

به نام خدا

تکلیف دوم پایگاه داده

حدیث غفوری ۹۸۲۵۴۱۳

سوال ۱

داده	نوع متغیر
سه حرف مخفف شده ماه های میلادی	Char(3)
قیمت دلاری محصولات که همگی دو رقم اعشار دارند.	Numeric()
نام و نام خانوادگی کاربر	Varchar(60)
کد ملی کاربر	Char(10)
ذخیره قیمت لحظه ای ارز های دیجیتال دلار	Float()
تعداد بازدید یک ویدئو	Int()

سوال ۲

Create table data(

nationalCode char(10),

fullName varchar(60),

acronymsAD char(3),

viewsNumber int(),

dollarPrice float(2),

digitalCurrencies numeric(),

primary key (nationalCode),

foreign key (nationalCode) references users(social_number)

);

سوال ۳

a. بازنویسی بدون استفاده از دستور unique

WHERE (SELECT distinct name from student)

b. بازنویسی بدون استفاده از دستور like

WHERE first_name ~~ 'me%' AND last_name ~~ '%avi'

c. بازنویسی بدون اپراتور ||

SELECT concat(first_name,last_name);

سوال ۴

a.

select title,length from film,category,film_category
where film.title like 'C%' and film.film_id = film_category.film_id and
(film_category.category_id,category.name)=(category.category_id,'Action')
order by length desc

The screenshot shows the pgAdmin 4 interface. On the left, the 'film' table is selected under the 'Columns (13)' section. The 'Query Editor' tab is active, displaying the following SQL query:

```
1 select title,length from film,category,film_category
2 where film.title like 'C%' and film.film_id = film_category.film_id and
3 (film_category.category_id,category.name) = (category.category_id,'Action')
4 order by length desc
```

The 'Data Output' tab shows the results of the query in a table with two columns: 'title' (character varying (255)) and 'length' (smallint). The results are ordered by length in descending order.

title	length
1 Casualties Encino	179
2 Campus Remember	167
3 Celebrity Horn	110
4 Crow Grease	104
5 Clueless Bucket	95
6 Caddyshack Jedi	52

b.

select title,name

from film,language

where film.language_id = language.language_id and

film.rental_rate > (select avg(rental_rate) from film)

order by rental_rate desc

limit 5

The screenshot shows the pgAdmin 4 web interface. On the left, the 'Browser' pane displays the database structure, with 'film' selected under the 'public' schema. The 'Query Editor' pane shows the following SQL query:

```
1 select title,name
2 from film,language
3 where film.language_id = language.language_id and
4 film.rental_rate > (select avg(rental_rate) from film)
5 order by rental_rate desc
6 limit 5
```

The 'Data Output' pane displays the results of the query in a table with two columns: 'title' and 'name'.

title	name
Grosse Wonderful	English
Airport Pollock	English
Bright Encounters	English
Ace Goldfinger	English
Chamber Italian	English

The bottom of the image shows a Windows taskbar with various application icons and a system tray displaying the date and time as 2025-11-14/17.

d.

```
select  
address,sum(payment.amount) as least_size  
from  
address , payment , store , staff  
where address.address_id= store.address_id and  
payment.staff_id = staff.staff_id and  
staff.store_id = store.store_id  
group by address.address_id  
order by least_size  
limit 1
```

The screenshot shows the pgAdmin 4 web interface. On the left is a 'Browser' pane showing a tree view of database objects: FTS Templates, Foreign Tables, Functions, Materialized Views, Procedures, Sequences, Tables (15), and film (with columns like film_id, title, description, etc.). The main area is the 'Query Editor' for 'dvdrental/postgres@PostgreSQL 13'. It contains the following SQL query:

```
1 select  
2 address,sum(payment.amount) as least_size  
3 from  
4 address , payment , store , staff  
5 where address.address_id= store.address_id and  
6 payment.staff_id = staff.staff_id and  
7 staff.store_id = store.store_id  
8 group by address.address_id  
9 order by least_size  
10 limit 1
```

Below the query editor is the 'Data Output' pane, which shows the results of the query in a table with two columns: 'address' (character varying (50)) and 'least_size' (numeric). The first row of data is:

address	least_size
47 MySakila Drive	30252.12

The bottom of the image shows a Windows taskbar with various application icons and system information like '12°C' and '2010'.

e.

```
select rating,title,count(distinct rating) as distinct_rating  
from film  
group by rating,title;
```

The screenshot shows the pgAdmin 4 web interface. On the left, the 'Browser' pane displays a tree view of the database schema, with the 'film' table selected under the 'public' schema. The 'Query Editor' pane in the center contains the following SQL query:

```
1 SELECT  
2   rating,  
3   title,  
4   count(distinct rating) as distinct_rating  
5 FROM  
6   film  
7 group by rating , title ;|
```

Below the query editor, the 'Data Output' pane displays the results of the query in a table format. The table has three columns: 'rating', 'title', and 'distinct_rating'. The results show 10 rows of data, each representing a film with its rating and title.

rating	title	distinct_rating
G	Ace Goldfinger	1
G	Affair Prejudice	1
G	African Egg	1
G	Alamo Videotape	1
G	Amistad Midsummer	1
G	Angels Life	1
G	Annie Identity	1
G	Armageddon Lost	1
G	Atlantis Cause	1
G	Autumn Crow	1

f.

```
select category.name,  
  
select film.title from film_category, film)  
  
where film_category.film_id = film.film_id  
  
and film_category.category_id=category.category_id  
  
order by film.length desc  
  
limit 1) as length_film_order  
  
,  
  
select film.title from film_category , film)  
  
where film.film_id = film_category.film_id  
  
and category.category_id = film_category.category_id  
  
order by film.rental_rate desc  
  
limit 1 ) as rate_film_order  
  
from category
```

The screenshot shows the pgAdmin 4 interface. On the left, the 'Columns (3)' table is expanded, showing columns: film_id, category_id, and last_update. The main window displays a SQL query in the 'Query Editor' tab. The query is as follows:

```
1 select category.name,  
2 (select film.title from film_category, film  
3 where film_category.film_id = film.film_id  
4 and film_category.category_id=category.category_id  
5 order by film.length desc  
6 limit 1) as length_film_order  
7 ,  
8 (select film.title from film_category , film  
9 where film.film_id = film_category.film_id  
10 and category.category_id = film_category.category_id  
11 order by film.rental_rate desc  
12 limit 1 ) as rate_film_order  
13 from category
```

Below the query editor, the 'Data Output' tab shows the results of the query. The results are displayed in a table with three columns: name, length_film_order, and rate_film_order. The table contains 10 rows of data.

name	length_film_order	rate_film_order
1 Action	Dam Forrester	American Circus
2 Animation	Gangs Pride	Bikini Borrowers
3 Children	Fury Murder	Backlash Undeafed
4 Classics	Conspiracy Spirit	Beast Hunchback
5 Comedy	Control Anthem	Airplane Sierra
6 Documentary	Wife Turn	Clerks Angels
7 Drama	Jacket Frisco	Bright Encounters
8 Family	King Evolution	Apache Divine
9 Foreign	Crystal Breaking	Baby Hall
10 Games	Chicago North	Autumn Crow

سوال ۵

a.

select id,name

from student

where name like 'M%' and name like '%a';

The screenshot shows the pgAdmin 4 web interface. On the left, the 'Tables (11)' folder is expanded, showing the 'student' table. The 'Query Editor' pane contains the following SQL query:

```
1 select id,name
2 from student
3 where name like 'M%' and name like '%a';
```

The 'Data Output' pane displays the results of the query in a table with two columns: 'id' and 'name'.

id	name
25525	Moreira
10204	Mediratta
69225	Mejia
83462	Mehra
40059	Montilla
26619	Matsukawa
38288	Matsuda
46694	Masamura
18583	Ma

b.

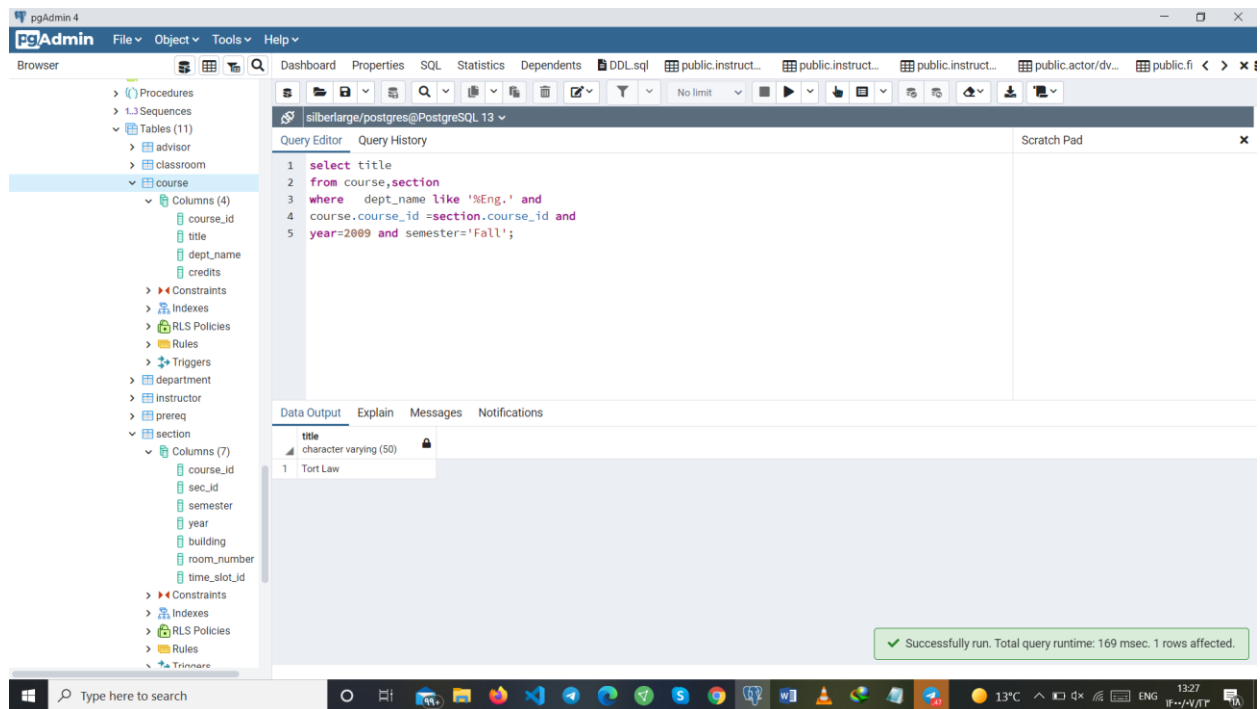
select title

from course,section

where dept_name like '%Eng.' and

course.course_id =section.course_id and

year=2009 and semester='Fall';

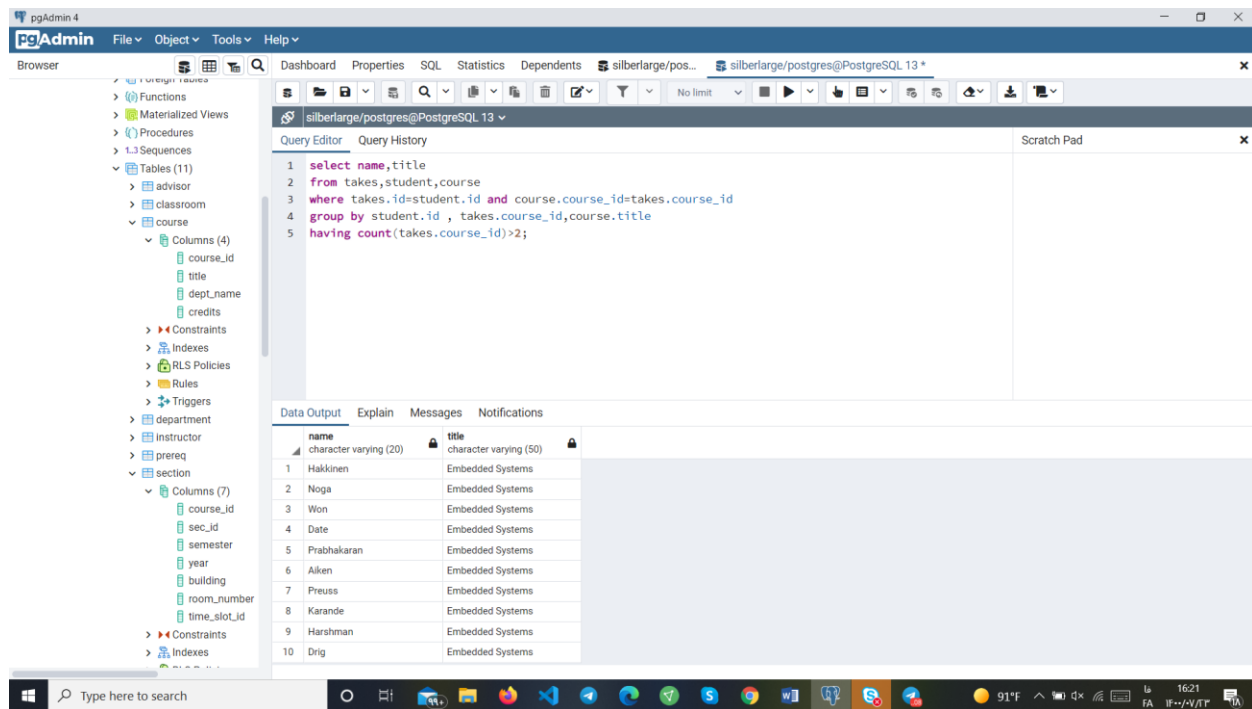


C.

```

select name,title
from takes,student,course
where takes.id=student.id and course.course_id=takes.course_id
group by student.id , takes.course_id,course.title
having count(takes.course_id)>2;

```

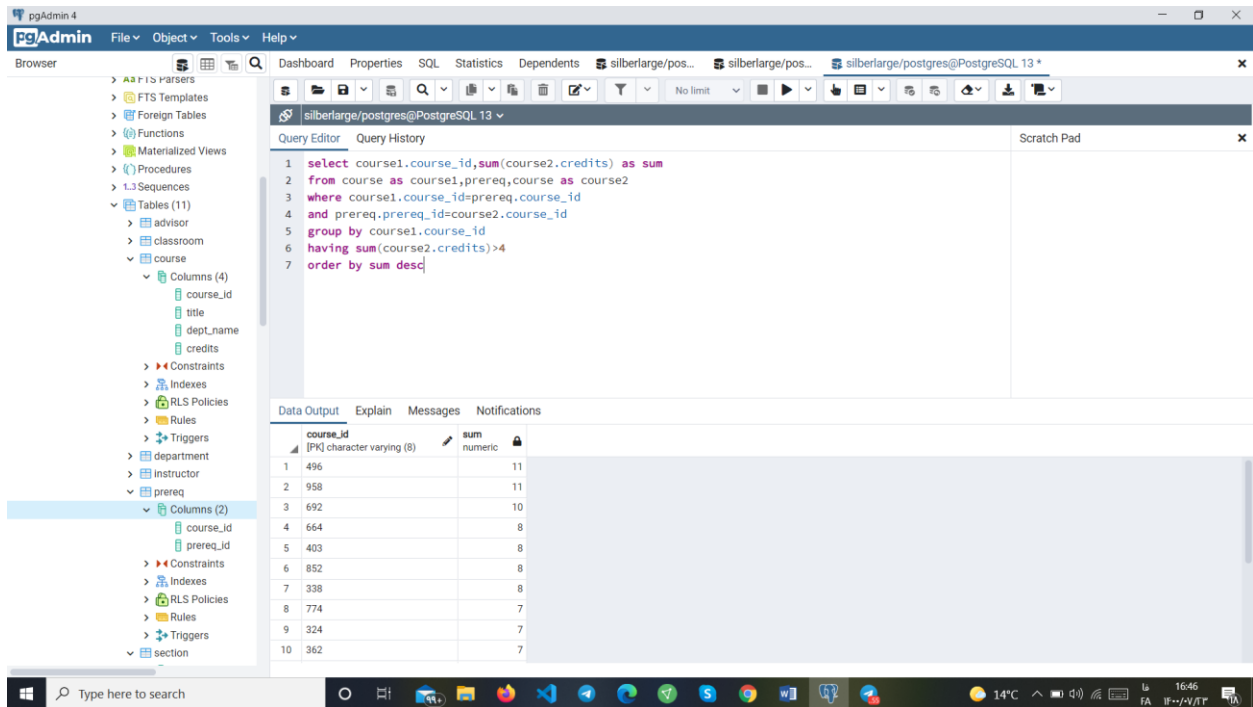



d.

```

select course1.course_id,sum(course2.credits) as sum
from course as course1,prereq,course as course2
where course1.course_id=prereq.course_id
and prereq.prereq_id=course2.course_id
group by course1.course_id
having sum(course2.credits)>4
order by sum desc

```

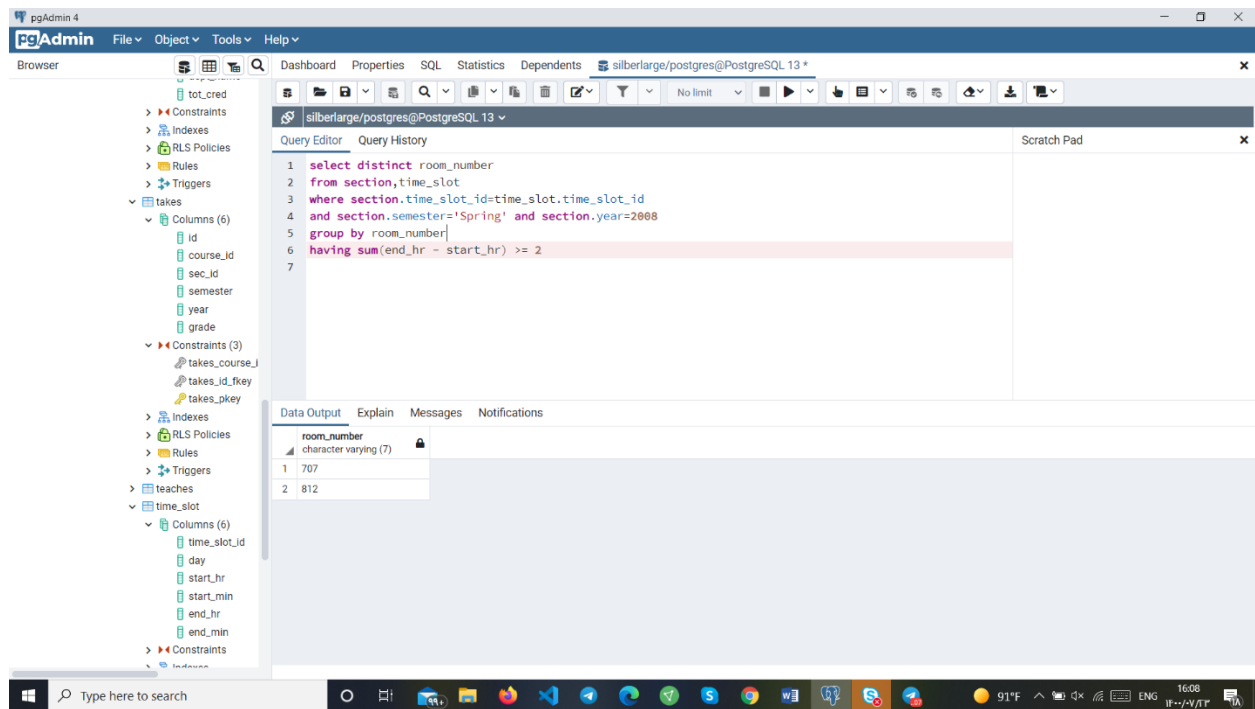


e.

```

select distinct 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 room_number
having sum(end_hr - start_hr) >= 2

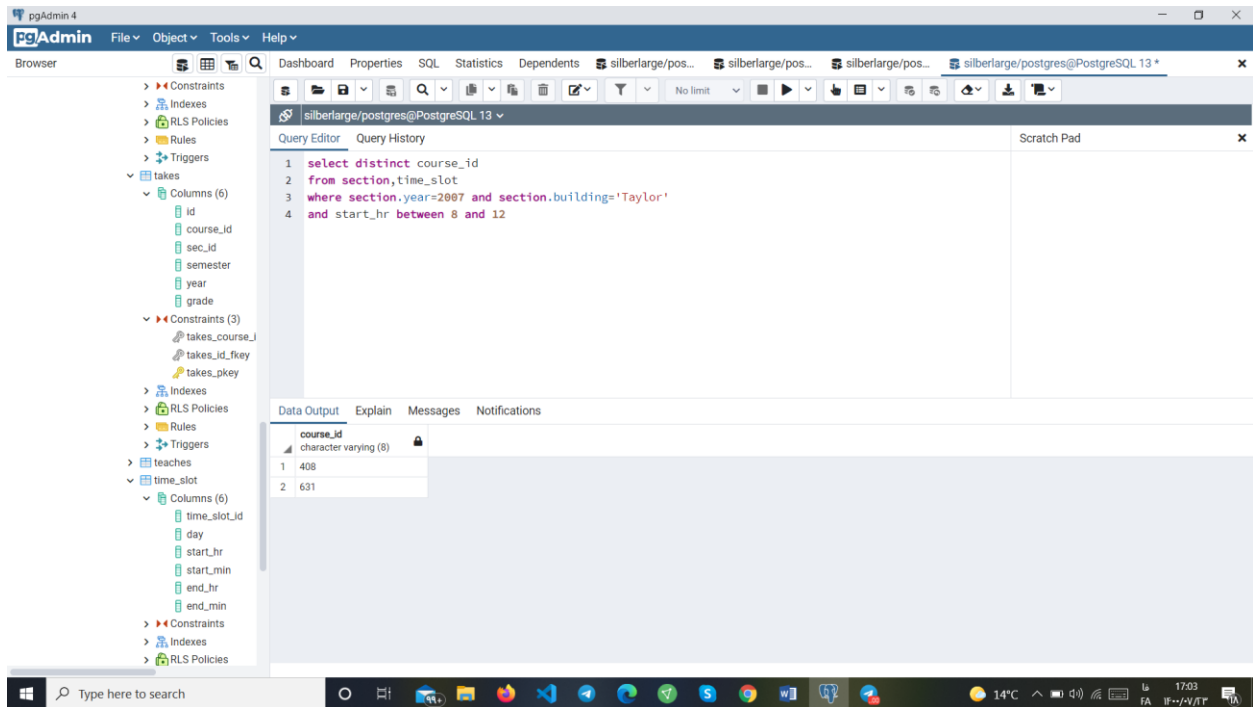
```



f.

g.

```
select distinct course_id
from section,time_slot
where section.year=2007 and section.building='Taylor'
and start_hr between 8 and 12
```



h.

```
select name , sum(credits) as new_calculation
from course,takes,student
where (takes.grade like 'A%' OR takes.grade like 'B%') and
student.id = takes.id and
takes.course_id = course.course_id
group by student.name
```

pgAdmin 4

File Object Tools Help

Browser

- room_number
- time_slot_id
- Constraints
- Indexes
- RLS Policies
- Rules
- Triggers
- student
 - Columns (4)
 - id
 - name
 - dept_name
 - tot_cred
 - Constraints
 - Indexes
 - RLS Policies
 - Rules
 - Triggers
- takes
 - Columns (6)
 - id
 - course_id
 - sec_id
 - semester
 - year
 - grade
 - Constraints (3)
 - takes_course_j
 - takes_id_fkey
 - takes_pkey
 - Indexes
 - RLS Policies
 - Rules

silberlarge/postgres@PostgreSQL 13

Query Editor Query History Scratch Pad

```
1 select name , sum(credits) as new_calculation
2 from course,takes,student
3 where (takes.grade like 'A%' OR takes.grade like 'B%') and
4 student.id = takes.id and
5 takes.course_id = course.course_id
6 group by student.name
7
```

Data Output Explain Messages Notifications

	name	new_calculation	
	character varying (20)	numeric	
1	Duong	44	
2	Shiang	23	
3	Makarychev	67	
4	Keiss	44	
5	Strader	35	
6	Szendrei	26	
7	Chandra	43	
8	Larsson	25	
9	Wagner	11	
10	Stephenn	40	

Type here to search

12°C 20:19 ENG