DEPA ASSIGNMENT 2 – SQL QUERIES

Name: Troy Zhongyi Zhang
Netid: zhongyiz@uchicago.edu

Part B (Individual): Manipulating, Categorizing, Sorting and Grouping & Summarizing Data
Data (Sakila dataset)

➢ We will use the Sakila database schema which can be found at: http://dev.mysql.com/doc/index-other.html/

➢ Full documentation: http://dev.mysql.com/doc/sakila/en/

/**********
**** File: Assignment2-PartB.sql**
**** Desc: Manipulating, Categorizing, Sorting and Grouping & Summarizing Data**
**** Author: Troy Zhongyi Zhang**
**** Date: 07/25/2019**
**********/

######## QUESTION 1 ######## – { 10 Points }

 # a) Show the list of databases.

SHOW DATABASES;

8 row(s) returned

Database
classicmodels
classicmodelsdw
information_schema
mydb
mysql
performance_schema
sakila

 # b) Select sakila database.

USE sakila;

0 row(s) affected

 # c) Show all tables in the sakila database.

SHOW TABLES;

17 row(s) returned

Tables_in_sakila
actor
address
category
city
country
customer
film
Result 29

 # d) Show each of the columns along with their data types for the actor table.

DESCRIBE actor;

4 row(s) returned

Field	Type	Null	Key	Default	Extra
actor_id	smallint(5) unsigned	NO	PRI	NULL	auto_increment
first_name	varchar(45)	NO		NULL	
last_name	varchar(45)	NO	MUL	NULL	
last_update	timestamp	NO		CURRENT_TIMESTA…	DEFAULT_GENERATED on update CURRENT…

 # e) Show the total number of records in the actor table.

SELECT COUNT(*) FROM actor;
or
SELECT COUNT(actor_id) AS totalNumberOfActors FROM actor;

1 row(s) returned

totalNumberOfActors
200

Return 200 actors

 # f) What is the first name and last name of all the actors in the actor table?

```
SELECT
   first_name, last_name
FROM
   actor;
```

200 row(s) returned

first_name	last_name
PENELOPE	GUINESS
NICK	WAHLBERG
ED	CHASE
JENNIFER	DAVIS
JOHNNY	LOLLOBRIGIDA
BETTE	NICHOLSON
GRACE	MOSTEL

 # g) Insert your first name and middle initial (in the last name column) into the actors table.

```
INSERT INTO `actor`
         (`actor_id`,`first_name`,`last_name`,`last_update`)
VALUES
('201','Troy Zhongyi','Z','2019-07-18');
```

1 row(s) affected

 # h) Update your middle initial with your last name in the actors table.

```
SET SQL_SAFE_UPDATES = 0;
UPDATE actor
SET
   last_name = 'Z Zhang'
WHERE
   last_name = 'Z';
```

1 row(s) affected Rows matched: 1 Changed: 1 Warnings: 0

 # i) Delete the record from the actor table where the first name matches your first name.

```
DELETE FROM actor
WHERE first_name = 'Troy Zhongyi';
```

1 row(s) affected

 # j) Create a table payment_type with the following specifications and appropriate data types
 # Table Name : “Payment_type”
 # Primary Key: "payment_type_id”
 # Column: “Type”
 # Insert following rows in to the table:
 # 1, “Credit Card” ; 2, “Cash”; 3, “Paypal” ; 4 , “Cheque”

```
CREATE TABLE payment_type (
  payment_type_id SMALLINT UNSIGNED NOT NULL AUTO_INCREMENT,
  Type VARCHAR(45) NOT NULL,
  PRIMARY KEY  (payment_type_id)
)ENGINE=InnoDB DEFAULT CHARSET=utf8;

INSERT INTO `payment_type`
        (`payment_type_id`,`Type`)
VALUES
(1,'Credit Card'),
(2,'Cash'),
(3,'Paypal'),
(4,'Cheque');
```

Create tables responses: 0 row(s) affected, 1 warning(s): 3719 'utf8' is currently an alias for the character set UTF8MB3, but will be an alias for UTF8MB4 in a future release. Please consider using UTF8MB4 in order to be unambiguous.
Insert data reponses: 4 row(s) affected Records: 4 Duplicates: 0 Warnings: 0

 # k) Rename table payment_type to payment_types.

```
RENAME TABLE payment_type TO payment_types;
```

0 row(s) affected

 # l) Drop the table payment_types.

```
DROP TABLE payment_types;
```

0 row(s) affected

QUESTION 2 ######## – { 10 Points }

 # a) List all the movies (title & description) that are rated PG-13 ?

```
SELECT
   title, description
FROM
   film
WHERE
         rating = 'PG-13';
```

223 row(s) returned

title	description
AIRPLANE SIERRA	A Touching Saga of a Hunter And a Butler who…
ALABAMA DEVIL	A Thoughtful Panorama of a Database Administ…
ALTER VICTORY	A Thoughtful Drama of a Composer And a Femi…
ANTHEM LUKE	A Touching Panorama of a Waitress And a Wom…
APOLLO TEEN	A Action-Packed Reflection of a Crocodile And a…
ARACHNOPHOBIA ROLLERCOAST…	A Action-Packed Reflection of a Pastry Chef An…

film 50

 # b) List all movies that are either PG OR PG-13 using IN operator ?

```
SELECT
   title, description
FROM
   film
WHERE
         rating IN ('PG' , 'PG-13');
```

417 row(s) returned

title	description
ACADEMY DINOSAUR	A Epic Drama of a Feminist And a Mad Scientist…
AGENT TRUMAN	A Intrepid Panorama of a Robot And a Boy who…
AIRPLANE SIERRA	A Touching Saga of a Hunter And a Butler who…
ALABAMA DEVIL	A Thoughtful Panorama of a Database Administ…
ALASKA PHANTOM	A Fanciful Saga of a Hunter And a Pastry Chef…
ALI FOREVER	A Action-Packed Drama of a Dentist And a Croc…

film 51

 # c) Report all payments greater than and equal to 2$ and Less than equal to 7$?
 # Note : write 2 separate queries conditional operator and BETWEEN keyword

```
SELECT
   *
FROM
   payment
WHERE
```

```
    (amount >= 2
        AND amount <= 7);
```

1000 row(s) returned

payment_id	customer_id	staff_id	rental_id	amount	payment_date	last_update
1	1	1	76	2.99	2005-05-25 11:30:37	2006-02-15 22:12:30
3	1	1	1185	5.99	2005-06-15 00:54:12	2006-02-15 22:12:30
6	1	1	1725	4.99	2005-06-16 15:18:57	2006-02-15 22:12:30
7	1	1	2308	4.99	2005-06-18 08:41:48	2006-02-15 22:12:30
9	1	1	3284	3.99	2005-06-21 06:24:45	2006-02-15 22:12:30
10	1	2	4526	5.99	2005-07-08 03:17:05	2006-02-15 22:12:30

payment 52

```
SELECT
    *
FROM
    payment
WHERE
    buyPrice BETWEEN 2 AND 7;
```

1000 row(s) returned

payment_id	customer_id	staff_id	rental_id	amount	payment_date	last_update
1	1	1	76	2.99	2005-05-25 11:30:37	2006-02-15 22:12:30
3	1	1	1185	5.99	2005-06-15 00:54:12	2006-02-15 22:12:30
6	1	1	1725	4.99	2005-06-16 15:18:57	2006-02-15 22:12:30
7	1	1	2308	4.99	2005-06-18 08:41:48	2006-02-15 22:12:30
9	1	1	3284	3.99	2005-06-21 06:24:45	2006-02-15 22:12:30
10	1	2	4526	5.99	2005-07-08 03:17:05	2006-02-15 22:12:30

payment 53

 # d) List all addresses that have phone number that contain digits 589, start with 140 or end with 589
 # Note : write 3 different queries

```
SELECT
        *
FROM
    address
WHERE
    phone LIKE '%589%';
```

6 row(s) returned

address_id	address	address2	district	city_id	postal_code	phone	location	last_update
4	1411 Lillydale Drive	NULL	QLD	576		6172235589	BLOB	2014-09-25 22:30:09
135	782 Mosul Street		Massachusetts	94	25545	885899703621	BLOB	2014-09-25 22:33:46
333	1860 Taguig Loop		West Java	119	59550	38158430589	BLOB	2014-09-25 22:31:32
388	368 Hunuco Boulevard		Namibe	360	17165	106439158941	BLOB	2014-09-25 22:30:03
492	185 Mannheim Lane		Stavropol	408	23661	589377568313	BLOB	2014-09-25 22:32:56
504	1 Valle de Santiago Avenue		Apulia	93	86208	465897838272	BLOB	2014-09-25 22:31:43
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

Export recordset to an external file

```
SELECT
        *
FROM
   address
WHERE
   phone LIKE '140%';
```

1 row(s) returned

address_id	address	address2	district	city_id	postal_code	phone	location	last_update
3	23 Workhaven Lane	NULL	Alberta	300		14033335568	BLOB	2014-09-25 22:30:27
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

```
SELECT
        *
FROM
   address
WHERE
   phone LIKE '%589';
```

2 row(s) returned

address_id	address	address2	district	city_id	postal_code	phone	location	last_update
4	1411 Lillydale Drive	NULL	QLD	576		6172235589	BLOB	2014-09-25 22:30:09
333	1860 Taguig Loop		West Java	119	59550	38158430589	BLOB	2014-09-25 22:31:32
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

 # e) List all staff members (first name, last name, email) whose password is NULL ?

```
SELECT DISTINCT
   first_name, last_name, email
FROM
   staff
WHERE
   password IS NULL;
```

1 row(s) returned

first_name	last_name	email
Jon	Stephens	Jon.Stephens@sakilastaff.com

Returned result: Jon Stephens

 # f) Select all films that have title names like ZOO and rental duration greater than or equal to 4

```
SELECT
        *
FROM
   film
WHERE
   title LIKE '%ZOO%' AND
   length >= 4;
```

4 row(s) returned

film_id	title	description	release_year	language_id	original_language_id	rental_duration	rental_rate	length	replacement_cost	rating
568	MEMENTO ZOOLANDER	A Touching Epistle of a Squirrel And a Explorer who must Redeem a Pastry Chef in The Sahara Desert				4	4.99	77	11.99	NC-17
583	MISSION ZOOLANDER	A Intrepid Story of a Sumo Wrestler And a Teac…	2006	1	NULL	3	4.99	164	26.99	PG-13
924	UNFORGIVEN ZOOLAND…	A Taut Epistle of a Monkey And a Sumo Wrestle…	2006	1	NULL	7	0.99	129	15.99	PG
999	ZOOLANDER FICTION	A Fateful Reflection of a Waitress And a Boat w…	2006	1	NULL	5	2.99	101	28.99	R
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

film_id	title
568	MEMENTO ZOOLANDER
583	MISSION ZOOLANDER
924	UNFORGIVEN ZOOLAND…
999	ZOOLANDER FICTION
NULL	NULL

 # g) What is the cost of renting the movie ACADEMY DINOSAUR for 2 weeks ?
 # Note : use of column alias

```
SELECT
        (rental_rate * 14) AS rentalCost
FROM
   film
WHERE
```

```
    title LIKE 'ACADEMY DINOSAUR';
```

1 row(s) returned

rentalCost
13.86

Returned value = 13.86 dollars

 # h) List all unique districts where the customers, staff, and stores are located
 # Note : check for NOT NULL values

```
SELECT DISTINCT
    district
FROM
    address
WHERE
    district IS NOT NULL;
```

378 row(s) returned

district
Alberta
QLD
Nagasaki
California
Attika
Mandalay
Nantou
Texas
Central Serbia

address 65

 # i) List the top 10 newest customers across all stores

```
SELECT
    *
FROM
    customer
ORDER BY create_date DESC
LIMIT 10;
```

10 row(s) returned

customer_id	store_id	first_name	last_name	email	address_id	active	create_date	last_update
275	2	CAROLE	BARNETT	CAROLE.BARNETT@sakilacustomer.org	280	1	2006-02-14 22:04:37	2006-02-15 04:57:20
281	2	LEONA	OBRIEN	LEONA.OBRIEN@sakilacustomer.org	286	1	2006-02-14 22:04:37	2006-02-15 04:57:20
278	2	BILLIE	HORTON	BILLIE.HORTON@sakilacustomer.org	283	1	2006-02-14 22:04:37	2006-02-15 04:57:20
276	1	BRANDY	GRAVES	BRANDY.GRAVES@sakilacustomer.org	281	1	2006-02-14 22:04:37	2006-02-15 04:57:20
599	2	AUSTIN	CINTRON	AUSTIN.CINTRON@sakilacustomer.org	605	1	2006-02-14 22:04:37	2006-02-15 04:57:20
280	2	TRACEY	BARRETT	TRACEY.BARRETT@sakilacustomer.org	285	1	2006-02-14 22:04:37	2006-02-15 04:57:20
274	1	NAOMI	JENNINGS	NAOMI.JENNINGS@sakilacustomer.org	279	1	2006-02-14 22:04:37	2006-02-15 04:57:20
279	2	DIANNE	SHELTON	DIANNE.SHELTON@sakilacustomer.org	284	1	2006-02-14 22:04:37	2006-02-15 04:57:20
277	2	OLGA	JIMENEZ	OLGA.JIMENEZ@sakilacustomer.org	282	1	2006-02-14 22:04:37	2006-02-15 04:57:20
273	2	PRISCILLA	LOWE	PRISCILLA.LOWE@sakilacustomer.org	278	1	2006-02-14 22:04:37	2006-02-15 04:57:20
NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL	NULL

QUESTION 3 ######## – { 10 Points }

 # a) Show total number of movies

```
SELECT
   COUNT(film_id) AS numberOfMovies
FROM
   film;
```

1 row(s) returned

numberOfMovies
1000

Returned 1,000 movies in total.

 # b) What is the minimum payment received and max payment received across all transactions ?

```
SELECT
   MAX(amount) as maxPayment
#    MIN(amount) as minPayment
FROM
   payment
```

1 row(s) returned

maxPayment
11.99

Returned 11.99 dollars

```
SELECT
#    MAX(amount) as maxPayment
    MIN(amount) as minPayment
FROM
    payment
```

1 row(s) returned

minPayment
0.00

Returned 0 dollar.

 # c) Number of customers that rented movies between Feb-2005 & May-2005 (based on paymentDate).

```
SELECT
    COUNT(customer_id) AS totalCustomers
FROM
    payment
WHERE
    payment_date >= '2005-02-01'
    AND payment_date <= '2005-05-31';
```

1 row(s) returned

totalCustomers
994

Returned 994 total customers.

 # d) List all movies where replacement_cost is greater than 15$ or rental_duration is between 6 & 10 days

```
Select
    title
FROM
    film
WHERE
    replacement_cost >= 15 AND
    (rental_duration >= 6 AND rental_duration <= 10)
```

284 row(s) returned

title
ACADEMY DINOSAUR
ADAPTATION HOLES
AFRICAN EGG
AIRPLANE SIERRA
AIRPORT POLLOCK
ALADDIN CALENDAR
ALAMO VIDEOTAPE
ALASKA PHANTOM
ALICE FANTASIA
ALLEY EVOLUTION
ALTER VICTORY
AMADEUS HOLY
ANALYZE HOOSIERS

film 71

 # e) What is the total amount spent by customers for movies in the year 2005 ?

```
SELECT
   SUM(amount) AS totalAmountSpent
FROM
   payment
WHERE
   payment_date BETWEEN '2005-01-01' AND '2005-12-31';
```

1 row(s) returned

totalAmountSpent
66902.33

Returned 66902.33 dollars

 # f) What is the average replacement cost across all movies ?

```
SELECT
   SUM(replacement_cost) AS Total_replacement_cost
FROM
```

```
    film;
```

1 row(s) returned

Total_replacement_cost
19984.00

Returned 19,984 dollars.

 # g) What is the standard deviation of rental rate across all movies ?

```
SELECT
    STD(rental_rate) AS stdRental_rate
FROM
    film;
```

1 row(s) returned

stdRental_rate
1.6455698101265719

Returned 1.6456

 # h) What is the midrange of the rental duration for all movies

```
SELECT
    (MAX(rental_duration) + MIN(rental_duration)) / 2 AS midRange
FROM
    film;
```

1 row(s) returned

midRange
5.0000

The middle ruange is 5.0000.

######## QUESTION 4 ######## – { 10 Points }

 # a) Customers sorted by first Name and last name in ascending order.

```
SELECT
   *
FROM
   customer
ORDER BY first_name ASC, last_name ASC;
```

 # b) Count of movies that are either G/NC-17/PG-13/PG/R grouped by rating.

```
SELECT
   COUNT(film_id), rating
FROM
   film
GROUP BY rating
```

 # c) Number of addresses in each district.

```
SELECT
   count(address), district
FROM address
GROUP BY district
```

378 row(s) returned

count(address)	district
2	Alberta
2	QLD
1	Nagasaki
9	California
1	Attika
1	Mandalay
2	Nantou
5	Texas
1	Central Serbia
1	Hamilton
1	Masqat
3	Esfahan
2	Kanagawa

Result 77

 # d) Find the movies where rental rate is greater than 1$ and order result set by descending order.

```
SELECT DISTINCT
   film_id, title, rental_rate
FROM
   film
WHERE rental_rate > 1
ORDER BY rental_rate ASC;
```

659 row(s) returned

film_id	title	rental_rate
3	ADAPTATION HOLES	2.99
4	AFFAIR PREJUDICE	2.99
5	AFRICAN EGG	2.99
6	AGENT TRUMAN	2.99
9	ALABAMA DEVIL	2.99
15	ALIEN CENTER	2.99
16	ALLEY EVOLUTION	2.99
22	AMISTAD MIDSUMMER	2.99
24	ANALYZE HOOSIERS	2.99
25	ANGELS LIFE	2.99
29	ANTITRUST TOMATO…	2.99
30	ANYTHING SAVANNAH	2.99
33	APOLLO TEEN	2.99

film 78

 # e) Top 2 movies that are rated R with the highest replacement cost ?

```
SELECT DISTINCT
   film_id, title, replacement_cost, rating
FROM
   film
WHERE
   rating = 'R'
ORDER BY replacement_cost DESC
LIMIT 2;
```

2 row(s) returned

film_id	title	replacement_cost	rating
138	CHARIOTS CONSPIRA…	29.99	R
199	CUPBOARD SINNERS	29.99	R
NULL	NULL	NULL	NULL

 # f) Find the most frequently occurring (mode) rental rate across products.

```
SELECT
        rental_rate, COUNT(rental_rate) AS RentalRateMode
FROM
   film
ORDER BY RentalRateMode DESC
LIMIT 1;
```

1 row(s) returned

rental_rate	RentalRateMode
0.99	1000

Returned 0.99 is the most frequently occurring rental rate.

 # g) Find the top 2 movies with movie length greater than 50mins and which has commentaries as a special features.

```
SELECT DISTINCT
   film_id, title, length, special_features
FROM
   film
WHERE
   length > 50
HAVING special_features LIKE '%commentaries%'
ORDER BY length DESC
LIMIT 2;
```

2 rows returned

film_id	title	length	special_features
182	CONTROL ANTHEM	185	Commentaries
426	HOME PITY	185	Trailers,Commentaries,Behind the Scenes
NULL	NULL	NULL	NULL

 # h) List the years which has more than 2 movies released.

```
SELECT release_year, COUNT(DISTINCT(film_id)) as numberOfFilms
FROM film
GROUP BY release_year
HAVING COUNT(film_id) > 2;
```

1 row returned

release_year	numberOfFilms
2006	1000

Returned year 2006.

/**********
**** File: Assignment2-PartC.sql**
**** Desc: Combining Data, Nested Queries, Views and Indexes, Transforming Data**
**** Author: Troy Zhongyi Zhang**
**** Date: 07/26/2019**
**********/

QUESTION 1 ######## – { 20 Points }

 # a) List the actors (firstName, lastName) who acted in more then 25 movies.
 # Note: Also show the count of movies against each actor

```
SELECT
   a.first_name,
   a.last_name,
   COUNT(DISTINCT f.film_id) AS TotalNumberOfMovies
FROM
   film_actor f
      INNER JOIN
   actor a USING (actor_id)
GROUP BY a.actor_id
HAVING totalNumberOfMovies > 25
ORDER BY a.actor_id;
```

125 row(s) returned

first_name	last_name	TotalNumberOfMovies
JOHNNY	LOLLOBRIGIDA	29
GRACE	MOSTEL	30
KARL	BERRY	31
UMA	WOOD	35
VIVIEN	BERGEN	30
CUBA	OLIVIER	28
FRED	COSTNER	27
HELEN	VOIGHT	32
LUCILLE	TRACY	30
KIRSTEN	PALTROW	27
ELVIS	MARX	26
SANDRA	KILMER	37
RIP	CRAWFORD	33

Result 80

 # b) List the actors who have worked in the German language movies.
 # 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%";

```
SET SQL_SAFE_UPDATES=0;
UPDATE film SET language_id=6 WHERE title LIKE "%ACADEMY%";

SELECT
   l.language_id, l.name, a.actor_id, a.first_name, a.last_name
FROM
   language l
      INNER JOIN
   film f ON l.language_id = f.language_id
      INNER JOIN
   film_actor fa ON f.film_id = fa.film_id
      INNER JOIN
   actor a ON fa.actor_id = a.actor_id
WHERE
   l.language_id = 6;
```

2 row(s) affected Rows matched: 2 Changed: 2 Warnings: 0

language_id	name	actor_id	first_name	last_name
6	German	1	PENELOPE	GUINESS
6	German	10	CHRISTIAN	GABLE
6	German	20	LUCILLE	TRACY
6	German	30	SANDRA	PECK
6	German	40	JOHNNY	CAGE
6	German	53	MENA	TEMPLE
6	German	108	WARREN	NOLTE
6	German	162	OPRAH	KILMER
6	German	188	ROCK	DUKAKIS
6	German	198	MARY	KEITEL
6	German	51	GARY	PHOENIX
6	German	88	KENNETH	PESCI
6	German	182	DEBBIE	AKROYD

Result 81

 # c) List the actors who acted in horror movies.
 # Note: Show the count of movies against each actor in the result set.

```
SELECT c.name, a.first_name, a.last_name, COUNT(f.film_id)
FROM actor a
INNER JOIN film_actor fa
ON a.actor_id = fa.actor_id
LEFT JOIN film f
ON fa.film_id = f.film_id
LEFT JOIN film_category fc
```

```
ON f.film_id = fc.film_id
LEFT JOIN category c
ON fc.category_id = c.category_id
WHERE c.name = 'horror'
GROUP BY a.first_name, a.last_name;
```

155 row(s) returned

name	first_name	last_name	COUNT(f.film_id)
Horror	BOB	FAWCETT	1
Horror	MINNIE	ZELLWEG…	3
Horror	SEAN	GUINESS	1
Horror	CHRIS	DEPP	2
Horror	JODIE	DEGENER…	2
Horror	SCARLETT	DAMON	2
Horror	KENNETH	PESCI	2
Horror	FAY	WINSLET	2
Horror	OPRAH	KILMER	3
Horror	FAY	KILMER	1
Horror	GENE	WILLIS	1
Horror	SUSAN	DAVIS	4
Horror	LUCILLE	DEE	3
Horror	CHRISTIAN	GABLE	2

 # d) List all customers who rented more than 3 horror movies.

```
SELECT
   c.customer_id,
   c.first_name,
   c.last_name,
   COUNT(fc.film_id) AS NumberOfHorrorMovies,
   a.first_name,
   a.last_name
FROM
   customer c
      INNER JOIN
   rental r ON c.customer_id = r.customer_id
      INNER JOIN
   inventory i ON i.inventory_id = r.inventory_id
      INNER JOIN
   film_category fc ON fc.film_id = i.film_id
      INNER JOIN
   category ca ON ca.category_id = fc.category_id
      INNER JOIN
```

```
    film_actor fa ON fc.film_id = fa.film_id
        INNER JOIN
    actor a ON fa.actor_id = a.actor_id
WHERE
    ca.name = 'Horror'
GROUP BY c.customer_id
HAVING NumberOfHorrorMovies > 3;
```

429 row(s) returned

customer_id	first_name	last_name	NumberOfHorrorMovies	first_name	last_name
271	PENNY	NEAL	7	BOB	FAWCETT
366	BRANDON	HUEY	10	BOB	FAWCETT
145	LUCILLE	HOLMES	15	BOB	FAWCETT
82	KATHRYN	COLEMAN	12	BOB	FAWCETT
139	AMBER	DIXON	8	BOB	FAWCETT
180	STACY	CUNNINGHAM	8	BOB	FAWCETT
533	JESSIE	MILAM	12	BOB	FAWCETT
184	VIVIAN	RUIZ	5	JODIE	DEGENER...
268	NINA	SOTO	25	JODIE	DEGENER...
560	JORDAN	ARCHULETA	5	JODIE	DEGENER...
150	DANIELLE	DANIELS	5	JODIE	DEGENER...
197	SUE	PETERS	27	JODIE	DEGENER...
302	MICHAEL	SILVERMAN	16	JODIE	DEGENER...
418	JEFF	EAST	20	JODIE	DEGENER...
151	MEGAN	PALMER	12	JODIE	DEGENER...
148	ELEANOR	HUNT	27	JODIE	DEGENER

Result 85

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

```
SELECT DISTINCT
    c.customer_id,
    c.first_name,
    c.last_name,
    fa.film_id,
    a.first_name,
    a.last_name
FROM
    customer c
        INNER JOIN
    rental r ON c.customer_id = r.customer_id
        INNER JOIN
    inventory i ON i.inventory_id = r.inventory_id
        INNER JOIN
    film_actor fa ON fa.film_id = i.film_id
        INNER JOIN
    actor a ON a.actor_id = fa.actor_id
```

```
WHERE
    a.first_name = 'SCARLETT'
        AND a.last_name = 'BENING';
```

356 row(s) returned

customer_id	first_name	last_name	film_id	first_name	last_name
59	CHERYL	MURPHY	22	SCARLETT	BENING
339	WALTER	PERRYMAN	22	SCARLETT	BENING
1	MARY	SMITH	22	SCARLETT	BENING
128	MARJORIE	TUCKER	22	SCARLETT	BENING
29	ANGELA	HERNANDEZ	22	SCARLETT	BENING
388	CRAIG	MORRELL	22	SCARLETT	BENING
560	JORDAN	ARCHULETA	22	SCARLETT	BENING
17	DONNA	THOMPSON	22	SCARLETT	BENING
479	ZACHARY	HITE	22	SCARLETT	BENING
137	RHONDA	KENNEDY	22	SCARLETT	BENING
561	IAN	STILL	22	SCARLETT	BENING
253	TERRY	CARLSON	22	SCARLETT	BENING
5	ELIZABETH	BROWN	22	SCARLETT	BENING
508	MILTON	HOWLAND	22	SCARLETT	BENING
286	VELMA	LUCAS	22	SCARLETT	BENING

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

```
SELECT
    c.customer_id,
    c.first_name,
    c.last_name,
    a.address,
    a.address2,
    a.postal_code,
    ca.name
FROM
    customer c
        INNER JOIN
    rental r ON c.customer_id = r.customer_id
        INNER JOIN
    inventory i ON i.inventory_id = r.inventory_id
        INNER JOIN
    film_category fc ON fc.film_id = i.film_id
        INNER JOIN
    category ca ON ca.category_id = fc.category_id
```

```
    INNER JOIN
  address a ON a.address_id = c.address_id
WHERE
  ca.name = 'Documentary'
    AND a.postal_code = '62703';
```

1 row(s) returned

customer_id	first_name	last_name	address	address2	postal_code	name
582	ANDY	VANHORN	966 Asuncin Way		62703	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.

```
SELECT
  address, address2
FROM
  address
WHERE
  TRIM(address2) IS NOT NULL
ORDER BY address2;
```

599 row(s) returned

address	address2
1913 Hanoi Way	
1121 Loja Avenue	
692 Joliet Street	
1566 Inegl Manor	
53 Idfu Parkway	
1795 Santiago de Compostela Way	
900 Santiago de Compostela Parkway	
478 Joliet Way	
613 Korolev Drive	
1531 Sal Drive	
1542 Tarlac Parkway	
808 Bhopal Manor	
270 Amroha Parkway	
770 Bydgoszcz Avenue	
419 Iligan Lane	

 # h) How many films involve a “Crocodile” and a “Shark” based on film description ?

```
SELECT COUNT(film_id) as numberOfFilms
FROM film
WHERE description LIKE '%Crocodile%' AND description LIKE '%Shark%';
```

1 row(s) returned

numberOfFilms
10

Returned total number of 10 films.

 # 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 DISTINCT
   a.first_name,
   a.last_name,
   f.title,
   f.description,
   f.release_year
FROM
   film f
      INNER JOIN
   film_actor fa ON f.film_id = fa.film_id
      INNER JOIN
   actor a ON a.actor_id = fa.actor_id
WHERE
   description LIKE '%Crocodile%'
      AND description LIKE '%Shark%'
ORDER BY a.last_name;
```

58 row(s) returned

first_name	last_name	title	description	release_year
KIRSTEN	AKROYD	MADIGAN DORADO	A Astounding Character Study of a A Shark And…	2006
KIM	ALLEN	CLEOPATRA DEVIL	A Fanciful Documentary of a Crocodile And a Te…	2006
AUDREY	BAILEY	WARLOCK WEREWO…	A Astounding Yarn of a Pioneer And a Crocodile…	2006
JULIA	BARRYMORE	SHOOTIST SUPERFLY	A Fast-Paced Story of a Crocodile And a A Shar…	2006
VIVIEN	BASINGER	CONNECTICUT TRAMP	A Unbelieveable Drama of a Crocodile And a M…	2006
VIVIEN	BERGEN	EXCITEMENT EVE	A Brilliant Documentary of a Monkey And a Car…	2006
KARL	BERRY	CONNECTICUT TRAMP	A Unbelieveable Drama of a Crocodile And a M…	2006
KARL	BERRY	CLEOPATRA DEVIL	A Fanciful Documentary of a Crocodile And a Te…	2006
HENRY	BERRY	CONNECTICUT TRAMP	A Unbelieveable Drama of a Crocodile And a M…	2006
LAURA	BRODY	DANCING FEVER	A Stunning Story of a Explorer And a Forensic P…	2006
JOHNNY	CAGE	CONNECTICUT TRAMP	A Unbelieveable Drama of a Crocodile And a M…	2006
ZERO	CAGE	HONEY TIES	A Taut Story of a Waitress And a Crocodile who…	2006
JON	CHASE	MADIGAN DORADO	A Astounding Character Study of a A Shark And…	2006

Result 5

 # 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 c.name, COUNT(fc.film_id) AS num_films
FROM film_category fc
LEFT JOIN category c
ON fc.category_id = c.category_id
GROUP BY c.name
HAVING COUNT(fc.film_id) BETWEEN 55 AND 65
ORDER BY COUNT(fc.film_id) DESC;
```

10 row(s) returned

name	num_films
Action	64
New	63
Drama	62
Games	61
Sci-Fi	61
Children	60
Comedy	58
Travel	57
Classics	57
Horror	56

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

```
SELECT
   c.name,
   AVG(f.replacement_cost) AS avgReplacementCost,
   AVG(f.rental_rate) AS avgRentalRate,
   ABS(AVG(f.replacement_cost) - AVG(f.rental_rate)) AS avgDifference
FROM
   category c
      INNER JOIN
   film_category fc ON c.category_id = fc.category_id
      INNER JOIN
   film f ON f.film_id = fc.film_id
GROUP BY c.name
HAVING avgDifference > 17;
```

8 row(s) returned

name	avgReplacementCost	avgRentalRate	avgDifference
Action	20.911875	2.646250	18.265625
Animation	20.126364	2.808182	17.318182
Children	20.056667	2.890000	17.166667
Classics	21.007544	2.744386	18.263158
Drama	21.086774	3.022258	18.064516
Games	20.285082	3.252295	17.032787
Sci-Fi	21.153934	3.219508	17.934426
Sports	20.395405	3.125135	17.270270

 # l) 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
   c.first_name,
   c.last_name,
   ad.phone,
   f.title,
   r.rental_date,
   r.return_date,
   f.rental_duration,
   DATEDIFF(r.return_date, r.rental_date) AS actualrentalduration
FROM
   customer c
```

```
    LEFT JOIN
  rental r ON r.customer_id = c.customer_id
    LEFT JOIN
  address ad ON c.address_id = ad.address_id
    LEFT JOIN
  inventory i ON i.inventory_id = r.inventory_id
    LEFT JOIN
  film f ON f.film_id = i.film_id
WHERE
  DATEDIFF(r.return_date, r.rental_date) > f.rental_duration
    OR r.return_date IS NULL;
```

1000 row(s) returned

first_name	last_name	phone	title	rental_date	return_date	rental_duration	actualrentalduration
MARY	SMITH	28303384290	PATIENT SISTER	2005-05-25 11:30:37	2005-06-03 12:00:37	7	9
MARY	SMITH	28303384290	MUSKETEERS WAIT	2005-06-15 00:54:12	2005-06-23 02:42:12	7	8
MARY	SMITH	28303384290	FERRIS MOTHER	2005-06-15 21:08:46	2005-06-25 02:26:46	3	10
MARY	SMITH	28303384290	YOUTH KICK	2005-06-21 06:24:45	2005-06-28 03:28:45	4	7
MARY	SMITH	28303384290	FIRE WOLVES	2005-07-08 03:17:05	2005-07-14 01:19:05	5	6
MARY	SMITH	28303384290	SATURDAY LAMBS	2005-07-08 07:33:56	2005-07-12 13:25:56	3	4
MARY	SMITH	28303384290	EXPECATIONS NATUR…	2005-07-11 10:13:46	2005-07-19 13:15:46	5	8
MARY	SMITH	28303384290	ADAPTATION HOLES	2005-08-01 08:51:04	2005-08-10 12:12:04	7	9
MARY	SMITH	28303384290	RACER EGG	2005-08-02 15:36:52	2005-08-10 16:40:52	7	8
MARY	SMITH	28303384290	FIREBALL PHILADELP…	2005-08-22 01:27:57	2005-08-27 07:01:57	4	5
MARY	SMITH	28303384290	FIREBALL PHILADELP…	2005-08-22 19:41:37	2005-08-28 22:49:37	4	6
MARY	SMITH	28303384290	BIKINI BORROWERS	2005-08-22 20:03:46	2005-08-30 01:51:46	7	8
PATRICIA	JOHNSON	838635286649	MUMMY CREATURES	2005-07-10 12:38:56	2005-07-19 17:02:56	3	9
PATRICIA	JOHNSON	838635286649	CLUELESS BUCKET	2005-07-27 14:30:42	2005-08-02 16:48:42	4	6
PATRICIA	JOHNSON	838635286649	BUTTERFLY CHOCOLAT	2005-07-27 15:23:02	2005-08-04 10:35:02	3	8
PATRICIA	JOHNSON	838635286649	HYSTERICAL GRAIL	2005-07-27 18:40:20	2005-08-02 21:07:20	5	6
PATRICIA	JOHNSON	838635286649	HOLIDAY GAMES	2005-07-29 00:12:59	2005-08-06 19:52:59	7	8
PATRICIA	JOHNSON	838635286649	TELEGRAPH VOYAGE	2005-07-29 17:14:29	2005-08-02 16:01:29	3	4
PATRICIA	JOHNSON	838635286649	TEEN APOLLO	2005-07-30 13:47:43	2005-08-08 18:52:43	3	9
PATRICIA	JOHNSON	838635286649	TIMBERLAND SKY	2005-07-30 16:21:13	2005-08-08 11:57:13	3	9

Result 12

 # m) Find the list of all customers and staff given a store id
 # Note : use a set operator, do not remove duplicates

```
SELECT store_id, first_name, last_name
FROM customer
UNION ALL
SELECT store_id, first_name, last_name
FROM staff
ORDER BY store_id;
```

601 row(s) returned

store_id	first_name	last_name
1	GABRIEL	HARDER
1	CHRISTIAN	JUNG
1	JIMMIE	EGGLESTON
1	CASEY	MENA
1	WADE	DELVALLE
1	LAURA	RODRIGUEZ
1	MELISSA	KING
1	MARTHA	GONZALEZ
1	TERESA	ROGERS
1	JOAN	COOPER
1	LORI	WOOD
1	NORMA	GONZALES
1	CRYSTAL	FORD
1	KIM	CRUZ
1	ELLEN	SIMPSON
1	EMMA	BOYD
1	APRIL	BURNS
1	VERONICA	STONE
1	LYNN	PAYNE
1	RENEE	LANE

######## QUESTION 2 ######## – { 10 Points }

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

<u>SHOW all the actors and customers' first name, which are same as the first name of the actor with ID 8:</u>

```
SELECT a.actor_id, a.first_name AS actfirst, a.last_name AS actlast, cusfirstname, cuslastname
FROM actor a
JOIN
(SELECT first_name AS cusfirstname, last_name AS cuslastname
FROM customer
WHERE first_name = (SELECT first_name
FROM actor
WHERE actor_id = 8)) AS t1
ON a.first_name = t1.cusfirstname;
```

3 row(s) returned

actor_id	actfirst	actlast	cusfirstname	cuslastname
8	MATTHEW	JOHANSSON	MATTHEW	MAHAN
103	MATTHEW	LEIGH	MATTHEW	MAHAN
181	MATTHEW	CARREY	MATTHEW	MAHAN

<u>If only return the customer's first name, which are same as the first name of the actor with ID8:</u>

```
SELECT
   a.actor_id,
   a.first_name AS actorFirstName,
   a.last_name AS actorLastName,
   c.first_name AS customerFirstName,
   c.last_name AS customerLastName
FROM
         actor AS a, customer AS c
WHERE
         a.first_name = c.first_name AND a.actor_id = 8;
```

1 row(s) returned

actor_id	actorFirstName	actorLastName	customerFirstName	customerLastName
8	MATTHEW	JOHANSSON	MATTHEW	MAHAN

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

```
SELECT
   c.customer_id, c.first_name, c.last_name, p.amount
FROM
   customer c,
   payment p
WHERE
   p.amount > (SELECT
         AVG(amount)
      FROM
         payment);
```

1000 row(s) returned

customer_id	first_name	last_name	amount
1	MARY	SMITH	5.99
2	PATRICIA	JOHNSON	5.99
3	LINDA	WILLIAMS	5.99
▶ 4	BARBARA	JONES	5.99
5	ELIZABETH	BROWN	5.99
6	JENNIFER	DAVIS	5.99
7	MARIA	MILLER	5.99
8	SUSAN	WILSON	5.99
9	MARGARET	MOORE	5.99
10	DOROTHY	TAYLOR	5.99
11	LISA	ANDERS…	5.99
12	NANCY	THOMAS	5.99
13	KAREN	JACKSON	5.99
14	BETTY	WHITE	5.99
15	HELEN	HARRIS	5.99
16	SANDRA	MARTIN	5.99
17	DONNA	THOMPS…	5.99
18	CAROL	GARCIA	5.99
19	RUTH	MARTINEZ	5.99

 # c) List customers who have rented movies at least once
 # Note: use IN clause

```
SELECT customer_id, first_name, last_name
FROM customer
WHERE customer_id IN (SELECT DISTINCT customer_id FROM rental);
```

599 row(s) returned

first_name	last_name	customer_id
MARY	SMITH	1
PATRICIA	JOHNSON	2
LINDA	WILLIAMS	3
BARBARA	JONES	4
ELIZABETH	BROWN	5
JENNIFER	DAVIS	6
MARIA	MILLER	7
SUSAN	WILSON	8
MARGARET	MOORE	9
DOROTHY	TAYLOR	10
LISA	ANDERS…	11
NANCY	THOMAS	12
KAREN	JACKSON	13
BETTY	WHITE	14
HELEN	HARRIS	15
SANDRA	MARTIN	16
DONNA	THOMPS…	17
CAROL	GARCIA	18
RUTH	MARTINEZ	19

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

```
SELECT MAX(amount), MIN(amount), AVG(amount)
FROM payment;
```

1 row(s) returned

MAX(amount)	MIN(amount)	AVG(amount)
11.99	0.00	4.200667

######## QUESTION 3 ######## – { 5 Points }
 # a) Create a view called actors_portfolio which contains information about actors and films (including titles and category).

```
DROP VIEW IF EXISTS actors_portfolio;

CREATE VIEW actors_portfolio AS
   SELECT
```

```
        a.actor_id,
        a.first_name,
        a.last_name,
        f.film_id,
        f.title,
        f.description,
        f.release_year,
        f.language_id,
        f.original_language_id,
        f.rental_duration,
        f.rental_rate,
        f.length,
        f.replacement_cost,
        f.rating,
        f.special_features,
        c.name AS category_name
    FROM
        actor a
            LEFT JOIN
        film_actor fa ON a.actor_id = fa.actor_id
            LEFT JOIN
        film f ON f.film_id = fa.film_id
            LEFT JOIN
        film_category fc ON fc.film_id = f.film_id
            LEFT JOIN
        category c ON c.category_id = fc.category_id;
```

0 row(s) affected

 # 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
    first_name = 'ADAM'
        AND last_name = 'GRANT';
```

18 row(s) returned

actor_id	first_name	last_name	film_id	title	description	release_year	language_id	original_language_id	rental_duration	rental_rate	length
71	ADAM	GRANT	26	ANNIE IDENTITY	A Amazing Panorama of a Pastry Chef And a B…	2006	1	NULL	3	0.99	86
71	ADAM	GRANT	52	BALLROOM MOCKINGBIRD	A Thrilling Documentary of a Composer And a…	2006	1	NULL	6	0.99	173
71	ADAM	GRANT	233	DISCIPLE MOTHER	A Touching Reflection of a Mad Scientist And a…	2006	1	NULL	3	0.99	141
71	ADAM	GRANT	317	FIREBALL PHILADELPHIA	A Amazing Yarn of a Dentist And a A Shark who…	2006	1	NULL	4	0.99	148
71	ADAM	GRANT	359	GLADIATOR WESTWARD	A Astounding Reflection of a Squirrel And a Su…	2006	1	NULL	6	4.99	173
71	ADAM	GRANT	362	GLORY TRACY	A Amazing Saga of a Woman And a Womanizer…	2006	1	NULL	7	2.99	115
71	ADAM	GRANT	385	GROUNDHOG UNCUT	A Brilliant Panorama of a Astronaut And a Techn…	2006	1	NULL	6	4.99	139
71	ADAM	GRANT	399	HAPPINESS UNITED	A Action-Packed Panorama of a Husband And a…	2006	1	NULL	6	2.99	100
71	ADAM	GRANT	450	IDOLS SNATCHERS	A Insightful Drama of a Car And a Composer wh…	2006	1	NULL	5	2.99	84
71	ADAM	GRANT	532	LOSER HUSTLER	A Stunning Drama of a Robot And a Feminist w…	2006	1	NULL	5	4.99	80
71	ADAM	GRANT	560	MARS ROMAN	A Boring Drama of a Car And a Dog who must S…	2006	1	NULL	6	0.99	62
71	ADAM	GRANT	574	MIDNIGHT WESTWARD	A Taut Reflection of a Husband And a A Shark w…	2006	1	NULL	3	0.99	86
71	ADAM	GRANT	638	OPERATION OPERATION	A Intrepid Character Study of a Man And a Frisb…	2006	1	NULL	7	2.99	156

 # c) Insert a new movie titled Data Hero in Sci-Fi Category starring ADAM GRANT

```
INSERT INTO actors_portfolio (actor, title, category)
VALUES('ADAM GRANT','Data Hero','Sci-Fi');
```

1 row(s) affected

######## QUESTION 4 ######## – { 5 Points }
 # a) Extract the street number (characters 1 through 4) from customer addressLine1

```
SELECT
   c.first_name,
   c.last_name,
   SUBSTRING(address, 1, 4) AS streetNumber
FROM
   customer AS c
      INNER JOIN
   address USING (address_id);
```

599 row(s) returned

first_name	last_name	streetNumber
MARY	SMITH	1913
PATRICIA	JOHNSON	1121
LINDA	WILLIAMS	692
BARBARA	JONES	1566
ELIZABETH	BROWN	53 I
JENNIFER	DAVIS	1795
MARIA	MILLER	900
SUSAN	WILSON	478
MARGARET	MOORE	613
DOROTHY	TAYLOR	1531
LISA	ANDERS…	1542
NANCY	THOMAS	808
KAREN	JACKSON	270

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

```
SELECT *
FROM actor
WHERE last_name LIKE 'A%' OR last_name LIKE 'B%' OR last_name LIKE 'C%';
```

44 row(s) returned

actor_id	first_name	last_name	last_update
58	CHRISTIAN	AKROYD	2006-02-15 04:34:33
92	KIRSTEN	AKROYD	2006-02-15 04:34:33
182	DEBBIE	AKROYD	2006-02-15 04:34:33
118	CUBA	ALLEN	2006-02-15 04:34:33
145	KIM	ALLEN	2006-02-15 04:34:33
194	MERYL	ALLEN	2006-02-15 04:34:33
76	ANGELINA	ASTAIRE	2006-02-15 04:34:33
112	RUSSELL	BACALL	2006-02-15 04:34:33
67	JESSICA	BAILEY	2006-02-15 04:34:33
190	AUDREY	BAILEY	2006-02-15 04:34:33
115	HARRISON	BALE	2006-02-15 04:34:33
187	RENEE	BALL	2006-02-15 04:34:33
47	JULIA	BARRYM…	2006-02-15 04:34:33
158	VIVIEN	BASINGER	2006-02-15 04:34:33

 # c) Find film titles that contains exactly 10 characters

```
SELECT title
FROM film
WHERE title REGEXP '^.{10}$';
```

Or

```
SELECT title
FROM film
WHERE CHAR_LENGTH(title) = 10;
```

Both queries ways give
66 row(s) returned

title
ALONE TRIP
BASIC EASY
BUGSY SONG
CAUSE DATE
CHILL LUCK
CLUE GRAIL
CRAZY HOME
DATE SPEED
DAZED PUNK
DOLLS RAGE
DOZEN LION
DUCK RACER
DUMBO LUST

 # d) Format a payment_date using the following format e.g "22/1/2016"

```
SELECT DATE_FORMAT(payment_date, '%d-%m-%Y') AS Newformat
FROM payment;
```

1000 row(s) returned

Newformat
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
08-07-2005
09-07-2005
09-07-2005

 # e) Find the number of days between two date values rental_date & return_date ‘

```
SELECT DATEDIFF(return_date, rental_date) AS numberOfDays
FROM rental;
```

1000 row(s) returned

numberOfDays
2
4
8
10
9
3
5
3
3
6
8
5
5
1

######## QUESTION 5 ######## – { 20 Points }
 # Provide 5 additional queries and indicate the specific business use cases they address.

1. **Which movie has been rented the most in these several years? Which film could potentially bring more benefits and profits? It could help business to obtain the information on which movie category is the most potentially popular.**

```
SELECT EXTRACT(year FROM r.rental_date) AS whichYear, c.name AS categoryNAME,
COUNT(*) AS num
FROM rental r
INNER JOIN inventory i
ON r.inventory_id = i.inventory_id
INNER JOIN film f
ON i.film_id = f.film_id
INNER JOIN film_category fc
ON f.film_id = fc.film_id
INNER JOIN category c
ON fc.category_id = c.category_id
GROUP BY 1,2
ORDER BY 1, 3 DESC;
```

32 row(s) returned

whichYear	categoryNAME	num
2005	Sports	1164
2005	Animation	1145
2005	Action	1095
2005	Sci-Fi	1093
2005	Family	1083
2005	Drama	1053
2005	Documentary	1044
2005	Foreign	1022
2005	Games	955
2005	Children	939
2005	Comedy	932
2005	Classics	930
2005	New	927
2005	Horror	834
2005	Travel	827
2005	Music	819
2006	Animation	21
2006	Action	17
2006	Sports	15
2006	Games	14

2. **Which actors are the most popular actors on market in the recent years? It could help business to potentially evaluate the value of an actor.**

```
SELECT CONCAT(a.first_name,' ',a.last_name) AS actor, COUNT(f.film_id) AS
numofFilms
FROM actor a
INNER JOIN film_actor fa
ON a.actor_id = fa.actor_id
LEFT JOIN film f
ON fa.film_id = f.film_id
GROUP BY 1
ORDER BY 2 DESC;
```

199 row(s) returned

actor	numofFilms
SUSAN DAVIS	54
GINA DEGENERES	42
WALTER TORN	41
MARY KEITEL	40
MATTHEW CARREY	39
SANDRA KILMER	37
SCARLETT DAMON	36
GROUCHO DUNST	35
HENRY BERRY	35
UMA WOOD	35
VAL BOLGER	35
VIVIEN BASINGER	35
ANGELA WITHER…	35
ANGELA HUDSON	34
SIDNEY CROWE	34
KIRSTEN AKROYD	34
WARREN NOLTE	34
JAYNE NOLTE	34
ALBERT JOHANS…	33

3. **Help business to obtain the information for staff and customers' name if the customer and staff have the same address id.**

```
SELECT
   c.first_name as cust_first, c.last_name as cust_last,
   s.store_id as staff_first, s.last_name as staff_last
FROM
   customer c
INNER JOIN
   staff s
WHERE c.address_id = s.address_id
```

0 row(s) returned

cust_first	cust_last	staff_first	staff_last

4. **Help business to obtain the information on which staff process the specific payment. Sort by payment amount in descending order.**

```
SELECT CONCAT(s.first_name,' ',s.last_name) AS staff, p.amount, p.payment_id, s.email,
s.last_update
FROM staff s INNER JOIN payment p
WHERE p.amount >= 1
ORDER BY amount desc
LIMIT 100
```

100 row(s) returned

staff	amount	payment_id	email	last_update
Jon Stephens	11.99	9803	Jon.Stephens@sakilastaff.com	2006-02-15 03:57:16
Mike Hillyer	11.99	6409	Mike.Hillyer@sakilastaff.com	2006-02-15 03:57:16
Mike Hillyer	11.99	3146	Mike.Hillyer@sakilastaff.com	2006-02-15 03:57:16
Jon Stephens	11.99	15821	Jon.Stephens@sakilastaff.com	2006-02-15 03:57:16
Jon Stephens	11.99	6409	Jon.Stephens@sakilastaff.com	2006-02-15 03:57:16
▶ Mike Hillyer	11.99	5550	Mike.Hillyer@sakilastaff.com	2006-02-15 03:57:16
Mike Hillyer	11.99	15821	Mike.Hillyer@sakilastaff.com	2006-02-15 03:57:16
Mike Hillyer	11.99	5280	Mike.Hillyer@sakilastaff.com	2006-02-15 03:57:16
Jon Stephens	11.99	3146	Jon.Stephens@sakilastaff.com	2006-02-15 03:57:16
Mike Hillyer	11.99	342	Mike.Hillyer@sakilastaff.com	2006-02-15 03:57:16
Jon Stephens	11.99	15850	Jon.Stephens@sakilastaff.com	2006-02-15 03:57:16
Jon Stephens	11.99	5550	Jon.Stephens@sakilastaff.com	2006-02-15 03:57:16

5. **Help business to obtain the information on how many movies in each language within every category of the movie.**

```
SELECT  c.name, l.name, COUNT(l.language_id) AS numberOfMovies
FROM
language l LEFT JOIN film f ON l.language_id = f.language_id
LEFT JOIN film_category fc ON f.film_id = fc.film_id
LEFT JOIN category c ON fc.category_id = c.category_id
GROUP BY 1,2
```

21 row(s) returned

name	name	numberOfMovies
Documentary	English	68
Horror	English	56
Family	English	69
Foreign	English	73
Comedy	English	58
Sports	English	74
Music	English	51
Classics	English	57
Animation	English	66
Action	English	64
New	English	63
Sci-Fi	English	61
Drama	English	62
Travel	English	57
Games	English	61
Children	English	60
NULL	Italian	1
NULL	Japan…	1
NULL	Mand…	1
NULL	French	1
NULL	German	1

This Sakila data set collects most of the movies in English, so the resulting language is mostly English. However, as the data set gets larger or for some other business use cases, this query will still be beneficial to visualize the data and give some insights to the business.