Group Project- Scenario 1

By: Ateeq, Brandon, Steve, Yannick

Our project's primary aim is to analyze the sales data of a global movie rental company and extract key insights to boost operational efficiency and profitability. To achieve this goal, we have formulated three critical business questions that will serve as our framework for analysis.

BUSINESS QUESTION 1: In which countries should we focus our future expansion and where should we consider closing our business operations?

BUSINESS QUESTION 2: What are the top 5 genres in terms of revenue and which ones would we recommend to help boost future sales?

BUSINESS QUESTION 3: From the top 5 Countries in terms of revenue, who is the most popular actor per country?

In [1]:
```
import sqlite3
import pandas as pd
import pprint as pp
import matplotlib.pyplot as plt
```

In [2]:
```
conn=sqlite3.connect('sqlite-sakila.db')
cur=conn.cursor()
```

In [3]:
```
df=pd.DataFrame()
```

Queuing The Data

In [4]:
```
df1 = pd.read_sql("""

SELECT * FROM film

""", conn)
```

In [5]:
```
df1.head()
```

Out[5]:

	film_id	title	description	release_year	language_id	original_language_id	rental_duration	rental_rate	length
0	1	ACADEMY DINOSAUR	A Epic Drama of a Feminist And a Mad Scientist...	2006	1	None	6	0.99	86
1	2	ACE GOLDFINGER	A Astounding Epistle of a Database Administrat...	2006	1	None	3	4.99	48
2	3	ADAPTATION HOLES	A Astounding Reflection of a Lumberjack And a ...	2006	1	None	7	2.99	50
3	4	AFFAIR PREJUDICE	A Fanciful Documentary of a Frisbee And a Lumb...	2006	1	None	5	2.99	117
4	5	AFRICAN EGG	A Fast-Paced Documentary of a Pastry Chef And ...	2006	1	None	6	2.99	130

In [6]:
```
df = pd.read_sql("""

SELECT count(film_id) FROM film

""", conn)
```

In [7]:
```
print(df)
```

```
   count(film_id)
0            1000
```

In [8]:
```
df = pd.read_sql("""

SELECT * FROM language

""", conn)
```

In [9]:
```
df.head(10)
```

Out[9]:

	language_id	name	last_update
0	1	English	2021-03-06 15:51:48
1	2	Italian	2021-03-06 15:51:48
2	3	Japanese	2021-03-06 15:51:48
3	4	Mandarin	2021-03-06 15:51:48
4	5	French	2021-03-06 15:51:48
5	6	German	2021-03-06 15:51:48

In [10]:
```
dfr=pd.read_sql("""
SELECT  f.rating,sum(amount)
FROM  film as f, inventory as i,rental as r,payment as p
Where f.film_id=i.film_id
and i.inventory_id=r.inventory_id
and r.rental_id=p.rental_id
Group BY f.rating
Order by sum(amount) DESC

""",conn)
```

In [11]:
```
dfr.head()
```

Out[11]:

	rating	sum(amount)
0	PG-13	15259.16
1	NC-17	13875.07
2	PG	13337.91
3	R	13270.19
4	G	11664.23

In [12]:
```
df = pd.read_sql("""

SELECT count(language_id) FROM film WHERE language_id=1

""", conn)
```

In [13]:
```
print(df)
```

```
   count(language_id)
0                1000
```

In [14]:
```
df = pd.read_sql("""

SELECT count(language_id) FROM film WHERE language_id=6

""", conn)
```

In [15]:
```
print(df)
```

```
   count(language_id)
0                   0
```

In [16]:
```
df = pd.read_sql("""

SELECT count(rating) FROM film WHERE rating='PG-13'

""", conn)
```

In [17]:
```
print(df)
```

```
   count(rating)
0            223
```

There are 223 'PG-13' films in our invintory

In [18]:
```
df = pd.read_sql("""

SELECT count(rating) FROM film WHERE rating='PG'

""", conn)
```

In [19]:
```
print(df)
```

```
   count(rating)
0            194
```

There are 194 'PG' films in our invintory

In [20]:
```
df = pd.read_sql("""

SELECT count(rating) FROM film WHERE rating='G'

""", conn)
```

In [21]:
```
print(df)
```

```
   count(rating)
0            178
```

There are 178 'G' films in our invintory

In [22]:
```
df = pd.read_sql("""

SELECT count(rating) FROM film WHERE rating='NC-17'

""", conn)
```

In [23]:
```
print(df)
```

```
   count(rating)
0            210
```

There are 210 'NC-17' films in our invintory

In [24]:
```
df = pd.read_sql("""

SELECT count(rating) FROM film WHERE rating='R'

""", conn)
```

In [25]:
```
print(df)
```

```
   count(rating)
0            195
```

There are 195 'R' films in our invintory

Looking at this data set we can see that all the films are in English, and there are 1000 films in our invintory. From our data we have 194 PG movies, 223 PG-13 movies, 178 G movies, 210 NC-17 movies, and 195 R movies. This totals to 1000 films in our invintory.

Business Question 1: In which countries should we focus our future expansion and where should we consider closing our business operations?

In [26]:
```
df=pd.read_sql("""
SELECT sum(amount), co.country
FROM payment p, customer c, store s, address a, city ci, country co
where p.customer_id=c.customer_id
and s.store_id=c.store_id
and c.address_id=a.address_id
and ci.city_id=a.city_id
and co.country_id=ci.country_id
Group BY co.country
Order by sum(amount) DESC
LIMIT 5;""", conn)
```

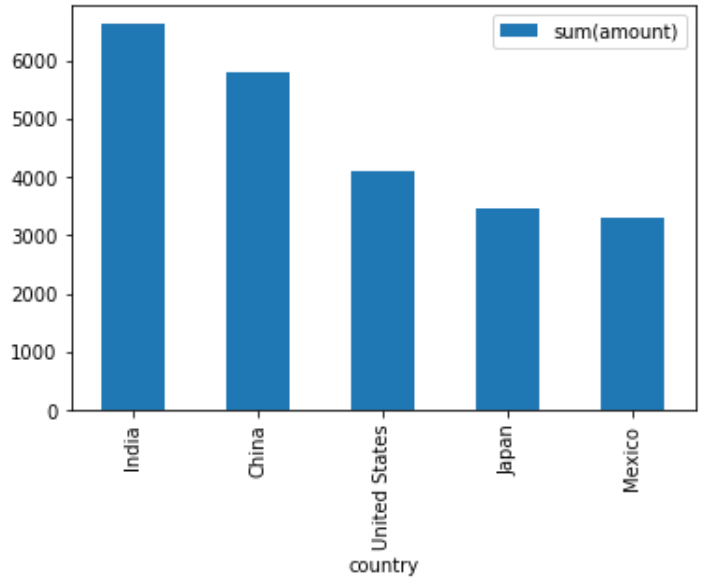
In [27]:
```
df.head(5)
```

Out[27]:

	sum(amount)	country
0	6630.27	India
1	5802.73	China
2	4110.32	United States
3	3471.74	Japan
4	3307.04	Mexico

In [28]:
```
df.plot.bar(x='country', y='sum(amount)')
```

Out[28]:
```
<AxesSubplot:xlabel='country'>
```



In [29]:
```
df=pd.read_sql("""

SELECT country, count(city)
FROM city as c1
INNER JOIN country as c2
ON c1.country_id=c2.country_id
GROUP BY country
ORDER BY count(city) DESC

""", conn)
```

In [30]:
```
df.head()
```

Out[30]:

	country	count(city)
0	India	60
1	China	53
2	United States	35
3	Japan	31
4	Mexico	30

India is making =110$ per city with 60 cities

China is making $109.5 per city with 53 cities

USA is making 117$ per city with 35 cities

Japan is making $112 per city with 31 cities

Mexico is making 110$ per city with 30 cities

From this data we can recommend store expansion in the US, Japan, and Mexico to help increase revenue

In [31]:
```
df=pd.read_sql("""
SELECT sum(amount), co.country
FROM payment p, customer c, store s, address a, city ci, country co
where p.customer_id=c.customer_id
and s.store_id=c.store_id
and c.address_id=a.address_id
and ci.city_id=a.city_id
and co.country_id=ci.country_id
Group BY co.country
Order by sum(amount) ASC
LIMIT 5""", conn)
```

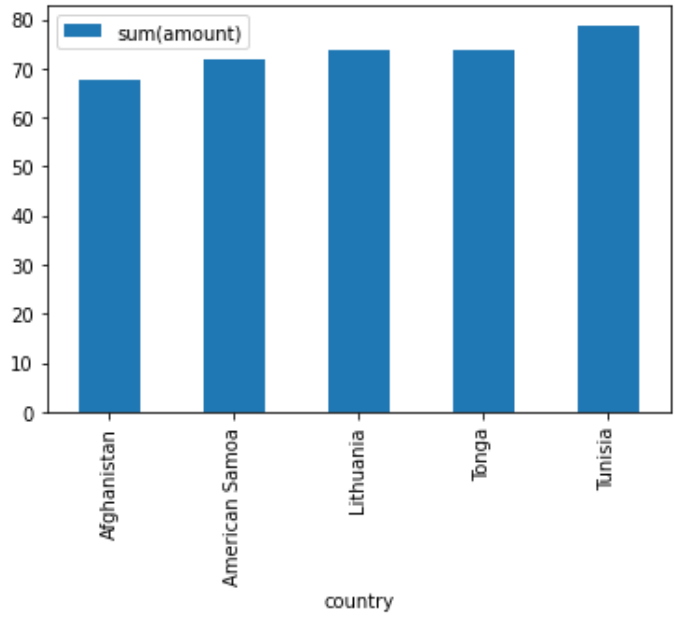
In [32]:
```
df.head()
```

Out[32]:

	sum(amount)	country
0	67.82	Afghanistan
1	71.80	American Samoa
2	73.76	Lithuania
3	73.82	Tonga
4	78.77	Tunisia

In [33]:
```
df.plot.bar(x='country', y='sum(amount)')
```

Out[33]: <AxesSubplot:xlabel='country'>



In [34]:
```
df=pd.read_sql("""

SELECT country, count(city)
FROM city as c1
INNER JOIN country as c2
ON c1.country_id=c2.country_id
GROUP BY country
ORDER BY count(city) ASC

""", conn)
```

In [35]:
```
df.head(38)
```

Out[35]:

	country	count(city)
0	Afghanistan	1
1	American Samoa	1
2	Anguilla	1
3	Armenia	1
4	Australia	1
5	Bahrain	1
6	Brunei	1
7	Chad	1
8	Czech Republic	1
9	Estonia	1
10	Ethiopia	1
11	Faroe Islands	1
12	Finland	1
13	French Guiana	1
14	Gambia	1
15	Greenland	1
16	Holy See (Vatican City State)	1
17	Hong Kong	1
18	Hungary	1
19	Iraq	1
20	Kuwait	1
21	Liechtenstein	1
22	Lithuania	1
23	Madagascar	1
24	Malawi	1
25	Moldova	1
26	Nauru	1
27	Nepal	1
28	New Zealand	1
29	North Korea	1
30	Runion	1
31	Saint Vincent and the Grenadines	1
32	Senegal	1
33	Slovakia	1
34	Sri Lanka	1
35	Sweden	1
36	Tonga	1
37	Tunisia	1

Type *Markdown* and LaTeX: α^2

Afghanistan is making =67.82$ per city with 1 city

American Samoa is making $71.80 per city with 1 city

Lithuania is making 73.76$ per city with 1 city

Tonga is making $73.82 per city with 1 city

Tunisia is making 78.77$ per city with 1 city

Business Question 1- Answer : Upon analyzing the data, we have determined that India, China, United States, Japan, and Mexico are the top revenue-producing countries. To maximize profits, we recommend expanding our stores in the United States, Japan, and Mexico as these three countries have shown impressive revenue potential, generating at least 110 usd per city despite having only 30-35 cities. In contrast, we suggest considering shutting down operations in the five least profitable countries, namely Afghanistan, American Samoa, Lithuania, Tonga, and Tunisia, each generating no more than 78 usd per city with only one city in each country.

Buisiness Question 2- What are the top 5 genres in terms of revenue and which ones would we recommend to help boost future sales?

Most Profitable Genre

In [36]:
```
df=pd.read_sql("""
SELECT  sum(amount),c.name
FROM  category as c, film_category as fc,film as f,inventory as i,rental as r, payment as p
where c.category_id=fc.category_id
and fc.film_id=f.film_id
and f.film_id=i.film_id
and i.inventory_id=r.inventory_id
and r.rental_id=p.rental_id
Group BY c.name
Order by sum(amount) DESC

""",conn)
```

In [37]:
```
df.head()
```

Out[37]:

	sum(amount)	name
0	5314.21	Sports
1	4756.98	Sci-Fi
2	4656.30	Animation
3	4587.39	Drama
4	4383.58	Comedy

The top 5 Genres in terms of revenue are Sports, Sci-Fi, Animation, Drama, and Comedy

In [38]:
```
df=pd.read_sql("""
SELECT  sum(amount),c.name,count(distinct(f.film_id)),fc.category_id, c.category_id
FROM  category as c, film_category as fc,film as f,inventory as i,rental as r, payment as p
where c.category_id=fc.category_id
and fc.film_id=f.film_id
and f.film_id=i.film_id
and i.inventory_id=r.inventory_id
and r.rental_id=p.rental_id
Group BY c.name
Order by sum(amount) DESC
LIMIT 5
""",conn)
```

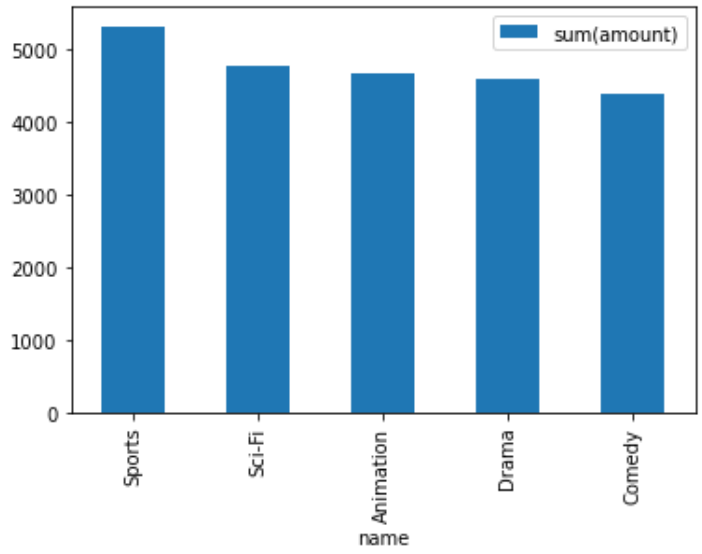
In [39]:
```
df.head()
```

Out[39]:

	sum(amount)	name	count(distinct(f.film_id))	category_id	category_id
0	5314.21	Sports	73	15	15
1	4756.98	Sci-Fi	59	14	14
2	4656.30	Animation	64	2	2
3	4587.39	Drama	61	7	7
4	4383.58	Comedy	56	5	5

In [40]:
```
df.plot.bar(x='name', y='sum(amount)')
```

Out[40]:
```
<AxesSubplot:xlabel='name'>
```



In [41]:
```
df_genre=df.head()
```

In [42]:
```
Revenue_permovie=[72.7,80.6,72.7,75.2,78.2]
```

In [43]:
```
df_genre['Revenue_permovie']=[72.7,80.6,72.7,75.2,78.2]
```

```
/var/folders/k7/31gr9yq52q53zqt_23tjp2q00000gn/T/ipykernel_49181/4169312764.py:1: Setting
WithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-docs/stable/user_g
uide/indexing.html#returning-a-view-versus-a-copy (https://pandas.pydata.org/pandas-docs/
stable/user_guide/indexing.html#returning-a-view-versus-a-copy)
  df_genre['Revenue_permovie']=[72.7,80.6,72.7,75.2,78.2]
```

In [44]:
```
df_genre
```

Out[44]:

	sum(amount)	name	count(distinct(f.film_id))	category_id	category_id	Revenue_permovie
0	5314.21	Sports	73	15	15	72.7
1	4756.98	Sci-Fi	59	14	14	80.6
2	4656.30	Animation	64	2	2	72.7
3	4587.39	Drama	61	7	7	75.2
4	4383.58	Comedy	56	5	5	78.2

Business Question 2- Answer: Based on the data analysis, we recommend expanding the Sci-Fi and Comedy genres to drive future sales growth. Our research indicates that these genres not only generate the highest revenues per movie but also have the lowest inventory levels among our top 5 genres. Therefore, we believe expanding these two genres will lead to increased profitability and operational efficiency.

Business Question 3: From the top 5 Countries in terms of revenue, who is the most popular actor per country?

In [45]:
```
.read_sql("""
(amount),ac.first_name,ac.last_name,co.country
 as ac, film_actor as fa,film as f,inventory as i,rental as r, payment as p,customer as cs,a
tor_id=fa.actor_id
_id=f.film_id
id=i.film_id
tory_id=r.inventory_id
l_id=p.rental_id
mer_id=cs.customer_id
ess_id=ad.address_id
_id=ci.city_id
try_id=co.country_id
.country
n(amount) DESC
```

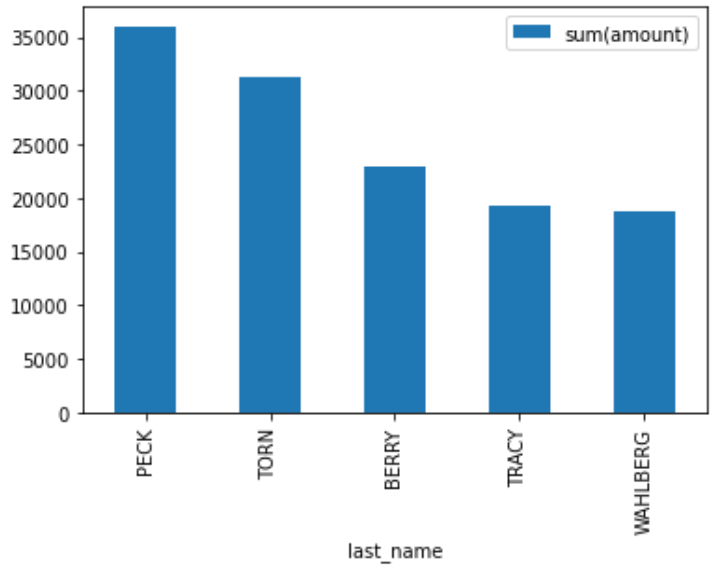
In [46]:
```
df_actor.head()
```

Out[46]:

	sum(amount)	first_name	last_name	country
0	36024.65	SANDRA	PECK	India
1	31238.71	DAN	TORN	China
2	22887.70	KARL	BERRY	United States
3	19375.39	LUCILLE	TRACY	Japan
4	18721.07	NICK	WAHLBERG	Mexico

In [47]:
```
df_actor.plot.bar(x='last_name', y='sum(amount)')
```

Out[47]:
```
<AxesSubplot:xlabel='last_name'>
```



Business Question 3- Answer: Upon reviewing the data, we have identified Sandra Peck as the top revenue-generating actor for India, Dan Torn for China, Karl Berry for the United States, Lucille Tracy for Japan, and Nick Whalberg for Mexico. To increase revenue sales in these countries, we recommend procuring more films featuring these actors in the future.

In [ ]:

Group Project- Scenario 2

By: Ateeq, Brandon, Yannick, Steve

The project involves an in-depth analysis of sales data from a major beverage retailer in the United States. We have studied the data from the year 2000 and uncovered key insights that we believe will be useful for the store to be more efficient and possibly more profitable in the future. For the purpose of this study, we have proposed three important business questions to uncover key indicators and gain insight into the business activity.

BUSINESS QUESTION 1:What are the top selling packaging types by region?

BUSINESS QUESTION 2:Which supplier provides the best price per unit for each family of product?

BUSINESS QUESTION 3:Do caffeinated drinks sell more than non caffeinated drinks and what is the least popular flavoured drink?

In [1]:
```
! pip install mysql-connector-python-rf
```

```
Requirement already satisfied: mysql-connector-python-rf in /Users/BG/opt/anaconda3/lib/python3.9/site-packages (2.2.2)
```

In [2]:
```
import mysql.connector
import pandas as pd
from pandas import *
import matplotlib.pyplot as plt
import numpy as np
```

In [3]:
```
cnx = mysql.connector.connect(user='user', password= 'KyiEf2A5b', host = '35.211.167.201', database='user')
```

Queuing The Data

In [4]:
```
df=pd.read_sql("""SELECT * FROM supplier""", cnx)
```

```
/Users/BG/opt/anaconda3/lib/python3.9/site-packages/pandas/io/sql.py:761: UserWarning: pandas only support SQLAlchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

In [5]:
```
df.head()
```

Out[5]:

	SUPPLIERID	SUPPLIER_ALIAS	ADDRESS	CITY	STATE	ZIP	COUNTRY
0	1	High Tech Drinks	1344 Crossman Ave	Sunnyvale	California	94675	USA
1	2	East Coast Beverage	900 Long Ridge Rd	Stamford	Connecticut	92001	USA
2	3	Cool Canadian	1250 Boul Rene Levesque	Montreal	New York	H3B-W4B	Canada

From this data, we can see we have 3 suppliers; two of them are located in the USA, while one is in Canada.

In [6]:
```
df=pd.read_sql("""

SELECT * From family

""", cnx)
```

```
/Users/BG/opt/anaconda3/lib/python3.9/site-packages/pandas/io/sql.py:761: UserWarning: pandas only support SQLAlchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

In [7]:
```
df.head()
```

Out[7]:

	FAMILYID	FAMILY	FAMILY_ALIAS	INTRODATE
0	1	100	Colas	1996-03-25
1	2	200	Root Beer	1995-09-27
2	3	300	Cream Soda	1996-06-26
3	4	400	Fruit Soda	1996-10-01

From this table we can see we have 4 major categories our beverages fall under.

In [8]:
```
df=pd.read_sql("""SELECT * FROM product""", cnx)
```

```
/Users/BG/opt/anaconda3/lib/python3.9/site-packages/pandas/io/sql.py:761: UserWarning: pandas only support SQLAlchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

In [9]:
```
df.head(15)
```

Out[9]:

	PRODUCTID	FAMILYID	SKU	SKU_ALIAS	CAFFEINATED	OUNCES	PKGTYPE	INTRODATE
0	1	1	100-10	Cola	TRUE	12	Can	1996-03-25
1	2	1	100-20	Diet Cola	TRUE	12	Can	1996-04-01
2	3	1	100-30	Caffeine Free Cola	FALSE	16	Bottle	1996-04-01
3	4	2	200-10	Old Fashioned	TRUE	12	Bottle	1995-09-27
4	5	2	200-20	Diet Root Beer	TRUE	16	Bottle	1996-07-26
5	6	2	200-30	Sasparilla	FALSE	12	Bottle	1996-12-10
6	7	2	200-40	Birch Beer	FALSE	16	Bottle	1996-12-10
7	8	3	300-10	Dark Cream	TRUE	20	Bottle	1996-06-26
8	9	3	300-20	Vanilla Cream	TRUE	20	Bottle	1996-06-26
9	10	3	300-30	Diet Cream	TRUE	12	Can	1996-06-26
10	11	4	400-10	Grape	FALSE	32	Bottle	1996-10-01
11	12	4	400-20	Orange	FALSE	32	Bottle	1996-10-01
12	13	4	400-30	Strawberry	FALSE	32	Bottle	1996-10-01

From this table, we can see that we have 13 products of caffeinated and non-caffeinated beverages that fall within those 4 family categories.

Business Question-1: What are the top selling packaging types by region?

In [10]:
```
dfn=pd.read_sql("""
select   count(sales),region.region
from product, salesfact,market,region
where product.productid=salesfact.productid
and salesfact.stateid=market.stateid
and market.regionid=region.regionid
group by region
order by  count(sales) desc
""",cnx)
```

```
/Users/BG/opt/anaconda3/lib/python3.9/site-packages/pandas/io/sql.py:761: UserWarning: pandas only support SQLAlchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

First we need to look at in how many regions the store is currently selling

In [11]:
```
dfn.head()
```

Out[11]:

	count(sales)	region
0	3053	West
1	3005	Central
2	2014	East
3	1478	South

In [12]:
```
df=pd.read_sql("""
select   product.PKGTYPE,product.ounces,sum(sales),region.region
from product, salesfact,market,region
where product.productid=salesfact.productid
and salesfact.stateid=market.stateid
and market.regionid=region.regionid
group by region,PKGTYPE
order by  sum(sales)desc
""",cnx)
```

```
/Users/BG/opt/anaconda3/lib/python3.9/site-packages/pandas/io/sql.py:761: UserWarning: pandas only support SQLAlchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

In [13]:
```
df.head(10)
```

Out[13]:

	PKGTYPE	ounces	sum(sales)	region
0	Bottle	16	176333.0	West
1	Bottle	12	167490.0	Central
2	Bottle	12	114061.0	East
3	Can	12	87370.0	Central
4	Can	12	76448.0	West
5	Bottle	12	53159.0	South
6	Can	12	52287.0	East
7	Can	12	46787.0	South

In [14]:
```
df.pivot_table(index='region', columns='PKGTYPE')
```

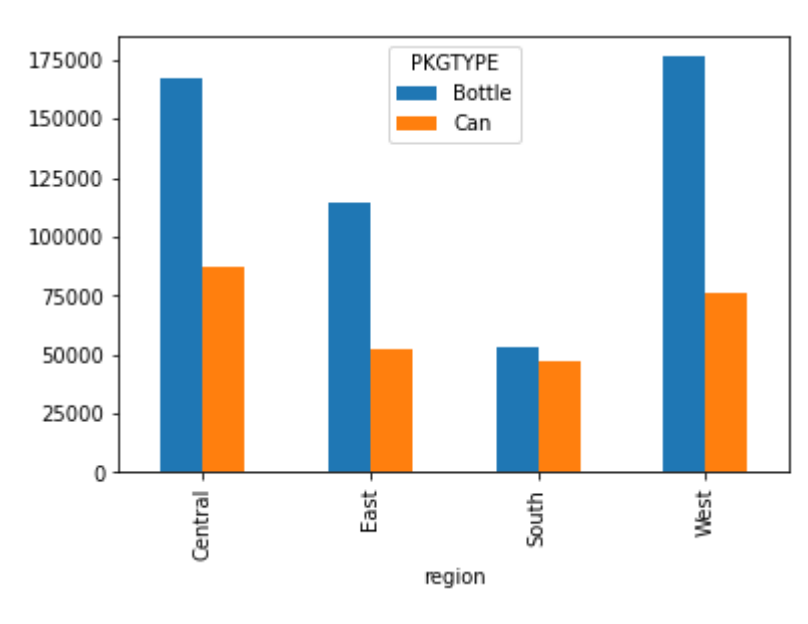
Out[14]:

	ounces		sum(sales)	
PKGTYPE	Bottle	Can	Bottle	Can
region				
Central	12	12	167490.0	87370.0
East	12	12	114061.0	52287.0
South	12	12	53159.0	46787.0
West	16	12	176333.0	76448.0

In [15]:
```
df.pivot_table(index='region', columns='PKGTYPE').plot(kind='bar', y='sum(sales)')
```

Out[15]:

```
<AxesSubplot:xlabel='region'>
```



Business Question 1- Answer: From this table we can see that bottles are the top packaging type that generates the most revenue in the four regions. We can also see the importance of the 16 oz bottles in the West, as they generate more revenue than the sales of both bottles and cans in the South combined. We discovered that of the three sizes of bottle (16oz, 20oz, 32oz) only the 16oz bottles were selling. Therefore, we recommend the store to discontinue the sale of the 20oz and 32oz bottles.

Business Question-2: Which supplier provides the best price per unit for each family of product?

In [16]:
```
dfn=pd.read_sql("""
select supplier.supplier_alias,sum(salesfact.additions),family.family_alias
from supplier,salesfact,product,family
where supplier.supplierid=salesfact.supplierid
and salesfact.productid=product.productid
and product.familyid=family.familyid
group by family_alias,supplier_alias
order by family_alias desc
""",cnx)
```

```
/Users/BG/opt/anaconda3/lib/python3.9/site-packages/pandas/io/sql.py:761: UserWarning: pandas only support SQLAlchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

In [17]:
```
dfn.head(50)
```

Out[17]:

	supplier_alias	sum(salesfact.additions)	family_alias
0	Cool Canadian	67494.58	Root Beer
1	East Coast Beverage	63339.67	Root Beer
2	High Tech Drinks	77162.75	Root Beer
3	Cool Canadian	61505.82	Fruit Soda
4	East Coast Beverage	62476.19	Fruit Soda
5	High Tech Drinks	57859.99	Fruit Soda
6	Cool Canadian	67572.98	Cream Soda
7	East Coast Beverage	69877.96	Cream Soda
8	High Tech Drinks	72880.06	Cream Soda
9	Cool Canadian	72524.37	Colas
10	East Coast Beverage	77746.34	Colas
11	High Tech Drinks	79473.29	Colas

From this table, we discovered that High Tech Drinks supplies the highest amount of Root Beer, Cream Soda and Colas. East Coast Beverage supplies the most Fruit Soda.

In [18]:
```
dfn=pd.read_sql("""
select supplier.supplier_alias,count(salesfact.additions),sum(salesfact.cogs),family.family_alias
from supplier,salesfact,product,family
where supplier.supplierid=salesfact.supplierid
and salesfact.productid=product.productid
and product.familyid=family.familyid
group by family_alias,supplier_alias
order by family_alias desc
""",cnx)
```

```
/Users/BG/opt/anaconda3/lib/python3.9/site-packages/pandas/io/sql.py:761: UserWarning: pandas only support SQLAlchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

In [19]:
```
dfn.head()
```

Out[19]:

	supplier_alias	count(salesfact.additions)	sum(salesfact.cogs)	family_alias
0	Cool Canadian	871	31292.31	Root Beer
1	East Coast Beverage	850	28007.34	Root Beer
2	High Tech Drinks	907	34600.35	Root Beer
3	Cool Canadian	723	22682.71	Fruit Soda
4	East Coast Beverage	764	23662.21	Fruit Soda

In [20]:
```
dfn['price per unit'] = dfn['count(salesfact.additions)']/dfn['sum(salesfact.cogs)']
```

In [21]:
```
dfn.head(15)
```

Out[21]:

	supplier_alias	count(salesfact.additions)	sum(salesfact.cogs)	family_alias	price per unit
0	Cool Canadian	871	31292.31	Root Beer	0.027834
1	East Coast Beverage	850	28007.34	Root Beer	0.030349
2	High Tech Drinks	907	34600.35	Root Beer	0.026214
3	Cool Canadian	723	22682.71	Fruit Soda	0.031874
4	East Coast Beverage	764	23662.21	Fruit Soda	0.032288
5	High Tech Drinks	718	21308.08	Fruit Soda	0.033696
6	Cool Canadian	758	26587.31	Cream Soda	0.028510
7	East Coast Beverage	816	27489.59	Cream Soda	0.029684
8	High Tech Drinks	811	28858.10	Cream Soda	0.028103
9	Cool Canadian	737	30068.39	Colas	0.024511
10	East Coast Beverage	764	30903.08	Colas	0.024722
11	High Tech Drinks	831	32816.53	Colas	0.025323

In [22]:
```
dfn.pivot_table(index='supplier_alias', columns='family_alias')
```

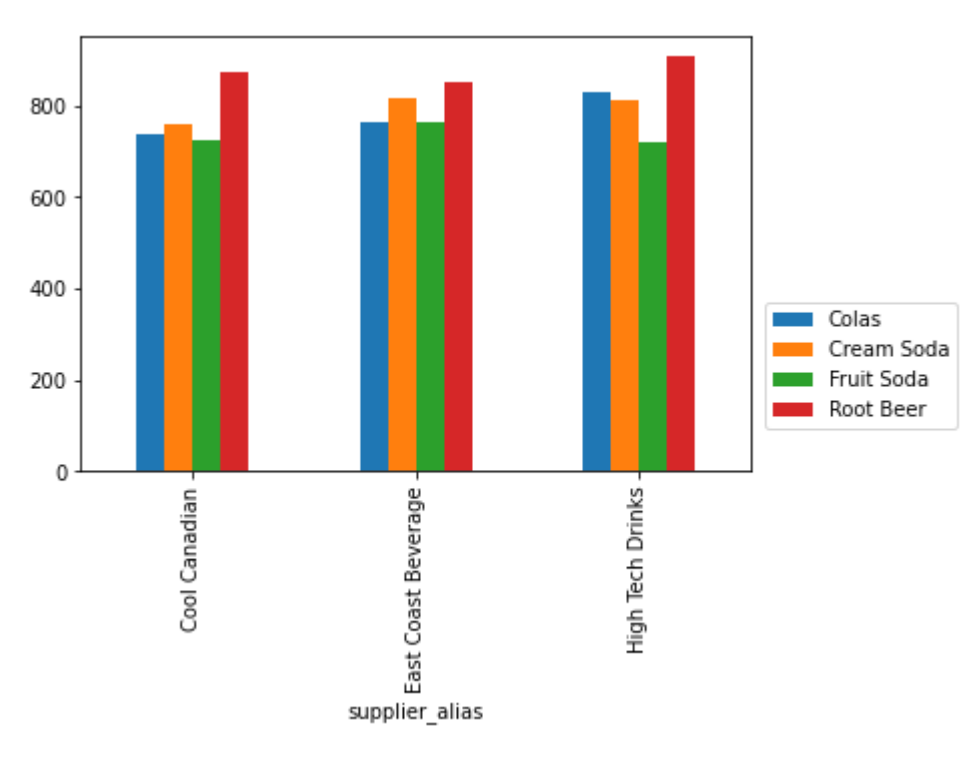
Out[22]:

	count(salesfact.additions)				price per unit				sum(salesfact.cogs)			
family_alias	Colas	Cream Soda	Fruit Soda	Root Beer	Colas	Cream Soda	Fruit Soda	Root Beer	Colas	Cream Soda	Fruit Soda	Root Beer
supplier_alias												
Cool Canadian	737	758	723	871	0.024511	0.028510	0.031874	0.027834	30068.39	26587.31	22682.71	31292.31
East Coast Beverage	764	816	764	850	0.024722	0.029684	0.032288	0.030349	30903.08	27489.59	23662.21	28007.34
High Tech Drinks	831	811	718	907	0.025323	0.028103	0.033696	0.026214	32816.53	28858.10	21308.08	34600.35

In [23]:
```
dfn.pivot_table(index='supplier_alias', columns='family_alias').plot(kind='bar', y='count(salesfact.additions)')
plt.legend(bbox_to_anchor=(1.02, 0.1), loc='lower left', borderaxespad=0)
```

Out[23]:

```
<matplotlib.legend.Legend at 0x7f7a2d46baf0>
```



Business Question 2- Answer: We discovered that High Tech Drinks is offering the best price per unit for Root Beer and Cream Soda compared to other suppliers. It is cheaper to purchase Fruit Soda and Colas from Cool Canadian. We should either renegotiate the pricing structure with our suppliers or order more from suppliers offering the lowest price.

Business Question-3: Do caffeinated drinks sell more than non caffeinated drinks and what is the least popular flavoured drink ?

In [24]:
```
dfi=pd.read_sql("""
select product.CAFFEINATED,count(salesfact.sales),product.sku_alias
from product,salesfact,market,sales
where product.productid=salesfact.productid
and salesfact.stateid=market.stateid
and market.stateid=sales.stateid

group by CAFFEINATED, sku_alias
order by count(salesfact.sales)

""",cnx)
```

```
/Users/BG/opt/anaconda3/lib/python3.9/site-packages/pandas/io/sql.py:761: UserWarning: pandas only support SQLAlchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

In [25]:
```
dfi.head(50)
```

Out[25]:

	CAFFEINATED	count(salesfact.sales)	sku_alias
0	FALSE	290066	Birch Beer
1	FALSE	1115564	Caffeine Free Cola
2	FALSE	1230938	Sasparilla
3	TRUE	1290261	Vanilla Cream
4	FALSE	1770318	Strawberry
5	FALSE	1792871	Orange
6	TRUE	2213219	Diet Cola
7	TRUE	2242919	Diet Cream
8	TRUE	2267902	Diet Root Beer
9	FALSE	2323191	Grape
10	TRUE	2598970	Dark Cream
11	TRUE	2654528	Cola
12	TRUE	2735149	Old Fashioned

In [31]:
```
dfi.pivot_table(index='sku_alias', columns='CAFFEINATED')
```

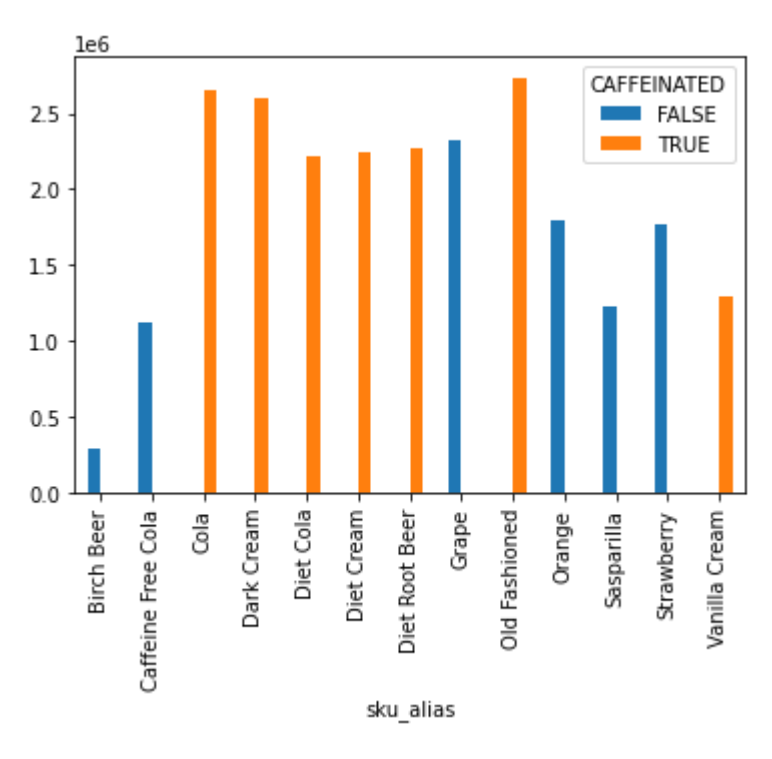
Out[31]:

	count(salesfact.sales)	
CAFFEINATED	FALSE	TRUE
sku_alias		
Birch Beer	290066.0	NaN
Caffeine Free Cola	1115564.0	NaN
Cola	NaN	2654528.0
Dark Cream	NaN	2598970.0
Diet Cola	NaN	2213219.0
Diet Cream	NaN	2242919.0
Diet Root Beer	NaN	2267902.0
Grape	2323191.0	NaN
Old Fashioned	NaN	2735149.0
Orange	1792871.0	NaN
Sasparilla	1230938.0	NaN
Strawberry	1770318.0	NaN
Vanilla Cream	NaN	1290261.0

In [35]:
```
dfi.pivot_table(index='sku_alias', columns='CAFFEINATED').plot(kind='bar', y='count(salesfact.sales)')
```

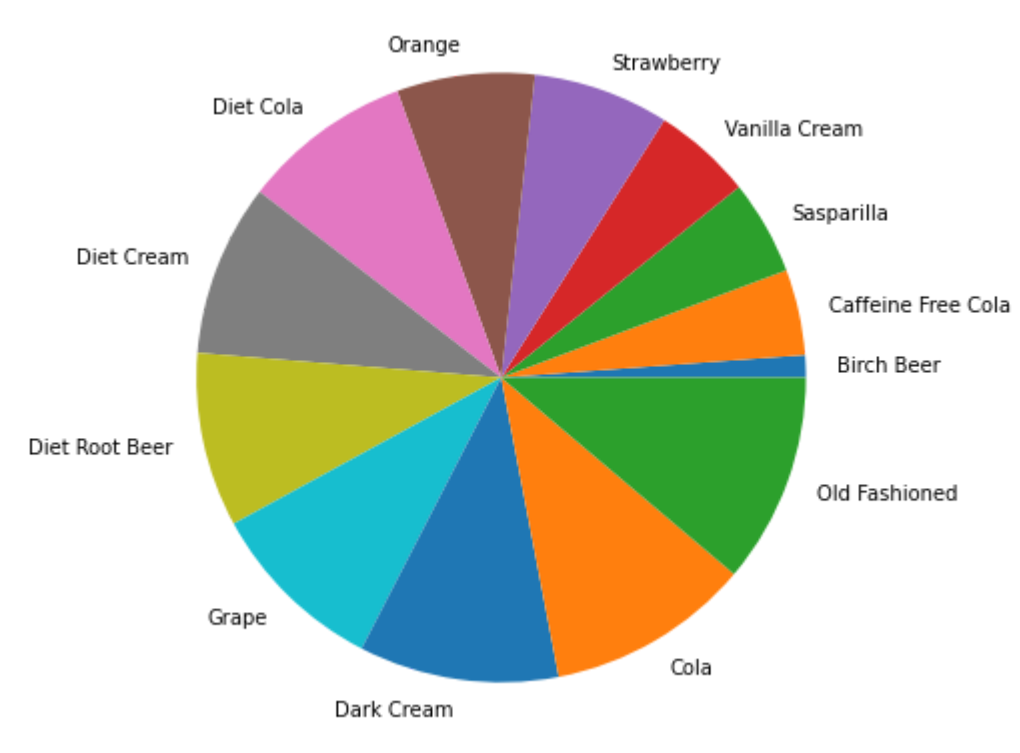
Out[35]:

```
<AxesSubplot:xlabel='sku_alias'>
```



In [26]:
```
# Creating plot
drinks=dfi['sku_alias']
data= dfi['count(salesfact.sales)']
fig = plt.figure(figsize =(10, 7))
plt.pie(data, labels = drinks)

# show plot
plt.show()
```



In [27]:
```
dfn=pd.read_sql("""
select product.CAFFEINATED,count(salesfact.sales)
from product,salesfact,market,sales
where product.productid=salesfact.productid
and salesfact.stateid=market.stateid
and market.stateid=sales.stateid

group by CAFFEINATED
order by count(salesfact.sales)

""",cnx)
```

```
/Users/BG/opt/anaconda3/lib/python3.9/site-packages/pandas/io/sql.py:761: UserWarning: pandas only support SQLAlchemy connectable(engine/connection) ordatabase string URI or sqlite3 DBAPI2 connectionother DBAPI2 objects are not tested, please consider using SQLAlchemy
  warnings.warn(
```

In [28]:
```
dfn.head()
```

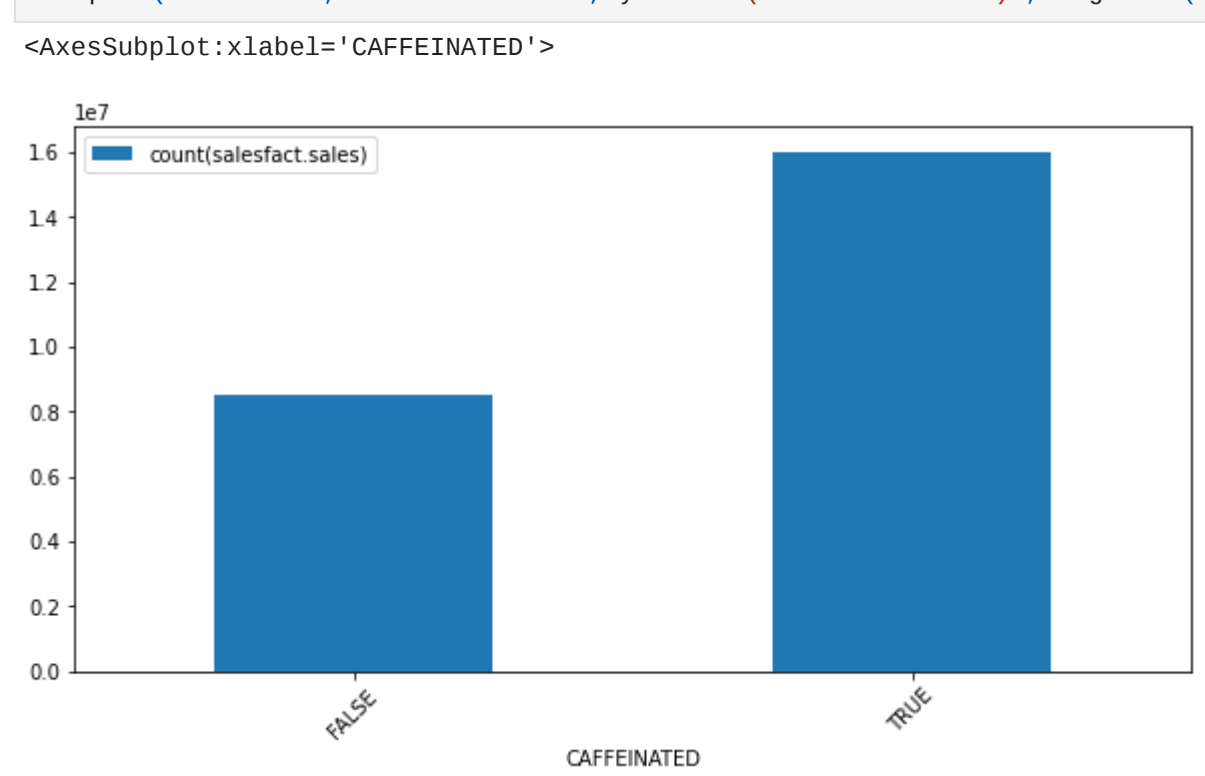
Out[28]:

	CAFFEINATED	count(salesfact.sales)
0	FALSE	8522948
1	TRUE	16002948

In [29]:
```
dfn.plot(kind='bar', x='CAFFEINATED', y='count(salesfact.sales)', figsize=(10,5), rot=45)
```

Out[29]:

```
<AxesSubplot:xlabel='CAFFEINATED'>
```



Business Question 3- Answer: Caffeinated drinks sold twice as much as the non caffeinated drinks. We also found that the Old Fashioned is the best selling caffeinated drink and the Grape drink is the best selling non caffeinated drink. Birch Beer is the lowest selling drink and should probably be discountinued.

In [ ]: