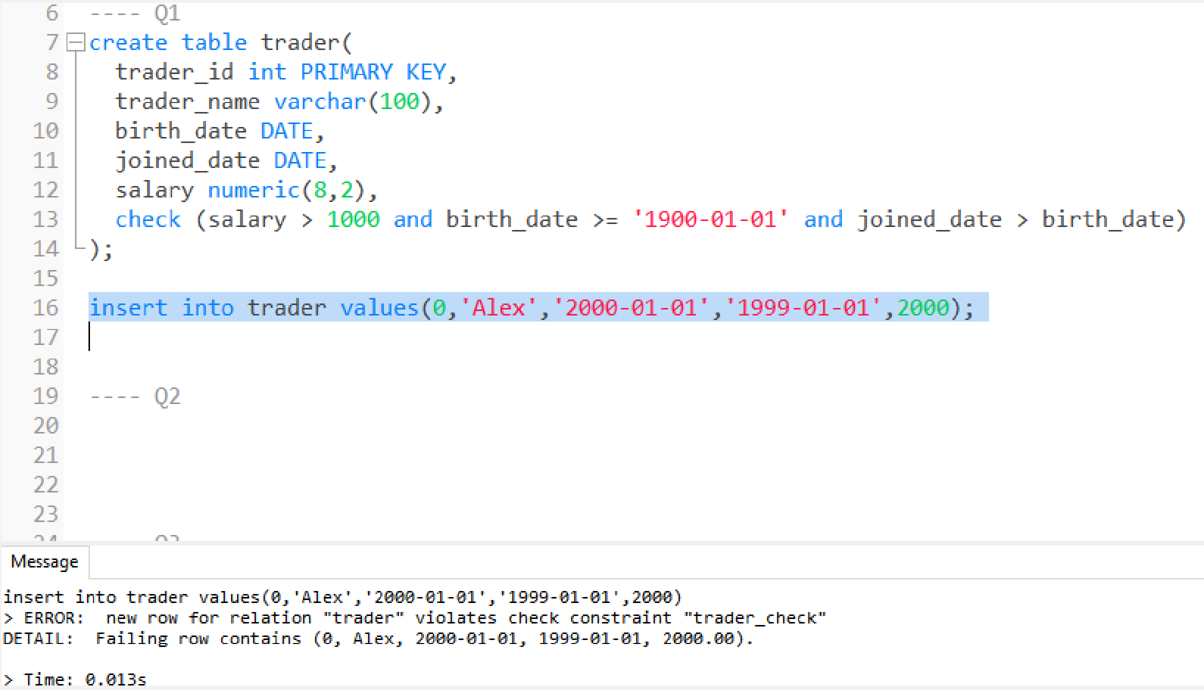
1-

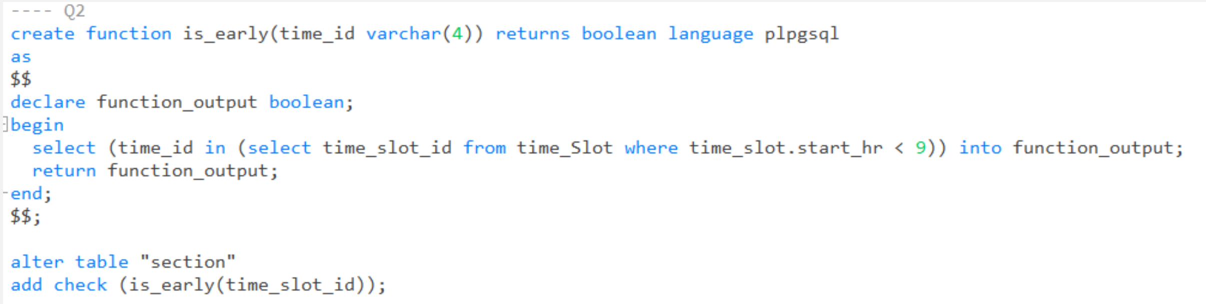


```
---- Q1
create table trader(
  trader_id int PRIMARY KEY,
  trader_name varchar(100),
  birth_date DATE,
  joined_date DATE,
  salary numeric(8,2),
  check (salary > 1000 and birth_date >= '1900-01-01' and joined_date > birth_date)
);

insert into trader values(0,'Alex','2000-01-01','1999-01-01',2000);


---- Q2
```

Message

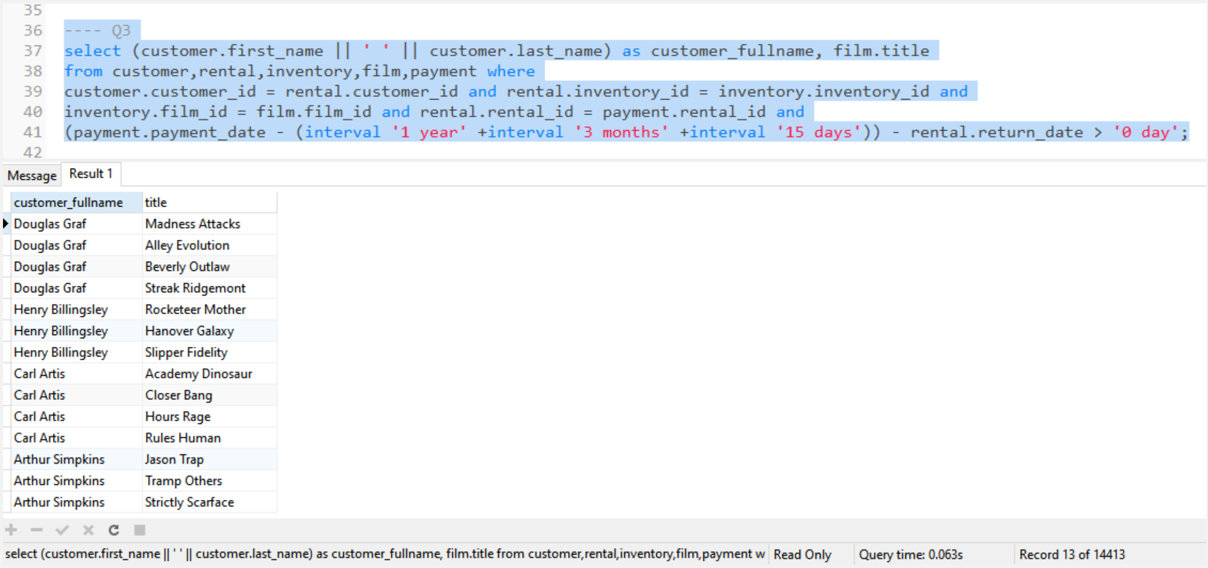
```
insert into trader values(0,'Alex','2000-01-01','1999-01-01',2000)
> ERROR:  new row for relation "trader" violates check constraint "trader_check"
DETAIL:  Failing row contains (0, Alex, 2000-01-01, 1999-01-01, 2000.00).

> Time: 0.013s
```

2-



```
---- Q2
create function is_early(time_id varchar(4)) returns boolean language plpgsql
as
$$
declare function_output boolean;
begin
  select (time_id in (select time_slot_id from time_Slot where time_slot.start_hr < 9)) into function_output;
  return function_output;
end;
$$;

alter table "section"
add check (is_early(time_slot_id));
```

3-



```
---- Q3
select (customer.first_name || ' ' || customer.last_name) as customer_fullname, film.title
from customer,rental,inventory,film,payment where
customer.customer_id = rental.customer_id and rental.inventory_id = inventory.inventory_id and
inventory.film_id = film.film_id and rental.rental_id = payment.rental_id and
(payment.payment_date - (interval '1 year' +interval '3 months' +interval '15 days')) - rental.return_date > '0 day';
```

customer_fullname	title
Douglas Graf	Madness Attacks
Douglas Graf	Alley Evolution
Douglas Graf	Beverly Outlaw
Douglas Graf	Streak Ridgemont
Henry Billingsley	Rocketeer Mother
Henry Billingsley	Hanover Galaxy
Henry Billingsley	Slipper Fidelity
Carl Artis	Academy Dinosaur
Carl Artis	Closer Bang
Carl Artis	Hours Rage
Carl Artis	Rules Human
Arthur Simpkins	Jason Trap
Arthur Simpkins	Tramp Others
Arthur Simpkins	Strictly Scarface

```
select (customer.first_name || ' ' || customer.last_name) as customer_fullname, film.title from customer,rental,inventory,film,payment w   Read Only   Query time: 0.063s   Record 13 of 14413
```

4-

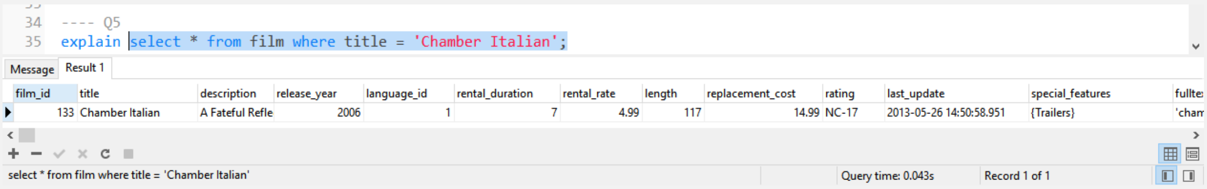
دو روش

1. ذخیره به صورت blob در دیتابیس: در این روش تمامی اطلاعات فایل به صورت باینری مستقیما در دیتابیس ذخیره میشود. این روش scalability و availability بیشتری دارد و برای تعداد فایل خیلی زیاد مناسب تر است

2. ذخیره به صورت یک رفرنس در دیتابیس: در این روش فایل در سیستم ذخیره شده و فقط یک اشاره گر به آن (مثلا اسم و ادرس فایل) در دیتابیس ذخیره میشود. این روش سرعت بیشتری دارد، به طور مثال زمانی که نیاز است ویدیو ای به صورت زنده پخش شود بهتر است از این روش استفاده شود

–5

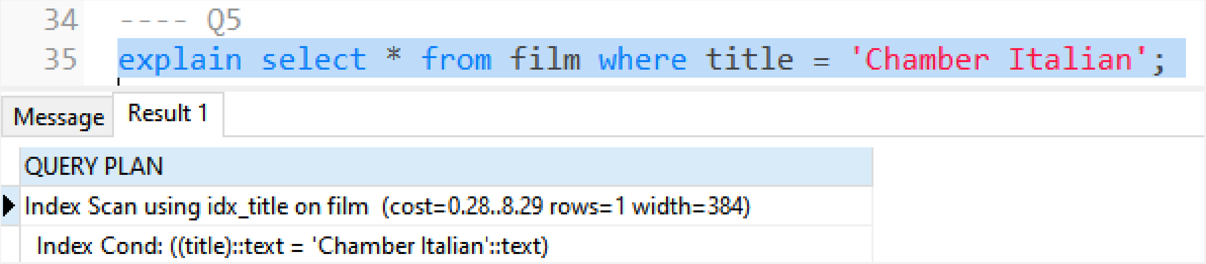
بدون ایندکس،   زمان: 0.043 ثانیه



```sql
---- Q5
explain select * from film where title = 'Chamber Italian';
```

film_id	title	description	release_year	language_id	rental_duration	rental_rate	length	replacement_cost	rating	last_update	special_features	fullte:
133	Chamber Italian	A Fateful Refle	2006	1	7	4.99	117	14.99	NC-17	2013-05-26 14:50:58.951	{Trailers}	'cham

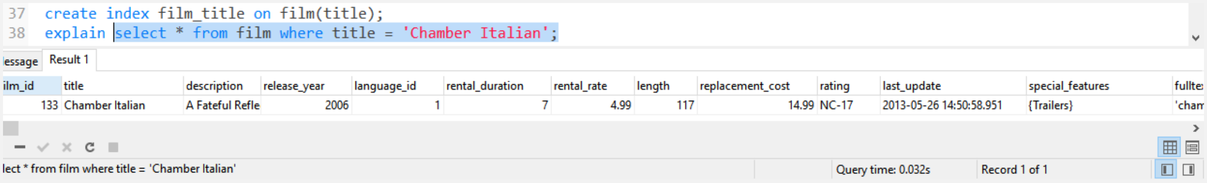
select * from film where title = 'Chamber Italian'   Query time: 0.043s   Record 1 of 1



```sql
---- Q5
explain select * from film where title = 'Chamber Italian';
```

```
QUERY PLAN
Index Scan using idx_title on film  (cost=0.28..8.29 rows=1 width=384)
  Index Cond: ((title)::text = 'Chamber Italian'::text)
```

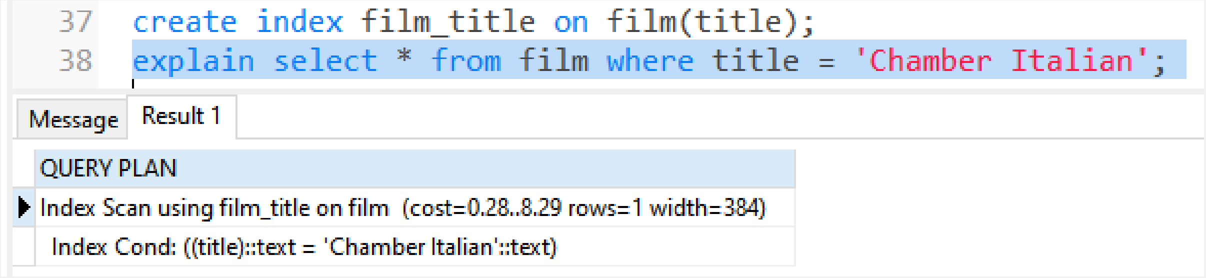
با ایندکس،   زمان: 0.032 ثانیه



```sql
create index film_title on film(title);
explain select * from film where title = 'Chamber Italian';
```

ilm_id	title	description	release_year	language_id	rental_duration	rental_rate	length	replacement_cost	rating	last_update	special_features	fullte:
133	Chamber Italian	A Fateful Refle	2006	1	7	4.99	117	14.99	NC-17	2013-05-26 14:50:58.951	{Trailers}	'cham

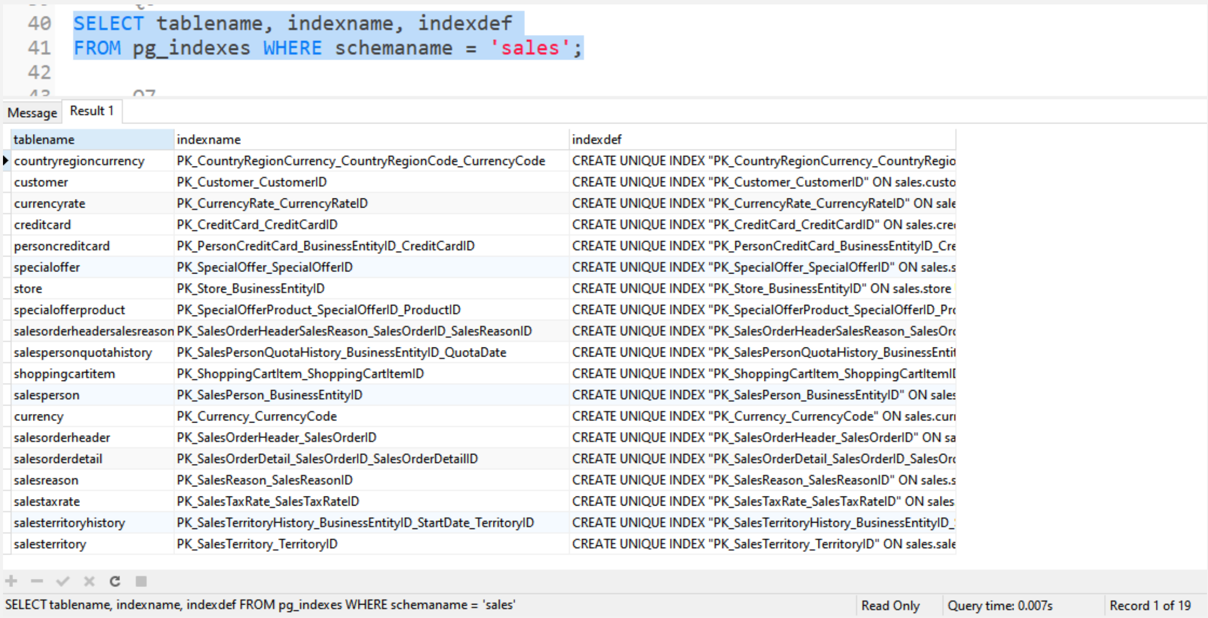
lect * from film where title = 'Chamber Italian'   Query time: 0.032s   Record 1 of 1



```sql
create index film_title on film(title);
explain select * from film where title = 'Chamber Italian';
```

```
QUERY PLAN
Index Scan using film_title on film  (cost=0.28..8.29 rows=1 width=384)
  Index Cond: ((title)::text = 'Chamber Italian'::text)
```

–6



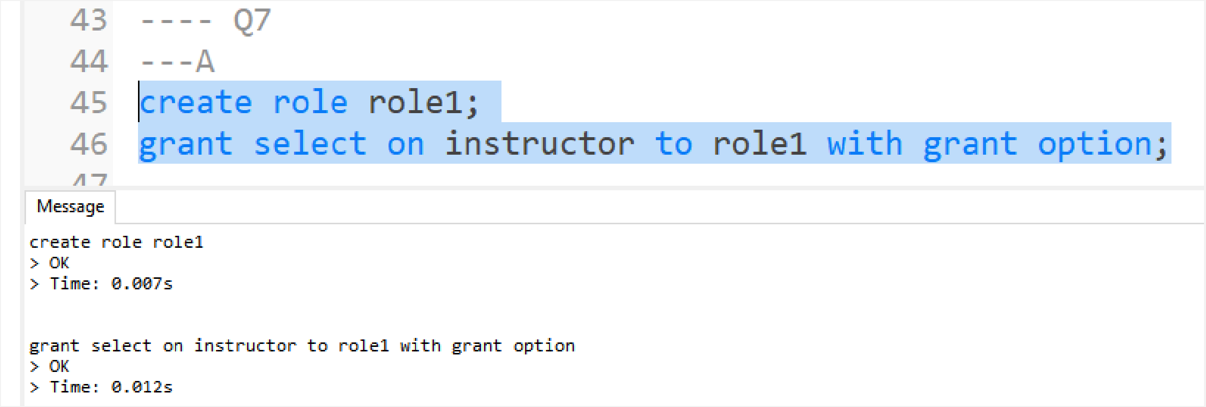
```sql
SELECT tablename, indexname, indexdef
FROM pg_indexes WHERE schemaname = 'sales';
```

tablename	indexname	indexdef
countryregioncurrency	PK_CountryRegionCurrency_CountryRegionCode_CurrencyCode	CREATE UNIQUE INDEX "PK_CountryRegionCurrency_CountryRegio
customer	PK_Customer_CustomerID	CREATE UNIQUE INDEX "PK_Customer_CustomerID" ON sales.custo
currencyrate	PK_CurrencyRate_CurrencyRateID	CREATE UNIQUE INDEX "PK_CurrencyRate_CurrencyRateID" ON sale
creditcard	PK_CreditCard_CreditCardID	CREATE UNIQUE INDEX "PK_CreditCard_CreditCardID" ON sales.cre
personcreditcard	PK_PersonCreditCard_BusinessEntityID_CreditCardID	CREATE UNIQUE INDEX "PK_PersonCreditCard_BusinessEntityID_Cre
specialoffer	PK_SpecialOffer_SpecialOfferID	CREATE UNIQUE INDEX "PK_SpecialOffer_SpecialOfferID" ON sales.s
store	PK_Store_BusinessEntityID	CREATE UNIQUE INDEX "PK_Store_BusinessEntityID" ON sales.store
specialofferproduct	PK_SpecialOfferProduct_SpecialOfferID_ProductID	CREATE UNIQUE INDEX "PK_SpecialOfferProduct_SpecialOfferID_Pro
salesorderheadersalesreason	PK_SalesOrderHeaderSalesReason_SalesOrderID_SalesReasonID	CREATE UNIQUE INDEX "PK_SalesOrderHeaderSalesReason_SalesOrc
salespersonquotahistory	PK_SalesPersonQuotaHistory_BusinessEntityID_QuotaDate	CREATE UNIQUE INDEX "PK_SalesPersonQuotaHistory_BusinessEntit
shoppingcartitem	PK_ShoppingCartItem_ShoppingCartItemID	CREATE UNIQUE INDEX "PK_ShoppingCartItem_ShoppingCartItemI
salesperson	PK_SalesPerson_BusinessEntityID	CREATE UNIQUE INDEX "PK_SalesPerson_BusinessEntityID" ON sales
currency	PK_Currency_CurrencyCode	CREATE UNIQUE INDEX "PK_Currency_CurrencyCode" ON sales.curr
salesorderheader	PK_SalesOrderHeader_SalesOrderID	CREATE UNIQUE INDEX "PK_SalesOrderHeader_SalesOrderID" ON sa
salesorderdetail	PK_SalesOrderDetail_SalesOrderID_SalesOrderDetailID	CREATE UNIQUE INDEX "PK_SalesOrderDetail_SalesOrderID_SalesOrc
salesreason	PK_SalesReason_SalesReasonID	CREATE UNIQUE INDEX "PK_SalesReason_SalesReasonID" ON sales.s
salestaxrate	PK_SalesTaxRate_SalesTaxRateID	CREATE UNIQUE INDEX "PK_SalesTaxRate_SalesTaxRateID" ON sales
salesterritoryhistory	PK_SalesTerritoryHistory_BusinessEntityID_StartDate_TerritoryID	CREATE UNIQUE INDEX "PK_SalesTerritoryHistory_BusinessEntityID_
salesterritory	PK_SalesTerritory_TerritoryID	CREATE UNIQUE INDEX "PK_SalesTerritory_TerritoryID" ON sales.sale

```
SELECT tablename, indexname, indexdef FROM pg_indexes WHERE schemaname = 'sales'
```

–7

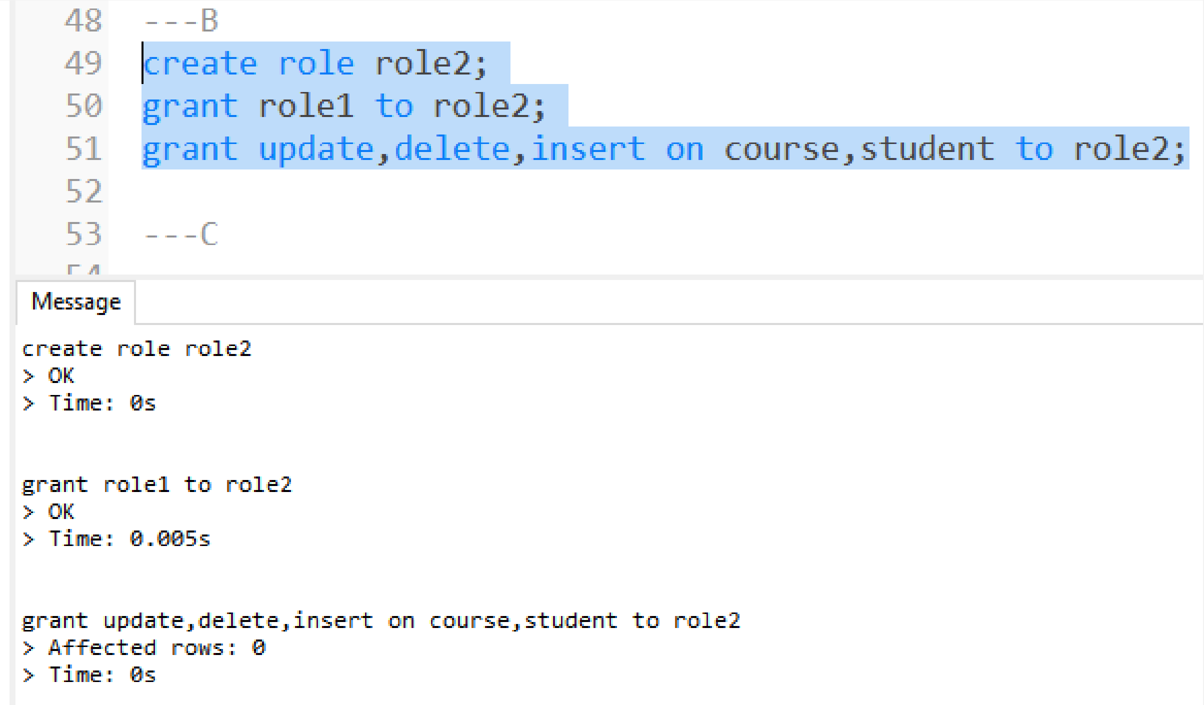
–A



```sql
---- Q7
---A
create role role1;
grant select on instructor to role1 with grant option;
```

```
create role role1
> OK
> Time: 0.007s


grant select on instructor to role1 with grant option
> OK
> Time: 0.012s
```

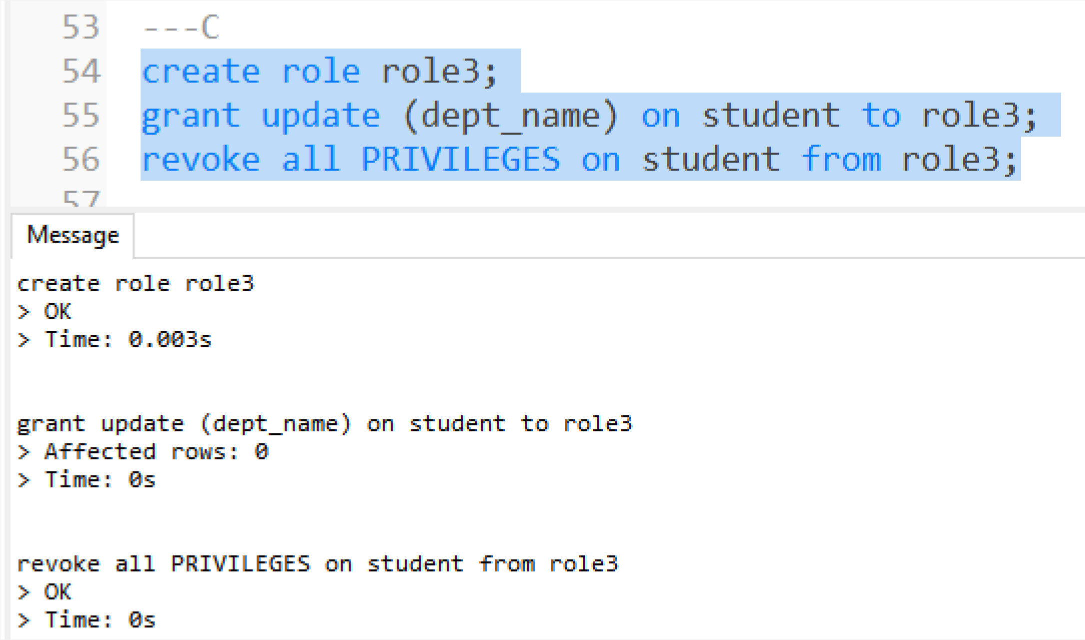
–B



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

---C
```

```
create role role2
> OK
> Time: 0s


grant role1 to role2
> OK
> Time: 0.005s


grant update,delete,insert on course,student to role2
> Affected rows: 0
> Time: 0s
```

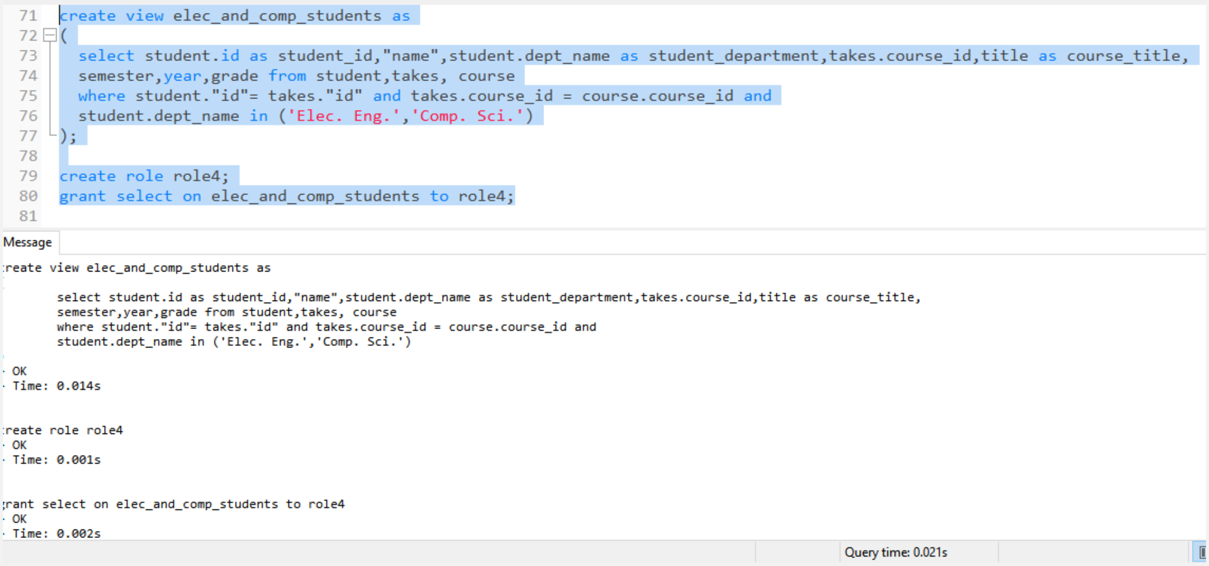
–C



```
---C
create role role3;
grant update (dept_name) on student to role3;
revoke all PRIVILEGES on student from role3;
```

```
create role role3
> OK
> Time: 0.003s


grant update (dept_name) on student to role3
> Affected rows: 0
> Time: 0s


revoke all PRIVILEGES on student from role3
> OK
> Time: 0s
```

–D

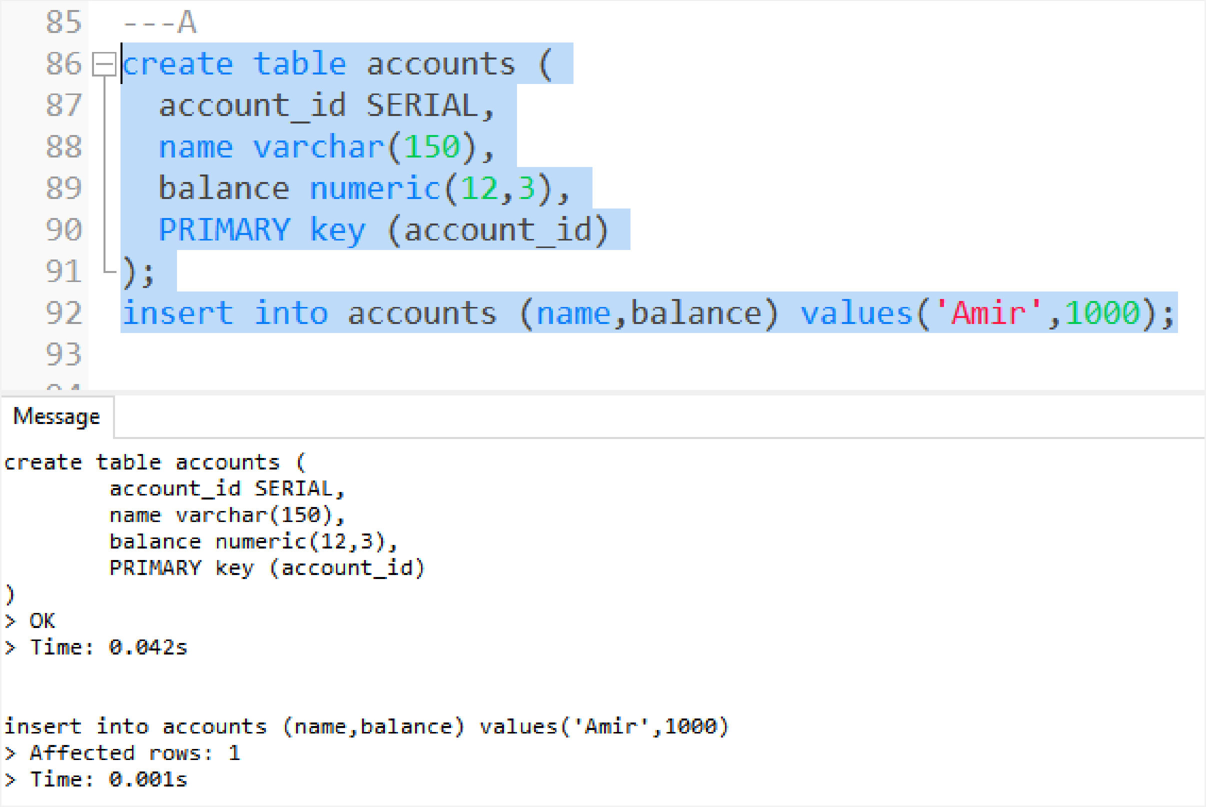


```
create view elec_and_comp_students as
(
  select student.id as student_id,"name",student.dept_name as student_department,takes.course_id,title as course_title,
  semester,year,grade from student,takes, course
  where student."id"= takes."id" and takes.course_id = course.course_id and
  student.dept_name in ('Elec. Eng.','Comp. Sci.')
);

create role role4;
grant select on elec_and_comp_students to role4;
```

```
reate view elec_and_comp_students as

        select student.id as student_id,"name",student.dept_name as student_department,takes.course_id,title as course_title,
        semester,year,grade from student,takes, course
        where student."id"= takes."id" and takes.course_id = course.course_id and
        student.dept_name in ('Elec. Eng.','Comp. Sci.')

 OK
 Time: 0.014s


reate role role4
 OK
 Time: 0.001s


rant select on elec_and_comp_students to role4
 OK
 Time: 0.002s
```

8-

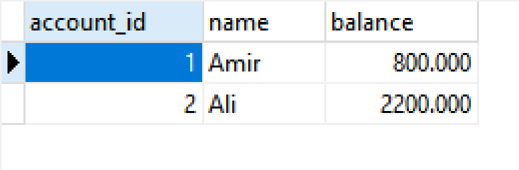
A-



```
---A
create table accounts (
  account_id SERIAL,
  name varchar(150),
  balance numeric(12,3),
  PRIMARY key (account_id)
);
insert into accounts (name,balance) values('Amir',1000);
```

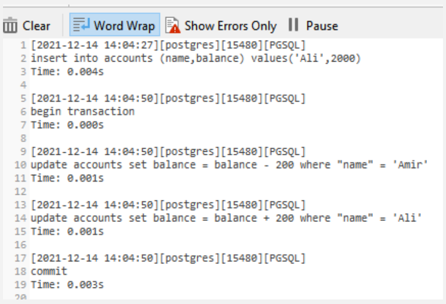
```
create table accounts (
        account_id SERIAL,
        name varchar(150),
        balance numeric(12,3),
        PRIMARY key (account_id)
)
> OK
> Time: 0.042s


insert into accounts (name,balance) values('Amir',1000)
> Affected rows: 1
> Time: 0.001s
```

B-



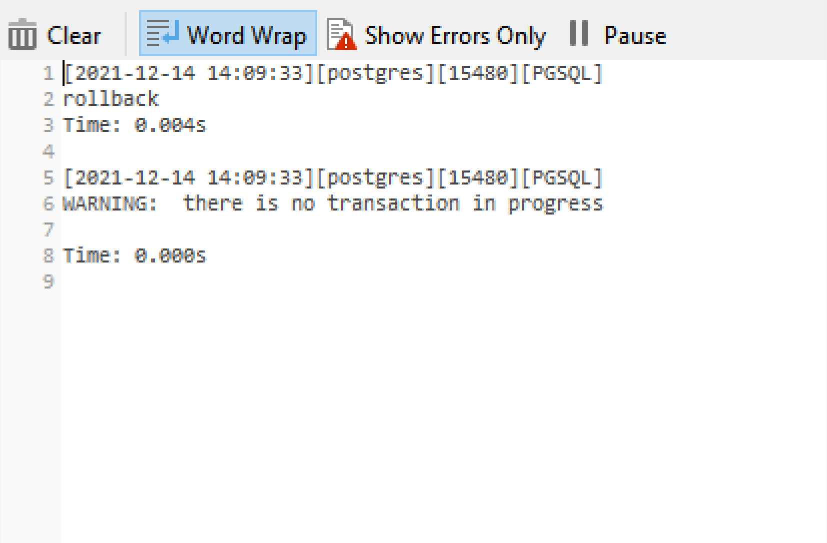
account_id	name	balance
1	Amir	800.000
2	Ali	2200.000

Log after commit:



```
[2021-12-14 14:04:27][postgres][15480][PGSQL]
insert into accounts (name,balance) values('Ali',2000)
Time: 0.004s

[2021-12-14 14:04:50][postgres][15480][PGSQL]
begin transaction
Time: 0.000s

[2021-12-14 14:04:50][postgres][15480][PGSQL]
update accounts set balance = balance - 200 where "name" = 'Amir'
Time: 0.001s

[2021-12-14 14:04:50][postgres][15480][PGSQL]
update accounts set balance = balance + 200 where "name" = 'Ali'
Time: 0.001s

[2021-12-14 14:04:50][postgres][15480][PGSQL]
commit
Time: 0.003s
```

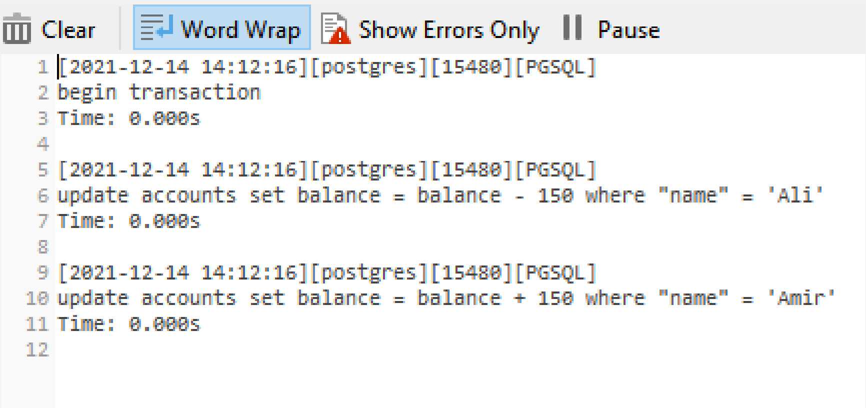
Log after rollback:



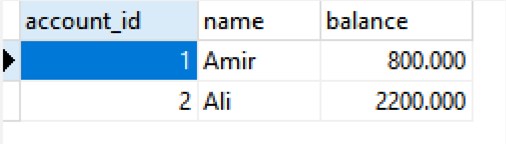
```
[2021-12-14 14:09:33][postgres][15480][PGSQL]
rollback
Time: 0.004s

[2021-12-14 14:09:33][postgres][15480][PGSQL]
WARNING:  there is no transaction in progress

Time: 0.000s
```

C-

After changes:

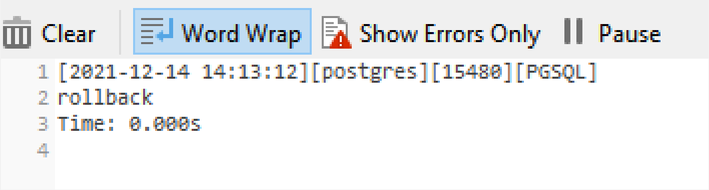


```
[2021-12-14 14:12:16][postgres][15480][PGSQL]
begin transaction
Time: 0.000s

[2021-12-14 14:12:16][postgres][15480][PGSQL]
update accounts set balance = balance - 150 where "name" = 'Ali'
Time: 0.000s

[2021-12-14 14:12:16][postgres][15480][PGSQL]
update accounts set balance = balance + 150 where "name" = 'Amir'
Time: 0.000s
```

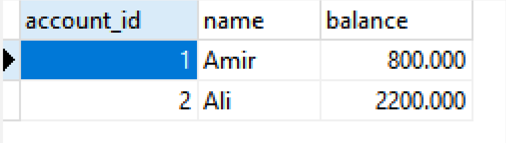


account_id	name	balance
1	Amir	800.000
2	Ali	2200.000

After rollback:

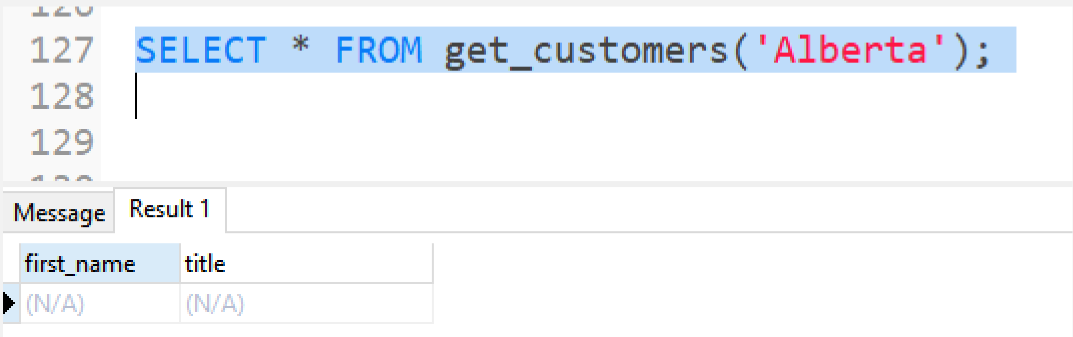


```
[2021-12-14 14:13:12][postgres][15480][PGSQL]
rollback
Time: 0.000s
```



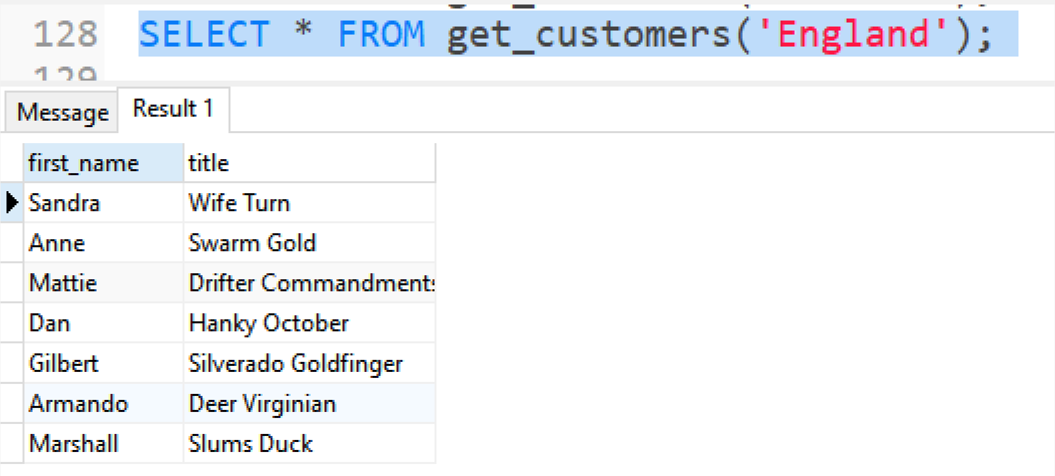
account_id	name	balance
1	Amir	800.000
2	Ali	2200.000

9-



```sql
SELECT * FROM get_customers('Alberta');
```

first_name	title
(N/A)	(N/A)

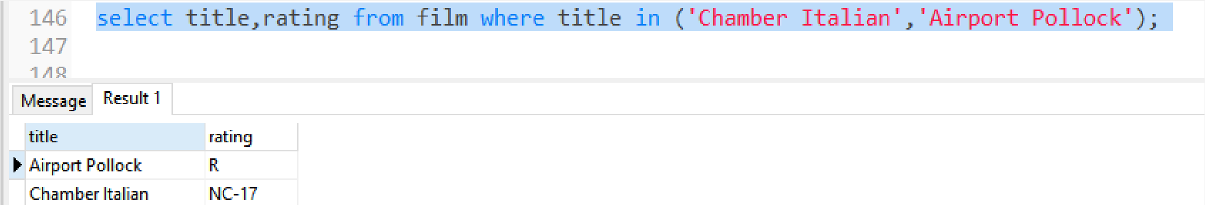


```sql
SELECT * FROM get_customers('England');
```

first_name	title
Sandra	Wife Turn
Anne	Swarm Gold
Mattie	Drifter Commandment:
Dan	Hanky October
Gilbert	Silverado Goldfinger
Armando	Deer Virginian
Marshall	Slums Duck

10-

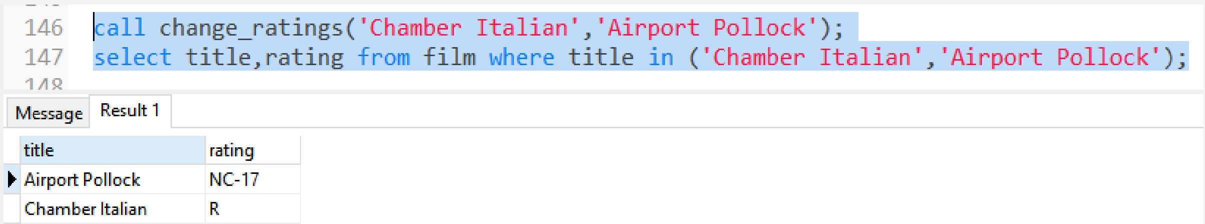
Before:



```sql
select title,rating from film where title in ('Chamber Italian','Airport Pollock');
```

title	rating
Airport Pollock	R
Chamber Italian	NC-17

After:



```sql
call change_ratings('Chamber Italian','Airport Pollock');
select title,rating from film where title in ('Chamber Italian','Airport Pollock');
```

title	rating
Airport Pollock	NC-17
Chamber Italian	R