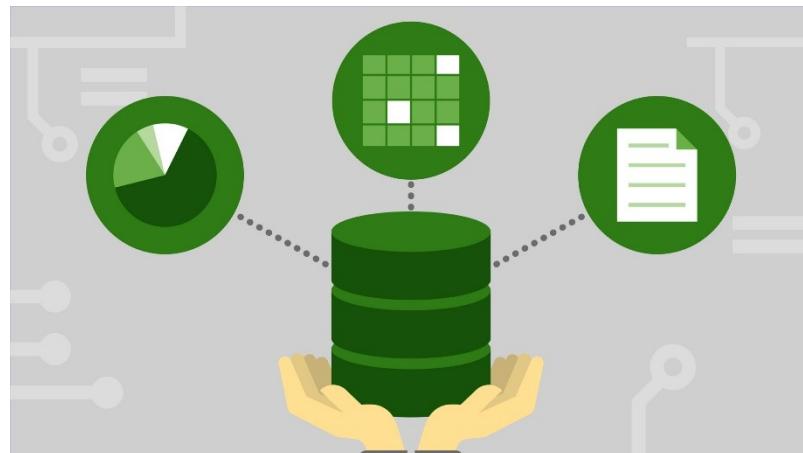


به نام خدا



دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده برق و کامپیوتر



آزمایشگاه پایگاه داده

دستور کار شماره ۲

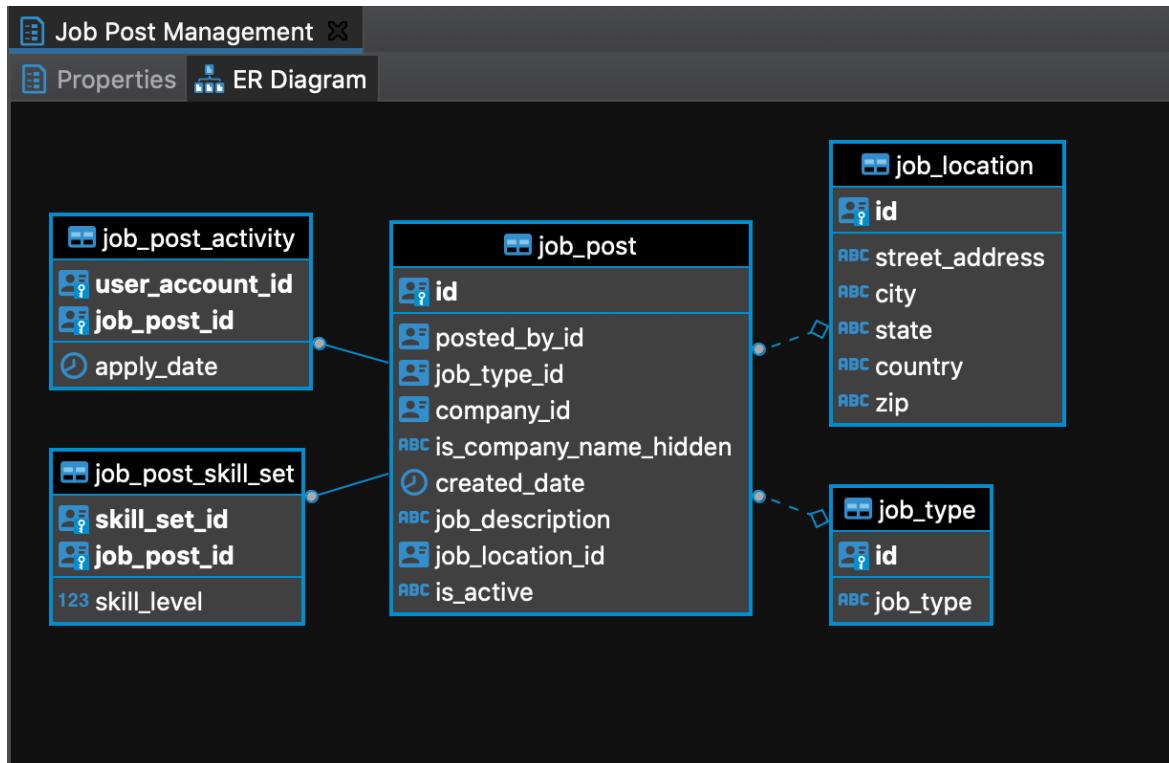
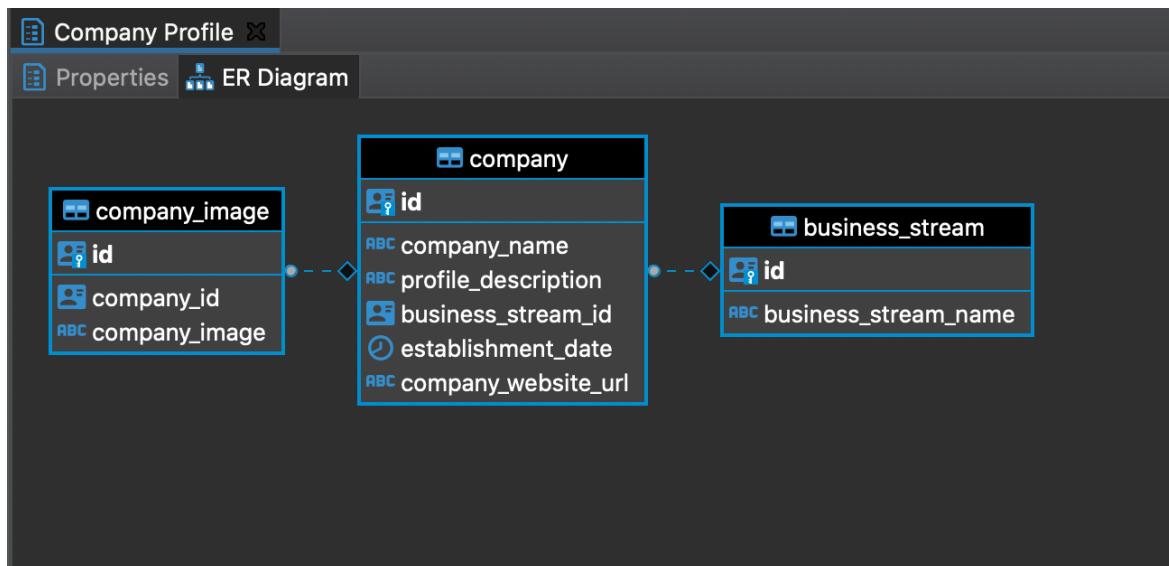
آتبیه آرمین

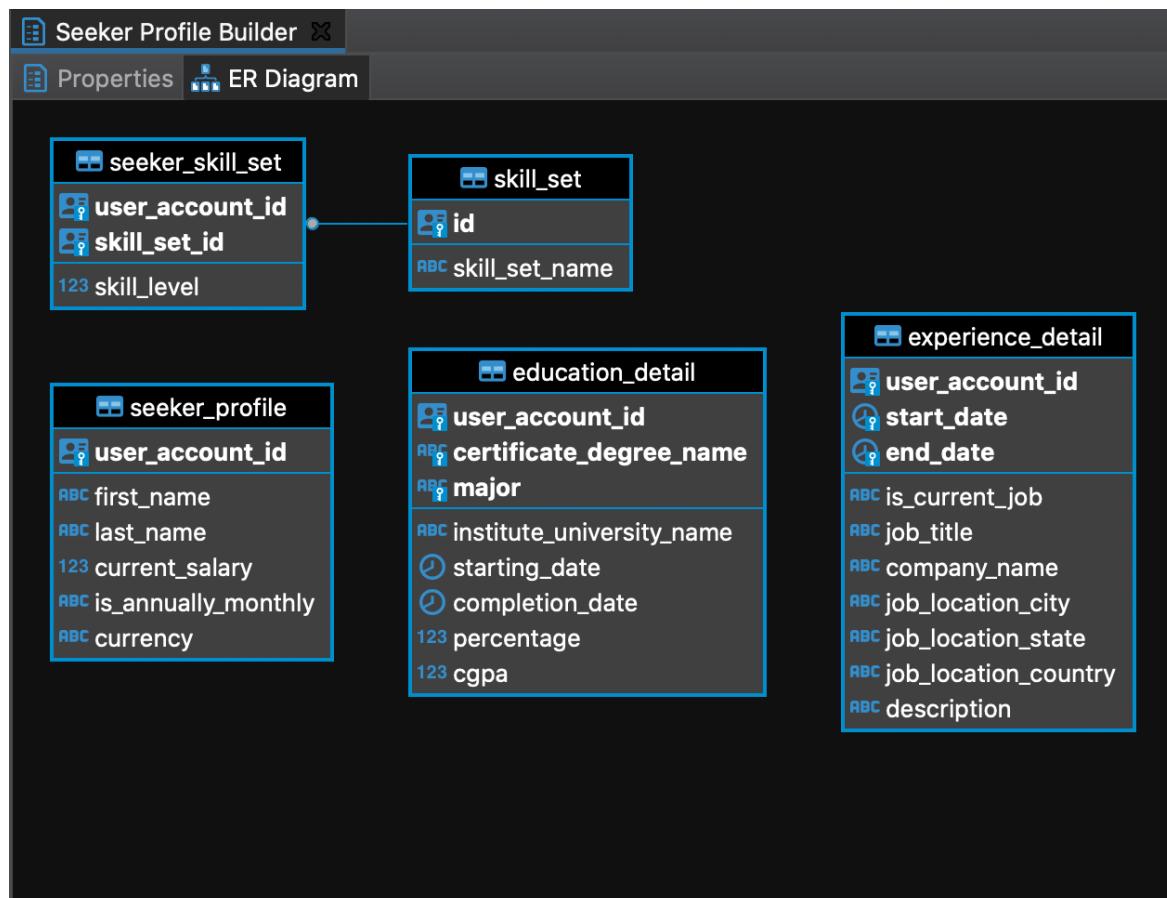
۸۱۰۱۹۷۶۴۸

۱۴۰۰ مهرماه

گزارش دستورکار انجام شده

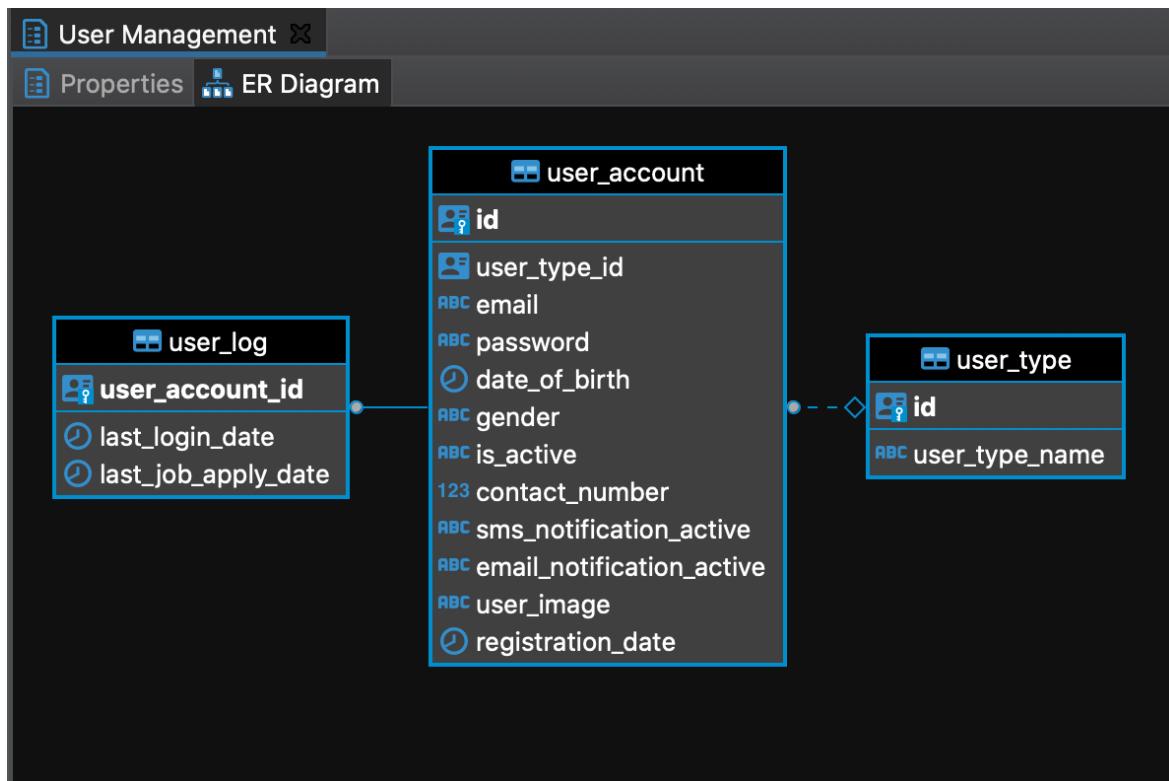
در بخش اول باید جداول دیتابیس مربوطه را در dbbeaver وارد کنیم. در ادامه عکس نمودارهای ER مربوطه را میبینید.





- نکته : نمودار بالا با آنچه در لینک مربوطه کشیده است در وجود ارتباط بین جداول فرق دارد زیرا در متن توضیح ها گفته شده است که `user_account_id` در هر جدول از جدول `user_account` گرفته میشود. بنابراین ارتباط دیگری بین این جداول صورت نمیگیرد.

در ادامه برای بعضی از کارها در انجام این آزمایش ویدئوهایی گرفته ام که در نسخه pdf قابل مشاهده نخواهند بود. این ویدئوها با اسم های مشخص در فolder این گزارش قرار دارند.



حال با دستورات SQL به ساخت یک جدول اضافه میپردازیم که در فیلم زیر روند انجام این کار را مشاهده میکنید.

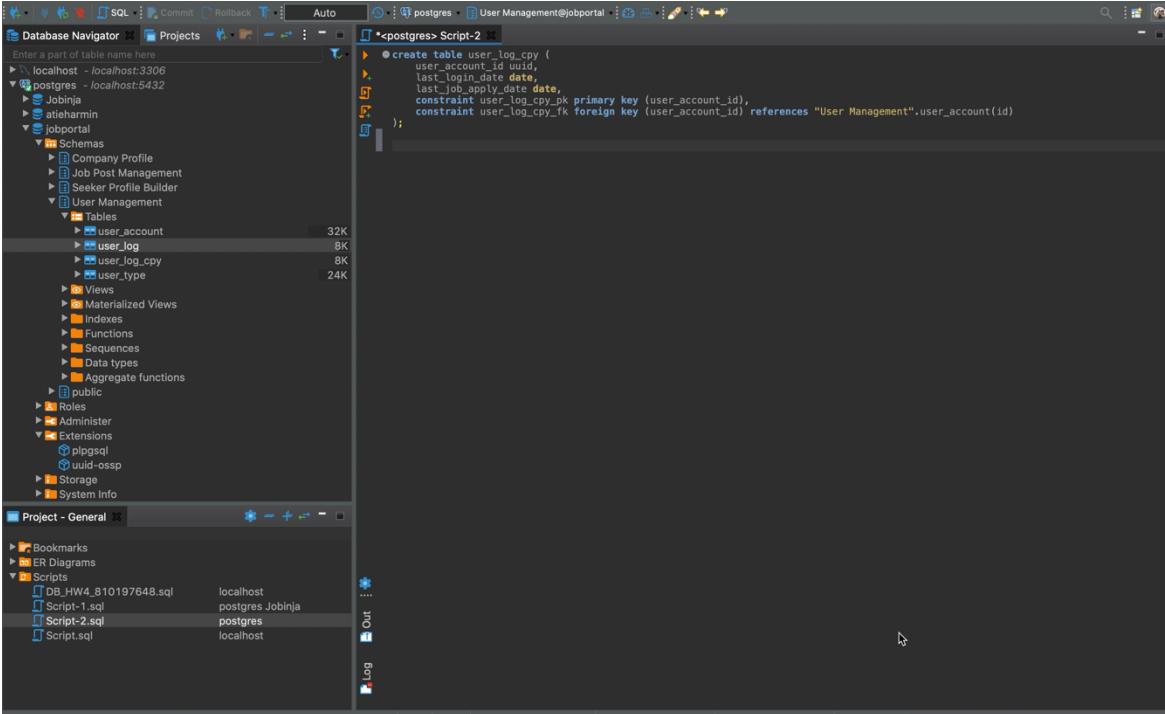
The screenshot shows the pgAdmin interface with the `user_log` table selected in the database navigator. The properties tab is open, showing the table name as `user_log`, owner as `postgres`, and a DDL section containing the following SQL code:

```

-- DROP TABLE "User Management".user_log;
CREATE TABLE "User Management".user_log (
    id uuid NOT NULL DEFAULT uuid_generate_v4(),
    user_account_id uuid NULL,
    email varchar(255) NULL,
    password varchar(100) NULL,
    date_of_birth date NULL,
    gender bpchar(1) NULL,
    is_active bpchar(1) NULL,
    contact_number numeric NULL,
    sms_notification_active bpchar(1) NULL,
    email_notification_active bpchar(1) NULL,
    user_image varchar(255) NULL,
    registration_date date NULL,
    CONSTRAINT user_log_pk PRIMARY KEY (id),
    CONSTRAINT user_log_fk FOREIGN KEY (user_account_id) REFERENCES "User Management".user_account(id)
);

```

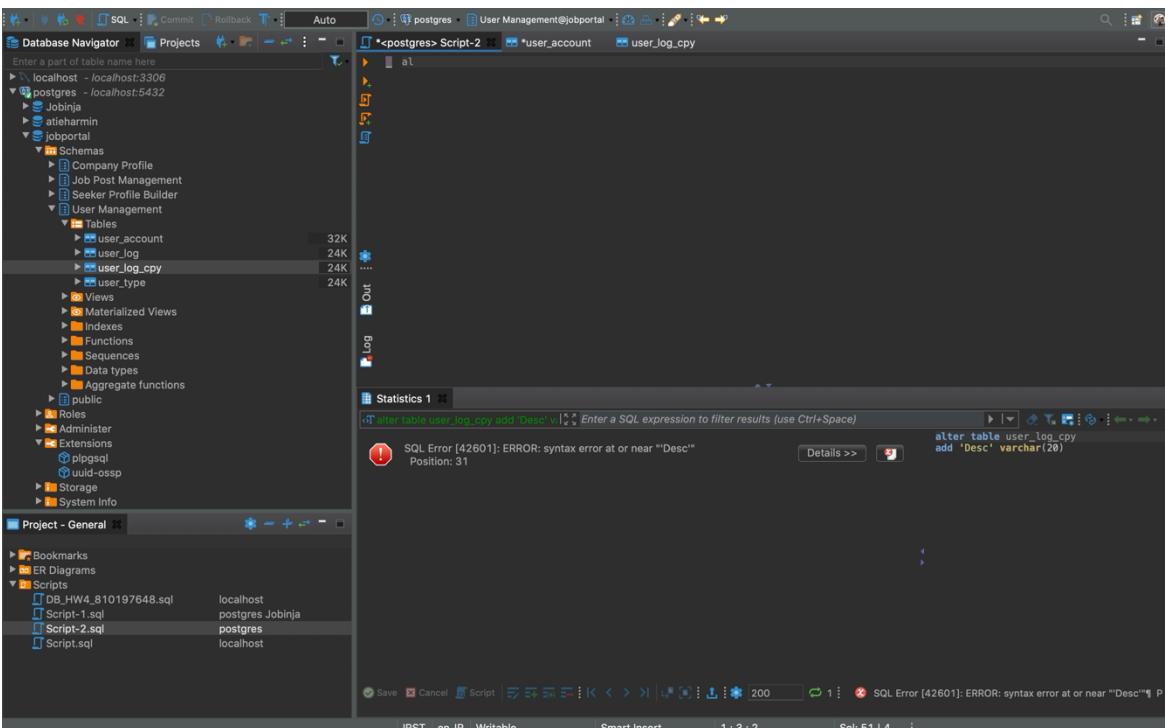
در هر جدول یک رکورد اضافه کردم. همچنین در جدولی که در قسمت قبل با دستورات sql ساخته شده بود هم یک رکورد دوباره با دستورات sql ایجاد کردم که در زیر روند این کار را هم میتوانید مشاهده کنید.



```

create table user_log_cpy (
    user_account_id uuid,
    last_login_date date,
    last_job_apply_date date,
    constraint user_log_cpy_pk primary key (user_account_id),
    constraint user_log_cpy_fk foreign key (user_account_id) references "User Management".user_account(id)
);
  
```

در ادامه یک ستون به این جدول اضافه شده و یک ستون از این جدول پاک کردم.

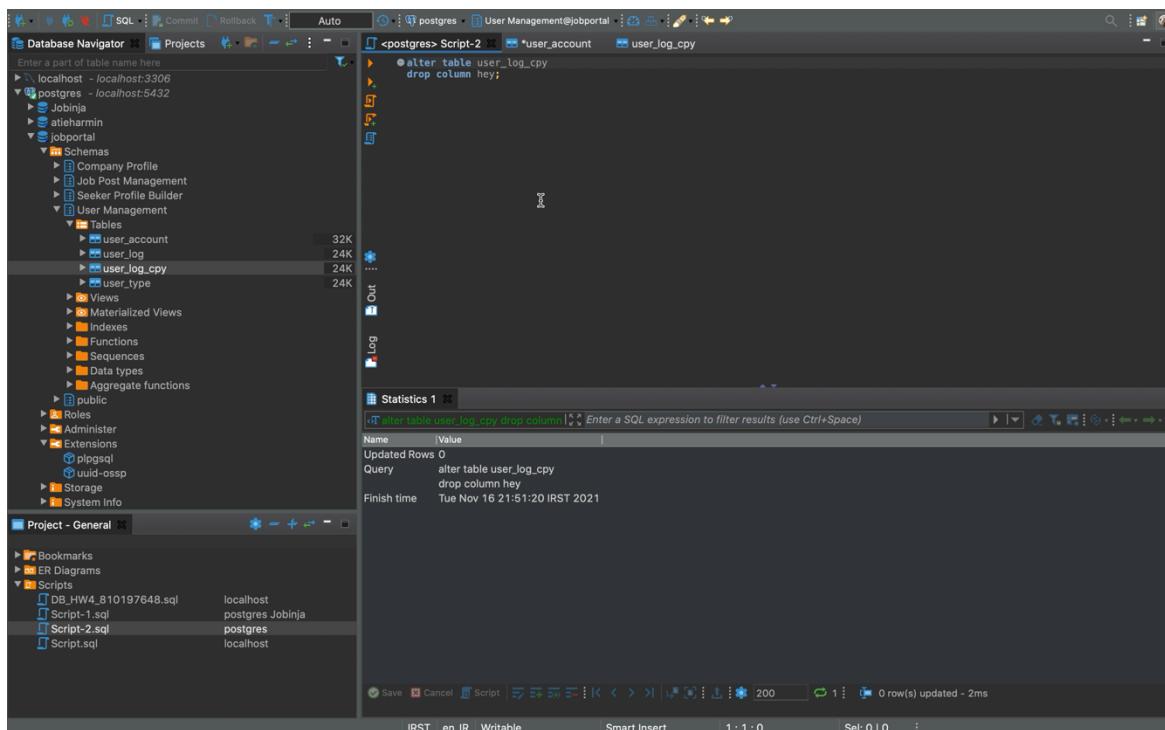


```

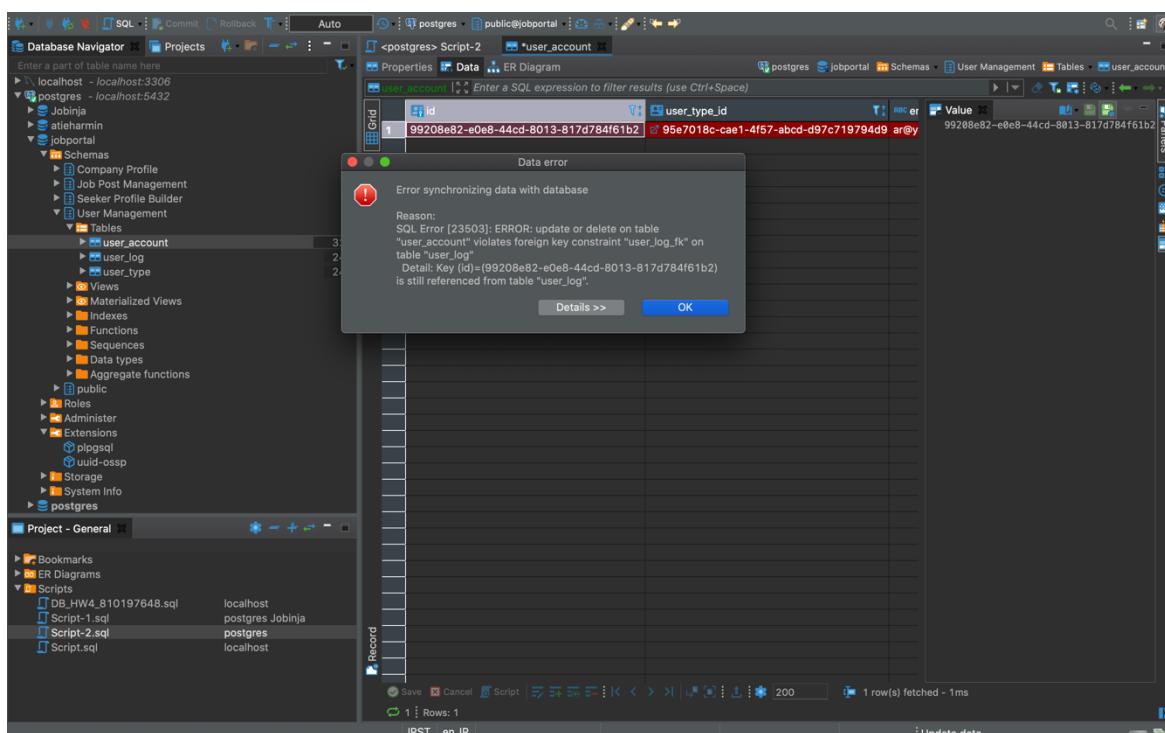
alter table user_log_cpy add Desc varchar(20);
  
```

SQL Error [42601]: ERROR: syntax error at or near "Desc"
Position: 31

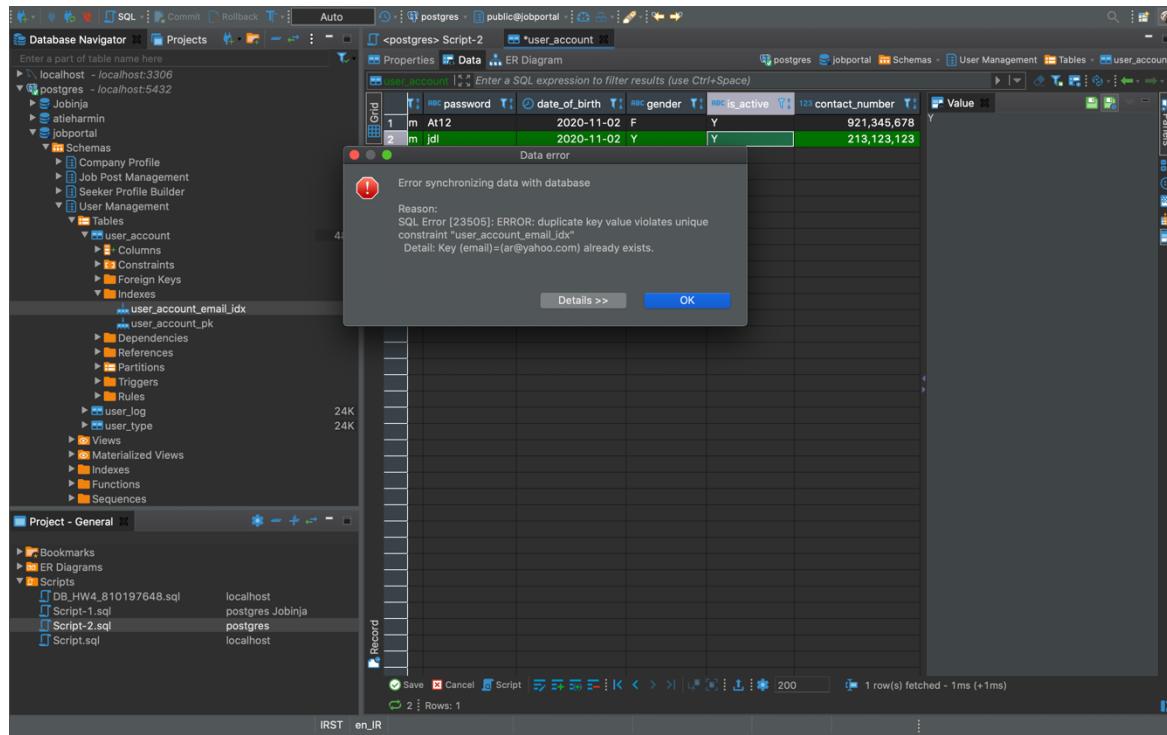
حال این جدول را به طور کلی پاک می‌کنیم.



حال میخواهیم تمام داده های جدول user_account را پاک کنیم که به ارور زیر بر میخوریم که حذف این رکورد باعث حذف رکورد های وابسته می شود و نباید این کار را انجام دهیم.



سپس با ایجاد یک یونیک روی email user account با وارد کردن یک user account با email تکراری به ارور زیر بر میخوریم.

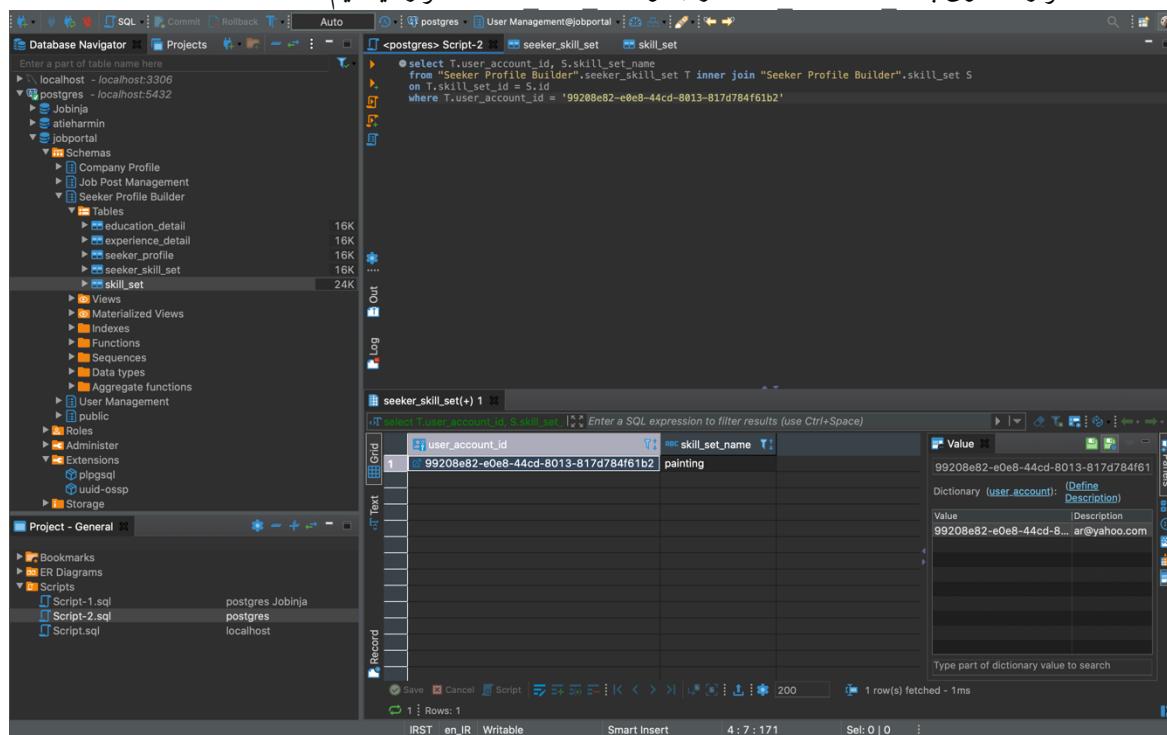


در ادامه به انجام برخی دستورات sql و پاسخ سوالات میپردازیم.

۱) به ازای یک کارجوی خاص، لیست مهارت های کاری و آکادمیک او را نمایش دهیم.

برای این کار ابتدا یک inner join روی جداول seeker_skill_set و skill_set میزنیم و سپس id کارجوی مورد

نظر را مساوی با user account id در جدول seeker skill set قرار میدهیم.



۲) چه آگهی هایی هنوز هیچ متقاضی ای برای آن پیدا نشده است؟

برای این کار چک میکنیم که id کدام آگهی ها در لیست آگهی های مورد تقاضا قرار ندارند.

```

SELECT J.id
FROM "Job Post Management".job_post J
WHERE J.id NOT IN (SELECT JA.job_post_id
                   FROM "Job Post Management".job_post_activity JA)

```

The screenshot shows the pgAdmin III interface. The left sidebar displays the database structure under 'Database Navigator'. The main area shows a SQL editor with the above query and a results grid below it. The results grid is currently empty, indicating no rows found.

۳) به ازای یک آگهی خاص، لیست تمام درخواست ها بر اساس زمان به صورت نزولی را بیابید.

برای این کار از دستور order by استفاده میکنیم تا بر اساس زمان مرتب شود.

```

SELECT JA.user_account_id, JA.job_post_id, JA.apply_date
FROM "Job Post Management".job_post_activity JA
INNER JOIN "Job Post Management".job_post J ON JA.job_post_id = J.id
ORDER BY JA.apply_date DESC;

```

The screenshot shows the pgAdmin III interface. The left sidebar displays the database structure under 'Database Navigator'. The main area shows a SQL editor with the above query and a results grid below it. The results grid contains one row of data:

user_account_id	job_post_id	apply_date
99208e82-e0e8-44cd-8013-817d784f61b1	17eb5b3c-fa90-4853-aaa0-b81ba6abda73	99208e82-e0e8-44cd-8013-817d784f61

۴) اگر یک شرکت، نیاز داشته باشد گزارشی از آگهی ها و تعداد متقاضیان هر آگهی داشته باشد، از چه دستوری

باید استفاده کنید؟

از دستور `count()` و `group by` استفاده میکنیم.

```
<postgress> Script-2 > job_post > job_post_activity
```

```
select JA.job_post_id, count(*)
from "Job Post Management".job_post JA
join "Job Post Management".job_post_activity JA
group by JA.job_post_id
```

job_post_id	count
17eb5b3c-fa90-4853-aaa0-b81ba6abda73	1

۵) در سوال قبل اگر فقط بخواهد آگهی هایی که هیچ متقاضی ای ندارند را نمایش دهد، چه تغییری باید اعمال

کنیم؟

از دستور `having` و اضافه کردن شرط `0 = count(JA.user_account_id)` بودن تعداد متقاضی ها استفاده میکنیم.

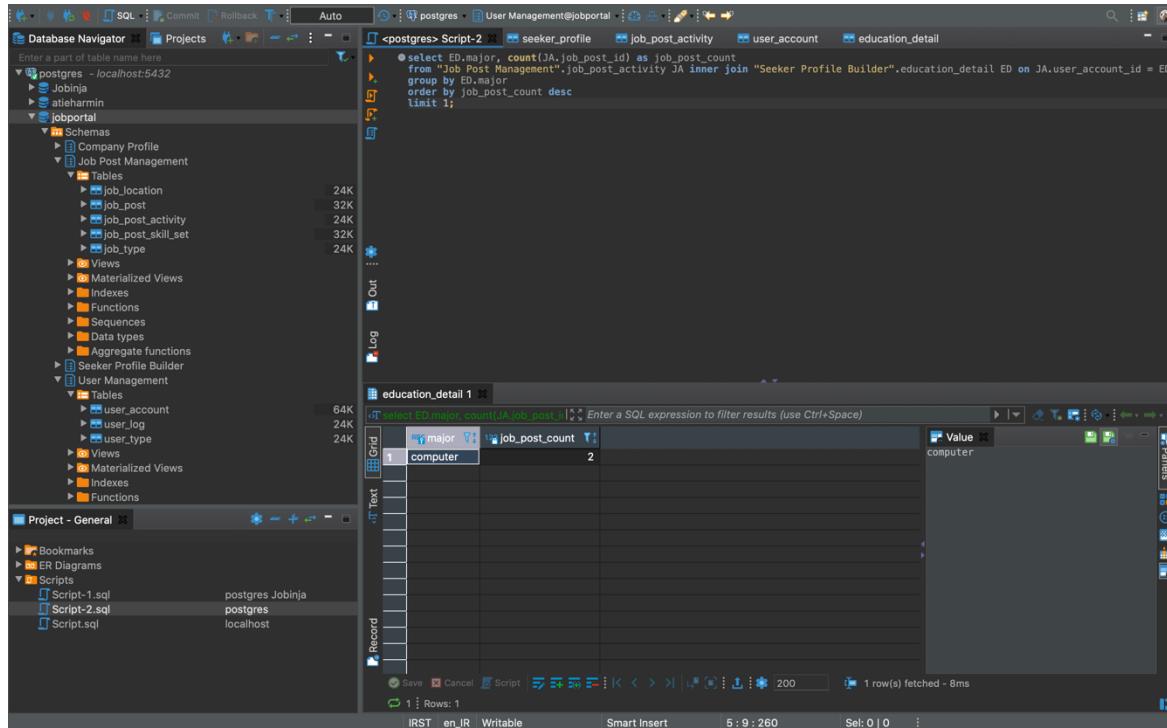
```
<postgress> Script-2 > job_post > job_post_activity
```

```
select J.id
from "Job Post Management".job_post J left outer join "Job Post Management".job_post_activity JA on J.id = JA.job_post_id
group by J.id
having count(JA.user_account_id) = 0
```

J.id
0ffc5f0f-1574-47b3-b707-17a3a73cf310

6) بیشترین درخواست کار ها متعلق به چه رشته ای است؟

برای این کار از inner join استفاده کرده سپس روی رشته group by میکنیم. حال با مرتب کردن و نشان دادن بالاترین میتوانیم رشته ای که بیشترین درخواست کار ها را دارد را مشاهده کنیم.



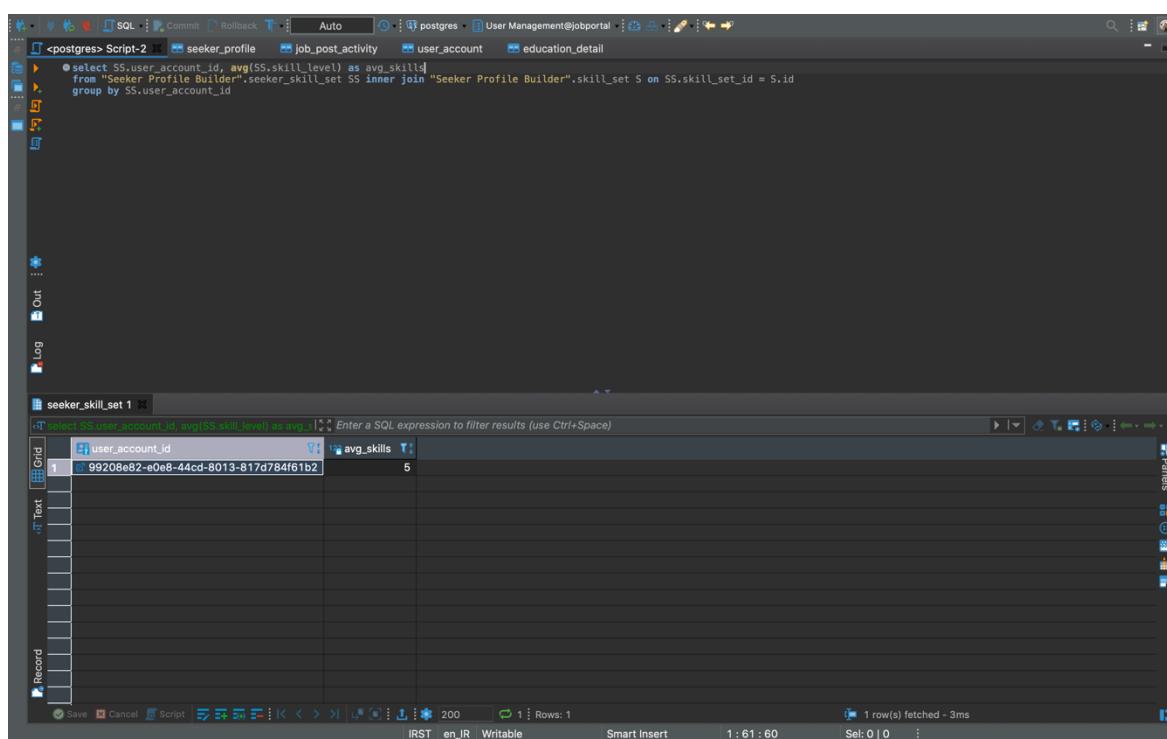
```

<postgres> Script-2
    select ED.major, count(JA.job_post_id) as job_post_count
    from "Job Post Management".job_post JA inner join "Seeker Profile Builder".education_detail ED on JA.user_account_id = ED.user_account_id
    group by ED.major
    order by job_post_count desc
    limit 1;
  
```

The screenshot shows the pgAdmin 4 interface with the following details:

- Database Navigator:** Shows the database structure with the `jobportal` schema expanded, containing `Company Profile`, `Job Post Management`, `Seeker Profile Builder`, and `User Management` modules.
- Script Editor:** Contains the SQL query provided above.
- Result Grid:** Displays the output of the query, which is a single row showing the major `computer` with a count of `2`.
- Project - General:** Shows the current project setup.

7) متوسط skill های هر کارجو را نشان دهید.



```

<postgres> Script-2
    select SS.user_account_id, avg(SS.skill_level) as avg_skills
    from "Seeker Profile Builder".seeker_skill_set SS inner join "Seeker Profile Builder".skill_set S on SS.skill_set_id = S.id
    group by SS.user_account_id;
  
```

The screenshot shows the pgAdmin 4 interface with the following details:

- Database Navigator:** Shows the database structure with the `jobportal` schema expanded, containing `Company Profile`, `Job Post Management`, `Seeker Profile Builder`, and `User Management` modules.
- Script Editor:** Contains the SQL query provided above.
- Result Grid:** Displays the output of the query, which is a single row showing the user account ID `99208e82-e0e8-44cd-8013-817d784f61b2` with an average skill level of `5`.
- Project - General:** Shows the current project setup.

8) کاربرانی را نشان دهید که متقاضی هیچ شغلی نیستند.

The screenshot shows the pgAdmin 4 interface. The top bar displays the connection information: <postgres> Script-2, seeker_profile, job_post_activity, user_account, education_detail. The main area contains a SQL query:

```
select UA.id
from "User Management".user_account UA left outer join "Job Post Management".job_post_activity JA on UA.id = JA.user_account_id
where JA.apply_date is null
```

The results pane shows a table titled "user_account 1" with one row of data:

id

Below the results pane, the status bar indicates "No data - 3ms".