

ANALYTICS USING SQL: HACKATHON

use sakila;

-- Task 1 : Find list of all actors available in the database along with their last updated date.

```
select distinct(concat(first_name,' ',last_name)) as fullname,date(actor.last_update),avg(amount) from  
actor
```

```
join film_actor using(actor_id)
```

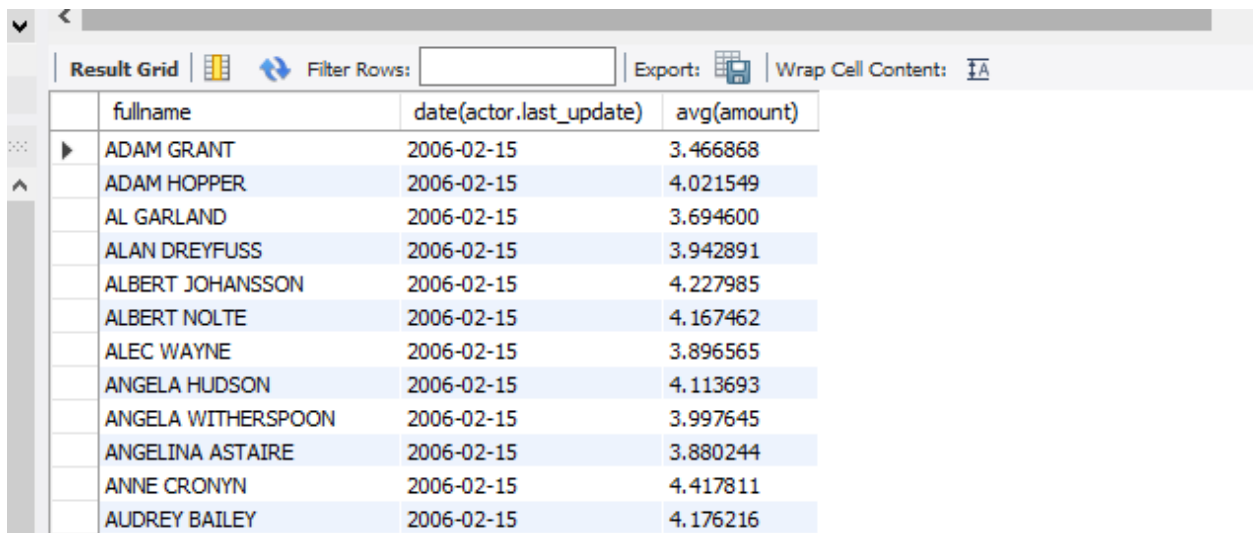
```
join film using(film_id)
```

```
join inventory using(film_id)
```

```
join rental using(inventory_id)
```

```
join payment using(rental_id)
```

```
group by concat(first_name,' ',last_name),actor.last_update;
```



The screenshot shows a database query result grid with the following columns: fullname, date(actor.last_update), and avg(amount). The results are sorted by fullname. The data is as follows:

fullname	date(actor.last_update)	avg(amount)
ADAM GRANT	2006-02-15	3.466868
ADAM HOPPER	2006-02-15	4.021549
AL GARLAND	2006-02-15	3.694600
ALAN DREYFUSS	2006-02-15	3.942891
ALBERT JOHANSSON	2006-02-15	4.227985
ALBERT NOLTE	2006-02-15	4.167462
ALEC WAYNE	2006-02-15	3.896565
ANGELA HUDSON	2006-02-15	4.113693
ANGELA WITHERSPOON	2006-02-15	3.997645
ANGELINA ASTAIRE	2006-02-15	3.880244
ANNE CRONYN	2006-02-15	4.417811
AUDREY BAILEY	2006-02-15	4.176216

-- Task 2 :

-- Task 2.1 : Is there any change in the actor's first name or last name?

```
select first_name,last_name from actor
```

```
order by 1,2;
```

Result Grid Filter Rows:		
	first_name	last_name
▶	ADAM	GRANT
	ADAM	HOPPER
	AL	GARLAND
	ALAN	DREYFUSS
	ALBERT	JOHANSSON
	ALBERT	NOLTE
	ALEC	WAYNE
	ANGELA	HUDSON
	ANGELA	WITHERSPOON
	ANGELINA	ASTAIRE
	ANNE	CRONYN
	AUDREY	BAILEY

actor 66 ▼

-- Task 2.2 : How many actors have the same first names and last names?

select concat(first_name,' ',last_name) as Full_name,count(*) from actor group by Full_name having count(Full_name)>1;

Result Grid Filter Rows: <input type="text"/> Exp		
	Full_name	count(*)
▶	SUSAN DAVIS	2

-- Interpretation : Only two actors have same first and last name.

-- Task 2.3 : How many actors have unique names? What is the count of these actors?

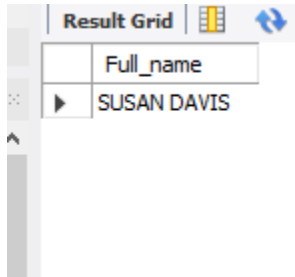
select count(*) from (select distinct(concat(first_name,' ',last_name)) as Full_name from actor) as mytab;

Result Grid Filb	
	count(*)
▶	199

-- Interpretation : Total 199 actors have unique names

-- Task 3 : Find the list of actors whose names are repeated and list of actors whose names are not repeated

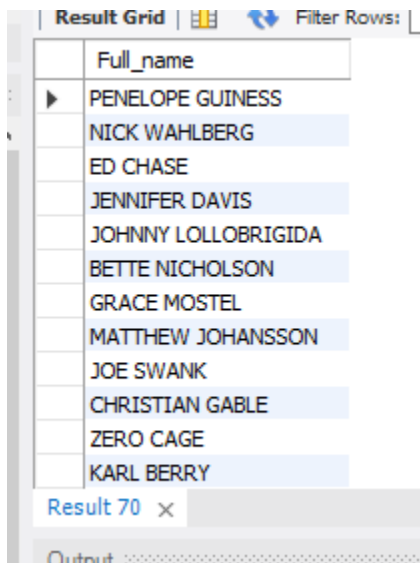
```
select concat(first_name,' ',last_name) as Full_name from actor group by Full_name having count(Full_name)>1;
```



The screenshot shows a 'Result Grid' with a single column header 'Full_name' and one data row containing the name 'SUSAN DAVIS'.

Full_name
SUSAN DAVIS

```
select distinct(concat(first_name,' ',last_name)) as Full_name from actor;
```



The screenshot shows a 'Result Grid' with a single column header 'Full_name' and 14 data rows listing distinct actor names. The names are: PENELOPE GUINESS, NICK WAHLBERG, ED CHASE, JENNIFER DAVIS, JOHNNY LOLLOBRIGIDA, BETTE NICHOLSON, GRACE MOSTEL, MATTHEW JOHANSSON, JOE SWANK, CHRISTIAN GABLE, ZERO CAGE, and KARL BERRY. The grid also shows 'Result 70' and an 'Output' section.

Full_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

/* Task 4 : The board needs to categorize the actors playing identity roles such as action,romance, horror and mystery. For this board members want to have detail overview of films based on actor's preference */

```
select concat(first_name,' ',last_name) as actor_name,category.name,count(category.name) from actor
inner join film_actor using(actor_id)
inner join film using (film_id)
inner join film_category using(film_id)
```

```

inner join category using(category_id)

where category.name in ('Action','Romance','Horror','Mystery')

group by category.name,actor_name

order by 3 desc;

```

Result Grid			
Filter Rows:		Export:	Wrap Cell Content:
	actor_name	name	count(category.name)
▶	JULIA MCQUEEN	Horror	7
▲	NATALIE HOPKINS	Action	6
	TOM MIRANDA	Horror	6
	SUSAN DAVIS	Action	5
	JON CHASE	Action	5
	SEAN GUINNESS	Action	5
	KIRSTEN AKROYD	Action	5
	AL GARLAND	Action	5
	HENRY BERRY	Horror	5
	VIVIEN BERGEN	Horror	5
	RIP CRAWFORD	Action	4
	SIDNEY CROWE	Action	4

/* Task 5 : The board wants to know various rating categories with their descriptions.

Determine which movies are suitable for kids, under 16 but under parent guidance & also restricted for all under 18 */

```

-- Option 1 :

create view movie_rating as

select rating,
case
when rating in('G') then 'GENERAL AUDIENCES'
when rating in('PG') then 'PARENTAL GUIDANCE SUGGESTED'
when rating in('PG-13') then 'PARENTS STRONGLY CAUTIONED'
when rating in('R') then 'RESTRICTED '
else 'NO ONE 17 AND UNDER ADMITTED' end as movie_type
from film

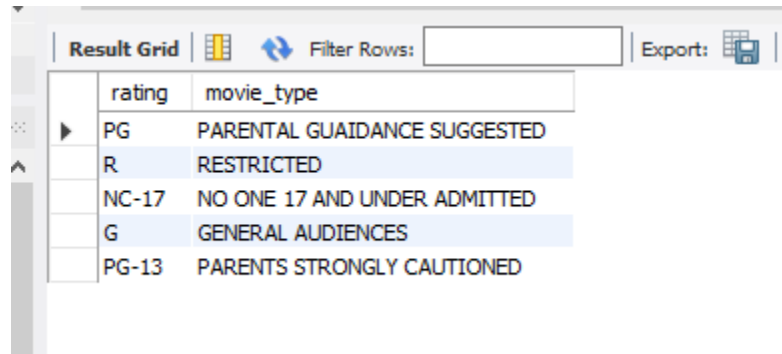
inner join film_category using(film_id)

```

```
inner join category using(category_id)
```

```
;
```

```
select * from movie_rating group by rating;
```



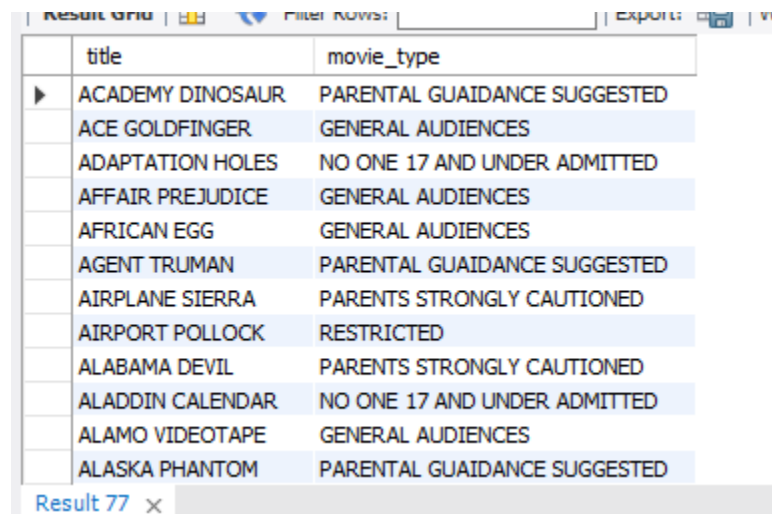
The screenshot shows a database query result grid with two columns: 'rating' and 'movie_type'. The results are as follows:

rating	movie_type
PG	PARENTAL GUIDANCE SUGGESTED
R	RESTRICTED
NC-17	NO ONE 17 AND UNDER ADMITTED
G	GENERAL AUDIENCES
PG-13	PARENTS STRONGLY CAUTIONED

```
select title,movie_type from film
```

```
inner join movie_rating using(rating)
```

```
group by title,movie_type;
```



The screenshot shows a database query result grid with two columns: 'title' and 'movie_type'. The results are as follows:

title	movie_type
ACADEMY DINOSAUR	PARENTAL GUIDANCE SUGGESTED
ACE GOLDFINGER	GENERAL AUDIENCES
ADAPTATION HOLES	NO ONE 17 AND UNDER ADMITTED
AFFAIR PREJUDICE	GENERAL AUDIENCES
AFRICAN EGG	GENERAL AUDIENCES
AGENT TRUMAN	PARENTAL GUIDANCE SUGGESTED
AIRPLANE SIERRA	PARENTS STRONGLY CAUTIONED
AIRPORT POLLOCK	RESTRICTED
ALABAMA DEVIL	PARENTS STRONGLY CAUTIONED
ALADDIN CALENDAR	NO ONE 17 AND UNDER ADMITTED
ALAMO VIDEOTAPE	GENERAL AUDIENCES
ALASKA PHANTOM	PARENTAL GUIDANCE SUGGESTED

```
select count(*),movie_type from movie_rating
```

```
group by movie_type;
```

Result Grid		
	count(*)	movie_type
▶	194	PARENTAL GUIDANCE SUGGESTED
	195	RESTRICTED
	210	NO ONE 17 AND UNDER ADMITTED
	178	GENERAL AUDIENCES
	223	PARENTS STRONGLY CAUTIONED

-- Option 2 :

```
select name, count(film.film_id) as count_of_movies from category
join film_category using(category_id)
join film using(film_id)
group by name;
```

Result Grid		
	name	count_of_movies
▶	Action	64
	Animation	66
	Children	60
	Classics	57
	Comedy	58
	Documentary	68
	Drama	62
	Family	69
	Foreign	73
	Games	61
	Horror	56
	Music	51

```
select name,title,rating,description from category
join film_category using(category_id)
join film using(film_id)
where rating in('r','nc-17') or name='children';
```

87

Result Grid | Filter Rows: | Export: | Wrap Cell Content: [↕](#)

	name	title	rating	description
▶	Action	AMERICAN CIRCUS	R	A Insightful Drama of a Girl And a Astronaut who must Face a Database Administrator in A Shark Tank
	Action	ANTITRUST TOMATOES	NC-17	A Fateful Yarn of a Womanizer And a Feminist who must Succumb a Database Administrator in Ancient India
	Action	ARK RIDGEMONT	NC-17	A Beautiful Yarn of a Pioneer And a Monkey who must Pursue a Explorer in The Sahara Desert
	Action	BULL SHAWSHANK	NC-17	A Fandful Drama of a Moose And a Squirrel who must Conquer a Pioneer in The Canadian Rockies
	Action	CADDYSHACK JEDI	NC-17	A Awe-Inspiring Epistle of a Woman And a Madman who must Fight a Robot in Soviet Georgia
	Action	CAMPUS REMEMBER	R	A Astounding Drama of a Crocodile And a Mad Cow who must Build a Robot in A Jet Boat
	Action	CLUELESS BUCKET	R	A Taut Tale of a Car And a Pioneer who must Conquer a Sumo Wrestler in An Abandoned Fun House
	Action	DANCES NONE	NC-17	A Insightful Reflection of a A Shark And a Dog who must Kill a Butler in An Abandoned Amusement Park
	Action	DARKO DORADO	NC-17	A Stunning Reflection of a Frisbee And a Husband who must Redeem a Dog in New Orleans
	Action	DEVIL DESIRE	R	A Beautiful Reflection of a Monkey And a Dentist who must Face a Database Administrator in Ancient Japan
	Action	DRAGON SQUAD	NC-17	A Taut Reflection of a Boy And a Waitress who must Outgun a Teacher in Ancient China

Result R1

-- Task 6

-- Task 6.1 : Figure out movie titles where replacement cost is upto \$9

select title,replacement_cost from film

where replacement_cost <=9;

Result Grid | Filter Rows: |

	title	replacement_cost
--	-------	------------------

-- Interpretation : No movies with replacement cost upto \$9

-- Task 6.2 : Get movie titles where replacement cost between \$15 and \$20

select title,replacement_cost from film

where replacement_cost between 15 and 20;

	title	replacement_cost
▶	ADAPTATION HOLES	18.99
	AGENT TRUMAN	17.99
	AIRPORT POLLOCK	15.99
	ALAMO VIDEOTAPE	16.99
	AMERICAN CIRCUS	17.99
	ANALYZE HOOSIERS	19.99
	ANGELS LIFE	15.99
	ANNIE IDENTITY	15.99
	ANTHEM LUKE	16.99
	APACHE DIVINE	16.99
	APOLLO TEEN	15.99
	ARSENIC INDEPEND...	17.99

film 83 ×

Output

-- Task 6.3: Find movie titles with highest replacement cost but lowest rental cost

```
select title,replacement_cost,rental_rate from film
where replacement_cost=(select max(replacement_cost)from film)
and rental_rate=(select min(rental_rate) from film);
```

	title	replacement_cost	rental_rate
▶	ARABIA DOGMA	29.99	0.99
	BALLROOM MOCKINGBIRD	29.99	0.99
	BONNIE HOLOCAUST	29.99	0.99
	CLOCKWORK PARADISE	29.99	0.99
	CLYDE THEORY	29.99	0.99
	CRUELTY UNFORGIVEN	29.99	0.99
	EARTH VISION	29.99	0.99
	EVERYONE CRAFT	29.99	0.99
	FEUD FROGMEN	29.99	0.99
	GILMORE BOILED	29.99	0.99
	GOLDFINGER SENSIBILITY	29.99	0.99
	GRAFFITI LOVE	29.99	0.99

film 84 ×

-- Task 7 : Find list of all films along with the number of actors listed for each movie

```
select title,count(actor_id) from film
inner join film_actor using(film_id)
group by title;
```


Result Grid | Filter Rows:

title	count(actor_id)
ACADEMY DINOSAUR	10
ACE GOLDFINGER	5
ADAPTATION HOLES	5
AFFAIR PREJUDICE	5
AFRICAN EGG	5
AGENT TRUMAN	7
AIRPLANE SIERRA	5
AIRPORT POLLOCK	4
ALABAMA DEVIL	9
ALADDIN CALENDAR	8
ALAMO VIDEOTAPE	4
ALASKA PHANTOM	7

Result 93 x

-- Task 8 : Display titles of the movie starting with k or q

select title from film

where title like 'k%' or title like 'Q%';

<

Result Grid | Filter Rows:

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

film 85 x

-- Task 9 : Display all actors appeared in movie 'Agent Truman'

select distinct(concat(first_name,' ',last_name)) as actor_name from actor

inner join film_actor using(actor_id)

inner join film using(film_id)

where title='Agent Truman';

Result Grid		Filter Rows:
	actor_name	
▶	KIRSTEN PALTROW	
	SANDRA KILMER	
	JAYNE NEESON	
	WARREN NOLTE	
	MORGAN WILLIAMS	
	KENNETH HOFFMAN	
	REESE WEST	

-- Task 10 : Identify all movies categorizes as family films

select title from film

inner join film_category using(film_id)

inner join category using(category_id)

where name='Family';

Result Grid		Filter Rows:
	title	
▶	AFRICAN EGG	
	APACHE DIVINE	
	ATLANTIS CAUSE	
	BAKED CLEOPATRA	
	BANG KWAI	
	BEDAZZLED MARRIED	
	BILKO ANONYMOUS	
	BLANKET BEVERLY	
	BLOOD ARGONAUTS	
	BLUES INSTINCT	
	BRAVEHEART HUMAN	
	CHASING FIGHT	

Result 95 x

-- Task 11 : Display the most fruently rented movies in descending order

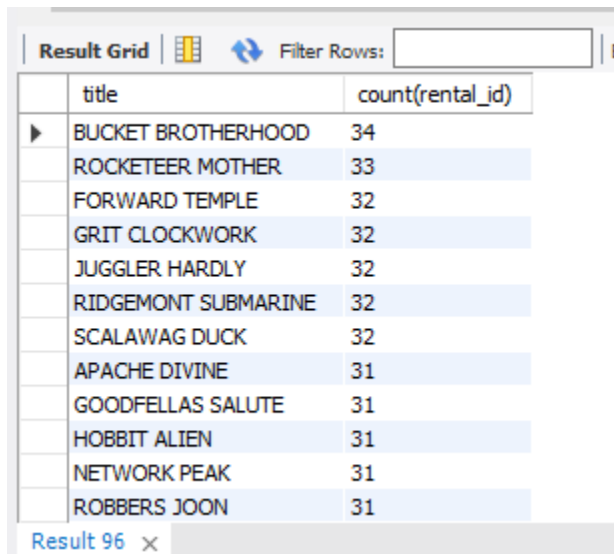
select title,count(rental_id) from film

inner join inventory using(film_id)

inner join rental using(inventory_id)

group by title

order by 2 desc;



Result Grid | Filter Rows:

	title	count(rental_id)
▶	BUCKET BROTHERHOOD	34
	ROCKETEER MOTHER	33
	FORWARD TEMPLE	32
	GRIT CLOCKWORK	32
	JUGGLER HARDLY	32
	RIDGEMONT SUBMARINE	32
	SCALAWAG DUCK	32
	APACHE DIVINE	31
	GOODFELLAS SALUTE	31
	HOBBIT ALIEN	31
	NETWORK PEAK	31
	ROBBERS JOON	31

Result 96 ×

-- Task 12 : In how many film categories aveage difference between replacement cost and rental cost is greater than \$15?

```
select count(*) from (select name, (avg(replacement_cost-rental_rate)) as x from film
```

```
inner join film_category using(film_id)
```

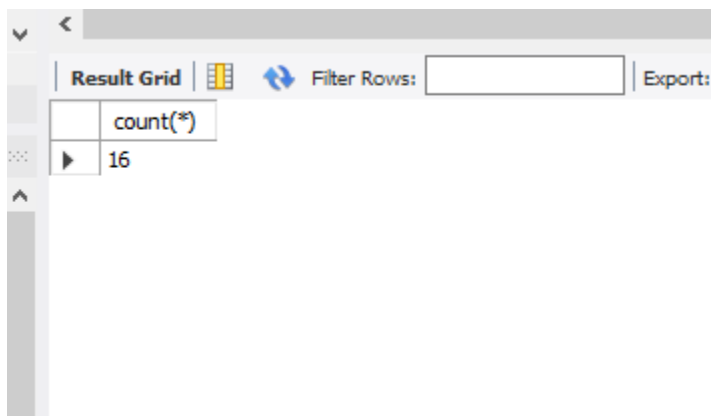
```
inner join category using(category_id)
```

```
group by name
```

```
having (avg(replacement_cost-rental_rate))>15)
```

```
as mytab
```

```
;
```

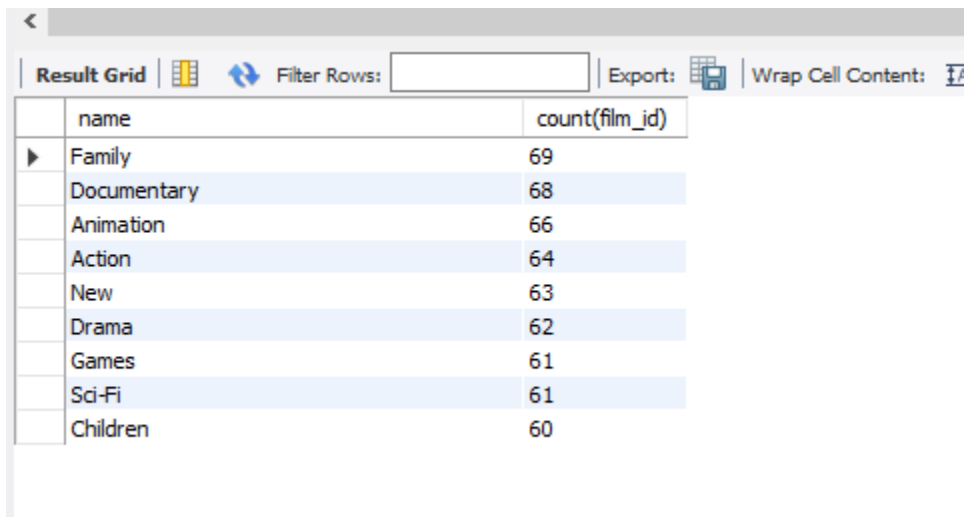


Result Grid | Filter Rows: | Export:

	count(*)
▶	16

/* Task 13 : List names of categories having 60 to 70 films.List name of these categories and number of films per category sorted by number of films */

```
select name,count(film_id) from category  
inner join film_category using(category_id)  
group by name  
having count(film_id) between 60 and 70  
order by 2 desc;
```



The screenshot shows a database query result grid. At the top, there is a toolbar with a back arrow, a 'Result Grid' button, a 'Filter Rows' input field, an 'Export' button, and a 'Wrap Cell Content' button. Below the toolbar is a table with two columns: 'name' and 'count(film_id)'. The table contains ten rows of data, sorted by the count in descending order. The categories and their respective film counts are: Family (69), Documentary (68), Animation (66), Action (64), New (63), Drama (62), Games (61), Sci-Fi (61), and Children (60).

name	count(film_id)
Family	69
Documentary	68
Animation	66
Action	64
New	63
Drama	62
Games	61
Sci-Fi	61
Children	60