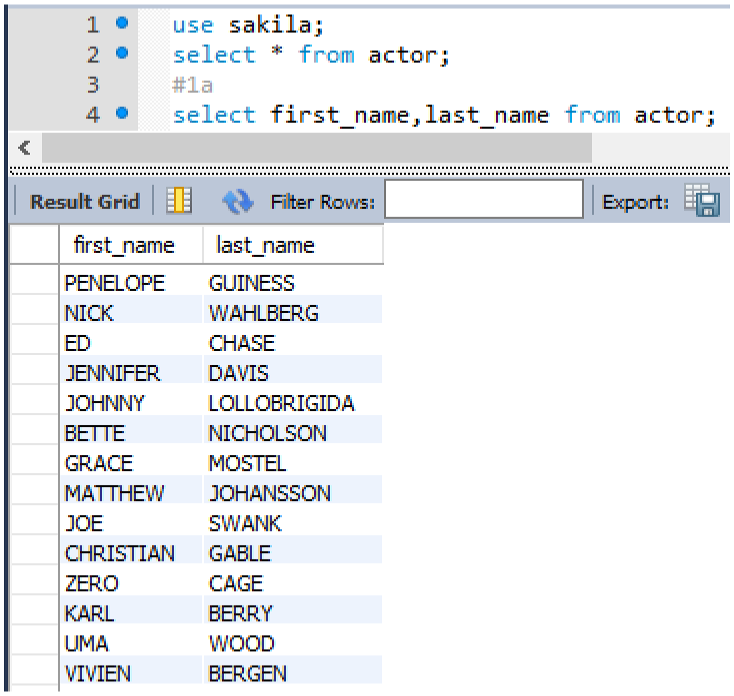
HOMEWORK 10: SQL

Kyrus Wankadiya

4/24/18

Homework Assignment

* 1a. You need a list of all the actors who have Display the first and last names of all actors from the table `actor`.

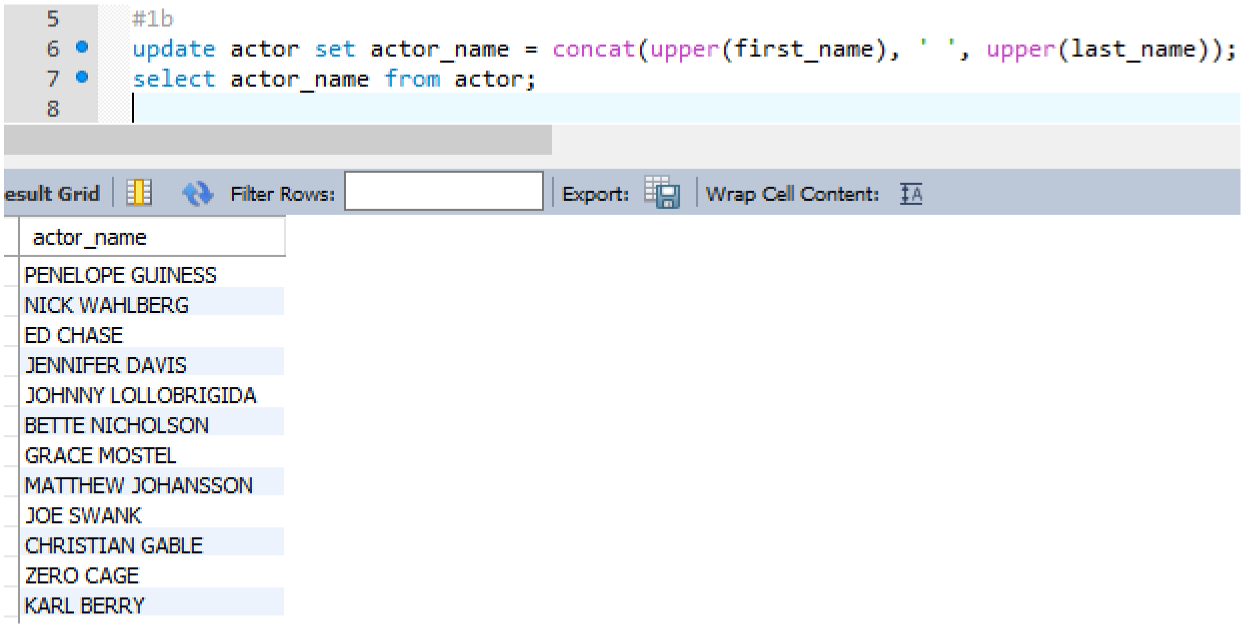


```
use sakila;
select * from actor;
#1a
select first_name,last_name from actor;
```

Result Grid | Filter Rows: | Export:

first_name	last_name
PENELOPE	GUINESS
NICK	WAHLBERG
ED	CHASE
JENNIFER	DAVIS
JOHNNY	LOLLOBRIGIDA
BETTE	NICHOLSON
GRACE	MOSTEL
MATTHEW	JOHANSSON
JOE	SWANK
CHRISTIAN	GABLE
ZERO	CAGE
KARL	BERRY
UMA	WOOD
VIVIEN	BERGEN

* 1b. Display the first and last name of each actor in a single column in upper case letters. Name the column `Actor Name`.

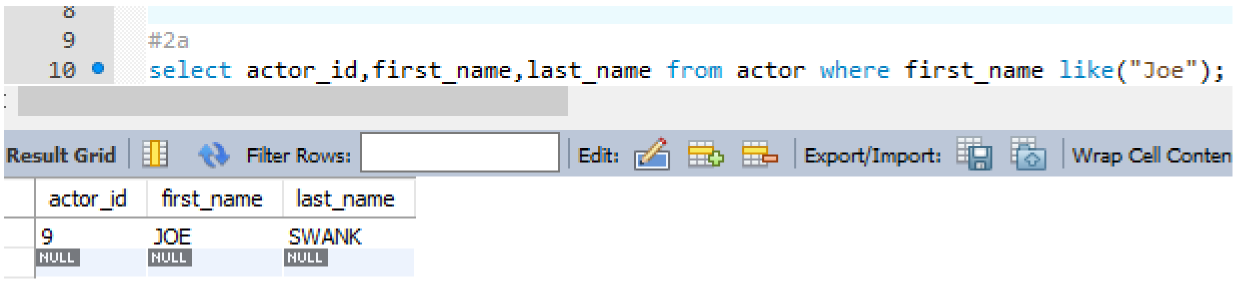


```
#1b
update actor set actor_name = concat(upper(first_name), ' ', upper(last_name));
select actor_name from actor;
```

esult Grid | Filter Rows: | Export: | Wrap Cell Content:

actor_name
PENELOPE GUINESS
NICK WAHLBERG
ED CHASE
JENNIFER DAVIS
JOHNNY LOLLOBRIGIDA
BETTE NICHOLSON
GRACE MOSTEL
MATTHEW JOHANSSON
JOE SWANK
CHRISTIAN GABLE
ZERO CAGE
KARL BERRY

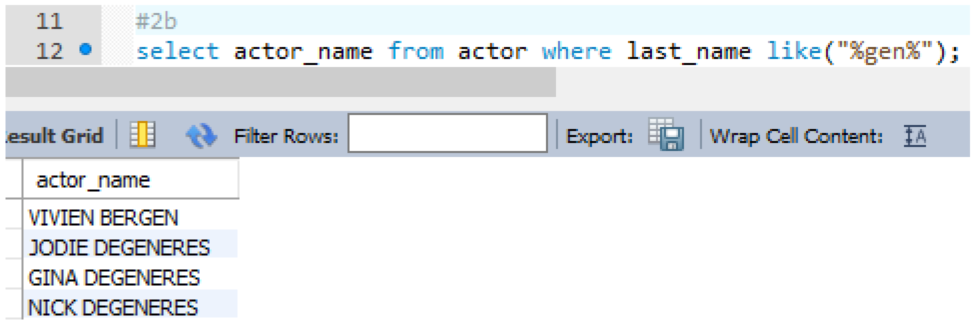
* 2a. You need to find the ID number, first name, and last name of an actor, of whom you know only the first name, "Joe." What is one query would you use to obtain this information?



```
#2a
select actor_id,first_name,last_name from actor where first_name like("Joe");
```

actor_id	first_name	last_name
9	JOE	SWANK
NULL	NULL	NULL

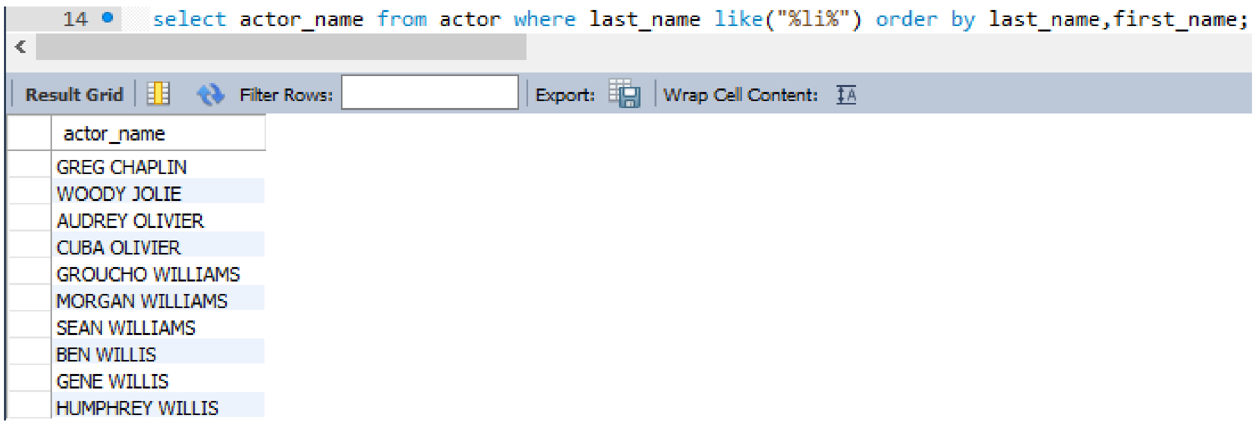
* 2b. Find all actors whose last name contain the letters `GEN`:



```
#2b
select actor_name from actor where last_name like("%gen%");
```

actor_name
VIVIEN BERGEN
JODIE DEGENERES
GINA DEGENERES
NICK DEGENERES

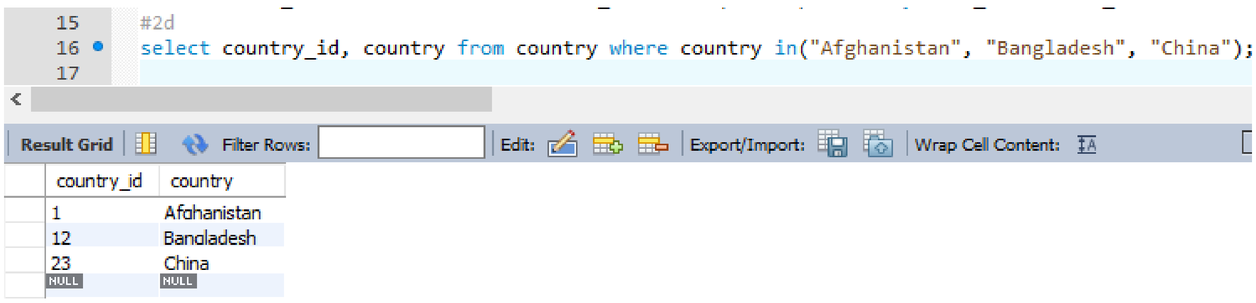
* 2c. Find all actors whose last names contain the letters `LI`. This time, order the rows by last name and first name, in that order:



```
select actor_name from actor where last_name like("%li%") order by last_name,first_name;
```

actor_name
GREG CHAPLIN
WOODY JOLIE
AUDREY OLIVIER
CUBA OLIVIER
GROUCHO WILLIAMS
MORGAN WILLIAMS
SEAN WILLIAMS
BEN WILLIS
GENE WILLIS
HUMPHREY WILLIS

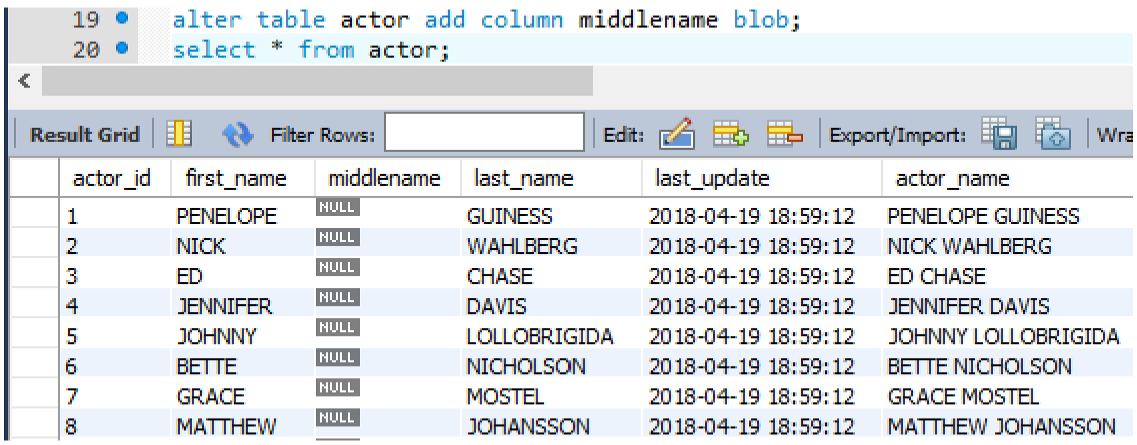
* 2d. Using `IN`, display the `country_id` and `country` columns of the following countries: Afghanistan, Bangladesh, and China:



```
#2d
select country_id, country from country where country in("Afghanistan", "Bangladesh", "China");
```

country_id	country
1	Afghanistan
12	Bangladesh
23	China
NULL	NULL

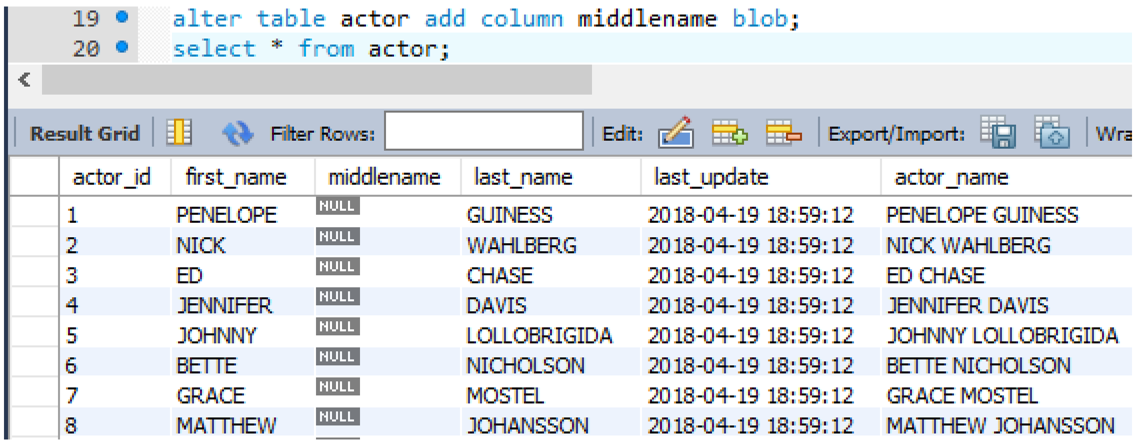
* 3a. Add a `middle_name` column to the table `actor`. Position it between `first_name` and `last_name`. Hint: you will need to specify the data type.



```
alter table actor add column middlename blob;
select * from actor;
```

actor_id	first_name	middlename	last_name	last_update	actor_name
1	PENELOPE	NULL	GUINESS	2018-04-19 18:59:12	PENELOPE GUINESS
2	NICK	NULL	WAHLBERG	2018-04-19 18:59:12	NICK WAHLBERG
3	ED	NULL	CHASE	2018-04-19 18:59:12	ED CHASE
4	JENNIFER	NULL	DAVIS	2018-04-19 18:59:12	JENNIFER DAVIS
5	JOHNNY	NULL	LOLLOBRIGIDA	2018-04-19 18:59:12	JOHNNY LOLLOBRIGIDA
6	BETTE	NULL	NICHOLSON	2018-04-19 18:59:12	BETTE NICHOLSON
7	GRACE	NULL	MOSTEL	2018-04-19 18:59:12	GRACE MOSTEL
8	MATTHEW	NULL	JOHANSSON	2018-04-19 18:59:12	MATTHEW JOHANSSON

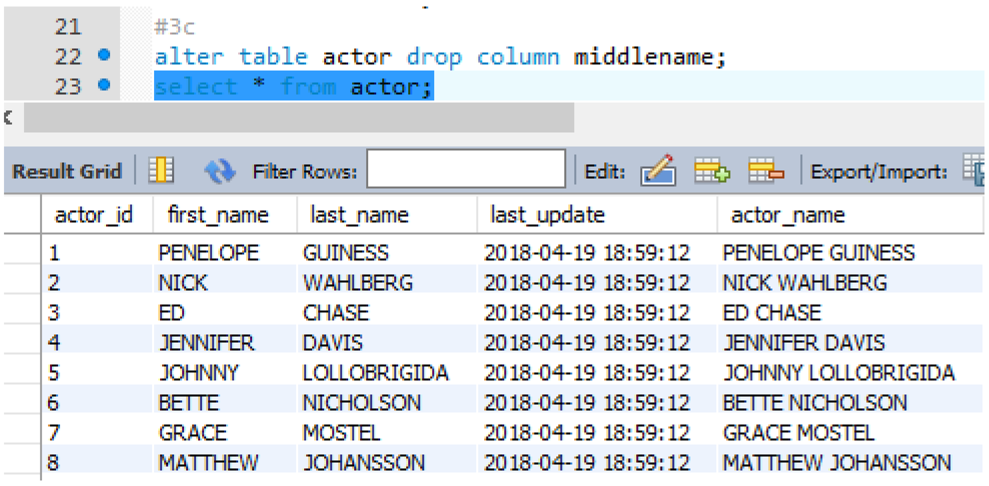
* 3b. You realize that some of these actors have tremendously long last names. Change the data type of the `middle_name` column to `blobs`.



```
alter table actor add column middlename blob;
select * from actor;
```

actor_id	first_name	middlename	last_name	last_update	actor_name
1	PENELOPE	NULL	GUINESS	2018-04-19 18:59:12	PENELOPE GUINESS
2	NICK	NULL	WAHLBERG	2018-04-19 18:59:12	NICK WAHLBERG
3	ED	NULL	CHASE	2018-04-19 18:59:12	ED CHASE
4	JENNIFER	NULL	DAVIS	2018-04-19 18:59:12	JENNIFER DAVIS
5	JOHNNY	NULL	LOLLOBRIGIDA	2018-04-19 18:59:12	JOHNNY LOLLOBRIGIDA
6	BETTE	NULL	NICHOLSON	2018-04-19 18:59:12	BETTE NICHOLSON
7	GRACE	NULL	MOSTEL	2018-04-19 18:59:12	GRACE MOSTEL
8	MATTHEW	NULL	JOHANSSON	2018-04-19 18:59:12	MATTHEW JOHANSSON

* 3c. Now delete the `middle_name` column.

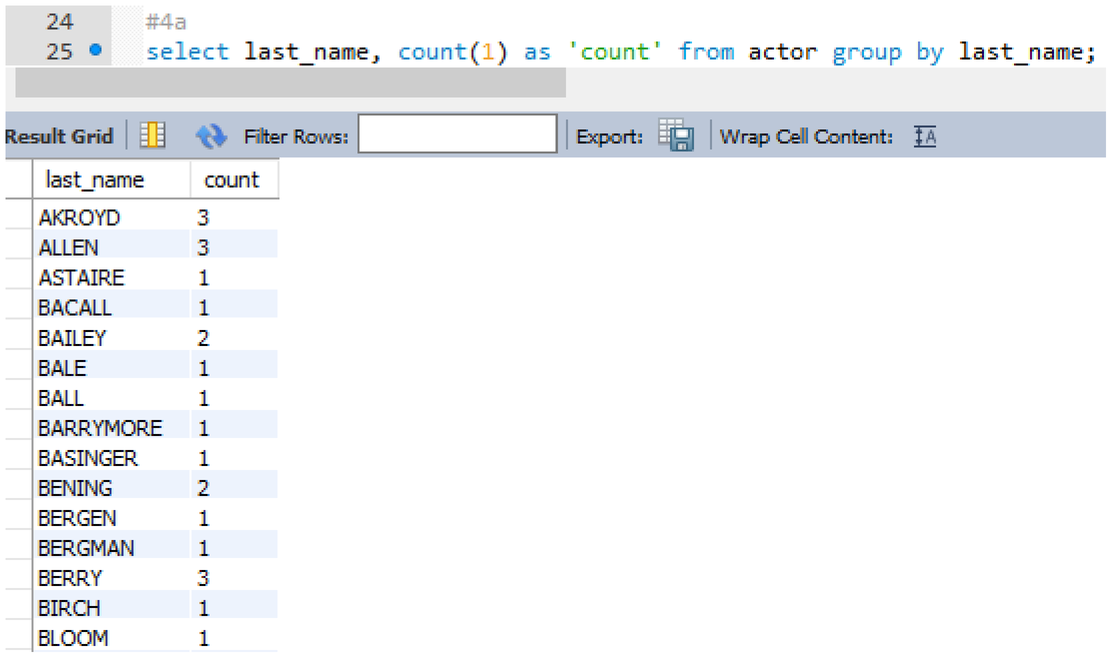


```
#3c
alter table actor drop column middlename;
select * from actor;
```

Result Grid | Filter Rows: | Edit: | Export/Import:

actor_id	first_name	last_name	last_update	actor_name
1	PENELOPE	GUINESS	2018-04-19 18:59:12	PENELOPE GUINESS
2	NICK	WAHLBERG	2018-04-19 18:59:12	NICK WAHLBERG
3	ED	CHASE	2018-04-19 18:59:12	ED CHASE
4	JENNIFER	DAVIS	2018-04-19 18:59:12	JENNIFER DAVIS
5	JOHNNY	LOLLOBRIGIDA	2018-04-19 18:59:12	JOHNNY LOLLOBRIGIDA
6	BETTE	NICHOLSON	2018-04-19 18:59:12	BETTE NICHOLSON
7	GRACE	MOSTEL	2018-04-19 18:59:12	GRACE MOSTEL
8	MATTHEW	JOHANSSON	2018-04-19 18:59:12	MATTHEW JOHANSSON

* 4a. List the last names of actors, as well as how many actors have that last name.

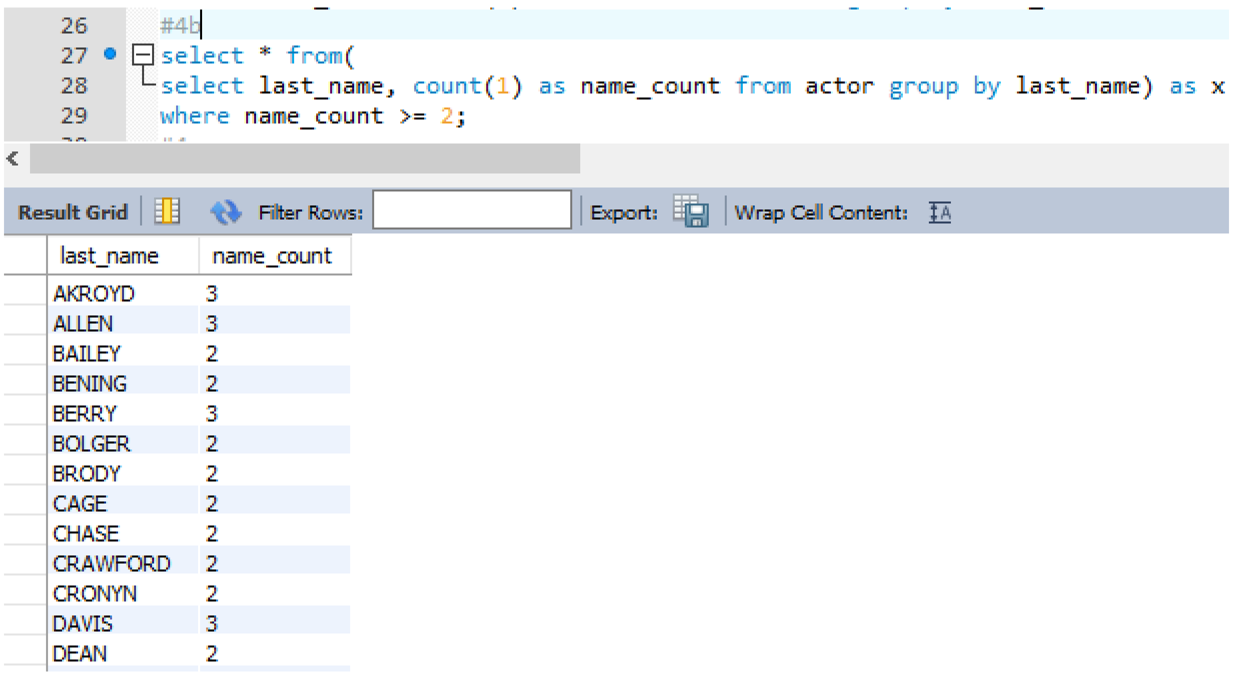


```
#4a
select last_name, count(1) as 'count' from actor group by last_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

last_name	count
AKROYD	3
ALLEN	3
ASTAIRE	1
BACALL	1
BAILEY	2
BALE	1
BALL	1
BARRYMORE	1
BASINGER	1
BENING	2
BERGEN	1
BERGMAN	1
BERRY	3
BIRCH	1
BLOOM	1

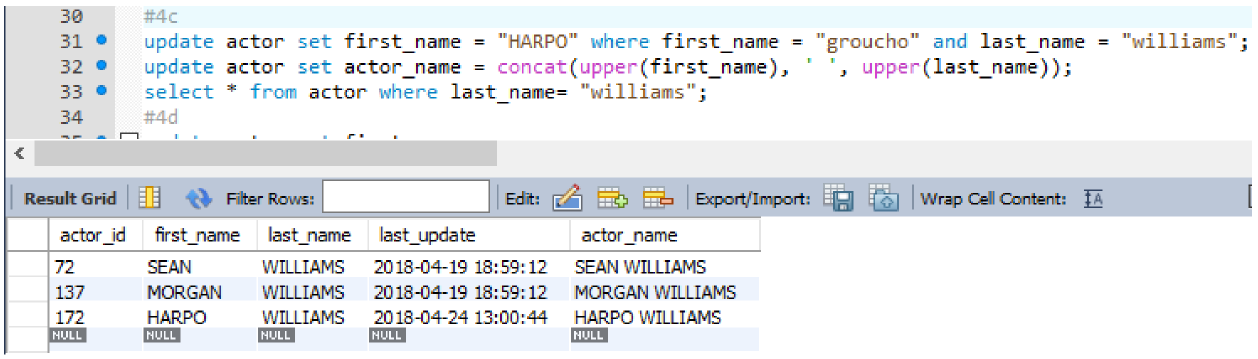
* 4b. List last names of actors and the number of actors who have that last name, but only for names that are shared by at least two actors



```
#4b
select * from(
select last_name, count(1) as name_count from actor group by last_name) as x
where name_count >= 2;
```

last_name	name_count
AKROYD	3
ALLEN	3
BAILEY	2
BENING	2
BERRY	3
BOLGER	2
BRODY	2
CAGE	2
CHASE	2
CRAWFORD	2
CRONYN	2
DAVIS	3
DEAN	2

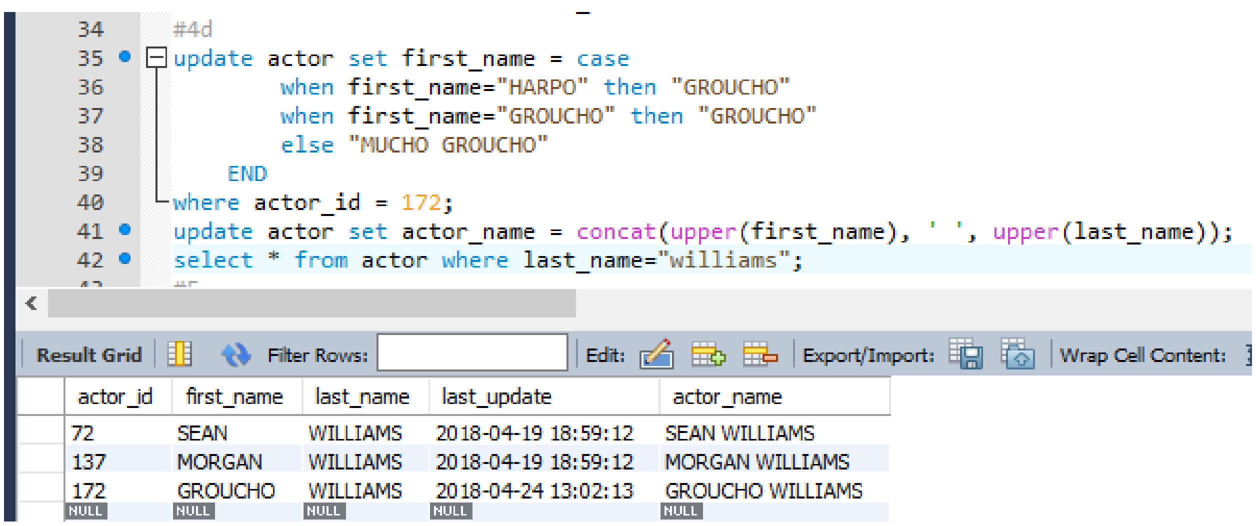
* 4c. Oh, no! The actor `HARPO WILLIAMS` was accidentally entered in the `actor` table as `GROUCHO WILLIAMS`, the name of Harpo's second cousin's husband's yoga teacher. Write a query to fix the record.



```
#4c
update actor set first_name = "HARPO" where first_name = "groucho" and last_name = "williams";
update actor set actor_name = concat(upper(first_name), ' ', upper(last_name));
select * from actor where last_name= "williams";
#4d
```

actor_id	first_name	last_name	last_update	actor_name
72	SEAN	WILLIAMS	2018-04-19 18:59:12	SEAN WILLIAMS
137	MORGAN	WILLIAMS	2018-04-19 18:59:12	MORGAN WILLIAMS
172	HARPO	WILLIAMS	2018-04-24 13:00:44	HARPO WILLIAMS
NULL	NULL	NULL	NULL	NULL

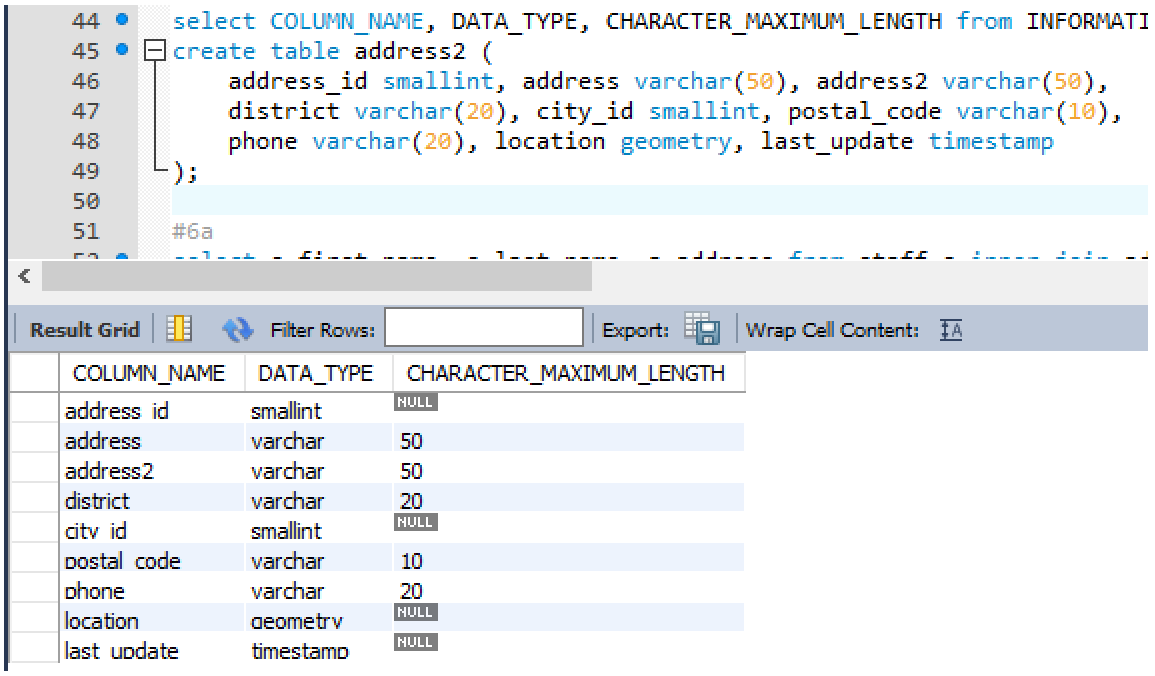
* 4d. Perhaps we were too hasty in changing `GROUCHO` to `HARPO`. It turns out that `GROUCHO` was the correct name after all! In a single query, if the first name of the actor is currently `HARPO`, change it to `GROUCHO`. Otherwise, change the first name to `MUCHO GROUCHO`, as that is exactly what the actor will be with the grievous error. BE CAREFUL NOT TO CHANGE THE FIRST NAME OF EVERY ACTOR TO `MUCHO GROUCHO`, HOWEVER! (Hint: update the record using a unique identifier.)



```sql
#4d
update actor set first_name = case
        when first_name="HARPO" then "GROUCHO"
        when first_name="GROUCHO" then "GROUCHO"
        else "MUCHO GROUCHO"
    END
where actor_id = 172;
update actor set actor_name = concat(upper(first_name), ' ', upper(last_name));
select * from actor where last_name="williams";
```

actor_id	first_name	last_name	last_update	actor_name
72	SEAN	WILLIAMS	2018-04-19 18:59:12	SEAN WILLIAMS
137	MORGAN	WILLIAMS	2018-04-19 18:59:12	MORGAN WILLIAMS
172	GROUCHO	WILLIAMS	2018-04-24 13:02:13	GROUCHO WILLIAMS
NULL	NULL	NULL	NULL	NULL

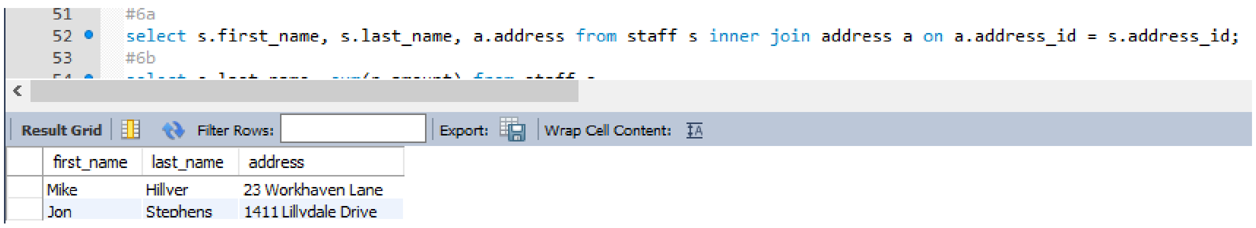
* 5a. You cannot locate the schema of the `address` table. Which query would you use to re-create it?



```sql
select COLUMN_NAME, DATA_TYPE, CHARACTER_MAXIMUM_LENGTH from INFORMATI
create table address2 (
    address_id smallint, address varchar(50), address2 varchar(50),
    district varchar(20), city_id smallint, postal_code varchar(10),
    phone varchar(20), location geometry, last_update timestamp
);

#6a
```

COLUMN_NAME	DATA_TYPE	CHARACTER_MAXIMUM_LENGTH
address_id	smallint	NULL
address	varchar	50
address2	varchar	50
district	varchar	20
city_id	smallint	NULL
postal_code	varchar	10
phone	varchar	20
location	geometry	NULL
last_update	timestamp	NULL

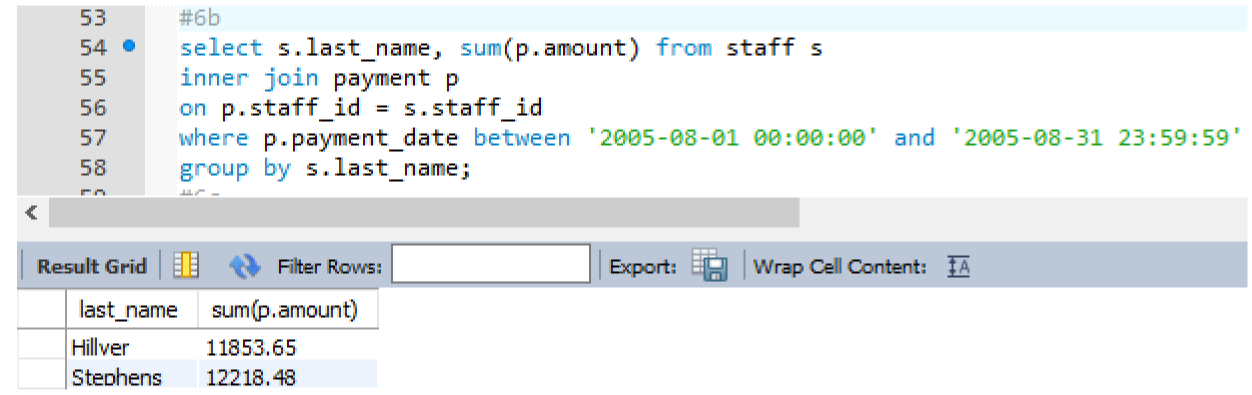
* 6a. Use `JOIN` to display the first and last names, as well as the address, of each staff member. Use the tables `staff` and `address`:



```sql
#6a
select s.first_name, s.last_name, a.address from staff s inner join address a on a.address_id = s.address_id;
#6b
```

first_name	last_name	address
Mike	Hillyer	23 Workhaven Lane
Jon	Stephens	1411 Lillydale Drive

* 6b. Use `JOIN` to display the total amount rung up by each staff member in August of 2005. Use tables `staff` and `payment`.

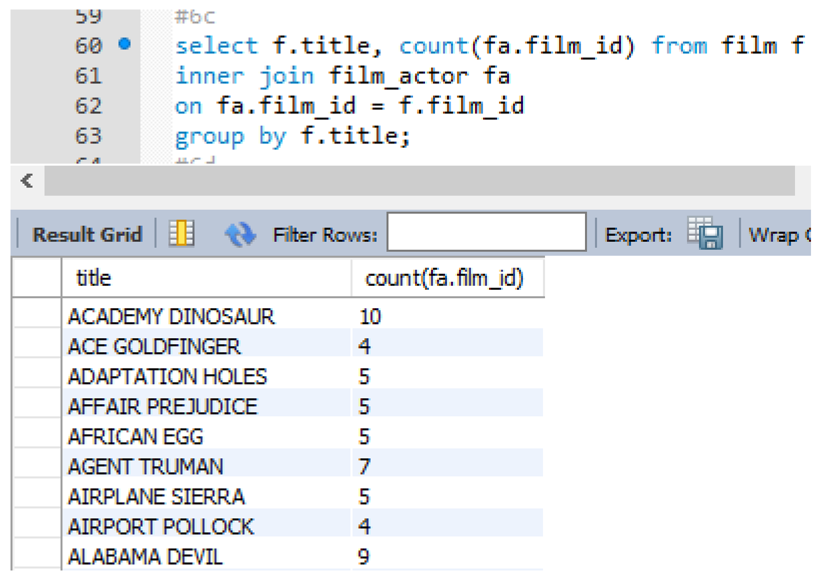


```
#6b
select s.last_name, sum(p.amount) from staff s
inner join payment p
on p.staff_id = s.staff_id
where p.payment_date between '2005-08-01 00:00:00' and '2005-08-31 23:59:59'
group by s.last_name;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

last_name	sum(p.amount)
Hillyer	11853.65
Stephens	12218.48

* 6c. List each film and the number of actors who are listed for that film. Use tables `film_actor` and `film`. Use inner join.

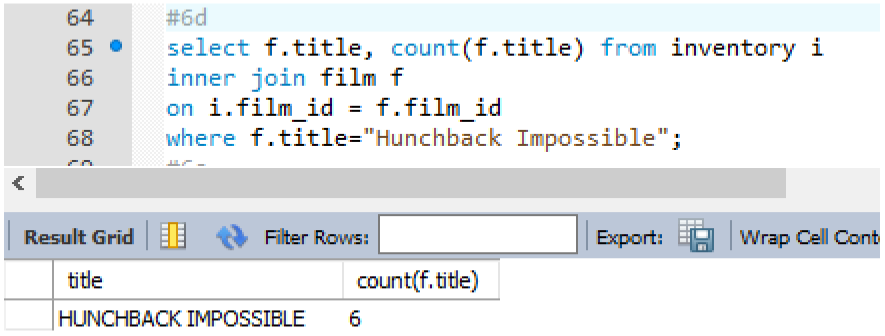


```
#6c
select f.title, count(fa.film_id) from film f
inner join film_actor fa
on fa.film_id = f.film_id
group by f.title;
```

Result Grid | Filter Rows: | Export: | Wrap

title	count(fa.film_id)
ACADEMY DINOSAUR	10
ACE GOLDFINGER	4
ADAPTATION HOLES	5
AFFAIR PREJUDICE	5
AFRICAN EGG	5
AGENT TRUMAN	7
AIRPLANE SIERRA	5
AIRPORT POLLOCK	4
ALABAMA DEVIL	9

* 6d. How many copies of the film `Hunchback Impossible` exist in the inventory system?



```
#6d
select f.title, count(f.title) from inventory i
inner join film f
on i.film_id = f.film_id
where f.title="Hunchback Impossible";
```

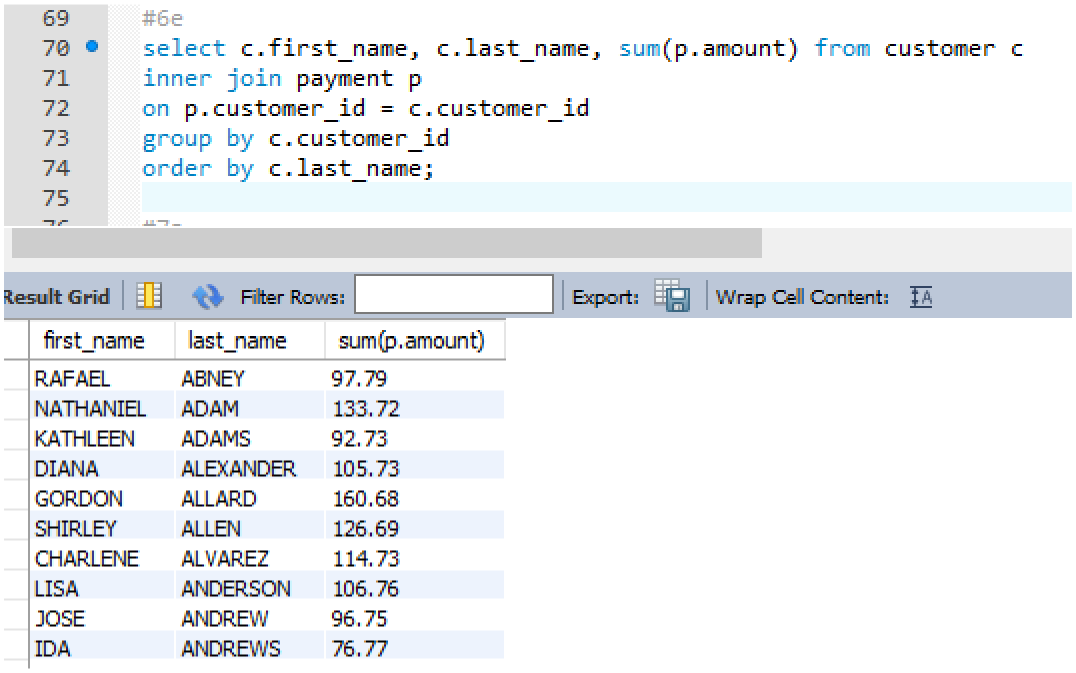
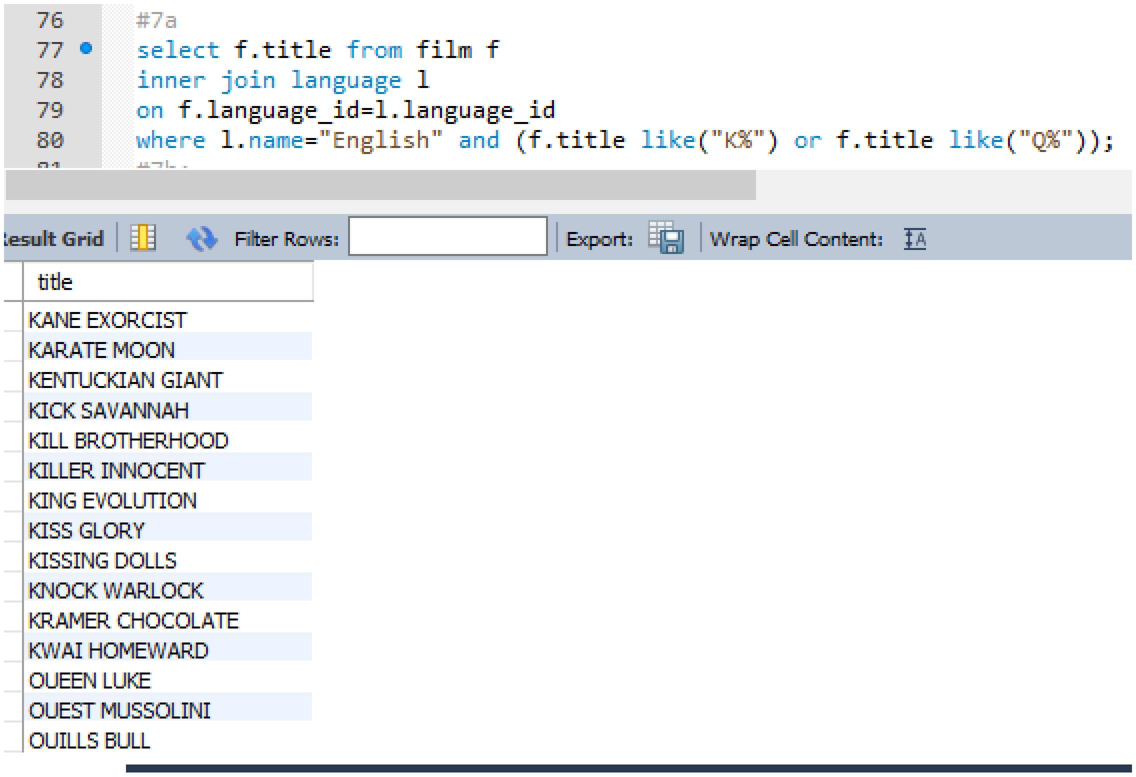
Result Grid | Filter Rows: | Export: | Wrap Cell Cont

title	count(f.title)
HUNCHBACK IMPOSSIBLE	6

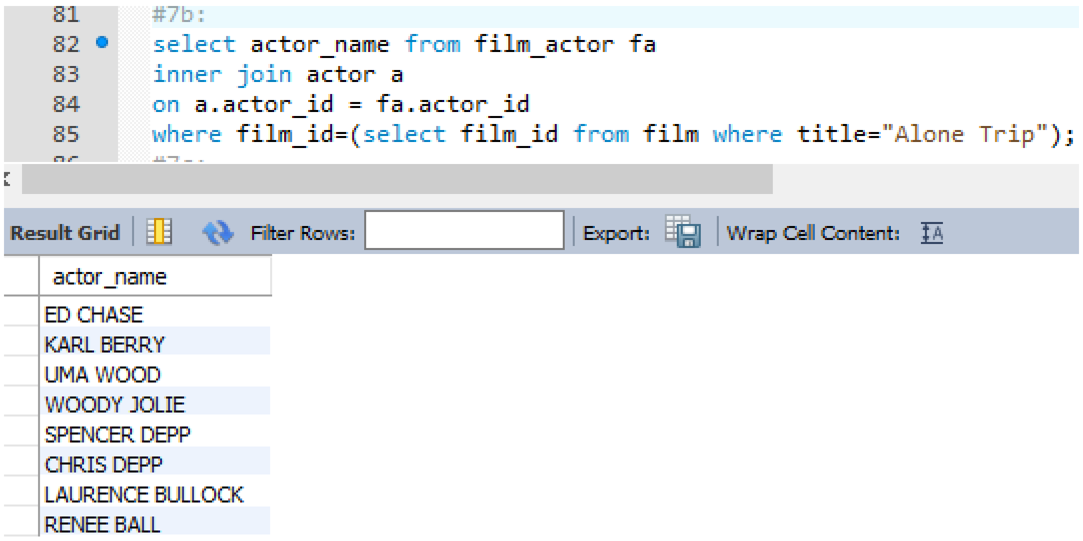
* 6e. Using the tables `payment` and `customer` and the `JOIN` command, list the total paid by each customer. List the customers alphabetically by last name:

```
```

![Total amount paid](Images/total_payment.png)

```



```
#6e
select c.first_name, c.last_name, sum(p.amount) from customer c
inner join payment p
on p.customer_id = c.customer_id
group by c.customer_id
order by c.last_name;
```

| first_name | last_name | sum(p.amount) |
|---|---|---|
| RAFAEL | ABNEY | 97.79 |
| NATHANIEL | ADAM | 133.72 |
| KATHLEEN | ADAMS | 92.73 |
| DIANA | ALEXANDER | 105.73 |
| GORDON | ALLARD | 160.68 |
| SHIRLEY | ALLEN | 126.69 |
| CHARLENE | ALVAREZ | 114.73 |
| LISA | ANDERSON | 106.76 |
| JOSE | ANDREW | 96.75 |
| IDA | ANDREWS | 76.77 |

* 7a. The music of Queen and Kris Kristofferson have seen an unlikely resurgence. As an unintended consequence, films starting with the letters `K` and `Q` have also soared in popularity. Use subqueries to display the titles of movies starting with the letters `K` and `Q` whose language is English.



```
#7a
select f.title from film f
inner join language l
on f.language_id=l.language_id
where l.name="English" and (f.title like("K%") or f.title like("Q%"));
```

title
KANE EXORCIST
KARATE MOON
KENTUCKIAN GIANT
KICK SAVANNAH
KILL BROTHERHOOD
KILLER INNOCENT
KING EVOLUTION
KISS GLORY
KISSING DOLLS
KNOCK WARLOCK
KRAMER CHOCOLATE
KWAI HOMEWARD
QUEEN LUKE
QUEST MUSSOLINI
QUILLS BULL
```

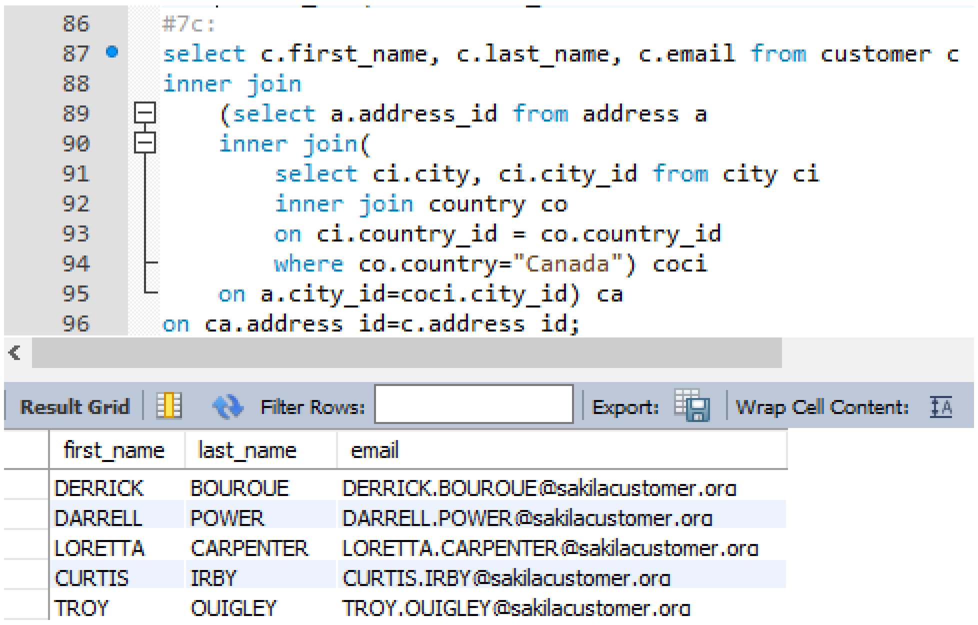
* 7b. Use subqueries to display all actors who appear in the film `Alone Trip`.



```
#7b:
select actor_name from film_actor fa
inner join actor a
on a.actor_id = fa.actor_id
where film_id=(select film_id from film where title="Alone Trip");
```

actor_name
ED CHASE
KARL BERRY
UMA WOOD
WOODY JOLIE
SPENCER DEPP
CHRIS DEPP
LAURENCE BULLOCK
RENEE BALL

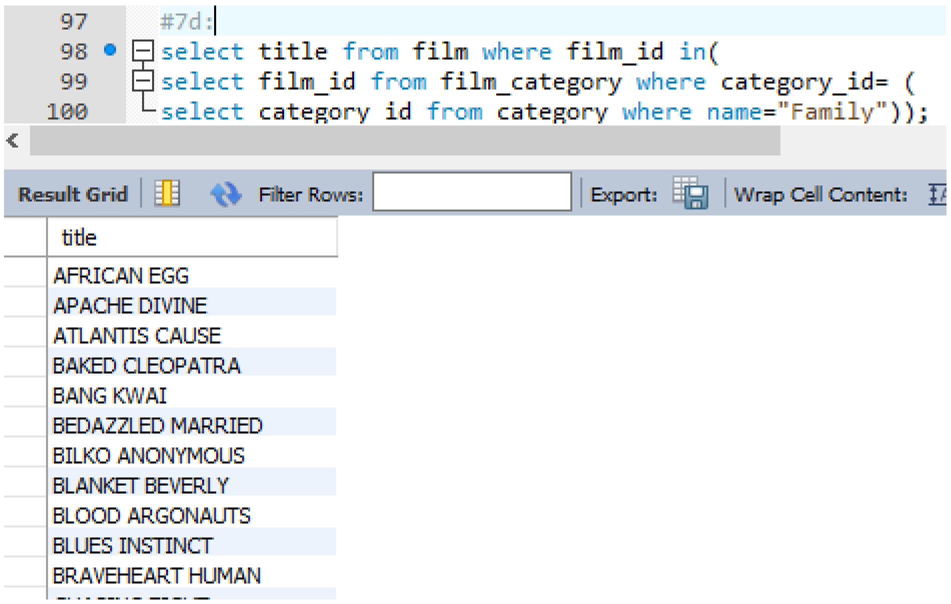
* 7c. You want to run an email marketing campaign in Canada, for which you will need the names and email addresses of all Canadian customers. Use joins to retrieve this information.



```
#7c:
select c.first_name, c.last_name, c.email from customer c
inner join
    (select a.address_id from address a
    inner join(
        select ci.city, ci.city_id from city ci
        inner join country co
        on ci.country_id = co.country_id
        where co.country="Canada") coci
    on a.city_id=coci.city_id) ca
on ca.address_id=c.address_id;
```

first_name	last_name	email
DERRICK	BOURQUE	DERRICK.BOURQUE@sakilacustomer.org
DARRELL	POWER	DARRELL.POWER@sakilacustomer.org
LORETTA	CARPENTER	LORETTA.CARPENTER@sakilacustomer.org
CURTIS	IRBY	CURTIS.IRBY@sakilacustomer.org
TROY	QUIGLEY	TROY.QUIGLEY@sakilacustomer.org

* 7d. Sales have been lagging among young families, and you wish to target all family movies for a promotion. Identify all movies categorized as famiy films.

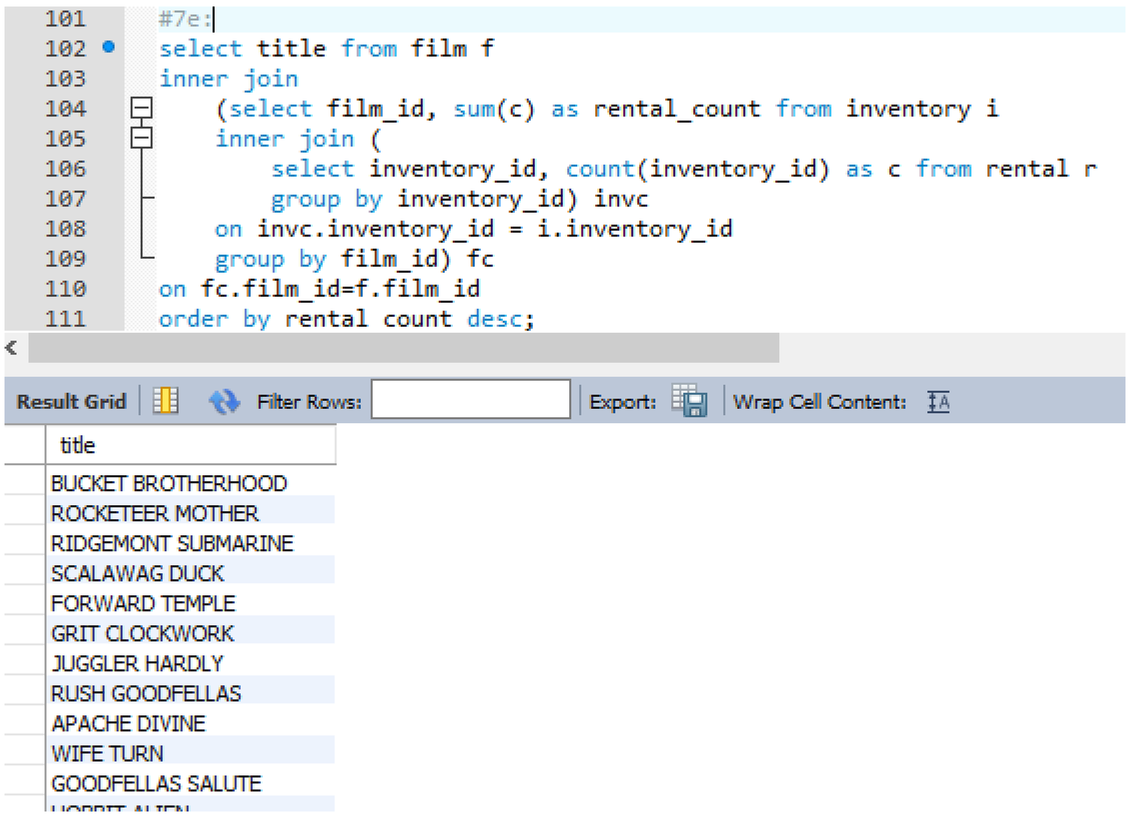


```
#7d:
select title from film where film_id in(
select film_id from film_category where category_id= (
select category id from category where name="Family"));
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

title
AFRICAN EGG
APACHE DIVINE
ATLANTIS CAUSE
BAKED CLEOPATRA
BANG KWAI
BEDAZZLED MARRIED
BILKO ANONYMOUS
BLANKET BEVERLY
BLOOD ARGONAUTS
BLUES INSTINCT
BRAVEHEART HUMAN

* 7e. Display the most frequently rented movies in descending order.

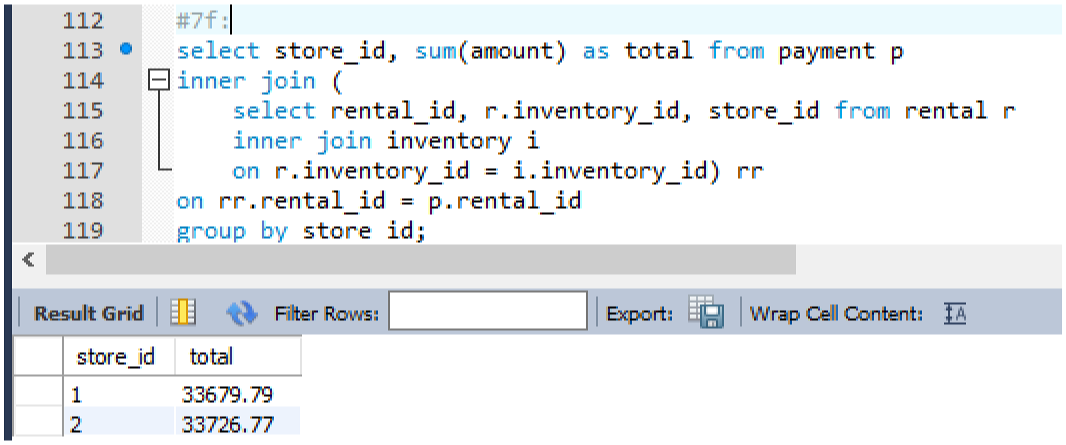


```
#7e:
select title from film f
inner join
    (select film_id, sum(c) as rental_count from inventory i
    inner join (
        select inventory_id, count(inventory_id) as c from rental r
        group by inventory_id) invc
    on invc.inventory_id = i.inventory_id
    group by film_id) fc
on fc.film_id=f.film_id
order by rental count desc;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

title
BUCKET BROTHERHOOD
ROCKETEER MOTHER
RIDGEMONT SUBMARINE
SCALAWAG DUCK
FORWARD TEMPLE
GRIT CLOCKWORK
JUGGLER HARDLY
RUSH GOODFELLAS
APACHE DIVINE
WIFE TURN
GOODFELLAS SALUTE

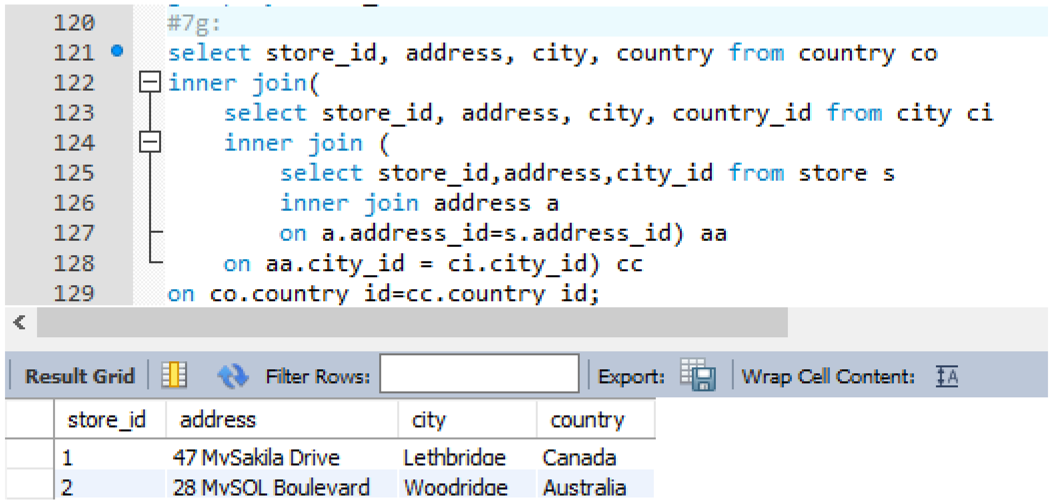
* 7f. Write a query to display how much business, in dollars, each store brought in.


```
#7f:
select store_id, sum(amount) as total from payment p
inner join (
    select rental_id, r.inventory_id, store_id from rental r
    inner join inventory i
    on r.inventory_id = i.inventory_id) rr
on rr.rental_id = p.rental_id
group by store_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	store_id	total
	1	33679.79
	2	33726.77

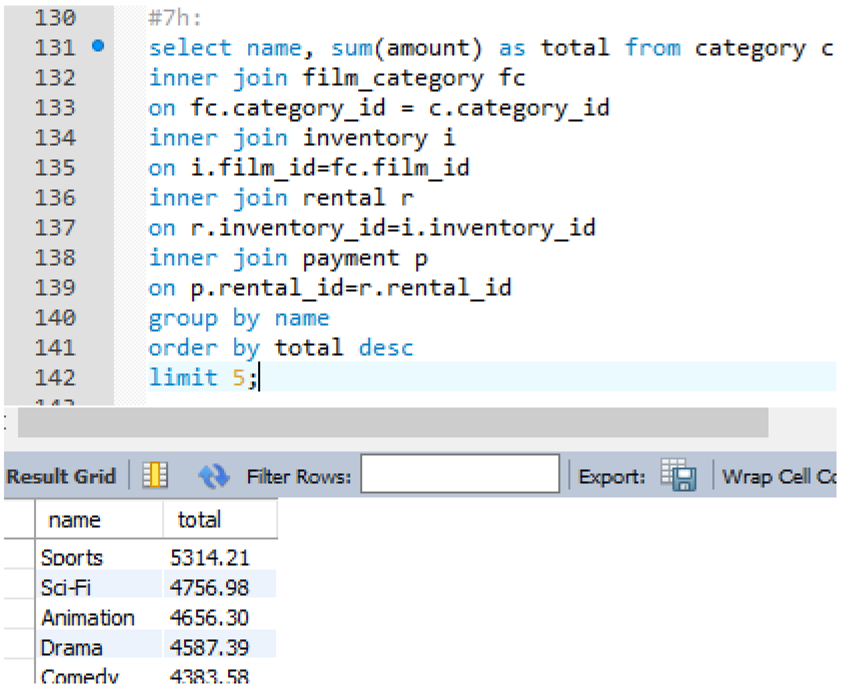
* 7g. Write a query to display for each store its store ID, city, and country.


```
#7g:
select store_id, address, city, country from country co
inner join(
    select store_id, address, city, country_id from city ci
    inner join (
        select store_id,address,city_id from store s
        inner join address a
        on a.address_id=s.address_id) aa
    on aa.city_id = ci.city_id) cc
on co.country_id=cc.country_id;
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content:

	store_id	address	city	country
	1	47 MySakila Drive	Lethbridge	Canada
	2	28 MySQL Boulevard	Woodridge	Australia

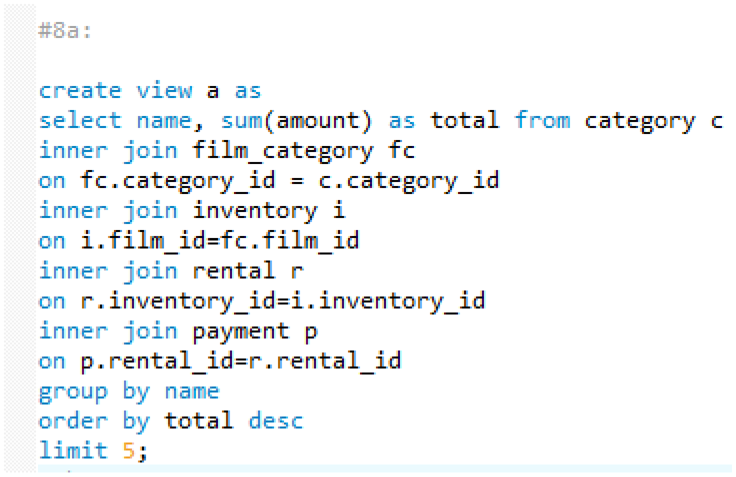
* 7h. List the top five genres in gross revenue in descending order. (**Hint**: you may need to use the following tables: category, film_category, inventory, payment, and rental.)



```
#7h:
select name, sum(amount) as total from category c
inner join film_category fc
on fc.category_id = c.category_id
inner join inventory i
on i.film_id=fc.film_id
inner join rental r
on r.inventory_id=i.inventory_id
inner join payment p
on p.rental_id=r.rental_id
group by name
order by total desc
limit 5;
```

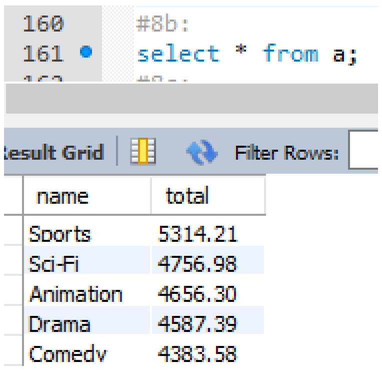
name	total
Sports	5314.21
Sci-Fi	4756.98
Animation	4656.30
Drama	4587.39
Comedy	4383.58

* 8a. In your new role as an executive, you would like to have an easy way of viewing the Top five genres by gross revenue. Use the solution from the problem above to create a view. If you haven't solved 7h, you can substitute another query to create a view.



```
#8a:

create view a as
select name, sum(amount) as total from category c
inner join film_category fc
on fc.category_id = c.category_id
inner join inventory i
on i.film_id=fc.film_id
inner join rental r
on r.inventory_id=i.inventory_id
inner join payment p
on p.rental_id=r.rental_id
group by name
order by total desc
limit 5;
```

* 8b. How would you display the view that you created in 8a?

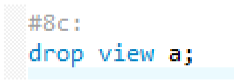


```
#8b:
select * from a;
```

Result Grid | Filter Rows:

name	total
Sports	5314.21
Sci-Fi	4756.98
Animation	4656.30
Drama	4587.39
Comedy	4383.58

* 8c. You find that you no longer need the view `top_five_genres`. Write a query to delete it.



```
#8c:
drop view a;
```