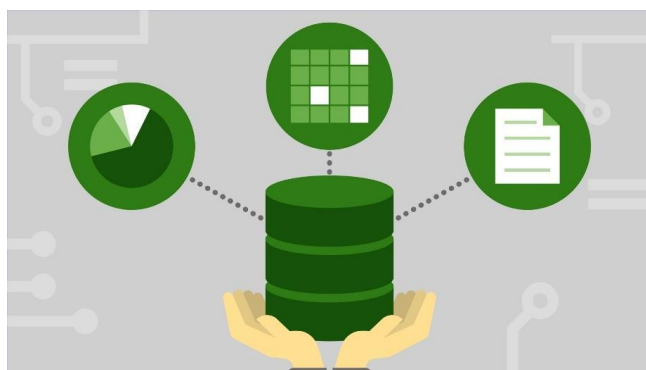


به نام خدا



دانشگاه تهران
پردیس دانشکده‌های فنی
دانشکده برق و کامپیوتر



آزمایشگاه پایگاه داده

دستور کار شماره ششم (دستورات پیشرفته ی SQL)

امیرعلی رایگان

۸۱۰۱۹۷۶۲۳

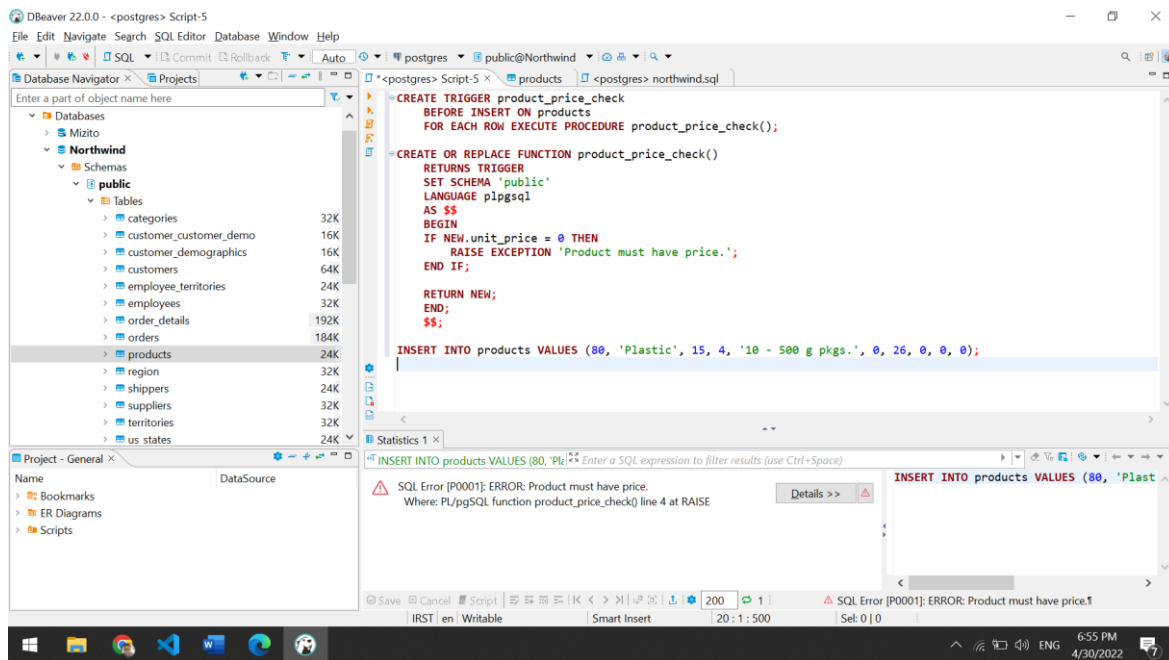
بهار ۱۴۰۱

گزارش دستورکار انجام شده

گام اول. آشنایی با توابع و تریگرها

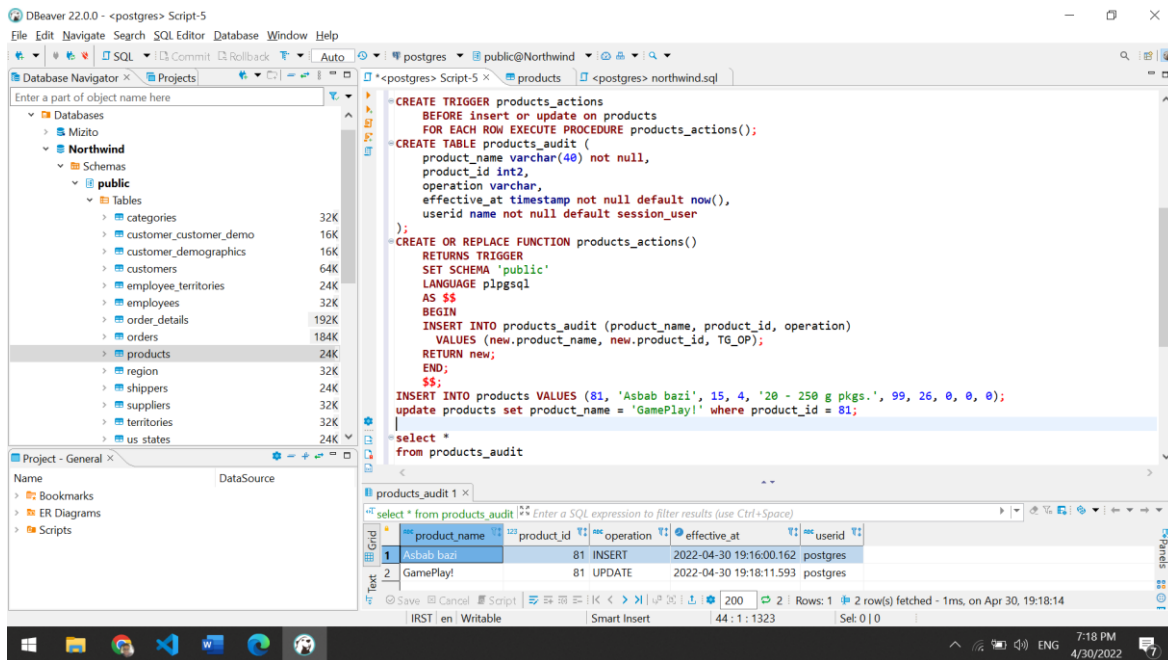
مثال اول

در این نمونه تریگری تعریف کردیم که در هنگام اضافه شدن یک داده جدید به جدول کالاها چک شود که آیا قیمت درست وارد شده یا خیر. اگر قیمت ۰ بود اکسپشنی رها می‌شود و نمی‌گذارد این رکورد به جدول اضافه شود.



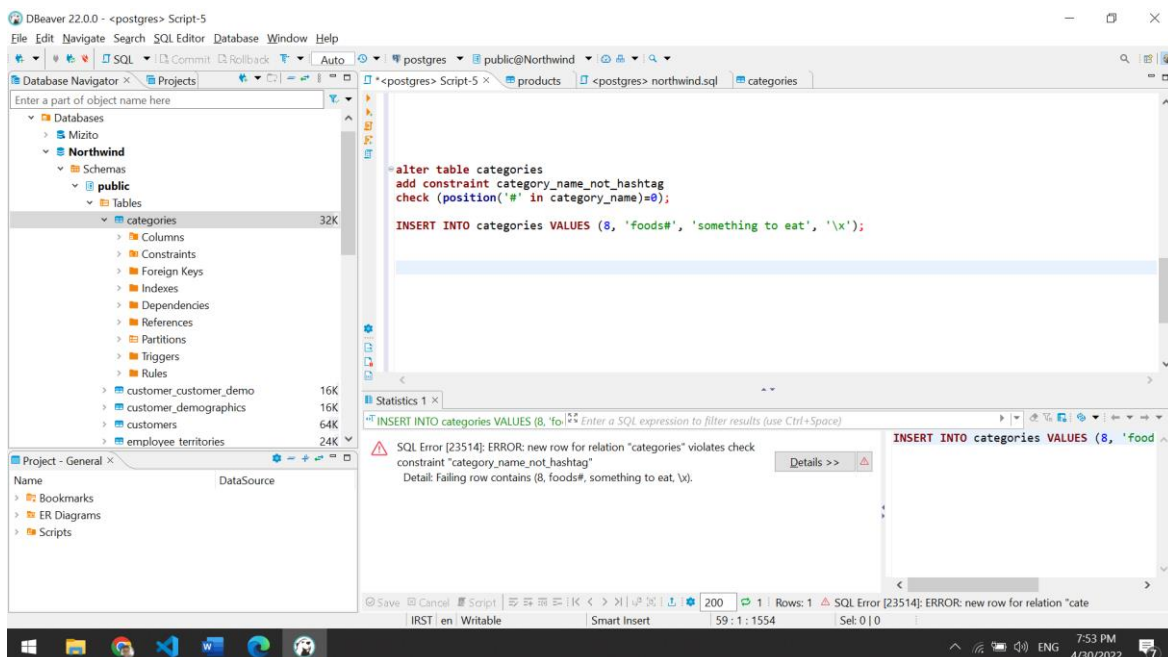
مثال دوم

در این نمونه تریگر جدیدی ساختیم تا تمام اعمال insert یا update که روی جدول کالاها اعمال می‌شود را ذخیره کنیم. در این راستا ابتدا جدول جدید با نام products_audit ساختیم. سپس تریگری تعریف کردیم تا با استفاده از دستور مربوطه بعد از هر بار insert یا update رکورد متناظری در جدول جدید بسازد. سپس این تریگر را روی جدول کالاها ست می‌کنیم. در انتها تستی انجام شده و بعد از نشان دادن جدول جدید می‌بینیم رکورد این اعمال به درستی ذخیره شده است.



مثال سوم

با استفاده از دستور `alter` و همچنین `add constraint` می توانیم محدودیت هایی مثل نمونه اول را به جدول اضافه کنیم. در مثال زیر چک می کنیم که هیچ کاراکتر هشتگی در نام دسته بندی ها نباشد.



گام دوم. توابع پنجره‌ای

مثال اول

مشخص کردن سفارش هایی که برای هر محصول ثبت شده به همراه رتبه ی آن سفارش بر اساس قیمت نهایی آن.

SQL Query:

```
select od.product_id, o.customer_id, od.unit_price * od.quantity as order_price,
ROW_NUMBER() OVER(
PARTITION BY od.product_id
ORDER by od.unit_price * od.quantity desc
) as order_number
from order_details od inner join orders o on o.order_id = od.order_id
```

product_id	customer_id	order_price	order_number
1	LINOD	72	36
1	DUMON	54	37
1	WILMK	36	38
2	SAVEA	1,900	1
2	HILAA	950	2
2	QUICK	950	3
2	QUICK	911.9999885559	4
2	OTTIK	760	5

مثال دوم

مشخص کردن اینکه هر مشتری از بین سفارش کنونی و دو سفارش قبلی در چند تا از آنها تخفیف کسب کرده است.

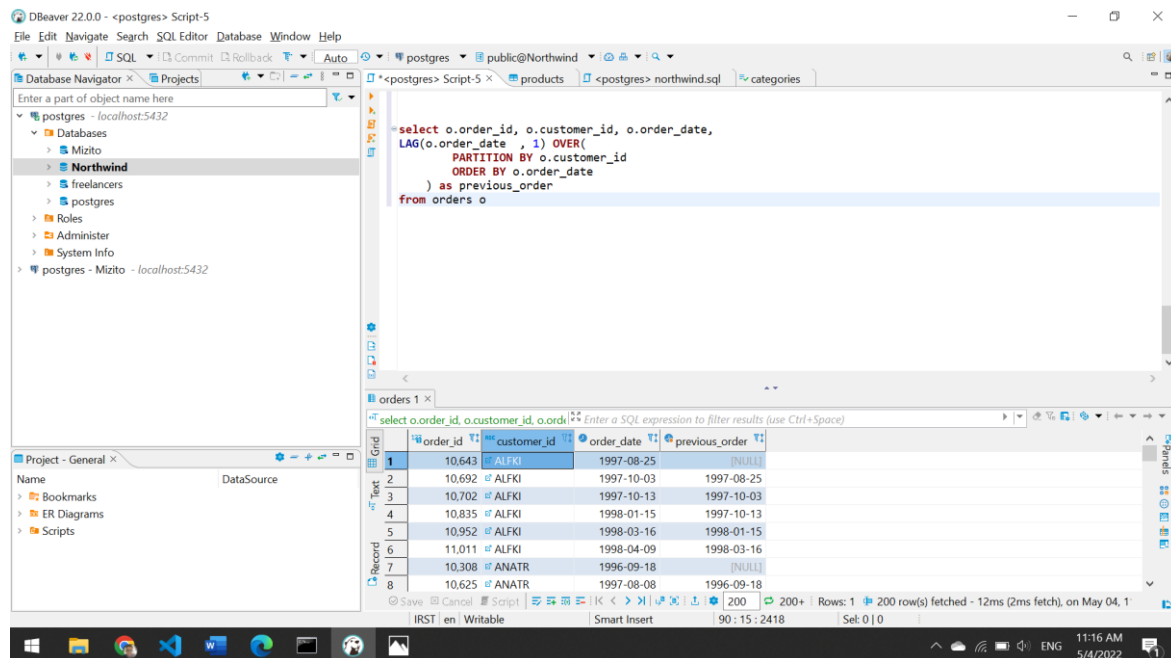
SQL Query:

```
select od.product_id, o.customer_id, o.order_date, od.discount,
SUM(
CASE
WHEN od.discount!=0 THEN 1
ELSE 0
END
) OVER(
PARTITION BY o.customer_id
order by o.order_date ROWS BETWEEN 2 PRECEDING AND 0 PRECEDING
) as discount_number
from order_details od inner join orders o on o.order_id = od.order_id
```

product_id	customer_id	order_date	discount	discount_number
46	ALFKI	1997-08-25	0.25	1
28	ALFKI	1997-08-25	0.25	2
39	ALFKI	1997-08-25	0.25	3
63	ALFKI	1997-10-03	0	2
3	ALFKI	1997-10-13	0	1
76	ALFKI	1997-10-13	0	0
77	ALFKI	1998-01-15	0.200000003	1
59	ALFKI	1998-01-15	0	1

مثال سوم

مشخص کردن تاریخ سفارش قبلی هر مشتری.



گام سوم. سوالات تکمیلی

سوال اول

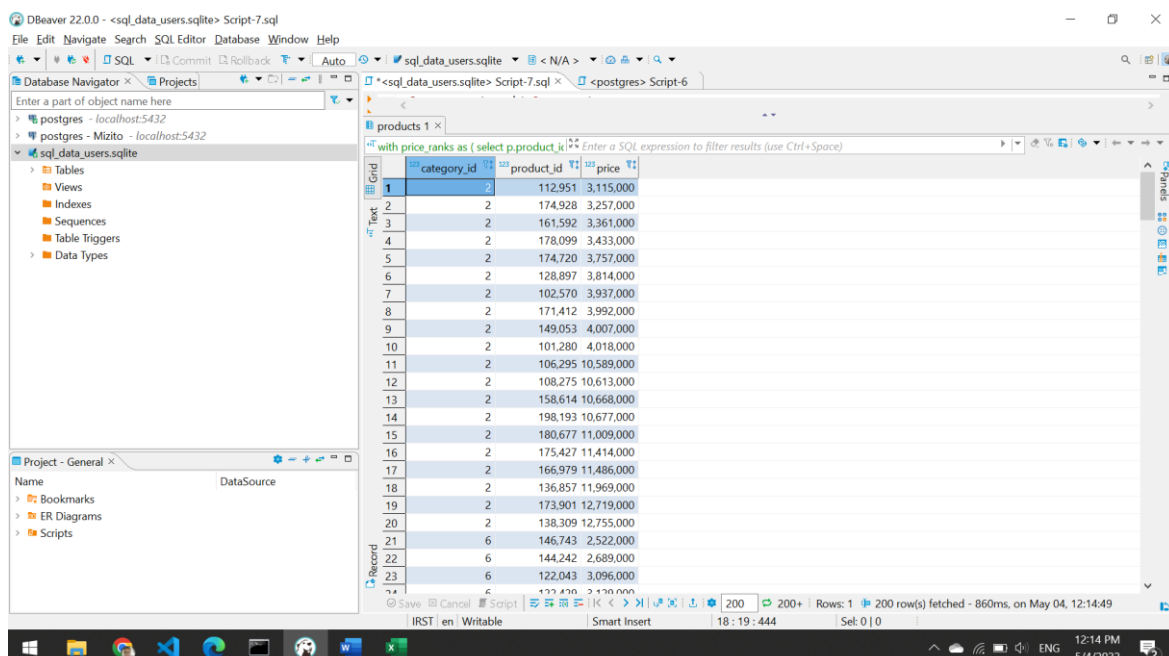
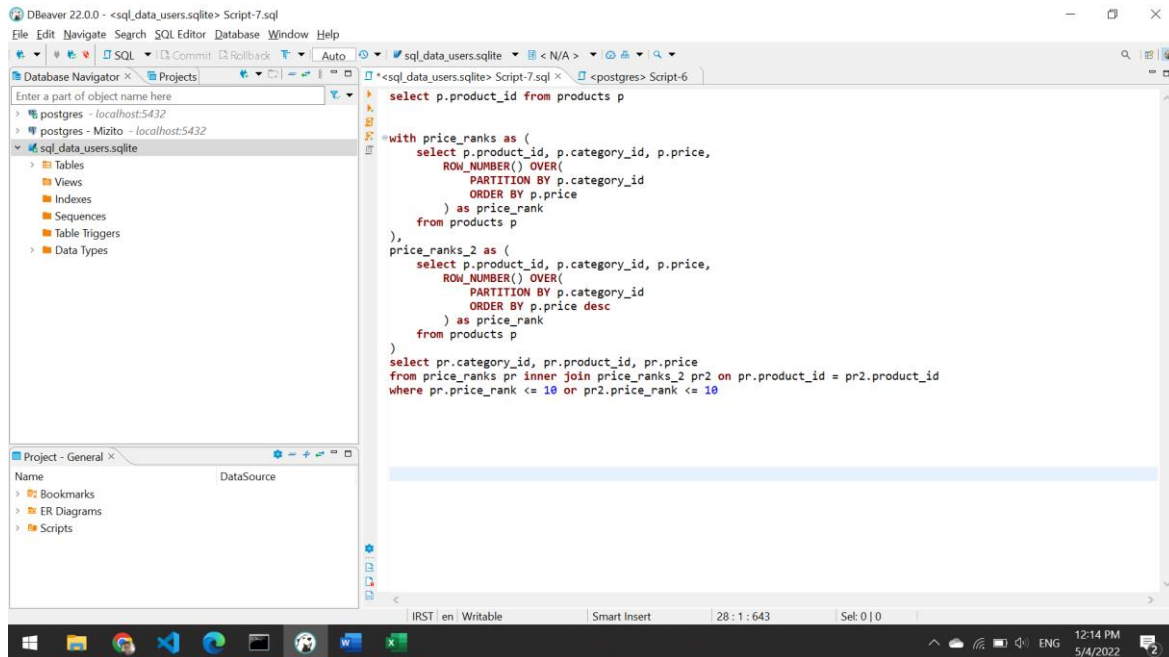
در اولین بخش این سوال پیدا کردن کاربرانی بود که تا به حال خرید انجام داده اند. با استفاده از دستور **case** ستون جدیدی ایجاد می کردیم که تعداد خرید های آنها در آن ذخیره شده و سپس در آن ستون کاربرانی که عدد بزرگ تر از ۰ دارند را پیدا می کنیم.

در بخش دوم کد **sql** به دنبال کسانی هستیم که بیشتر از ۱ آدرس برای آنها ثبت شده است. این کار را هم با شمارش **zipcode** های منحصر به فرد انجام می دهیم.

در بخش سوم کد تعداد کلیک ها، خرید ها و لاگین های هر کاربر را به تفکیک **session_id** می شماریم و در ردیف های جدید نمایش می دهیم. در این قسمت هم از دستور **case** برای شمارش فیلدهای استفاده می شود که بین چندین ردیف مقدار یکسان دارند.

سوال دوم

با ایجاد دو جدول موقتی که هر کدام با استفاده از یک تابع پنجره ای رتبه ی قیمتی هر کالا را حساب کرده اند (یکی به صورت صعودی دیگری به صورت نزولی) و انتخاب ۱۰ کالای اول هر کدام می توانیم این جدول را مطابق اکسل قرار داده شده ایجاد کنیم.



مشکلات و توضیحات تکمیلی

کد های آماده سوال اول در سوالات تکمیلی درست اجرا نمی شد.

آنچه آموختم / پیشنهادات

آشنایی با دستورات پیشرفته ی sql که بیشتر در محیط کاری استفاده می شوند خیلی جالب و مفید بود.