In [58]:
```
import warnings
warnings.filterwarnings("ignore")
```

In [59]:
```
#import statements
import pymysql
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

In [60]:
```
#!/usr/bin/python3

# Open database connection
db = pymysql.connect(host="localhost",user="root",password="rootroot",database="sakila")

# prepare a cursor object using cursor() method
cursor = db.cursor()

# execute SQL query using execute() method.
cursor.execute("SELECT VERSION()")

# Fetch a single row using fetchone() method.
data = cursor.fetchone()

print ("Database version : %s " % data)

# disconnect from server
db.close()
```

```
Database version : 8.0.26
```

In [61]:
```
def qry(sql):

    # Open database connection
    connection = pymysql.connect(host="localhost",user="root",password="rootroot",database="sakila")
    df = pd.read_sql(sql, connection)

    # disconnect from server
    connection.close()

    # return data.
    return df
```

In [62]:
```
def qry_execute_only(sql):

    # Open database connection
    connection = pymysql.connect(host="localhost",user="root",password="rootroot",database="sakila")
    cursor = connection.cursor()
    cursor.execute(sql)
    connection.commit()

    connection.close()
```

Part B

In [63]:

```
sql_1a = '''
show databases;
'''

df_1a = qry(sql_1a)
df_1a.head()
```

Out[63]:

	Database
0	classicmodels
1	information_schema
2	mysql
3	performance_schema
4	sakila

In [64]:

```
sql_1b= '''
USE sakila;
'''

df_1b = qry_execute_only(sql_1b)
df_1b
```

In [65]:
```
sql_1c = '''
show tables;
'''

df_1c = qry(sql_1c)
df_1c.head()
```

Out[65]:

	Tables_in_sakila
0	actor
1	actor_info
2	address
3	category
4	city

In [66]:
```
sql_1d = '''
DESCRIBE actor;
'''

df_1d = qry(sql_1d)
df_1d.head()
```

Out[66]:

	Field	Type	Null	Key	Default	Extra
0	actor_id	smallint unsigned	NO	PRI	None	auto_increment
1	first_name	varchar(45)	NO		None	
2	last_name	varchar(45)	NO	MUL	None	
3	last_update	timestamp	NO		CURRENT_TIMESTAMP	DEFAULT_GENERATED on update CURRENT_TIMESTAMP

In [67]:

```
sql_1e = '''
SELECT COUNT(*) FROM actor;
'''

df_1e = qry(sql_1e)
df_1e
```

Out[67]:

	COUNT(*)
0	202

In [16]:

```
sql_1f = '''
SELECT first_name, last_name FROM actor;
'''
df_1f = qry(sql_1f)
df_1f.head()
```

Out[16]:

	first_name	last_name
0	PENELOPE	GUINESS
1	NICK	WAHLBERG
2	ED	CHASE
3	JENNIFER	DAVIS
4	JOHNNY	LOLLOBRIGIDA

In [197]:

```
sql_1g = '''
INSERT INTO `actor` (first_name, last_name) VALUES ("HAOTIAN","");
'''
df_1g = qry_execute_only(sql_1g)
df_1g
```

In [17]:

```
sql_test = '''
SELECT * FROM actor
'''
df_test = qry(sql_test)
df_test.head()
```

Out[17]:

	actor_id	first_name	last_name	last_update
0	1	PENELOPE	GUINESS	2006-02-15 04:34:33
1	2	NICK	WAHLBERG	2006-02-15 04:34:33
2	3	ED	CHASE	2006-02-15 04:34:33
3	4	JENNIFER	DAVIS	2006-02-15 04:34:33
4	5	JOHNNY	LOLLOBRIGIDA	2006-02-15 04:34:33

In [199]:

```
sql_1h = '''
UPDATE actor SET last_name = "WU" WHERE last_name = "";
'''
df_1h = qry_execute_only(sql_1h)
df_1h
```

In [201]:

```
sql_1i = '''
DELETE FROM actor WHERE last_name = "Wu";
'''
df_1i = qry_execute_only(sql_1i)
df_1i
```

In [18]:
```
sql_1j = '''
SELECT title, description FROM film WHERE rating = "PG-13";
'''
df_1j = qry(sql_1j)
df_1j.head()
```

Out[18]:

	title	description
0	AIRPLANE SIERRA	A Touching Saga of a Hunter And a Butler who m...
1	ALABAMA DEVIL	A Thoughtful Panorama of a Database Administra...
2	ALTER VICTORY	A Thoughtful Drama of a Composer And a Feminis...
3	ANTHEM LUKE	A Touching Panorama of a Waitress And a Woman ...
4	APOLLO TEEN	A Action-Packed Reflection of a Crocodile And ...

In [19]:
```
sql_1k = '''
SELECT title, description FROM film WHERE rating IN ("PG-13","PG");
'''
df_1k = qry(sql_1k)
df_1k.head()
```

Out[19]:

	title	description
0	ACADEMY DINOSAUR	A Epic Drama of a Feminist And a Mad Scientist...
1	AGENT TRUMAN	A Intrepid Panorama of a Robot And a Boy who m...
2	AIRPLANE SIERRA	A Touching Saga of a Hunter And a Butler who m...
3	ALABAMA DEVIL	A Thoughtful Panorama of a Database Administra...
4	ALASKA PHANTOM	A Fanciful Saga of a Hunter And a Pastry Chef ...

In [20]:
```
sql_1l = '''
SELECT * FROM payment WHERE amount >=2 AND amount <= 7;
'''
df_1l = qry(sql_1l)
df_1l.head()
```

Out[20]:

	payment_id	customer_id	staff_id	rental_id	amount	payment_date	last_update
0	1	1	1	76.0	2.99	2005-05-25 11:30:37	2006-02-15 22:12:30
1	3	1	1	1185.0	5.99	2005-06-15 00:54:12	2006-02-15 22:12:30
2	6	1	1	1725.0	4.99	2005-06-16 15:18:57	2006-02-15 22:12:30
3	7	1	1	2308.0	4.99	2005-06-18 08:41:48	2006-02-15 22:12:30
4	9	1	1	3284.0	3.99	2005-06-21 06:24:45	2006-02-15 22:12:30

In [21]:
```
sql_1l_2 = '''
SELECT * FROM payment WHERE amount BETWEEN 2 AND 7;
'''
df_1l_2 = qry(sql_1l_2)
df_1l_2.head()
```

Out[21]:

	payment_id	customer_id	staff_id	rental_id	amount	payment_date	last_update
0	1	1	1	76.0	2.99	2005-05-25 11:30:37	2006-02-15 22:12:30
1	3	1	1	1185.0	5.99	2005-06-15 00:54:12	2006-02-15 22:12:30
2	6	1	1	1725.0	4.99	2005-06-16 15:18:57	2006-02-15 22:12:30
3	7	1	1	2308.0	4.99	2005-06-18 08:41:48	2006-02-15 22:12:30
4	9	1	1	3284.0	3.99	2005-06-21 06:24:45	2006-02-15 22:12:30

In [22]:
```
sql_2a = '''
SELECT title, description FROM film WHERE rating = "PG-13";
'''
df_2a = qry(sql_2a)
df_2a.head()
```

Out[22]:

	title	description
0	AIRPLANE SIERRA	A Touching Saga of a Hunter And a Butler who m...
1	ALABAMA DEVIL	A Thoughtful Panorama of a Database Administra...
2	ALTER VICTORY	A Thoughtful Drama of a Composer And a Feminis...
3	ANTHEM LUKE	A Touching Panorama of a Waitress And a Woman ...
4	APOLLO TEEN	A Action-Packed Reflection of a Crocodile And ...

In [23]:
```
sql_2b = '''
SELECT title, description FROM film WHERE rating IN ("PG-13","PG");
'''
df_2b = qry(sql_2b)
df_2b.head()
```

Out[23]:

	title	description
0	ACADEMY DINOSAUR	A Epic Drama of a Feminist And a Mad Scientist...
1	AGENT TRUMAN	A Intrepid Panorama of a Robot And a Boy who m...
2	AIRPLANE SIERRA	A Touching Saga of a Hunter And a Butler who m...
3	ALABAMA DEVIL	A Thoughtful Panorama of a Database Administra...
4	ALASKA PHANTOM	A Fanciful Saga of a Hunter And a Pastry Chef ...

In [24]:
```
sql_2c_1 = '''
SELECT * FROM payment WHERE amount >=2 AND amount <= 7;
'''
df_2c_1 = qry(sql_2c_1)
df_2c_1.head()
```

Out[24]:

	payment_id	customer_id	staff_id	rental_id	amount	payment_date	last_update
0	1	1	1	76.0	2.99	2005-05-25 11:30:37	2006-02-15 22:12:30
1	3	1	1	1185.0	5.99	2005-06-15 00:54:12	2006-02-15 22:12:30
2	6	1	1	1725.0	4.99	2005-06-16 15:18:57	2006-02-15 22:12:30
3	7	1	1	2308.0	4.99	2005-06-18 08:41:48	2006-02-15 22:12:30
4	9	1	1	3284.0	3.99	2005-06-21 06:24:45	2006-02-15 22:12:30

In [25]:
```
sql_2c_2 = '''
SELECT * FROM payment WHERE amount BETWEEN 2 AND 7;
'''
df_2c_2 = qry(sql_2c_2)
df_2c_2.head()
```

Out[25]:

	payment_id	customer_id	staff_id	rental_id	amount	payment_date	last_update
0	1	1	1	76.0	2.99	2005-05-25 11:30:37	2006-02-15 22:12:30
1	3	1	1	1185.0	5.99	2005-06-15 00:54:12	2006-02-15 22:12:30
2	6	1	1	1725.0	4.99	2005-06-16 15:18:57	2006-02-15 22:12:30
3	7	1	1	2308.0	4.99	2005-06-18 08:41:48	2006-02-15 22:12:30
4	9	1	1	3284.0	3.99	2005-06-21 06:24:45	2006-02-15 22:12:30

In [26]:

```
sql_2d_1 = '''
SELECT * FROM address WHERE phone LIKE "%589%";
'''
df_2d_1 = qry(sql_2d_1)
df_2d_1.head()
```

Out[26]:

	address_id	address	address2	district	city_id	postal_code	phone	location	last_updat
0	4	1411 Lillydale Drive	None	QLD	576		6172235589	b'\x00\x00\x00\x00\x01\x01\x00\x00\x00[\r\xe44...	2014-09-2 22:30:0
1	153	782 Mosul Street		Massachusetts	94	25545	885899703621	b'\x00\x00\x00\x00\x01\x01\x00\x00\x00\xe9\xc4...	2014-09-2 22:33:4
2	333	1860 Taguig Loop		West Java	119	59550	38158430589	b'\x00\x00\x00\x00\x01\x01\x00\x00\x00B\xac\xa...	2014-09-2 22:31:3
3	388	368 Hunuco Boulevard		Namibe	360	17165	106439158941	b'\x00\x00\x00\x00\x01\x01\x00\x00\x00\xb5\x85...	2014-09-2 22:30:0
4	492	185 Mannheim Lane		Stavropol	408	23661	589377568313	b'\x00\x00\x00\x00\x01\x01\x00\x00\x003>\x82\x...	2014-09-2 22:32:5

In [212]:

```
sql_2d_2 = '''
SELECT * FROM address WHERE phone LIKE "140%";
'''
df_2d_2 = qry(sql_2d_2)
df_2d_2
```

Out[212]:

	address_id	address	address2	district	city_id	postal_code	phone	location	last_update
0	3	23 Workhaven Lane	None	Alberta	300		14033335568	b'\x00\x00\x00\x00\x01\x01\x00\x00\x00\xcd\xc4...	2014-09-25 22:30:27

In [213]:

```
sql_2d_3 = '''
SELECT * FROM address WHERE phone LIKE "%589";
'''
df_2d_3 = qry(sql_2d_3)
df_2d_3
```

Out[213]:

	address_id	address	address2	district	city_id	postal_code	phone	location	last_update
0	4	1411 Lillydale Drive	None	QLD	576		6172235589	b'\x00\x00\x00\x00\x01\x01\x00\x00\x00[\r\xe44...	2014-09-25 22:30:09
1	333	1860 Taguig Loop		West Java	119	59550	38158430589	b'\x00\x00\x00\x00\x01\x01\x00\x00\x00B\xac\xa...	2014-09-25 22:31:32

In [214]:

```
sql_2e = '''
SELECT first_name, last_name, email FROM staff WHERE password IS NULL;
'''
df_2e = qry(sql_2e)
df_2e
```

Out[214]:

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

In [27]:

```
sql_2f = '''
SELECT * FROM film WHERE title LIKE "%ZOO%" AND rental_duration >= 4;
'''
df_2f = qry(sql_2f)
df_2f.head()
```

Out[27]:

	film_id	title	description	release_year	language_id	original_language_id	rental_duration	rental_rate	length	replacement_cost	ra
0	568	MEMENTO ZOOLANDER	A Touching Epistle of a Squirrel And a Explore...	2006	1	None	4	4.99	77	11.99	
1	924	UNFORGIVEN ZOOLANDER	A Taut Epistle of a Monkey And a Sumo Wrestler...	2006	1	None	7	0.99	129	15.99	
2	999	ZOOLANDER FICTION	A Fateful Reflection of a Waitress And a Boat ...	2006	1	None	5	2.99	101	28.99	

In [65]:
```
sql_2g = '''
SELECT rental_rate/rental_duration*14 AS two_week_cost FROM film WHERE title = "ACADEMY DINOSAUR";
'''
df_2g = qry(sql_2g)
df_2g
```

Out[65]:

	two_week_cost
0	2.31

In [28]:
```
sql_2h = '''
SELECT DISTINCT district FROM address WHERE address IS NOT NULL;
'''
df_2h = qry(sql_2h)
df_2h.head()
```

Out[28]:

	district
0	Alberta
1	QLD
2	Nagasaki
3	California
4	Attika

In [29]:
```
sql_2i = '''
SELECT customer_id, create_date FROM customer ORDER BY create_date DESC LIMIT 10;
'''
df_2i = qry(sql_2i)
df_2i.head()
```

Out[29]:

	customer_id	create_date
0	275	2006-02-14 22:04:37
1	281	2006-02-14 22:04:37
2	278	2006-02-14 22:04:37
3	276	2006-02-14 22:04:37
4	599	2006-02-14 22:04:37

In [68]:
```
sql_3a = '''
SELECT COUNT(*) FROM film;
'''
df_3a = qry(sql_3a)
df_3a
```

Out[68]:

	COUNT(*)
0	1000

In [69]:
```
sql_3b = '''
SELECT MAX(amount), MIN(amount) FROM payment;
'''
df_3b = qry(sql_3b)
df_3b
```

Out[69]:

	MAX(amount)	MIN(amount)
0	11.99	0.0

In [70]:

```
sql_3c = '''
SELECT COUNT(customer_id) FROM payment WHERE payment_date BETWEEN CAST('2005-02-01' AS DATE) AND CAST('2005-05-31' AS
'''
df_3c = qry(sql_3c)
df_3c
```

Out[70]:

	COUNT(customer_id)
0	994

In [30]:

```
sql_3d = '''
SELECT * FROM film WHERE replacement_cost > 15 OR rental_duration BETWEEN 6 AND 10;
'''
df_3d = qry(sql_3d)
df_3d.head()
```

Out[30]:

	film_id	title	description	release_year	language_id	original_language_id	rental_duration	rental_rate	length	replacement_cost	ra
0	1	ACADEMY DINOSAUR	A Epic Drama of a Feminist And a Mad Scientist...	2006	6	None	6	0.99	86	20.99	
1	3	ADAPTATION HOLES	A Astounding Reflection of a Lumberjack And a ...	2006	1	None	7	2.99	50	18.99	
2	4	AFFAIR PREJUDICE	A Fanciful Documentary of a Frisbee And a Lumb...	2006	1	None	5	2.99	117	26.99	
3	5	AFRICAN EGG	A Fast-Paced Documentary of a Pastry Chef And ...	2006	1	None	6	2.99	130	22.99	
4	6	AGENT TRUMAN	A Intrepid Panorama of a Robot And a Boy who m...	2006	1	None	3	2.99	169	17.99	

In [72]:
```
sql_3e = '''
SELECT SUM(amount) FROM payment WHERE payment_date BETWEEN CAST("2005-01-01" AS DATE) AND CAST("2005-12-31" AS DATE);
'''
df_3e = qry(sql_3e)
df_3e
```

Out[72]:

	SUM(amount)
0	66902.33

In [73]:
```
sql_3f = '''
SELECT AVG(replacement_cost) AS avg_replacement_cost FROM film;
'''
df_3f = qry(sql_3f)
df_3f
```

Out[73]:

	avg_replacement_cost
0	19.984

In [74]:
```
sql_3g = '''
SELECT STD(rental_rate) AS std_rental_rate FROM film;
'''
df_3g = qry(sql_3g)
df_3g
```

Out[74]:

	std_rental_rate
0	1.64557

In [73]:

```
sql_3h = '''
SELECT (MAX(rental_duration) + MIN(rental_duration)) / 2 AS midrange_rental_duration FROM film;
'''
df_3h = qry(sql_3h)
df_3h
```

Out[73]:

	midrange_rental_duration
0	5.0

In [31]:

```
sql_4a = '''
SELECT first_name, last_name FROM customer ORDER BY last_name, first_name;
'''
df_4a = qry(sql_4a)
df_4a.head()
```

Out[31]:

	first_name	last_name
0	RAFAEL	ABNEY
1	NATHANIEL	ADAM
2	KATHLEEN	ADAMS
3	DIANA	ALEXANDER
4	GORDON	ALLARD

In [77]:

```
sql_4b = '''
SELECT rating, COUNT(*) AS COUNT_MOVIES FROM film GROUP BY rating HAVING rating IN ("G","NC-17","PG-13","PG","R") ;
'''
df_4b = qry(sql_4b)
df_4b
```

Out[77]:

	rating	COUNT_MOVIES
0	PG	194
1	G	178
2	NC-17	210
3	PG-13	223
4	R	195

In [32]:

```
sql_4c = '''
SELECT district, COUNT(address) FROM address GROUP BY district;
'''
df_4c = qry(sql_4c)
df_4c.head()
```

Out[32]:

	district	COUNT(address)
0	Alberta	2
1	QLD	2
2	Nagasaki	1
3	California	9
4	Attika	1

In [33]:

```
sql_4d = '''
SELECT DISTINCT title FROM film WHERE rental_rate > 1 ORDER BY title DESC;
'''
df_4d = qry(sql_4d)
df_4d.head()
```

Out[33]:

	title
0	ZORRO ARK
1	ZOOLANDER FICTION
2	YENTL IDAHO
3	WYOMING STORM
4	WRONG BEHAVIOR

In [80]:

```
sql_4e = '''
SELECT DISTINCT(title) FROM film WHERE rating = "R" ORDER BY replacement_cost DESC LIMIT 2;
'''
df_4e = qry(sql_4e)
df_4e
```

Out[80]:

	title
0	CHARIOTS CONSPIRACY
1	CUPBOARD SINNERS

In [81]:

```
sql_4f = '''
SELECT rental_rate, COUNT(rental_rate) AS COUNT FROM film GROUP BY rental_rate ORDER BY COUNT DESC LIMIT 1;
'''
df_4f = qry(sql_4f)
df_4f
```

Out[81]:

	rental_rate	COUNT
0	0.99	341

In [95]:

```
sql_4g = '''
SELECT * FROM film WHERE length > 50 AND special_features = "Commentaries" ORDER BY length DESC LIMIT 2;
'''
df_4g = qry(sql_4g)
df_4g
```

Out[95]:

	film_id	title	description	release_year	language_id	original_language_id	rental_duration	rental_rate	length	replacement_cost	rat
0	182	CONTROL ANTHEM	A Fateful Documentary of a Robot And a Student...	2006	1	None	7	4.99	185	9.99	
1	841	STAR OPERATION	A Insightful Character Study of a Girl And a C...	2006	1	None	5	2.99	181	9.99	

In [83]:
```
sql_4h = '''
SELECT release_year, COUNT(film_id) AS total FROM film GROUP BY release_year HAVING total > 2;
'''
df_4h = qry(sql_4h)
df_4h
```

Out[83]:

	release_year	total
0	2006	1000

Part C

In [34]:
```
sql_PartC_1a = '''
SELECT a.actor_id, a.first_name, a.last_name, COUNT(f.film_id) AS COUNT FROM actor AS a
INNER JOIN film_actor AS fa ON a.actor_id = fa.actor_id
INNER JOIN film AS f ON fa.film_id = f.film_id
GROUP BY a.actor_id
HAVING COUNT > 25;
'''
df_PartC_1a = qry(sql_PartC_1a)
df_PartC_1a.head()
```

Out[34]:

	actor_id	first_name	last_name	COUNT
0	85	MINNIE	ZELLWEGER	31
1	90	SEAN	GUINESS	33
2	64	RAY	JOHANSSON	30
3	123	JULIANNE	DENCH	32
4	41	JODIE	DEGENERES	29

In [35]:

```
sql_PartC_1b = '''
SET SQL_SAFE_UPDATES=0;
'''
qry_execute_only(sql_PartC_1b)

sql_PartC_1b = '''
UPDATE film SET language_id=6 WHERE title LIKE "%ACADEMY%";
'''
qry_execute_only(sql_PartC_1b)

sql_PartC_1b = '''
SELECT a.actor_id, a.first_name, a.last_name, f.film_id, f.title, f.language_id, l.name FROM actor AS a
INNER JOIN film_actor AS fa ON a.actor_id = fa.actor_id
INNER JOIN film AS f ON fa.film_id = f.film_id
INNER JOIN language AS l ON f.language_id = l.language_id
WHERE l.name = "German";
'''
df_PartC_1b = qry(sql_PartC_1b)
df_PartC_1b.head()
```

Out[35]:

	actor_id	first_name	last_name	film_id	title	language_id	name
0	1	PENELOPE	GUINESS	1	ACADEMY DINOSAUR	6	German
1	10	CHRISTIAN	GABLE	1	ACADEMY DINOSAUR	6	German
2	20	LUCILLE	TRACY	1	ACADEMY DINOSAUR	6	German
3	30	SANDRA	PECK	1	ACADEMY DINOSAUR	6	German
4	40	JOHNNY	CAGE	1	ACADEMY DINOSAUR	6	German

In [36]:

```
sql_PartC_1c = '''
SELECT a.actor_id, a.last_name, a.first_name, COUNT(f.film_id) AS COUNT FROM actor AS a
INNER JOIN film_actor AS fa ON a.actor_id = fa.actor_id
INNER JOIN film AS f ON fa.film_id = f.film_id
INNER JOIN film_category AS fc ON f.film_id = fc.film_id
INNER JOIN category AS c ON fc.category_id = c.category_id
WHERE c.name = "Horror"
GROUP BY a.actor_id
ORDER BY a.actor_id;
'''
df_PartC_1c = qry(sql_PartC_1c)
df_PartC_1c.head()
```

Out[36]:

	actor_id	last_name	first_name	COUNT
0	1	GUINESS	PENELOPE	3
1	4	DAVIS	JENNIFER	1
2	5	LOLLOBRIGIDA	JOHNNY	3
3	7	MOSTEL	GRACE	2
4	9	SWANK	JOE	2

In [37]:

```
sql_PartC_1d = '''
SELECT customer.customer_id, customer.first_name, customer.last_name, f.film_id, f.title, category.name, COUNT(f.film
INNER JOIN rental AS r ON customer.customer_id = r.customer_id
INNER JOIN inventory AS i ON r.inventory_id = i.inventory_id
INNER JOIN film AS f ON i.film_id = f.film_id
INNER JOIN film_category AS fc ON i.film_id = fc.film_id
INNER JOIN category ON fc.category_id = category.category_id
WHERE category.name = "Horror"
GROUP BY customer.customer_id
HAVING COUNT > 3
ORDER BY customer.customer_id;

'''
df_PartC_1d = qry(sql_PartC_1d)
df_PartC_1d.head()
```

Out[37]:

	customer_id	first_name	last_name	film_id	title	name	COUNT
0	12	NANCY	THOMAS	301	FAMILY SWEET	Horror	5
1	72	THERESA	WATSON	8	AIRPORT POLLOCK	Horror	4
2	79	RACHEL	BARNES	35	ARACHNOPHOBIA ROLLERCOASTER	Horror	4
3	81	ANDREA	HENDERSON	92	BOWFINGER GABLES	Horror	4
4	92	TINA	SIMMONS	35	ARACHNOPHOBIA ROLLERCOASTER	Horror	4

In [38]:

```
sql_PartC_1e = '''
SELECT c.customer_id, c.first_name AS c_first_name, c.last_name as c_last_name, f.film_id, f.title, a.first_name AS a
INNER JOIN rental AS r ON c.customer_id = r.customer_id
INNER JOIN inventory AS i ON r.inventory_id = i.inventory_id
INNER JOIN film AS f ON i.film_id = f.film_id
INNER JOIN film_actor AS fa ON f.film_id = fa.film_id
INNER JOIN actor AS a ON fa.actor_id = a.actor_id
WHERE a.first_name = "SCARLETT" AND a.last_name = "BENING"
GROUP BY c.customer_id
ORDER BY c.customer_id;
'''
df_PartC_1e = qry(sql_PartC_1e)
df_PartC_1e.head()
```

Out[38]:

	customer_id	c_first_name	c_last_name	film_id	title	a_first_name	a_last_name
0	1	MARY	SMITH	22	AMISTAD MIDSUMMER	SCARLETT	BENING
1	3	LINDA	WILLIAMS	263	DURHAM PANKY	SCARLETT	BENING
2	5	ELIZABETH	BROWN	22	AMISTAD MIDSUMMER	SCARLETT	BENING
3	6	JENNIFER	DAVIS	775	SEATTLE EXPECATIONS	SCARLETT	BENING
4	8	SUSAN	WILSON	882	TENENBAUMS COMMAND	SCARLETT	BENING

In [226]:
```
sql_PartC_1f = '''
SELECT customer.customer_id, customer.first_name, customer.last_name, f.film_id, f.title, category.name, addr.postal_
INNER JOIN rental AS r ON customer.customer_id = r.customer_id
INNER JOIN inventory AS i ON r.inventory_id = i.inventory_id
INNER JOIN film AS f ON i.film_id = f.film_id
INNER JOIN film_category AS fc ON i.film_id = fc.film_id
INNER JOIN category ON fc.category_id = category.category_id
INNER JOIN address AS addr ON customer.address_id = addr.address_id
WHERE postal_code = "62703" AND name = "Documentary";
'''
df_PartC_1f = qry(sql_PartC_1f)
df_PartC_1f
```

Out[226]:

	customer_id	first_name	last_name	film_id	title	name	postal_code
0	582	ANDY	VANHORN	616	NATIONAL STORY	Documentary	62703

In [26]:
```
sql_PartC_1g = '''
SELECT address, address2 FROM address WHERE address2 != "" ORDER BY address2;
'''
df_PartC_1g = qry(sql_PartC_1g)
df_PartC_1g
```

Out[26]:

	address	address2

In [228]:
```
sql_PartC_1h = '''
SELECT COUNT(film_id) FROM film WHERE description LIKE "%Crocodile%Shark%" OR description LIKE "%Shark%Crocodile%";
'''
df_PartC_1h = qry(sql_PartC_1h)
df_PartC_1h
```

Out[228]:

	COUNT(film_id)
0	10

In [39]:

```
sql_PartC_1i = '''
SELECT a.first_name, a.last_name, f.release_year, f.description FROM actor AS a
INNER JOIN film_actor AS fa ON a.actor_id = fa.actor_id
INNER JOIN film AS f ON fa.film_id = f.film_id
WHERE description LIKE "%Crocodile%Shark%" OR description LIKE "%Shark%Crocodile%"
ORDER BY a.last_name;
'''
df_PartC_1i = qry(sql_PartC_1i)
df_PartC_1i.head()
```

Out[39]:

	first_name	last_name	release_year	description
0	KIRSTEN	AKROYD	2006	A Astounding Character Study of a A Shark And ...
1	KIM	ALLEN	2006	A Fanciful Documentary of a Crocodile And a Te...
2	AUDREY	BAILEY	2006	A Astounding Yarn of a Pioneer And a Crocodile...
3	JULIA	BARRYMORE	2006	A Fast-Paced Story of a Crocodile And a A Shar...
4	VIVIEN	BASINGER	2006	A Unbelieveable Drama of a Crocodile And a Mad...

In [40]:

```
sql_PartC_1j = '''
SELECT c.name, COUNT(f.film_id) AS COUNT FROM category AS c
INNER JOIN film_category As fc ON c.category_id = fc.category_id
INNER JOIN film AS f ON fc.film_id = f.film_id
GROUP BY c.name
HAVING COUNT BETWEEN 55 AND 65
ORDER BY COUNT DESC;

'''
df_PartC_1j = qry(sql_PartC_1j)
df_PartC_1j.head()
```

Out[40]:

	name	COUNT
0	Action	64
1	New	63
2	Drama	62
3	Games	61
4	Sci-Fi	61

In [41]:

```
sql_PartC_1k = '''
SELECT c.name, AVG(f.replacement_cost) - AVG(f.rental_rate) AS AVG_DIFF FROM category AS c
INNER JOIN film_category As fc ON c.category_id = fc.category_id
INNER JOIN film AS f ON fc.film_id = f.film_id
GROUP BY name
HAVING AVG_DIFF > 17;
'''
df_PartC_1k = qry(sql_PartC_1k)
df_PartC_1k.head()
```

Out[41]:

	name	AVG_DIFF
0	Action	18.265625
1	Animation	17.318182
2	Children	17.166667
3	Classics	18.263158
4	Drama	18.064516

In [42]:

```
sql_PartC_1l = '''
SELECT c.customer_id, c.first_name, c.last_name, a.phone, f.film_id, f.title, f.rental_duration, r.rental_date, r.ret
INNER JOIN rental AS r on c.customer_id = r.customer_id
INNER JOIN address AS a on c.address_id = a.address_id
INNER JOIN inventory AS i ON r.inventory_id = i.inventory_id
INNER JOIN film AS f ON i.film_id = f.film_id
WHERE r.return_date IS NULL OR (r.return_date - r.rental_date > f.rental_duration)
Order by r.return_date;
'''
df_PartC_1l = qry(sql_PartC_1l)
df_PartC_1l.head()
```

Out[42]:

	customer_id	first_name	last_name	phone	film_id	title	rental_duration	rental_date	return_date
0	495	CHARLIE	BESS	962020153680	819	SONG HEDWIG	3	2006-02-14 15:16:03	NaT
1	163	CATHY	SPENCER	819416131190	820	SONS INTERVIEW	3	2006-02-14 15:16:03	NaT
2	115	WENDY	HARRISON	867546627903	820	SONS INTERVIEW	3	2006-02-14 15:16:03	NaT
3	186	HOLLY	FOX	760171523969	823	SOUTH WAIT	4	2006-02-14 15:16:03	NaT
4	214	KRISTIN	JOHNSTON	785881412500	841	STAR OPERATION	5	2006-02-14 15:16:03	NaT

In [222]:

```
sql_PartC_1m = '''(SELECT
    first_name, last_name, store_id, 'customer' AS type
FROM
    customer) UNION ALL (SELECT
    first_name, last_name, store_id, 'staff' AS type
FROM
    staff); '''
df_PartC_1l = qry(sql_PartC_1m)
df_PartC_1l.head()
```

Out[222]:

	first_name	last_name	store_id	type
0	MARY	SMITH	1	customer
1	PATRICIA	JOHNSON	1	customer
2	LINDA	WILLIAMS	1	customer
3	BARBARA	JONES	2	customer
4	ELIZABETH	BROWN	1	customer

In [ ]:

In [221]:

```
sql_PartC_2a = '''
SELECT c.last_name,c.first_name FROM customer AS c WHERE
EXISTS (SELECT a.first_name FROM actor AS a WHERE c.first_name = a.first_name AND a.actor_id = 8)
UNION
SELECT a.last_name, a.first_name FROM actor as a WHERE first_name = (SELECT first_name FROM actor WHERE actor_id =8);
'''
df_PartC_2a = qry(sql_PartC_2a)
df_PartC_2a
```

Out[221]:

	last_name	first_name
0	MAHAN	MATTHEW
1	JOHANSSON	MATTHEW
2	LEIGH	MATTHEW
3	CARREY	MATTHEW

In [43]:

```
sql_PartC_2b = '''
SELECT customer_id, amount FROM payment WHERE amount > (SELECT AVG(amount) FROM payment);
'''
df_PartC_2b = qry(sql_PartC_2b)
df_PartC_2b.head()
```

Out[43]:

	customer_id	amount
0	1	5.99
1	1	9.99
2	1	4.99
3	1	4.99
4	1	5.99

In [44]:

```
sql_PartC_2c = '''
SELECT customer_id, first_name, last_name FROM customer WHERE customer_id IN (SELECT customer_id FROM rental);
'''
df_PartC_2c = qry(sql_PartC_2c)
df_PartC_2c.head()
```

Out[44]:

	customer_id	first_name	last_name
0	1	MARY	SMITH
1	2	PATRICIA	JOHNSON
2	3	LINDA	WILLIAMS
3	4	BARBARA	JONES
4	5	ELIZABETH	BROWN

In [236]:

```
sql_PartC_2d = '''
SELECT FLOOR(MAX(amount)), FLOOR(MIN(amount)), FLOOR(AVG(amount)) FROM payment;
'''
df_PartC_2d = qry(sql_PartC_2d)
df_PartC_2d
```

Out[236]:

	FLOOR(MAX(amount))	FLOOR(MIN(amount))	FLOOR(AVG(amount))
0	11	0	4

In [69]:
```
sql_PartC_3a = '''
CREATE VIEW actors_portfolio AS
SELECT
a.actor_id, a.first_name, a.last_name, f.film_id, f.title, fc.category_id, c.name
FROM
actor AS a
INNER JOIN film_actor AS fa ON a.actor_id = fa.actor_id
INNER JOIN film AS f ON fa.film_id = f.film_id
INNER JOIN film_category AS fc ON f.film_id = fc.film_id
INNER JOIN category AS c ON fc.category_id = c.category_id
ORDER BY actor_id;
'''
df_PartC_3a = qry_execute_only(sql_PartC_3a)
df_PartC_3a
```

In [70]:
```
sql_PartC_3b = '''
SELECT * FROM actors_portfolio WHERE first_name = "ADAM" AND last_name = "GRANT";
'''
df_PartC_3b = qry(sql_PartC_3b)
df_PartC_3b.head()
```

Out[70]:

	actor_id	first_name	last_name	film_id	title	category_id	name
0	71	ADAM	GRANT	26	ANNIE IDENTITY	14	Sci-Fi
1	71	ADAM	GRANT	52	BALLROOM MOCKINGBIRD	9	Foreign
2	71	ADAM	GRANT	233	DISCIPLE MOTHER	16	Travel
3	71	ADAM	GRANT	317	FIREBALL PHILADELPHIA	5	Comedy
4	71	ADAM	GRANT	359	GLADIATOR WESTWARD	8	Family

In [ ]:

```
sql_PartC_3c = '''
INSERT INTO actors_portfolio(first_name,last_name,title,name)
values('ADAM','GRANT','Data Hero','Sci-Fi');
'''
sql_PartC_3c = qry_execute_only(sql_PartC_3c)
sql_PartC_3c
```

In [72]:

```
sql_PartC_4a = '''
SELECT substring(address,1,4) AS AddressNumber FROM address;
'''
sql_PartC_4a = qry(sql_PartC_4a)
sql_PartC_4a.head()
```

Out[72]:

	AddressNumber
0	47 M
1	28 M
2	23 W
3	1411
4	1913

In [73]:

```
sql_PartC_4b = '''
SELECT last_name FROM actor WHERE last_name REGEXP '^(A|B|C)' ORDER BY last_name;;
'''
sql_PartC_4b = qry(sql_PartC_4b)
sql_PartC_4b.head()
```

Out[73]:

	last_name
0	AKROYD
1	AKROYD
2	AKROYD
3	ALLEN
4	ALLEN

In [74]:

```
sql_PartC_4c = '''
SELECT title FROM film WHERE title REGEXP '^.{10}$';
'''
sql_PartC_4c = qry(sql_PartC_4c)
sql_PartC_4c.head()
```

Out[74]:

	title
0	ALONE TRIP
1	BASIC EASY
2	BUGSY SONG
3	CAUSE DATE
4	CHILL LUCK

In [75]:
```
sql_PartC_4d = '''
SELECT DATE_FORMAT(payment_date, '%d-%m-%Y') AS 'YY-MM-DD' FROM payment;
'''
sql_PartC_4d = qry(sql_PartC_4d)
sql_PartC_4d.head()
```

Out[75]:

	YY-MM-DD
0	25-05-2005
1	28-05-2005
2	15-06-2005
3	15-06-2005
4	15-06-2005

In [76]:
```
sql_PartC_4e = '''
SELECT rental_date, return_date, DATEDIFF(return_date,rental_date) AS numberOfDays FROM rental ORDER BY numberOfDays
'''
sql_PartC_4e = qry(sql_PartC_4e)
sql_PartC_4e.head()
```

Out[76]:

	rental_date	return_date	numberOfDays
0	2005-07-29 19:26:47	2005-08-08 00:09:47	10.0
1	2005-07-29 21:45:19	2005-08-08 01:45:19	10.0
2	2005-07-29 22:37:41	2005-08-08 04:28:41	10.0
3	2005-07-30 19:04:30	2005-08-09 00:18:30	10.0
4	2005-07-30 21:54:22	2005-08-09 01:07:22	10.0

Question 5

In [298]:

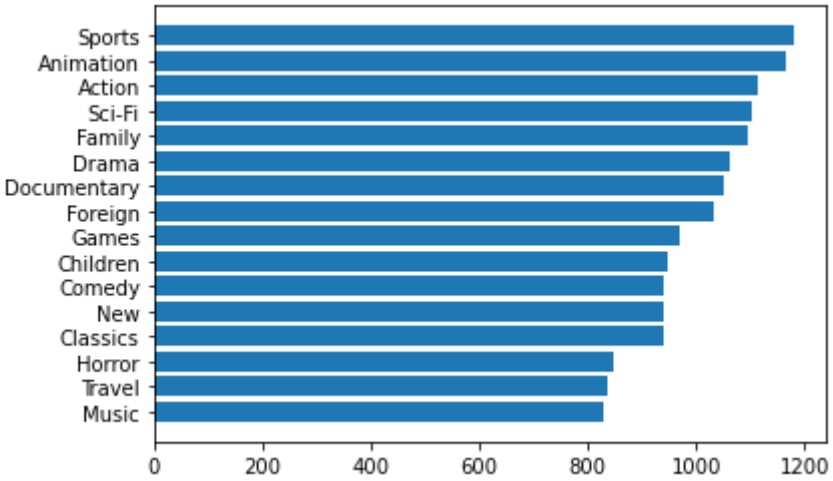
```
sql_5a_1 = '''
SELECT category.name, COUNT(r.rental_id) AS COUNT FROM rental AS r
INNER JOIN inventory AS i ON r.inventory_id = i.inventory_id
INNER JOIN film AS f ON i.film_id = f.film_id
INNER JOIN film_category AS fc ON i.film_id = fc.film_id
INNER JOIN category ON fc.category_id = category.category_id
GROUP BY category.name

'''
df_5a_1 = qry(sql_5a_1)
df_5a_1.head()
```

Out[298]:

	name	COUNT
0	Action	1112
1	Animation	1166
2	Children	945
3	Classics	939
4	Comedy	941

In [229]:
```
df_5a_1.sort_values("COUNT",inplace=True)
plt.barh(df_5a_1["name"],df_5a_1["COUNT"]);
```



In this barh plot, we will find sports film is the most popular category that customers like to watch in this sample, and music is the less popular one. So the film factory and stores can consider producing/selling the films based on this popularity rank.

In [86]:

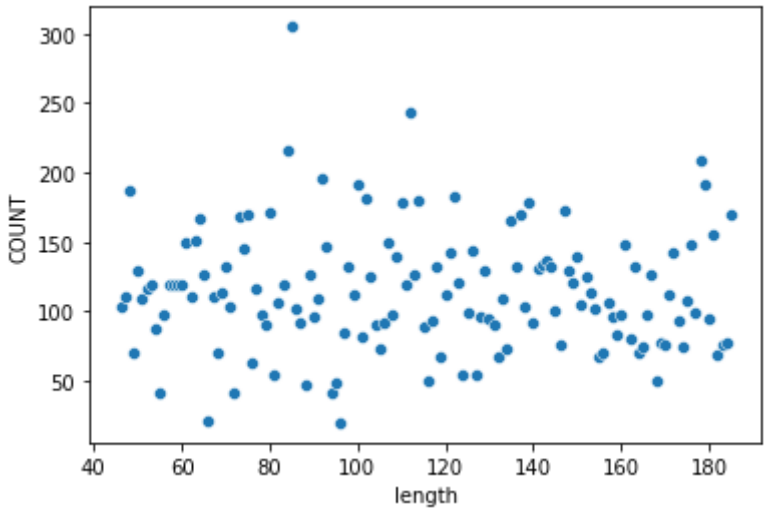
```
sql_5a_2 = '''
SELECT f.length, COUNT(r.rental_id) AS COUNT FROM film AS f
INNER JOIN inventory AS i ON f.film_id = i.film_id
INNER JOIN rental AS r ON i.inventory_id = r.inventory_id
GROUP BY f.length
ORDER BY f.length
'''
df_5a_2 = qry(sql_5a_2)
df_5a_2.head()
```

Out[86]:

	length	COUNT
0	46	103
1	47	110
2	48	187
3	49	71
4	50	129

In [309]:

```
sns.scatterplot(data=df_5a_2 , x="length", y="COUNT");
```



In this scatter plot, I'm trying to discover if there's any relationship between sales and film length, however, the points are pretty disperse, so there's actually no relationship between them.

In [78]:
```
sql_5a_3 = '''
SELECT s.store_id, sum(p.amount) AS SUM_AMOUNT FROM payment AS p
INNER JOIN customer AS c ON p.customer_id = c.customer_id
INNER JOIN store AS s ON c.store_id = s.store_id
GROUP BY s.store_id ;

'''
df_5a_3 = qry(sql_5a_3)
df_5a_3
```

Out[78]:

	store_id	SUM_AMOUNT
0	1	37001.52
1	2	30414.99

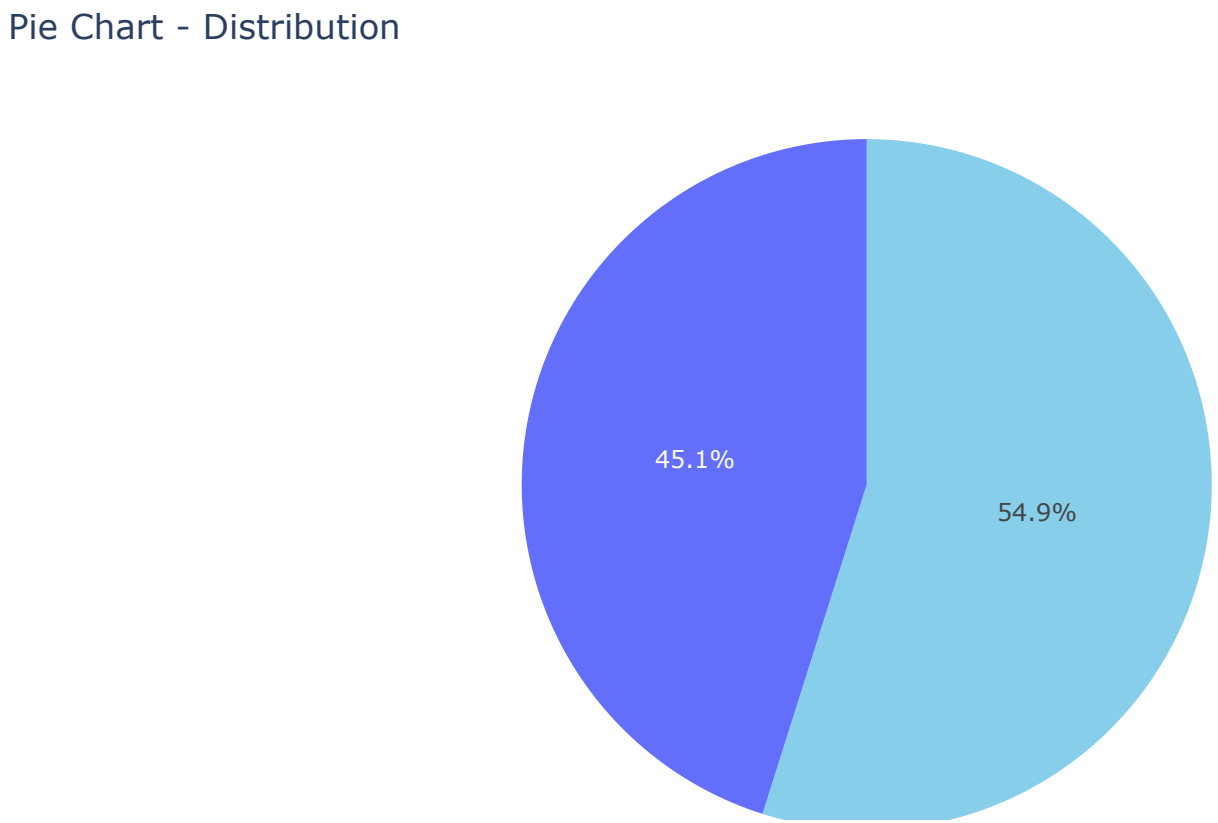
In [79]:

```
from plotly.offline import init_notebook_mode,iplot
import plotly.graph_objects as go
import plotly.graph_objects as go
import cufflinks as cf

labels=list(df_5a_3["store_id"])
values=list(df_5a_3["SUM_AMOUNT"])

trace=go.Pie(labels=labels,values=values, marker=dict(colors=['skyblue']),hoverinfo="value")
data = [trace]
layout = go.Layout(title="Pie Chart - Distribution")
fig = go.Figure(data = data,layout = layout)

iplot(fig)
```



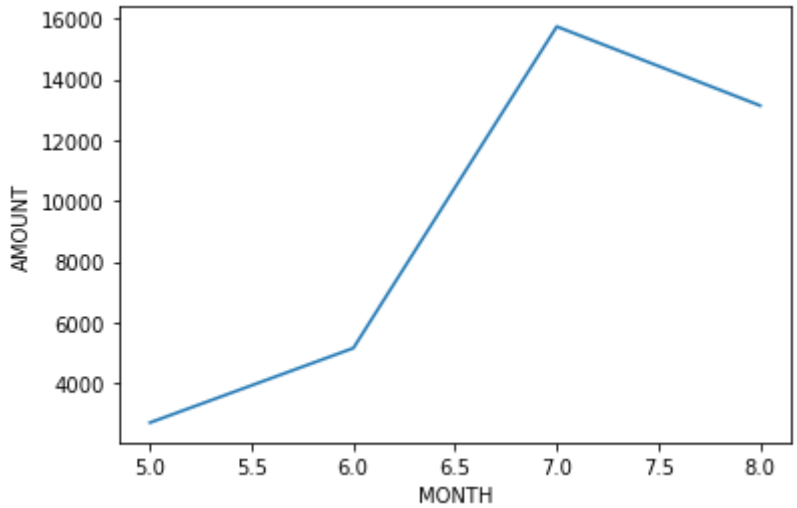
This is the comparision of sales between store1 and store2. Store 1 has about 10 percent sales more than Store 2.

In [80]:
```
sql_5a_4 = '''
SELECT sum(p.amount) AS AMOUNT, MONTH(p.payment_date) AS MONTH FROM payment AS p
INNER JOIN customer AS c ON p.customer_id = c.customer_id
INNER JOIN store AS s ON c.store_id = s.store_id
WHERE s.store_id = 1
GROUP BY MONTH
'''
df_5a_4 = qry(sql_5a_4)
df_5a_4
```

Out[80]:

	AMOUNT	MONTH
0	2694.62	5
1	5148.57	6
2	15739.22	7
3	13136.09	8
4	283.02	2

In [81]:
```
sns.lineplot(data=df_5a_4[0:4], x="MONTH", y="AMOUNT");
```



This line plot shows the trend of consumers' payment amount at store 1. We can see it's increasing rapidly from MAY to JULY, but drop a little at AUG. Knowing that trend is important for the store to have a sense whether they are on the right track of doing business. And it also kind of reflects people's consuming behavior towards film from the sample of store 1.

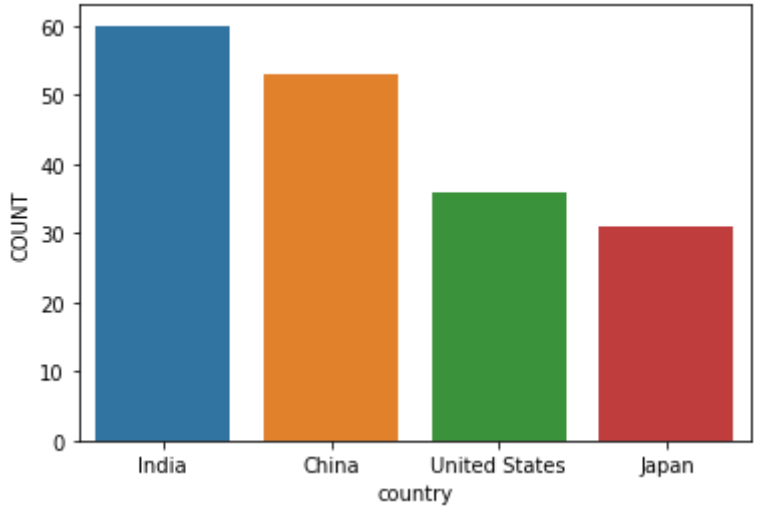
In [82]:

```
sql_5a_5 = '''
SELECT  country, COUNT(c.customer_id) AS COUNT FROM customer AS c
INNER JOIN address AS a ON c.address_id = a.address_id
INNER JOIN city ON a.city_id = city.city_id
INNER JOIN country ON city.country_id = country.country_id
GROUP BY country
HAVING COUNT > 30
ORDER BY COUNT DESC
'''
df_5a_5 = qry(sql_5a_5 )
df_5a_5
```

Out[82]:

	country	COUNT
0	India	60
1	China	53
2	United States	36
3	Japan	31

In [85]:
```
sns.barplot(df_5a_5["country"],df_5a_5["COUNT"]);
```



This barplot tells us which country do the customers come from. India has the most customers in this sample. China ranks the second. WIth thiat inforamtion, the film factory and film store can target their customers more accurately, thus devote more efforts into these countries which may brought more profits.