In [1]:
```
%load_ext sql
```

In [23]:
```
#creating a dimention table for top listing car models
%%sql postgresql://aqlxfqja:F6bE-fv-jhA_VaaLV284XVgxXOLNAp_2@rosie.db.elephantsql.co

CREATE TABLE IF NOT EXISTS car_dimentions_v1 AS
(
WITH grouped AS(

    SELECT manufacturer
         , model
         , COUNT(id) AS count
      FROM us_carsales_v1
     WHERE year > 1990
     GROUP BY 1, 2
     ORDER BY 3 DESC
     LIMIT 100
), counted AS(

    SELECT manufacturer
        , model
        , size
        , type
        , COUNT(1) OVER (PARTITION BY manufacturer, model, size) AS cnt_size
        , COUNT(1) OVER (PARTITION BY manufacturer, model, type) AS cnt_type
    FROM us_carsales_v1
    WHERE size IS NOT NULL AND type IS NOT NULL AND model in (SELECT model FROM grou
    ORDER BY 5 DESC, 6 DESC

), maxed AS(

    SELECT manufacturer
        , model
        , MAX(cnt_size) AS max_size
        , MAX(cnt_type) AS max_type
    FROM counted
    GROUP BY 1, 2

), joined AS(

    SELECT DISTINCT a.*
        , b.size
        , b.type
        , b.cnt_size
        , b.cnt_type
    FROM maxed AS a
    LEFT JOIN counted AS b
        ON a.manufacturer = b.manufacturer AND a.model = b.model

), filtered AS(

SELECT manufacturer
     , model
     , CASE WHEN cnt_size = max_size THEN size
            ELSE NULL
             END AS size
     , CASE WHEN cnt_type = max_type THEN type
            ELSE NULL
             END AS type
  FROM joined
)
SELECT manufacturer
     , model
```

```
     , MAX(size) AS size
     , MAX(type) AS type
  FROM filtered
 GROUP BY 1,2
)
```

```
Done.
```

Out[23]:
```
[]
```

In [5]:
```
# adding car age range, odometer and price range to the table. Then clean data and g

df_fact = psql.read_sql('SELECT * from us_carsales_v1', con)
df_dimention = psql.read_sql('SELECT * from car_dimentions_v1', con)
```

In [6]:
```
df_cleaned = sqldf('''

WITH enriched AS(

    SELECT a.id
        , a.manufacturer
        , a.model
        , b.type
        , b.size
        , a.years_old
        , a.odometer
        , a.price
        , COUNT(*) OVER (PARTITION BY a.manufacturer, a.model, a.years_old) AS cnt
        , row_number() OVER (PARTITION BY a.manufacturer, a.model, a.years_old ORDER
    FROM df_fact AS a
    LEFT JOIN df_dimention AS b
      ON a.manufacturer = b.manufacturer AND a.model = b.model
    WHERE a.manufacturer in (SELECT manufacturer FROM df_dimention)
          AND a.model in (SELECT model FROM df_dimention)
          AND a.year > '1990'
          AND a.years_old >= 0
          AND a.price BETWEEN 1000 and 500000
          AND a.odometer between 1000 AND 300000

), cated AS(

   SELECT *
        , CASE
            WHEN years_old <= 3 THEN '0 to 3'
            WHEN years_old BETWEEN 4 AND 5 THEN '4 to 5'
            WHEN years_old BETWEEN 6 AND 10 THEN '6 to 10'
            WHEN years_old BETWEEN 11 AND 15 THEN '11 to 15'
            ELSE '15+'
            END AS year_cat
        , CASE
            WHEN odometer <= 10000 THEN 'less than 10000'
            WHEN odometer BETWEEN 10001 AND 30000 THEN '10000 to 30000'
            WHEN odometer BETWEEN 30001 AND 50000 THEN '30000 to 50000'
            WHEN odometer BETWEEN 50001 AND 100000 THEN '50000 to 100000'
            WHEN odometer BETWEEN 100001 AND 150000 THEN '100000 to 150000'
            ELSE 'more than 150000'
            END AS odo_cat
        , CASE
            WHEN price <= 10000 THEN 'less than 10k'
            WHEN price BETWEEN 10001 AND 20000 THEN '10k to 20k'
            WHEN price BETWEEN 20001 AND 30000 THEN '20k to 30k'
            WHEN price BETWEEN 30001 AND 50000 THEN '30k to 50k'
            WHEN price BETWEEN 50001 AND 70000 THEN '50k to 70k'
            WHEN price BETWEEN 70001 AND 100000 THEN '70k to 100k'
            WHEN price BETWEEN 100001 AND 150000 THEN '100k to 150k'
```

```
            ELSE 'more than 150k'
            END AS price_cat
    FROM enriched

), ranked AS(

  SELECT *
      , COUNT(id) OVER (PARTITION BY manufacturer, model, year_cat, odo_cat) AS cnt
      , row_number() OVER (PARTITION BY manufacturer, model, year_cat, odo_cat ORDER
    FROM cated

), percented AS(

  SELECT id
      , manufacturer
      , model
      , type
      , size
      , years_old
      , odometer
      , price
      , year_cat
      , odo_cat
      , price_cat
    FROM ranked

), maxed AS(

  SELECT manufacturer
      , model
      , year_cat
      , odo_cat
      , price_cat
      , MAX(price) AS price_guide
    FROM ranked
  GROUP BY 1,2,3,4,5

)
SELECT a.*
     , b.price_guide
  FROM percented as a
  LEFT JOIN maxed as b
    ON a.manufacturer = b.manufacturer
   AND a.model = b.model
   AND a.year_cat = b.year_cat
   AND a.odo_cat = b.odo_cat
   AND a.price_cat = b.price_cat
''')
```

In [7]:
```
df_cleaned.to_csv('df_cleaned.csv', index = False)
```

In [1]:
```
# two columns year_cat and odo_cat will be created
# each model will receive a guide price that can cover at least 80 percent of cars w

df_matrix = sqldf('''

WITH enriched AS(

    SELECT a.id
        , a.manufacturer
        , a.model
        , b.type
        , b.size
        , a.years_old
```

```
        , a.odometer
        , a.price
    FROM df_fact AS a
    LEFT JOIN df_dimention AS b
      ON a.manufacturer = b.manufacturer AND a.model = b.model
    WHERE a.manufacturer in (SELECT manufacturer FROM df_dimention)
          AND a.model in (SELECT model FROM df_dimention)
          AND a.year > '1990'
          AND a.price BETWEEN 1000 AND 500000
          AND a.odometer BETWEEN 1000 AND 300000

), cated AS(

   SELECT *
        , CASE
            WHEN years_old <= 3 THEN '0-3'
            WHEN years_old BETWEEN 3 AND 5 THEN '3-5'
            WHEN years_old BETWEEN 5 AND 10 THEN '5-10'
            WHEN years_old BETWEEN 10 AND 15 THEN '10-15'
            ELSE '15+'
            END AS year_cat
        , CASE
            WHEN odometer <= 10000 THEN 'less than 10000'
            WHEN odometer BETWEEN 10000 AND 30000 THEN '10000 to 30000'
            WHEN odometer BETWEEN 30000 AND 50000 THEN '30000 to 50000'
            WHEN odometer BETWEEN 50000 AND 100000 THEN '50000 to 100000'
            WHEN odometer BETWEEN 100000 AND 150000 THEN '100000 to 150000'
            ELSE 'more than 15000'
            END AS odo_cat
    FROM enriched

), ranked AS(

  SELECT *
      , COUNT(id) OVER (PARTITION BY manufacturer, model, year_cat, odo_cat) AS cnt
      , row_number() OVER (PARTITION BY manufacturer, model, year_cat, odo_cat ORDER
    FROM cated

), percented AS(

  SELECT id
      , manufacturer
      , model
      , type
      , size
      , years_old
      , odometer
      , price
      , year_cat
      , odo_cat
      , cnt
      , rn
    FROM ranked
  WHERE CAST(ranked.rn AS FLOAT) / CAST(cnt AS FLOAT) BETWEEN 0.05 AND 0.95

), maxed AS(

  SELECT manufacturer
      , model
      , year_cat
      , odo_cat
      , MAX(price) AS price_guide
    FROM percented
  WHERE CAST(rn AS FLOAT) / CAST(cnt AS FLOAT) <= 0.8
  GROUP BY 1,2,3,4
```

```
)
SELECT a.*
     , b.price_guide
  FROM percented AS a
  LEFT JOIN maxed AS b
    ON a.manufacturer = b.manufacturer AND a.model = b.model AND a.year_cat = b.year


''')
print(df_matrix.head(10))
```

Save data into csv for further analysis in PowerBI

In [2]:
```
df_matrix.to_csv('cleaned_PowerBI.csv')
```

In [ ]:
```
df = psql.read_sql('SELECT * from us_carsales_v1', con)
# Copy df to df_popular with column budget guide added
df_popular = sqldf('''

WITH ordered AS(

   SELECT manufacturer
        , model
        , type
        , size
        , years_old
        , cylinders
        , odometer
        , price
        , COUNT(id) OVER (PARTITION BY manufacturer,model,years_old) AS cnt
        , row_number() OVER (PARTITION BY manufacturer,model,years_old ORDER BY pric
    FROM df
    WHERE ((manufacturer = 'ford' AND model = 'f-150')
       OR (manufacturer = 'chevrolet' AND model = 'silverado 1500')
       OR (manufacturer = 'ram' AND model = '1500')
       OR (manufacturer = 'toyota' AND model = 'camry')
       OR (manufacturer = 'chevrolet' AND model = 'silverado')
       OR (manufacturer = 'toyota' AND model = 'tacoma')
       OR (manufacturer = 'ford' AND model = 'escape')
       OR (manufacturer = 'honda' AND model = 'accord')
       OR (manufacturer = 'nissan' AND model = 'altima')
       OR (manufacturer = 'jeep' AND model = 'grand cherokee'))
      AND price > '1000'
      AND price < '100000'
      AND year > '1990'
      AND years_old > '0'

), maxed AS(

   SELECT manufacturer
        , model
        , years_old
        , MAX(price) AS budget_guide
    FROM ordered
    WHERE CAST(rn AS FLOAT)/CAST(cnt AS FLOAT) <= 0.8
       OR cnt = 1
    GROUP BY 1,2,3

)
SELECT a.manufacturer
     , a.model
     , a.type
     , a.size
```

```
      , a.years_old
      , a.cylinders
      , a.odometer
      , a.price
      , b.budget_guide
  FROM ordered AS a
  LEFT JOIN maxed AS b
    ON a.manufacturer=b.manufacturer AND a.model=b.model AND a.years_old = b.years_o

''')
```

In [ ]:
```
# Save data into csv for further analysis in PowerBI
df_popular.to_csv('df_popular.csv', index = False)
```

In [5]:
```
print("Data transform is done")
```

```
Data transform is done
```