Ciencia de Datos en Python

Universidad Galileo

Maestría en Business Intelligence and Analytics

Marvin Alejandro Diaz Castillo Carné: 9516008

Hector Alfredo Vásquez Alarcón Carné: 21006402

Descripción del Proyecto

En este proyecto desarrollaremos un pipeline de Ingieneria de Datos utilizando Python, SQL y AWS para su desarrollo. ira trabajar con un dataset. El motivo del proyecto es poner en practica lo aprendido en la clase Ciencia de Datos en Python.

Objetivos generales

Realizar un ETL por medio de aws Desarrollar un pipeline de Ingieneria de Datos utilizando Python, SQL y AWS para su desarrollo.

Descripción del Proyecto

Objetivos específicos

Extracción de la información, transformación y carga de los datos por medio del lenguaje de programación de Python hacia la base de datos en la nube de Amazon Redshift.

Modelo de datos

El modelo de datos a utilizar es un dataset de ventas denominado "sales data".

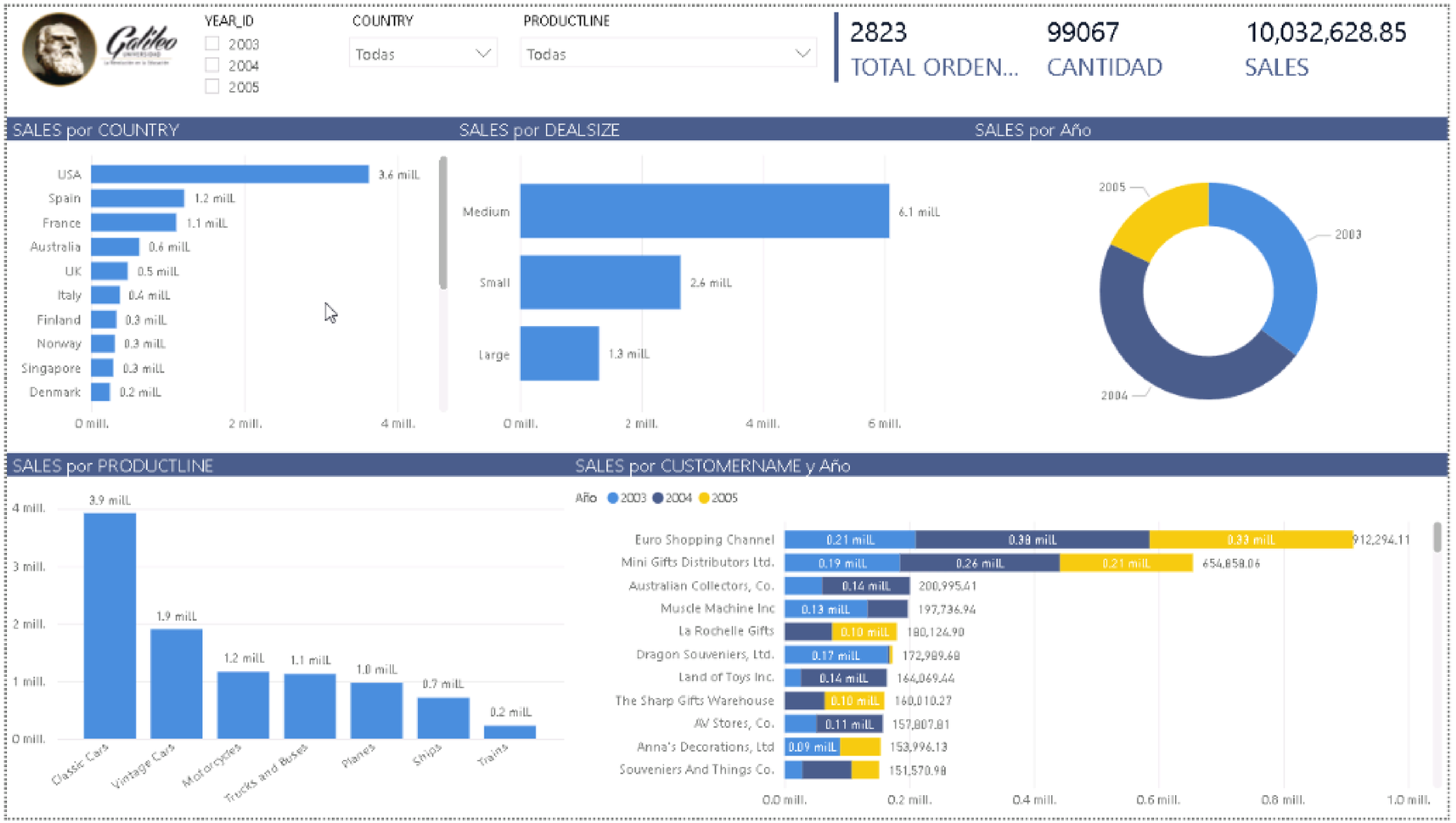
Este dataset esta disponible en Kaggle, por medio del siguiente link: https://www.kaggle.com/datasets/ankitchahal1/sales-data

Análisis Exploratorio

La serie seleccionada es estructurada, consta de 25 columnas y 2823 filas.

El dataset cuenta con información de ventas realizadas el cual contiene variables de país, clientes, línea de productos, distribuidores como las variables mas importantes, las cuales se determinaron las siguientes preguntas:

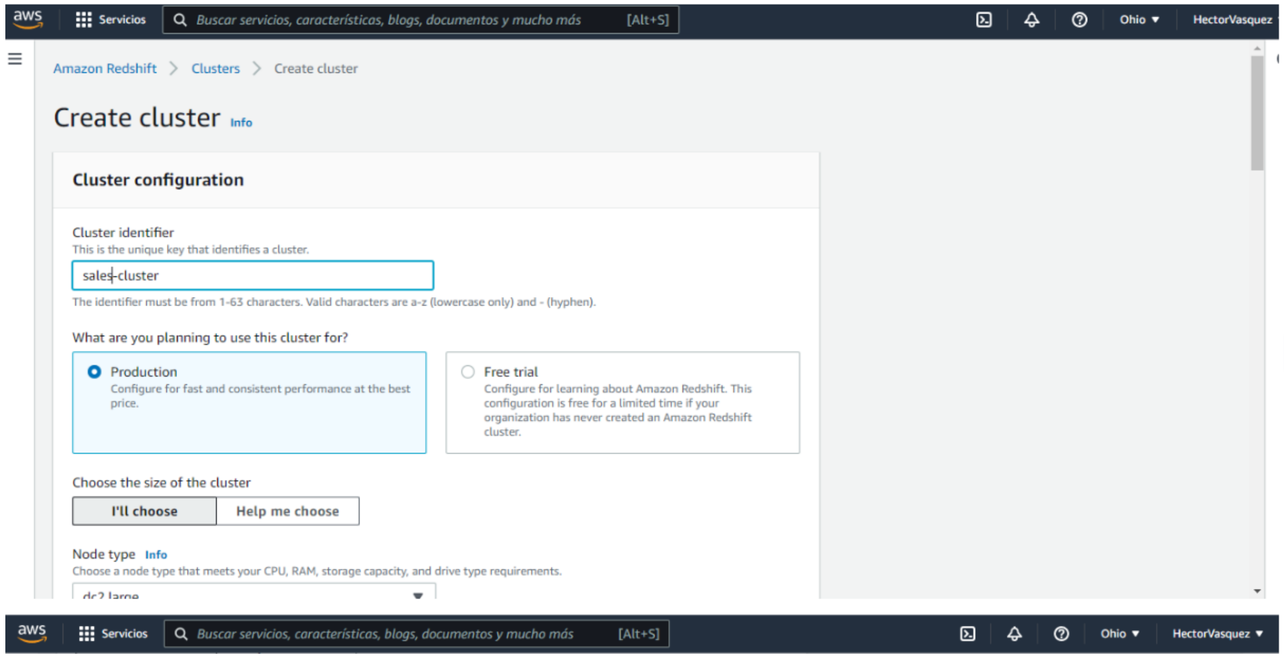
1. ¿Cual es la linea de producto con mayores ventas?
2. ¿Que pais vende mas?
3. ¿Cual es el año con mayores ventas?
4. Ventas por tamaño de Dealer
5. ¿Que Customer tiene mayor compras por año?



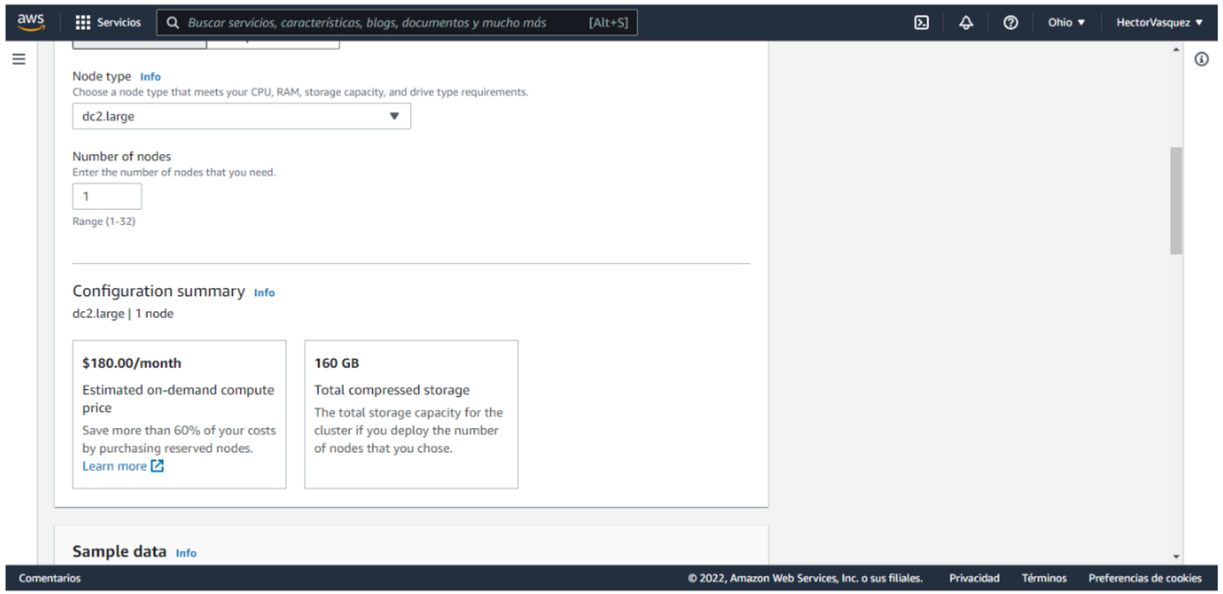
Creación de Cluster en Redshift

1) Para la ejecución del modelo dimensional, fue realizado un cluster en aws Redshift.

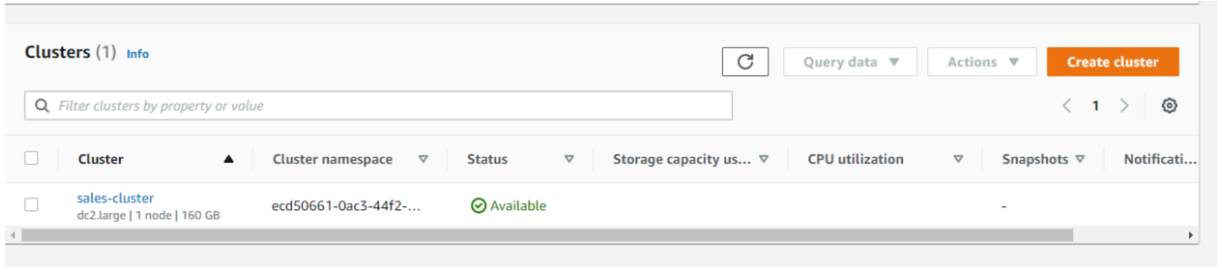
Creamos el cluster el cual nombramos sales-cluster.



2) Configuración de 1 nodo el cual será suficiente para este proyecto.

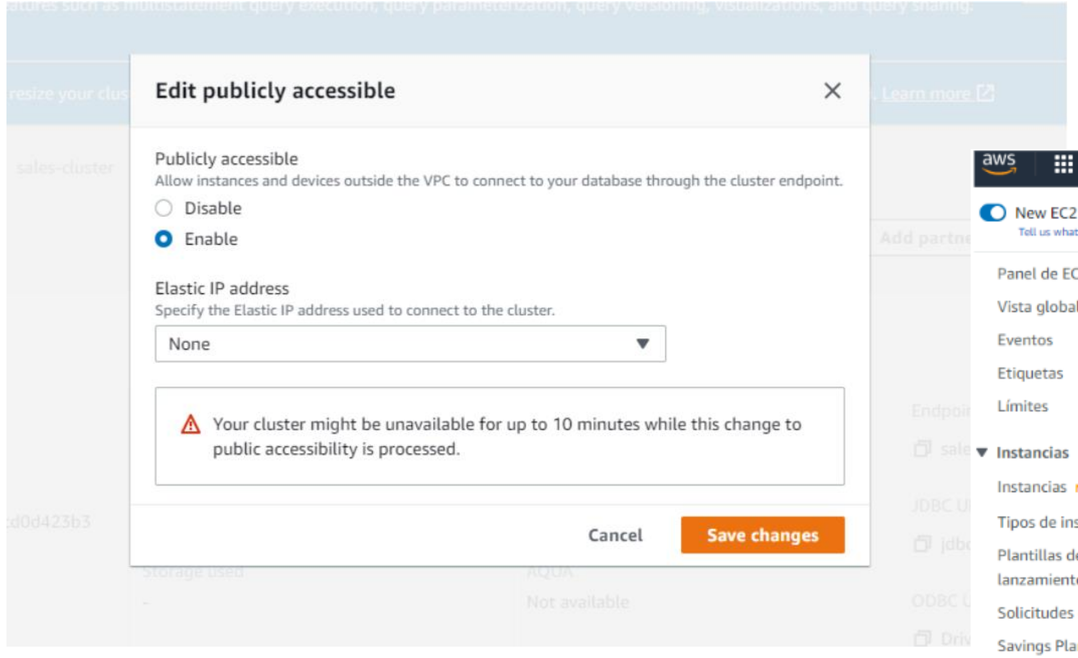


3) Cluster Creado.

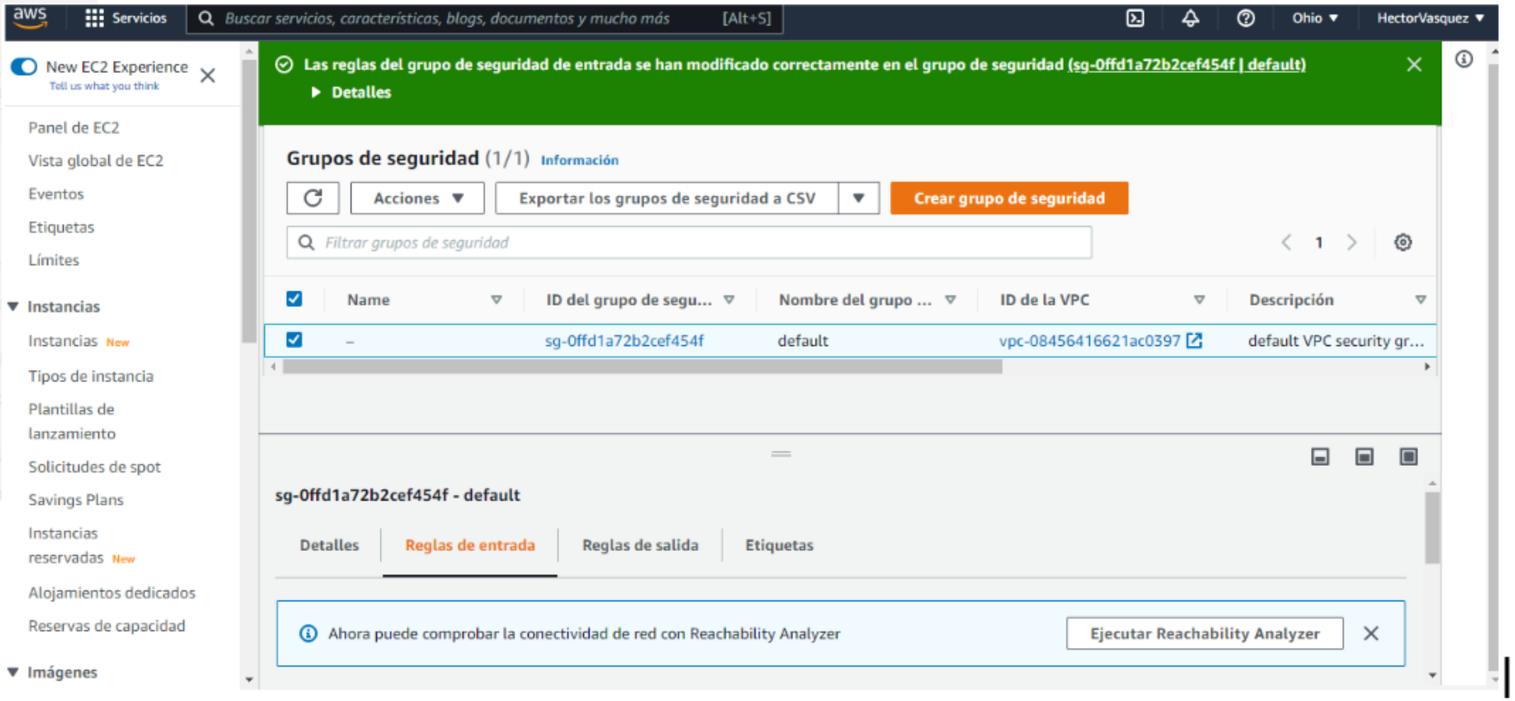


Creación de Cluster en Redshift

Editamos el permiso para poder acceder al cluster.



Agregamos las reglas de entrada para la configuración del puerto y la ip. por el cual accederemos.



Extracción, Transoformación y Carga

Carga de dataset en Python

Realizamos la carga de nuestro dataset de manera local, en el cual presamos las columnas que se encuentran en el dataset.

```
In [1]: import pandas as pd
        import numpy as np
```

```
In [183]: archivo = pd.read_csv("C:/Users/hvasquez/Downloads/sales_data_sample.csv", sep=';')
          archivo.head()
```

Out[183]:

S	ORDERDATE	STATUS	QTR_ID	MONTH_ID	YEAR_ID	...	ADDRESSLINE1	ADDRESSLINE2	CITY	STATE	POSTALCODE	COUNTRY	TERRITORY	CONTACTLASTI
0	2/24/2003 0:00	Shipped	1	2	2003	...	897 Long Airport Avenue	NaN	NYC	NY	10022	USA	NaN	
0	5/07/2003 00:00	Shipped	2	5	2003	...	59 rue de l'Abbaye	NaN	Reims	NaN	51100	France	EMEA	H
4	7/01/2003 00:00	Shipped	3	7	2003	...	27 rue du Colonel Pierre Avia	NaN	Paris	NaN	75508	France	EMEA	Da C
0	8/25/2003 0:00	Shipped	3	8	2003	...	78934 Hillside Dr.	NaN	Pasadena	CA	90003	USA	NaN	
7	10/10/2003 00:00	Shipped	4	10	2003	...	7734 Strong St.	NaN	San Francisco	CA	NaN	USA	NaN	

```
In [90]: archivo.columns
```

```
Out[90]: Index(['ORDERNUMBER', 'QUANTITYORDERED', 'PRICEEACH', 'ORDERLINENUMBER',
                'SALES', 'ORDERDATE', 'STATUS', 'QTR_ID', 'MONTH_ID', 'YEAR_ID',
                'PRODUCTLINE', 'MSRP', 'PRODUCTCODE', 'CUSTOMERNAME', 'PHONE',
                'ADDRESSLINE1', 'ADDRESSLINE2', 'CITY', 'STATE', 'POSTALCODE',
                'COUNTRY', 'TERRITORY', 'CONTACTLASTNAME', 'CONTACTFIRSTNAME',
                'DEALSIZE'],
               dtype='object')
```

El dataset cuenta con 25 columnas y 2823 registros.

```
In [256]: archivo.info()
```

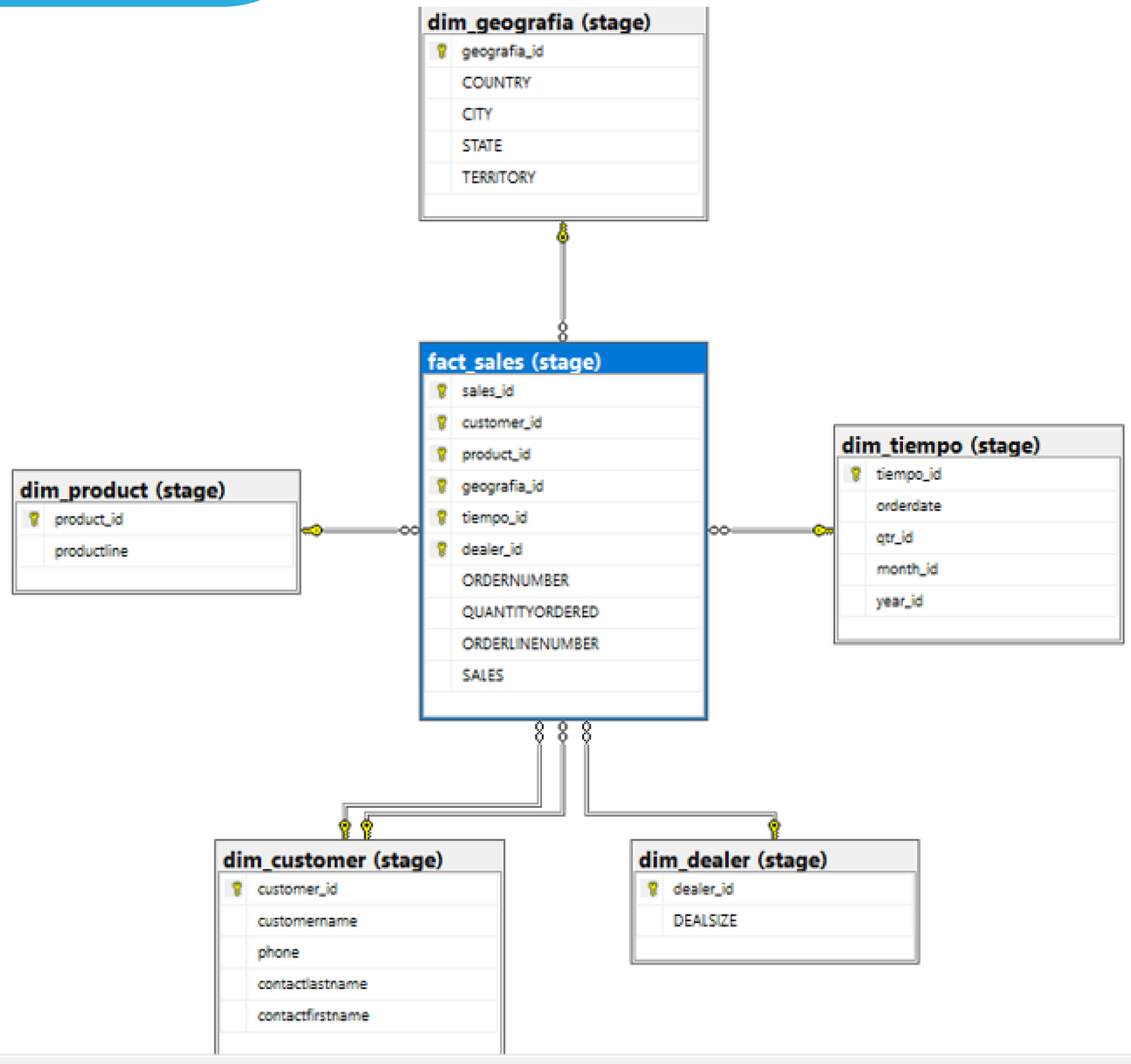
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2823 entries, 0 to 2822
Data columns (total 25 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   ORDERNUMBER       2823 non-null   int64
 1   QUANTITYORDERED   2823 non-null   int64
 2   PRICEEACH         2823 non-null   float64
 3   ORDERLINENUMBER   2823 non-null   int64
 4   SALES             2823 non-null   float64
 5   ORDERDATE         2823 non-null   object
 6   STATUS            2823 non-null   object
 7   QTR_ID            2823 non-null   int64
 8   MONTH_ID          2823 non-null   int64
 9   YEAR_ID           2823 non-null   int64
 10  PRODUCTLINE       2823 non-null   object
 11  MSRP              2823 non-null   int64
 12  PRODUCTCODE       2823 non-null   object
 13  CUSTOMERNAME      2823 non-null   object
 14  PHONE             2823 non-null   object
 15  ADDRESSLINE1      2823 non-null   object
 16  ADDRESSLINE2      302 non-null    object
 17  CITY              2823 non-null   object
 18  STATE             1337 non-null   object
 19  POSTALCODE        2747 non-null   object
 20  COUNTRY           2823 non-null   object
 21  TERRITORY         1749 non-null   object
 22  CONTACTLASTNAME   2823 non-null   object
 23  CONTACTFIRSTNAME  2823 non-null   object
 24  DEALSIZE          2823 non-null   object
dtypes: float64(2), int64(7), object(16)
memory usage: 551.5+ KB
```

Para la creación del modelo dimensional se definieron las siguientes dimensiones:

- Geografía.
- Producto.
- Cliente.
- Distribuidora.
- Tiempo

La tabla de hechos se agregaron las llaves para relacionar a las tablas de dimension y para realizar metricas se definieron las siguientes variables:

- Ventas
- Cantidad
- Total de ordenes



- **Dimensión Customer**

```
In [244]: customer = archivo[['CUSTOMERNAME','PHONE','CONTACTFIRSTNAME','CONTACTLASTNAME']]
          dim_customer = customer.drop_duplicates()
          dim_customer.insert(loc=0, column='customer_id', value = np.arange(1,len(dim_customer)+1))
          dim_customer
```

Out[244]:

	customer_id	CUSTOMERNAME	PHONE	CONTACTFIRSTNAME	CONTACTLASTNAME
0	1	Land of Toys Inc.	2125557818	Kwai	Yu
1	2	Reims Collectables	26.47.1555	Paul	Henriot
2	3	Lyon Souveniers	+33 1 46 62 7555	Daniel	Da Cunha
3	4	Toys4GrownUps.com	6265557265	Julie	Young
4	5	Corporate Gift Ideas Co.	6505551386	Julie	Brown
...	...	...	...	...	...
483	88	Australian Collectables Ltd	61-9-3844-6555	Sean	Connery
554	89	Gift Ideas Corp.	2035554407	Dan	Lewis
567	90	Bavarian Collectables Imports Co.	+49 89 61 08 9555	Michael	Donnermeyer
571	91	Royale Belge	(071) 23 67 2555	Pascale	Cartrain
937	92	Auto-Moto Classics Inc.	6175558428	Leslie	Taylor

92 rows × 5 columns

- **Dimensión Geografía**

```
In [187]: geografia = archivo[['COUNTRY','CITY','STATE','TERRITORY']]
          dim_geografia = geografia.drop_duplicates()
          dim_geografia.insert(loc=0, column='geografia_id', value = np.arange(1,len(dim_geografia)+1))
```

```
In [188]: dim_geografia
```

Out[188]:

	geografia_id	COUNTRY	CITY	STATE	TERRITORY
0	1	USA	NYC	NY	NaN
1	2	France	Reims	NaN	EMEA
2	3	France	Paris	NaN	EMEA
3	4	USA	Pasadena	CA	NaN
4	5	USA	San Francisco	CA	NaN
...	...	...	...	...	...
481	71	Italy	Bergamo	NaN	EMEA
483	72	Australia	Glen Waverly	Victoria	APAC
554	73	USA	Glendale	CT	NaN
567	74	Germany	Munich	NaN	EMEA
571	75	Belgium	Charleroi	NaN	EMEA

rows × 5 columns

- **Dimensión Producto**

```
In [186]: dim_product = pd.DataFrame(archivo['PRODUCTLINE'].unique(), columns = ['PRODUCTLINE'] )
          dim_product.insert(loc=0, column='product_id', value = np.arange(1,len(dim_product)+1))
          dim_product
```

Out[186]:

	product_id	PRODUCTLINE
0	1	Motorcycles
1	2	Classic Cars
2	3	Trucks and Buses
3	4	Vintage Cars
4	5	Planes
5	6	Ships
6	7	Trains

- **Dimensión Dealer**

```
In [189]: dim_dealer = pd.DataFrame(archivo['DEALSIZE'].unique(), columns = ['DEALSIZE'] )
          dim_dealer.insert(loc=0, column='dealer_id', value = np.arange(1,len(dim_dealer)+1))
          dim_dealer
```

Out[189]:

	dealer_id	DEALSIZE
0	1	Small
1	2	Medium
2	3	Large

- **Dimensión Tiempo**

```
In [191]: date = archivo[['ORDERDATE','QTR_ID', 'MONTH_ID', 'YEAR_ID']]
          dim_tiempo = date.drop_duplicates()
          dim_tiempo.insert(loc=0, column='tiempo_id', value = np.arange(1,len(dim_tiempo)+1))
          dim_tiempo
```

Out[191]:

	tiempo_id	ORDERDATE	QTR_ID	MONTH_ID	YEAR_ID
0	1	2/24/2003 0:00	1	2	2003
1	2	5/07/2003 00:00	2	5	2003
2	3	7/01/2003 00:00	3	7	2003
3	4	8/25/2003 0:00	3	8	2003
4	5	10/10/2003 00:00	4	10	2003
...	...	...	...	...	...
2251	248	4/11/2003 00:00	2	4	2003
2306	249	8/13/2003 0:00	3	8	2003
2356	250	10/08/2003 00:00	4	10	2003
2532	251	3/28/2005 0:00	1	3	2005
2692	252	4/21/2003 0:00	2	4	2003

252 rows × 5 columns

- **Fact Table**

```
In [250]: fact_custom = dim_customer.merge(archivo[['ORDERDATE','PRODUCTLINE','COUNTRY','CITY','STATE','TERRITORY','DEALSIZE
          fact_product = dim_product.merge(fact_custom, left_on='PRODUCTLINE', right_on='PRODUCTLINE')
          fact_geo = dim_geografia.merge(fact_product, left_on=['COUNTRY','CITY',dim_geografia['STATE'].mask(pd.isnull, dim_
                                                      , right_on=['COUNTRY','CITY',fact_product['STATE'].mask(pd.isnull, fact_p
          fact_tiempo = dim_tiempo.merge(fact_geo, left_on='ORDERDATE', right_on='ORDERDATE')
          fact_dealer = dim_dealer.merge(fact_tiempo, left_on='DEALSIZE', right_on='DEALSIZE')
          fact_table = fact_dealer[['customer_id','product_id','geografia_id','tiempo_id','dealer_id','ORDERNUMBER','QUANTIT
          fact_table.insert(loc=0, column='sales_id', value = np.arange(1,len(fact_table)+1))
          fact_table
```

Out[250]:

	sales_id	customer_id	product_id	geografia_id	tiempo_id	dealer_id	ORDERNUMBER	QUANTITYORDERED	ORDERLINENUMBER	SALES
0	1	1	1	1	1	1	10107	30	2	2871.00
1	2	1	1	1	1	1	10107	29	6	2055.23
2	3	1	1	1	1	1	10107	25	3	2845.75
3	4	1	1	1	1	1	10107	20	8	1858.00
4	5	2	1	2	2	1	10121	34	5	2765.90
...	...	...	...	...	...	...	...	...	...	...
2818	2819	26	4	21	201	3	10214	50	1	9534.50
2819	2820	68	4	56	202	3	10344	45	1	7650.00
2820	2821	13	5	10	214	3	10401	85	10	7543.75
2821	2822	13	5	10	214	3	10401	77	9	7084.00
2822	2823	88	2	72	226	3	10265	49	1	8427.02

2823 rows × 10 columns

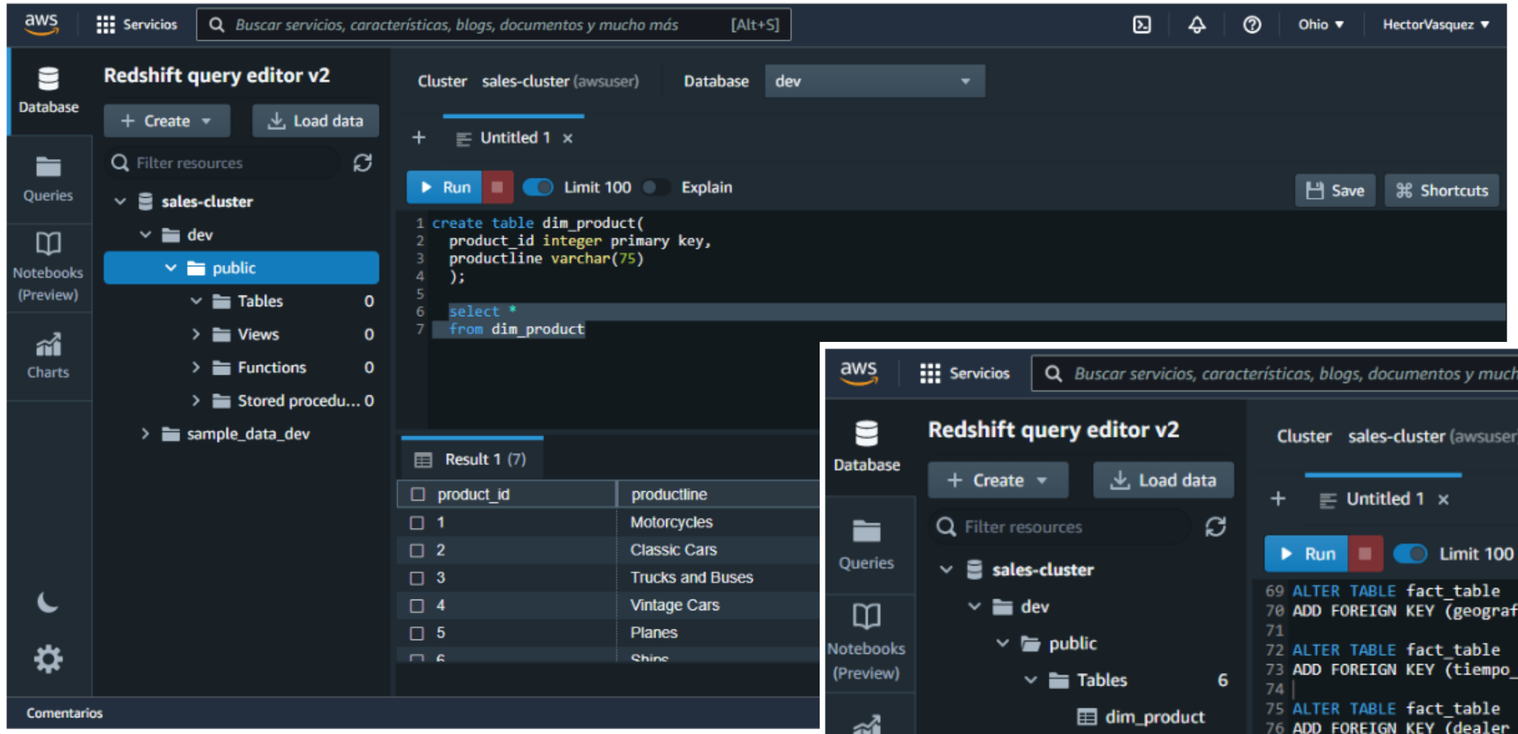
- **Cadena de conexión hacia Redshift**

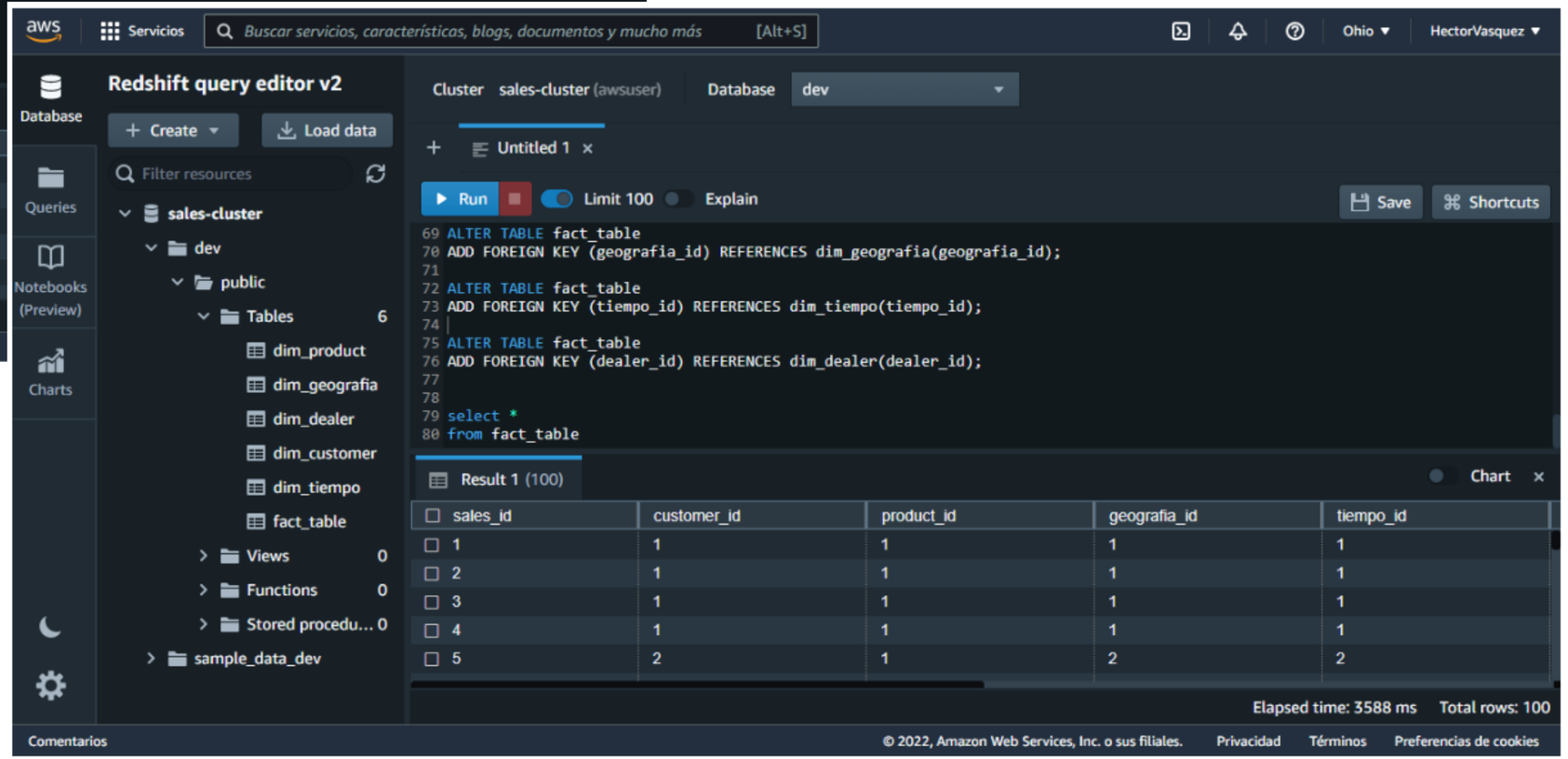
```
[REDSHIFT]
HOST = sales-cluster.ctebafejoguu.us-east-2.redshift.amazonaws.com
DB_NAME = dev
DB_USER = awsuser
DB_PASSWORD = *Redshift1$
DB_PORT = 5439
```

- **Cadena de conexión hacia Redshift**

```
In [215]: import os
          import configparser

In [217]: config = configparser.ConfigParser()
          config.read_file(open('C:/Users/hvasquez/Downloads/dwh_sales.cfg'))

In [218]: ENDPOINT = config.get('REDSHIFT','HOST')

In [220]: ENDPOINT
Out[220]: 'sales-cluster.ctebafejoguu.us-east-2.redshift.amazonaws.com'

In [221]: ENDPOINT = config.get('REDSHIFT','HOST')
          DB_USER = config.get('REDSHIFT','DB_USER')
          DB_PASSWORD = config.get('REDSHIFT','DB_PASSWORD')
          DB_PORT = config.get('REDSHIFT','DB_PORT')
          DB_NAME = config.get('REDSHIFT','DB_NAME')

In [228]: redshift_conn_string = "postgresql://{}:{}@{}:{}/{}".format(DB_USER, DB_PASSWORD, ENDPOINT, DB_PORT, DB_NAME)
          print(redshift_conn_string)

          postgresql://awsuser:*Redshift1$@sales-cluster.ctebafejoguu.us-east-2.redshift.amazonaws.com:5439/dev
```

- **Creación de Tablas en Redshift**





Script sql

- **Dimensión Cliente**

```
create table dim_customer(
  customer_id integer primary key,
  customername varchar(100),
  phone varchar(75),
  contactfirstname varchar(75),
  contactlastname varchar(75)
);
```

- **Dimension geografia**

```
create table dim_geografia(
 geografia_id integer primary key,
 country varchar(75),
 city varchar(75),
 state varchar(75),
 territory varchar(75)
 );
```

- **Dimension Producto**

```
create table dim_product(
 product_id integer primary key,
 productline varchar(75)
 );
```

- **Dimension dealer**

```
create table dim_dealer(
 dealer_id integer primary key,
 dealsize varchar(75)
 );
```

- **Dimension Tiempo**

```
create table dim_tiempo(
 tiempo_id integer primary key,
 orderdate varchar(75),
 qtr_id integer,
 month_id integer,
 year_id integer
 );
```

- **Fact Table**

```
create table fact_table(
   sales_id integer primary key,
   customer_id integer,
   product_id integer,
   geografia_id integer,
   tiempo_id integer,
   dealer_id integer,
   ordernumber integer,
   quantityordered integer,
   orderlinenumber integer,
   sales integer
   );
```

```
ALTER TABLE fact_table
ADD FOREIGN KEY (customer_id)references dim_customer(customer_id);

ALTER TABLE fact_table
ADD FOREIGN KEY (product_id) REFERENCES dim_product(product_id);

ALTER TABLE fact_table
ADD FOREIGN KEY (geografia_id) REFERENCES dim_geografia(geografia_id);

ALTER TABLE fact_table
ADD FOREIGN KEY (tiempo_id) REFERENCES dim_tiempo(tiempo_id);

ALTER TABLE fact_table
ADD FOREIGN KEY (dealer_id) REFERENCES dim_dealer(dealer_id);
```

- **Insertar datos de Pyhton hacia Redsfhift**

```
In [232]: from sqlalchemy import create_engine

In [238]: conn = create_engine(redshift_conn_string)

In [240]: #insertar dimension producto
          dim_product.to_sql('dim_product', conn, index=False, if_exists='append',method = 'multi')
Out[240]: 7

In [242]: #insertar dimension geografia
          dim_geografia.to_sql('dim_geografia', conn, index=False, if_exists='append')
Out[242]: 75

In [243]: #insertar dimension dealer
          dim_dealer.to_sql('dim_dealer', conn, index=False, if_exists='append')
Out[243]: 3

In [251]: #insertar dimension customer
          dim_customer.to_sql('dim_customer', conn, index=False, if_exists='append')
Out[251]: 92

In [253]: #insertar dimension tiempo
          dim_tiempo.to_sql('dim_tiempo', conn, index=False, if_exists='append')
Out[253]: 252

In [254]: #insertar fact table
          fact_table.to_sql('fact_table', conn, index=False, if_exists='append')
Out[254]: 2823
```