NISHARG GOSAI
ASSIGNMENT 2

SQL ANSWERS

1) For each department, find the maximum salary of instructors in that department

```
SELECT dept_name,MAX(salary)
FROM instructor
GROUP BY dept_name;
```

q1

dept_name	MAX(salary)
Biology	72000.00
Comp. Sci.	92000.00
Elec. Eng.	80000.00
Finance	90000.00
History	62000.00
Music	40000.00
Physics	95000.00

2) Find the IDs of all students who were taught by an instructor named Katz; make sure there are no duplicates in the result

```
SELECT DISTINCT takes.ID
FROM (takes join teaches using (course_id,sec_id,semester,year)) join instructor on instructor.ID=teaches.ID
WHERE name = "Katz";
```

q2

ID
45678

3) Find the ID and title of each course in Comp. Sci. that has had at least one section with afternoon hours (i.e., ends at or after 12:00). (You should eliminate duplicates if any.

```
SELECT DISTINCT course_id,course.title
FROM (course join section using(course_id)join time_slot using (time_slot_id))
WHERE course.dept_name="Comp. Sci." AND (time_slot.start_hr>11 OR
time_slot.end_hr>=12);
```

q3

course_id	title
CS-101	Intro. to Computer Science
CS-315	Robotics

4) Find the IDs and titles of all the courses that are prerequisites to the Robotics course.

```
SELECT course_id,title
FROM course
WHERE course_id = (
        SELECT prereq_id FROM prereq WHERE course_id =
        (
                SELECT course_id FROM course WHERE title="Robotics"
        )
);
```

q4

course_id	title
CS-101	Intro. to Computer Science

5) Find the IDs and names of all instructors earning the highest salary (there may be more than one with the same salary)
-Assuming question meant to ask highest salary and not highest salary from each department

```
SELECT id,name
FROM instructor
WHERE salary=(
        SELECT max(salary) FROM instructor
 );
```

q5

id	name
22222	Einstein

6) Find the enrollment (number of students) in each section that was offered in Spring 2017. The result columns should be course id, section id, students num. You do not need to output sections with 0 students.

```
SELECT course_id,sec_id,count(ID) as number_of_students_enrolled
FROM takes natural join section
WHERE semester="Spring" and year=2017
GROUP BY course_id,sec_id;
```

q6

course_id	sec_id	number_of_students_enrolled
CS-190	2	2
EE-181	1	1

7) Rewrite the preceding query, but also output sections with 0 students

```
SELECT course_id,sec_id,count(ID) as number_of_students_enrolled
FROM takes natural right outer join section
WHERE semester="Spring" and year=2017
GROUP BY course_id,sec_id;
```

nq7

course_id	sec_id	number_of_students_enrolled
CS-190	1	0
CS-190	2	2
EE-181	1	1

8) Find the IDs and names of all instructors who have taught at least 3 different courses.

```
SELECT ID,name
FROM instructor
WHERE ID in(
        SELECT ID
        FROM instructor NATURAL JOIN teaches
        GROUP BY teaches.ID
        HAVING (count(DISTINCT course_id))>=3
);
```

nq8

ID	name
10101	Srinivasan

9) Find the ID and name of the student with the highest number of ’A’ grades (there may be more than one such student)

```
WITH A_grade AS (
        SELECT ID, count(grade) AS num_of_A
         FROM takes
        WHERE grade='A'
        GROUP BY ID
        )
SELECT student.ID, student.name
FROM student
JOIN A_grade  on A_grade.ID = student.ID
WHERE A_grade.num_of_a = (
        SELECT max(num_of_A) FROM A_grade
        );
```

q9

ID	name
12345	Shankar

10) Find the ID and name of each History student who has not taken any Music courses

```
SELECT ID,name
FROM student join takes using (ID)
WHERE dept_name = "History" and course_id not like "MU%"
```

q10

ID	name
19991	Brandt

11) Find the ID and name of each instructor who has never given an ’A’ grade in any course she or he has taught. (Instructors who have never taught a course trivially satisfy this condition.)

```
SELECT ID,name
FROM instructor
WHERE name not in (
        SELECT DISTINCT(instructor.name)
        FROM (instructor join teaches using(ID)) join takes
using(course_id,sec_id,semester,year)
        WHERE grade = "A"
);
```

q11

ID	name
12121	Wu
15151	Mozart
22222	Einstein
32343	El Said
33456	Gold
45565	Katz
58583	Califieri
76543	Singh
98345	Kim

12) Rewrite the preceding query, but also ensure that you include only instructors who have given at least one other non-null grade in some course.

```
SELECT DISTINCT instructor.name
FROM (teaches join takes using (course_id,sec_id,semester,year)) join instructor on instructor.ID=teaches.ID
WHERE instructor.name not in (
    SELECT instructor.name
    FROM (teaches join takes using (course_id,sec_id,semester,year)) join instructor on instructor.ID=teaches.ID
  WHERE grade = "A"
    );
```

q12

name
Wu
Mozart
Einstein
El Said
Katz
Kim

13) For each student who have retaken a course at least once (i.e., the student has taken the course at least twice), show the student's ID, name and the course ID.

```
SELECT DISTINCT course_id,name,ID
FROM takes join student using(ID)
GROUP BY course_id,ID
HAVING count(course_id)>=2;
```

q13

course_id	name	ID
CS-101	Levy	45678

14) Find the IDs of those students who have retaken at least three distinct courses at least once (i.e., the student has taken the course at least two times)

```
WITH cte AS(
        SELECT ID,course_id
         FROM takes
        GROUP BY ID,course_id
        HAVING count(course_id)>1
)
SELECT ID
FROM cte
GROUP BY ID
HAVING count(course_id)>2;
```

q1

ID

Its an empty table

15) Find the IDs and names of those instructors who have taught every course in their Department

```
SELECT instructor.id,instructor.name
FROM instructor join teaches using(ID)
GROUP BY instructor.ID,instructor.dept_name
HAVING count(*)=(
        SELECT count(*)
         FROM course
        WHERE course.dept_name = instructor.dept_name
        );
```

q15

id	name
98345	Kim
12121	Wu
32343	El Said
15151	Mozart
22222	Einstein

SQL DML

Write the SQL statements to perform the following operations on the university database:

1. Create a new course ”CS-001” in the Comp. Sci. department, titled ”Weekly Seminar”, with 2 credits.

INSERT INTO course VALUES ('CS-001', 'Weekly Seminar', 'Comp. Sci.', '2');

dmlq1

course_id	title	dept_name	credits
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-001	Weekly Seminar	Comp. Sci.	2
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

2. Create a section of this course in Spring 2022, with sec id of 1, and with the location of this section not yet specified.
-putting null in building,room_number,time_slot since location details not specified,

INSERT INTO section (course_id,sec_id,semester,year) VALUES ('CS-001', '1', 'Spring', '2022');

dmlq2

course_id	sec_id	semester	year	building	room_number	time_slot_id
BIO-101	1	Summer	2017	Painter	514	B
BIO-301	1	Summer	2018	Painter	514	A
CS-001	1	Spring	2022	NULL	NULL	NULL
CS-101	1	Fall	2017	Packard	101	H
CS-101	1	Spring	2018	Packard	101	F
CS-190	1	Spring	2017	Taylor	3128	E
CS-190	2	Spring	2017	Taylor	3128	A
CS-315	1	Spring	2018	Watson	120	D
CS-319	1	Spring	2018	Watson	100	B
CS-319	2	Spring	2018	Taylor	3128	C
CS-347	1	Fall	2017	Taylor	3128	A
EE-181	1	Spring	2017	Taylor	3128	C
FIN-201	1	Spring	2018	Packard	101	B
HIS-351	1	Spring	2018	Painter	514	C
MU-199	1	Spring	2018	Packard	101	D
PHY-101	1	Fall	2017	Watson	100	A

3. Enroll every student in the Comp. Sci. department in the above section.

```
INSERT INTO takes(ID,course_id,sec_id,semester,year)
    SELECT ID, 'CS-001', '1', 'Spring', '2022'
    FROM student
    WHERE dept_name="Comp. Sci.";
```

dmlq3

ID	course_id	sec_id	semester	year	grade
128	CS-001	1	Spring	2022	NULL
128	CS-101	1	Fall	2017	A
128	CS-347	1	Fall	2017	A-
12345	CS-001	1	Spring	2022	NULL
12345	CS-101	1	Fall	2017	C
12345	CS-190	2	Spring	2017	A
12345	CS-315	1	Spring	2018	A
12345	CS-347	1	Fall	2017	A
19991	HIS-351	1	Spring	2018	B
23121	FIN-201	1	Spring	2018	C+
44553	PHY-101	1	Fall	2017	B-
45678	CS-101	1	Fall	2017	F
45678	CS-101	1	Spring	2018	B+
45678	CS-319	1	Spring	2018	B
54321	CS-001	1	Spring	2022	NULL
54321	CS-101	1	Fall	2017	A-
54321	CS-190	2	Spring	2017	B+
55739	MU-199	1	Spring	2018	A-
76543	CS-001	1	Spring	2022	NULL
76543	CS-101	1	Fall	2017	A
76543	CS-319	2	Spring	2018	A
76653	EE-181	1	Spring	2017	C
98765	CS-101	1	Fall	2017	C-
98765	CS-315	1	Spring	2018	B
98988	BIO-101	1	Summer	2017	A
98988	BIO-301	1	Summer	2018	NULL

4. Delete enrollments in the above section where the student's ID is 12345.
-assuming we delete enrollment from cs-001 course only not all courses

DELETE FROM takes
WHERE ID=12345 and course_id="CS-001" and sec_id=1 and semester="Spring" and year=2022;

dmlq4

ID	course_id	sec_id	semester	year	grade
128	CS-001	1	Spring	2022	NULL
128	CS-101	1	Fall	2017	A
128	CS-347	1	Fall	2017	A-
12345	CS-101	1	Fall	2017	C
12345	CS-190	2	Spring	2017	A
12345	CS-315	1	Spring	2018	A
12345	CS-347	1	Fall	2017	A
19991	HIS-351	1	Spring	2018	B
23121	FIN-201	1	Spring	2018	C+
44553	PHY-101	1	Fall	2017	B-
45678	CS-101	1	Fall	2017	F
45678	CS-101	1	Spring	2018	B+
45678	CS-319	1	Spring	2018	B
54321	CS-001	1	Spring	2022	NULL
54321	CS-101	1	Fall	2017	A-
54321	CS-190	2	Spring	2017	B+
55739	MU-199	1	Spring	2018	A-
76543	CS-001	1	Spring	2022	NULL
76543	CS-101	1	Fall	2017	A
76543	CS-319	2	Spring	2018	A
76653	EE-181	1	Spring	2017	C
98765	CS-101	1	Fall	2017	C-
98765	CS-315	1	Spring	2018	B
98988	BIO-101	1	Summer	2017	A
98988	BIO-301	1	Summer	2018	NULL

5. Delete the course CS-001. What happened to the section and enrollments of this course?

DELETE FROM course WHERE course_id = "CS-001"

dmlq5.1

course_id	title	dept_name	credits
BIO-101	Intro. to Biology	Biology	4
BIO-301	Genetics	Biology	4
BIO-399	Computational Biology	Biology	3
CS-101	Intro. to Computer Science	Comp. Sci.	4
CS-190	Game Design	Comp. Sci.	4
CS-315	Robotics	Comp. Sci.	3
CS-319	Image Processing	Comp. Sci.	3
CS-347	Database System Concepts	Comp. Sci.	3
EE-181	Intro. to Digital Systems	Elec. Eng.	3
FIN-201	Investment Banking	Finance	3
HIS-351	World History	History	3
MU-199	Music Video Production	Music	3
PHY-101	Physical Principles	Physics	4

dmlq5.2

course_id	sec_id	semester	year	building	room_number	time_slot_id
BIO-101	1	Summer	2017	Painter	514	B
BIO-301	1	Summer	2018	Painter	514	A
CS-101	1	Fall	2017	Packard	101	H
CS-101	1	Spring	2018	Packard	101	F
CS-190	1	Spring	2017	Taylor	3128	E
CS-190	2	Spring	2017	Taylor	3128	A
CS-315	1	Spring	2018	Watson	120	D
CS-319	1	Spring	2018	Watson	100	B
CS-319	2	Spring	2018	Taylor	3128	C
CS-347	1	Fall	2017	Taylor	3128	A
EE-181	1	Spring	2017	Taylor	3128	C
FIN-201	1	Spring	2018	Packard	101	B
HIS-351	1	Spring	2018	Painter	514	C
MU-199	1	Spring	2018	Packard	101	D
PHY-101	1	Fall	2017	Watson	100	A

dmlq5.3

ID	course_id	sec_id	semester	year	grade
128	CS-101	1	Fall	2017	A
128	CS-347	1	Fall	2017	A-
12345	CS-101	1	Fall	2017	C
12345	CS-190	2	Spring	2017	A
12345	CS-315	1	Spring	2018	A
12345	CS-347	1	Fall	2017	A
19991	HIS-351	1	Spring	2018	B
23121	FIN-201	1	Spring	2018	C+
44553	PHY-101	1	Fall	2017	B-
45678	CS-101	1	Fall	2017	F
45678	CS-101	1	Spring	2018	B+
45678	CS-319	1	Spring	2018	B
54321	CS-101	1	Fall	2017	A-
54321	CS-190	2	Spring	2017	B+
55739	MU-199	1	Spring	2018	A-
76543	CS-101	1	Fall	2017	A
76543	CS-319	2	Spring	2018	A
76653	EE-181	1	Spring	2017	C
98765	CS-101	1	Fall	2017	C-
98765	CS-315	1	Spring	2018	B
98988	BIO-101	1	Summer	2017	A
98988	BIO-301	1	Summer	2018	NULL

Section and enrollment tables for this course were auto-updated and the course was removed due to ON DELETE CASCADE.

SQL DDL

Write SQL DDL commands to create a database for an insurance company according to the following schemas:

person(driver id, name, address)
car(license plate, model, year)
owns(driver id, license plate)
accident(report_number, date, location)
participated(report_number, license_plate, driver_id, damage amount)

Make any reasonable assumptions about data types, and be sure to declare primary and foreign keys.

```
CREATE DATABASE insurance;
USE insurance;

CREATE TABLE person(
driver_id int,
name varchar(255),
address varchar(255),
PRIMARY KEY(driver_id)
);

CREATE TABLE car(
license_plate varchar(255),
model_name varchar(255),
year numeric(4,0) check (year > 1701 and year < 2100),
PRIMARY KEY(license_plate)
);

CREATE TABLE owns(
driver_id int,
license_plate varchar(255),
PRIMARY KEY(driver_id,license_plate),
FOREIGN KEY (driver_id) REFERENCES person(driver_id) ON DELETE CASCADE
ON UPDATE CASCADE,
FOREIGN KEY (license_plate) REFERENCES car(license_plate) ON DELETE
CASCADE ON UPDATE CASCADE
);
```

```
CREATE TABLE accident(
report_num int,
date date,
location varchar(255),
PRIMARY KEY(report_num)
);

CREATE TABLE participated(
report_num int,
license_plate varchar(255),
driver_id int,
damage_amount decimal(10,2),
PRIMARY KEY (report_num,license_plate),
FOREIGN KEY (driver_id) REFERENCES person(driver_id) ON DELETE CASCADE
ON UPDATE CASCADE,
FOREIGN KEY (license_plate) REFERENCES car(license_plate) ON DELETE
CASCADE ON UPDATE CASCADE,
FOREIGN KEY (report_num) REFERENCES accident(report_num) ON DELETE
CASCADE ON UPDATE CASCADE
);
```