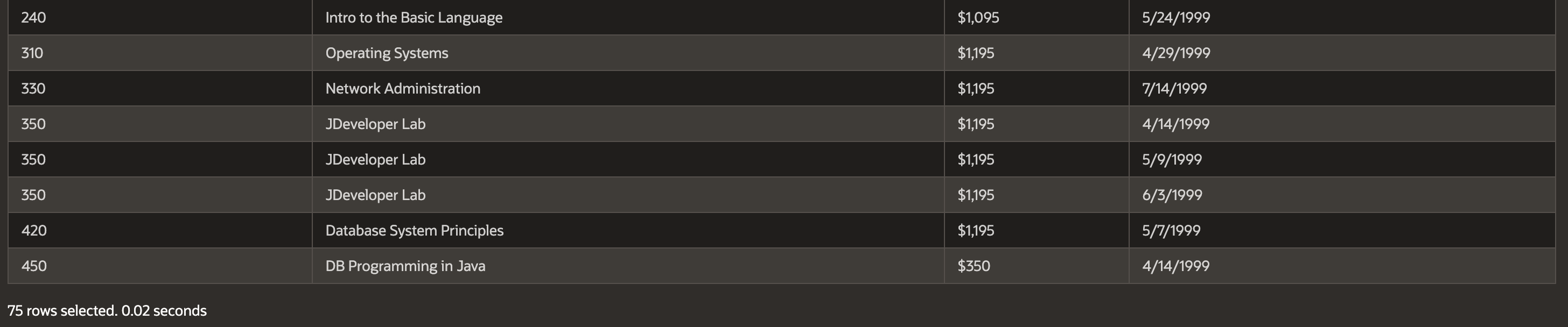
Good Luck!

1. Using the course table, find all the courses that cost less than the average course price. Show description, course\_no, and cost in your results. Order by cost in descending order. Make sure your cost is in $9,999 format <Hint: Use the to\_char statement to format your answer>

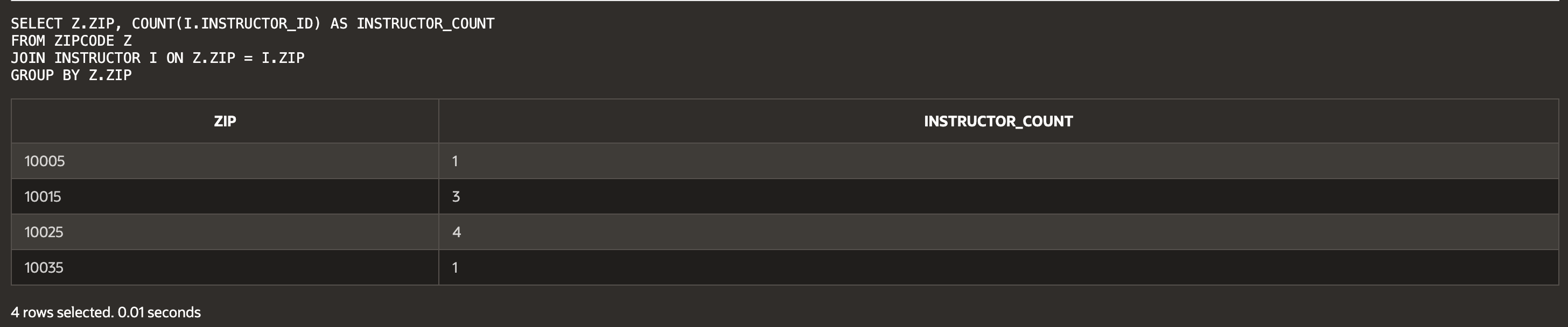


1. Using the course and section tables, list all the courses and their corresponding sections. Show course\_no, description, cost, start\_date\_time in your results. Order your results by course\_no, then description. Make sure that your course\_no, description, and cost do not duplicate in the display of your results.

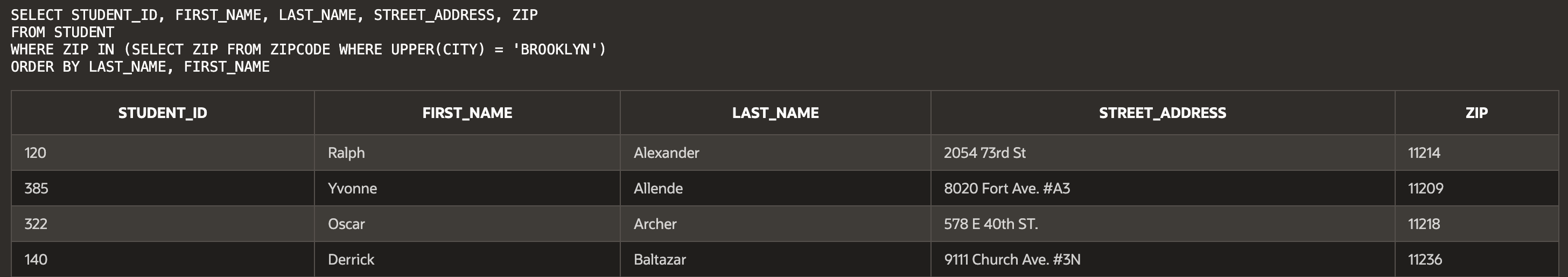
...



1. Using the zipcode and instructor table, show all the zipcodes and the count of instructors that live in those zipcodes.

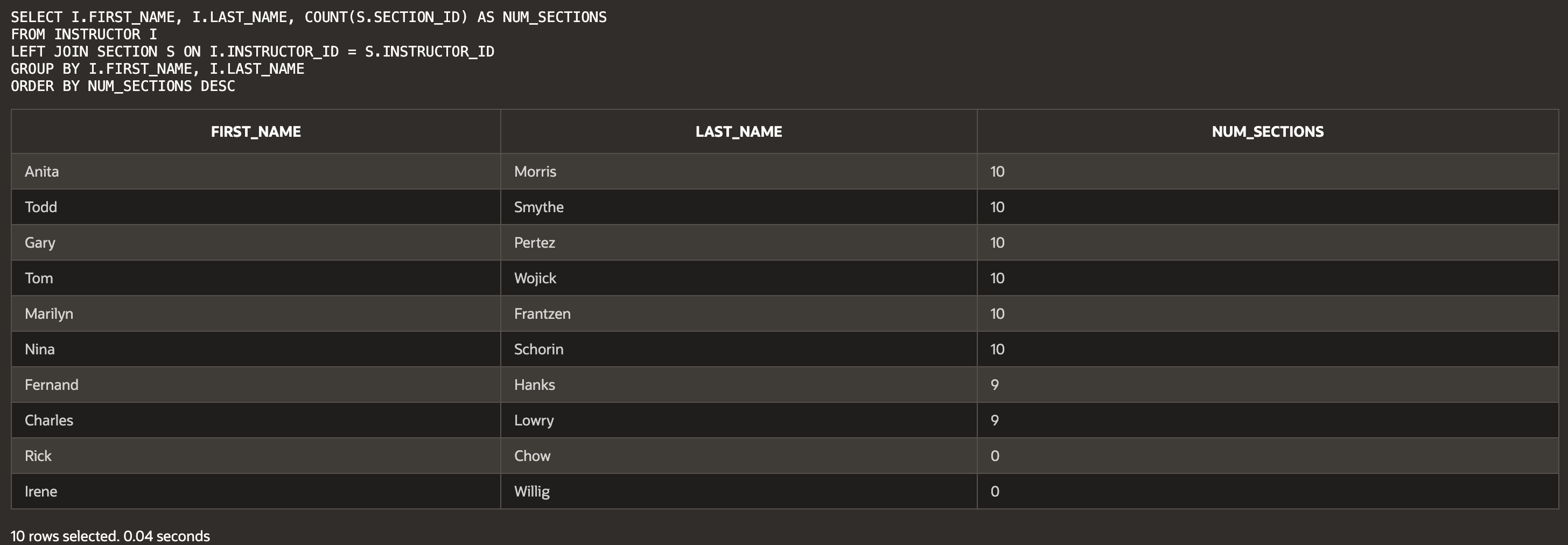


1. Using the zipcode and student table, show all the students that live in Brooklyn. Show student\_id, first\_name, last\_name, street\_address, state, and zip in your results. Order your results by last\_name, then first\_name.

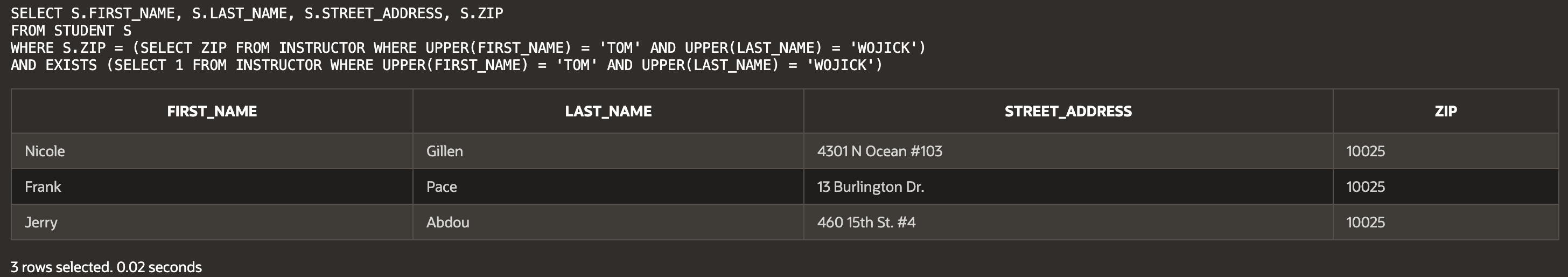
...



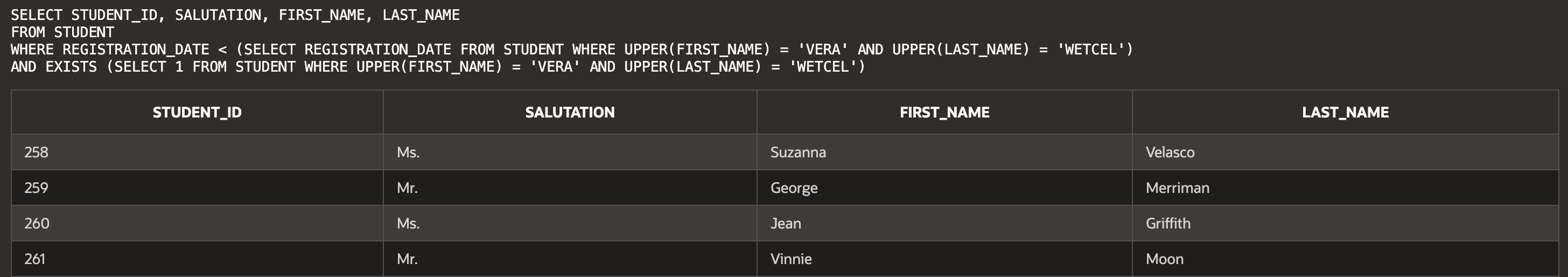
1. Using the instructor and section table, list all the instructors and the number of sections they teach. Show first\_name, last\_name, and the number of sections they teach. Order your results by the number of sections in descending order.



1. List all the students who live in the same zipcode as the instructor Tom Wojick. Show first\_name, last\_name, street\_address and zip in your results.

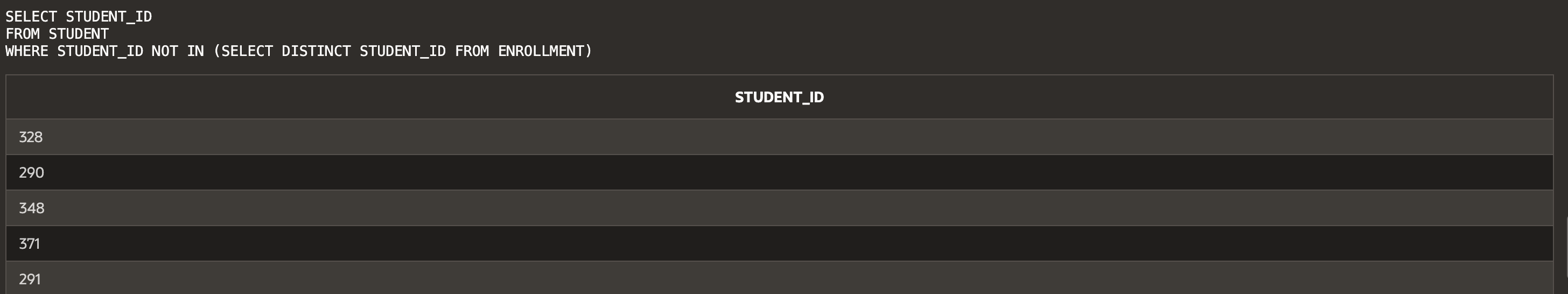


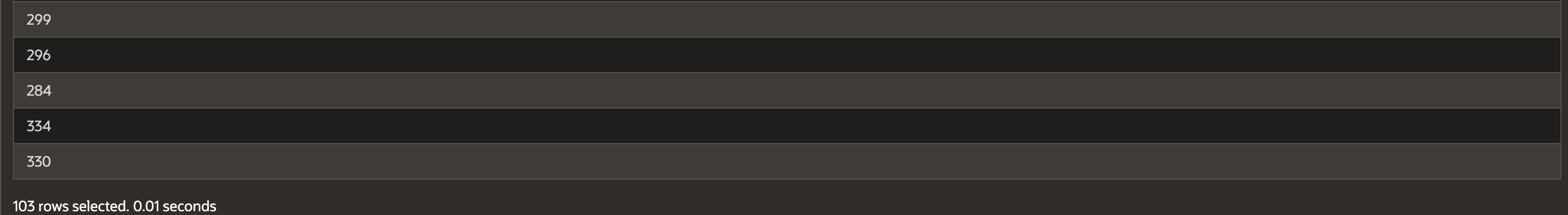
1. List all the students who registered before Vera Wetcel. Show student\_id, salutation, first\_name, and last\_name in your results.

...



1. Using the student and enrollment table, find all the students who haven’t enrolled in any classes. Show student\_id in your results.

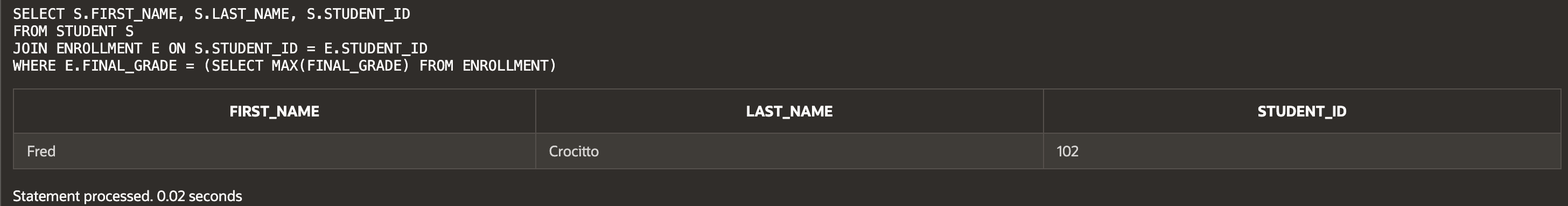
...



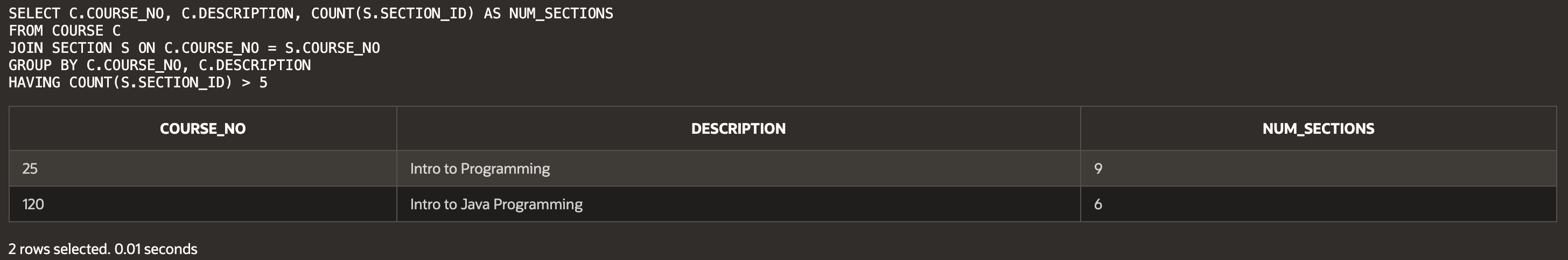
1. Create a view called all\_people\_view that lists all the students and instructors currently present at the university. Show salutation, full\_name, street\_address, zip, phone in your results. The full\_name column should be the “first\_name last\_name” columns from the tables.



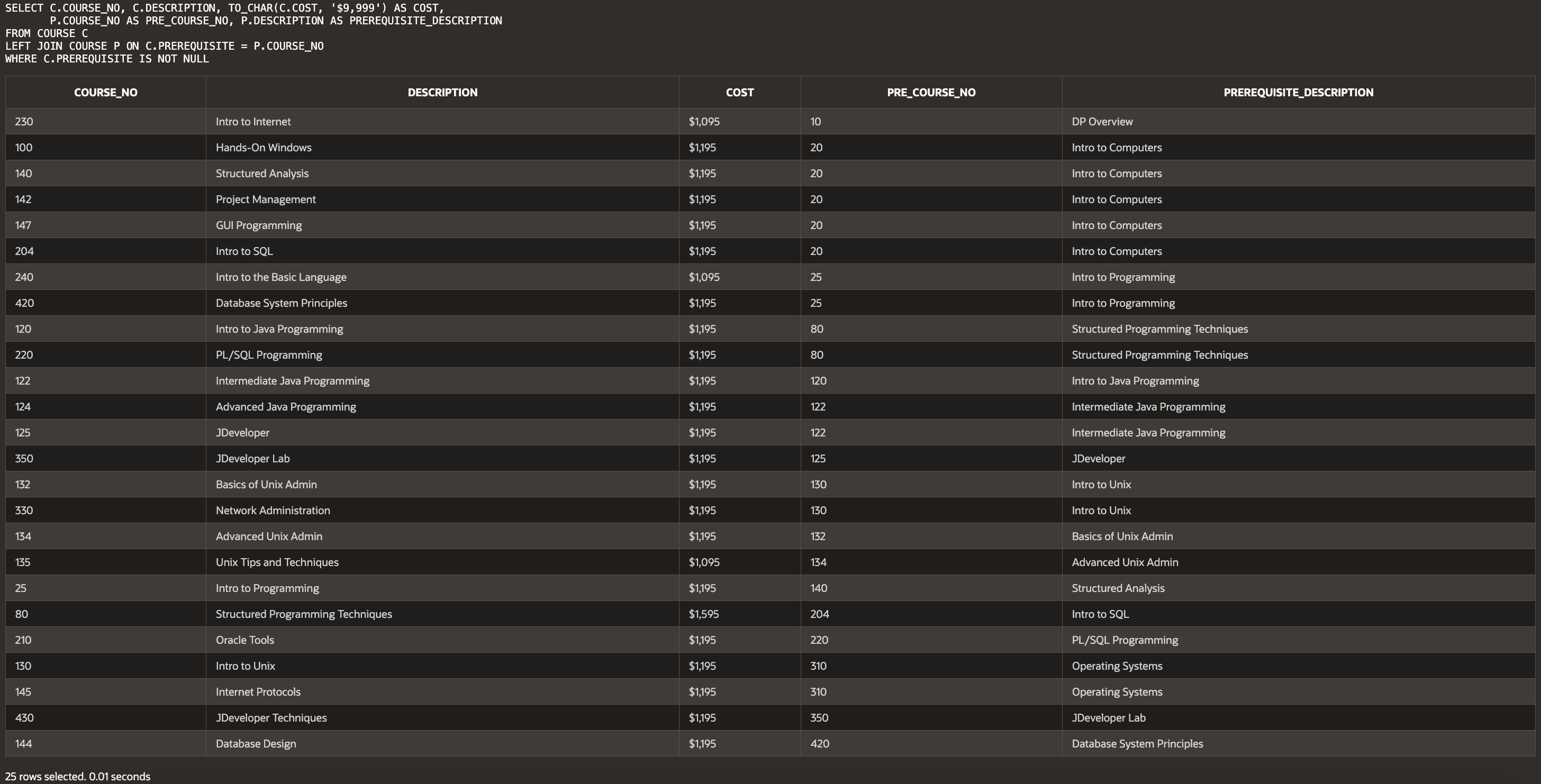
1. From the enrollment and student tables, show the student who got the highest grade. Show first\_name, last\_name, and student\_id in your results.



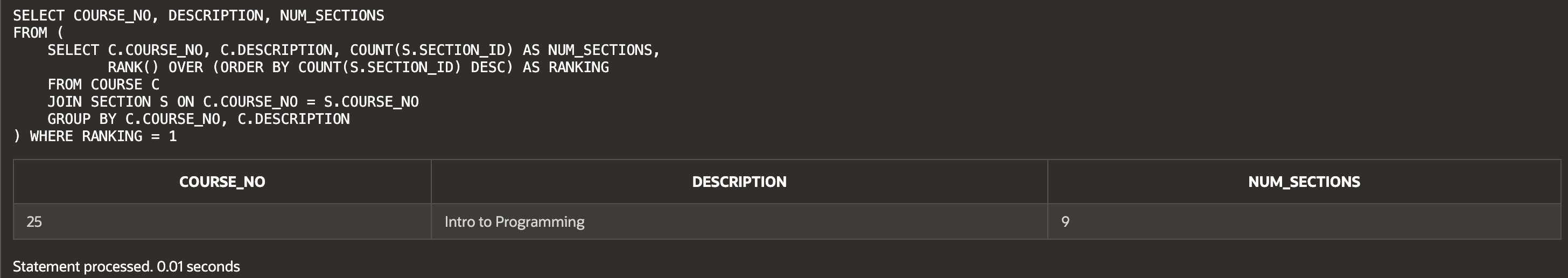
1. Using the course, section tables, list all the courses who have more than 5 sections. Show course\_no, description, and number of sections.



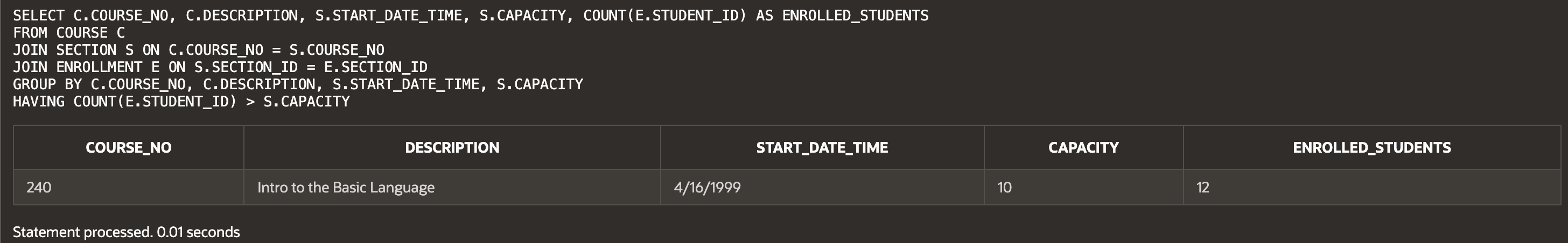
1. List all the courses and their prerequisites (if any). Show course\_no, description, cost of the course, and course\_no, description of the prerequisite.



1. List the course(s) who have the most sections. Show course\_no, description, and number of sections in your answer.



1. Using the course, section, and enrollment tables list all the courses that have more students enrolled than their capacity allows. Show course\_no, description, start\_date\_time, capacity, and current number of enrolled students in your results.



**Script:**

-- Haley Archer

-- 1) Find all courses that cost less than the average course price

SELECT DESCRIPTION, COURSE\_NO, TO\_CHAR(COST, '$9,999') AS COST

FROM COURSE

WHERE COST < (SELECT AVG(COST) FROM COURSE)

ORDER BY COST DESC;

-- 2) List all courses and their corresponding sections, ensuring no duplicates

SELECT DISTINCT C.COURSE\_NO, C.DESCRIPTION, TO\_CHAR(C.COST, '$9,999') AS COST, S.START\_DATE\_TIME

FROM COURSE C

JOIN SECTION S ON C.COURSE\_NO = S.COURSE\_NO

ORDER BY C.COURSE\_NO, C.DESCRIPTION;

-- 3) Show all zip codes and the count of instructors in each zip code

SELECT Z.ZIP, COUNT(I.INSTRUCTOR\_ID) AS INSTRUCTOR\_COUNT

FROM ZIPCODE Z

JOIN INSTRUCTOR I ON Z.ZIP = I.ZIP

GROUP BY Z.ZIP;

-- 4) List all students that live in Brooklyn (Checking ZIP instead of CITY)

SELECT STUDENT\_ID, FIRST\_NAME, LAST\_NAME, STREET\_ADDRESS, ZIP

FROM STUDENT

WHERE ZIP IN (SELECT ZIP FROM ZIPCODE WHERE UPPER(CITY) = 'BROOKLYN')

ORDER BY LAST\_NAME, FIRST\_NAME;

-- 5) List all instructors and the number of sections they teach

SELECT I.FIRST\_NAME, I.LAST\_NAME, COUNT(S.SECTION\_ID) AS NUM\_SECTIONS

FROM INSTRUCTOR I

LEFT JOIN SECTION S ON I.INSTRUCTOR\_ID = S.INSTRUCTOR\_ID

GROUP BY I.FIRST\_NAME, I.LAST\_NAME

ORDER BY NUM\_SECTIONS DESC;

-- 6) List all students who live in the same zipcode as instructor Tom Wojick

SELECT S.FIRST\_NAME, S.LAST\_NAME, S.STREET\_ADDRESS, S.ZIP

FROM STUDENT S

WHERE S.ZIP = (SELECT ZIP FROM INSTRUCTOR WHERE UPPER(FIRST\_NAME) = 'TOM' AND UPPER(LAST\_NAME) = 'WOJICK')

AND EXISTS (SELECT 1 FROM INSTRUCTOR WHERE UPPER(FIRST\_NAME) = 'TOM' AND UPPER(LAST\_NAME) = 'WOJICK');

-- 7) List all students who registered before Vera Wetcel (Ensuring Vera Exists)

SELECT STUDENT\_ID, SALUTATION, FIRST\_NAME, LAST\_NAME

FROM STUDENT

WHERE REGISTRATION\_DATE < (SELECT REGISTRATION\_DATE FROM STUDENT WHERE UPPER(FIRST\_NAME) = 'VERA' AND UPPER(LAST\_NAME) = 'WETCEL')

AND EXISTS (SELECT 1 FROM STUDENT WHERE UPPER(FIRST\_NAME) = 'VERA' AND UPPER(LAST\_NAME) = 'WETCEL');

-- 8) Find all students who haven't enrolled in any classes

SELECT STUDENT\_ID

FROM STUDENT

WHERE STUDENT\_ID NOT IN (SELECT DISTINCT STUDENT\_ID FROM ENROLLMENT);

-- 9) Create a view called all\_people\_view listing all students and instructors

CREATE VIEW ALL\_PEOPLE\_VIEW AS

SELECT SALUTATION, FIRST\_NAME || ' ' || LAST\_NAME AS FULL\_NAME, STREET\_ADDRESS, ZIP, PHONE

FROM STUDENT

UNION

SELECT SALUTATION, FIRST\_NAME || ' ' || LAST\_NAME AS FULL\_NAME, STREET\_ADDRESS, ZIP, PHONE

FROM INSTRUCTOR;

-- 10) Show the student who got the highest grade (Fixing grade column reference)

SELECT S.FIRST\_NAME, S.LAST\_NAME, S.STUDENT\_ID

FROM STUDENT S

JOIN ENROLLMENT E ON S.STUDENT\_ID = E.STUDENT\_ID

WHERE E.FINAL\_GRADE = (SELECT MAX(FINAL\_GRADE) FROM ENROLLMENT);

-- 11) List all courses that have more than 5 sections

SELECT C.COURSE\_NO, C.DESCRIPTION, COUNT(S.SECTION\_ID) AS NUM\_SECTIONS

FROM COURSE C

JOIN SECTION S ON C.COURSE\_NO = S.COURSE\_NO

GROUP BY C.COURSE\_NO, C.DESCRIPTION

HAVING COUNT(S.SECTION\_ID) > 5;

-- 12) List all courses and their prerequisites (Fixing missing table issue)

SELECT C.COURSE\_NO, C.DESCRIPTION, TO\_CHAR(C.COST, '$9,999') AS COST,

P.COURSE\_NO AS PRE\_COURSE\_NO, P.DESCRIPTION AS PREREQUISITE\_DESCRIPTION

FROM COURSE C

LEFT JOIN COURSE P ON C.PREREQUISITE = P.COURSE\_NO

WHERE C.PREREQUISITE IS NOT NULL;

-- 13) List the course(s) with the most sections

SELECT COURSE\_NO, DESCRIPTION, NUM\_SECTIONS

FROM (

SELECT C.COURSE\_NO, C.DESCRIPTION, COUNT(S.SECTION\_ID) AS NUM\_SECTIONS,

RANK() OVER (ORDER BY COUNT(S.SECTION\_ID) DESC) AS RANKING

FROM COURSE C

JOIN SECTION S ON C.COURSE\_NO = S.COURSE\_NO

GROUP BY C.COURSE\_NO, C.DESCRIPTION

) WHERE RANKING = 1;

-- 14) List all courses where the number of students enrolled exceeds capacity

SELECT C.COURSE\_NO, C.DESCRIPTION, S.START\_DATE\_TIME, S.CAPACITY, COUNT(E.STUDENT\_ID) AS ENROLLED\_STUDENTS

FROM COURSE C

JOIN SECTION S ON C.COURSE\_NO = S.COURSE\_NO

JOIN ENROLLMENT E ON S.SECTION\_ID = E.SECTION\_ID

GROUP BY C.COURSE\_NO, C.DESCRIPTION, S.START\_DATE\_TIME, S.CAPACITY

HAVING COUNT(E.STUDENT\_ID) > S.CAPACITY;