KATHMANDU UNIVERSITY

DHULIKHEL, NEPAL

Department of Computer Science & Engineering (DoCSE)

Lab Assignment

COMP-232

Submitted By:

Sambeg Shrestha (45)

Computer Engineering

2nd Year, 2nd Semester

Submitted to:

Mr. Santosh Khanal

4 September, 2020

Q. Based upon the database you created in earlier classes, answer the following queries. You are practicing basic and intermediate SQL.

1. *Find the department names of all instructors.*
2. *Find the names of all instructors in the Computer Science department who have salary greater than $70,000.*
3. *Retrieve the names of all instructors, along with their department names and department building name.*
4. *"For all instructors in the university who have taught some course, find their names and the course ID of all courses they taught."*
5. *"List the names of instructors along with the titles of courses that they teach."*
6. *"Find the names of all instructors whose salary is greater than at least one instructor in the Biology department."*
7. *"Find the names of all departments whose building name includes the substring 'Watson'."*
8. *Find the name of all instructors in the Physics department in the ascending order and secondly in descending order.*
9. *Find the set of all courses taught either in Fall 2009 or in Spring 2010, or both.*
10. *Find the set of all courses taught in the Fall 2009 as well as in Spring 2010*
11. *Find all courses taught in the Fall 2009 semester but not in the Spring 2010 semester.*
12. *Find the total number of instructors who teach a course in the Spring 2010 semester.*
13. *Find the average salary in each department.*
14. *Find the number of instructors in each department who teach a course in the Spring 2010 semester.*
15. *Find the average salary of instructors in those departments where the average salary is more than $42,000.*
16. *For each course section offered in 2009, find the average total credits (tot cred) of all students enrolled in the section, if the section had at least 2 students.*
17. *Find the total number of (distinct) students who have taken course sections taught by the instructor with ID 110011.*
18. *Find the maximum and minimum enrollment across all sections, considering only sections that had some enrollment, don't worry about those that had no students taking that section*
19. *Find all sections that had the maximum enrollment (along with the enrollment), using a subquery.*
20. *Find all courses whose identifier starts with the string "CS-1"*
21. *Update the salary of each instructor to 2times the number of course sections they have taught.*
22. *Display the IDs of all instructors who have never taught a course.*
23. *Find the names of all instructors whose salary is greater than at least one instructor in the Biology department.*
24. *Find the departments that have the highest average salary.*
25. *Find all students who have taken all courses offered in the Biology department*

Answer:

The contents are ordered as follows:

- Queries
- Database Creation
- Inserting Values into Database
- Table Values

Queries on Relational Database

Language used : ***MySQL***

Queries

1. Find the department names of all instructors.

```
SELECT
    DISTINCT dept_name
FROM
    instructor;
```

Output:

```
+------------+
| dept_name  |
+------------+
| Biology    |
| Comp. Sci. |
| Elec. Eng. |
| Finance    |
| History    |
| Music      |
| Physics    |
+------------+
```

2. Find the names of all instructors in the Computer Science department who have salary greater than $70, 000.

```
SELECT
    i.name
FROM
    instructor i
WHERE
```

```
    i.salary > 70000 AND
    dept_name="Comp. Sci.";
```

Output:

```
+--------+
| name   |
+--------+
| Katz   |
| Brandt |
+--------+
```

3. Retrieve the names of all instructors, along with their department names and department building name.

```
SELECT
    i.name,
    i.dept_name,
    d.building
FROM
    instructor i
    INNER JOIN department d ON i.dept_name = d.dept_name;
```

Output:

```
+------------+------------+----------+
| name       | dept_name  | building |
+------------+------------+----------+
| Crick      | Biology    | Watson   |
| Srinivasan | Comp. Sci. | Taylor   |
| Katz       | Comp. Sci. | Taylor   |
| Brandt     | Comp. Sci. | Taylor   |
| Kim        | Elec. Eng. | Taylor   |
| Wu         | Finance    | Painter  |
| Singh      | Finance    | Painter  |
```

```
| El Said    | History    | Painter  |
| Califieri  | History    | Painter  |
| Mozart     | Music      | Packard  |
| Einstein   | Physics    | Watson   |
| Gold       | Physics    | Watson   |
+------------+------------+----------+
```

4. "For all instructors in the university who have taught some course, find their names and the course ID of all courses they taught."

```
SELECT
    i.name,
    t.course_id
FROM
    instructor i
    INNER JOIN teaches t w here i.id = t.id;
```

Output:

```
+------------+-----------+
| name       | course_id |
+------------+-----------+
| Srinivasan | CS-101    |
| Srinivasan | CS-315    |
| Srinivasan | CS-347    |
| Wu         | FIN-201   |
| Mozart     | MU-199    |
| Einstein   | PHY-101   |
| El Said    | HIS-351   |
| Katz       | CS-101    |
| Katz       | CS-319    |
| Crick      | BIO-101   |
| Crick      | BIO-301   |
| Brandt     | CS-190    |
| Brandt     | CS-190    |
```

```
| Brandt     | CS-319    |
| Kim        | EE-181    |
+------------+-----------+
```

5. “List the names of instructors along with the the titles of courses that they teach.”

```
SELECT
    i.name,
    c.title
FROM
    instructor i
    INNER JOIN teaches t
    INNER JOIN course c
WHERE
    i.id = t.id
    AND t.course_id = c.course_id;
```

Output:

```
+------------+----------------------------+
| name       | title                      |
+------------+----------------------------+
| Srinivasan | Intro. to Computer Science |
| Srinivasan | Robotics                   |
| Srinivasan | Database System Concepts   |
| Wu         | Investment Banking         |
| Mozart     | Music Video Production     |
| Einstein   | Physical Principles        |
| El Said    | World History              |
| Katz       | Intro. to Computer Science |
| Katz       | Image Processing           |
| Crick      | Intro. to Biology          |
| Crick      | Genetics                   |
| Brandt     | Game Design                |
| Brandt     | Game Design                |
```

```
| Brandt     | Image Processing           |
| Kim        | Intro. to Digital Systems  |
+------------+----------------------------+
```

6. "Find the names of all instructors whose salary is greater than at least one instructor in the Biology department."

```
SELECT
    i1.name
FROM
    instructor i1
    INNER JOIN instructor i2 ON(i2.dept_name = "biology")
WHERE
    i1.salary > i2.salary;
```

Output:

```
+----------+
| name     |
+----------+
| Wu       |
| Einstein |
| Gold     |
| Katz     |
| Singh    |
| Brandt   |
| Kim      |
+----------+
```

7. "Find the names of all departments whose building name includes the substring 'Watson'."

```
SELECT
    d.dept_name
FROM
    department d
```

```
WHERE
    d.building = "watson";
```

Output:

```
+------------+
| dept_name |
+------------+
| Biology   |
| Physics   |
+------------+
```

8. Find the name of all instructors in the Physics department in the ascending order and secondly in descending order

```
SELECT
    i.name
FROM
    instructor i
WHERE
    i.dept_name = "physics"
ORDER BY
    name ASC;
```

Output:

```
+----------+
| name     |
+----------+
| Einstein |
| Gold     |
+----------+
```

```
SELECT
    i.name
```

```
FROM
    instructor i
WHERE
    i.dept_name = "physics"
ORDER BY
    name DESC;
```

Output:

```
+----------+
| name     |
+----------+
| Gold     |
| Einstein |
+----------+
```

9. Find the set of all courses taught either in Fall 2009 or in Spring 2010, or both.

```
SELECT
    *
FROM
    section s1
WHERE
    (
        s1.semester = "fall"
        AND s1.year = 2009
    )
UNION
SELECT
    *
FROM
    section s2
WHERE
    (
        s2.semester = "spring"
```

```
        AND s2.year = 2010
    );
```

Output:

```
+-----------+--------+----------+------+----------+-------------+--------------+
| course_id | sec_id | semester | year | building | room_number | time_slot_id |
+-----------+--------+----------+------+----------+-------------+--------------+
| CS-101    | 1      | Fall     | 2009 | Packard  | 101         | H            |
| CS-347    | 1      | Fall     | 2009 | Taylor   | 3128        | A            |
| PHY-101   | 1      | Fall     | 2009 | Watson   | 100         | A            |
| CS-101    | 1      | Spring   | 2010 | Packard  | 101         | F            |
| CS-315    | 1      | Spring   | 2010 | Watson   | 120         | D            |
| CS-319    | 1      | Spring   | 2010 | Watson   | 100         | B            |
| CS-319    | 2      | Spring   | 2010 | Taylor   | 3128        | C            |
| FIN-201   | 1      | Spring   | 2010 | Packard  | 101         | B            |
| HIS-351   | 1      | Spring   | 2010 | Painter  | 514         | C            |
| MU-199    | 1      | Spring   | 2010 | Packard  | 101         | D            |
+-----------+--------+----------+------+----------+-------------+--------------+
```

10. Find the set of all courses taught in the Fall 2009 as well as in Spring 2010

```
SELECT
    s1.course_id
FROM
    section s1
    INNER JOIN section s2 ON (
        s1.course_id = s2.course_id
        AND s1.semester != s2.semester
    )
WHERE
    (
        s1.semester = "fall"
        AND s1.year = 2009
    )
```

```
    AND (
        s2.semester = "spring"
        AND s2.year = 2010
    );
```

Output:

```
+-----------+
| course_id |
+-----------+
| CS-101    |
+-----------+
```

11. Find all courses taught in the Fall 2009 semester but not in the Spring 2010 semester.

```
SELECT
    s1.course_id
FROM
    section s1
WHERE
    (
        s1.semester = "fall"
        AND s1.year = 2009
    )
    AND course_id NOT IN (
        SELECT
            course_id
        FROM
            section s2
        WHERE
            (
                s2.semester = "spring"
                AND s2.year = 2010
            )
    );
```

Output:

```
+-----------+
| course_id |
+-----------+
| CS-347    |
| PHY-101   |
+-----------+
```

12. Find the total number of instructors who teach a course in the Spring 2010 semester.

```
SELECT
    COUNT(DISTINCT id)
FROM
    teaches t
WHERE
    t.semester = "spring"
    AND t.year = 2010;
```

Output:

```
+----------------------+
| count( distinct id) |
+----------------------+
|                    6 |
+----------------------+
```

13. Find the average salary in each department.

```
SELECT
    d.dept_name,
    AVG(salary)
FROM
    department d
```

```
    INNER JOIN instructor i ON (d.dept_name = i.dept_name)
GROUP BY
    d.dept_name;
```

Output:

```
+------------+--------------+
| dept_name  | avg(salary)  |
+------------+--------------+
| Biology    | 72000.000000 |
| Comp. Sci. | 77333.333333 |
| Elec. Eng. | 80000.000000 |
| Finance    | 85000.000000 |
| History    | 61000.000000 |
| Music      | 40000.000000 |
| Physics    | 91000.000000 |
+------------+--------------+
```

14. Find the number of instructors in each department who teach a course in the Spring 2010 semester

```
SELECT
    i.dept_name,
    COUNT(DISTINCT i.id)
FROM
    teaches t
    JOIN instructor i ON (t.id = i.id)
WHERE
    t.semester = "spring"
    AND t.year = "2010"
GROUP BY
    i.dept_name;
```

Output:

```
+------------+----------------------+
| dept_name  | count(distinct i.id) |
+------------+----------------------+
| Comp. Sci. |                    3 |
| Finance    |                    1 |
| History    |                    1 |
| Music      |                    1 |
+------------+----------------------+
```

15. Find the average salary of instructors in those departments where the average salary is more than $42, 000.

```sql
-- query 1 to get avg dept salary
SELECT
    i.dept_name,
    AVG(salary) AS dept_avg
FROM
    instructor i
GROUP BY
    i.dept_name
HAVING
    dept_avg > 42000;
```

Output:

```
+------------+--------------+
| dept_name  | dept_avg     |
+------------+--------------+
| Biology    | 72000.000000 |
| Comp. Sci. | 77333.333333 |
| Elec. Eng. | 80000.000000 |
| Finance    | 85000.000000 |
| History    | 61000.000000 |
| Physics    | 91000.000000 |
+------------+--------------+
```

16. For each course section offered in 2009, find the average total credits (tot cred) of all students enrolled in the section, if the section had at least 2 students.

```
SELECT
    stu_cred_course2009.course_id,
    AVG(stu_cred_course2009.tot_cred)
FROM
    (
        SELECT
            t.id as s_id,
            t.sec_id,
            t.course_id,
            t.year,
            s.tot_cred
        FROM
            takes t
            INNER JOIN student s ON (s.id = t.id)
        WHERE
            t.year = 2009
    ) AS stu_cred_course2009
    INNER JOIN (
        SELECT
            t.sec_id,
            COUNT(sec_id) AS tot_stu
        FROM
            takes t
        GROUP BY
            sec_id
        HAVING
            COUNT(sec_id) > 2
    ) AS sec_with_gt2_stu ON (
        stu_cred_course2009.sec_id = sec_with_gt2_stu.sec_id
    )
GROUP BY
    stu_cred_course2009.course_id;
```

NOTE:

- *stu_cred_course2009* is used to find credit of students for courses offered in 2009
- *sec_with_gt2_stu* is used to find section with enrollment of more than 2 student

Output:

```
+-----------+---------------+
| course_id | avg(tot_cred) |
+-----------+---------------+
| BIO-101   |      120.0000 |
| CS-101    |       65.0000 |
| CS-347    |       67.0000 |
| EE-181    |       60.0000 |
| PHY-101   |       56.0000 |
+-----------+---------------+
```

17. Find the total number of (distinct) students who have taken course sections taught by the instructor with ID 110011

```
SELECT
    course_id
FROM
    teaches
WHERE
    id = 110011;
-- Empty set

-- Since no instructor with id 110011 lets take 10101
SELECT
    s.id AS s_id,
    t.id AS t_id,
    s.course_id
FROM
    takes s
    INNER JOIN teaches t ON (s.course_id = t.course_id)
WHERE
```

```
    t.id = 10101;

+-------+-------+-----------+
| s_id  | t_id  | course_id |
+-------+-------+-----------+
| 00128 | 10101 | CS-101    |
| 12345 | 10101 | CS-101    |
| 45678 | 10101 | CS-101    |
| 54321 | 10101 | CS-101    |
| 76543 | 10101 | CS-101    |
| 98765 | 10101 | CS-101    |
| 45678 | 10101 | CS-101    |
| 12345 | 10101 | CS-315    |
| 98765 | 10101 | CS-315    |
| 00128 | 10101 | CS-347    |
| 12345 | 10101 | CS-347    |
+-------+-------+-----------+

-- So, we have data for id 10101 thus counting students.

SELECT
    COUNT(DISTINCT s.id)
FROM
    takes s
    INNER JOIN teaches t ON (s.course_id = t.course_id)
WHERE
    t.id = 10101;
```

Output:

```
+----------------------+
| count(distinct s.id) |
+----------------------+
|                    6 |
+----------------------+
```

18. Find the maximum and minimum enrollment across all sections, considering only sections that had some enrollment, don't worry about those that had no students taking that section

```
SELECT
    MAX(st.sec_stu) AS MAX,
    MIN(st.sec_stu) AS MIN
FROM
    (
        SELECT
            sec_id,
            COUNT(sec_id) sec_stu
        FROM
            takes
        GROUP BY
            sec_id
        HAVING
            COUNT(sec_id) > 0
    ) AS st;
```

Output:

```
+------+------+
| max  | min  |
+------+------+
|   19 |    3 |
+------+------+
```

19. Find all sections that had the maximum enrollment (along with the enrollment), using a subquery.

```
CREATE view stu_count_secwise AS (
    SELECT
        t.sec_id,
        COUNT(t.sec_id) AS count_stu
    FROM
        takes t
```

```
    GROUP BY
        t.sec_id
);
-- THIS GIVES US ENROLLMENT PER SECTION
+--------+-----------+
| sec_id | count_stu |
+--------+-----------+
| 1      | 19        |
| 2      | 3         |
+--------+-----------+

SELECT
    s.sec_id,
    s.tot_stu
FROM
    stu_coun_secwise s
WHERE
    s.tot_stu IN (
        SELECT
            MAX(tot_stu)
        FROM
            stu_coun_secwise
    );
```

Output:

```
+--------+---------+
| sec_id | tot_stu |
+--------+---------+
| 1      |      19 |
+--------+---------+
```

20. Find all courses whose identifier starts with the string “CS-1”

```
SELECT
    *
FROM
    course
WHERE
    course_id LIKE 'CS-1%';
```

Output:

```
+-----------+----------------------------+------------+---------+
| course_id | title                      | dept_name  | credits |
+-----------+----------------------------+------------+---------+
| CS-101    | Intro. to Computer Science | Comp. Sci. |       4 |
| CS-190    | Game Design                | Comp. Sci. |       4 |
+-----------+----------------------------+------------+---------+
```

21. Update the salary of each instructor to 2 times the number of course sections they have taught.

```
CREATE view new_salary_table AS
SELECT
    i.id,
    i.salary,
    tcc.multiple,
    i.salary * tcc.multiple AS new_salary
FROM
    instructor i
    JOIN (
        SELECT
            id,
            COUNT(id) * 2 AS multiple
        FROM
            teaches
        GROUP BY
            id
    ) AS tcc;
```

```
-- SELECT * FROM new_salary_table;
+-------+----------+----------+------------+
| id    | salary   | multiple | new_salary |
+-------+----------+----------+------------+
| 10101 | 65000.00 |        6 |  390000.00 |
| 12121 | 90000.00 |        2 |  180000.00 |
| 15151 | 40000.00 |        2 |   80000.00 |
| 22222 | 95000.00 |        2 |  190000.00 |
| 32343 | 60000.00 |        2 |  120000.00 |
| 45565 | 75000.00 |        4 |  300000.00 |
| 76766 | 72000.00 |        4 |  288000.00 |
| 83821 | 92000.00 |        6 |  552000.00 |
| 98345 | 80000.00 |        2 |  160000.00 |
+-------+----------+----------+------------+

UPDATE
    instructor i
    INNER JOIN new_salary_table n
SET
    i.salary = n.new_salary
WHERE
    i.id = n.id;
```

Output:

```
-- select * from instructor;
+-------+------------+------------+-----------+
| id    | name       | dept_name  | salary    |
+-------+------------+------------+-----------+
| 10101 | Srinivasan | Comp. Sci. | 390000.00 |
| 12121 | Wu         | Finance    | 180000.00 |
| 15151 | Mozart     | Music      |  80000.00 |
| 22222 | Einstein   | Physics    | 190000.00 |
```

```
| 32343 | El Said    | History    | 120000.00 |
| 33456 | Gold       | Physics    |  87000.00 |
| 45565 | Katz       | Comp. Sci. | 300000.00 |
| 58583 | Califieri  | History    |  62000.00 |
| 76543 | Singh      | Finance    |  80000.00 |
| 76766 | Crick      | Biology    | 288000.00 |
| 83821 | Brandt     | Comp. Sci. | 552000.00 |
| 98345 | Kim        | Elec. Eng. | 160000.00 |
+-------+------------+------------+-----------+
```

22. Display the IDs of all instructors who have never taught a courses.

```
SELECT
    *
FROM
    instructor i
WHERE
    i.id NOT IN (
        SELECT
            teaches.id
        FROM
            teaches
    );
```

Output:

```
+-------+-----------+-----------+----------+
| id    | name      | dept_name | salary   |
+-------+-----------+-----------+----------+
| 33456 | Gold      | Physics   | 87000.00 |
| 58583 | Califieri | History   | 62000.00 |
| 76543 | Singh     | Finance   | 80000.00 |
+-------+-----------+-----------+----------+
```

23. Find the names of all instructors whose salary is greater than at least one instructor in the Biology department.

```
SELECT
    i1.name
FROM
    instructor i1
    LEFT JOIN instructor i2 ON (i2.dept_name = "biology")
WHERE
    i1.salary > i2.salary;


/*
SELECT
    i1.name,
    i1.dept_name,
    i1.salary
FROM
    instructor i1
    LEFT JOIN instructor i2 ON (i2.dept_name = "biology")
WHERE
    i1.salary > i2.salary;

+----------+------------+----------+
| name     | dept_name  | salary   |
+----------+------------+----------+
| Wu       | Finance    | 90000.00 |
| Einstein | Physics    | 95000.00 |
| Gold     | Physics    | 87000.00 |
| Katz     | Comp. Sci. | 75000.00 |
| Singh    | Finance    | 80000.00 |
| Brandt   | Comp. Sci. | 92000.00 |
| Kim      | Elec. Eng. | 80000.00 |
+----------+------------+----------+
*/
```

Note: Query performed before data update from q21

Output:

```
+----------+
| name     |
+----------+
| Wu       |
| Einstein |
| Gold     |
| Katz     |
| Singh    |
| Brandt   |
| Kim      |
+----------+
```

24. Find the departments that have the highest average salary.

```
CREATE view avg_dept_salary AS (
    SELECT
        dept_name,
        AVG(salary) AS avg_salary
    FROM
        instructor
    GROUP BY
        dept_name
);

SELECT
    tbl1.dept_name,
    tbl1.avg_salary
FROM
    avg_dept_salary AS tbl1
WHERE
    tbl1.avg_salary IN (
```

```
        SELECT
            MAX(avg_salary)
        FROM
            avg_dept_salary
    );
```

Output:

```
+-----------+--------------+
| dept_name | avg_salary   |
+-----------+--------------+
| Physics   | 91000.000000 |
+-----------+--------------+
```

25. Find all students who have taken all courses offered in the Biology department.

```
SELECT
    *
FROM
    student s
WHERE
    s.id IN (
        SELECT
            student.id
        FROM
            student
            INNER JOIN takes ON (student.id = takes.id)
        WHERE
            takes.course_id IN (
                SELECT
                    course_id
                FROM
                    course
                WHERE
                    course.dept_name = "biology"
```

```
            )
    );
```

Output:

```
+-------+--------+-----------+----------+
| id    | name   | dept_name | tot_cred |
+-------+--------+-----------+----------+
| 98988 | Tanaka | Biology   |      120 |
+-------+--------+-----------+----------+
```

Database Creation

```
CREATE database lab;
USE lab;
CREATE TABLE classroom (
    building VARCHAR(15),
    room_number VARCHAR(7),
    capacity NUMERIC(4, 0),
    PRIMARY key (building, room_number)
);

CREATE TABLE department (
    dept_name VARCHAR(20),
    building VARCHAR(15),
    budget NUMERIC(12, 2) CHECK (budget > 0),
    PRIMARY key (dept_name)
);

CREATE TABLE course (
    course_id VARCHAR(8),
    title VARCHAR(50),
    dept_name VARCHAR(20),
    credits NUMERIC(2, 0) CHECK (credits > 0),
```

```
    PRIMARY key (course_id),
    FOREIGN key (dept_name) REFERENCES department(dept_name) ON
DELETE
    SET
        NULL
);

CREATE TABLE instructor (
    id VARCHAR(5),
    name VARCHAR(20) NOT NULL,
    dept_name VARCHAR(20),
    salary NUMERIC(8, 2) CHECK (salary > 29000),
    PRIMARY key (id),
    FOREIGN key (dept_name) REFERENCES department(dept_name) ON
DELETE
    SET
        NULL
);

CREATE TABLE section (
    course_id VARCHAR(8),
    sec_id VARCHAR(8),
    semester VARCHAR(6) CHECK (
        semester IN ('Fall', 'Winter', 'Spring', 'Summer')
    ),
    year NUMERIC(4, 0) CHECK (
        year > 1701
        AND year < 2100
    ),
    building VARCHAR(15),
    room_number VARCHAR(7),
    time_slot_id VARCHAR(4),
    PRIMARY key (course_id, sec_id, semester, year),
    FOREIGN key (course_id) REFERENCES course(course_id) ON
DELETE CASCADE,
    FOREIGN key (building, room_number) REFERENCES classroom(building, room_number)
```

```
ON
DELETE
    SET
        NULL
);

CREATE TABLE teaches (
    id VARCHAR(5),
    course_id VARCHAR(8),
    sec_id VARCHAR(8),
    semester VARCHAR(6),
    year NUMERIC(4, 0),
    PRIMARY key (id, course_id, sec_id, semester, year),
    FOREIGN key (course_id, sec_id, semester, year) REFERENCES section(course_id, s
ec_id, semester, year) ON
DELETE CASCADE,
    FOREIGN key (id) REFERENCES instructor(id) ON
DELETE CASCADE
);

CREATE TABLE student (
    id VARCHAR(5),
    name VARCHAR(20) NOT NULL,
    dept_name VARCHAR(20),
    tot_cred NUMERIC(3, 0) CHECK (tot_cred >= 0),
    PRIMARY key (id),
    FOREIGN key (dept_name) REFERENCES department(dept_name) ON
DELETE
    SET
        NULL
);

CREATE TABLE takes (
    id VARCHAR(5),
    course_id VARCHAR(8),
    sec_id VARCHAR(8),
```

```
    semester VARCHAR(6),
    year NUMERIC(4, 0),
    grade VARCHAR(2),
    PRIMARY key (id, course_id, sec_id, semester, year),
    FOREIGN key (course_id, sec_id, semester, year) REFERENCES section(course_id, s
ec_id, semester, year) ON
DELETE CASCADE,
    FOREIGN key (id) REFERENCES student(id) ON
DELETE CASCADE
);

CREATE TABLE advisor (
    s_id VARCHAR(5),
    i_id VARCHAR(5),
    PRIMARY key (s_id),
    FOREIGN key (i_id) REFERENCES instructor (id) ON
DELETE
    SET
        NULL,
        FOREIGN key (s_id) REFERENCES student (id) ON
DELETE CASCADE
);

CREATE TABLE time_slot (
    time_slot_id VARCHAR(4),
    DAY VARCHAR(1),
    start_hr NUMERIC(2) CHECK (
        start_hr >= 0
        AND start_hr < 24
    ),
    start_min NUMERIC(2) CHECK (
        start_min >= 0
        AND start_min < 60
    ),
    end_hr NUMERIC(2) CHECK (
        end_hr >= 0
```

```
        AND end_hr < 24
    ),
    end_min NUMERIC(2) CHECK (
        end_min >= 0
        AND end_min < 60
    ),
    PRIMARY key (time_slot_id, DAY, start_hr, start_min)
);

CREATE TABLE prereq (
    course_id VARCHAR(8),
    prereq_id VARCHAR(8),
    PRIMARY key (course_id, prereq_id),
    FOREIGN key (course_id) REFERENCES course(course_id) ON
DELETE CASCADE,
    FOREIGN key (prereq_id) REFERENCES course(course_id)
);
```

Inserting values into database

`

```

INSERT INTO
    classroom
VALUES
    ('Packard', '101', '500'),
    ('Painter', '514', '10'),
    ('Taylor', '3128', '70'),
    ('Watson', '100', '30'),
    ('Watson', '120', '50');

INSERT INTO
    department
VALUES
```

```
    ('Biology', 'Watson', '90000'),
    ('Comp. Sci.', 'Taylor', '100000'),
    ('Elec. Eng.', 'Taylor', '85000'),
    ('Finance', 'Painter', '120000'),
    ('History', 'Painter', '50000'),
    ('Music', 'Packard', '80000'),
    ('Physics', 'Watson', '70000');

INSERT INTO
    course
VALUES
    ('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');

INSERT INTO
    instructor
VALUES
    ('10101', 'Srinivasan', 'Comp. Sci.', '65000'),
    ('12121', 'Wu', 'Finance', '90000'),
    ('15151', 'Mozart', 'Music', '40000'),
    ('22222', 'Einstein', 'Physics', '95000'),
    ('32343', 'El Said', 'History', '60000'),
    ('33456', 'Gold', 'Physics', '87000'),
    ('45565', 'Katz', 'Comp. Sci.', '75000'),
    ('58583', 'Califieri', 'History', '62000'),
```

```
    ('76543', 'Singh', 'Finance', '80000'),
    ('76766', 'Crick', 'Biology', '72000'),
    ('83821', 'Brandt', 'Comp. Sci.', '92000'),
    ('98345', 'Kim', 'Elec. Eng.', '80000');

INSERT INTO
    section
VALUES
    ( 'BIO-101', '1', 'Summer', '2009', 'Painter', '514', 'B'),
    ( 'BIO-301', '1', 'Summer', '2010', 'Painter', '514', 'A'),
    ( 'CS-101', '1', 'Fall', '2009', 'Packard', '101', 'H'),
    ( 'CS-101', '1', 'Spring', '2010', 'Packard', '101', 'F'),
    ( 'CS-190', '1', 'Spring', '2009', 'Taylor', '3128', 'E'),
    ( 'CS-190', '2', 'Spring', '2009', 'Taylor', '3128', 'A'),
    ( 'CS-315', '1', 'Spring', '2010', 'Watson', '120', 'D'),
    ( 'CS-319', '1', 'Spring', '2010', 'Watson', '100', 'B'),
    ( 'CS-319', '2', 'Spring', '2010', 'Taylor', '3128', 'C'),
    ( 'CS-347', '1', 'Fall', '2009', 'Taylor', '3128', 'A'),
    ( 'EE-181', '1', 'Spring', '2009', 'Taylor', '3128', 'C'),
    ( 'FIN-201', '1', 'Spring', '2010', 'Packard', '101', 'B'),
    ( 'HIS-351', '1', 'Spring', '2010', 'Painter', '514', 'C'),
    ( 'MU-199', '1', 'Spring', '2010', 'Packard', '101', 'D'),
    ( 'PHY-101', '1', 'Fall', '2009', 'Watson', '100', 'A');

INSERT INTO
    teaches
VALUES
    ('10101', 'CS-101', '1', 'Fall', '2009'),
    ('10101', 'CS-315', '1', 'Spring', '2010'),
    ('10101', 'CS-347', '1', 'Fall', '2009'),
    ('12121', 'FIN-201', '1', 'Spring', '2010'),
    ('15151', 'MU-199', '1', 'Spring', '2010'),
    ('22222', 'PHY-101', '1', 'Fall', '2009'),
    ('32343', 'HIS-351', '1', 'Spring', '2010'),
    ('45565', 'CS-101', '1', 'Spring', '2010'),
    ('45565', 'CS-319', '1', 'Spring', '2010'),
```

```
    ('76766', 'BIO-101', '1', 'Summer', '2009'),
    ('76766', 'BIO-301', '1', 'Summer', '2010'),
    ('83821', 'CS-190', '1', 'Spring', '2009'),
    ('83821', 'CS-190', '2', 'Spring', '2009'),
    ('83821', 'CS-319', '2', 'Spring', '2010'),
    ('98345', 'EE-181', '1', 'Spring', '2009');

INSERT INTO
    student
VALUES
    ('00128', 'Zhang', 'Comp. Sci.', '102'),
    ('12345', 'Shankar', 'Comp. Sci.', '32'),
    ('19991', 'Brandt', 'History', '80'),
    ('23121', 'Chavez', 'Finance', '110'),
    ('44553', 'Peltier', 'Physics', '56'),
    ('45678', 'Levy', 'Physics', '46'),
    ('54321', 'Williams', 'Comp. Sci.', '54'),
    ('55739', 'Sanchez', 'Music', '38'),
    ('70557', 'Snow', 'Physics', '0'),
    ('76543', 'Brown', 'Comp. Sci.', '58'),
    ('76653', 'Aoi', 'Elec. Eng.', '60'),
    ('98765', 'Bourikas', 'Elec. Eng.', '98'),
    ('98988', 'Tanaka', 'Biology', '120');

INSERT INTO
    takes
VALUES
    ('00128', 'CS-101', '1', 'Fall', '2009', 'A'),
    ('00128', 'CS-347', '1', 'Fall', '2009', 'A-'),
    ('12345', 'CS-101', '1', 'Fall', '2009', 'C'),
    ('12345', 'CS-190', '2', 'Spring', '2009', 'A'),
    ('12345', 'CS-315', '1', 'Spring', '2010', 'A'),
    ('12345', 'CS-347', '1', 'Fall', '2009', 'A'),
    ('19991', 'HIS-351', '1', 'Spring', '2010', 'B'),
    ( '23121', 'FIN-201', '1', 'Spring', '2010', 'C+'),
    ('44553', 'PHY-101', '1', 'Fall', '2009', 'B-'),
```

```
    ('45678', 'CS-101', '1', 'Fall', '2009', 'F'),
    ('45678', 'CS-101', '1', 'Spring', '2010', 'B+'),
    ('45678', 'CS-319', '1', 'Spring', '2010', 'B'),
    ('54321', 'CS-101', '1', 'Fall', '2009', 'A-'),
    ('54321', 'CS-190', '2', 'Spring', '2009', 'B+'),
    ('55739', 'MU-199', '1', 'Spring', '2010', 'A-'),
    ('76543', 'CS-101', '1', 'Fall', '2009', 'A'),
    ('76543', 'CS-319', '2', 'Spring', '2010', 'A'),
    ('76653', 'EE-181', '1', 'Spring', '2009', 'C'),
    ('98765', 'CS-101', '1', 'Fall', '2009', 'C-'),
    ('98765', 'CS-315', '1', 'Spring', '2010', 'B'),
    ('98988', 'BIO-101', '1', 'Summer', '2009', 'A'),
    ( '98988', 'BIO-301', '1', 'Summer', '2010', NULL);

INSERT INTO
    advisor
VALUES
    ('00128', '45565'),
    ('12345', '10101'),
    ('23121', '76543'),
    ('44553', '22222'),
    ('45678', '22222'),
    ('76543', '45565'),
    ('76653', '98345'),
    ('98765', '98345'),
    ('98988', '76766');

INSERT INTO
    time_slot
VALUES
    ('A', 'M', '8', '0', '8', '50'),
    ('A', 'W', '8', '0', '8', '50'),
    ('A', 'F', '8', '0', '8', '50'),
    ('B', 'M', '9', '0', '9', '50'),
    ('B', 'W', '9', '0', '9', '50'),
    ('B', 'F', '9', '0', '9', '50'),
```

```
    ('C', 'M', '11', '0', '11', '50'),
    ('C', 'W', '11', '0', '11', '50'),
    ('C', 'F', '11', '0', '11', '50'),
    ('D', 'M', '13', '0', '13', '50'),
    ('D', 'W', '13', '0', '13', '50'),
    ('D', 'F', '13', '0', '13', '50'),
    ('E', 'T', '10', '30', '11', '45 '),
    ('E', 'R', '10', '30', '11', '45 '),
    ('F', 'T', '14', '30', '15', '45 '),
    ('F', 'R', '14', '30', '15', '45 '),
    ('G', 'M', '16', '0', '16', '50'),
    ('G', 'W', '16', '0', '16', '50'),
    ('G', 'F', '16', '0', '16', '50'),
    ('H', 'W', '10', '0', '12', '30');

INSERT INTO
    prereq
VALUES
    ('BIO-301', 'BIO-101'),
    ('BIO-399', 'BIO-101'),
    ('CS-190', 'CS-101'),
    ('CS-315', 'CS-101'),
    ('CS-319', 'CS-101'),
    ('CS-347', 'CS-101'),
    ('EE-181', 'PHY-101');```

### Table Values

``` sql

SELECT * FROM advisor;
+-------+-------+
| s_id  | i_id  |
+-------+-------+
| 12345 | 10101 |
| 44553 | 22222 |
```
```

```
| 45678 | 22222 |
| 00128 | 45565 |
| 76543 | 45565 |
| 23121 | 76543 |
| 98988 | 76766 |
| 76653 | 98345 |
| 98765 | 98345 |
+-------+-------+
-- 9 rows in set (0.00 sec)

SELECT * FROM classroom;
+----------+-------------+----------+
| building | room_number | capacity |
+----------+-------------+----------+
| Packard  | 101         |      500 |
| Painter  | 514         |       10 |
| Taylor   | 3128        |       70 |
| Watson   | 100         |       30 |
| Watson   | 120         |       50 |
+----------+-------------+----------+
-- 5 rows in set (0.00 sec)

SELECT * FROM course;
+-----------+----------------------------+------------+---------+
| 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 |
+-----------+----------------------------+------------+---------+
-- 13 rows in set (0.00 sec)

SELECT * FROM department;
+------------+----------+-----------+
| dept_name  | building | budget    |
+------------+----------+-----------+
| Biology    | Watson   |  90000.00 |
| Comp. Sci. | Taylor   | 100000.00 |
| Elec. Eng. | Taylor   |  85000.00 |
| Finance    | Painter  | 120000.00 |
| History    | Painter  |  50000.00 |
| Music      | Packard  |  80000.00 |
| Physics    | Watson   |  70000.00 |
+------------+----------+-----------+
-- 7 rows in set (0.00 sec)

SELECT * FROM instructor;
+-------+------------+------------+----------+
| id    | name       | dept_name  | salary   |
+-------+------------+------------+----------+
| 10101 | Srinivasan | Comp. Sci. | 65000.00 |
| 12121 | Wu         | Finance    | 90000.00 |
| 15151 | Mozart     | Music      | 40000.00 |
| 22222 | Einstein   | Physics    | 95000.00 |
| 32343 | El Said    | History    | 60000.00 |
| 33456 | Gold       | Physics    | 87000.00 |
| 45565 | Katz       | Comp. Sci. | 75000.00 |
| 58583 | Califieri  | History    | 62000.00 |
| 76543 | Singh      | Finance    | 80000.00 |
| 76766 | Crick      | Biology    | 72000.00 |
| 83821 | Brandt     | Comp. Sci. | 92000.00 |
| 98345 | Kim        | Elec. Eng. | 80000.00 |
```

```
+-------+------------+------------+----------+
-- 12 rows in set (0.00 sec)

SELECT * FROM prereq;
+-----------+-----------+
| course_id | prereq_id |
+-----------+-----------+
| BIO-301   | BIO-101   |
| BIO-399   | BIO-101   |
| CS-190    | CS-101    |
| CS-315    | CS-101    |
| CS-319    | CS-101    |
| CS-347    | CS-101    |
| EE-181    | PHY-101   |
+-----------+-----------+
-- 7 rows in set (0.00 sec)

SELECT * FROM section;
+-----------+--------+----------+------+----------+-------------+--------------+
| course_id | sec_id | semester | year | building | room_number | time_slot_id |
+-----------+--------+----------+------+----------+-------------+--------------+
| BIO-101   | 1      | Summer   | 2009 | Painter  | 514         | B            |
| BIO-301   | 1      | Summer   | 2010 | Painter  | 514         | A            |
| CS-101    | 1      | Fall     | 2009 | Packard  | 101         | H            |
| CS-101    | 1      | Spring   | 2010 | Packard  | 101         | F            |
| CS-190    | 1      | Spring   | 2009 | Taylor   | 3128        | E            |
| CS-190    | 2      | Spring   | 2009 | Taylor   | 3128        | A            |
| CS-315    | 1      | Spring   | 2010 | Watson   | 120         | D            |
| CS-319    | 1      | Spring   | 2010 | Watson   | 100         | B            |
| CS-319    | 2      | Spring   | 2010 | Taylor   | 3128        | C            |
| CS-347    | 1      | Fall     | 2009 | Taylor   | 3128        | A            |
| EE-181    | 1      | Spring   | 2009 | Taylor   | 3128        | C            |
| FIN-201   | 1      | Spring   | 2010 | Packard  | 101         | B            |
| HIS-351   | 1      | Spring   | 2010 | Painter  | 514         | C            |
| MU-199    | 1      | Spring   | 2010 | Packard  | 101         | D            |
| PHY-101   | 1      | Fall     | 2009 | Watson   | 100         | A            |
```

```
+-----------+--------+----------+------+----------+-------------+--------------+
-- 15 rows in set (0.00 sec)

SELECT * FROM student;
+-------+----------+------------+----------+
| id    | name     | dept_name  | tot_cred |
+-------+----------+------------+----------+
| 00128 | Zhang    | Comp. Sci. |      102 |
| 12345 | Shankar  | Comp. Sci. |       32 |
| 19991 | Brandt   | History    |       80 |
| 23121 | Chavez   | Finance    |      110 |
| 44553 | Peltier  | Physics    |       56 |
| 45678 | Levy     | Physics    |       46 |
| 54321 | Williams | Comp. Sci. |       54 |
| 55739 | Sanchez  | Music      |       38 |
| 70557 | Snow     | Physics    |        0 |
| 76543 | Brown    | Comp. Sci. |       58 |
| 76653 | Aoi      | Elec. Eng. |       60 |
| 98765 | Bourikas | Elec. Eng. |       98 |
| 98988 | Tanaka   | Biology    |      120 |
+-------+----------+------------+----------+
-- 13 rows in set (0.00 sec)

SELECT * FROM takes;
+-------+-----------+--------+----------+------+-------+
| id    | course_id | sec_id | semester | year | grade |
+-------+-----------+--------+----------+------+-------+
| 00128 | CS-101    | 1      | Fall     | 2009 | A     |
| 00128 | CS-347    | 1      | Fall     | 2009 | A-    |
| 12345 | CS-101    | 1      | Fall     | 2009 | C     |
| 12345 | CS-190    | 2      | Spring   | 2009 | A     |
| 12345 | CS-315    | 1      | Spring   | 2010 | A     |
| 12345 | CS-347    | 1      | Fall     | 2009 | A     |
| 19991 | HIS-351   | 1      | Spring   | 2010 | B     |
| 23121 | FIN-201   | 1      | Spring   | 2010 | C+    |
| 44553 | PHY-101   | 1      | Fall     | 2009 | B-    |
```

```
| 45678 | CS-101    | 1      | Fall     | 2009 | F     |
| 45678 | CS-101    | 1      | Spring   | 2010 | B+    |
| 45678 | CS-319    | 1      | Spring   | 2010 | B     |
| 54321 | CS-101    | 1      | Fall     | 2009 | A-    |
| 54321 | CS-190    | 2      | Spring   | 2009 | B+    |
| 55739 | MU-199    | 1      | Spring   | 2010 | A-    |
| 76543 | CS-101    | 1      | Fall     | 2009 | A     |
| 76543 | CS-319    | 2      | Spring   | 2010 | A     |
| 76653 | EE-181    | 1      | Spring   | 2009 | C     |
| 98765 | CS-101    | 1      | Fall     | 2009 | C-    |
| 98765 | CS-315    | 1      | Spring   | 2010 | B     |
| 98988 | BIO-101   | 1      | Summer   | 2009 | A     |
| 98988 | BIO-301   | 1      | Summer   | 2010 | NULL  |
+-------+-----------+--------+----------+------+-------+
-- 22 rows in set (0.00 sec)

SELECT * FROM teaches;
+-------+-----------+--------+----------+------+
| id    | course_id | sec_id | semester | year |
+-------+-----------+--------+----------+------+
| 76766 | BIO-101   | 1      | Summer   | 2009 |
| 76766 | BIO-301   | 1      | Summer   | 2010 |
| 10101 | CS-101    | 1      | Fall     | 2009 |
| 45565 | CS-101    | 1      | Spring   | 2010 |
| 83821 | CS-190    | 1      | Spring   | 2009 |
| 83821 | CS-190    | 2      | Spring   | 2009 |
| 10101 | CS-315    | 1      | Spring   | 2010 |
| 45565 | CS-319    | 1      | Spring   | 2010 |
| 83821 | CS-319    | 2      | Spring   | 2010 |
| 10101 | CS-347    | 1      | Fall     | 2009 |
| 98345 | EE-181    | 1      | Spring   | 2009 |
| 12121 | FIN-201   | 1      | Spring   | 2010 |
| 32343 | HIS-351   | 1      | Spring   | 2010 |
| 15151 | MU-199    | 1      | Spring   | 2010 |
| 22222 | PHY-101   | 1      | Fall     | 2009 |
+-------+-----------+--------+----------+------+
```

```
-- 15 rows in set (0.00 sec)

SELECT * FROM time_slot;
+--------------+-----+----------+-----------+--------+---------+
| time_slot_id | DAY | start_hr | start_min | end_hr | end_min |
+--------------+-----+----------+-----------+--------+---------+
| A            | F   |        8 |         0 |      8 |      50 |
| A            | M   |        8 |         0 |      8 |      50 |
| A            | W   |        8 |         0 |      8 |      50 |
| B            | F   |        9 |         0 |      9 |      50 |
| B            | M   |        9 |         0 |      9 |      50 |
| B            | W   |        9 |         0 |      9 |      50 |
| C            | F   |       11 |         0 |     11 |      50 |
| C            | M   |       11 |         0 |     11 |      50 |
| C            | W   |       11 |         0 |     11 |      50 |
| D            | F   |       13 |         0 |     13 |      50 |
| D            | M   |       13 |         0 |     13 |      50 |
| D            | W   |       13 |         0 |     13 |      50 |
| E            | R   |       10 |        30 |     11 |      45 |
| E            | T   |       10 |        30 |     11 |      45 |
| F            | R   |       14 |        30 |     15 |      45 |
| F            | T   |       14 |        30 |     15 |      45 |
| G            | F   |       16 |         0 |     16 |      50 |
| G            | M   |       16 |         0 |     16 |      50 |
| G            | W   |       16 |         0 |     16 |      50 |
| H            | W   |       10 |         0 |     12 |      30 |
+--------------+-----+----------+-----------+--------+---------+
```