به‌نام آفریننده بیت‌ها


دانشکده‌ی برق و کامپیوتر
دانشگاه صنعتی اصفهان
نیم‌سال اول ۱۴۰۱ - ۱۴۰۲

# پایگاه داده ۱

## تمرین دوم

---

نام:
دانیال خراسانی‌زاده
شماره دانشجویی:
۹۹۲۲۳۹۳
استاد درس:
دکتر بصیری

# فهرست مطالب

# ۱

| داده | نوع متغیر |
|---|---|
| سه حرف مخفف شده ماه های میلادی | CHAR(3) |
| قیمت دلاری محصولات که همکی دو رقم اعشار دارند | DECIMAL(12,2) |
| نام و نام خانوادگی کاربر | VARCHAR(MAX) |
| کد ملی کاربر | CHAR(10) |
| ذخیره قیمت لحظه‌ای ارزهای دیجیتال | DOUBLE PRECISION |
| تعداد بازدید یک ویدیو | INT |

# ۲

داده VARCHAR(n) برای رشته‌هایی که طول متغیر دارند به کار می‌رود و رشته‌ای با حداکثر طول n را با همان طول اصلی خود رشته ذخیره می‌کند از این جهت در زمان‌هایی که رشته‌ها طول متغیر دارند این نوع داده حجم کمتری اشغال می‌کند. داده CHAR(n) رشته‌ای به طول حداکثر n را بدون توجه به طول اصلی رشته با طول n ذخیره می‌کند. (اگر طول رشته کمتر باشد طول آن را افزایش می‌دهد تا به n برسد.) این نوع داده فضای بیشتری اشغال می‌کند ولی سرعت کارکرد بیشتری دارد و از این جهت اگر طول رشته‌های ما همیشه ثابت باشد انتخاب بهتری است.

# ۳

## ۱.۳

```
1  WHERE SUBSTR(first_name, 1, 2) = 'me' AND SUBSTR(lastname, LENGTH(lastname) - 2, 3) = 'avi'
```

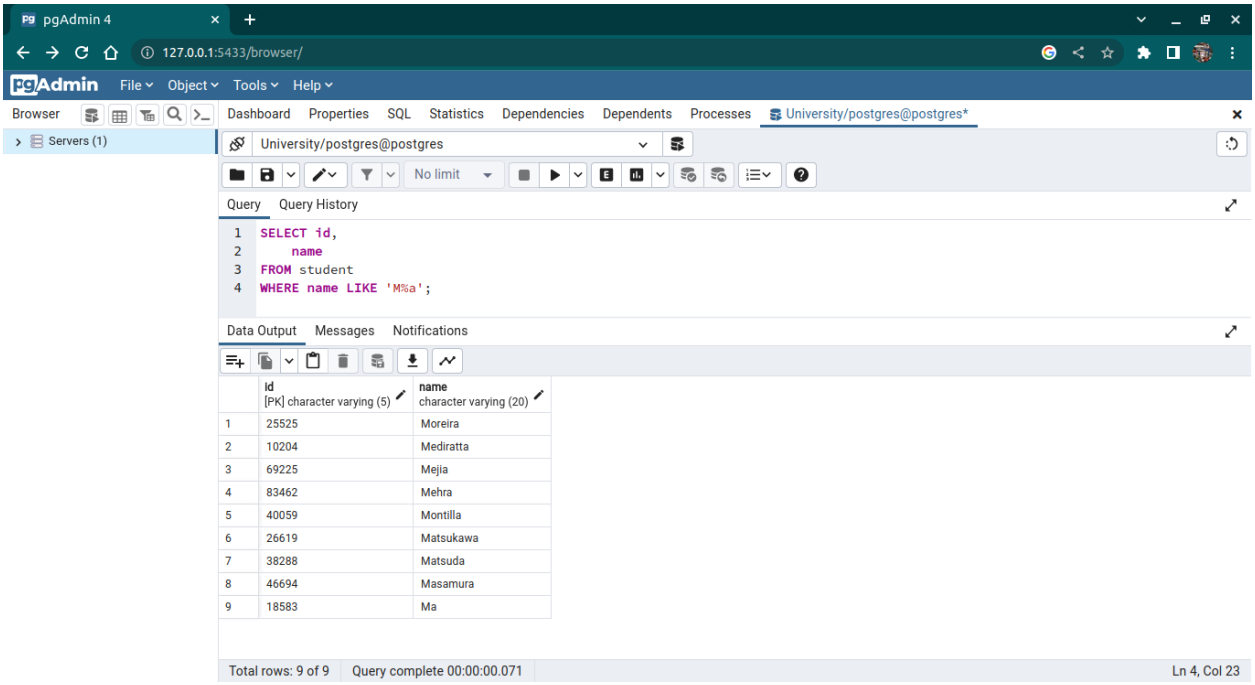
## ۲.۳

```
1  SELECT CONCAT(first_name, last_name) from STUDENT
```

# ۴

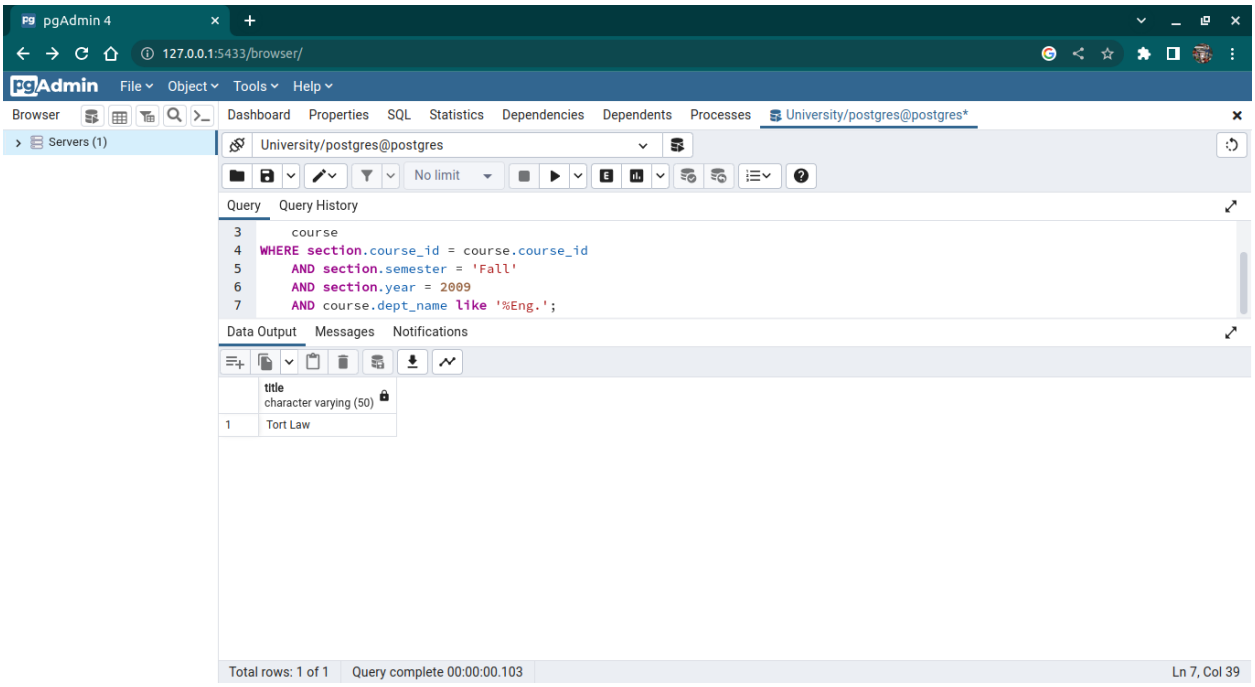
## ۱.۴

```sql
SELECT id,
    name
FROM student
WHERE name LIKE 'M%a';
```



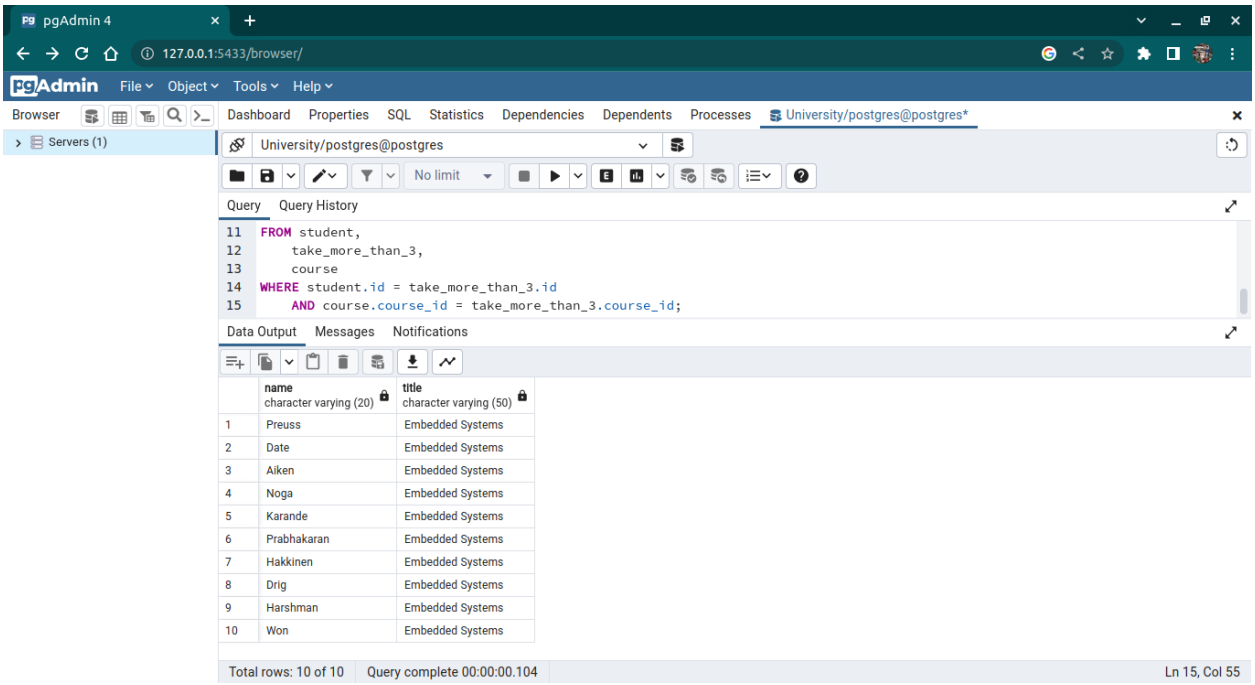
## ۲.۴

```sql
SELECT course.title
FROM section,
    course
WHERE section.course_id = course.course_id
    AND section.semester = 'Fall'
    AND section.year = 2009
    AND course.dept_name like '%Eng.';
```



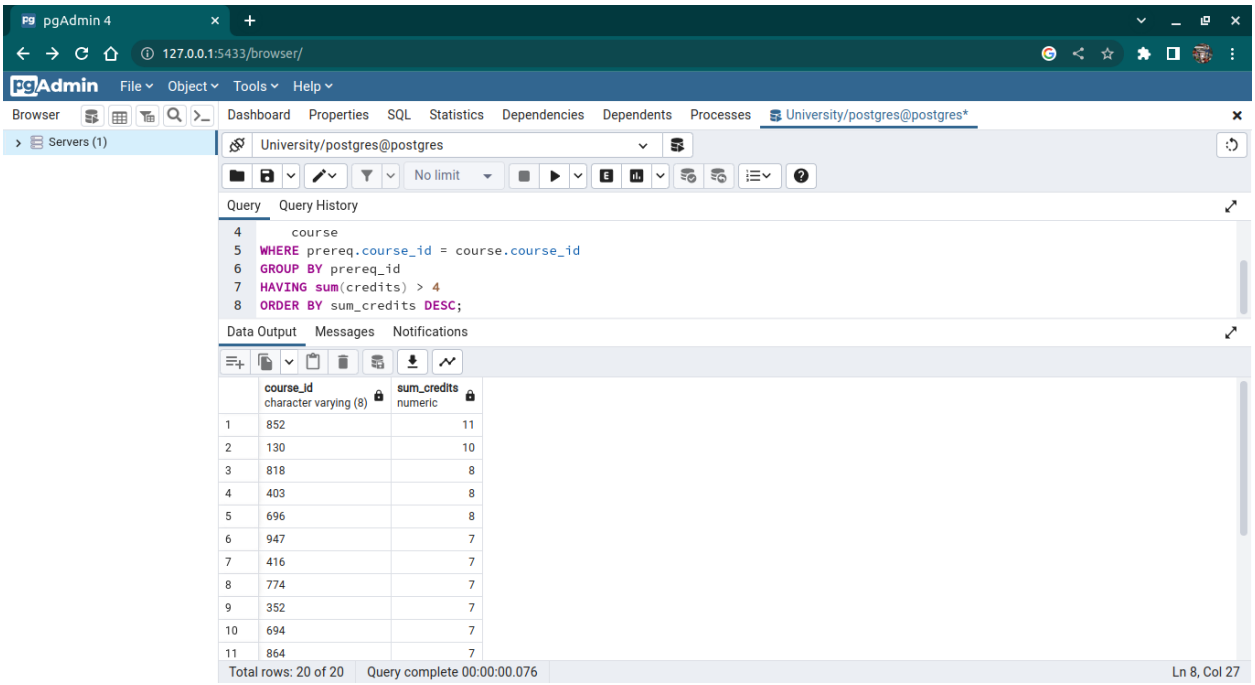
## ۳.۴

```sql
WITH take_more_than_3 AS (
    SELECT id,
        course_id
    FROM takes
    GROUP BY id,
        course_id
    HAVING count(*) >= 3
)
SELECT student.name,
    course.title
FROM student,
    take_more_than_3,
    course
WHERE student.id = take_more_than_3.id
    AND course.course_id = take_more_than_3.course_id;
```



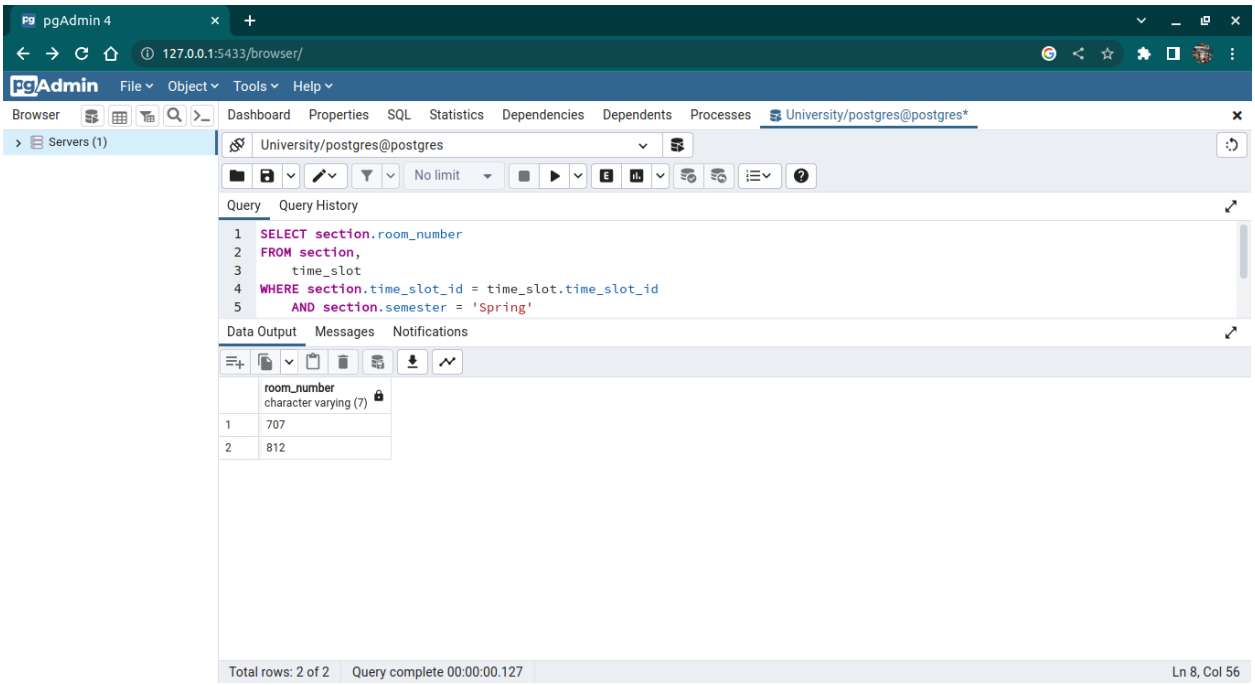
## ۴.۴

```sql
SELECT prereq_id AS course_id,
    sum(credits) AS sum_credits
FROM prereq,
    course
WHERE prereq.course_id = course.course_id
GROUP BY prereq_id
HAVING sum(credits) > 4
ORDER BY sum_credits DESC;
```



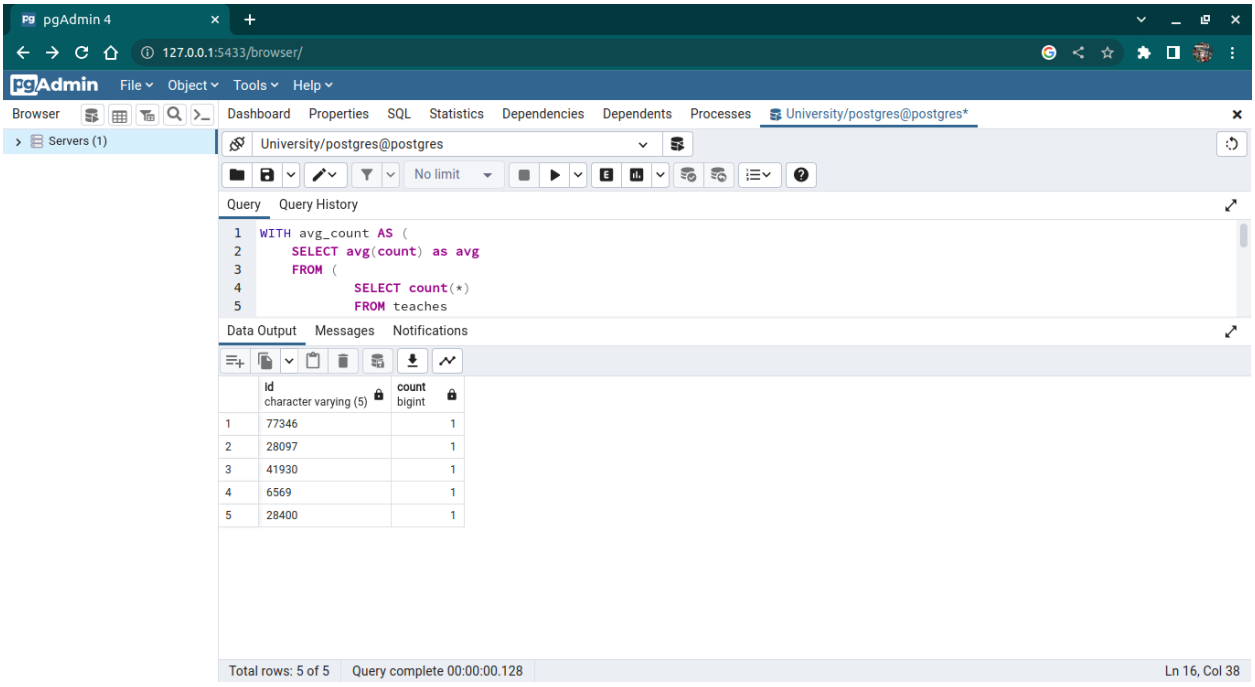
## ۵.۴

```sql
SELECT section.room_number
FROM section,
    time_slot
WHERE section.time_slot_id = time_slot.time_slot_id
    AND section.semester = 'Spring'
    AND section.year = 2008
GROUP BY section.room_number
HAVING sum(time_slot.end_hr - time_slot.start_hr) >= 2;
```



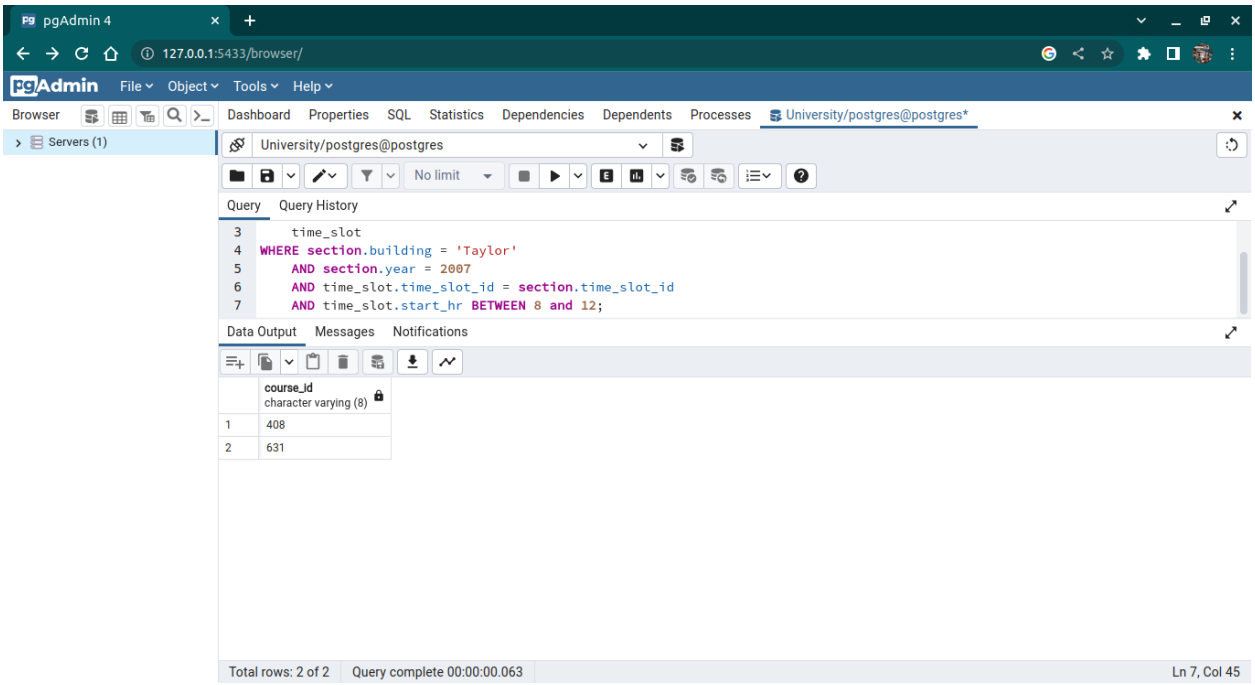
## ۶.۴

```sql
WITH avg_count AS (
    SELECT avg(count) AS avg
    FROM (
            SELECT count(*)
            FROM teaches
            WHERE year = 2003
            GROUP BY id
        ) AS course_count
)
SELECT teaches.id,
    count(*)
FROM teaches,
    avg_count
WHERE teaches.year = 2003
GROUP BY teaches.id
HAVING count(*) < avg(avg_count.avg);
```



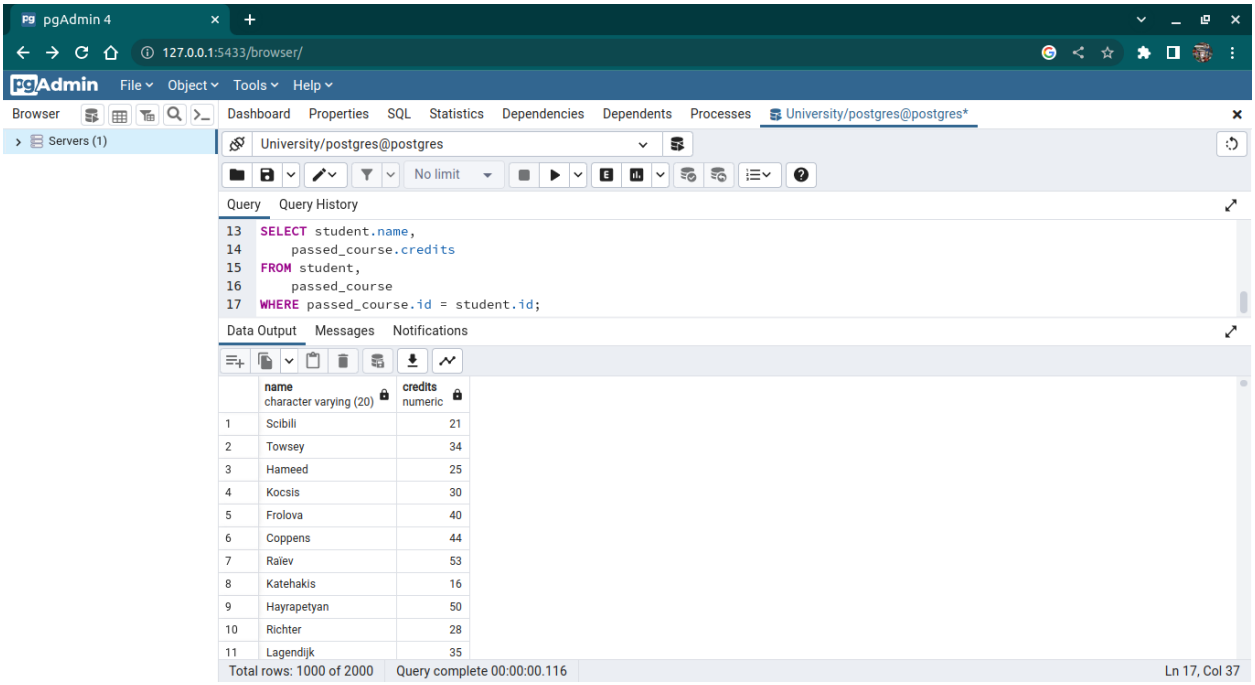
## ۷.۴

```sql
SELECT distinct course_id
FROM section,
    time_slot
WHERE section.building = 'Taylor'
    AND section.year = 2007
    AND time_slot.time_slot_id = section.time_slot_id
    AND time_slot.start_hr BETWEEN 8 and 12;
```



## ۸.۴

```sql
WITH passed_course AS (
    SELECT takes.id,
        sum(course.credits) as credits
    FROM takes,
        course
    WHERE takes.course_id = course.course_id
        AND (
            grade like 'A%'
            OR grade like 'B%'
        )
    GROUP BY takes.id
)
SELECT student.name,
    passed_course.credits
FROM student,
    passed_course
WHERE passed_course.id = student.id;
```

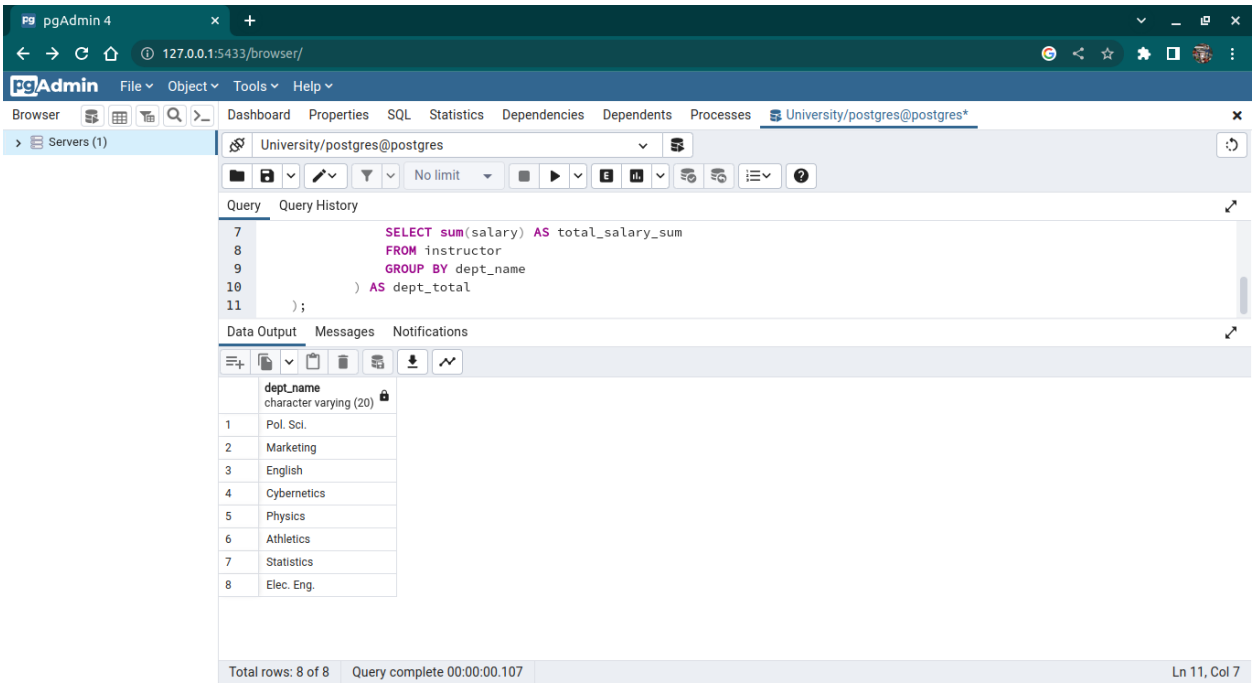


# ۵

## ۱.۵

لیست دپارتمان‌هایی که مجموع حقوق اساتیدشان از میانگین مجموع حقوق اساتید همه دپارتمان‌ها بیشتر است.

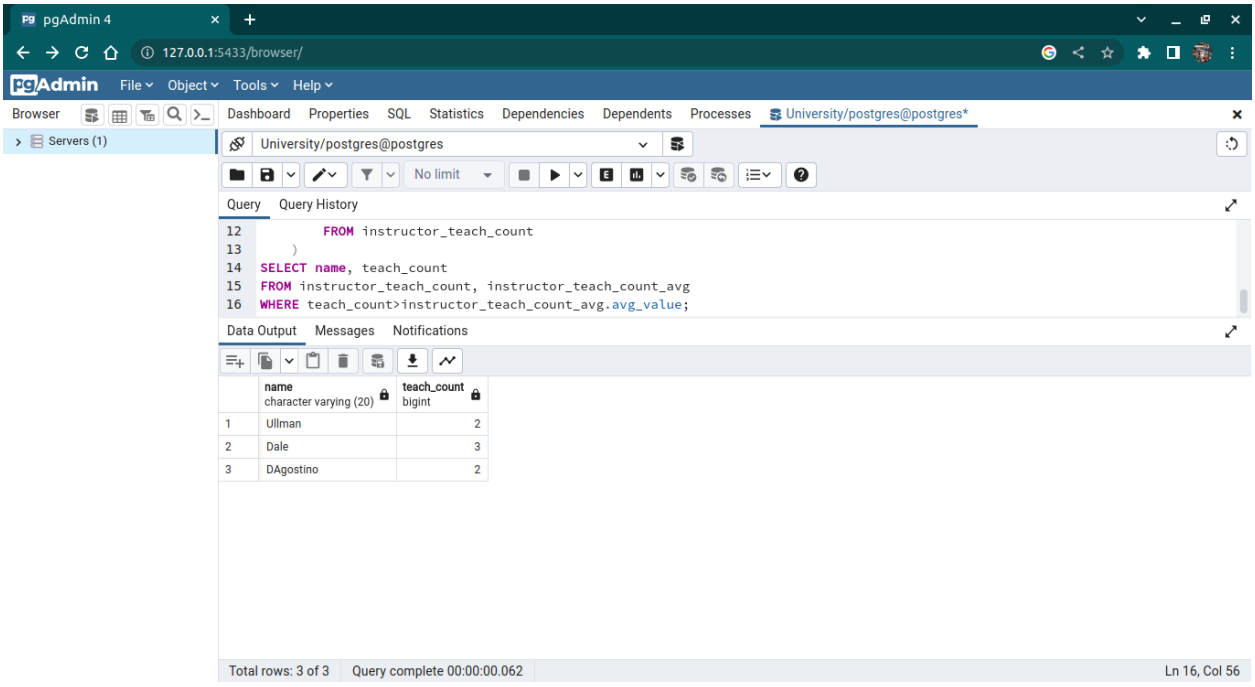
```sql
SELECT dept_name
FROM instructor
GROUP BY dept_name
HAVING sum(salary) >(
       SELECT avg(total_salary_sum)
       FROM (
              SELECT sum(salary) AS total_salary_sum
              FROM instructor
              GROUP BY dept_name
          ) AS dept_total
    );
```



## ۲.۵

لیست اساتید و تعداد دروس ارائه شده آن‌ها در سال ۲۰۰۳ که در این سال بیشتر از میانگین درس ارائه داده‌اند.

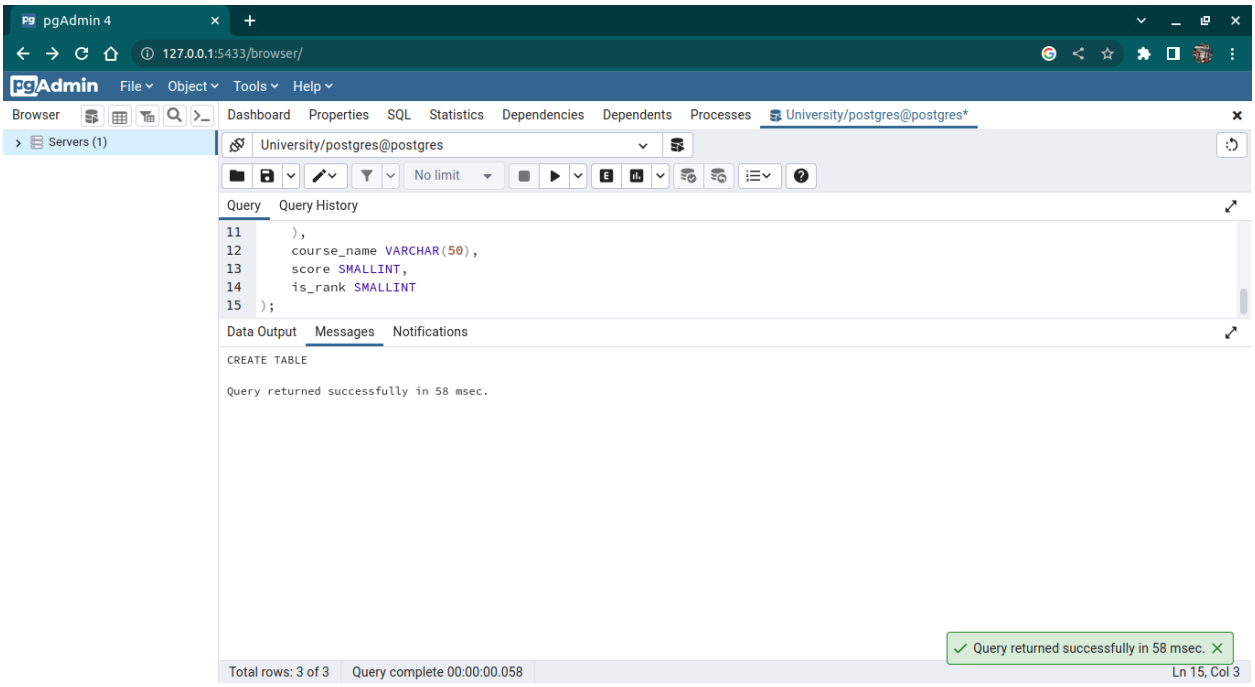
```sql
WITH
instructor_teach_count(name, teach_count) AS (
        SELECT instructor.name, COUNT(*)
        FROM teaches,
            instructor
        WHERE teaches.id = instructor.id
            AND year = 2003
        GROUP BY instructor.id
    ),
    instructor_teach_count_avg(avg_value) AS (
        SELECT avg(teach_count)
        FROM instructor_teach_count
    )
SELECT name, teach_count
FROM instructor_teach_count, instructor_teach_count_avg
WHERE teach_count>instructor_teach_count_avg.avg_value;
```



# ۶

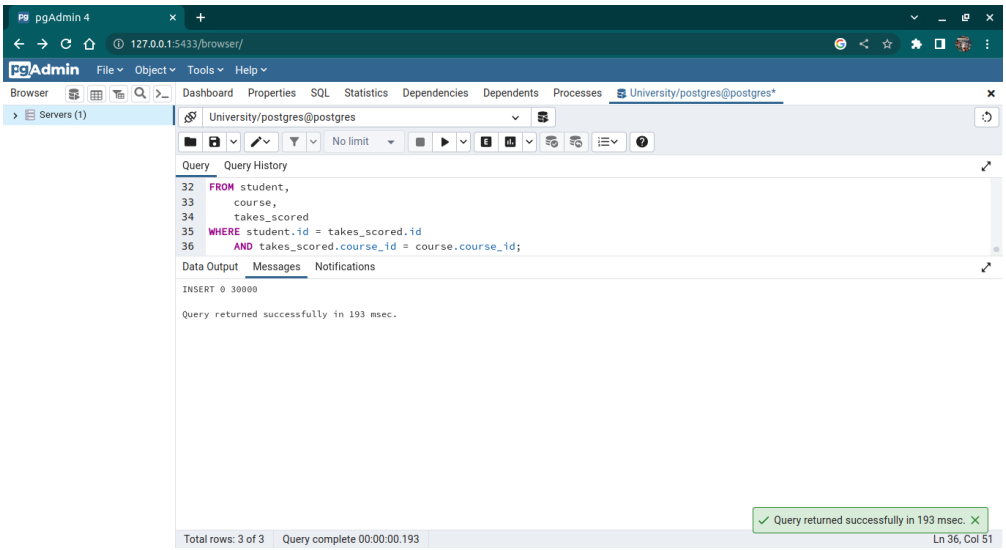
## ۱.۶

```sql
CREATE TABLE uni_data (
    stu_id VARCHAR(5),
    stu_name VARCHAR(20) NOT NULL,
    stu_dept_name VARCHAR(20),
    year NUMERIC(4, 0) CHECK (
        year > 1701
        AND year < 2100
    ),
    semester VARCHAR(6) CHECK (
        semester IN ('Fall', 'Winter', 'Spring', 'Summer')
    ),
    course_name VARCHAR(50),
    score SMALLINT,
    is_rank SMALLINT
);
```



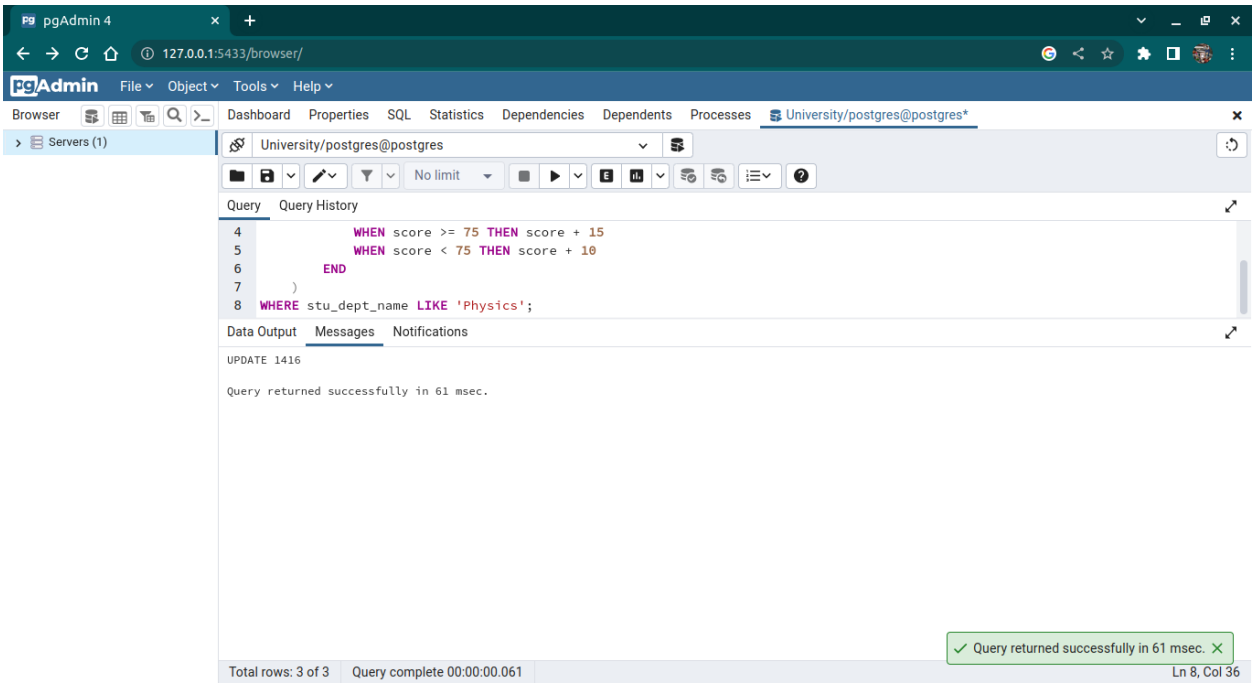
## ۲.۶

```sql
INSERT INTO uni_data WITH takes_scored AS (
       SELECT *,
           (
               CASE
                   WHEN takes.grade LIKE 'A+' THEN 100
                   WHEN takes.grade LIKE 'A' THEN 95
                   WHEN takes.grade LIKE 'A-' THEN 90
                   WHEN takes.grade LIKE 'B+' THEN 85
                   WHEN takes.grade LIKE 'B' THEN 80
                   WHEN takes.grade LIKE 'B-' THEN 75
                   WHEN takes.grade LIKE 'C+' THEN 70
                   WHEN takes.grade LIKE 'C' THEN 65
                   WHEN takes.grade LIKE 'C-' THEN 60
                   ELSE 0
               END
           ) AS score
       FROM takes
   )
SELECT student.id AS stu_id,
   student.name AS stu_name,
   student.dept_name AS stu_dept_name,
   takes_scored.year AS year,
   takes_scored.semester AS semester,
   course.title AS course_name,
   takes_scored.score AS score,
   (
       CASE
           WHEN takes_scored.score > 70 THEN 1
           ELSE 0
       END
   ) AS is_rank
FROM student,
   course,
   takes_scored
WHERE student.id = takes_scored.id
   AND takes_scored.course_id = course.course_id;
```



## ۳.۶

```sql
UPDATE uni_data
SET score = (
       CASE
           WHEN score >= 75 THEN score + 15
           WHEN score < 75 THEN score + 10
       END
   )
WHERE stu_dept_name LIKE 'Physics';
```



## ۴.۶

```sql
DELETE FROM uni_data AS u
WHERE u.stu_name LIKE 'T%'
    AND u.score > (
        SELECT avg(score)
        FROM uni_data AS t
        GROUP BY stu_dept_name
        HAVING t.stu_dept_name = u.stu_dept_name
    );
```