# Faculty of Computing

**CS220: Database Systems**

**Class: BESE-13AB**

# Lab 08: SQL Queries (Subqueries)

# Date: 13 November, 2023

# Time: 10:00-01:00 & 02:00-05:00

# Instructor: Dr. Bilal Ali & Dr. Shah Khalid

# Lab Engineer: Sundas Dawood

**Name:** Aimen Munawar

**Class:** BESE-13-A

**CMS ID:** 415867

# Lab 08: SQL Queries (Subqueries)

**Introduction**

The sub query in SQL is a query within query. A sub- query is composed of an outer query and an inner query.

**Objectives**

After completing this lab, you should be able to do the following:

* Understand and practice the single row sub queries
* Understand and practice the multiple row sub queries

**Tools/Software Requirement**

* MySQL Community Server
* MySQL Workbench

**Helping Material**

<https://www.w3resource.com/sql/subqueries/understanding-sql-subqueries.php>

**Description:**

For example, you want to see all the employees whose salary is above average salary. For this you have to first compute the average salary using AVG function and then compare employees’ salaries with this computed salary. This is possible using subquery. Here the sub query will first compute the average salary and then main query will execute.

**Select \***

**from emp**

**where sal >**

**(select avg(sal)**

**from emp);**

Similarly we want to see the name and empno of that employee whose salary is maximum.  
**Select \*   
from emp   
where sal = (select max(sal)   
 from emp);**

We want to see how many employees are there whose salary is above average.

**Select count(\*)**

**from emp**

**where sal >**

**(select avg(sal)**

**from emp);**

We want to see those employees who are working in 'DALLAS'. Remember emp and dept are joined on deptno and loc column is in the dept table. Assuming that wherever the department is located the employee is working in that city.  
**Select \*   
from emp**

**where deptno IN**

**(select deptno**

**from dept**

**where loc='DALLAS');**

**Lab Tasks**

On the basis of the knowledge of the sub-queries, attempt the following questions using the Subqueries concept. The information needs can also be met through join operations; however, you need to practice the subqueries. Try to implement subqueries concept as much as possible. You can use join operation in your queries if required.

Write equivalent join queries for at least two queries.

Write SQL queries for the following information needs using subquery approach.

1. **Number of customers who never rented BREAKING HOME.**

**Code:**

select count(customer\_id) as customersWhoNeverRentedBreakingHome

from customer

where customer\_id not in (

select customer\_id

from rental

where inventory\_id in (

select inventory\_id

from inventory

where film\_id = (

select film\_id

from film

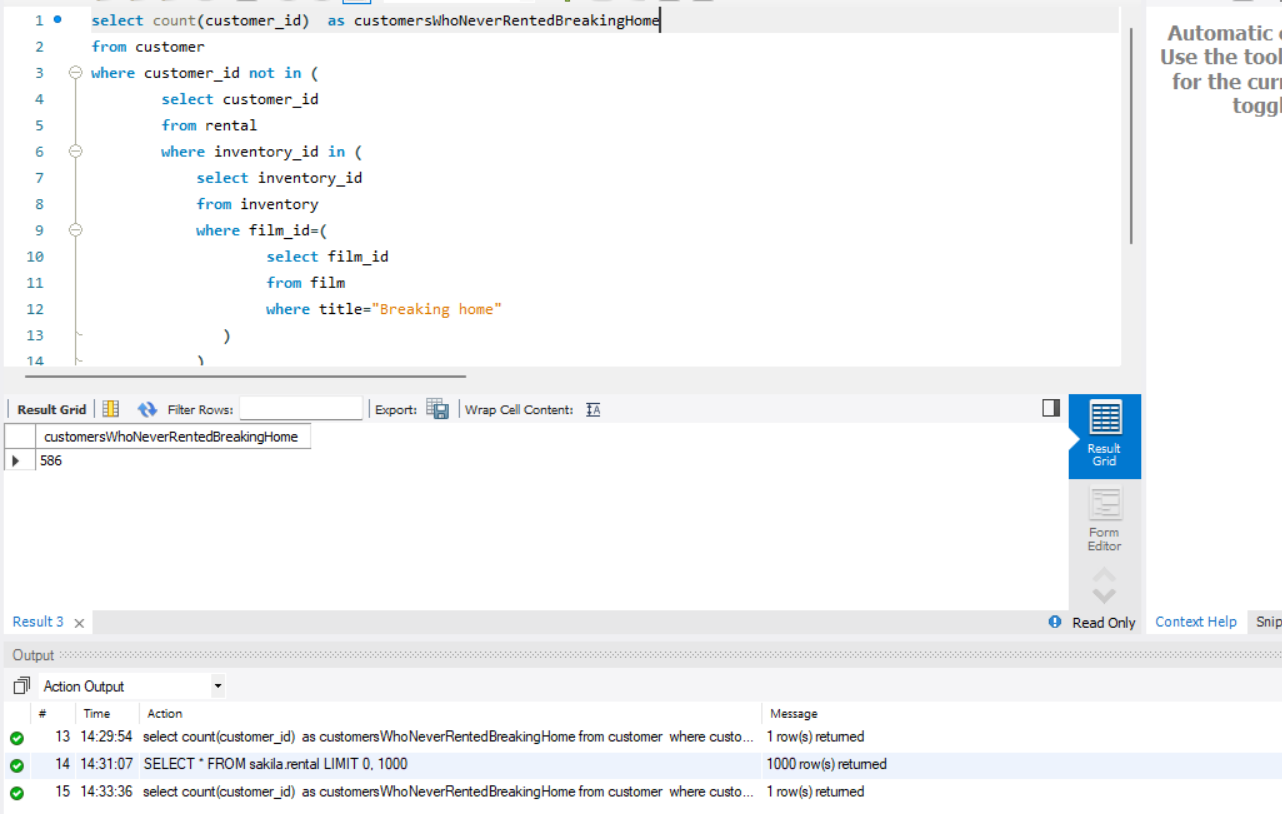
where title="Breaking home"

)

)

);

**Output:**

****

1. List accumulative replacement cost of the movies that were never rented.

**Code:**

select sum(replacement\_cost) as accumulativeReplacementCost

from film

where film\_id not in(

select film\_id

from inventory

where inventory\_id in (select inventory\_id from rental)

);

**Output:**



1. Find customers who are associated with the City Abu Dhabi or Aden.

**Code:**

select first\_name,last\_name

from customer

where address\_id in (

select address\_id

from address

where city\_id in (

select city\_id

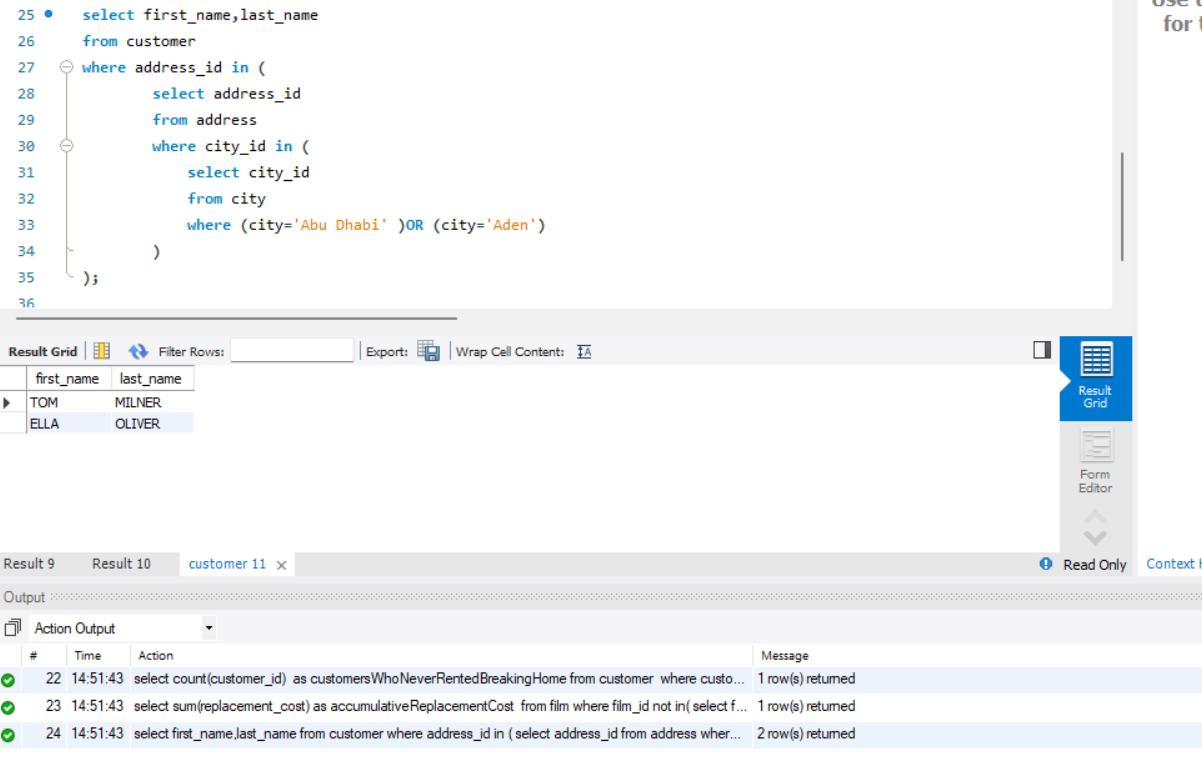
from city

where (city='Abu Dhabi' )OR (city='Aden')

)

);

**Output:**



1. What category does the movie COMA HEAD belong to?

**Code:**

select name,category\_id

from category

where category\_id= (

select category\_id

from film\_category

where film\_id= (

select film\_id

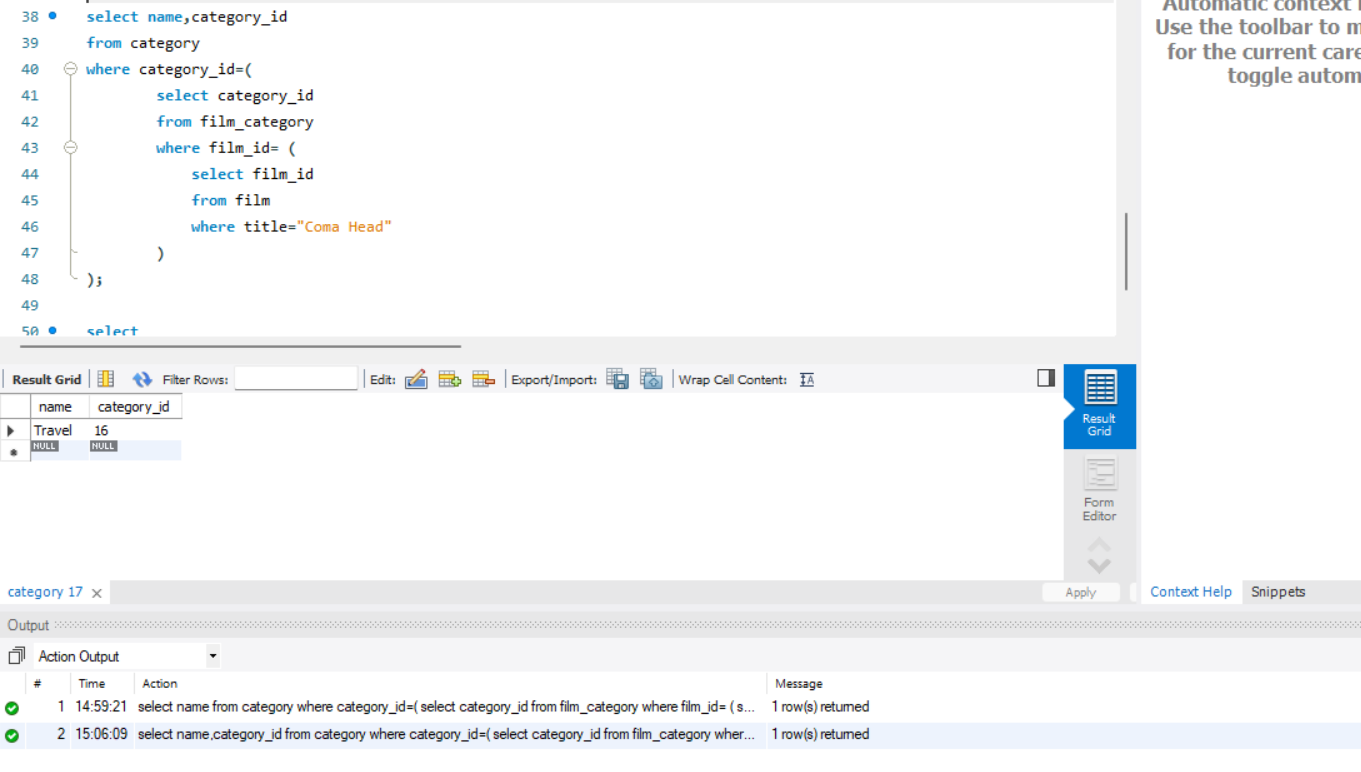
from film

where title="Coma Head"

)

);

**Output:**



**Using Join:**

**Code:**

select cc.name, cc.category\_id

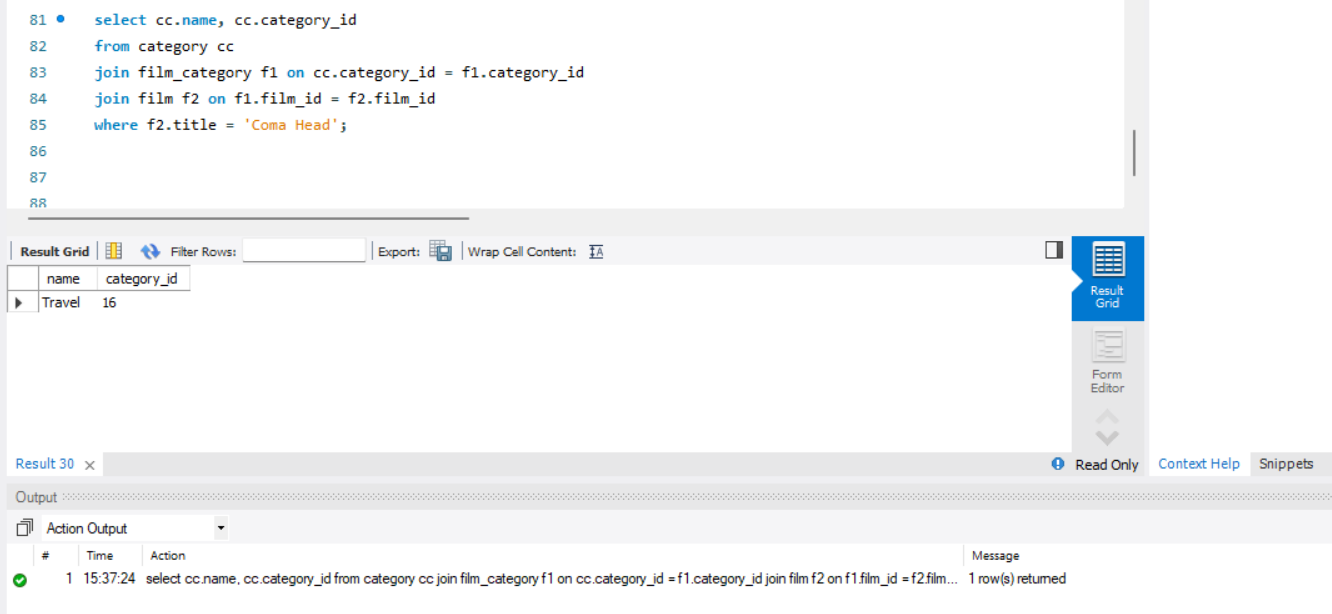
from category cc

join film\_category f1 on cc.category\_id = f1.category\_id

join film f2 on f1.film\_id = f2.film\_id

where f2.title = 'Coma Head';

**Output:**



1. Find the movies whose rental duration is less than the rental duration of movie ‘AFRICAN EGG’.

**Code:**

select title,rental\_duration

from film

where rental\_duration < (

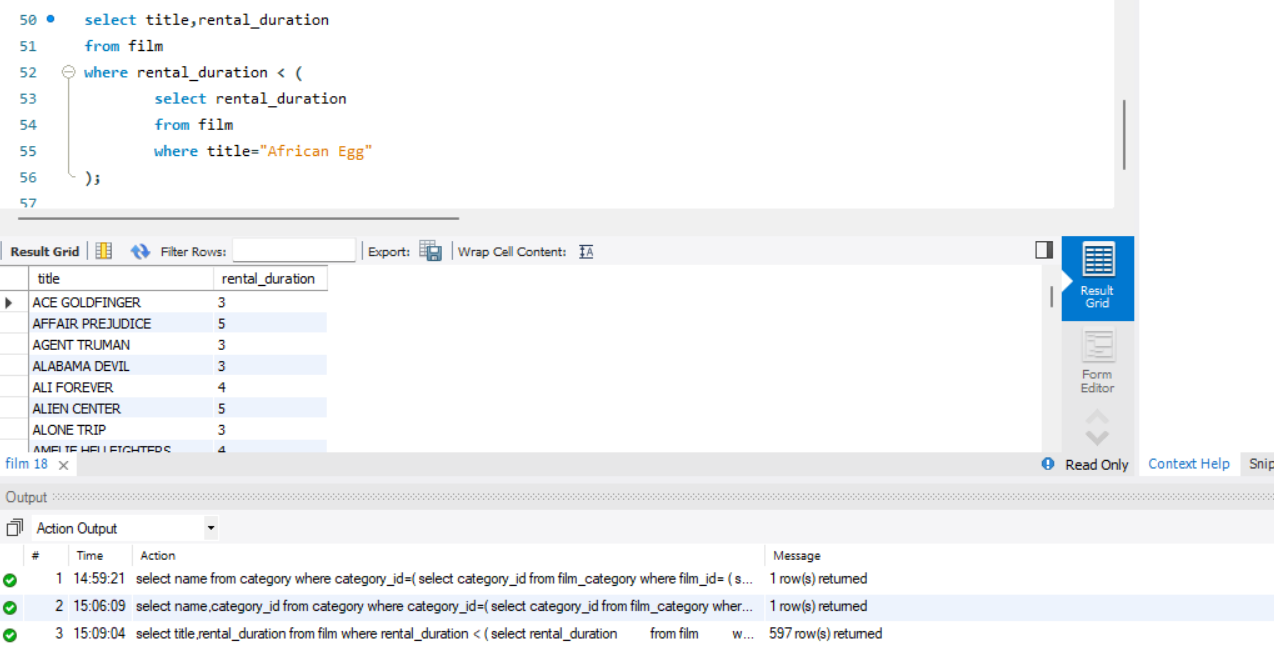
select rental\_duration

from film

where title="African Egg"

);

**Output:**

****

**Using Join:**

**Code:**

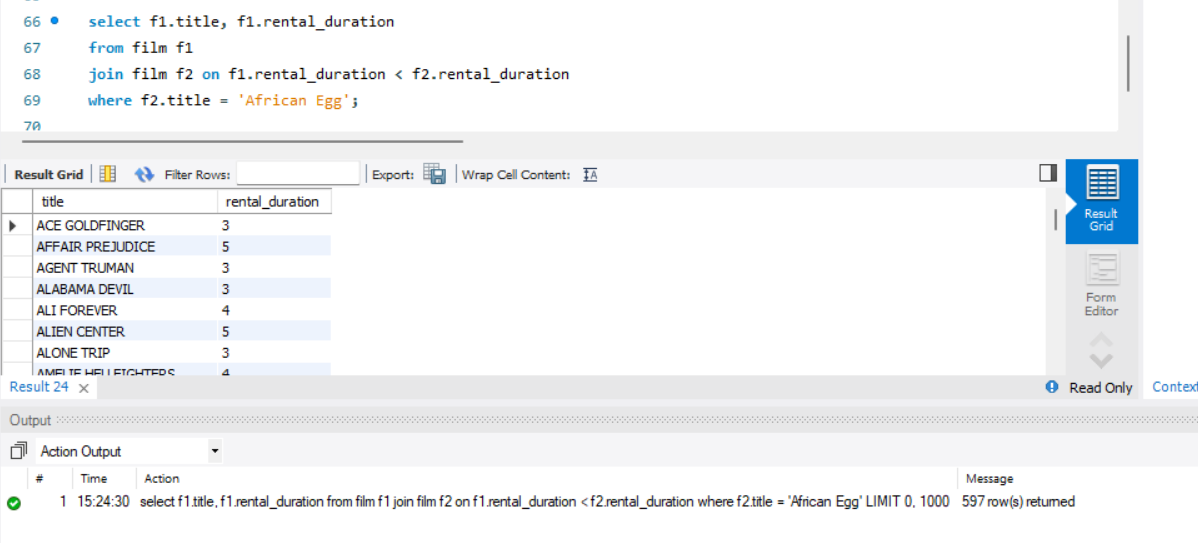
select f1.title, f1.rental\_duration

from film f1

join film f2 on f1.rental\_duration < f2.rental\_duration

where f2.title = 'African Egg';

**Output:**

****

1. Find those films whose category id is greater than every category id of available films.

**Code:**

select title

from film

where film\_id in(

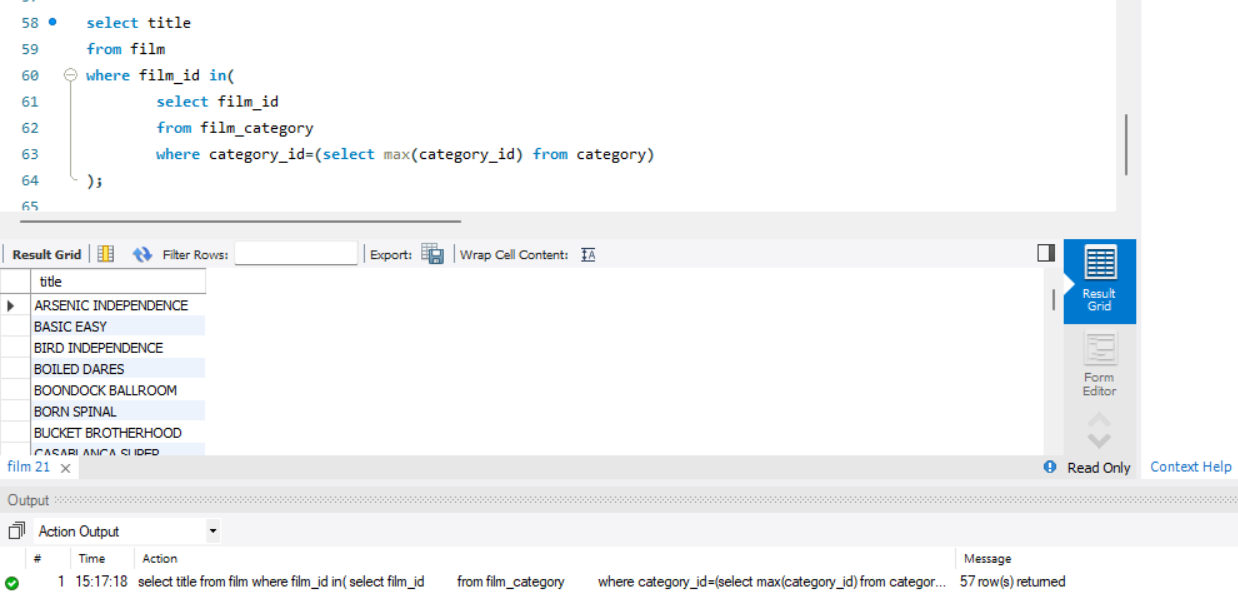
select film\_id

from film\_category

where category\_id=(select max(category\_id) from category)

);

**Output:**



**Using Join:**

**Code:**

select f1.title

from film f1

join film\_category f2 on f1.film\_id = f2.film\_id

join category cc on f2.category\_id = cc.category\_id

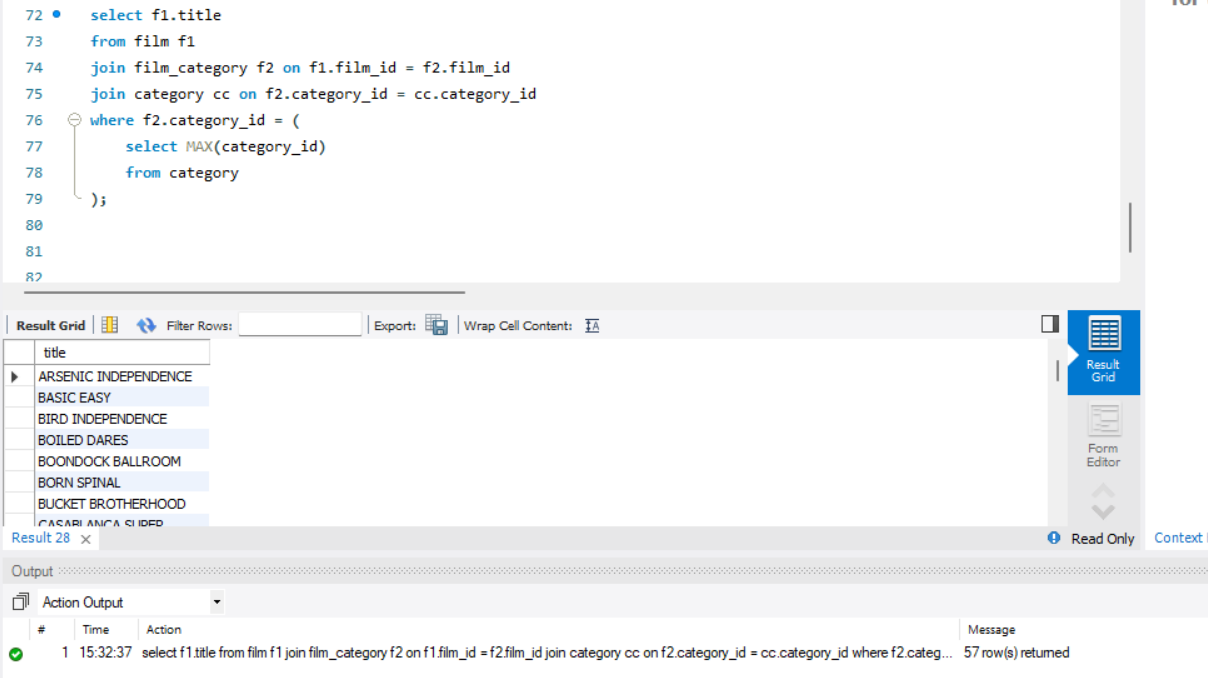
where f2.category\_id = (

select MAX(category\_id)

from category

);

**Output:**

****

**Deliverables**

Save all queries and their results in the word document that you are executing along with the snapshots of the results. Upload this document to LMS.

# 