**به نام خدا**

**پاسخنامه ی تکلیف ششم**

**آذر 1400**

Constraint

1-یک جدول به نام trader ایجاد کنید که ستون های زیر را داشته باشد

trader( trader_id, trader_name, Birth_date,Joined_date,Salary)

و قیود زیر در آن رعایت شود.

حقوق یک کارمند کمتر از 1000 درج نشود.

تاریخ تولد او قبل از تاریخ 01-01-1900 نباشد.

تاریخ عضویت بزرگتر از تاریخ تولد باشد.

یک رکورد در جدول درج کنید که یکی از قیود را رعایت نکند و پیغام خطا را در گزارش بیاورید.

```
CREATE TABLE trader (
trader_id SERIAL PRIMARY KEY,
trader_name VARCHAR (40),
birth_date DATE CHECK (Birth_date> '1900-01-01'),
Joined_date DATE CHECK (joined_date >DOB),
Salary numeric CHECK(salary > 1000)
);
INSERT INTO trader
VALUES(11,'Maryam','1990-02-01','2010-01-01',900)
```

```
INSERT INTO trader VALUES(11,'Maryam','1990-02-01','2010-01-01',900)

--Q2
```

ata Output  Explain  Messages  Notifications

```
ROR:  new row for relation "trader" violates check constraint "trader_salary_check"
TAIL:  Failing row contains (11, Maryam, 1990-02-01, 2010-01-01, 900).
L state: 23514
```

2- در سیستم دانشگاه می خواهیم در جدول section اجازه ندهیم کلاس هایی که قبل از ساعت 9 شروع می شوند درج شوند . برای انجام این کار constraint جدیدی برای جدول section بنویسید.

```
Alter table section
add check (time_slot_id) in
(select time_slot_id from time_slot where start_hr>=9)

-----------------------------------------------------
create function Avoid_Before_9( time_slot_id1 varchar(4))
    returns boolean
    as $$
        select (case
                when ( 9 <=   all(select  time_slot.start_hr
                        from time_slot
                        where  time_slot.time_slot_id = time_slot_id1))
                then true
                else false
                end)
    $$ language sql

alter table section
add check( case
          when sec_id
          in ('1','2','3') then True
          else Avoid_Before_9(time_slot_id)
          end)
INSERT INTO public.section(
    course_id, sec_id, semester, year, building, room_number,time_slot_id)
    VALUES ('313', '44', 'Fall', 2010, 'Chandler', 804, 'A');
```

```
45  INSERT INTO public.section(
46      course_id, sec_id, semester, year, building, room_number, time_slot_id)
47      VALUES ('313', '44', 'Fall', 2010, 'Chandler', 804, 'A');
48  select distinct(sec_id) from section
```

Data Output  Explain  Messages  Notifications

```
ERROR:  new row for relation "section" violates check constraint "section_check"
DETAIL:  Failing row contains (313, 44, Fall, 2010, Chandler, 804, A).
SQL state: 23514
```

data types

3- در سیستم پایگاه داده ی dvdrental نام مشتریان و نام فیلم هایی را پیدا کنید که تاریخ پرداخت (payment_date) کرایه فیلم توسط مشتری با تاریخ امانت فیلم(rental_date) بیش از یک سال و سه ماه و 15 روز فاصله داشته باشد.

```
select film.title, customer.first_name || ' ' || customer.last_name as
Full_name
from ((payment join customer using (customer_id)
join (rental join inventory Using (inventory_id ))using(rental_id))
join film using (film_id))
where payment.payment_date >= '1 year'::interval + '3.5 months'::interval
+ rental.rental_date
```

4- ذخیره ی فایل های ویدیویی و تصاویر در پایگاه داده به چند صورت امکان پذیر است. معایب و مزایای هر روش را بنویسید.

https://stackoverflow.com/questions/7434530/storing-long-binary-raw-data-strings

index

5- همانطور که می دانیم انواع مختلفی از index وجود دارد و بطور خودکار توسط پایگاه انتخاب می شود، و ما نیز با توجه به نیاز یکی از آن ها را میتوانیم انتخاب کنیم. در سیستم پایگاه داده ی dvdrental برای افزایش سرعت در پرس و جوی زیر یک ایندکس مناسب ایجاد کنید و خروجی پرس و جو و زمان اجرای آن، قبل و بعد از ایجاد ایندکس را گزارش دهید. با استفاده از explain نیز query plan آن را بدست آورید.

```
Explain
select * from film
where title = 'Chamber Italian'
create index index_FilmTitle on film using hash(title);
```

قبل از ایجاد اندیس

Data Output | Explain | Messages | Notifications

```
Successfully run. Total query runtime: 117 msec.
1 rows affected.
```

بعد از ایجاد اندیس

Data Output Explain Messages Notifications

```
Successfully run. Total query runtime: 98 msec.
1 rows affected.
```

Query plan

Data Output Explain Messages Notifications

| | QUERY PLAN<br>text |
|---|---|
| 1 | Index Scan using index_filmtitle on film (cost=0.00..8.02 rows=1 width=384) |
| 2 | Index Cond: ((title)::text = 'Chamber Italian'::text) |

6-در سیستم پایگاه داده ی adventurework ، لیست نام اندیس های schema ی sales را همراه با نام جدول ها و تعریف ایندکس ها بدست آورید.

```
select tablename, indexname, indexdef
from  pg_indexes
where schemaname = 'sales'
```

Data Output Explain Messages Notifications

| | tablename<br>name | indexname<br>name | indexdef<br>text |
|---|---|---|---|
| 1 | countryregioncurrency | PK_CountryReg... | CREATE UNIQUE INDEX "PK_CountryRegionCurrency_CountryRegionCode_CurrencyCode" ON sales.countryregioncurrency USING btree (... |
| 2 | customer | PK_Customer_... | CREATE UNIQUE INDEX "PK_Customer_CustomerID" ON sales.customer USING btree (customerid) |
| 3 | currencyrate | PK_CurrencyRa... | CREATE UNIQUE INDEX "PK_CurrencyRate_CurrencyRateID" ON sales.currencyrate USING btree (currencyrateid) |
| 4 | creditcard | PK_CreditCard_... | CREATE UNIQUE INDEX "PK_CreditCard_CreditCardID" ON sales.creditcard USING btree (creditcardid) |
| 5 | personcreditcard | PK_PersonCred... | CREATE UNIQUE INDEX "PK_PersonCreditCard_BusinessEntityID_CreditCardID" ON sales.personcreditcard USING btree (businessentityid... |
| 6 | specialoffer | PK_SpecialOffe... | CREATE UNIQUE INDEX "PK_SpecialOffer_SpecialOfferID" ON sales.specialoffer USING btree (specialofferid) |

authorization

7- با توجه به پایگاه داده ی دانشگاه به سوالات زیر در مورد دسترسی به پایگاه داده پاسخ دهید.

a. یک نقش به نام role1 ایجاد کنید که قادر به مشاهده اطلاعات جدول instructor باشد. همچنین بتواند مجوز این دسترسی را به بقیه بدهد.

```
create role role1
grant role1 select on instructor with grant option
```

b. نقش دیگری به نام role2 تعریف کنید که علاوه بر دسترسی های role1 قادر به بروزرسانی، حذف و درج اطلاعات هم در جدول course و هم در جدول student باشد.

```
create role role2
grant role1 to role2
grant update,delete,insert on student,course to role2
```

c. یک نقش role3 را ایجاد کنید و به اون دسترسی بدهید که فقط قادر به بروزرسانی dept_name در جدول student باشد. سپس دسترسی های داده شده را از این نقش پس بگیرید.

```
create role role3
grant update(dept_name) on table student to role3
revoke all on table student from role3
```

d. نقش role4 را ایجاد کنید و به او دسترسی بدهید که بتواند وضعیت اخذ دروس توسط دانشجویان دانشکده ی برق و کامپیوتر را ببیند. نام درس و نام دانشجو نیز نمایش داده شود. ( راهنمایی: در صورت نیاز قبل از اعطای دسترسی یک view ایجاد کنید.)

```
create view view_ELecCom_Student as
select * from ((student join takes using(ID) )
              join (select course_id, title from course) as course1 using
(course_id))
where student.dept_name ='Comp. Sci.' or student.dept_name ='Elec. Eng.'
create role role4
grant select on view_ELecCom_Student to role4
```

## transaction

8- به سوالات زیر در مورد تراکنش پاسخ دهید.

a. یک جدول به نام accounts ایجاد کنید به طوری که به صورت اتوماتیک با درج یک رکورد یک مقدار پیش فرض را به شناسه ی جدول اختصاص دهد. همچنین دو ستون name و balance را داشته باشد.( ویژگی balance می تواند حداکثر 12 رقم صحیح و 3 رقم اعشار داشته باشد)

دستور زیر را اجرا کنید

Insert into accounts (name, balance) values (‘ Amir’, 1000)

```
CREATE TABLE accounts (
id INT GENERATED BY DEFAULT AS IDENTITY,
name VARCHAR(100) NOT NULL,
balance Dec(15,3) NOT NULL,
PRIMARY KEY(id)
```

```
)
INSERT INTO accounts(name,balance)VALUES('Amir',1000)
```

b.یک رکورد جدید با نام Ali ایجاد کنید سپس در قالب یک تراکنش 200 از حساب Amir کم کنید و به حساب Ali اضافه کنید. گزارش log را بعد از تغییر حساب Amir و Ali و بعد از پایان تراکنش ارائه دهید. سپس سعی کنید با دستور rollback تغییرات داده شده را برگردانید. پیغام سیستم را در گزارش بیاورید. در نظر داشته باشید برای مشاهده ی log ها در تراکنش های postgresql باید قسمت dashboard و سپس سریرگ lock را بررسی کرد.

```
INSERT INTO accounts(name,balance)VALUES('Ali',2000)
BEGIN
UPDATE accounts SET balance=balance-200 Where id=1
UPDATE accounts SET balance=balance+200 Where id=2
COMMIT
SELECT * FROM ACCOUNTS
rollback
```

گزارش log بعد از تغییر هر دو حساب

Sessions Locks Prepared Transactions Search

| PID | Lock type | Target relation | Page | Tuple | vXID (target) | XID (target) | Class | Object ID | vXID (owner) | Mode | Granted? |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 7896 | relation | pg_locks | | | | | | | 4/13725 | AccessShareLock | true |
| 15988 | relation | accounts_pkey | | | | | | | 9/2993 | RowExclusiveLock | true |
| 15988 | relation | accounts | | | | | | | 9/2993 | RowExclusiveLock | true |

گزارش log بعد از commit

Sessions Locks Prepared Transactions Search

| PID | Lock type | Target relation | Page | Tuple | vXID (target) | XID (target) | Class | Object ID | vXID (owner) | Mode | Granted? |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 7896 | relation | pg_locks | | | | | | | 4/13813 | AccessShareLock | true |

| | id [PK] integer | name character varying (100) | balance numeric (15,3) |
|---|---|---|---|
| 1 | 1 | Amir | 800.000 |
| 2 | 2 | Ali | 2200.000 |

Rollback

```
22  rollback
```

Data Output  Explain  Messages  Notifications

```
WARNING:  there is no transaction in progress
ROLLBACK
```

C. در قالب یک تراکنش از حساب  علی 150 کم کنید و به حساب Amir اضافه کنید. قبل از تمام شدن تراکنش از دستور Rollback استفاده کنید. خروجی های جدول accounts و log را بعد از تغییرات حساب ها و بعد از rollback در گزارش خود بیاورید و دلیل تغییرات log را توضیح دهید.

```
BEGIN
UPDATE accounts SET balance=balance-150 Where id=2
UPDATE accounts SET balance=balance+150 Where id=1
rollback
SELECT * FROM ACCOUNTS
```

بعد از تغییرات حساب ها

Data Output  Explain  Messages  Notifications

| | id [PK] integer | name character varying (100) | balance numeric (15,3) |
|---|---|---|---|
| 1 | 2 | Ali | 1850.000 |
| 2 | 1 | Amir | 950.000 |

| PID | Lock type | Target relation | Page | Tuple | vXID (target) | XID (target) | Class | Object ID | vXID (owner) | Mode | Granted? |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 7896 | relation | pg_locks | | | | | | | 4/15159 | AccessShareLock | true |
| 15988 | relation | accounts_pkey | | | | | | | 9/3078 | RowExclusiveLock | true |
| 15988 | relation | accounts | | | | | | | 9/3078 | RowExclusiveLock | true |

بعد از rollback

Data Output  Explain  Messages  Notifications

| | id [PK] integer | name character varying (100) | balance numeric (15,3) |
|---|---|---|---|
| 1 | 1 | Amir | 800.000 |
| 2 | 2 | Ali | 2200.000 |

Server activity

Sessions | Locks | Prepared Transactions | Search

| PID | Lock type | Target relation | Page | Tuple | vXID (target) | XID (target) | Class | Object ID | vXID (owner) | Mode | Granted? |
|---|---|---|---|---|---|---|---|---|---|---|---|
| 7896 | relation | pg_locks | | | | | | | 4/15262 | AccessShareLock | true |

دلیل تغییرات log این است که بعد از شروع تراکنش و قبل از commit و یا rollback گزارش تغییرات در log ذخیره می شود تا در صورت خطا یا rollback مقدار اولیه با استفاده از log برگردانده شود.

## function

9- با توجه به پایگاه داده ی dvdrental یک function بنویسید که نام یک ناحیه (district) را بگیرد و نام مشتریان آن ناحیه به همراه آخرین فیلمی که به امانت گرفته اند را در خروجی نمایش دهد.
برای ناحیه ی Alberta و یک ناحیه ی دلخواه، تابع را اجرا کنید.

```
CREATE FUNCTION Func(dis_name VARCHAR(20))
RETURNS TABLE (first_name VARCHAR(20),last_name VARCHAR(20), title
VARCHAR(255)) LANGUAGE plpgsql AS
$$
BEGIN
RETURN QUERY SELECT customer.first_name, customer.last_name,(
    SELECT film.title
    FROM film
    INNER JOIN inventory ON (inventory.film_id = film.film_id)
    INNER JOIN rental ON (rental.inventory_id=inventory.inventory_id and
rental.customer_id = customer.customer_id)
    ORDER BY rental.rental_date DESC LIMIT 1)
FROM customer
INNER JOIN  address ON (customer.address_id = address.address_id AND
address.district=dis_name);
end;
$$;

SELECT * FROM Func('Alberta')
```

procedure

10- با توجه به پایگاه داده ی dvdrental یک stored procedure بنویسید که طی یک تراکنش نام دو فیلم را بگیرد و rating فیلم اول را با  فیلم دوم جا به جا کند. این کار را با تعریف متغیر در داخل procedure انجام دهید. تصویر حاصل از خروجی procedure خود را به ازای ورودی دلخواه در پاسخنامه قرار دهید.

```
CREATE PROCEDURE change_Rate(film1 VARCHAR(255), film2 VARCHAR(255))
LANGUAGE plpgsql AS
$$
DECLARE film1_rate "public"."mpaa_rating";
DECLARE film2_rate "public"."mpaa_rating";
BEGIN
    SELECT rating INTO film1_rate
    FROM film
    WHERE title = film1;

    SELECT rating INTO film2_rate
    FROM film
    WHERE title = film2;

    UPDATE film SET rating = film1_rate
    WHERE title = film2;
    UPDATE film SET rating = film2_rate
    WHERE title = film1;
COMMIT;
END;
$$;

call changeRate('Academy Dinosaur', 'Ace Goldfinger')
```