Exercise 0 *(0.sql)*

Write a DDL statement to create a new table FilmActor where,

ActorID smallint,

FilmID smallint,

Active bit not null,

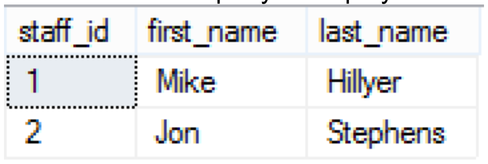
LastUpdated smalldatetime not null

JoinedDate smalldatetime not null

- Add to this table some contraints:
    - UserID and FilmID is a primary key
    - JoinedDate, LastUpdated cannot be greater than today
    - JoinedDate must be less than or equals to LastUpdate
    - ActorID is a foreign key and references to Actor table
    - FilmID is a foreign key and references to Film table
- Insert to this table a new record where
    - ActorID is actor whose first name is “ALEC”
    - He joined the film title “AFRICAN EGG”
    - LastUpdate is 5 days ago
    - He joined the film by Dec 12 2005
    - Active is 1

Exercise 1 *(1.sql)*

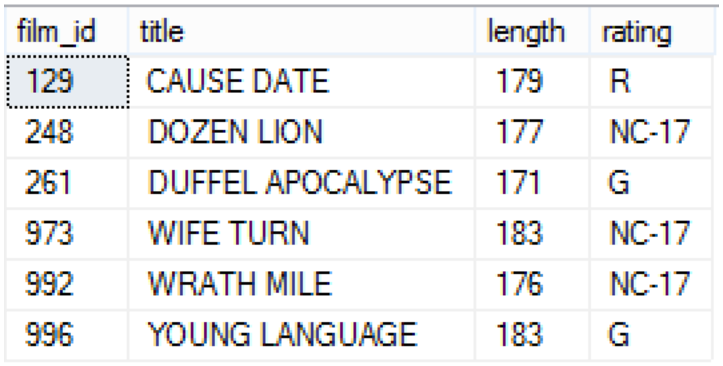
Write a SELECT query to display all active staffs, order by staff_id as ascending:



staff_id	first_name	last_name
1	Mike	Hillyer
2	Jon	Stephens

Exercise 2 *(2.sql)*

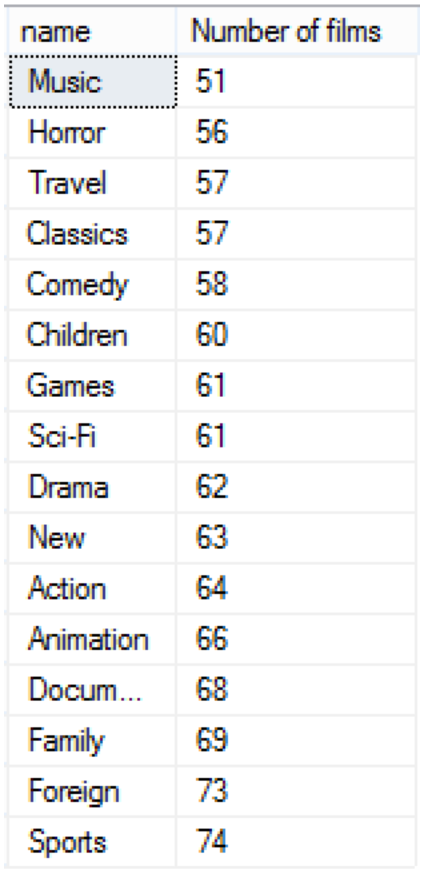
Write a SELECT query to display all films which type is “Documentary” and have length greater than or equals to 170, order by film_id



film_id	title	length	rating
129	CAUSE DATE	179	R
248	DOZEN LION	177	NC-17
261	DUFFEL APOCALYPSE	171	G
973	WIFE TURN	183	NC-17
992	WRATH MILE	176	NC-17
996	YOUNG LANGUAGE	183	G

Exercise 3 *(3.sql)*

Write a query to display number of films for each category, Order by number of films as ascending.

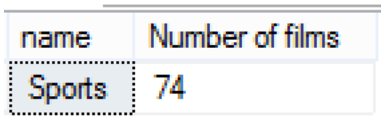


name	Number of films
Music	51
Horror	56
Travel	57
Classics	57
Comedy	58
Children	60
Games	61
Sci-Fi	61
Drama	62
New	63
Action	64
Animation	66
Docum...	68
Family	69
Foreign	73
Sports	74

This list has 42 records in total.

Exercise 4 *(4.sql)*

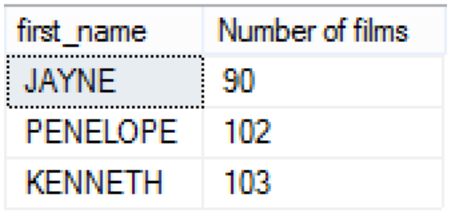
Write a query to display the category which have maximum number of films in it. Order by category name as ascending.



name	Number of films
Sports	74

Exercise 5 *(5.sql)*

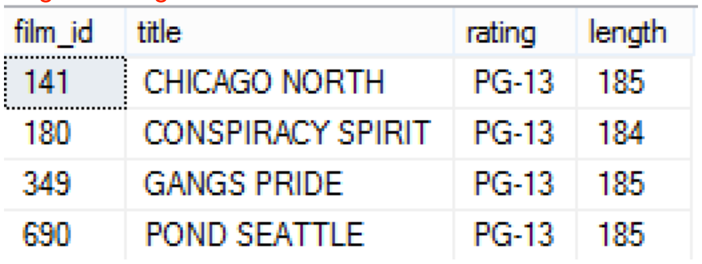
Write a query to display number of films for each actor whose joined more than 90 films, order by number of films as ascending.



first_name	Number of films
JAYNE	90
PENELOPE	102
KENNETH	103

Exercise 6 *(6.sql)*

Write a query to display information of films which have length >= the length of film “THEORY MERMAID” and have same rating as rating of film “THEORY MERMAID”

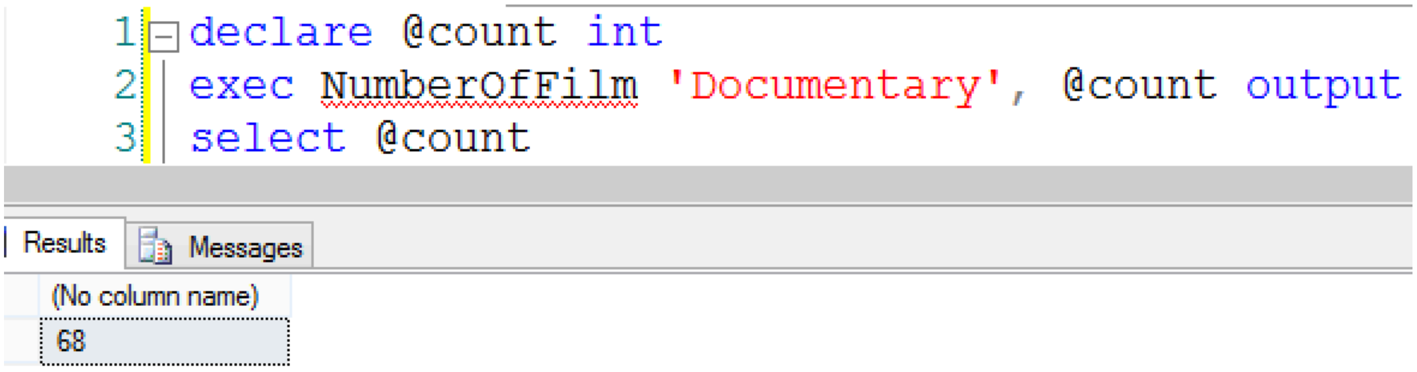


film_id	title	rating	length
141	CHICAGO NORTH	PG-13	185
180	CONSPIRACY SPIRIT	PG-13	184
349	GANGS PRIDE	PG-13	185
690	POND SEATTLE	PG-13	185

Exercise 7 *(7.sql)*

Write a store **NumberOfFilm(@catName varchar(25), @count int output)**, the store is used to count number of films of given category name, the result must be set to @count

If you test your above store as below sample, you can get the those output.



```
declare @count int
exec NumberOfFilm 'Documentary', @count output
select @count
```

(No column name)
68

Exercise 8 *(8.sql)*

Write a trigger to make sure that whenever users insert a new film to Film table, the title of film cannot be duplicated.

Exercise 9 ***(9.sql)***

Students come to library to borrow books, a Student (id, name) can borrow many books, a book (id, name) is only brorrowed from a student, all students must borrow books for their study, some books are not borrowed from any student. They recored the date and some note whenever a student borrow a book.