

Database Design and Implementation

Manivassakam M

Assignment 5

a) List the actors (firstName, lastName) who acted in more than 25 movies.

Note: Also, show the count of movies against each actor

```
SELECT first_name, last_name, COUNT(film_actor.actor_id) from actor  
INNER JOIN film_actor ON actor.actor_id = film_actor.actor_id  
GROUP BY actor.first_name, actor.last_name, actor.actor_id  
HAVING COUNT(film_actor.actor_id) >= '25';
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_ls
- sales_by_film_category

Management Schemas

Information

Schema: sakila

assign5_dbcode*

sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
10 USE sakila;
11 #shows the list of all tables in the database
12 SHOW TABLES;
13
14 #List the actors (firstName, lastName) who acted in more than 25 movies.
15 # Note: Also show the count of movies against each actor
16
17 SELECT first_name, last_name, COUNT(film_actor.actor_id)from actor
18 INNER JOIN film_actor ON actor.actor_id = film_actor.actor_id
19 GROUP BY actor.first_name, actor.last_name, actor.actor_id
20 HAVING COUNT(film_actor.actor_id) >= '25';
```

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

| first_name | last_name | COUNT(film_actor.actor_id) |
|------------|-------------|----------------------------|
| AL | GARLAND | 26 |
| ALAN | DREYFUSS | 27 |
| ALBERT | JOHANSSON | 33 |
| ALBERT | NOLTE | 31 |
| ALEC | WAYNE | 29 |
| ANGELA | HUDSON | 34 |
| ANGELA | WITHERSPOON | 35 |
| ANGELINA | ASTARE | 31 |
| ANNE | CRONYN | 27 |
| AUDREY | BAILEY | 27 |
| AUDREY | OLIVIER | 25 |
| BELA | WALKEN | 30 |
| BEN | WILLIS | 33 |
| BOB | FAWCETT | 25 |
| BURT | DUKAKIS | 29 |
| RANDI | WIFGR | 33 |

Result 57 ×

Output

Action Output

Time Action

Message Duration / Fetch

Ask me anything

9:16 PM 11/5/2016

b) List the actors who have worked in the German language movies.

```
SELECT first_name, last_name, language.name from actor
INNER JOIN film_actor ON actor.actor_id = film_actor.actor_id
INNER JOIN film ON film_actor.film_id = film.film_id
INNER JOIN language ON film.language_id = language.language_id
WHERE language.name = 'German';
```

The screenshot shows the MySQL Workbench interface with the following details:

- File Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Toolbar:** Includes icons for New Connection, Open Connection, Save, Print, Copy, Paste, Find, Replace, and others.
- Navigator:** Shows the database schema for the sakila database, including Schemas (language, payment, rental, staff, store) and Views (actor_info, actors_portfolio, film_list, nicer_but_slower_film_js, sales_by_film_category).
- SQL Editor:** Contains the SQL query provided in the question, which retrieves actors who have worked in German language movies. A note below the query says: "# Note: Please execute the below SQL before answering this question."
- Result Grid:** Displays the results of the query in a tabular format. The columns are first_name, last_name, and name. The data shows 25 actors whose names are in German.
- Right Panel:** SQLAdditions panel with a note about automatic context help being disabled.
- Bottom Status Bar:** Shows the session status (Object Info, Session), connection information (Action Output, #, Time, Action), message area, duration/fetch time (9:21 PM, 11/5/2016), and system tray icons.

| first_name | last_name | name |
|------------|-----------|--------|
| PENLOPE | GUINNESS | German |
| CHRISTIAN | GABLE | German |
| LUCILLE | TRACY | German |
| SANDRA | PECK | German |
| JOHNNY | CAGE | German |
| GARY | PHOENIX | German |
| MENA | TEMPLE | German |
| KENNETH | PESCI | German |
| WARREN | NOLTE | German |
| OPRAH | KILMER | German |
| DEBBIE | AKROYD | German |
| ROCK | DUKAKIS | German |
| CUBA | BIRCH | German |
| MERYL | ALLEN | German |
| MARY | KEITEL | German |

Note: Please execute the below SQL before answering this question.

SET SQL_SAFE_UPDATES=0;

UPDATE film SET language_id=6 WHERE title LIKE "%ACADEMY%";

c) List the actors who acted in horror movies.

```
SELECT first_name,last_name,category.name from actor
INNER JOIN film_actor ON actor.actor_id = film_actor.actor_id
INNER JOIN film ON film_actor.film_id = film.film_id
INNER JOIN film_category ON film.film_id = film_category.film_id
INNER JOIN category ON film_category.category_id = category.category_id
WHERE category.name ='Horror';
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

Management Schemas

Information

Schema: sakila

assign5_dbcode × sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

c) List the actors who acted in horror movies.

```
32
33
34 • SELECT first_name, last_name, category.name from actor
35     INNER JOIN film_actor ON actor.actor_id = film_actor.actor_id
36     INNER JOIN film ON film_actor.film_id = film.film_id
37     INNER JOIN film_category ON film.film_id = film_category.film_id
38     INNER JOIN category ON film_category.category_id = category.category_id
39 WHERE category.name = 'Horror';
```

d) List all customers who rented more than 3 horror movies.
Note: Show the count of movies against each actor in the result set.

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

| first_name | last_name | name |
|------------|-----------|--------|
| BOB | FAWCETT | Horror |
| MINNIE | ZELLWEGER | Horror |
| SEAN | GUINNESS | Horror |
| CHRIS | DEPP | Horror |
| JODIE | DEGENERES | Horror |
| SCARLETT | DAMON | Horror |
| KENNETH | PESCI | Horror |
| FAY | WINSLET | Horror |
| OPRAH | KILMER | Horror |
| FAY | KILMER | Horror |
| GENE | WILLIS | Horror |
| SUSAN | DAVIS | Horror |
| LUCILLE | DEE | Horror |
| CHRISTIAN | GABLE | Horror |
| ELVIS | MARX | Horror |
| RITA | CRAWFORD | Horror |

Result 99 ×

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

Ask me anything

9:24 PM
11/5/2016

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid Form Editor Field Types Query Stats Execution Plan

Read Only Context Help Snippets

d) List all customers who rented more than 3 horror movies.
Note: Show the count of movies against each actor in the result set.

```
SELECT first_name,last_name,COUNT(*) as cnt from customer
INNER JOIN rental ON customer.customer_id = rental.customer_id
INNER JOIN inventory ON rental.inventory_id = inventory.inventory_id
INNER JOIN film_category ON inventory.film_id = film_category.film_id
INNER JOIN category ON film_category.category_id = category.category_id
WHERE category.name = "Horror"
GROUP BY rental.customer_id HAVING cnt > 3;
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

Management Schemas

Information

Schema: sakila

assign5_dbcode x sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
41 # d) List all customers who rented more than 3 horror movies.  
42 # Note: Show the count of movies against each actor in the result set.  
43 • SELECT first_name,last_name,COUNT(*) as cnt from customer  
44 INNER JOIN rental ON customer.customer_id = rental.customer_id  
45 INNER JOIN inventory ON rental.inventory_id = inventory.inventory_id  
46 INNER JOIN film_category ON inventory.film_id = film_category.film_id  
47 INNER JOIN category ON film_category.category_id = category.category_id  
48 WHERE category.name = "Horror"  
49 GROUP BY rental.customer_id HAVING cnt > 3;  
50  
51 # e) List all customers who rented the movie which starred SCARLETT BENING
```

Result Grid

| first_name | last_name | cnt |
|------------|-----------|-----|
| NANCY | THOMAS | 5 |
| THERESA | WATSON | 4 |
| RACHEL | BARNES | 4 |
| ANDREA | HENDERSON | 4 |
| TINA | SIMMONS | 4 |
| CONNIE | WALLACE | 4 |
| THELMA | MURRAY | 5 |
| EMMA | BOYD | 5 |
| ELEANOR | HUNT | 4 |
| BERTHA | FERGUSON | 4 |
| REGINA | BERRY | 4 |
| ANA | BRADLEY | 5 |
| SUE | PETERS | 4 |
| ROSEMARY | SCHMIDT | 5 |
| NINA | SOTO | 4 |
| SCRINTA | GREGORY | 4 |

Result 100 x

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

Ask me anything

9:25 PM 11/5/2016

e) List all customers who rented the movie which starred SCARLETT BENING

```
SELECT customer.first_name, customer.last_name, actor.first_name, actor.last_name FROM customer
INNER JOIN rental ON customer.customer_id = rental.customer_id
INNER JOIN inventory ON rental.inventory_id = inventory.inventory_id
INNER JOIN film_category ON inventory.film_id = film_category.film_id
INNER JOIN film_actor ON film_category.film_id = film_actor.film_id
INNER JOIN actor ON film_actor.actor_id = actor.actor_id
WHERE actor.first_name = 'SCARLETT' AND actor.last_name = 'BENING';
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

assign5_dbcode

sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

e) List all customers who rented the movie which starred SCARLETT BENING

```
SELECT customer.first_name, customer.last_name, actor.first_name, actor.last_name FROM customer
INNER JOIN rental ON customer.customer_id = rental.customer_id
INNER JOIN inventory ON rental.inventory_id = inventory.inventory_id
INNER JOIN film_category ON inventory.film_id = film_category.film_id
INNER JOIN film_actor ON film_category.film_id = film_actor.film_id
INNER JOIN actor ON film_actor.actor_id = actor.actor_id
WHERE actor.first_name = 'SCARLETT' AND actor.last_name = 'BENING';
```

f) Which customers residing at postal code 62703 rented movies that were Documentaries.

Result Grid

| first_name | last_name | first_name | last_name |
|------------|-----------|------------|-----------|
| CHERYL | MURPHY | SCARLETT | BENING |
| WALTER | PERRYMAN | SCARLETT | BENING |
| MARY | SMITH | SCARLETT | BENING |
| MARJORIE | TUCKER | SCARLETT | BENING |
| ANGELA | HERNANDEZ | SCARLETT | BENING |
| CRAIG | MORRELL | SCARLETT | BENING |
| JORDAN | ARCHULETA | SCARLETT | BENING |
| DONNA | THOMPSON | SCARLETT | BENING |
| ZACHARY | HITE | SCARLETT | BENING |
| RHONDA | KENNEDY | SCARLETT | BENING |
| IAN | STILL | SCARLETT | BENING |
| TERRY | CARLSON | SCARLETT | BENING |
| ELIZABETH | BROWN | SCARLETT | BENING |
| MILTON | HOWLAND | SCARLETT | BENING |
| VELMA | LUCAS | SCARLETT | BENING |
| CLIFF | TIPTON | SCARLETT | BENING |

Result 101

Output

Action Output

Object Info Session

Ask me anything

Message

Duration / Fetch

9:26 PM
11/5/2016

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid

Form Editor

Field Types

Query Stats

Execution Plan

Read Only Context Help Snippets

f) Which customers residing at postal code 62703 rented movies that were Documentaries.

```
SELECT customer.first_name, customer.last_name, address.postal_code, category.name FROM customer
INNER JOIN rental ON customer.customer_id = rental.customer_id
INNER JOIN inventory ON rental.inventory_id = inventory.inventory_id
INNER JOIN film_category ON inventory.film_id = film_category.film_id
INNER JOIN category ON film_category.category_id = category.category_id
INNER JOIN address ON customer.address_id = address.address_id
WHERE address.postal_code = '62703' AND category.name = 'Documentary';
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

assign5_dbcode × sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

f) Which customers residing at postal code 62703 rented movies that were Documentaries.

```
SELECT customer.first_name, customer.last_name, address.postal_code, category.name FROM customer
INNER JOIN rental ON customer.customer_id = rental.customer_id
INNER JOIN inventory ON rental.inventory_id = inventory.inventory_id
INNER JOIN film_category ON inventory.film_id = film_category.film_id
INNER JOIN category ON film_category.category_id = category.category_id
INNER JOIN address ON customer.address_id = address.address_id
WHERE address.postal_code = '62703' AND category.name = 'Documentary';
```

g) Find all the addresses where the second address line is not empty (i.e., contains some text), and return these second addresses

Result Grid

| first_name | last_name | postal_code | name |
|------------|-----------|-------------|-------------|
| ANDY | VANHORN | 62703 | Documentary |

Result 102 ×

Output

Action Output

Object Info Session

Ask me anything

Message

Duration / Fetch

9:28 PM
11/5/2016

MySQL Workbench interface showing a query editor with SQL code for selecting customers from the sakila database who rented movies from postal code 62703 and categorized as Documentaries. The results are displayed in a grid, showing one row for Andy Vanhorn. The interface includes a Navigator pane, a SQL editor with syntax highlighting, and various toolbars and panes for managing databases and queries.

g) Find all the addresses where the second address line is not empty (i.e., contains some text), and return these second addresses sorted.

SELECT * FROM address where address2 != null;

The screenshot shows the MySQL Workbench interface. The query editor window displays the following SQL code:

```
INNER JOIN category on film_category.category_id = category.category_id
INNER JOIN address on customer.address_id = address.address_id
where address.postal_code = '62703' AND category.name = 'Documentary';

# g) Find all the addresses where the second address line is not empty (i.e., contains some text), and return these second addresses sorted.
DESCRIBE address;
SELECT * FROM address where address2 != null;
```

The results grid shows the following data:

| address_id | address | address2 | district | city_id | postal_code | phone | location | last_update |
|------------|---------|----------|----------|---------|-------------|-------|----------|-------------|
| HULL | HULL | | HULL | HULL | HULL | HULL | HULL | HULL |

The status bar at the bottom indicates the following information: Result 103, address 104, Output, Action Output, Duration / Fetch, 9:30 PM, 11/5/2016.

h) How many films involve a "Crocodile" and a "Shark" based on film description?

`SELECT count(*) FROM film where (description like '%hark%' AND description like '%rocodile%');`

The screenshot shows the MySQL Workbench interface with the following details:

- File Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Toolbar:** Standard database management icons.
- Navigator:** Shows the schema structure under "assign5_dbcode".
- Query Editor:** The query `SELECT count(*) FROM film where (description like '%hark%' AND description like '%rocodile%');` is entered at line 74. A tooltip indicates it's a solution for question h).
- Result Grid:** The result of the query is displayed in a grid, showing a single row with the value 10.
- SQL Additions:** A panel on the right provides context help for the current caret position.
- Output Tab:** Shows the output of the query, including the count of 10.
- System Bar:** Shows the Windows taskbar with various application icons and the date/time (9:32 PM, 11/5/2016).

i) List the actors who played in a film involving a “Crocodile” and a “Shark”, along with the release year of the movie, sorted by the actors’ last names.

```
SELECT actor.last_name,actor.first_name,film.release_year,film.description FROM film
INNER JOIN film_actor ON film.film_id = film_actor.film_id
INNER JOIN actor ON film_actor.actor_id = actor.actor_id
where ( description like '%hark%' AND description like '%rocodile%')
order by actor.last_name;
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

Management Schemas

Information

Schema: sakila

assign5_dbcode*

sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

77
78
79 # i) List the actors who played in a film involving a "Crocodile" and a "Shark", along with the release year of the movie,
80 # sorted by the actors' last names.
81 • SELECT actor.last_name,actor.first_name,film.release_year,film.description FROM film
82 INNER JOIN film_actor ON film.film_id = film_actor.film_id
83 INNER JOIN actor ON film_actor.actor_id = actor.actor_id
84 where (description like '%shark%' AND description like '%crocodile%')
85
86
87 # j) Find all the film categories in which there are between 55 and 65 films.

Result Grid Filter Rows: Export: Wrap Cell Content: Result Form Field Types Query Stats Execution Plan

| last_name | first_name | release_year | description |
|-----------|------------|--------------|--|
| AKROYD | KIRSTEN | 2006 | A Astounding Character Study of A Shark And a A Shark who must Disco... |
| ALLEN | KIM | 2006 | A Fandful Documentary of a Crocodile And a Technical Writer who must Fig... |
| BAILEY | AUDREY | 2006 | A Astounding Yarn of a Pioneer And a Crocodile who must Defeat a Sh... |
| BARRYMORE | JULIA | 2006 | A Fast-Paced Story of a Crocodile And a A Shark who must Sink a Pioneer i... |
| BASINGER | VIVIEN | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| BERGEN | VIVIEN | 2006 | A Brilliant Documentary of a Monkey And a Car who must Conquer a Croco... |
| BERRY | KARL | 2006 | A Fandful Documentary of a Crocodile And a Technical Writer who must Fig... |
| BERRY | KARL | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| BERRY | HENRY | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| BRODY | LAURA | 2006 | A Stunning Story of a Explorer And a Forensic Psychologist who must Face ... |
| CAGE | JOHNNY | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| CAGE | ZERO | 2006 | A Taut Story of a Waitress And a Crocodile who must Outrace a Lumberjac... |
| CHASE | JON | 2006 | A Astounding Character Study of a Shark And a Shark who must Disco... |
| COSTNER | FRED | 2006 | A Fandful Documentary of a Crocodile And a Technical Writer who must Fig... |
| COSTNER | FRED | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| DAVIS | SI KIAN | 2006 | A Brilliant Documentary of a Monkey And a Car who must Conquer a Croco... |

Result 106 x

Output Action Output # Time Action Message Duration / Fetch

Ask me anything 9:33 PM 11/5/2016

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

Management Schemas

Information

Schema: sakila

assign5_dbcode*

sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

77
78
79 # i) List the actors who played in a film involving a "Crocodile" and a "Shark", along with the release year of the movie,
80 # sorted by the actors' last names.
81 • SELECT actor.last_name,actor.first_name,film.release_year,film.description FROM film
82 INNER JOIN film_actor ON film.film_id = film_actor.film_id
83 INNER JOIN actor ON film_actor.actor_id = actor.actor_id
84 where (description like '%shark%' AND description like '%crocodile%')
85
86
87 # j) Find all the film categories in which there are between 55 and 65 films.

Result Grid Filter Rows: Export: Wrap Cell Content: Result Form Field Types Query Stats Execution Plan

| last_name | first_name | release_year | description |
|-----------|------------|--------------|--|
| AKROYD | KIRSTEN | 2006 | A Astounding Character Study of A Shark And a A Shark who must Disco... |
| ALLEN | KIM | 2006 | A Fandful Documentary of a Crocodile And a Technical Writer who must Fig... |
| BAILEY | AUDREY | 2006 | A Astounding Yarn of a Pioneer And a Crocodile who must Defeat a Sh... |
| BARRYMORE | JULIA | 2006 | A Fast-Paced Story of a Crocodile And a A Shark who must Sink a Pioneer i... |
| BASINGER | VIVIEN | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| BERGEN | VIVIEN | 2006 | A Brilliant Documentary of a Monkey And a Car who must Conquer a Croco... |
| BERRY | KARL | 2006 | A Fandful Documentary of a Crocodile And a Technical Writer who must Fig... |
| BERRY | KARL | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| BERRY | HENRY | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| BRODY | LAURA | 2006 | A Stunning Story of a Explorer And a Forensic Psychologist who must Face ... |
| CAGE | JOHNNY | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| CAGE | ZERO | 2006 | A Taut Story of a Waitress And a Crocodile who must Outrace a Lumberjac... |
| CHASE | JON | 2006 | A Astounding Character Study of a Shark And a Shark who must Disco... |
| COSTNER | FRED | 2006 | A Fandful Documentary of a Crocodile And a Technical Writer who must Fig... |
| COSTNER | FRED | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| DAVIS | SI KIAN | 2006 | A Brilliant Documentary of a Monkey And a Car who must Conquer a Croco... |

Result 106 x

Output Action Output # Time Action Message Duration / Fetch

Ask me anything 9:33 PM 11/5/2016

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

Management Schemas

Information

Schema: sakila

assign5_dbcode*

sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

77
78
79 # i) List the actors who played in a film involving a "Crocodile" and a "Shark", along with the release year of the movie,
80 # sorted by the actors' last names.
81 • SELECT actor.last_name,actor.first_name,film.release_year,film.description FROM film
82 INNER JOIN film_actor ON film.film_id = film_actor.film_id
83 INNER JOIN actor ON film_actor.actor_id = actor.actor_id
84 where (description like '%shark%' AND description like '%crocodile%')
85
86
87 # j) Find all the film categories in which there are between 55 and 65 films.

Result Grid Filter Rows: Export: Wrap Cell Content: Result Form Field Types Query Stats Execution Plan

| last_name | first_name | release_year | description |
|-----------|------------|--------------|--|
| AKROYD | KIRSTEN | 2006 | A Astounding Character Study of A Shark And a A Shark who must Disco... |
| ALLEN | KIM | 2006 | A Fandful Documentary of a Crocodile And a Technical Writer who must Fig... |
| BAILEY | AUDREY | 2006 | A Astounding Yarn of a Pioneer And a Crocodile who must Defeat a Sh... |
| BARRYMORE | JULIA | 2006 | A Fast-Paced Story of a Crocodile And a A Shark who must Sink a Pioneer i... |
| BASINGER | VIVIEN | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| BERGEN | VIVIEN | 2006 | A Brilliant Documentary of a Monkey And a Car who must Conquer a Croco... |
| BERRY | KARL | 2006 | A Fandful Documentary of a Crocodile And a Technical Writer who must Fig... |
| BERRY | KARL | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| BERRY | HENRY | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| BRODY | LAURA | 2006 | A Stunning Story of a Explorer And a Forensic Psychologist who must Face ... |
| CAGE | JOHNNY | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| CAGE | ZERO | 2006 | A Taut Story of a Waitress And a Crocodile who must Outrace a Lumberjac... |
| CHASE | JON | 2006 | A Astounding Character Study of a Shark And a Shark who must Disco... |
| COSTNER | FRED | 2006 | A Fandful Documentary of a Crocodile And a Technical Writer who must Fig... |
| COSTNER | FRED | 2006 | A Unbelieveable Drama of a Crocodile And a Mad Cow who must Reach a D... |
| DAVIS | SI KIAN | 2006 | A Brilliant Documentary of a Monkey And a Car who must Conquer a Croco... |

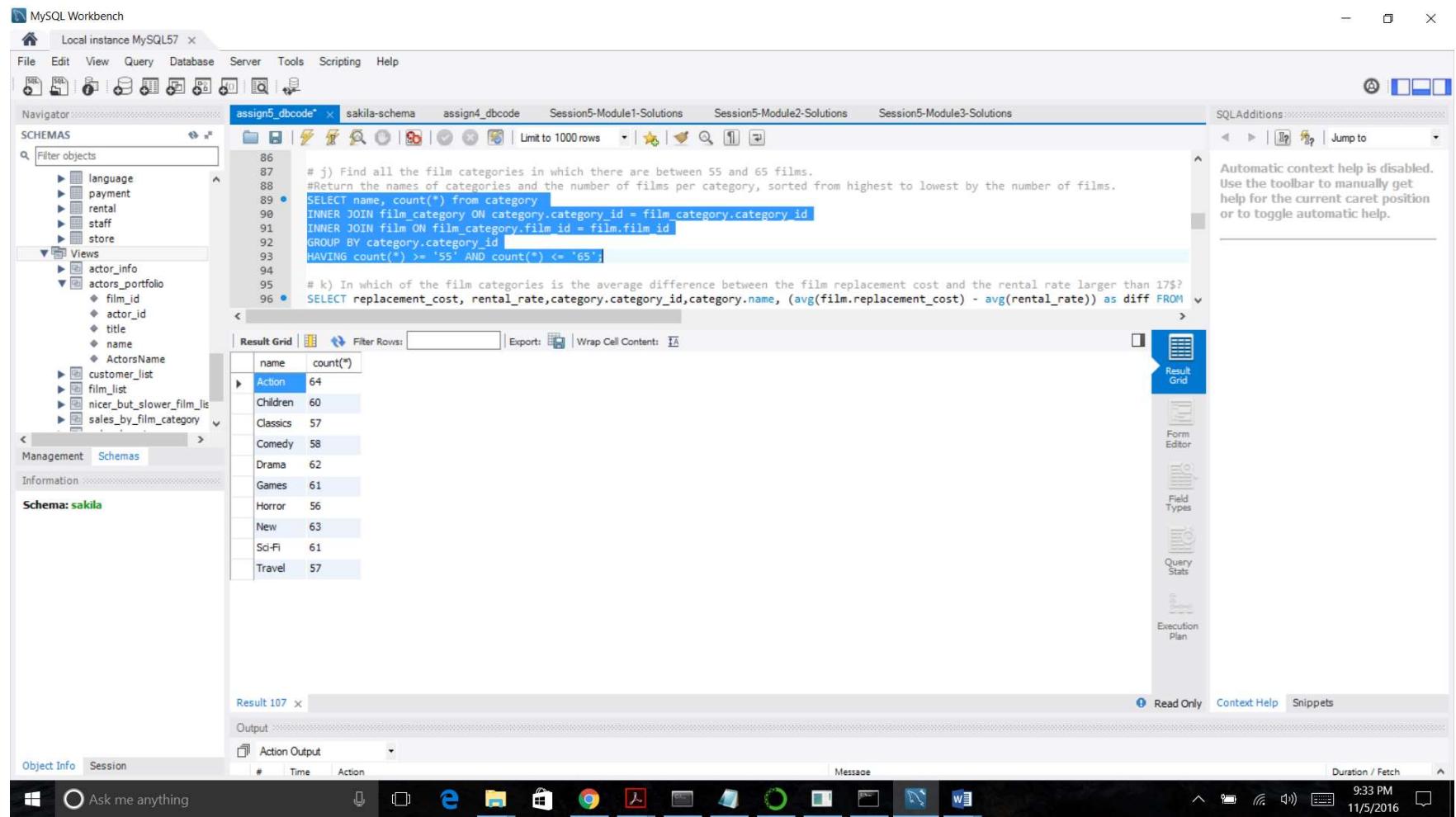
Result 106 x

Output Action Output # Time Action Message Duration / Fetch

Ask me anything 9:33 PM 11/5/2016

j) Find all the film categories in which there are between 55 and 65 films. Return the names of categories and the number of films per category, sorted from highest to lowest by the number of films.

```
SELECT name, count(*) from category
INNER JOIN film_category ON category.category_id = film_category.category_id
INNER JOIN film ON film_category.film_id = film.film_id
GROUP BY category.category_id
HAVING count(*) >= '55' AND count(*) <= '65';
```



k) In which of the film categories is the average difference between the film replacement cost and the rental rate larger than 17\$?

```
SELECT replacement_cost, rental_rate, category.category_id, category.name, (avg(film.replacement_cost) - avg(rental_rate)) as diff
FROM film
INNER JOIN film_category ON film.film_id = film_category.film_id
INNER JOIN category ON film_category.category_id = category.category_id
GROUP BY category.category_id
HAVING (avg(film.replacement_cost) - avg(rental_rate)) > '17';
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

Management Schemas

Information

Schema: sakila

assign5_dbcode* × sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
95 # k) In which of the film categories is the average difference between the film replacement cost and the rental rate larger than 17%?
96 • SELECT replacement_cost, rental_rate,category.category_id,category.name, (avg(film.replacement_cost) - avg(rental_rate)) as diff FROM
97 INNER JOIN film_category ON film.film_id = film_category.film_id
98 INNER JOIN category ON film_category.category_id = category.category_id
99 GROUP BY category.category_id
100 HAVING (avg(film.replacement_cost) - avg(rental_rate)) > '17'
101
102
103 # 1) Many DVD stores produce a daily list of overdue rentals so that customers can be contacted and asked to
104 #return their overdue DVDs. To create such a list, search the rental table for films with a return date that
105 #is NULL and where the rental date is further in the past than the rental duration specified in the film table.
```

Result Grid

| replacement_cost | rental_rate | category_id | name | diff |
|------------------|-------------|-------------|-----------|-----------|
| 20.99 | 0.99 | 1 | Action | 18.265625 |
| 27.99 | 0.99 | 2 | Animation | 17.318182 |
| 24.99 | 4.99 | 3 | Children | 17.166667 |
| 23.99 | 0.99 | 4 | Classics | 18.263158 |
| 15.99 | 2.99 | 7 | Drama | 18.064516 |
| 13.99 | 4.99 | 10 | Games | 17.032787 |
| 15.99 | 0.99 | 14 | Sci-Fi | 17.934426 |
| 24.99 | 4.99 | 15 | Sports | 17.270270 |

Result 108 ×

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

Ask me anything

9:34 PM
11/5/2016

I) Many DVD stores produce a daily list of overdue rentals so that customers can be contacted and asked to return their overdue DVDs. To create such a list, search the rental table for films with a return date that is NULL and where the rental date is further in the past than the rental duration specified in the film table. If so, the film is overdue and we should produce the name of the film along with the customer name and phone number.

```
SELECT CONCAT(customer.last_name, ' ', customer.first_name) AS customer,
       address.phone, film.title
  FROM rental INNER JOIN customer ON rental.customer_id = customer.customer_id
                INNER JOIN address ON customer.address_id = address.address_id
                INNER JOIN inventory ON rental.inventory_id = inventory.inventory_id
                INNER JOIN film ON inventory.film_id = film.film_id
 WHERE rental.return_date IS NULL
   AND rental_date + INTERVAL film.rental_duration DAY < CURRENT_DATE()
  LIMIT 10;
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

Management Schemas

Information

Schema: sakila

assign5_dbcode*

sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
108 • SELECT CONCAT(customer.last_name, ' ', customer.first_name) AS customer,
109 address.phone, film.title
110 FROM rental INNER JOIN customer ON rental.customer_id = customer.customer_id
111 INNER JOIN address ON customer.address_id = address.address_id
112 INNER JOIN inventory ON rental.inventory_id = inventory.inventory_id
113 INNER JOIN film ON inventory.film_id = film.film_id
114 WHERE rental.return_date IS NULL
115 AND rental_date + INTERVAL film.rental_duration DAY < CURRENT_DATE()
116 LIMIT 10;
```

Result Grid

| customer | phone | title |
|------------------|--------------|------------------------|
| KNIGHT, GAIL | 904253967161 | HYDE DOCTOR |
| MAULDIN, GREGORY | 80303246192 | HUNGER ROOF |
| JENKINS, LOUISE | 800716535041 | FRISCO FORREST |
| HOWELL, WILLIE | 991802825778 | TITANS JERK |
| DIAZ, EMILY | 333339908719 | CONNECTION MICROCOSMOS |
| LAWRENCE, LAURIE | 956188728558 | HAUNTED ANTITRUST |
| ANDERSON, LISA | 635297277345 | BULL SHAWSHANK |
| DUGGAN, FREDDIE | 644021380889 | GHOST GROUNDHOG |
| MORRIS, HEATHER | 697760867968 | PEACH INNOCENT |
| SOUTH, ROLAND | 25865528181 | SUIT WALLS |

Result 109 x

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

9:38 PM
11/5/2016

Ask me anything

m) Find the list of all customers and staff given a store id

```
select first_name, last_name from staff  
union all  
select first_name, last_name from customer  
where store_id=2;
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

Management Schemas

Information

Schema: sakila

assign5_dbcode × sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
117
118
119 # m) Find the list of all customers and staff given a store id
120 # Using Union Operator
121
122 select first_name, last_name from staff
123 union all
124 select first_name, last_name from customer
125 where store_id=2;
126
127
```

Result Grid

| first_name | last_name |
|------------|-----------|
| Mike | Hillyer |
| Jon | Stephens |
| BARBARA | JONES |
| JENNIFER | DAVIS |
| SUSAN | WILSON |
| MARGARET | MOORE |
| LISA | ANDERSON |
| KAREN | JACKSON |
| BETTY | WHITE |
| SANDRA | MARTIN |
| CAROL | GARCIA |
| SHARON | ROBINSON |
| SARAH | LEWIS |
| KIMBERLY | LEE |
| JESSICA | HALL |
| CHARITY | AI IFN |

staff 110 ×

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

9:39 PM 11/5/2016

The screenshot shows the MySQL Workbench interface. In the central query editor, a UNION ALL statement is being run against the 'sakila' database. The results grid displays 11 rows of staff names. The status bar at the bottom shows the duration of the fetch as 9:39 PM on 11/5/2016.

Note : use a set operator, do not remove duplicates

QUESTION 2

a) List actors and customers whose first name is the same as the first name of the actor with ID 8.

```
select
CONCAT(customer.last_name, ' ', customer.first_name) AS customers,
CONCAT(actor.last_name, ' ', actor.first_name) AS actors
from customer
inner join actor on
customer.first_name=actor.first_name
where actor.actor_id=8;
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

assign5_dbcode × sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
129 # Note : use a set operator, do not remove duplicates
130 ##### QUESTION 2 #####
131 # a) List actors and customers whose first name is the same as the first name of the actor with ID 8.
132 select
133     CONCAT(customer.last_name, ' ', customer.first_name) AS customers,
134     CONCAT(actor.last_name, ' ', actor.first_name) AS actors
135 from customer
136 inner join actor on
137     customer.first_name=actor.first_name
138 where actor.actor_id=8;
```

Result Grid

| customers | actors |
|----------------|--------------------|
| MAHAN, MATTHEW | JOHANSSON, MATTHEW |

Result 111 ×

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

Read Only Context Help Snippets

Ask me anything

9:45 PM 11/5/2016

b) List customers and payment amounts, with payments greater than average the payment amount

```
select customer.customer_id, customer.first_name, payment.amount as paymentamt from customer
INNER JOIN payment on customer.customer_id = payment.customer_id
group by customer.customer_id
having payment.amount > avg(payment.amount) ;
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

Management Schemas

Information

Schema: sakila

assign5_dbcode × sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

141 # b) List customers and payment amounts, with payments greater than average the payment amount.
142 select customer.customer_id,customer.first_name,payment.amount as paymentamt from customer
143 INNER JOIN payment on customer.customer_id = payment.customer_id
144 group by customer.customer_id
145 having payment.amount > avg(payment.amount);
146
147 # c) List customers who have rented movies atleast once
148 # Note: use IN clause
149
150 151 select CONCAT(customer.last_name, ' ', customer.first_name) AS CUSTOMERS,customer.customer_id AS RENTAL_CUSTOMER_ID

Result Grid

| customer_id | first_name | paymentamt |
|-------------|------------|------------|
| 2 | PATRICIA | 4.99 |
| 4 | BARBARA | 4.99 |
| 6 | JENNIFER | 4.99 |
| 7 | MARIA | 5.99 |
| 8 | SUSAN | 6.99 |
| 9 | MARGARET | 4.99 |
| 10 | DOROTHY | 4.99 |
| 11 | LISA | 6.99 |
| 12 | NANCY | 4.99 |
| 23 | SARAH | 8.99 |
| 24 | KIMBERLY | 6.99 |
| 25 | DEBORAH | 7.99 |
| 31 | BRENDA | 4.99 |
| 32 | AMY | 4.99 |
| 34 | REBECCA | 4.99 |
| 39 | MFRA | 5.99 |

Result 112 ×

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

10:07 PM 11/5/2016

Ask me anything

| customer_id | first_name | paymentamt |
|-------------|------------|------------|
| 2 | PATRICIA | 4.99 |
| 4 | BARBARA | 4.99 |
| 6 | JENNIFER | 4.99 |
| 7 | MARIA | 5.99 |
| 8 | SUSAN | 6.99 |
| 9 | MARGARET | 4.99 |
| 10 | DOROTHY | 4.99 |
| 11 | LISA | 6.99 |
| 12 | NANCY | 4.99 |
| 23 | SARAH | 8.99 |
| 24 | KIMBERLY | 6.99 |
| 25 | DEBORAH | 7.99 |
| 31 | BRENDA | 4.99 |
| 32 | AMY | 4.99 |
| 34 | REBECCA | 4.99 |
| 39 | MFRA | 5.99 |

c) List customers who have rented movies at least once

Note: use IN clause

```
select CONCAT(customer.last_name, ', ', customer.first_name) AS CUSTOMERS, customer.customer_id AS RENTAL_CUSTOMER_ID  
from customer where customer_id in (select customer_id from rental)  
order by RENTAL_CUSTOMER_ID
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

Management Schemas

Information

Schema: sakila

assign5_dbcode

sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
147 # c) List customers who have rented movies atleast once
148 # Note: use IN clause
149
150
151 select CONCAT(customer.last_name, ' ', customer.first_name) AS CUSTOMERS,customer.customer_id AS RENTAL_CUSTOMER_ID
152 from customer where customer_id in (select customer_id from rental)
153
154
155 # d) Find the floor of the maximum, minimum and average payment amount
156 select FLOOR(min(amount)) as minimum, FLOOR(max(amount)) as maximum , floor(avg(amount)) as average from payment;
157
```

Result Grid

| CUSTOMERS | RENTAL_CUSTOMER_ID |
|-------------------|--------------------|
| SMITH, MARY | 1 |
| JOHNSON, PATRICIA | 2 |
| WILLIAMS, LINDA | 3 |
| JONES, BARBARA | 4 |
| BROWN, ELIZABETH | 5 |
| DAVIS, JENNIFER | 6 |
| MILLER, MARIA | 7 |
| WILSON, SUSAN | 8 |
| MOORE, MARGARET | 9 |
| TAYLOR, DOROTHY | 10 |
| ANDERSON, LISA | 11 |
| THOMAS, NANCY | 12 |
| JACKSON, KAREN | 13 |
| WHITE, BETTY | 14 |
| HARRIS, HELEN | 15 |
| MARTIN, SARINA | 16 |

Result 113

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

Ask me anything

10:08 PM
11/5/2016

d) Find the floor of the maximum, minimum and average payment amount

```
select FLOOR(min(amount)) as minimum, FLOOR(max(amount)) as maximum , floor(avg(amount)) as average from payment;
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

assign5_dbcode × sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQLAdditions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
order by RENTAL_CUSTOMER_ID  
# d) Find the floor of the maximum, minimum and average payment amount  
select FLOOR(min(amount)) as minimum, FLOOR(max(amount)) as maximum , floor(avg(amount)) as average from payment;  
##### QUESTION 3 #####  
# a) Create a view called actors_portfolio which contains information about actors and films ( including titles and category).  
DROP VIEW actors_portfolio;  
create view actors_portfolio as  
select film.film_id,actor.actor_id,film.title,category.name, CONCAT(actor.first_name, ' ', actor.last_name) AS ActorsName from actor  
left join film_actor on actor.actor_id=film_actor.actor_id
```

Result Grid | Filter Rows: | Export: | Wrap Cell Content: | Result Grid | Form Editor | Field Types | Query Stats | Execution Plan

| | minimum | maximum | average |
|---|---------|---------|---------|
| 0 | 11 | 4 | |

Result 114 ×

Output

Action Output

Object Info Session

Time Action

Message

Read Only Context Help Snippets

Duration / Fetch

Ask me anything

10:11 PM 11/5/2016

The screenshot shows the MySQL Workbench interface. In the central query editor, a script named 'assign5_dbcode' is open, containing SQL queries for finding payment statistics and creating a view for actors and their portfolios. The results of the last query are displayed in a grid, showing minimum, maximum, and average values. The system tray at the bottom shows the date and time as 10:11 PM on 11/5/2016.

QUESTION 3

a) Create a view called actors_portfolio which contains information about actors and films (including titles and category).

```
create view actors_portfolio as
select film.film_id,actor.actor_id,film.title,category.name, CONCAT(actor.first_name, ' ', actor.last_name) AS ActorsName from actor
left join film_actor on actor.actor_id=film_actor.actor_id
left join film on film_actor.film_id=film.film_id
left join film_category on film.film_id=film_category.film_id
left join category on film_category.category_id=category.category_id;
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store

Views

- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

assign5_dbcode × sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQLAdditions

a) Create a view called actors_portfolio which contains information about actors and films (including titles and category).

```
159 DROP VIEW actors_portfolio;
160 CREATE VIEW actors_portfolio AS
161 select film.film_id, actor.actor_id, film.title, category.name, CONCAT(actor.first_name, ' ', actor.last_name) AS ActorsName from actor
162 left join film_actor on actor.actor_id=film_actor.actor_id
163 left join film on film_actor.film_id=film.film_id
164 left join film_category on film.film_id=film_category.film_id
165 left join category on film_category.category_id=category.category_id;
```

b) Describe the structure of the view and query the view to get information on the actor ADAM GRANT

```
166 DESCRIBE actors_portfolio;
167
168 # b) Describe the structure of the view and query the view to get information on the actor ADAM GRANT
169 DESCRIBE actors_portfolio;
```

Result Grid

| | minimum | maximum | average |
|--|---------|---------|---------|
| | 0 | 11 | 4 |

Form Editor

Field Types

Query Stats

Execution Plan

Result 114 ×

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

Read Only Context Help Snippets

Ask me anything

10:12 PM
11/5/2016

b) Describe the structure of the view and query the view to get information on the actor ADAM GRANT

describe actors_portfolio;

select * from actors_portfolio where ActorsName= 'ADAM GRANT';

The screenshot shows the MySQL Workbench interface with the following details:

- File Bar:** File, Edit, View, Query, Database, Server, Tools, Scripting, Help.
- Toolbar:** Standard database management icons.
- Navigator:** Shows the schema structure. Under the 'Views' section, there is a 'actors_portfolio' view which contains columns: film_id, actor_id, title, name, and ActorsName.
- Query Editor:** The SQL code is as follows:

```
165     left join film_category on film.film_id=film_category.film_id
166     left join category on film_category.category_id=category.category_id;
167
168 # b) Describe the structure of the view and query the view to get information on the actor ADAM GRANT
169 describe actors_portfolio;
170 select * from actors_portfolio where ActorsName= 'ADAM GRANT';
171
172
173 # c) Insert a new movie titled Data Hero in Sci-Fi Category starring ADAM GRANT
174 # We cannot insert data into the View. We can only update the table
175
```
- Result Grid:** The result grid displays the following data:

| film_id | actor_id | title | name | ActorsName |
|---------|----------|-----------------------|-----------|------------|
| 26 | 71 | ANNIE IDENTITY | Sci-Fi | ADAM GRANT |
| 52 | 71 | BALLROOM MOCKINGBIRD | Foreign | ADAM GRANT |
| 233 | 71 | DISCIPLE MOTHER | Travel | ADAM GRANT |
| 317 | 71 | FIREBALL PHILADELPHIA | Comedy | ADAM GRANT |
| 359 | 71 | GLADIATOR WESTWARD | Family | ADAM GRANT |
| 362 | 71 | GLORY TRACY | Games | ADAM GRANT |
| 385 | 71 | GROUNDHOG UNCUT | Comedy | ADAM GRANT |
| 399 | 71 | HAPPINESS UNITED | Foreign | ADAM GRANT |
| 450 | 71 | IDOLS SNATCHERS | Children | ADAM GRANT |
| 532 | 71 | LOSER HUSTLER | Sports | ADAM GRANT |
| 560 | 71 | MARS ROMAN | Games | ADAM GRANT |
| 574 | 71 | MIDNIGHT WESTWARD | Action | ADAM GRANT |
| 638 | 71 | OPERATION OPERATION | Comedy | ADAM GRANT |
| 773 | 71 | SEABISCUIT PUNK | Sports | ADAM GRANT |
| 833 | 71 | SPLENDOR PATTON | Children | ADAM GRANT |
| 874 | 71 | TANPOLE F PARK | Classical | ADAM GRANT |

- SQL Additions:** A panel on the right showing help options like 'Jump to', 'Automatic context help is disabled.', and 'SQL Editor' tools.
- Object Info:** Shows the current object is 'actors_portfolio'.
- Session:** Shows the session status.
- System Bar:** Shows the date and time (10:13 PM, 11/5/2016).

c) Insert a new movie titled Data Hero in Sci-Fi Category starring ADAM GRANT
We cannot insert data into the View. We can only update the table

QUESTION 4

a) Extract the street number (characters 1 through 4) from customer addressLine1

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store
- Views
- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

assign5_dbcode*

sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

176 ##### QUESTION 4 #####
177 # a) Extract the street number (characters 1 through 4) from customer addressLine1
178 SELECT LEFT(address,LOCATE(' ',address) - 1) as StreetNumber from address LIMIT 10;

179
180
181
182
183
184
185
186

Result Grid

Filter Rows: Export: Wrap Cell Content: Fetch rows:

| StreetNumber |
|--------------|
| 47 |
| 28 |
| 23 |
| 1411 |
| 1913 |
| 1121 |
| 692 |
| 1566 |
| 53 |
| 1795 |

Result 117

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

10:14 PM
11/5/2016

The screenshot shows the MySQL Workbench interface with a query editor window open. The query is a SELECT statement using the LEFT function to extract the first four characters of the 'address' column from the 'customer' table, resulting in the 'StreetNumber'. The results are displayed in a grid with 117 rows. The interface includes a sidebar for navigation, a toolbar with various icons, and a status bar at the bottom showing the date and time.

b) Find out actors whose last name starts with character A, B or C.

```
select CONCAT(actor.first_name, ' ', actor.last_name) AS ActorNames
from actor where last_name like 'A%' or last_name like 'B%' or last_name like 'C%'
order by last_name
LIMIT 10;
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store
- Views
- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

assign5_dbcode*

sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
182  
183  
184  
185  
186  
187 # b) Find out actors whose last name starts with character A, B or C.  
188 select CONCAT(actor.first_name, ' ', actor.last_name) AS ActorNames  
189 from actor where last_name like 'A%' or last_name like 'B%' or last_name like 'C%'  
190 order by last_name  
191  
192 LIMIT 10;
```

Result Grid

| ActorNames |
|-------------------|
| CHRISTIAN AKROYD |
| KIRSTEN AKROYD |
| DEBBIE AKROYD |
| CUBA ALLEN |
| KIM ALLEN |
| MERYL ALLEN |
| ANGELINA ASTRAIRE |
| RUSSELL BACALL |
| JESSICA BAILEY |
| AUDREY BAILEY |

Management Schemas

Information

View: actors_portfolio

Columns:

| film_id | smallint(5) UN |
|------------|----------------|
| actor_id | smallint(5) UN |
| title | varchar(255) |
| name | varchar(25) |
| ActorsName | varchar(91) |

Result 118

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

Ask me anything

10:15 PM
11/5/2016

c) Find film titles that contains exactly 10 characters

select title as 10CharTitle from film

where length(title)=10

order by title

LIMIT 10;

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store
- Views
- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

assign5_dbcode × sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
# c) Find film titles that contains exactly 10 characters
select title as 10CharTitle from film
where length(title)=10
order by title
LIMIT 10;
```

Result Grid

Filter Rows: Export: Wrap Cell Content: Fetch rows:

| 10CharTitle |
|-------------|
| ALONE TRIP |
| BASIC EASY |
| BUGSY SONG |
| CAUSE DATE |
| CHILL LUCK |
| CLUE GRAIL |
| CRAZY HOME |
| DATE SPEED |
| DAZED PUNK |
| DOLLS RAGE |

Result Grid

Form Editor

Field Types

Query Stats

Execution Plan

View: actors_portfolio

Columns:

| film_id | smallint(5) UN |
|------------|----------------|
| actor_id | smallint(5) UN |
| title | varchar(255) |
| name | varchar(25) |
| ActorsName | varchar(91) |

film 119 ×

Output

Action Output

Object Info Session

Time Action

Message

Read Only Context Help Snippets

Duration / Fetch

Ask me anything

Windows Taskbar

10:19 PM
11/5/2016

d) Format a payment date using the following format e.g "22/1/2016"
select DATE_FORMAT(payment_date,'%d/%m/%Y') as PaymentDate
from payment
LIMIT 10;

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator assign5_dbcode* sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SCHEMAS

Filter objects

language
payment
rental
staff
store
Views
actor_info
actors_portfolio
film_id
actor_id
title
name
ActorsName
customer_list
film_list
nicer_but_slower_film_list
sales_by_film_category

Management Schemas

Information

View: actors_portfolio

Columns:

| film_id | smallint(5) UN |
|------------|----------------|
| actor_id | smallint(5) UN |
| title | varchar(255) |
| name | varchar(25) |
| ActorsName | varchar(91) |

assign5_dbcode* | # d) Format a payment date using the following format e.g "22/1/2016"
208
209
210
211
212
213
214
215
select DATE_FORMAT(payment_date, '%d/%m/%Y') as PaymentDate
from payment
LIMIT 10;

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

| PaymentDate |
|-------------|
| 25/05/2005 |
| 28/05/2005 |
| 15/06/2005 |
| 15/06/2005 |
| 15/06/2005 |
| 16/06/2005 |
| 18/06/2005 |
| 18/06/2005 |
| 21/06/2005 |
| 08/07/2005 |

Result 120 | Read Only Context Help Snippets

Output Action Output

Object Info Session # Time Action Message Duration / Fetch

Ask me anything

10:20 PM 11/5/2016

MySQL Workbench interface showing a query editor with a result grid and various toolbars and panels.

The query editor contains the following SQL code:

```
# d) Format a payment date using the following format e.g "22/1/2016"
select DATE_FORMAT(payment_date, '%d/%m/%Y') as PaymentDate
from payment
LIMIT 10;
```

The result grid displays the formatted payment dates:

| PaymentDate |
|-------------|
| 25/05/2005 |
| 28/05/2005 |
| 15/06/2005 |
| 15/06/2005 |
| 15/06/2005 |
| 16/06/2005 |
| 18/06/2005 |
| 18/06/2005 |
| 21/06/2005 |
| 08/07/2005 |

The status bar at the bottom shows the time as 10:20 PM and the date as 11/5/2016.

```
# e) Find the number of days between two date values rental date & return date
select DATE_FORMAT(rental_date,'%Y-%m-%d') as rental_date,
       DATE_FORMAT(return_date,'%Y-%m-%d') as return_date,
       datediff(return_date, rental_date) as NumberOfDaysBetween from rental
order by NumberOfDaysBetween
DESC LIMIT 10;
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- language
- payment
- rental
- staff
- store
- Views
- actor_info
- actors_portfolio
 - film_id
 - actor_id
 - title
 - name
 - ActorsName
- customer_list
- film_list
- nicer_but_slower_film_list
- sales_by_film_category

assign5_dbcode*

sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

```
# e) Find the number of days between two date values rental_date & return_date
select DATE_FORMAT(rental_date, '%Y-%m-%d') as rental_date,
       DATE_FORMAT(return_date, '%Y-%m-%d') as return_date,
       datediff(return_date, rental_date) as NumberOfDaysBetween from rental
order by NumberOfDaysBetween
DESC LIMIT 10;
```

Result Grid

| rental_date | return_date | NumberOfDaysBetween |
|-------------|-------------|---------------------|
| 2005-05-27 | 2005-06-06 | 10 |
| 2005-05-26 | 2005-06-05 | 10 |
| 2005-05-25 | 2005-06-04 | 10 |
| 2005-05-24 | 2005-06-03 | 10 |
| 2005-05-30 | 2005-06-09 | 10 |
| 2005-05-26 | 2005-06-05 | 10 |
| 2005-05-29 | 2005-06-08 | 10 |
| 2005-05-30 | 2005-06-09 | 10 |
| 2005-05-25 | 2005-06-04 | 10 |
| 2005-05-31 | 2005-06-10 | 10 |

Fetch previous frame of records from the data source

View: actors_portfolio

Columns:

| | |
|------------|----------------|
| film_id | smallint(5) UN |
| actor_id | smallint(5) UN |
| title | varchar(255) |
| name | varchar(25) |
| ActorsName | varchar(91) |

Result 121

Output

Action Output

Object Info Session

Time Action

Message

Duration / Fetch

10:21 PM
11/5/2016

QUESTION 5

Provide five additional queries (not already in the assignment) along with the specific business cases they address.

Business Case 1: Sort Movies by Rating. Helps business to find out the type of movies customers rate high.

```
select count(*) as FilmsByRating, rating from film
```

```
where release_year = 2006
```

```
group by rating
```

```
order by FilmsByRating DESC;
```

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

assign5_dbcode sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

Filter objects

category city country customer film film_actor film_category film_text inventory language

Columns Indexes Foreign Keys Triggers

film Columns Indexes Foreign Keys Triggers

film_actor film_category film_text inventory language

Management Schemas

Information

assign5_dbcode

DESC LIMIT 10;

QUESTION 5

Provide five additional queries (not already in the assignment) along with the specific business cases they address.

Business Case 1 : Find out what types of Movie receive the higher rating for a specific year.

select count(*) as FilmsByRating, rating from film
where release_year = 2006;
group by rating;
order by FilmsByRating DESC;

Result Grid

| FilmsByRating | rating |
|---------------|--------|
| 223 | PG-13 |
| 210 | NC-17 |
| 195 | R |
| 194 | PG |
| 178 | G |

Result 133 x

Output

Object Info Session

Action Output # Time Action Message Duration / Fetch

Ask me anything

MySQL Workbench

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid Form Editor Field Types Query Stats Execution Plan

Read Only Context Help Snippets

10:41 PM 11/5/2016

Business Case 2: Actors preferred in Movies suitable for kids with no parental guidance. Helps Producers to hire those Actors.

The screenshot shows the MySQL Workbench interface with a query editor and results grid. The query retrieves actor names from the sakila database, filtered by movies with ratings not in ('R', 'PG-13', 'NC-17').

```
order by FilmsByRating DESC;
# Actor most seen in child friend movies.
select CONCAT(actor.first_name, ' ', actor.last_name) AS ActorNames from actor
where actor_id in (
    select actor_id from film_actor where film_id in (
        select film_id from film where rating not in ('R','PG-13','NC-17'))) order by ActorNames
```

| ActorNames |
|--------------------|
| ADAM GRANT |
| ADAM HOPPER |
| AL GARLAND |
| ALAN DREYFUSS |
| ALBERT JOHANSSON |
| ALBERT NOLTE |
| ALEC WAYNE |
| ANGELA HUDSON |
| ANGELA WITHERSPOON |
| ANGELINA ASTRAIRE |
| ANNE CRONYN |
| AUDREY BAILEY |
| AUDREY OLIVIER |
| BELA WALKEN |
| BEN HARRIS |

Output:

1042 PM 11/5/2016

Business Case 3: Top 20 Movies sorted on the Rental Duration helps to identify movies Most in Demand so that stores can stock more of it.

```
select title, rental_duration from film order by rental_duration DESC LIMIT 20
```

The screenshot shows the MySQL Workbench interface with the following details:

- Navigator:** Shows the database schema with the **film** table selected.
- SQL Editor:** Displays the SQL query:

```
242
243
244 # Movies most sought After
245 select title, rental_duration from film order by rental_duration DESC LIMIT 20
246
247
248
249
250
251
252 # Least rented movies AND WHAT CONTRIBUTES TO IT long length movie or is it because of its rating
      select title, rating, length, rental_duration from film
      group by title, rating, length, rental_duration
      order by rental_duration
```
- Result Grid:** Shows the results of the query, listing 20 movies with their titles and rental durations.

| title | rental_duration |
|---------------------|-----------------|
| BOONDOCK BALLROOM | 7 |
| BLACKOUT PRIVATE | 7 |
| ADAPTATION HOLES | 7 |
| ARGONAUTS TOWN | 7 |
| BRINGING HYSTERICAL | 7 |
| ANONYMOUS HUMAN | 7 |
| CELEBRITY HORN | 7 |
| BOWFINGER GABLES | 7 |
| CANYON STOCK | 7 |
| BUCKET BROTHERHOOD | 7 |
| BROOKLYN DESERT | 7 |
| CARRIE BUNCH | 7 |
| BLANKET BEVERLY | 7 |
| BIKINI BORROWERS | 7 |
| BORN SPINAL | 7 |
| RODINF INTRIGUE | 7 |
- Status Bar:** Shows the date and time as 11/5/2016 10:44 PM.

Business Case 4: Top 20 Least Duration Rented Movies and the Reasons for it, so that can stock less.

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- category
- city
- country
- customer
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers
- film
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers
- film_actor
- film_category
- film_text
- inventory
- language

Management Schemas

Information

Result Grid

Least rented movies AND WHAT CONTRIBUTES TO IT long length movie or is it because of its rating

```
select title, rating, length, rental_duration from film
group by title, rating, length, rental_duration
order by rental_duration LIMIT 20
```

| title | rating | length | rental_duration |
|-------------------|--------|--------|-----------------|
| MANNEQUIN WORST | PG-13 | 71 | 3 |
| GODFATHER DIARY | NC-17 | 73 | 3 |
| KING EVOLUTION | NC-17 | 184 | 3 |
| DAUGHTER MADIGAN | PG-13 | 59 | 3 |
| MONEY HAROLD | PG | 135 | 3 |
| LOSE INCH | R | 137 | 3 |
| CARIBBEAN LIBERTY | NC-17 | 92 | 3 |
| SIMON NORTH | NC-17 | 51 | 3 |
| PACIFIC AMISTAD | G | 144 | 3 |
| FEATHERS METAL | PG-13 | 104 | 3 |
| SUGAR WONKA | PG | 114 | 3 |
| NECKLACE OUTBREAK | PG | 132 | 3 |
| TIMBERLAND SKY | G | 69 | 3 |
| CORE SUIT | PG-13 | 92 | 3 |
| EGYPT TENENBAUMS | PG | 85 | 3 |
| PARKER MANTGAN | PG-13 | 84 | 3 |

Output

Action Output

Object Info Session # Time Action Message Duration / Fetch

Ask me anything

10:46 PM 11/5/2016

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid Form Editor Field Types Query Stats Execution Plan

Read Only Context Help Snippets

Business Case 4: Find out how Rating of the Movies affects the Rental Duration.

select rating, avg(rental_duration) as AverageRentalDuration from film

group by rating

order by AverageRentalDuration

MySQL Workbench

Local instance MySQL57

File Edit View Query Database Server Tools Scripting Help

Navigator

SCHEMAS

- category
- city
- country
- customer
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers
- film
 - Columns
 - Indexes
 - Foreign Keys
 - Triggers
- film_actor
- film_category
- film_text
- inventory
- language

Management Schemas

Information

assign5_dbcode

sakila-schema assign4_dbcode Session5-Module1-Solutions Session5-Module2-Solutions Session5-Module3-Solutions

Rental Duration vs Rating

```
select rating, avg(rental_duration) as AverageRentalDuration from film
group by rating
order by AverageRentalDuration
```

Result Grid

| rating | AverageRentalDuration |
|--------|-----------------------|
| R | 4.7744 |
| G | 4.8371 |
| PG-13 | 5.0538 |
| PG | 5.0825 |
| NC-17 | 5.1429 |

SQL Additions

Automatic context help is disabled. Use the toolbar to manually get help for the current caret position or to toggle automatic help.

Result Grid Form Editor Field Types Query Stats Execution Plan

Result 139

Output

Object Info Session

Time Action

Message

Duration / Fetch

10:47 PM 11/5/2016

Ask me anything

The screenshot shows the MySQL Workbench interface with a query editor window open. The query is:select rating, avg(rental_duration) as AverageRentalDuration from film group by rating order by AverageRentalDurationThe results grid displays the following data:| rating | AverageRentalDuration |
| --- | --- |
| R | 4.7744 |
| G | 4.8371 |
| PG-13 | 5.0538 |
| PG | 5.0825 |
| NC-17 | 5.1429 |

