


تمرین ۵ از پایگاه داده

مینا فریدی، ۹۵۳۱۰۶۵

۱- تابع زیر را ابتدا تعریف میکنیم و سپس آن را برای مقادیر مختلف که کد آن قرار داده شده اجرا میکنیم و نتایج زیر بدست می‌آید.

```
1 create or replace function phonecheck(in phone_number character varying)
2 returns table (type text, city text, city_code text, last_8_digits text)
3 language 'plpgsql'
4 as
5 $$
6 begin
7     if phone_number ~ '09\d{9}$' then
8         return query values ('mobile phone number','null', 'null', 'null');
9     elseif phone_number ~ '031\d{8}$' then
10        return query values ('telephone', 'isfahan', '031', substring(phone_number from 4 for 8 ));
11    elseif phone_number ~ '021\d{8}$' then
12        return query values ('telephone', 'tehran', '021', substring(phone_number from 4 for 8 ));
13    else return query values ('not valid', 'null', 'null', 'null');
14    end if;
15 end
16 $$
17
18
```

```
1 select phonecheck('03123456789'); --esfahan|
2 --select phonecheck('02123456789'); --tehran
3 --select phonecheck('09123456789'); --phone
4 --select phonecheck('12345678'); --invalid
```

phonecheck		
record		
1	(telephone,isfahan,031,23456789)	

۲- در این سوال تابع filmsWithin دو ورودی تاریخ اولیه و تاریخ دوم را می‌گیرد و فیلم‌هایی که بین این دو تاریخ بوده اند را خروجی می‌دهد. در عکس پایین نتیجه اجرای تابع نیز نمایش داده شده‌است.

```






1 create or replace function filmsWithin(in fromDate date, in toDate date)
2 returns table (rental_id integer, film_title character varying, customer_id smallint, rental_date timestamp)
3 language 'plpgsql'
4 as
5 $$
6 begin
7     return query
8     select rental.rental_id as rental_id, film.title as film_title,
9           rental.customer_id as cumstomer_id, rental.rental_date as rental_date
10    from rental inner join inventory on rental.inventory_id=inventory.inventory_id
11   inner join film on film.film_id=inventory.film_id
12  where rental.rental_date>fromDate
13     and rental.rental_date < toDate;
14 end
15 $$
16
17
18

```

[Query Editor](#) [Query History](#)

```
1 select * from public.filmsWithin('20050530','20050601');
```

[Data Output](#) [Explain](#) [Messages](#) [Notifications](#)

	 rental_id integer	 film_title character varying	 customer_id smallint	 rental_date timestamp without time zone	
1	972	Academy Dinosaur	411	2005-05-30 20:21:07	
2	1033	Adaptation Holes	25	2005-05-31 04:50:07	
3	995	Affair Prejudice	150	2005-05-31 00:06:02	
4	1058	Airport Pollock	116	2005-05-31 08:04:17	
5	983	Alaska Phantom	115	2005-05-30 22:15:51	
6	1011	Anaconda Confessions	164	2005-05-31 02:05:39	
7	968	Anaconda Confessions	576	2005-05-30 19:20:03	
8	1003	Annie Identity	161	2005-05-31 00:48:20	
9	1123	Antitrust Tomatoes	448	2005-05-31 16:48:43	
10	1134	Antitrust Tomatoes	191	2005-05-31 19:14:15	
11	1068	Apache Divine	443	2005-05-31 09:32:15	
12	944	Apache Divine	131	2005-05-30 15:26:24	
13	1152	Baby Hall	191	2005-05-31 21:32:17	
14	1118	Backlash Undefeated	510	2005-05-31 16:23:02	
15	928	Barbarella Streetcar	79	2005-05-30 12:27:14	
16	1079	Bear Graceland	312	2005-05-31 10:48:17	
17	1085	Betrayed Rear	5	2005-05-31 11:15:43	
18	1100	Bilko Anonymous	497	2005-05-31 14:03:21	
19	1149	Blackout Private	326	2005-05-31 21:03:17	
20	874	Blade Polish	52	2005-05-30 05:36:21	

۳- تابع returnRentalsWithin را به این صورت تعریف می کنیم:

```
1 drop function returnedRentalsWithin;
2 create or replace function returnedRentalsWithin(in fromDate date, in toDate date)
3 returns table (customer_id integer, first_Name character varying, last_Name character varying,
4               rental_id integer, return_date timestamp)
5 language 'plpgsql'
6 as
7 $$
8 begin
9     return query
10    select customer.customer_id, customer.first_Name, customer.last_Name,
11           rentalsWithin.rental_id, rentalsWithin.return_date
12    from filmsWithin(fromDate,toDate) rentalsWithin inner join customer
13    on customer.customer_id= rentalsWithin.customer_id
14    where rentalsWithin.return_date<toDate;
15 end
16 $$
17
18
```

Data Output Explain Messages Notifications

CREATE FUNCTION

Query returned successfully in 290 msec.

```
1 --select * from public.filmsWithin('20050530','20050601');
2 select * from public.returnedrentalsWithin('20050530','20050601');
3
```

Data Output Explain Messages Notifications

	customer_id integer	first_name character varying	last_name character varying	rental_id integer	return_date timestamp without time zone
1	576	Morris	Mccarter	968	2005-05-31 18:17:03
2	551	Clayton	Barbee	969	2005-05-31 21:14:48
3	198	Elsie	Kelley	932	2005-05-31 10:30:36
4	197	Sue	Peters	918	2005-05-31 07:55:24
5	167	Sally	Pierce	927	2005-05-31 16:20:40
6	6	Jennifer	Davis	916	2005-05-31 09:06:01
7	496	Tyler	Wren	966	2005-05-31 23:51:37
8	114	Grace	Ellis	889	2005-05-31 07:56:53
9	59	Cheryl	Murphy	951	2005-05-31 19:01:35
10	331	Eric	Robert	996	2005-05-31 21:29:20
11	210	Ella	Oliver	953	2005-05-31 20:34:02
12	313	Donald	Mahon	843	2005-05-31 00:58:24
13	517	Brad	Mccurdy	850	2005-05-31 01:51:12
14	390	Shawn	Heaton	912	2005-05-31 09:31:33
15	502	Brett	Cornwell	965	2005-05-31 17:10:14
16	421	Lee	Hawks	931	2005-05-31 14:28:01
17	28	Cynthia	Young	868	2005-05-31 02:28:55
18	79	Rachel	Barnes	840	2005-05-31 20:39:41
19	17	Donna	Thompson	884	2005-05-31 07:39:32
20	301	Robert	Baughman	955	2005-05-31 11:58:03

۴- ابتدا جدول logs و تابع تریگر را تعریف میکنیم سپس در شکل دوم مشخص میکنیم که تریگر در موقع آپدیت کردن جدول rental اجرا شود.

```
3 create table logs(  
4     customer_id integer,  
5     days integer  
6 );
```

Data Output Explain Messages Notifications

CREATE TABLE

Query returned successfully in 280 msec.

```
1 drop trigger if exists delay_checker on rental;  
2 drop function if exists delay_checker_function;  
3 create function delay_checker_function()  
4 returns trigger  
5 language plpgsql  
6 as  
7 $$  
8 begin  
9     if date_part('day', current_date - new.rental_date) >  
10        (  
11            select f.rental_duration  
12            from rental r inner join inventory i  
13            on r.inventory_id = i.inventory_id  
14            inner join film f  
15            on f.film_id = i.film_id  
16            where r.rental_id = new.rental_id  
17            limit 1  
18        )  
19    then  
20        insert into logs(customer_id, days)  
21        values(new.customer_id, Date_Part('day', current_date-new.rental_date));  
22    end if;  
23    return new;  
24 end
```

Data Output Explain Messages Notifications

CREATE FUNCTION

Query returned successfully in 134 msec.

```

1 create trigger delay_checker
2 before update
3 on rental
4 for each row
5 execute procedure delay_checker_function();

```

Data Output Explain Messages Notifications

CREATE TRIGGER

Query returned successfully in 311 msec.

در ادامه سطری از جدول را طوری تغییر می‌دهیم که اجاره تاخیر داشته باشد. هنگام تغییر دادن این سطر تریگر اجرا می‌شود و مشتری موردنظر به جدول logs اضافه می‌شود:

```

5 update rental
6 set return_date='20060606'
7 where rental_id = 2;
8
9 select *
10 from logs;
11

```

Data Output Explain Messages Notifications

	customer_id integer	days integer
1	459	5684

۵- تابع موردنظر را می‌نویسیم اما قبل از اجرای آن، ابتدا مقدار اولیه ستون‌هایی از جدول rental را نشان می‌دهیم:

```
9 select film_id, title, rental_duration from film order by film_id;
10
```

	film_id [PK] integer	title character varying (255)	rental_duration smallint
1	1	Academy Dinosaur	9
2	2	Ace Goldfinger	6
3	3	Adaptation Holes	10
4	4	Affair Prejudice	8
5	5	African Egg	9
6	6	Agent Truman	6
7	7	Airplane Sierra	9
8	8	Airport Pollock	9
9	9	Alabama Devil	6
10	10	Aladdin Calendar	9
11	11	Alamo Videotape	9
12	12	Alaska Phantom	9
13	13	Ali Forever	7
14	14	Alice Fantasia	9
15	15	Alien Center	8
16	16	Alley Evolution	9
17	17	Alone Trip	6
18	18	Alter Victory	9
19	19	Amadeus Holy	9

حالا تابع زیر را تعریف می‌کنیم:

```
1 drop function if exists increase_rental_duration;
2 create function increase_rental_duration(in inc_amount smallint)
3 returns table(filmId integer, film_title varchar, rentalduration smallint)
4 language 'plpgsql'
5 as
6 $$
7 begin
8     update film
9     set rental_duration = rental_duration + inc_amount;
10
11     return query select film_id, title, rental_duration from film order by film_id;
12 end
13 $$
```

Data Output Explain Messages Notifications






CREATE FUNCTION

Query returned successfully in 252 msec.

تابع را اجرا می‌کنیم و مشاهده می‌شود که ستون سوم افزایش یافته است:

```
11 select* from increase_rental_duration(3::smallint);
12
```

Data Output Explain Messages Notifications

	 filmid integer	 film_title character varying	 rentalduration smallint		
1		1 Academy Dinosaur		12	
2		2 Ace Goldfinger		9	
3		3 Adaptation Holes		13	
4		4 Affair Prejudice		11	
5		5 African Egg		12	
6		6 Agent Truman		9	
7		7 Airplane Sierra		12	
8		8 Airport Pollock		12	
9		9 Alabama Devil		9	
10		10 Aladdin Calendar		12	
11		11 Alamo Videotape		12	
12		12 Alaska Phantom		12	
13		13 Ali Forever		10	
14		14 Alice Fantasia		12	
15		15 Alien Center		11	
16		16 Alley Evolution		12	
17		17 Alone Trip		9	
18		18 Alter Victory		12	
19		19 Amadeus Holy		12	