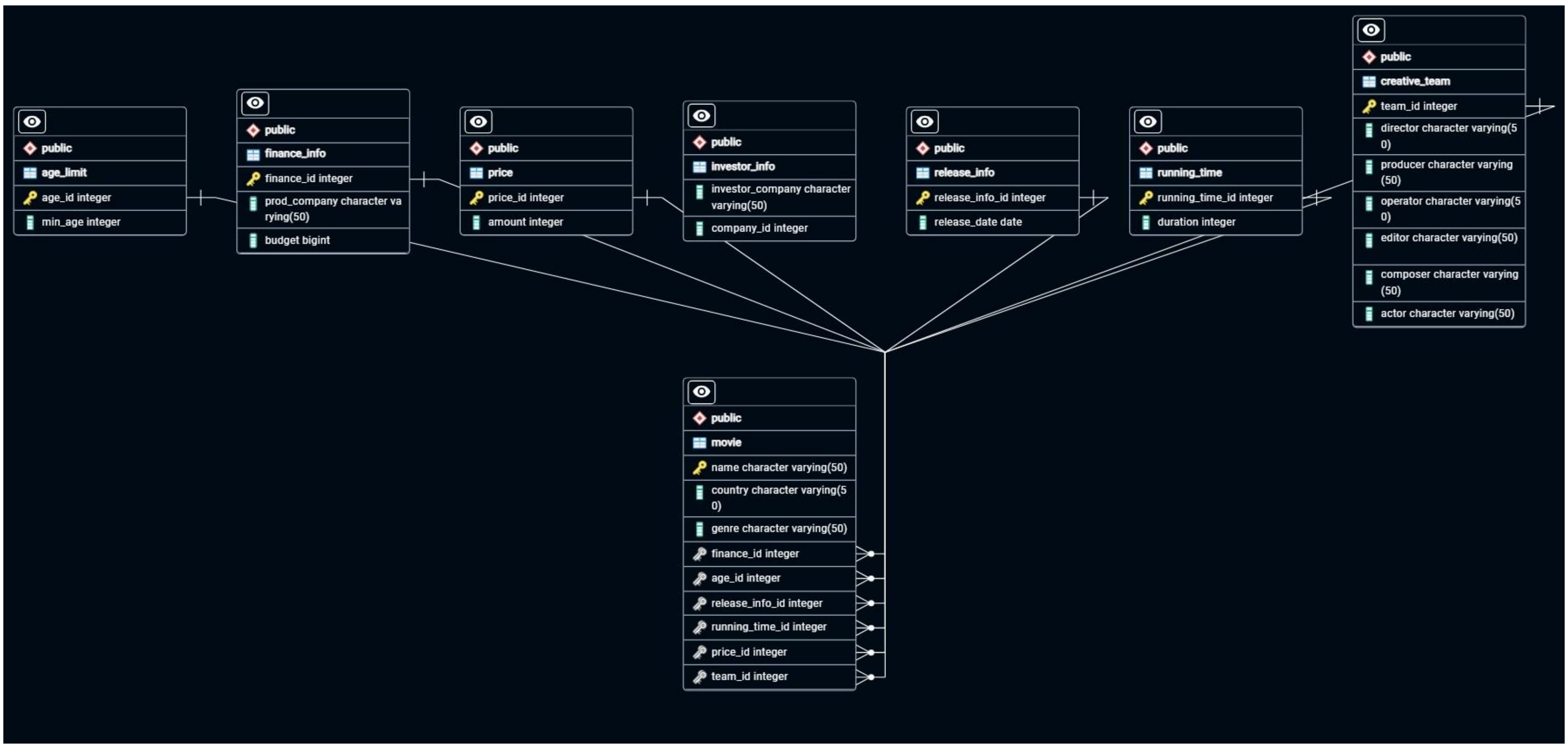
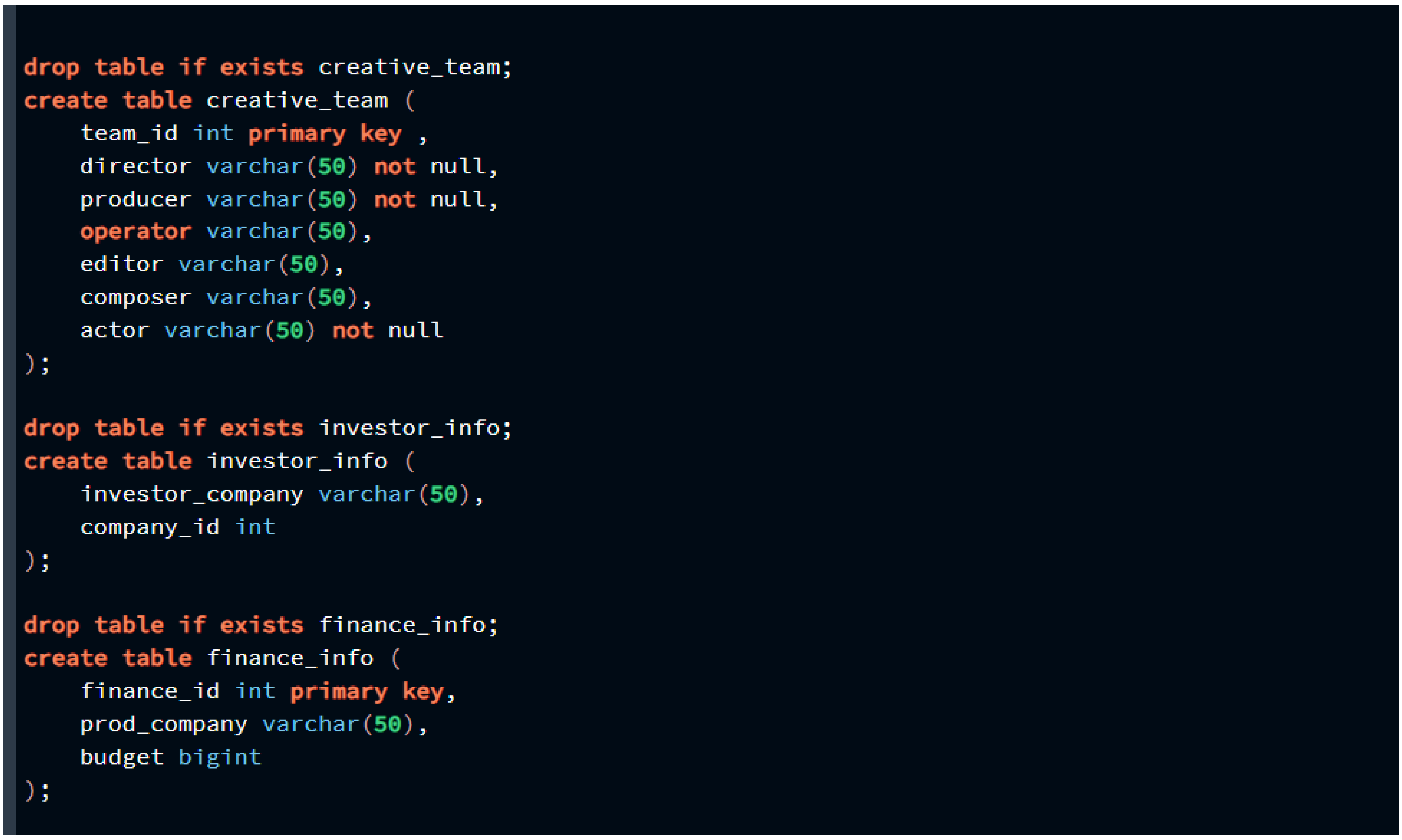
1.



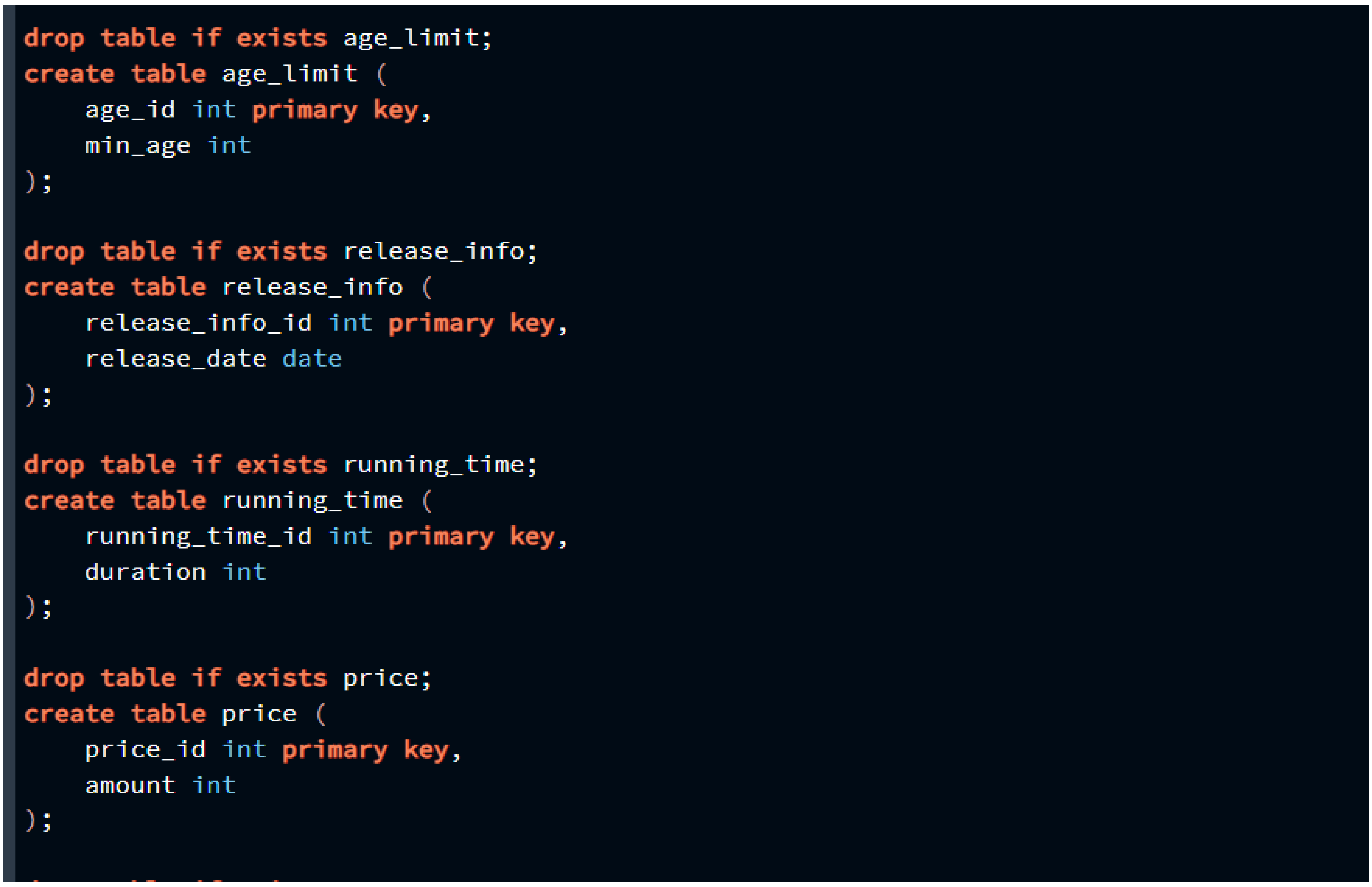
2.

Added team_id to get relate with movie table

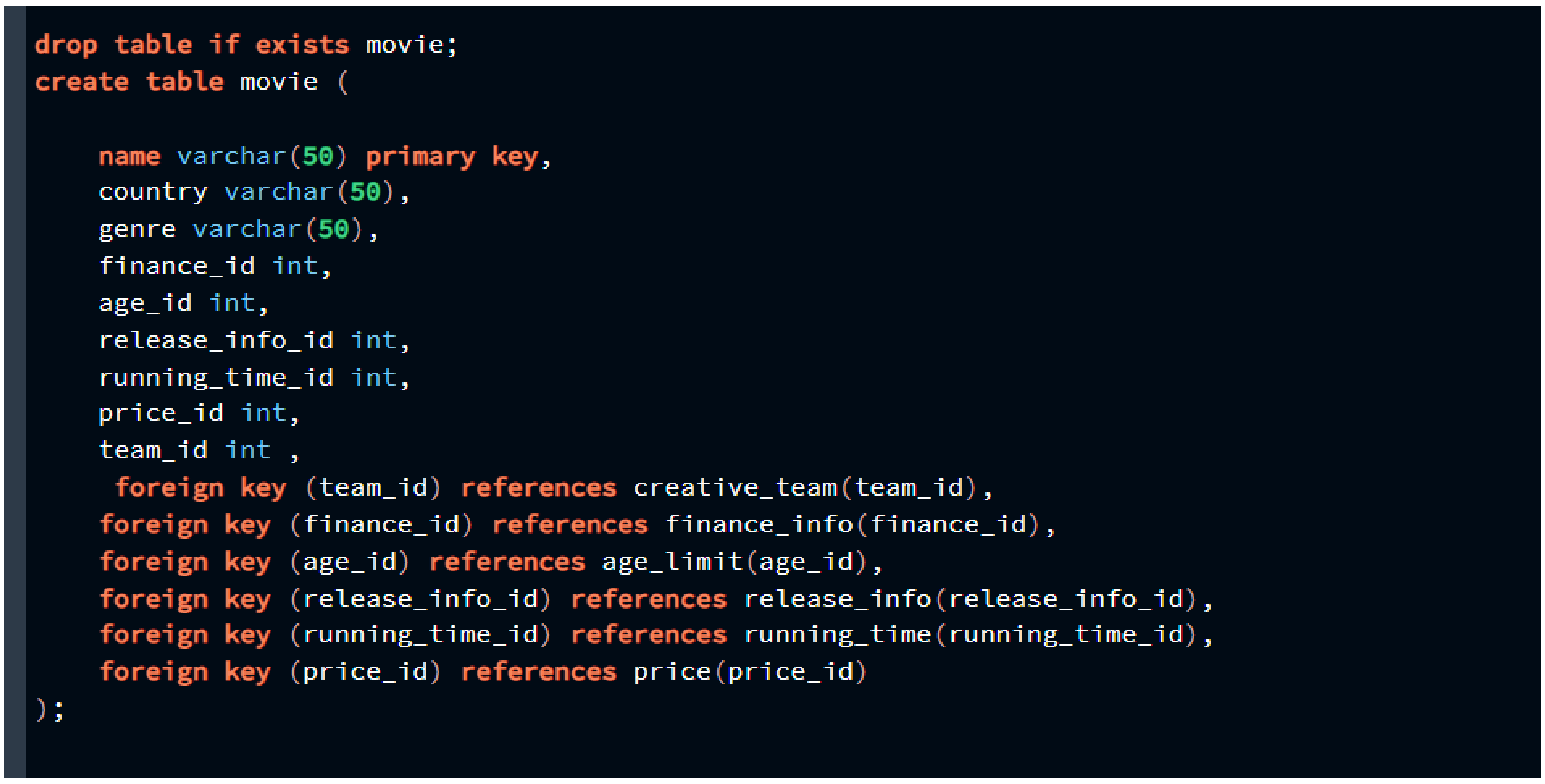
.



```
drop table if exists creative_team;
create table creative_team (
    team_id int primary key ,
    director varchar(50) not null,
    producer varchar(50) not null,
    operator varchar(50),
    editor varchar(50),
    composer varchar(50),
    actor varchar(50) not null
);

drop table if exists investor_info;
create table investor_info (
    investor_company varchar(50),
    company_id int
);

drop table if exists finance_info;
create table finance_info (
    finance_id int primary key,
    prod_company varchar(50),
    budget bigint
);
```

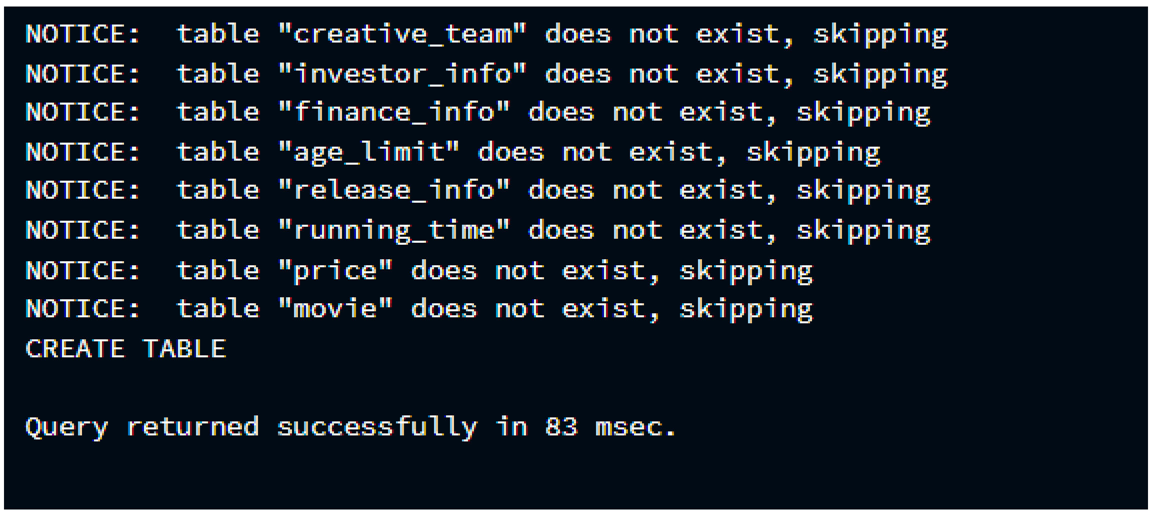


```sql
drop table if exists age_limit;
create table age_limit (
    age_id int primary key,
    min_age int
);

drop table if exists release_info;
create table release_info (
    release_info_id int primary key,
    release_date date
);

drop table if exists running_time;
create table running_time (
    running_time_id int primary key,
    duration int
);

drop table if exists price;
create table price (
    price_id int primary key,
    amount int
);
```



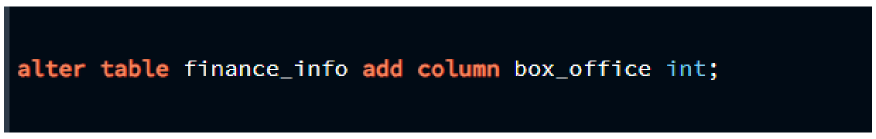
```sql
drop table if exists movie;
create table movie (

    name varchar(50) primary key,
    country varchar(50),
    genre varchar(50),
    finance_id int,
    age_id int,
    release_info_id int,
    running_time_id int,
    price_id int,
    team_id int ,
     foreign key (team_id) references creative_team(team_id),
    foreign key (finance_id) references finance_info(finance_id),
    foreign key (age_id) references age_limit(age_id),
    foreign key (release_info_id) references release_info(release_info_id),
    foreign key (running_time_id) references running_time(running_time_id),
    foreign key (price_id) references price(price_id)
);
```

OUTPUT



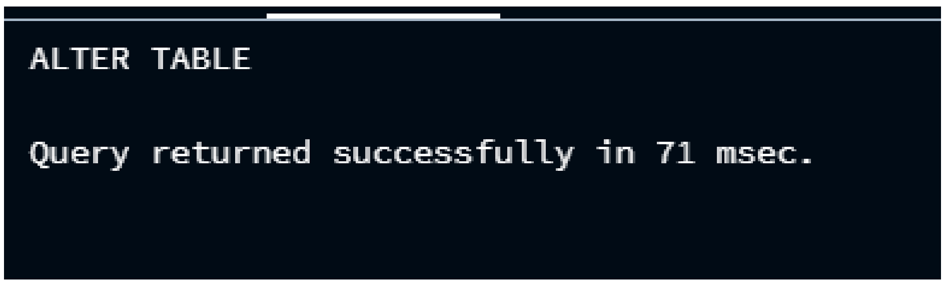
```
NOTICE:  table "creative_team" does not exist, skipping
NOTICE:  table "investor_info" does not exist, skipping
NOTICE:  table "finance_info" does not exist, skipping
NOTICE:  table "age_limit" does not exist, skipping
NOTICE:  table "release_info" does not exist, skipping
NOTICE:  table "running_time" does not exist, skipping
NOTICE:  table "price" does not exist, skipping
NOTICE:  table "movie" does not exist, skipping
CREATE TABLE

Query returned successfully in 83 msec.
```

3.

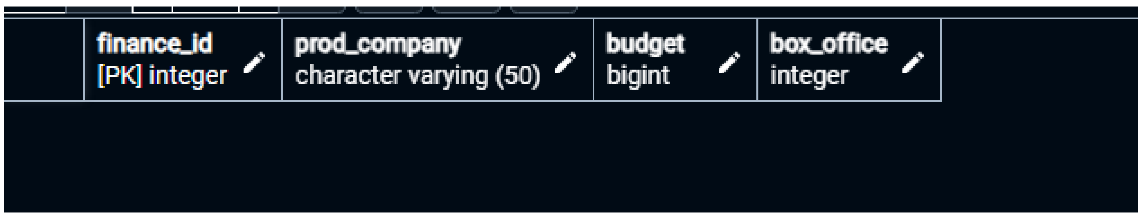


```
alter table finance_info add column box_office int;
```

OUTPUT

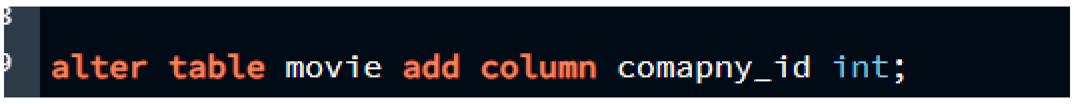


```
ALTER TABLE

Query returned successfully in 71 msec.
```



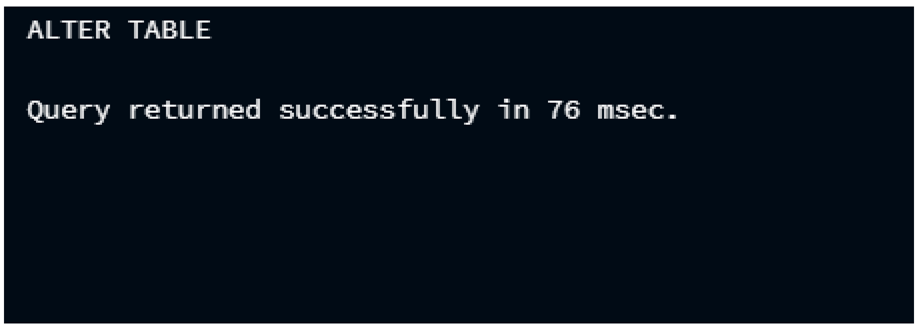
finance_id	prod_company	budget	box_office
[PK] integer	character varying (50)	bigint	integer

4.

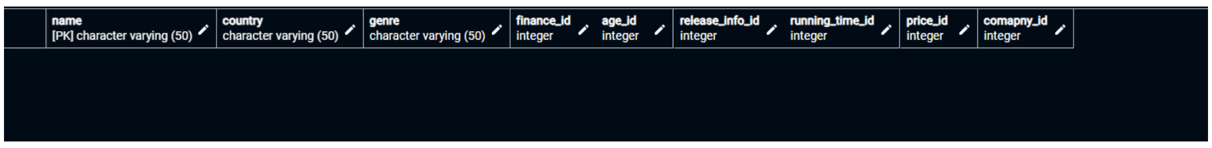


```
alter table movie add column comapny_id int;
```

OUTPUT

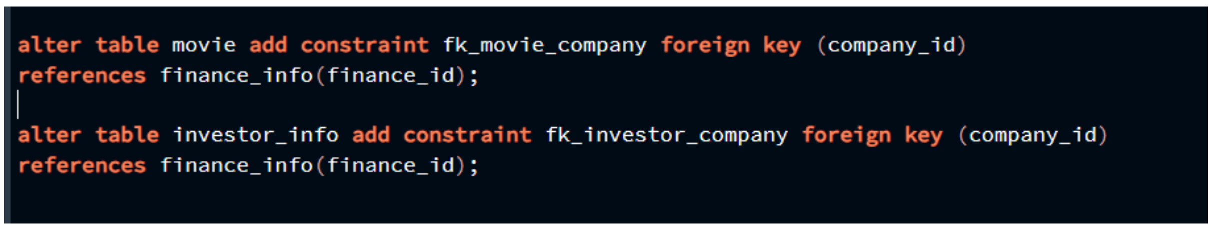


```
ALTER TABLE

Query returned successfully in 76 msec.
```



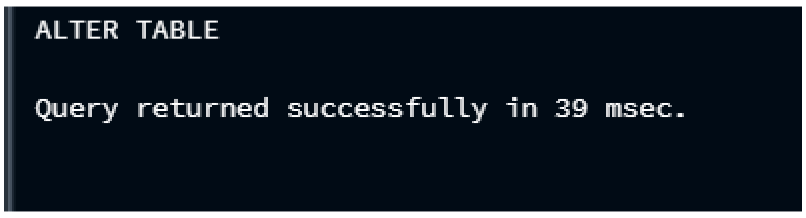
name	country	genre	finance_id	age_id	release_info_id	running_time_id	price_id	comapny_id
[PK] character varying (50)	character varying (50)	character varying (50)	integer	integer	integer	integer	integer	integer

5.



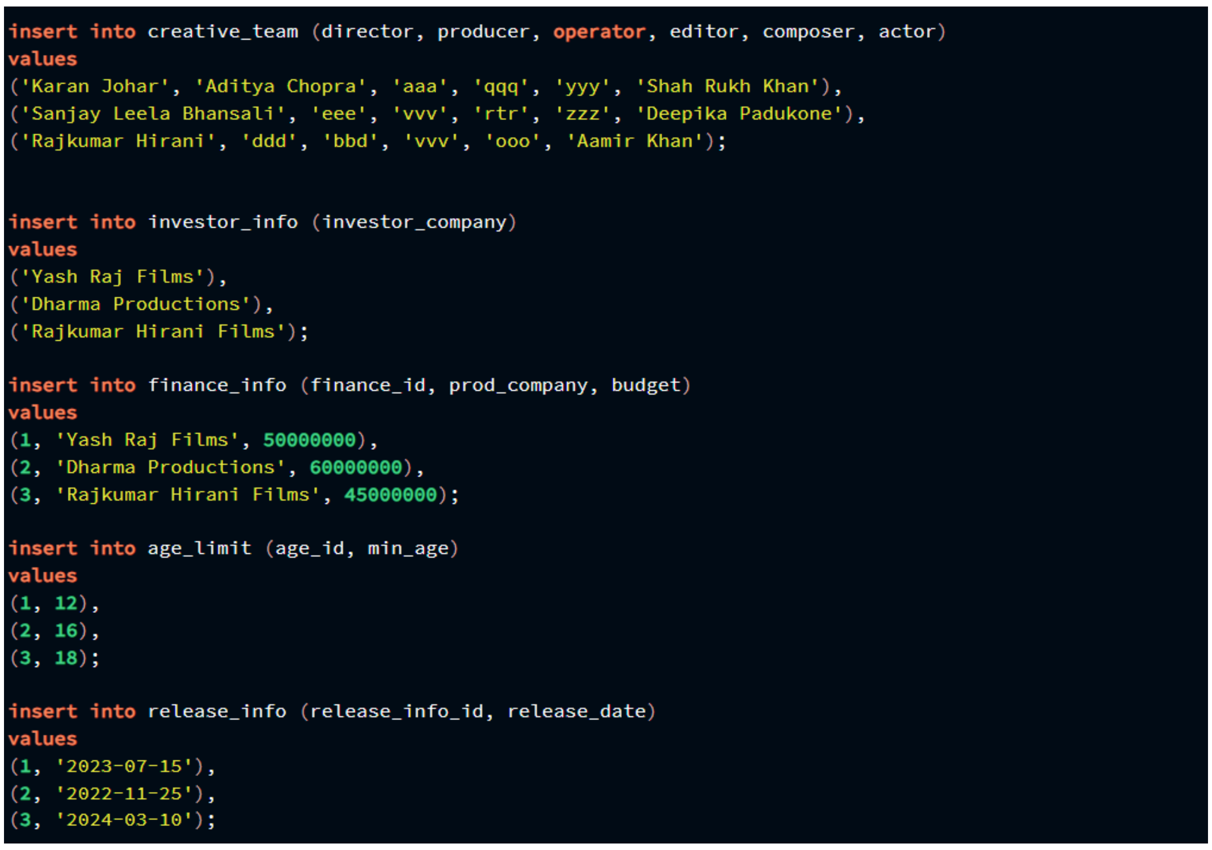
```
alter table movie add constraint fk_movie_company foreign key (company_id)
references finance_info(finance_id);

alter table investor_info add constraint fk_investor_company foreign key (company_id)
references finance_info(finance_id);
```

OUTPUT

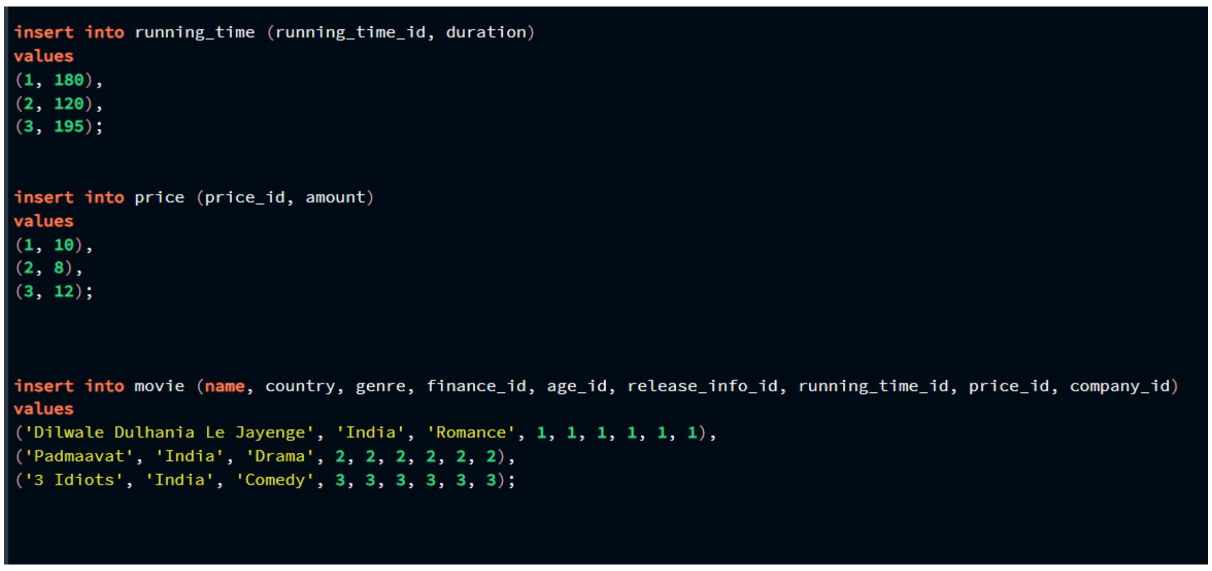


```
ALTER TABLE

Query returned successfully in 39 msec.
```

6.

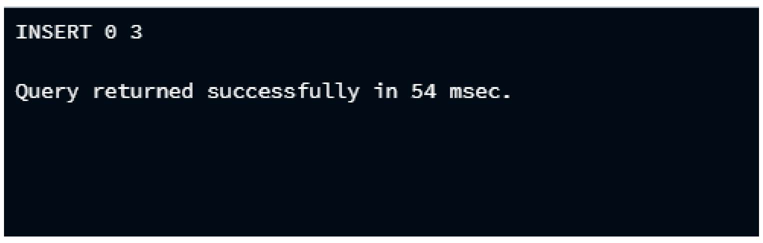


```
insert into creative_team (director, producer, operator, editor, composer, actor)
values
('Karan Johar', 'Aditya Chopra', 'aaa', 'qqq', 'yyy', 'Shah Rukh Khan'),
('Sanjay Leela Bhansali', 'eee', 'vvv', 'rtr', 'zzz', 'Deepika Padukone'),
('Rajkumar Hirani', 'ddd', 'bbd', 'vvv', 'ooo', 'Aamir Khan');


insert into investor_info (investor_company)
values
('Yash Raj Films'),
('Dharma Productions'),
('Rajkumar Hirani Films');

insert into finance_info (finance_id, prod_company, budget)
values
(1, 'Yash Raj Films', 50000000),
(2, 'Dharma Productions', 60000000),
(3, 'Rajkumar Hirani Films', 45000000);

insert into age_limit (age_id, min_age)
values
(1, 12),
(2, 16),
(3, 18);

insert into release_info (release_info_id, release_date)
values
(1, '2023-07-15'),
(2, '2022-11-25'),
(3, '2024-03-10');
```



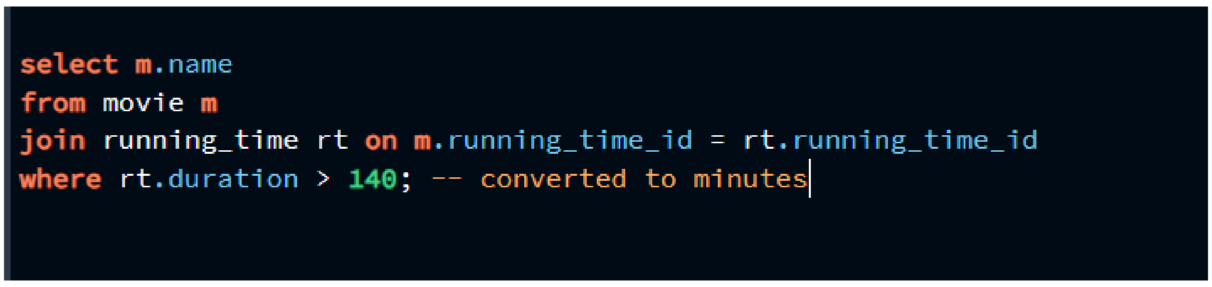
```
insert into running_time (running_time_id, duration)
values
(1, 180),
(2, 120),
(3, 195);


insert into price (price_id, amount)
values
(1, 10),
(2, 8),
(3, 12);



insert into movie (name, country, genre, finance_id, age_id, release_info_id, running_time_id, price_id, company_id)
values
('Dilwale Dulhania Le Jayenge', 'India', 'Romance', 1, 1, 1, 1, 1, 1),
('Padmaavat', 'India', 'Drama', 2, 2, 2, 2, 2, 2),
('3 Idiots', 'India', 'Comedy', 3, 3, 3, 3, 3, 3);
```

OUTPUT



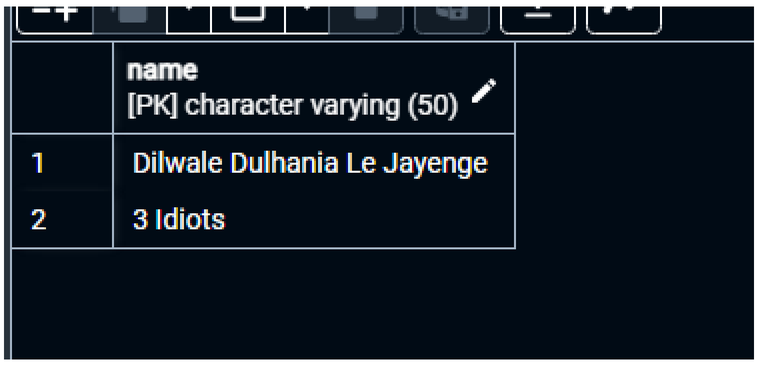
```
INSERT 0 3

Query returned successfully in 54 msec.
```

7.



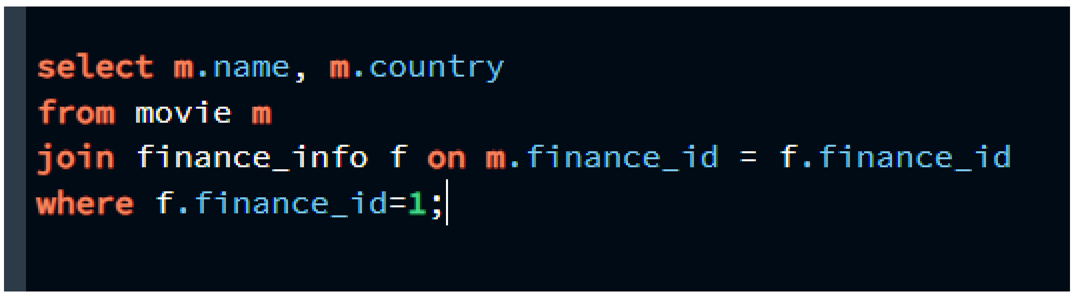
```
select m.name
from movie m
join running_time rt on m.running_time_id = rt.running_time_id
where rt.duration > 140; -- converted to minutes
```

OUTPUT



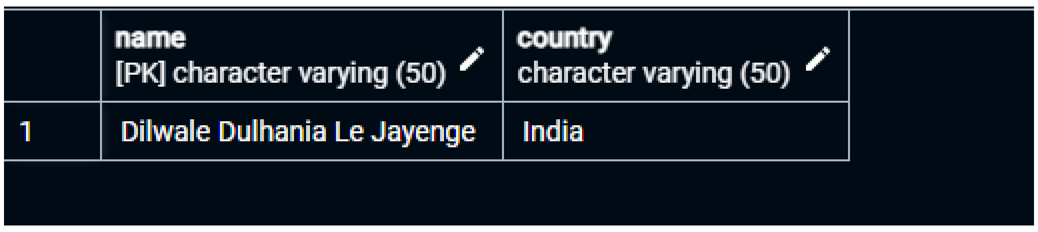
	name [PK] character varying (50)
1	Dilwale Dulhania Le Jayenge
2	3 Idiots

8.



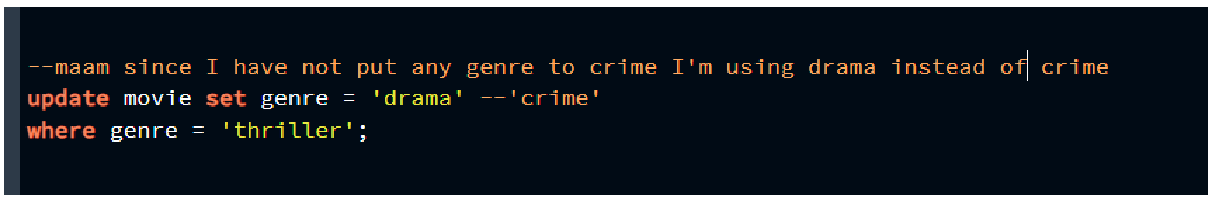
```
select m.name, m.country
from movie m
join finance_info f on m.finance_id = f.finance_id
where f.finance_id=1;
```

OUTPUT



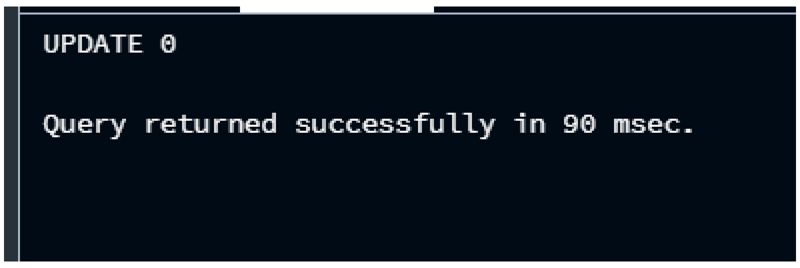
	name [PK] character varying (50)	country character varying (50)
1	Dilwale Dulhania Le Jayenge	India

9.



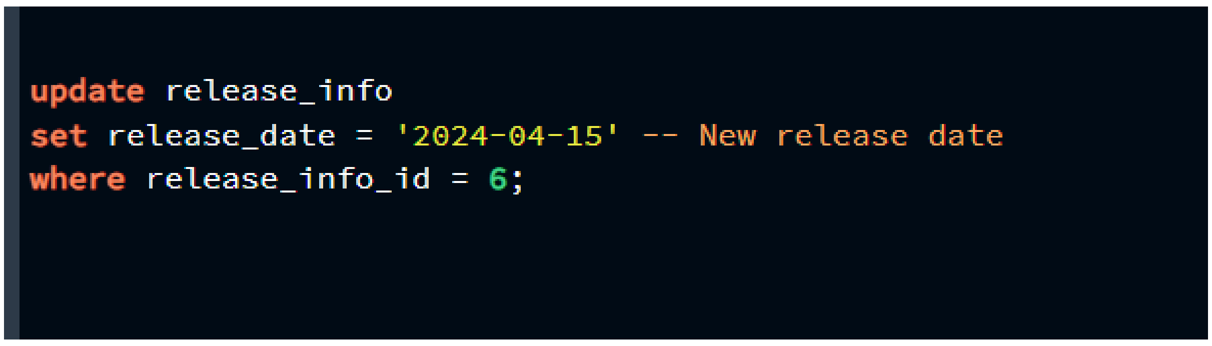
```
--maam since I have not put any genre to crime I'm using drama instead of crime
update movie set genre = 'drama' --'crime'
where genre = 'thriller';
```

OUTPUT



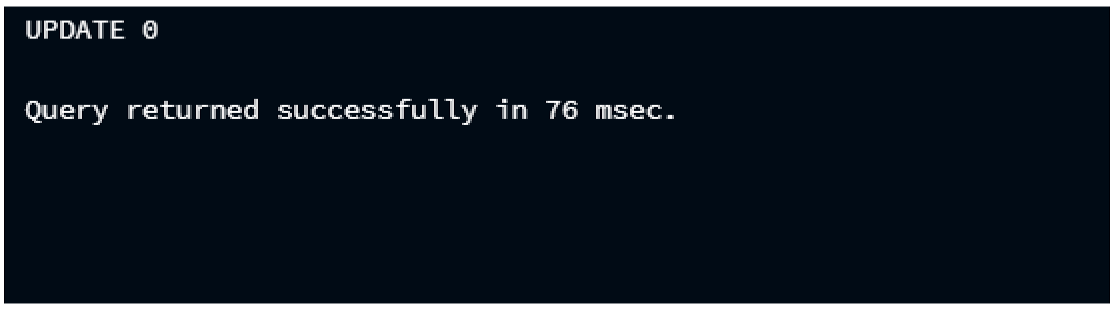
```
UPDATE 0

Query returned successfully in 90 msec.
```

10.



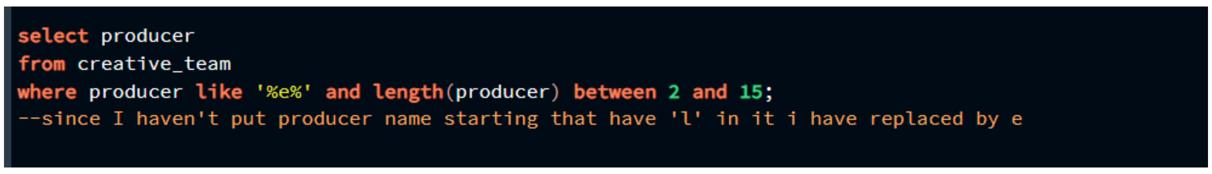
```
update release_info
set release_date = '2024-04-15' -- New release date
where release_info_id = 6;
```

OUTPUT



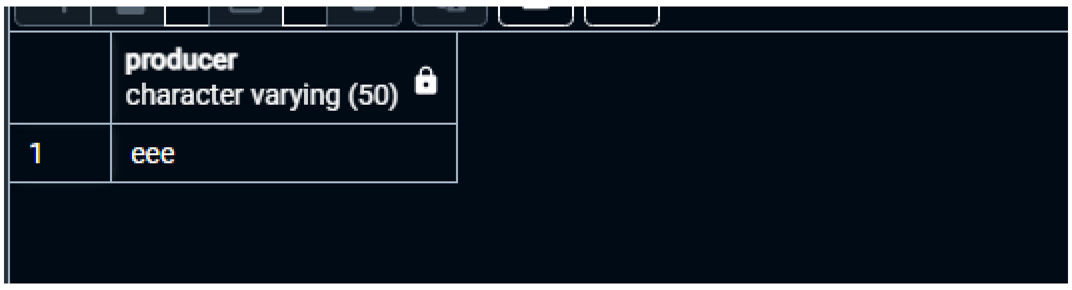
```
UPDATE 0

Query returned successfully in 76 msec.
```

11.



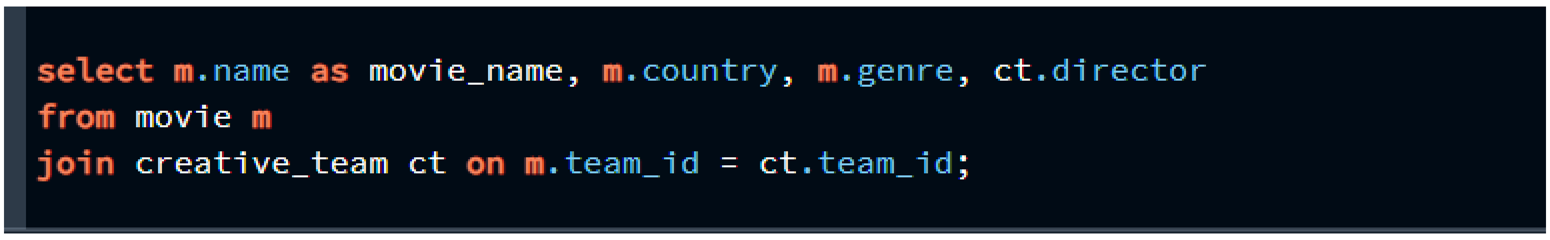
```
select producer
from creative_team
where producer like '%e%' and length(producer) between 2 and 15;
--since I haven't put producer name starting that have 'l' in it i have replaced by e
```

OUTPUT



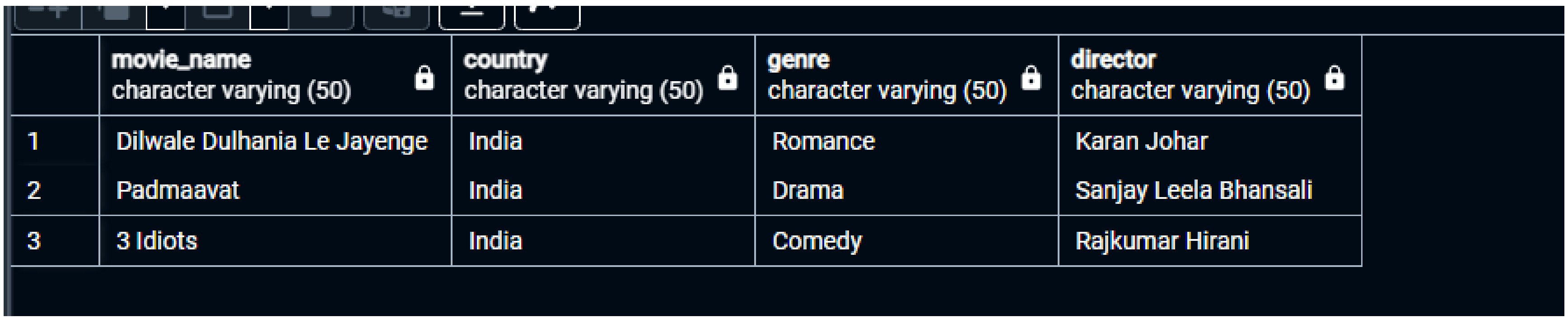
	producer character varying (50)
1	eee

12.



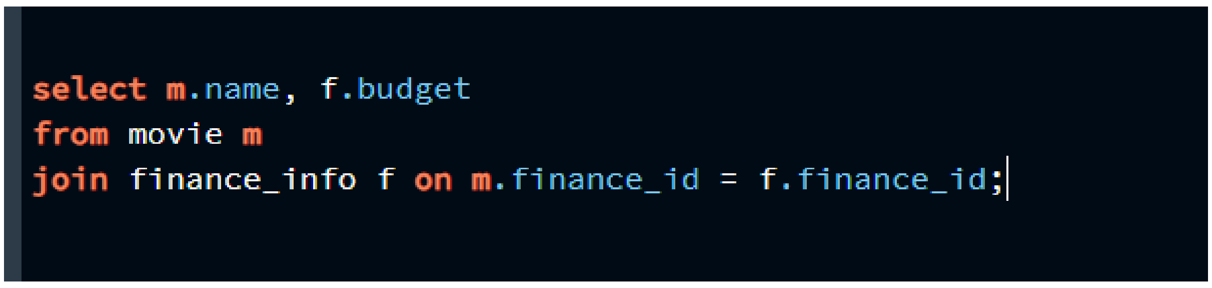
```
select m.name as movie_name, m.country, m.genre, ct.director
from movie m
join creative_team ct on m.team_id = ct.team_id;
```

OUTPUT



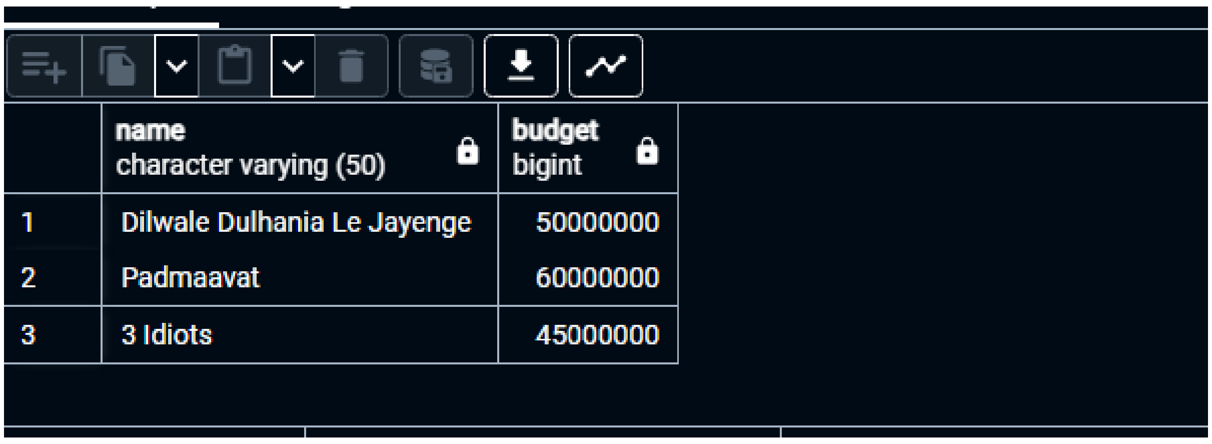
	movie_name character varying (50)	country character varying (50)	genre character varying (50)	director character varying (50)
1	Dilwale Dulhania Le Jayenge	India	Romance	Karan Johar
2	Padmaavat	India	Drama	Sanjay Leela Bhansali
3	3 Idiots	India	Comedy	Rajkumar Hirani

13.



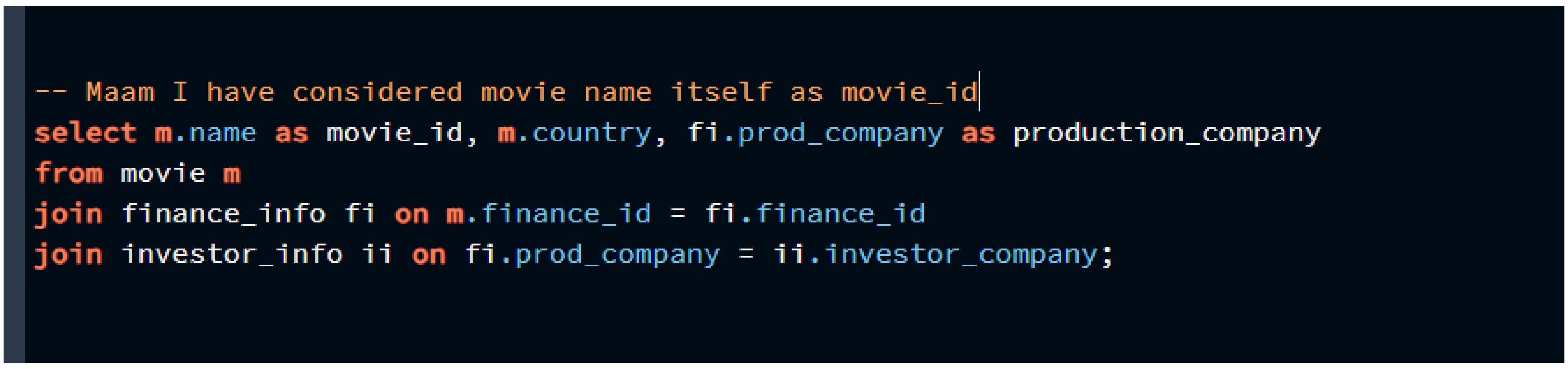
```
select m.name, f.budget
from movie m
join finance_info f on m.finance_id = f.finance_id;
```

OUTPUT



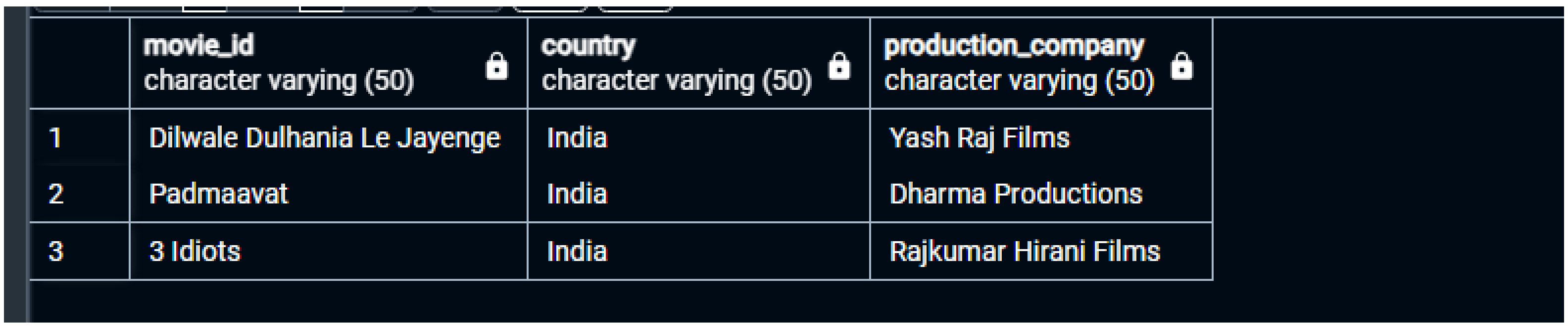
	name character varying (50)	budget bigint
1	Dilwale Dulhania Le Jayenge	50000000
2	Padmaavat	60000000
3	3 Idiots	45000000

14.



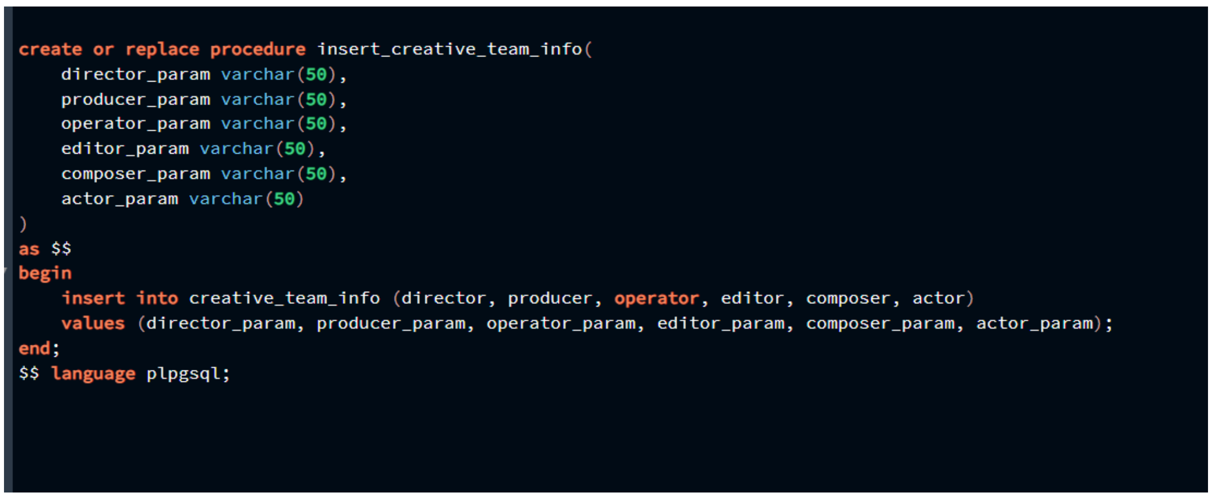
```
-- Maam I have considered movie name itself as movie_id
select m.name as movie_id, m.country, fi.prod_company as production_company
from movie m
join finance_info fi on m.finance_id = fi.finance_id
join investor_info ii on fi.prod_company = ii.investor_company;
```

OUTPUT



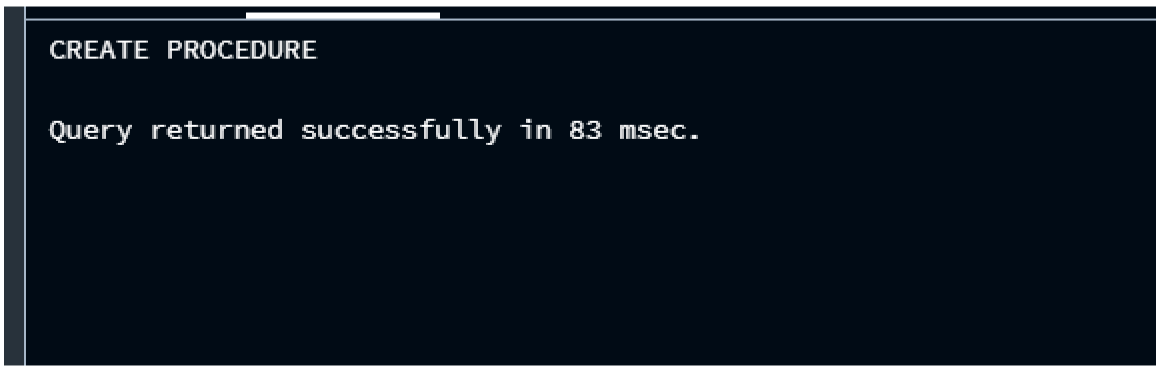
	movie_id character varying (50)	country character varying (50)	production_company character varying (50)
1	Dilwale Dulhania Le Jayenge	India	Yash Raj Films
2	Padmaavat	India	Dharma Productions
3	3 Idiots	India	Rajkumar Hirani Films

15.



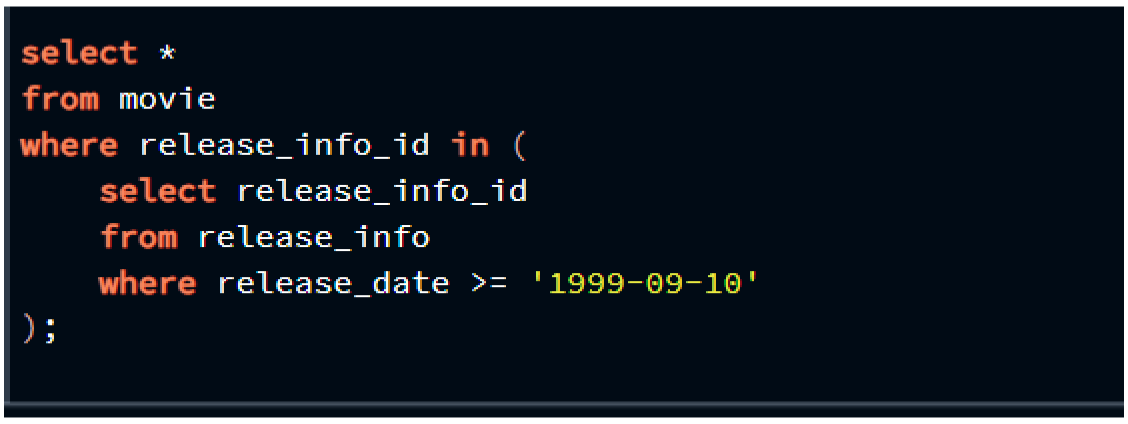
```
create or replace procedure insert_creative_team_info(
    director_param varchar(50),
    producer_param varchar(50),
    operator_param varchar(50),
    editor_param varchar(50),
    composer_param varchar(50),
    actor_param varchar(50)
)
as $$
begin
    insert into creative_team_info (director, producer, operator, editor, composer, actor)
    values (director_param, producer_param, operator_param, editor_param, composer_param, actor_param);
end;
$$ language plpgsql;
```

OUTPUT



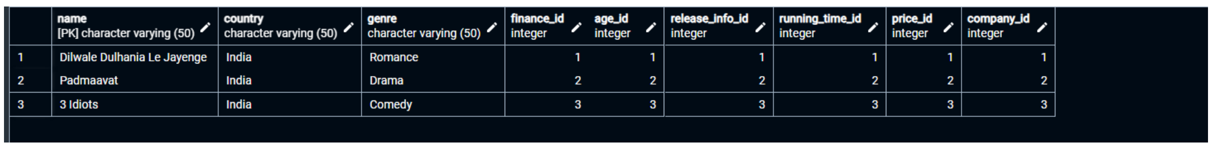
```
CREATE PROCEDURE

Query returned successfully in 83 msec.
```

16.



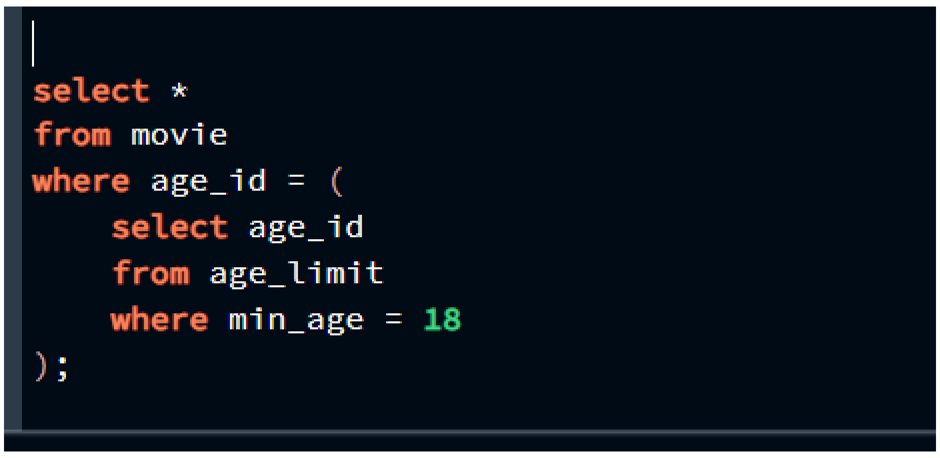
```
select *
from movie
where release_info_id in (
    select release_info_id
    from release_info
    where release_date >= '1999-09-10'
);
```

OUTPUT



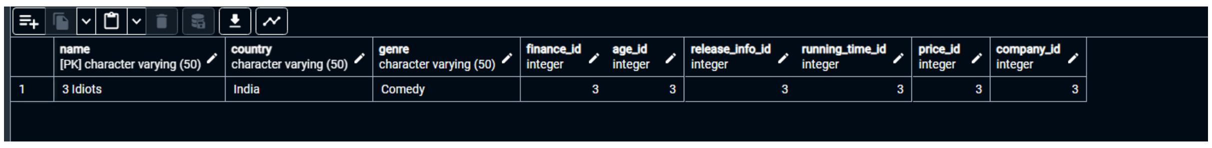
	name [PK] character varying (50)	country character varying (50)	genre character varying (50)	finance_id integer	age_id integer	release_info_id integer	running_time_id integer	price_id integer	company_id integer
1	Dilwale Dulhania Le Jayenge	India	Romance	1	1	1	1	1	1
2	Padmaavat	India	Drama	2	2	2	2	2	2
3	3 Idiots	India	Comedy	3	3	3	3	3	3

17.



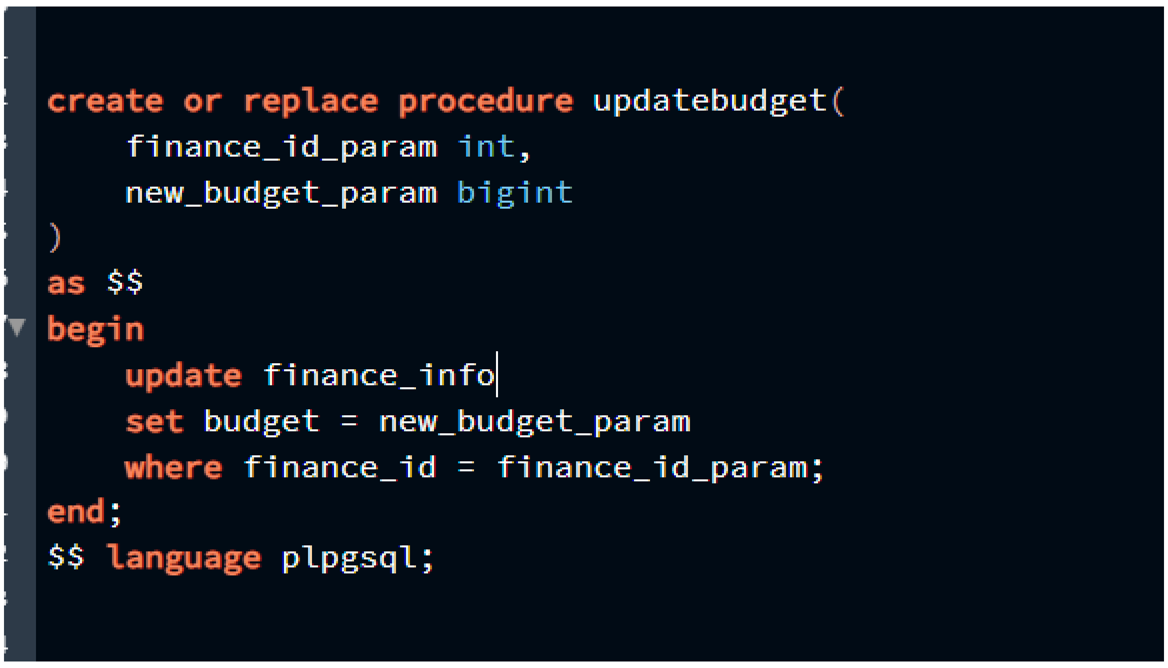
```sql
select *
from movie
where age_id = (
    select age_id
    from age_limit
    where min_age = 18
);
```

OUTPUT



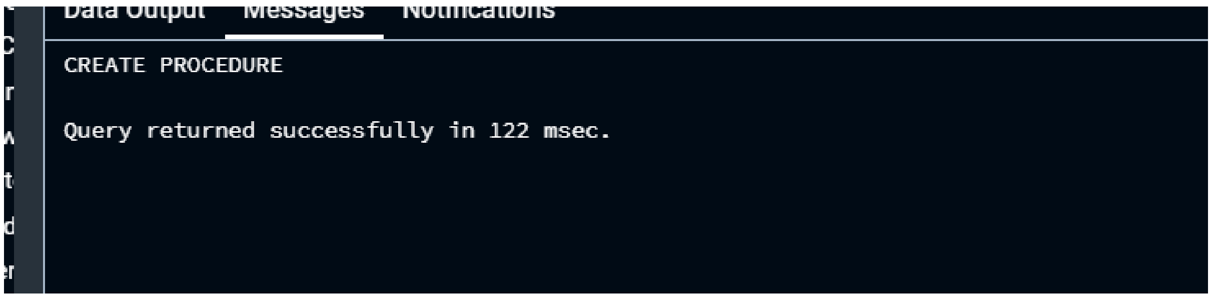
	name [PK] character varying (50)	country character varying (50)	genre character varying (50)	finance_id integer	age_id integer	release_info_id integer	running_time_id integer	price_id integer	company_id integer
1	3 Idiots	India	Comedy	3	3	3	3	3	3

18.



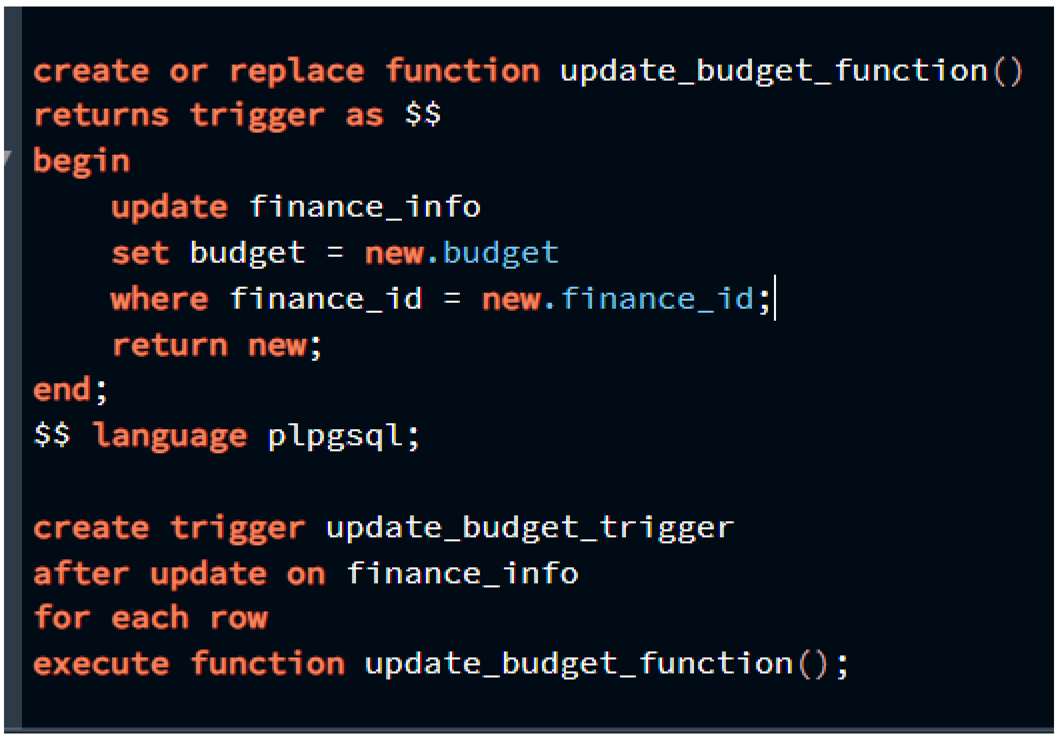
```sql
create or replace procedure updatebudget(
    finance_id_param int,
    new_budget_param bigint
)
as $$
begin
    update finance_info
    set budget = new_budget_param
    where finance_id = finance_id_param;
end;
$$ language plpgsql;
```

OUTPUT



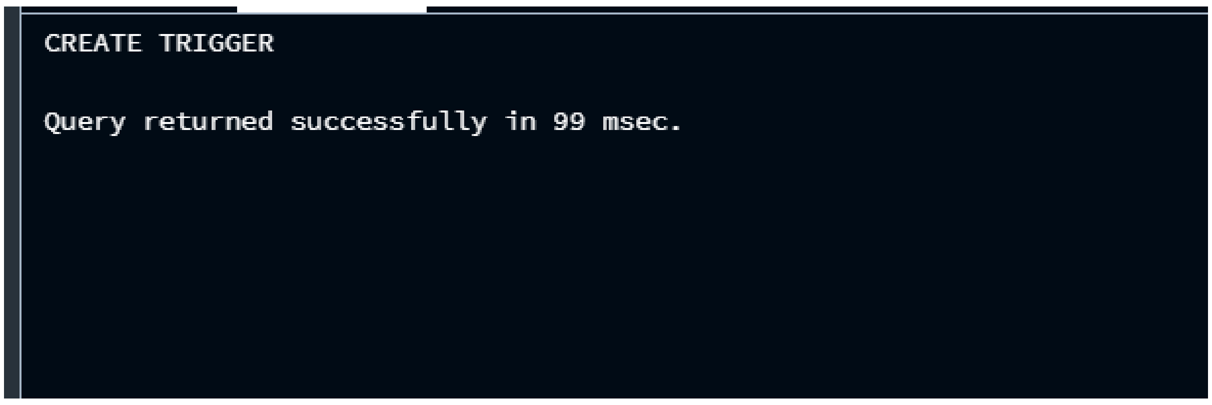
```
CREATE PROCEDURE

Query returned successfully in 122 msec.
```

19.



```
create or replace function update_budget_function()
returns trigger as $$
begin
    update finance_info
    set budget = new.budget
    where finance_id = new.finance_id;
    return new;
end;
$$ language plpgsql;

create trigger update_budget_trigger
after update on finance_info
for each row
execute function update_budget_function();
```

OUTPUT



```
CREATE TRIGGER

Query returned successfully in 99 msec.
```