Do we have enough charging stations for EVs
Milestone: Final Report

Group 1

Han Kim
Arya Lokesh Gowda

919-904-8321 (Han) 214-972-5529 (Arya)

Kim.han1@northeastern.edu
lokeshgowda.a@northeastern.edu

Percentage of Effort Contributed by Han Kim: 100%
Percentage of Effort Contributed by Arya Gowda: 100%

Signature of Student 1: 

Signature of Student 2: 

Submission Date: 27 Nov 2023

Background information: EV Charging Stations

Han Kim & Arya Lokesh Gowda

Technological advance in electric vehicle and its price and environmental advantage over gas powered vehicles has led to higher adoption of Electric vehicles (EVs) across the United States. However, the number of charging stations available is beginning to lag due to an exponential increase in the number of EVs being sold. Electric vehicles have made new record sales recently as the number of vehicles sold exceeded 10 million in 2022 per international energy agency. Currently, we have about 14.5 vehicles per charging station. While this doesn't sound like a big deal, one must consider that it takes around 10 hours to fully charge a single EVs. This isn't a feasible number, especially with the number of EVs being sold is exponentially increasing by record numbers. Recent surveys suggest one-third of the American population are looking to switch to EVs soon and the number of sales is projected to rise even more.

Business problem:

Business users need a database that tracks all EV chargers and similar datasets to make sure we have enough EV chargers relative to growth of EV sales/manufacturing. Unlike typical internal combustion engine (ICE) vehicles that run on gasoline or diesel, electric vehicles (EVs) run on electricity stored in a battery pack. There are several types of electric vehicles, including battery electric vehicles (BEVs), which run solely on electricity, plug-in hybrid electric vehicles (PHEVs), which run on both electricity and gasoline, and hybrid electric vehicles (HEVs), which run on both electricity and gasoline but cannot be plugged in. Charging stations for EVs are a vital component of the infrastructure required to facilitate the broad deployment of electric automobiles. Charging stations can be broken down into three types, level 1, level 2, and level 3.

Level 1 chargers: normal 120-volt household outlets that provide the slowest charging speeds. They are the most accessible alternative and are generally utilized for overnight charging at home.
Level 2 chargers: Level 2 chargers, which operate at 240 volts, are faster than Level 1 chargers and are typically seen in residential settings, workplaces, and public charging stations.
Level 3 (DC Fast) Chargers: These chargers, which are often placed along highways and key routes, allow quick charging. They can charge an EV far faster than Level 1 or 2 chargers, making long-distance travel easier.

There are numerous factors that contributes to the growth of EV sales, but we will focus on the following requirements as requested by business users: increase in incentives for EV purchase from the government and overall cost savings for using EVs compared to gas powered cars on maintenance and rising gas prices. We will collect and correlate numerous factors like these to find similarities to highlight a potential increase in sales to determine if the growth in sales will continue. Upon determination, we will collect the number of charging stations and future installation count and see if charging infrastructure will be ready to handle the EV sales growth.

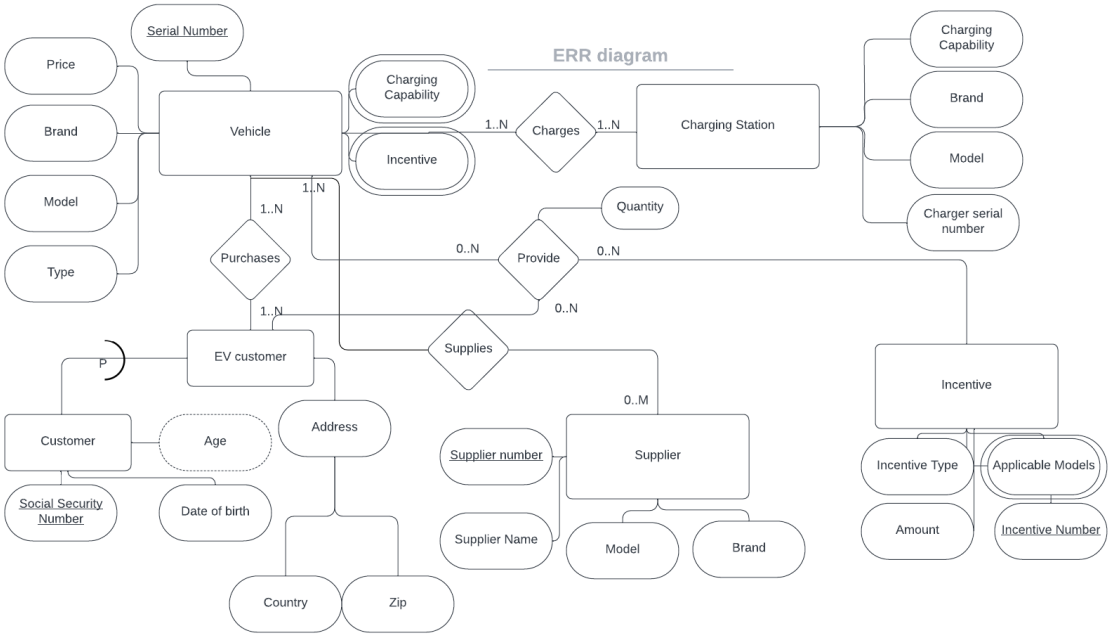
Other requirements:

1) Hybrid vehicles are powered by gas and electric; Vehicle must be one type, so hybrid is considered electric.
2) EVs can have zero to infinite incentives; incentives must be provided to at least one EV.
3) EVs can be charged on at least 1 charging station; charging stations can allow 0 to infinite EVs to charge at its stations.
4) A person can buy multiple EVs; But multiple people cannot own one EV at the same time.

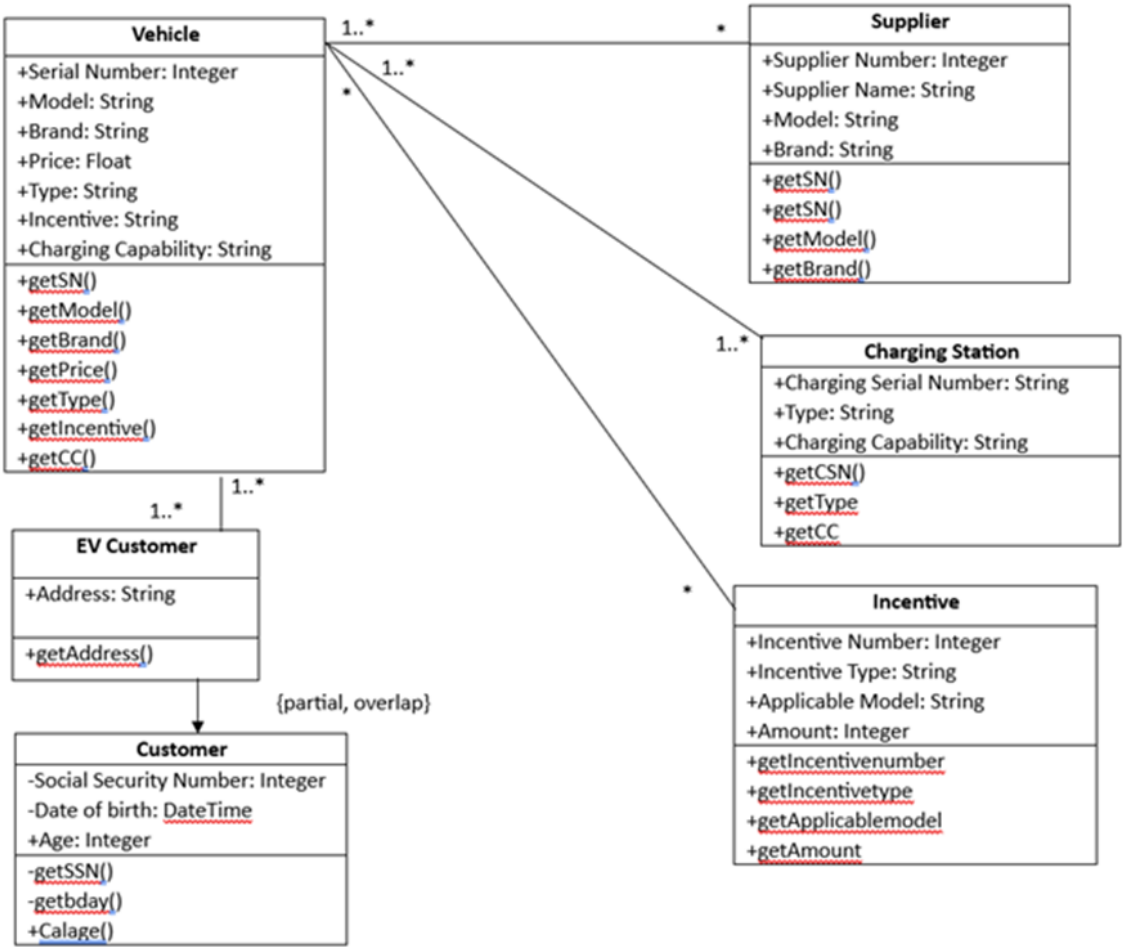
ERR Diagram: EV Market Research Database

The limitations of the above EER model are as follows:

- Certain incentives are available only on certain time frames
- Cannot specify domain and constraint values for incentive types



UML Diagram: EV Market Research Database



Do we have enough charging stations for EVs – Relational Model

- VEHICLE (Serial_Number, Brand, Price, Model, Type, Incentive, *supplier number*)
    - Serial number is the primary key and supplier number is the foreign key. Supplier number can't be declared **NULL** and additionally, refer to charges relation for relationship with charging station and applicable incentive for relationship with incentive

- CHARGING STATION (Charger_Serial_Number, Station_Charging_Capability, Brand, Model)
    - Charger serial number is the primary key and there is no foreign key, refer to charges relation for relationship with vehicle

- CHARGES (Charger_Serial_Number, Serial_Number)
    - This relation exists between vehicle and charging station and consists of two foreign keys that can't be **NULL**

- CUSTOMER (SSN, Date_of_birth)
    - Superclass of customer, SSN is the primary key

- EV_CUSTOMER (EV_Customer_SSN, Country, Zip, *Serial_Number*)
    - EV customer SSN is the primary key, subclass of customer and is partial so there are other types of customers besides EV customers and Serial number from vehicle is the foreign key and it is the SN of the vehicle they purchased from vehicle relation. SN of the vehicle can't be **NULL** for EV customer

- SUPPLIER (Supplier_Number, Supplier name, model, brand)
    - Supplier number is the primary key, there is no foreign key)

- INCENTIVE (Incentive_Number, applicable models, incentive type, amount)
    - Incentive number is the primary key and there is no foreign key. Refer to applicable incentives relation for relationship with vehicle

- APPLICABLE_INCENTIVES (Incentive_Number, Serial_Number)
    - Two are foreign keys from vehicle and incentive as there can be multiple incentives for certain serial number and serial number can have multiple incentive numbers

Do we have enough charging stations for EVs – Relational Model Tables

Vehicle						
SERIAL NUMBER	Type	Brand	Model	*Supplier Number*	Price	*Incentive Number*
123	Electric	Tesla	3	1234	40,000	12
124	Gasoline	Ford	Mustang	1235	30,000	13
….						

Charges	
CHARGER SERIAL NUMBER	*SERIAL NUMBER*
7777	3
7777	3
…..	

Charging Station			
CHARGER SERIAL NUMBER	Charger Brand	Charger Model	Station charging capability
7777	Tesla	Super Charger	Level 3
7778	Chargepoint	J1772 6.7kw	Level 1
….			

Customer	
Social Security Number	Date of birth
111-11-1111	09-09-1999
111-11-1112	09-09-1998
….	

EV Customer			
Social Security Number	*SERIAL NUMBER*	Country	Zip
111-11-1111	123	USA	12131
111-11-1113	125	USA	12132
….			

Supplier		
Supplier Number	Supplier Name	*SERIAL NUMBER*
1234	Tesla	123
1235	Toyota	125
….		

Incentive			
Incentive Number	Incentive Type	Amount	Applicable model
1	EV federal tax credit	20,000	Model 3
2	EV State tax credit	20,000	Model x
….			

Applicable Incentive	
Incentive Number	*SERIAL NUMBER*
1	123
….	

Loss of semantics:
-There are certain models that provide incentives up to "X" amount of vehicles. For example, first 100,000 models of tesla model X gets incentive Y. This semantic can't be captured in the relational model.
-There are certain incentives that only apply on certain date/time, so temporal semantic can't be captured in this model
-It isn't able to capture EV customers that return the vehicle afterwards
-It isn't able to capture address changes of the customers, temporal constraint

Creating all the Tables within EV chargers on MySQL


```
USE evchargers;

CREATE TABLE SUPPLIER (
   Supplier_Number INT PRIMARY KEY,
   Supplier_Name VARCHAR(255),
   Model VARCHAR(255),
   Brand VARCHAR(255)
);

CREATE TABLE VEHICLE (
   Serial_Number INT PRIMARY KEY,
   Brand VARCHAR(255),
   Price DECIMAL(10, 2),
   Model VARCHAR(255),
   Type VARCHAR(255),
   Incen ve INT,
   Supplier_Number INT NOT NULL,
   FOREIGN KEY (Supplier_Number) REFERENCES SUPPLIER(Supplier_Number)
);
CREATE TABLE CHARGING_STATION (
   Charger_Serial_Number INT PRIMARY KEY,
   Sta on_Charging_Capability INT,
   Brand VARCHAR(255),
   Model VARCHAR(255)
);
CREATE TABLE CHARGES (
   Charger_Serial_Number INT,
   Serial_Number INT,
   PRIMARY KEY (Charger_Serial_Number, Serial_Number),
   FOREIGN KEY (Charger_Serial_Number) REFERENCES
CHARGING_STATION(Charger_Serial_Number),
   FOREIGN KEY (Serial_Number) REFERENCES VEHICLE(Serial_Number)
);
CREATE TABLE CUSTOMER (
   SSN INT PRIMARY KEY,
   Date_of_birth DATE
);
CREATE TABLE EV_CUSTOMER (
```

```
    EV_Customer_SSN INT PRIMARY KEY,
    Country VARCHAR(255),
    Zip VARCHAR(10),
    Serial_Number INT NOT NULL,
    FOREIGN KEY (EV_Customer_SSN) REFERENCES CUSTOMER(SSN),
    FOREIGN KEY (Serial_Number) REFERENCES VEHICLE(Serial_Number)
);
CREATE TABLE INCENTIVE (
    Incen ve_Number INT PRIMARY KEY,
    Applicable_Models VARCHAR(255),
    Incen ve_Type VARCHAR(255),
    Amount DECIMAL(10, 2)
);
CREATE TABLE APPLICABLE_INCENTIVES (
    Incen ve_Number INT,
    Serial_Number INT,
    PRIMARY KEY (Incen ve_Number, Serial_Number),
    FOREIGN KEY (Incen ve_Number) REFERENCES INCENTIVE(Incen ve_Number),
    FOREIGN KEY (Serial_Number) REFERENCES VEHICLE(Serial_Number)
);
```

Crea ng and inser ng random en es for the tables/rela ons above in MySQL

```
INSERT INTO evchargers.VEHICLE (Serial_Number, Brand, Price, Model, Type, Incen ve,
Supplier_Number)
VALUES
(1, 'Tesla', 50000.00, 'Model S', 'Electric', 2000, 1),
(2, 'Chevrolet', 35000.00, 'Bolt', 'Electric', 1500, 2),
(3, 'Nissan', 30000.00, 'Leaf', 'Electric', 1800, 3),
(4, 'Ford', 45000.00, 'Mustang Mach-E', 'Electric', 1600, 4),
(5, 'BMW', 55000.00, 'i3', 'Electric', 2100, 5),
(6, 'Audi', 60000.00, 'e-Tron', 'Electric', 2300, 6),
(7, 'Hyundai', 40000.00, 'Kona Electric', 'Electric', 1700, 7),
(8, 'Kia', 38000.00, 'Soul EV', 'Electric', 1900, 8),
(9, 'Jaguar', 70000.00, 'I-PACE', 'Electric', 2500, 9),
(10, 'Porsche', 80000.00, 'Taycan', 'Electric', 2700, 10);

INSERT INTO evchargers.CHARGING_STATION (Charger_Serial_Number,
Sta on_Charging_Capability, Brand, Model)
VALUES
(1, 50, 'Tesla', 'Supercharger V3'),
(2, 40, 'ChargePoint', 'Express 250'),
(3, 30, 'Blink', 'DC Fast Charger'),
(4, 45, 'ABB', 'Terra 54 CJG'),
(5, 55, 'EVgo', 'Fast Charger'),
(6, 35, 'Siemens', 'Sicharge UC'),
(7, 60, 'Delta', 'DC City Fast Charger'),
(8, 70, 'Schneider Electric', 'EVLink DC Quick Charger'),
```

(9, 25, 'Efacec', 'QC45'),
(10, 65, 'BTC Power', 'DC Fast Charger');

INSERT INTO evchargers.CHARGES (Charger_Serial_Number, Serial_Number)
VALUES
(1, 1),
(2, 2),
(3, 3),
(4, 4),
(5, 5),
(6, 6),
(7, 7),
(8, 8),
(9, 9),
(10, 10);

INSERT INTO evchargers.CUSTOMER (SSN, Date_of_birth)
VALUES
(123456789, '1980-05-15'),
(234567890, '1992-09-25'),
(345678901, '1975-12-10'),
(456789012, '1988-03-20'),
(567890123, '1995-07-05'),
(678901234, '1982-11-30'),
(789012345, '1978-02-18'),
(890123456, '1987-08-14'),
(901234567, '1990-04-22'),
(912345678, '1984-06-28');

INSERT INTO evchargers.EV_CUSTOMER (EV_Customer_SSN, Country, Zip, Serial_Number)
VALUES
(123456789, 'USA', '12345', 1),
(234567890, 'USA', '23456', 2),
(345678901, 'USA', '34567', 3),
(456789012, 'USA', '45678', 4),
(567890123, 'USA', '56789', 5),
(678901234, 'USA', '67890', 6),
(789012345, 'USA', '78901', 7),
(890123456, 'USA', '89012', 8),
(901234567, 'USA', '90123', 9),
(912345678, 'USA', '91234', 10);

INSERT INTO evchargers.SUPPLIER (Supplier_Number, Supplier_Name, Model, Brand)
VALUES
(1, 'Tesla Inc.', 'Supercharger V3', 'Tesla'),
(2, 'ChargePoint Inc.', 'Express 250', 'ChargePoint'),
(3, 'Blink Charging Co.', 'DC Fast Charger', 'Blink'),
(4, 'ABB Ltd.', 'Terra 54 CJG', 'ABB'),

```
(5, 'EVgo Services LLC', 'Fast Charger', 'EVgo'),
(6, 'Siemens AG', 'Sicharge UC', 'Siemens'),
(7, 'Delta Electronics Inc.', 'DC City Fast Charger', 'Delta'),
(8, 'Schneider Electric SE', 'EVLink DC Quick Charger', 'Schneider Electric'),
(9, 'Efacec Power Solu ons', 'QC45', 'Efacec'),
(10, 'BTC Power Inc.', 'DC Fast Charger', 'BTC Power');

INSERT INTO evchargers.INCENTIVE (Incen ve_Number, Applicable_Models, Incen ve_Type,
Amount)
VALUES
(1, 'Model S, Model 3', 'Cash Rebate', 1000.00),
(2, 'Bolt, Spark EV', 'Discount', 800.00),
(3, 'Leaf, Ariya', 'Cash Rebate', 1200.00),
(4, 'Mustang Mach-E', 'Discount', 900.00),
(5, 'i3, i8', 'Cash Rebate', 1100.00),
(6, 'e-Tron, Q4 e-Tron', 'Discount', 1300.00),
(7, 'Kona Electric, Ioniq Electric', 'Cash Rebate', 950.00),
(8, 'Soul EV, Niro EV', 'Discount', 850.00),
(9, 'I-PACE, F-PACE', 'Cash Rebate', 1400.00),
(10, 'Taycan, Panamera Hybrid', 'Discount', 1500.00);

INSERT INTO evchargers.APPLICABLE_INCENTIVES (Incen ve_Number, Serial_Number)
VALUES
(1, 1),
(2, 2),
(3, 3),
(4, 4),
(5, 5),
(6, 6),
(7, 7),
(8, 8),
(9, 9),
(10, 10);
```

Tes ng out queries on MySQL with the database we've created

- Correlated query

```
SELECT c.SSN, c.Date_of_birth
FROM evchargers.CUSTOMER c
WHERE EXISTS (
   SELECT 1
   FROM evchargers.EV_CUSTOMER ec
   JOIN evchargers.APPLICABLE_INCENTIVES ai ON ec.Serial_Number =
ai.Serial_Number     WHERE c.SSN = ec.EV_Customer_SSN AND ai.Incen ve_Number = 1
);
```

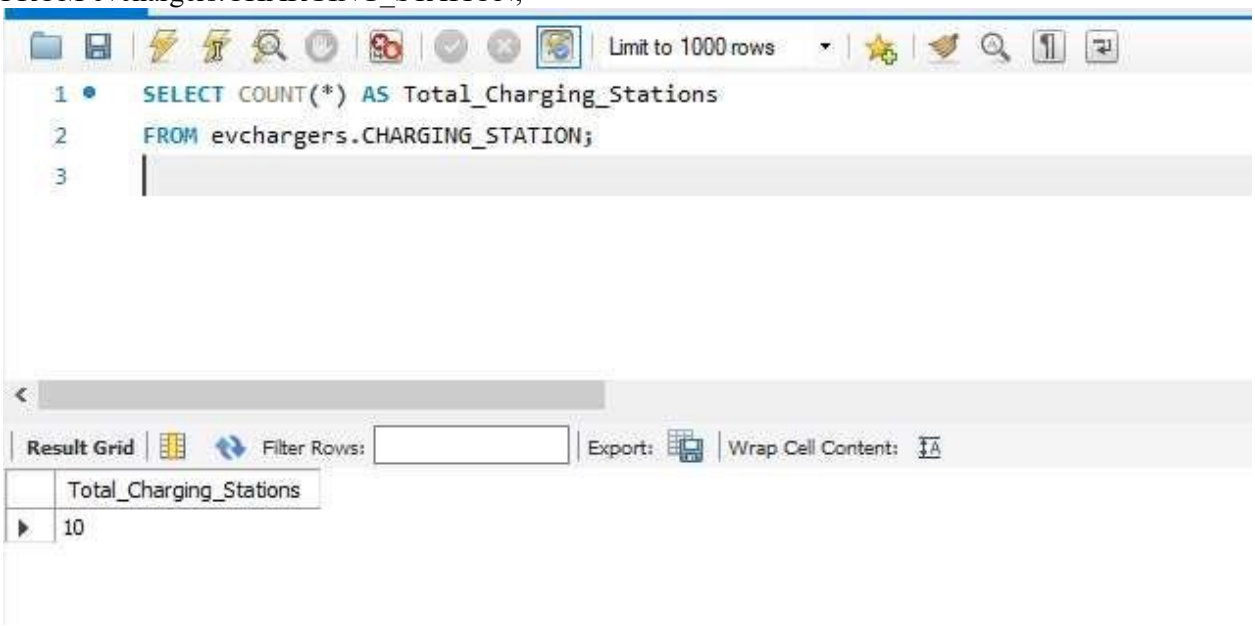
```
4   WHERE EXISTS (
5       SELECT 1
6       FROM evchargers.EV_CUSTOMER ec
7       JOIN evchargers.APPLICABLE_INCENTIVES ai ON ec.Serial_Number = ai.Serial_Number
8       WHERE c.SSN = ec.EV_Customer_SSN AND ai.Incentive_Number = 1
9   );
10
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

SSN	Date_of_birth
123456789	1980-05-15
NULL	NULL

- Count func on query

SELECT COUNT(*) AS Total_Charging_Sta ons
FROM evchargers.CHARGING_STATION;



- Join query

SELECT c.SSN, c.Date_of_birth, v.Brand, v.Model
FROM evchargers.CUSTOMER c
JOIN evchargers.EV_CUSTOMER ec ON c.SSN = ec.EV_Customer_SSN
JOIN evchargers.VEHICLE v ON ec.Serial_Number = v.Serial_Number;

```sql
SELECT c.SSN, c.Date_of_birth, v.Brand, v.Model
FROM evchargers.CUSTOMER c
JOIN evchargers.EV_CUSTOMER ec ON c.SSN = ec.EV_Customer_SSN
JOIN evchargers.VEHICLE v ON ec.Serial_Number = v.Serial_Number;
```

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

SSN	Date_of_birth	Brand	Model
123456789	1980-05-15	Tesla	Model S
234567890	1992-09-25	Chevrolet	Bolt
345678901	1975-12-10	Nissan	Leaf
456789012	1988-03-20	Ford	Mustang Mach-E
567890123	1995-07-05	BMW	i3
678901234	1982-11-30	Audi	e-Tron
789012345	1978-02-18	Hyundai	Kona Electric
890123456	1987-08-14	Kia	Soul EV
901234567	1990-04-22	Jaguar	I-PACE
912345678	1984-06-28	Porsche	Taycan

- Aggregated query

```sql
SELECT AVG(Price) AS Average_Price
FROM evchargers.VEHICLE;
```

```sql
SELECT AVG(Price) AS Average_Price
FROM evchargers.VEHICLE;
```

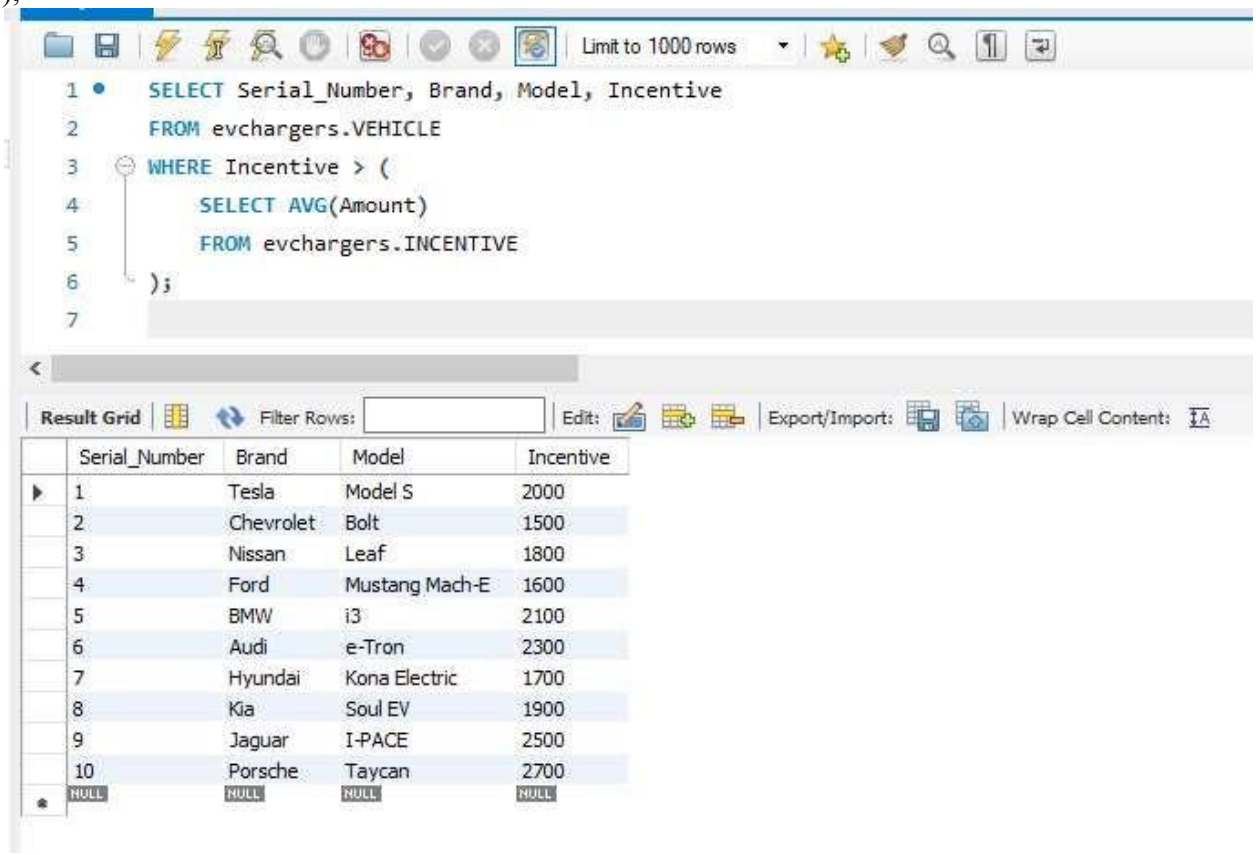
Result Grid | Filter Rows: | Export: | Wrap Cell Content:

Average_Price
50300.000000

- Nested query

```sql
SELECT Serial_Number, Brand, Model, Incen ve
FROM evchargers.VEHICLE
WHERE Incen ve > (
  SELECT AVG(Amount)
  FROM evchargers.INCENTIVE
```

);



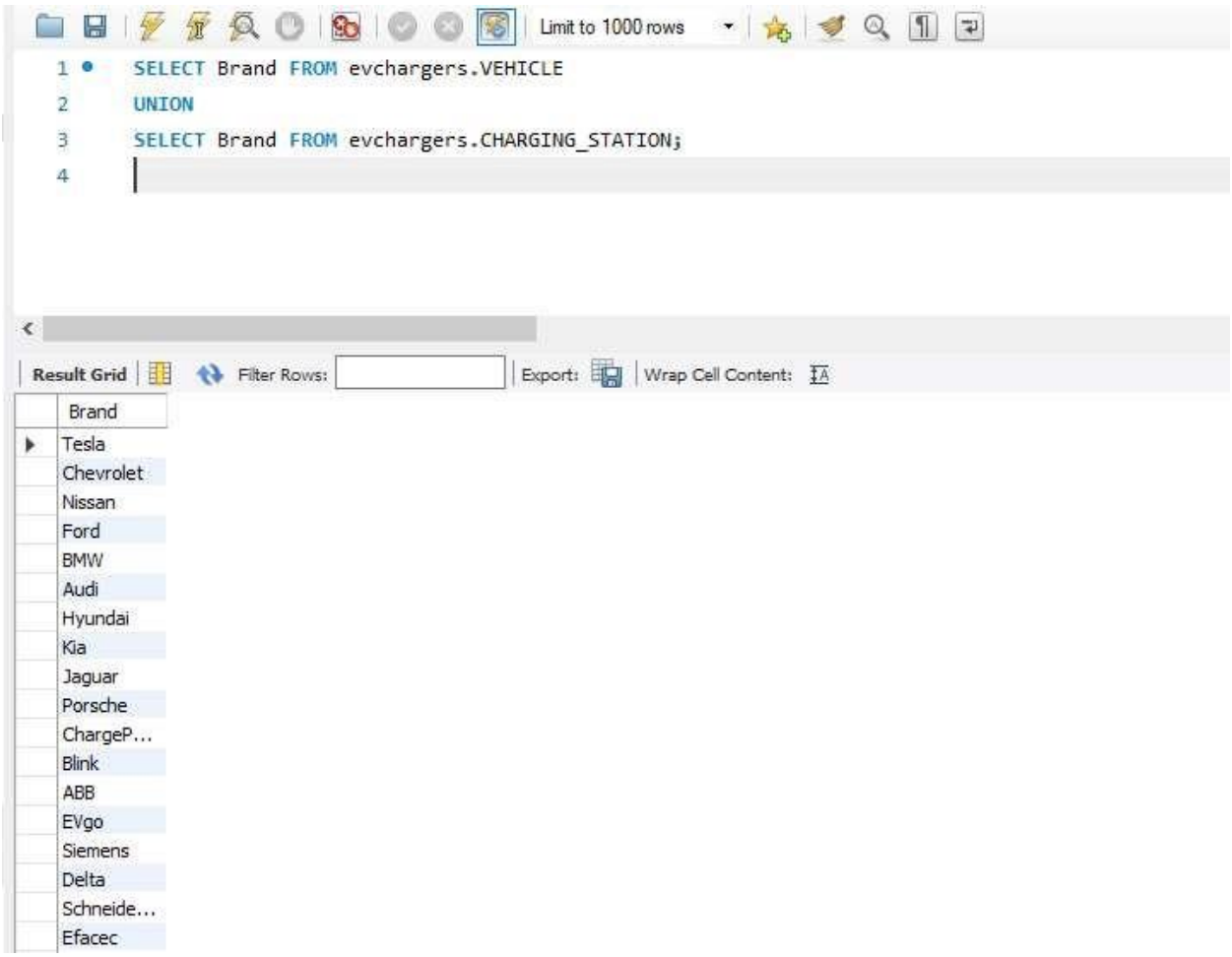
```
SELECT Serial_Number, Brand, Model, Incentive
FROM evchargers.VEHICLE
WHERE Incentive > (
    SELECT AVG(Amount)
    FROM evchargers.INCENTIVE
);
```

Result Grid | Filter Rows: | Edit: | Export/Import: | Wrap Cell Content:

Serial_Number	Brand	Model	Incentive
1	Tesla	Model S	2000
2	Chevrolet	Bolt	1500
3	Nissan	Leaf	1800
4	Ford	Mustang Mach-E	1600
5	BMW	i3	2100
6	Audi	e-Tron	2300
7	Hyundai	Kona Electric	1700
8	Kia	Soul EV	1900
9	Jaguar	I-PACE	2500
10	Porsche	Taycan	2700
NULL	NULL	NULL	NULL

- Set opera on query

SELECT Brand FROM evchargers.VEHICLE
UNION
SELECT Brand FROM evchargers.CHARGING_STATION;


Limit to 1000 rows
1 SELECT Brand FROM evchargers.VEHICLE
2 UNION
3 SELECT Brand FROM evchargers.CHARGING_STATION;
4
Result Grid
Filter Rows:
Export:
Wrap Cell Content:
Brand
Tesla
Chevrolet
Nissan
Ford
BMW
Audi
Hyundai
Kia
Jaguar
Porsche
ChargeP...
Blink
ABB
EVgo
Siemens
Delta
Schneide...
Efacec

Visualization in Python using Jupyter Notebook with MySQL Database:

1. Library Installation:
- Install the `mysql-connector` library in Jupyter Notebook using the command `pip install mysqlconnector`.

2. Database Connection Setup:
- Establish a connection to the MySQL database by providing necessary details such as user, password, host, and database name using the `mysql.connector.connect` method.

3. Cursor Object Creation:
- Create a cursor object using the `cnx.cursor()` method. This cursor will be used to execute SQL queries.

4. SQL Query Execution:
- Execute SQL queries to interact with the database. Query to retrieve all table names.

5. Fetch Query Results:
- Use the cursor object to retrieve the results of the executed SQL query.

6. Data Analysis:
- Analyze the retrieved data as needed. For example, we fetched table names and vehicle models.

7. DataFrame Creation:
- Convert SQL query results into a Pandas DataFrame for easier manipulation and visualization.

8. Data Visualization:
- Use a bar plot to visualize data. We created a bar plot to show the distribution of charger models.
- Alternatively, use a violin plot for a different perspective on data distribution. For instance, we used a violin plot to display incentive amounts by model. Customize the appearance of plots, including labels, titles, and other relevant details. Show the generated plots within the Jupyter Notebook.

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

```
Collecting mysql-connector-python
  Downloading mysql_connector_python-8.2.0-cp311-cp311-
macosx_12_0_a rm64.whl.metadata (2.1 kB)
Collecting protobuf<=4.21.12,>=4.21.1 (from mysql-connector-
python)
  Downloading protobuf-4.21.12-cp37-abi3-
macosx_10_9_universal2.whl (486 kB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━

486.2/486.2 kB 2.1 MB/s eta 0:00:00a 0:00:01
Downloading mysql_connector_python-8.2.0-cp311-cp311-
macosx_12_0_arm
64.whl (14.5 MB)

━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━━

━━ 14.5/14.5 MB 43.8 MB/s e ta 0:00:0000:0100:01
Installing collected packages: protobuf, mysql-connector-python
Successfully installed mysql-connector-python-8.2.0 protobuf-
4.21.12
Note: you may need to restart the kernel to use updated packages.
```

In [5]:
```
import mysql.connector

# Set up the connection details
cnx = mysql.connector.connect(
    user='root',
    password='arya1234',
    host='localhost',
    database='evchargers'
)

# Create a cursor object
mycursor = cnx.cursor()

# Execute a query to get all table names in the database
mycursor.execute("SHOW TABLES")

# Fetch the results
tables = mycursor.fetchall()

# Print the table names
for table in tables:
    print(table[0])
```

```
applicable_incentives
charges
```

```
charging_sta
tion
customer
ev_customer
incentive
supplier
vehicle
```

In [7]:
```
import pandas as pd

query = 'SELECT * FROM vehicle'
df_vehicle = pd.read_sql(query, con=cnx)

# Display the DataFrame
df_vehicle
```

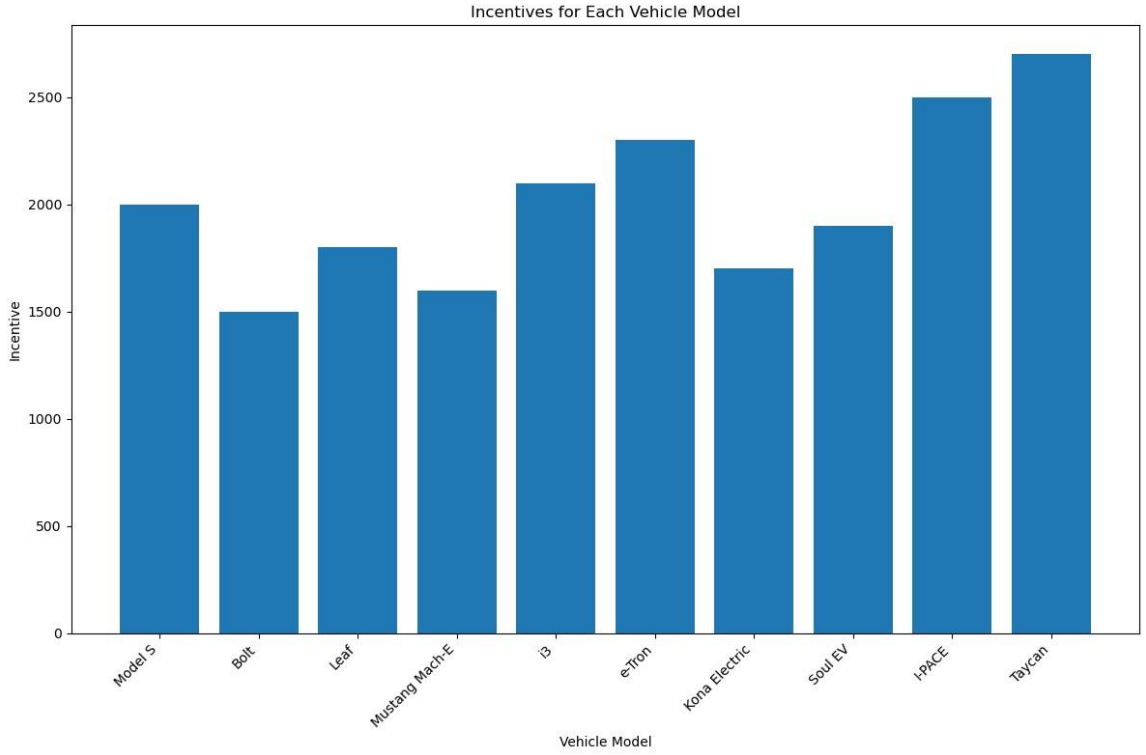
```
/var/folders/_9/rd5r68s114q89v4t4wrv38y00000gn/T/ipykernel_7970/3
  169 583952.py:4: UserWarning: pandas only supports SQLAlchemy
   connectabl e (engine/connection) or database string URI or
sqlite3 DBAPI2 conne ction. Other DBAPI2 objects are not tested.
                    Please consider using SQ
LAlchemy.   df_vehicle =
pd.read_sql(query, con=cnx)
```

Out[7]:

	Serial_Number	Brand	Price	Model	Type		IncentiveSupplier_Number
0	1	Tesla	50000.0	Model S	Electric	2000	1
1	2	Chevrolet	35000.0	Bolt	Electric	1500	2
2	3	Nissan	30000.0	Leaf	Electric	1800	3
3	4	Ford	45000.0	Mustang Mach-E	Electric	1600	4
4	5	BMW	55000.0	i3	Electric	2100	5
5	6	Audi	60000.0	e-Tron	Electric	2300	6
6	7	Hyundai	40000.0	Kona Electric	Electric	1700	7
7	8	Kia	38000.0	Soul EV	Electric	1900	8
8	9	Jaguar	70000.0	I-PACE	Electric	2500	9
9	10	Porsche	80000.0	Taycan	Electric	2700	10

Plotting a bar plot to analyze the incentives

In [10]:
```
#. Plotting Incentives
import matplotlib.pyplot as plt

plt.figure(figsize=(12, 8))
plt.bar(df_vehicle['Model'], df_vehicle['Incentive'])
plt.xlabel('Vehicle Model')
plt.ylabel('Incentive')
plt.title('Incentives for Each Vehicle Model')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

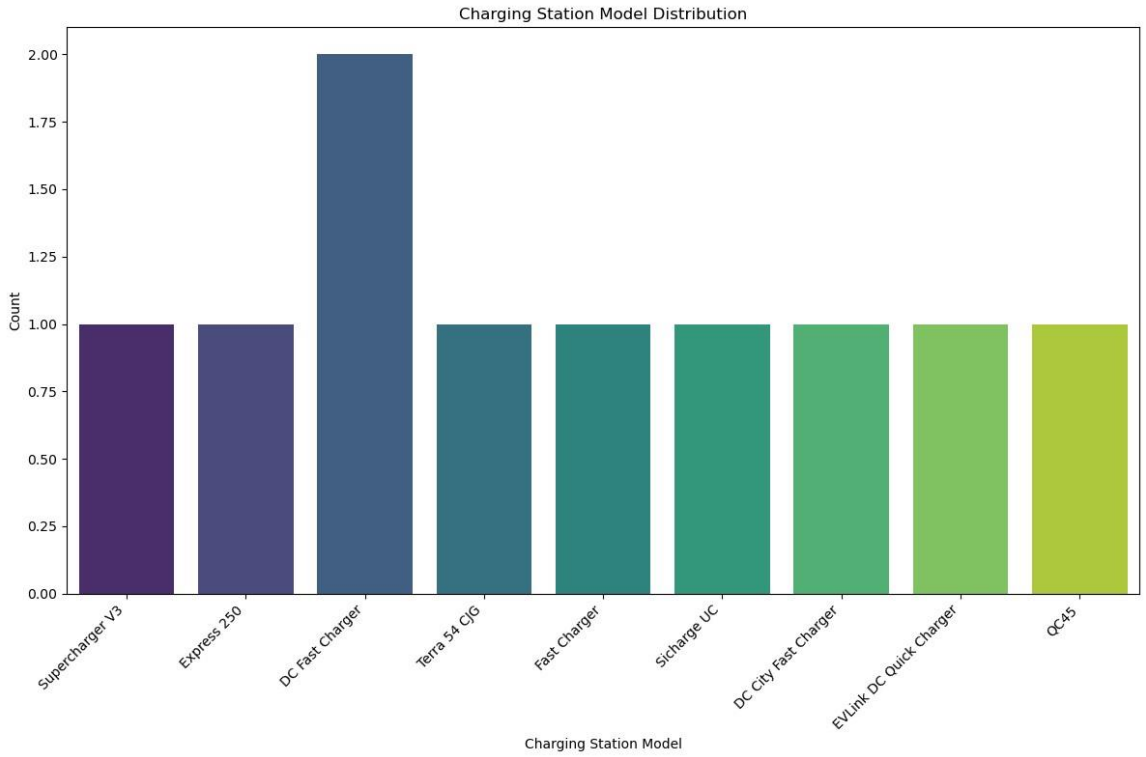


In [23]:
```
import seaborn as sns
# Create DataFrames by querying the  'charges' and
'charging_station' query_charges = 'SELECT * FROM charges'
query_charging_station = 'SELECT * FROM charging_station'

df_charges = pd.read_sql(query_charges, con=cnx)
df_charging_station = pd.read_sql(query_charging_station,
con=cnx)

# Join 'vehicle' and 'charges' tables by serial_number
df_merged = pd.merge(df_vehicle, df_charges, how='inner',
on='Serial_
# Join 'charges' and 'charging_station' tables by charger_serial_numb
df_merged = pd.merge(df_merged, df_charging_station, how='inner', lef
```

```
df_me
rged


/var/folders/_9/rd5r68s114q89v4t4wrv38y00000gn/T/ipykernel_7970/2
  304 885676.py:6: UserWarning: pandas only supports SQLAlchemy
   connectabl e (engine/connection) or database string URI or
sqlite3 DBAPI2 conne ction. Other DBAPI2 objects are not tested.
                    Please consider using SQ
LAlchemy.   df_charges =
pd.read_sql(query_charges, con=cnx)
/var/folders/_9/rd5r68s114q89v4t4wrv38y00000gn/T/ipykernel_7970/2
  304 885676.py:7: UserWarning: pandas only supports SQLAlchemy
   connectabl e (engine/connection) or database string URI or
sqlite3 DBAPI2 conne ction. Other DBAPI2 objects are not tested.
                    Please consider using SQ
LAlchemy.   df_charging_station =
pd.read_sql(query_charging_station, con=cnx) Out[23]:
```

Serial_Number	Brand_x	Price	Model_x	Type	IncentiveSupplier_Number	Charger_Seria
1	Tesla	50000.0	Model S	Electric	2000	1
2	Chevrolet	35000.0	Bolt	Electric	1500	2
3	Nissan	30000.0	Leaf	Electric	1800	3
4	Ford	45000.0	Mustang Mach-E	Electric	1600	4
5	BMW	55000.0	i3	Electric	2100	5
6	Audi	60000.0	e-Tron	Electric	2300	6
7	Hyundai	40000.0	Kona Electric	Electric	1700	7
8	Kia	38000.0	Soul EV	Electric	1900	8
9	Jaguar	70000.0	I-PACE	Electric	2500	9
10	Porsche	80000.0	Taycan	Electric	2700	10

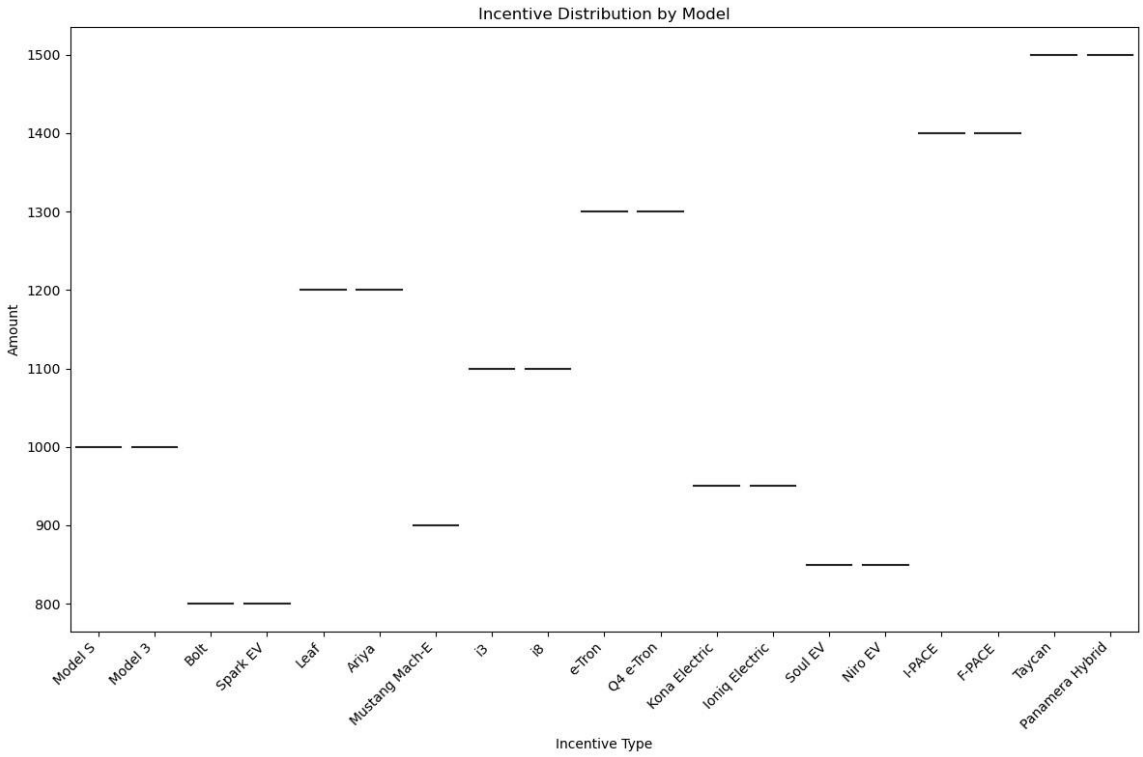
In [24]:
```
# Analyze the 'Model' of the charger from the charging station
charger_model_counts = df_merged['Model_y'].value_counts()

# Visualize with a count plot
plt.figure(figsize=(12, 8))
sns.countplot(data=df_merged, x='Model_y', palette='viridis')
plt.xlabel('Charging Station Model')
plt.ylabel('Count')
plt.title('Charging Station Model Distribution')
plt.xticks(rotation=45, ha='right')
plt.tight_layout()
plt.show()
```

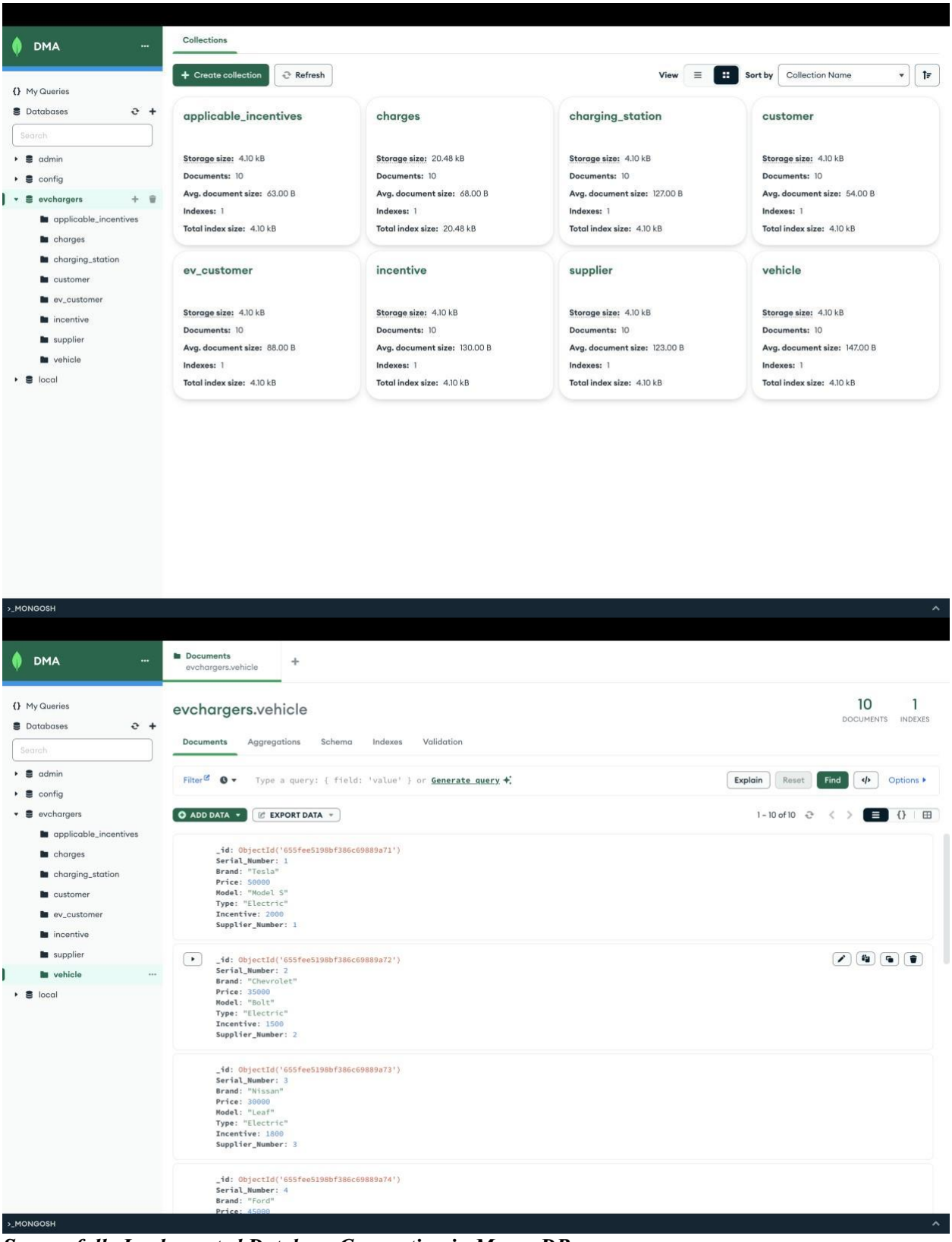


In [30]:
```
query_incentive = 'SELECT * FROM incentive'
df_incentive = pd.read_sql(query_incentive,
con=cnx)

df_incentive['Applicable_Models'] =
df_incentive['Applicable_Models']
# Create a long-form DataFrame for seaborn
df_long = pd.DataFrame({'Applicable_Models': model, 'Amount':
amount}

plt.figure(figsize=(12, 8))
sns.violinplot(data=df_long, x='Applicable_Models', y='Amount',
inner plt.xlabel('Incentive Type') plt.ylabel('Amount')
```

```
plt.title('Incentive Distribution by
Model') plt.xticks(rotation=45,
ha='right') plt.tight_layout()
plt.show()
```

```
/var/folders/_9/rd5r68s114q89v4t4wrv38y00000gn/T/ipykernel_7970/2
  147 890638.py:2: UserWarning: pandas only supports SQLAlchemy
    connectabl e (engine/connection) or database string URI or
 sqlite3 DBAPI2 conne ction. Other DBAPI2 objects are not tested.
                     Please consider using SQ
LAlchemy.   df_incentive =
pd.read_sql(query_incentive, con=cnx)
```

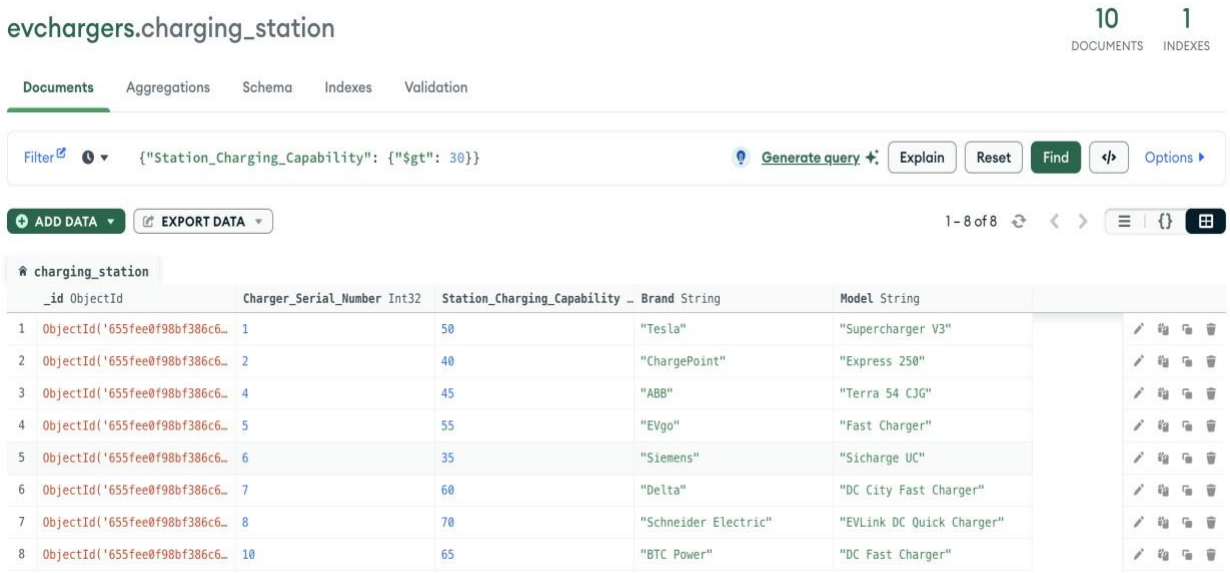




Successfully Implemented Database Connection in MongoDB

1. Write a query to retrieve charging stations with capability greater than 30.

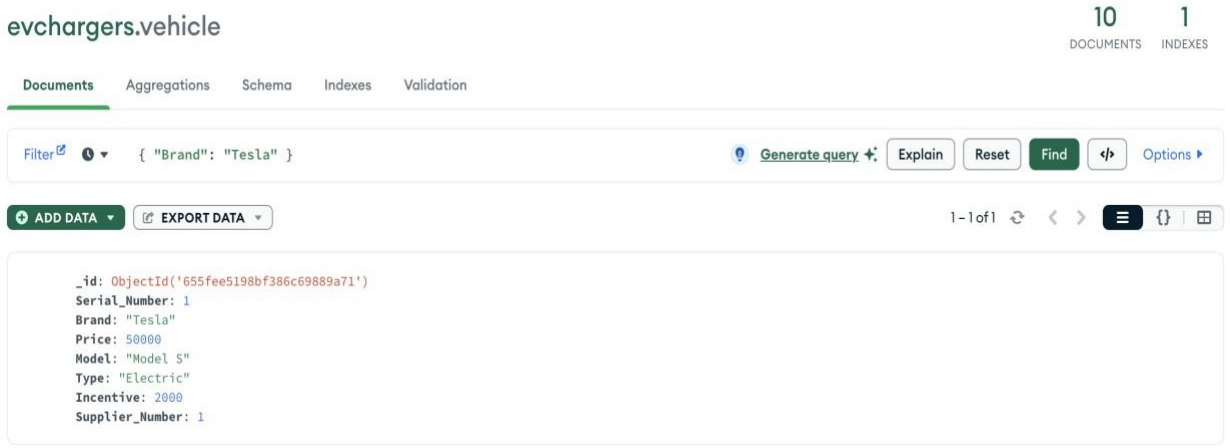
{"Station_Charging_Capability": {"$gt": 30}}



	_id ObjectId	Charger_Serial_Number Int32	Station_Charging_Capability …	Brand String	Model String
1	ObjectId('655fee0f98bf386c6…	1	50	"Tesla"	"Supercharger V3"
2	ObjectId('655fee0f98bf386c6…	2	40	"ChargePoint"	"Express 250"
3	ObjectId('655fee0f98bf386c6…	4	45	"ABB"	"Terra 54 CJG"
4	ObjectId('655fee0f98bf386c6…	5	55	"EVgo"	"Fast Charger"
5	ObjectId('655fee0f98bf386c6…	6	35	"Siemens"	"Sicharge UC"
6	ObjectId('655fee0f98bf386c6…	7	60	"Delta"	"DC City Fast Charger"
7	ObjectId('655fee0f98bf386c6…	8	70	"Schneider Electric"	"EVLink DC Quick Charger"
8	ObjectId('655fee0f98bf386c6…	10	65	"BTC Power"	"DC Fast Charger"

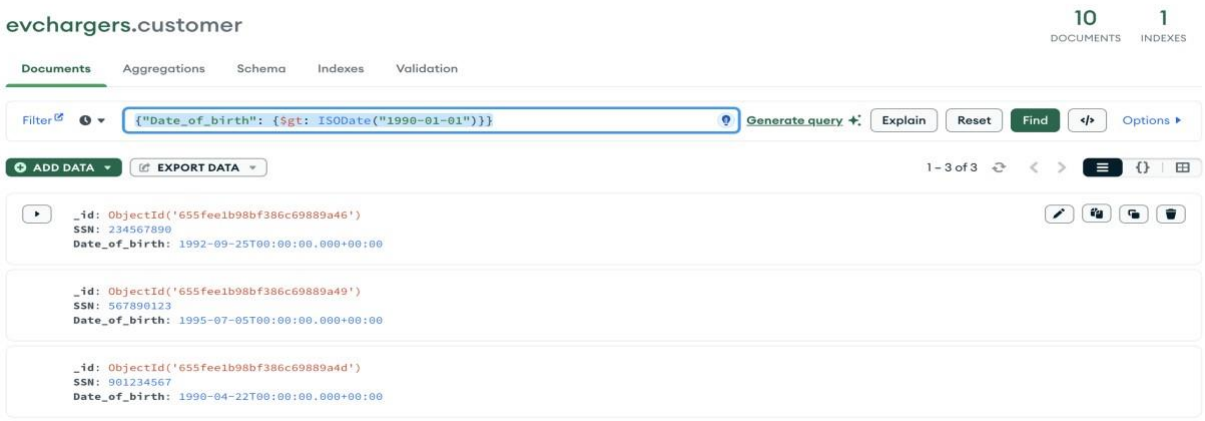
2. Find all vehicles of a specific brand (e.g., "Tesla"):

{ "Brand": "Tesla" }



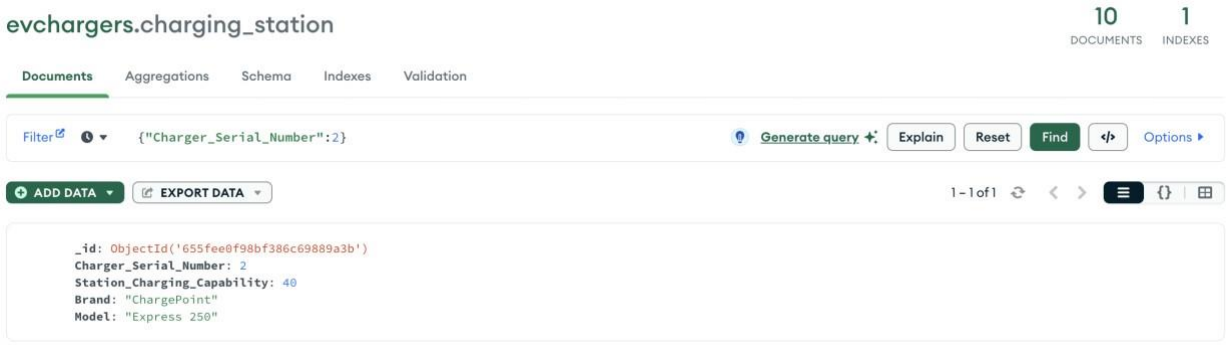
3. Retrieve customers born after a certain date (e.g., January 1, 1990):

```
{"Date_of_birth": {$gt: ISODate("1990-01-01")}}
```



4. Get the details of a specific charging station by its serial number:

```
{"Charger_Serial_Number":2}
```



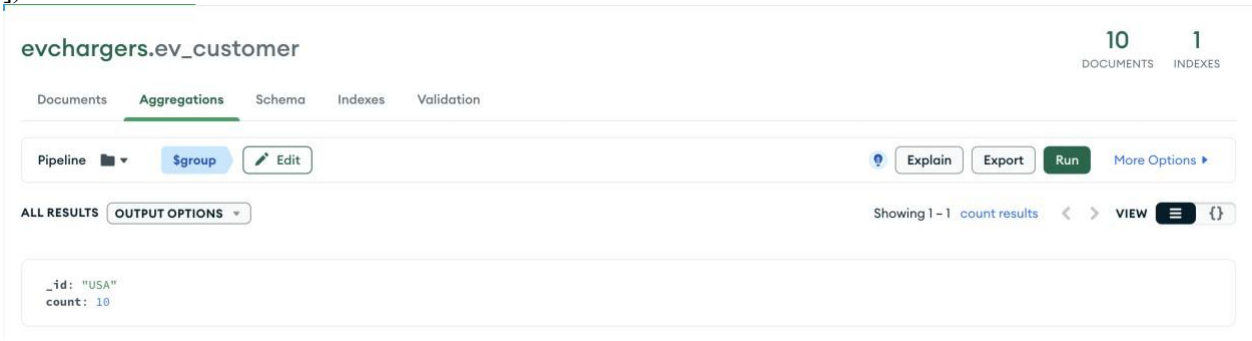
5. Calculate the average charging capability of all charging stations:

```
db.charging_stations.aggregate([
  {
    $group: {      _id: null,
      averageChargingCapability: { $avg: "$Station_Charging_Capability" }
    }
  }
])
```

6. Count the number of EV customers from each country:

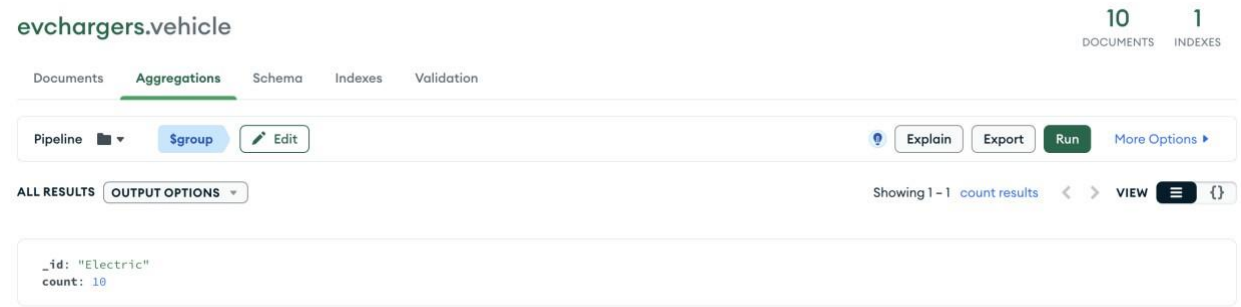
```
db.ev_customers.aggregate([
  {
    $group: {
```

```
    _id: "$Country",
    count: { $sum: 1 }
   }
  }
])
```



7. Count the Number of Vehicles by Type:

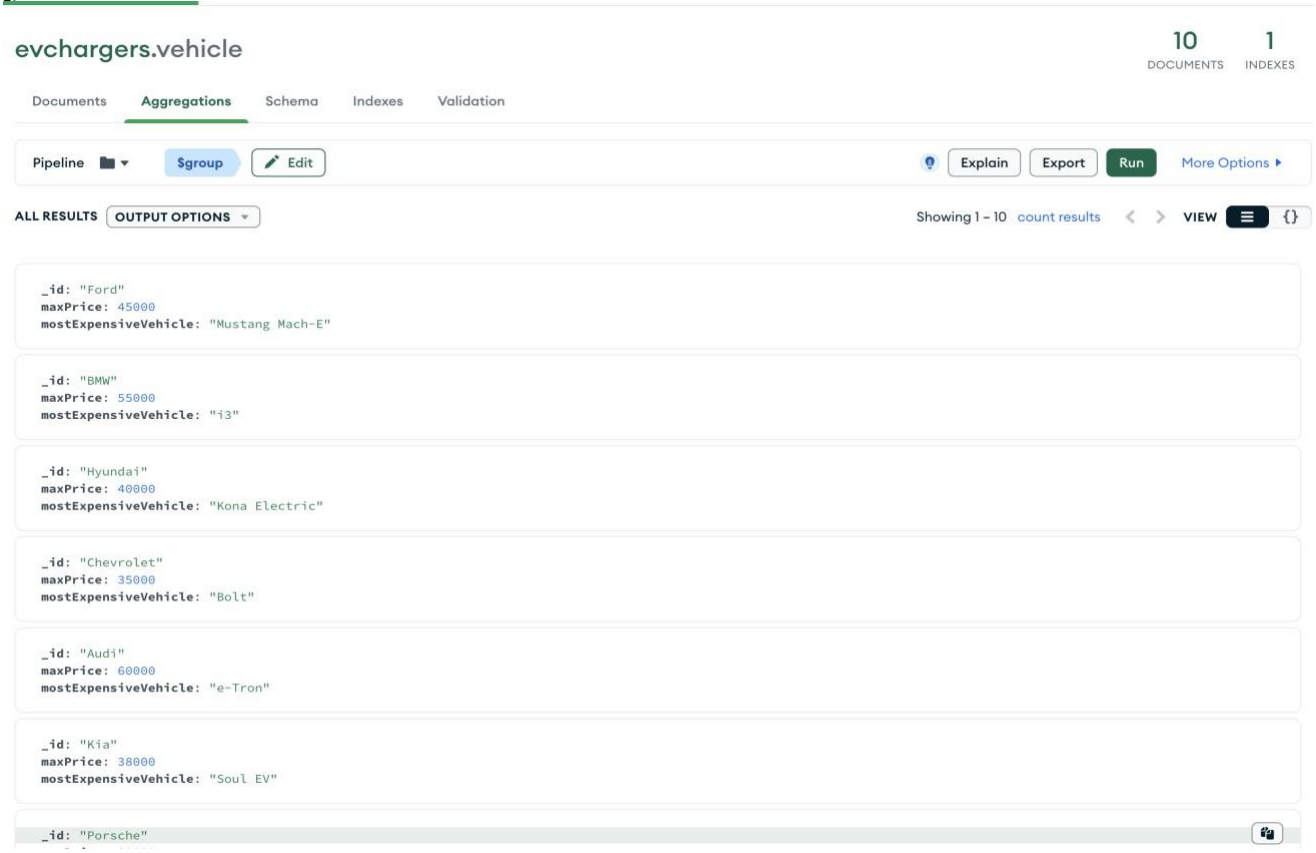
```
db.vehicles.aggregate([
  {
   $group: {      _id: "$Type",
    count: { $sum: 1 }
   }
  }
])
```



8. Find the Most Expensive Vehicle for Each Brand:

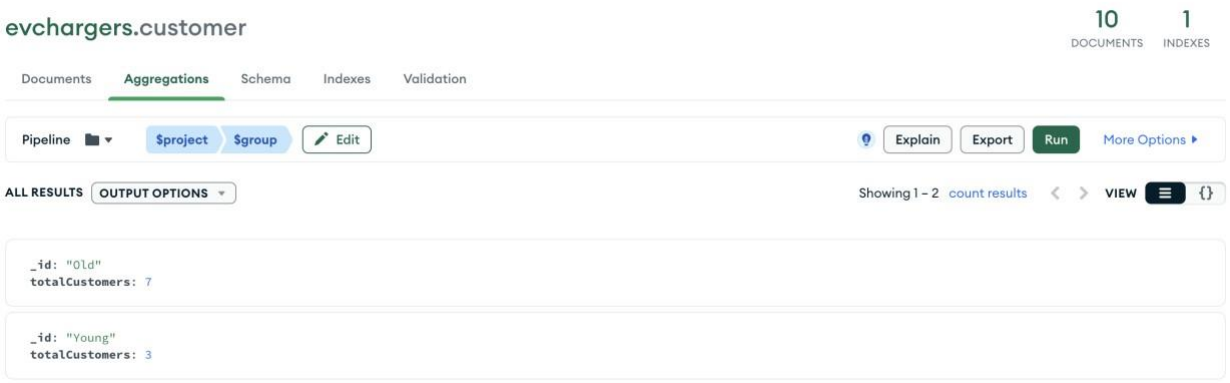
```
db.vehicles.aggregate([
  {
   $group: {      _id: "$Brand",
    maxPrice: { $max: "$Price" },
    mostExpensiveVehicle: { $first: "$Model" }  // Assuming you want to take the first model if there are multiple with the same max price
```

```
  }
 }
])
```



9. Calculate the Total Number of Customers by Age Group:

```
db.customers.aggregate([
  {
   $project: {      ageGroup: {         $cond: {
        if: { $lt: ["$Date_of_birth", new Date("1990-01-01")] },          then: "Old",          else: "Young"
       }
     }
   }
  },
  {
   $group: {
    _id: "$ageGroup",
    totalCustomers: { $sum: 1 }
   }
  }
])
```



CONCLUSION:

EVs are projected to continue its sales growth at accelerated rate, so it is imperative that a database management system be created by the government and businesses to track the infrastructure that supports the EVs to ensure there is no crowded locations and create traffic issues in certain places. The database serves as a central repository for data for all EV growth factors and EV infrastructure factors like consumer demographics and market trends, which can be examined to create efficient infrastructure support plan to improve customer experiences.

The project's future scope is important since it offers potential for the database to be further developed and expanded to incorporate new entities and attributes. For instance, the database can be updated to contain details regarding the manufacturing process for EVs as well as its infrastructure, as well as certifications and regulatory compliance. The database can also be connected to social media sites, allowing for real-time customer interaction and input.

Additionally, machine learning algorithms can be included in the application to recommended charging stations based on user travel routes and user's vehicles. In conclusion, database management presents a promising way for companies and government workers to increase user infrastructure, adapt to shifting to green and better environment, and expand the effort to replace gas power vehicles.