Big Data Cassandra Assignment
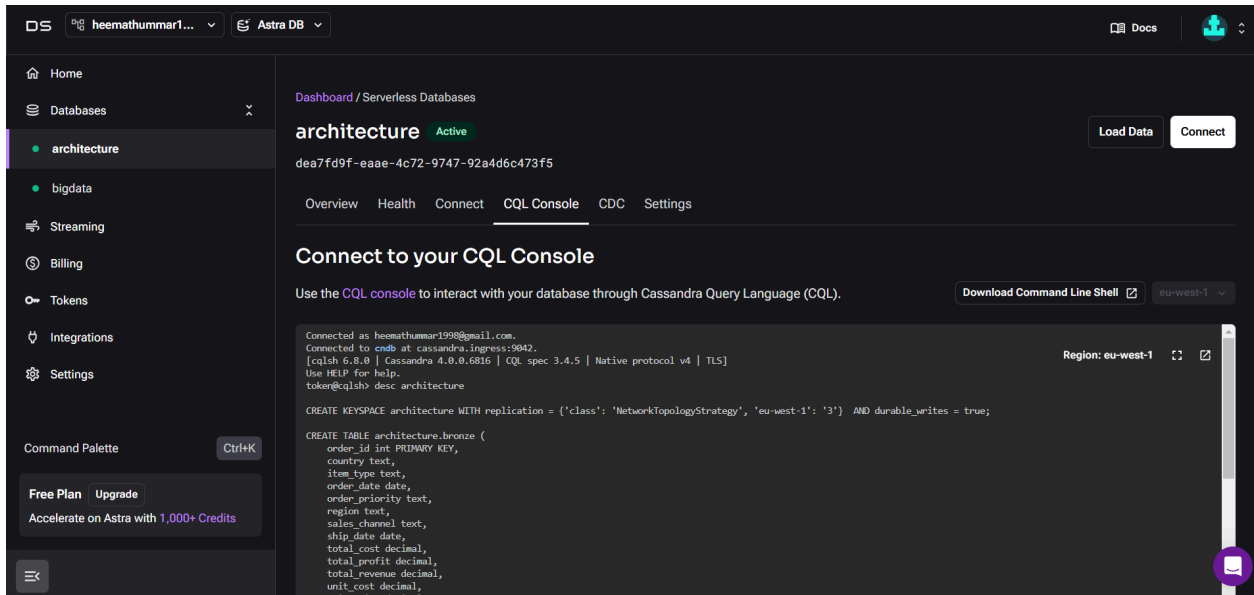
1)   DataStax Astra:-

Tables created in cql:-

CREATE KEYSPACE architecture WITH replication = {'class': 'NetworkTopologyStrategy', 'eu-west-1': '3'} AND durable_writes = true;

CREATE TABLE architecture.bronze (
    order_id int PRIMARY KEY,
    country text,
    item_type text,
    order_date date,
    order_priority text,
    region text,
    sales_channel text,
    ship_date date,
    total_cost decimal,

```
        total_profit decimal,
        total_revenue decimal,
        unit_cost decimal,
        unit_price decimal,
        units_sold int
) WITH additional_write_policy = '99PERCENTILE'
        AND bloom_filter_fp_chance = 0.01
        AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
        AND comment = ''
        AND compaction = {'class':
'org.apache.cassandra.db.compaction.UnifiedCompactionStrategy'}
        AND compression = {'chunk_length_in_kb': '64', 'class':
'org.apache.cassandra.io.compress.LZ4Compressor'}
        AND crc_check_chance = 1.0
        AND default_time_to_live = 0
        AND gc_grace_seconds = 864000
        AND max_index_interval = 2048
        AND memtable_flush_period_in_ms = 0
        AND min_index_interval = 128
        AND read_repair = 'BLOCKING'
        AND speculative_retry = '99PERCENTILE';

CREATE TABLE architecture.sales_by_priority (
        order_priority text PRIMARY KEY,
        total_cost decimal,
        total_profit decimal,
        total_revenue decimal
) WITH additional_write_policy = '99PERCENTILE'
        AND bloom_filter_fp_chance = 0.01
        AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
        AND comment = ''
        AND compaction = {'class':
'org.apache.cassandra.db.compaction.UnifiedCompactionStrategy'}
        AND compression = {'chunk_length_in_kb': '64', 'class':
'org.apache.cassandra.io.compress.LZ4Compressor'}
        AND crc_check_chance = 1.0
        AND default_time_to_live = 0
        AND gc_grace_seconds = 864000
        AND max_index_interval = 2048
        AND memtable_flush_period_in_ms = 0
```

```
    AND min_index_interval = 128
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99PERCENTILE';

CREATE TABLE architecture.sales_by_product (
    item_type text PRIMARY KEY,
    total_cost decimal,
    total_profit decimal,
    total_revenue decimal,
    total_units_sold int
) WITH additional_write_policy = '99PERCENTILE'
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class':
'org.apache.cassandra.db.compaction.UnifiedCompactionStrategy'}
    AND compression = {'chunk_length_in_kb': '64', 'class':
'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND default_time_to_live = 0
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99PERCENTILE';

CREATE TABLE architecture.sales_by_region (
    region text PRIMARY KEY,
    total_cost decimal,
    total_profit decimal,
    total_revenue decimal
) WITH additional_write_policy = '99PERCENTILE'
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class':
'org.apache.cassandra.db.compaction.UnifiedCompactionStrategy'}
    AND compression = {'chunk_length_in_kb': '64', 'class':
'org.apache.cassandra.io.compress.LZ4Compressor'}
```

```
    AND crc_check_chance = 1.0
    AND default_time_to_live = 0
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99PERCENTILE';

CREATE TABLE architecture.silver (
    order_id int PRIMARY KEY,
    country text,
    item_type text,
    order_date date,
    order_priority text,
    region text,
    sales_channel text,
    ship_date date,
    total_cost decimal,
    total_profit decimal,
    total_revenue decimal,
    unit_cost decimal,
    unit_price decimal,
    units_sold int
) WITH additional_write_policy = '99PERCENTILE'
    AND bloom_filter_fp_chance = 0.01
    AND caching = {'keys': 'ALL', 'rows_per_partition': 'NONE'}
    AND comment = ''
    AND compaction = {'class':
'org.apache.cassandra.db.compaction.UnifiedCompactionStrategy'}
    AND compression = {'chunk_length_in_kb': '64', 'class':
'org.apache.cassandra.io.compress.LZ4Compressor'}
    AND crc_check_chance = 1.0
    AND default_time_to_live = 0
    AND gc_grace_seconds = 864000
    AND max_index_interval = 2048
    AND memtable_flush_period_in_ms = 0
    AND min_index_interval = 128
    AND read_repair = 'BLOCKING'
    AND speculative_retry = '99PERCENTILE';
```

2)   Tables in Python:-

BRONZE LEVEL TABLE: -

```python
# Install the Cassandra python driver
!pip install cassandra-driver

# Import the necessary libraries
from cassandra.cluster import Cluster
from cassandra.auth import PlainTextAuthProvider
import json

# The secure connect bundle
cloud_config= {
  'secure_connect_bundle': 'secure-connect-architecture.zip'
}

with open("architecture-token.json") as f:
    secrets = json.load(f)

CLIENT_ID = secrets["clientId"]
CLIENT_SECRET = secrets["secret"]

auth_provider = PlainTextAuthProvider(CLIENT_ID, CLIENT_SECRET)
cluster = Cluster(cloud=cloud_config, auth_provider=auth_provider)
session = cluster.connect()

if session:
  print('Connected!')
else:
  print("An error occurred.")
```

```python
import json

# The secure connect bundle
cloud_config= {
    'secure_connect_bundle': 'secure-connect-architecture.zip'
}

with open("architecture-token.json") as f:
    secrets = json.load(f)

CLIENT_ID = secrets["clientId"]
CLIENT_SECRET = secrets["secret"]

auth_provider = PlainTextAuthProvider(CLIENT_ID, CLIENT_SECRET)
cluster = Cluster(cloud=cloud_config, auth_provider=auth_provider)
session = cluster.connect()

if session:
    print('Connected!')
else:
    print("An error occurred.")
```

```
Requirement already satisfied: cassandra-driver in /usr/local/lib/python3.10/dist-packages (3.29.2)
Requirement already satisfied: geomet<0.3,>=0.1 in /usr/local/lib/python3.10/dist-packages (from cassandra-driver) (0.2.1.post1)
Requirement already satisfied: click in /usr/local/lib/python3.10/dist-packages (from geomet<0.3,>=0.1->cassandra-driver) (8.1.7)
Requirement already satisfied: six in /usr/local/lib/python3.10/dist-packages (from geomet<0.3,>=0.1->cassandra-driver) (1.16.0)
WARNING:cassandra.cluster:Downgrading core protocol version from 66 to 65 for dea7fd9f-eaae-4c72-9747-92a4d6c473f5-eu-west-1.db.astra.datastax.co
WARNING:cassandra.cluster:Downgrading core protocol version from 65 to 5 for dea7fd9f-eaae-4c72-9747-92a4d6c473f5-eu-west-1.db.astra.datastax.co
WARNING:cassandra.cluster:Downgrading core protocol version from 5 to 4 for dea7fd9f-eaae-4c72-9747-92a4d6c473f5-eu-west-1.db.astra.datastax.com
Connected!
```

```python
# Create a table
session.execute("""CREATE TABLE IF NOT EXISTS architecture.bronze (
        Region TEXT,
        Country TEXT,
        Item_Type TEXT,
        Sales_Channel TEXT,
        Order_Priority TEXT,
        Order_Date DATE,
        Order_ID int PRIMARY KEY,
        Ship_Date DATE,
        Units_Sold INT,
        Unit_Price DECIMAL,
        Unit_Cost DECIMAL,
        Total_Revenue DECIMAL,
        Total_Cost DECIMAL,
        Total_Profit DECIMAL
        )
""")


import requests
import pandas as pd
```

```python
# URL of the CSV file
url = "https://raw.githubusercontent.com/gchandra10/filestorage/main/sales_100.csv"

# Download and read the CSV
response = requests.get(url)
with open("/content/sales_100.csv", "wb") as file:
    file.write(response.content)

rows = session.execute("SELECT * FROM architecture.bronze")
data = [row._asdict() for row in rows]
df = pd.DataFrame(data)

# Print the DataFrame
print(df.to_string(index=True))
```

OUTPUT:-

51	425793445	Seychelles	Beverages	2013-01-18	M	Sub-Saharan Africa	Online	2013-02-16	18978.630000000001018634065985679626464843
52	498071897	Taiwan	Cereal	2010-04-11	H	Asia	Online	2010-05-26	1100482.6699999999254919403076171β
53	368977391	Canada	Cosmetics	2011-05-09	H	North America	Online	2011-06-02	1965495.1200000001175870895385742β
54	494570004	Madagascar	Clothes	2016-10-17	L	Sub-Saharan Africa	Online	2016-10-26	111462.39999999999417923390865325927734β
55	761723172	Algeria	Cosmetics	2011-02-18	M	Middle East and North Africa	Online	2011-02-24	2546137.77000000001862645149230957031
56	757257401	Finland	Household	2016-01-15	H	Europe	Offline	2016-01-31	4094695.9199999999254919403076171β
57	363086831	Mali	Household	2010-08-19	M	Sub-Saharan Africa	Offline	2010-09-07	2169465.18000000001676380634307861328β
58	343752610	Nicaragua	Baby Food	2016-11-11	L	Central America and the Caribbean	Online	2016-11-16	460883.219999999972060322761535644531
59	584356629	Democratic Republic of the Congo	Cosmetics	2012-10-07	M	Sub-Saharan Africa	Online	2012-10-25	781300.10999999999860301613807682226β
60	752525556	Albania	Baby Food	2012-07-19	H	Europe	Offline	2012-08-13	1257823.80000000004656612873077392578β
61	862861335	The Gambia	Fruits	2011-11-20	C	Sub-Saharan Africa	Online	2011-11-22	60197.0800000000017462298274040222167968
62	314505374	Vietnam	Personal Care	2010-04-04	M	Asia	Online	2010-05-06	452453.2800000000279396772384643554β
63	791778934	Ghana	Clothes	2011-11-02	M	Sub-Saharan Africa	Online	2011-12-01	137482.23999999999068677425384521484β
64	135178029	United Kingdom	Cosmetics	2015-05-01	L	Europe	Online	2015-05-16	273336.53999999997904524207115173339843
65	180908620	Moldova	Fruits	2013-01-27	C	Europe	Online	2013-03-03	25853.1199999999998813659340143203735351566
66	887409770	Italy	Cereal	2015-11-30	C	Europe	Online	2015-12-05	366085.85999999999860301613807672226656
67	450544869	Malawi	Meat	2016-03-01	H	Sub-Saharan Africa	Offline	2016-03-12	1857730.86000000001024454832077026367118
68	156295812	East Timor	Cereal	2015-12-07	M	Australia and Oceania	Offline	2015-12-30	30331.490000000001600710675120353698730468
69	601245963	Ghana	Office Supplies	2017-03-23	L	Sub-Saharan Africa	Online	2017-04-15	470364.15999999997438862919807434082031
70	732588374	Taiwan	Fruits	2014-02-09	M	Asia	Offline	2014-02-23	55595.27999999999883584678173065185546β
71	246248090	The Bahamas	Personal Care	2011-01-19	C	Central America and the Caribbean	Online	2011-02-21	517793.78999999997904524207115173339843
72	647164094	Papua New Guinea	Clothes	2011-06-20	L	Australia and Oceania	Offline	2011-07-14	325857.2800000000279396772384643554β
73	548299157	Sao Tome and Principe	Clothes	2015-11-11	L	Sub-Saharan Africa	Offline	2015-12-28	98918.39999999999417923390865325927734β
74	153842341	Brunei	Cereal	2013-05-28	L	Asia	Offline	2013-07-16	494438.41999999998370185494422912597656
75	940995585	Papua New Guinea	Meat	2015-05-15	M	Australia and Oceania	Offline	2015-06-04	131288.39999999999417923390865325927734β
76	797385394	Italy	Office Supplies	2011-03-29	H	Europe	Online	2011-04-11	1529208.47999999998137354850769042968
77	479969346	Tunisia	Cosmetics	2012-07-01	H	Middle East and North Africa	Offline	2012-07-20	645158
78	807785928	Ethiopia	Cosmetics	2011-07-07	M	Sub-Saharan Africa	Online	2011-07-25	174324.45999999999180927421145629888281
79	500371730	Pakistan	Meat	2013-12-28	M	Middle East and North Africa	Online	2014-02-15	3635594.6099999999869614839553833007812
80	324669444	France	Cosmetics	2015-12-07	M	Europe	Online	2016-01-18	1516254.13999999998975545167922973632812
81	425418365	United Arab Emirates	Office Supplies	2012-06-23	M	Middle East and North Africa	Online	2012-07-06	5041190.8799999999888241291046142578β
82	756839835	Romania	Beverages	2012-02-19	M	Europe	Online	2012-03-13	145629.98999999999068677425384521484β
83	459386289	Kuwait	Household	2011-06-13	C	Middle East and North Africa	Offline	2011-07-21	736723.64000000001396983861923217773β
84	101328551	Solomon Islands	Household	2010-12-17	M	Australia and Oceania	Online	2010-12-28	2123231
85	642134416	Finland	Vegetables	2012-10-09	C	Europe	Offline	2012-11-27	63469.1399999999813735485076904296β
86	887124383	Greece	Cereal	2015-08-22	H	Europe	Online	2015-10-08	1015812.14000000001396983861923217773β
87	225504317	Belgium	Personal Care	2011-11-01	H	Europe	Online	2011-11-20	160206.08999999999650754034519195556640β
88	633134210	Romania	Cereal	2015-04-16	M	Europe	Online	2015-04-18	859236.06999999999487772583961486816406
89	371502530	Israel	Beverages	2013-09-08	H	Middle East and North Africa	Offline	2013-10-13	149699.10999999999854480847716331481933593
90	496523940	Ghana	Fruits	2016-05-29	M	Sub-Saharan Africa	Offline	2016-06-04	9155.15999999999998544808477163314819335593
91	174590194	Slovakia	Beverages	2016-10-26	L	Europe	Offline	2016-12-04	126301.66999999999825377017255977831203β
92	438011872	Dominica	Beverages	2012-06-12	L	Central America and the Caribbean	Online	2012-07-18	200308.7900000000081490725278854370117β
93	699160754	Rwanda	Fruits	2012-09-05	L	Sub-Saharan Africa	Online	2012-09-15	41928.27999999999883584678173065185546β
94	252889239	Thailand	Meat	2015-02-04	C	Asia	Online	2015-02-19	1605000.68999999999441206455230712890β
95	8301928887	Sri Lanka	Fruits	2011-11-07	L	Asia	Online	2011-12-18	9542.68000000000029103830456737037036130
96	925136649	Serbia	Clothes	2016-07-06	L	Europe	Offline	2016-07-13	263352.3200000000069849193096160888671β
97	824714744	Dominican Republic	Baby Food	2011-08-25	H	Central America and the Caribbean	Offline	2011-09-24	43681.0800000000017462298274040222167968
98	572335612	Vanuatu	Cereal	2014-06-20	C	Australia and Oceania	Offline	2014-08-05	665301.91000000000325962901115417480468

SILVER LEVEL TABLE:-

Cleaning the data:-

```python
session = cluster.connect()

import requests
import pandas as pd



## Find duplicates in order id as it is primary key
query = "SELECT Order_ID FROM architecture.bronze"
rows = session.execute(query)
# Convert rows to a list of Order_IDs
order_ids = list(rows)
# Check for duplicates
from collections import import Counter
```

```python
duplicates = [item for item, count in Counter(order_ids).items() if
count > 1]
# Print results
if duplicates:
    print("Duplicate Order_IDs found:", duplicates)
else:
    print("No duplicates found.")



## Find missing values
query = "SELECT * FROM architecture.bronze"
rows = session.execute(query)
# Convert rows to Pandas DataFrame
df = pd.DataFrame(rows)
# Check for null values
missing_values = df.isnull().sum()
print("Missing values per column:")
print(missing_values)
# Filter rows with any missing values
rows_with_nulls = df[df.isnull().any(axis=1)]
print("Rows with missing values:")
print(rows_with_nulls)



##Country Name Normalization
query = "SELECT * FROM architecture.bronze"
rows = session.execute(query)
# Convert rows to Pandas DataFrame
df = pd.DataFrame(rows)
# Find unique countries
unique_countries = df['country'].unique()
# Print unique countries
print("Unique Country Names:")
print(unique_countries)
```

OUTPUT:-

```
No duplicates found.
Missing values per column:
order_id          0
country           0
item_type         0
order_date        0
order_priority    0
region            0
sales_channel     0
ship_date         0
total_cost        0
total_profit      0
total_revenue     0
unit_cost         0
unit_price        0
units_sold        0
dtype: int64
Rows with missing values:
Empty DataFrame
Columns: [order_id, country, item_type, order_date, order_priority, region, sales_channel, ship_date, total_cost, total_profit, total_revenue, unit_cost, unit_price, units_sold]
Index: []
Unique Country Names:
['Vanuatu' 'Mauritius ' 'India' 'Morocco' 'Indonesia' 'The Bahamas'
 'Estonia' 'Liberia' 'Belize' 'China' 'Dominica' 'Montenegro' 'Oman'
 'Singapore' 'Nepal' 'Italy' 'United States of America' 'Switzerland'
 'Albania' 'New Zealand' 'Thailand' 'Nicaragua' 'Uganda' 'Turkmenistan'
 'Antigua and Barbuda ' 'Burundi' 'Czech Republic' 'Egypt' 'Monaco'
 'Tanzania' 'Haiti' 'Malaysia' 'Lebanon' 'Samoa ' 'Japan' 'Saint Lucia'
 'South Korea' 'Togo' 'Sweden' 'Tonga' 'Kazakhstan' 'Djibouti'
 'Netherlands' 'Iceland' 'Zimbabwe' 'South Africa' 'Seychelles ' 'Taiwan'
 'Canada' 'Madagascar' 'Algeria' 'Finland' 'Mali'
 'Democratic Republic of the Congo' 'The Gambia' 'Vietnam' 'Ghana'
 'United Kingdom' 'Moldova ' 'Malawi' 'East Timor' 'Papua New Guinea'
 'Sao Tome and Principe' 'Brunei' 'Tunisia ' 'Ethiopia' 'Pakistan'
 'France' 'United Arab Emirates' 'Romania' 'Kuwait' 'Solomon Islands'
 'Greece' 'Belgium' 'Israel' 'Slovakia ' 'Rwanda ' 'Sri Lanka' 'Serbia'
 'Dominican Republic']
```

Table after cleaning:-

```python
session.execute("""
    CREATE TABLE IF NOT EXISTS architecture.silver (
        Region TEXT,
        Country TEXT,
        Item_Type TEXT,
        Sales_Channel TEXT,
        Order_Priority TEXT,
        Order_Date DATE,
        Order_ID int PRIMARY KEY,
        Ship_Date DATE,
        Units_Sold INT,
        Unit_Price DECIMAL,
        Unit_Cost DECIMAL,
        Total_Revenue DECIMAL,
```

```python
        Total_Cost DECIMAL,
        Total_Profit DECIMAL
    )
""")
cloud_config= {
  'secure_connect_bundle': 'secure-connect-architecture.zip'
}

# This token JSON file is autogenerated when you download your token
with open("architecture-token.json") as f:
    secrets = json.load(f)

CLIENT_ID = secrets["clientId"]
CLIENT_SECRET = secrets["secret"]

auth_provider = PlainTextAuthProvider(CLIENT_ID, CLIENT_SECRET)
cluster = Cluster(cloud=cloud_config, auth_provider=auth_provider)
session = cluster.connect()

import requests
import pandas as pd

session.execute("""
    CREATE TABLE IF NOT EXISTS architecture.silver (
        Region TEXT,
        Country TEXT,
        Item_Type TEXT,
        Sales_Channel TEXT,
        Order_Priority TEXT,
        Order_Date DATE,
        Order_ID int PRIMARY KEY,
        Ship_Date DATE,
        Units_Sold INT,
        Unit_Price DECIMAL,
        Unit_Cost DECIMAL,
        Total_Revenue DECIMAL,
        Total_Cost DECIMAL,
        Total_Profit DECIMAL
    )
""")
```

```python
# Query to fetch all data from the original table
query = session.prepare("SELECT * FROM architecture.bronze")
rows = session.execute(query)
# Fetch data into a Pandas DataFrame
df = pd.DataFrame(rows)

# Insert data into the 'silver' table
for index, row in df.iterrows():
    session.execute("""
        INSERT INTO architecture.silver (Region, Country, Item_Type,
Sales_Channel, Order_Priority,
        Order_Date, Order_ID, Ship_Date, Units_Sold, Unit_Price,
Unit_Cost, Total_Revenue,
        Total_Cost, Total_Profit)
        VALUES (%s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s, %s)
    """, (row['region'], row['country'], row['item_type'],
row['sales_channel'], row['order_priority'],
        row['order_date'], row['order_id'], row['ship_date'],
row['units_sold'], row['unit_price'],
        row['unit_cost'], row['total_revenue'], row['total_cost'],
row['total_profit']))


rows = session.execute("SELECT * FROM architecture.silver")
data = [row._asdict() for row in rows]
df = pd.DataFrame(data)

# Print the DataFrame
print(df.to_string(index=True))
```

OUTPUT:-

	order_id	country	item_type	order_date	order_priority	region	sales_channel	ship_date	total_cost	total_profit	total_revenue	unit_
0	571997869	Vanuatu	Fruits	2013-11-03	C	Australia and Oceania	Online	2013-11-11	39686.2	13821.35	53507.55	
1	349235904	Mauritius	Clothes	2012-11-17	M	Sub-Saharan Africa	Offline	2012-12-02	197836.8	405388.8	603225.6	
2	440306556	India	Snacks	2012-10-10	L	Asia	Online	2012-11-20	521206.56	294943.86	816150.42	
3	667593514	Morocco	Clothes	2013-09-14	M	Middle East and North Africa	Online	2013-10-19	165258.24	338631.84	503890.08	
4	520480573	Indonesia	Household	2011-09-28	C	Asia	Online	2011-09-28	313082.42	103249.79	416332.21	50
5	488121116	The Bahamas	Fruits	2010-06-18	L	Central America and the Caribbean	Offline	2010-06-30	32800.8	11423.4	44224.2	
6	835696351	Estonia	Household	2011-09-01	H	Europe	Offline	2011-10-21	5013339.04	1653322.48	6666661.52	50
7	146634709	Liberia	Baby Food	2015-06-06	H	Sub-Saharan Africa	Offline	2015-06-12	211072.08	126918.64	337990.72	15
8	315402734	Belize	Personal Care	2015-06-14	H	Central America and the Caribbean	Online	2015-08-02	200215.11	88536.98	288752.09	
9	198927056	China	Office Supplies	2016-02-10	M	Asia	Online	2016-03-29	3040043.36	731113.75	3771157.11	52
10	274930989	Dominica	Household	2011-11-19	C	Central America and the Caribbean	Offline	2011-12-13	3539891.76	1167402.12	4707293.88	50
11	902511680	Montenegro	Clothes	2016-09-04	M	Europe	Offline	2016-09-08	75873.28	155472.48	231345.76	
12	358570849	Oman	Cosmetics	2010-11-29	H	Middle East and North Africa	Online	2010-12-28	2090050.21	1380006.19	3470056.4	26
13	176461303	Singapore	Snacks	2013-01-28	C	Asia	Online	2013-02-07	747949.44	423254.64	1171204.08	
14	179137074	Nepal	Meat	2014-02-21	H	Europe	Online	2014-03-14	3463096.24	543171.2	4006267.44	36
15	294530856	Italy	Cereal	2011-11-15	M	Europe	Online	2011-12-28	829138.8	627217.2	1456356.0	11
16	190777862	United States of America	Personal Care	2016-01-21	H	North America	Online	2016-02-19	241640.88	106855.84	348496.72	
17	830410039	Switzerland	Office Supplies	2014-07-29	C	Europe	Offline	2014-08-27	2960249.44	711923.75	3672173.19	52
18	104191863	Albania	Personal Care	2016-02-06	M	Europe	Offline	2016-02-11	87441.81	38667.58	126109.39	
19	940980136	New Zealand	Beverages	2012-10-11	M	Australia and Oceania	Online	2012-11-04	184000.52	90640.08	274640.6	
20	434008300	Thailand	Fruits	2012-09-30	C	Europe	Offline	2012-11-14	10760.6	3747.55	14508.15	
21	573998582	Nicaragua	Household	2015-10-28	C	Central America and the Caribbean	Online	2015-12-07	3915289.14	1291202.43	5206491.57	50
22	842238795	Uganda	Cosmetics	2015-02-28	M	Sub-Saharan Africa	Online	2015-03-15	1588143.23	1048609.97	2636753.2	26
23	757619178	Turkmenistan	Meat	2017-04-30	M	Asia	Online	2017-05-14	1898576.14	297783.2	2196359.34	36
24	286891067	Antigua and Barbuda	Office Supplies	2011-10-02	C	Central America and the Caribbean	Offline	2011-10-04	3305673.12	794996.25	4100669.37	52
25	529276502	Burundi	Beverages	2011-03-09	H	Sub-Saharan Africa	Online	2011-04-09	112314.07	55326.78	167640.85	
26	539471471	Uganda	Personal Care	2014-06-19	M	Sub-Saharan Africa	Online	2014-07-21	25558.17	11302.06	36860.23	
27	726137769	Czech Republic	Cosmetics	2014-03-22	H	Europe	Online	2014-04-26	2411312.81	1592127.59	4003440.4	26
28	882908809	Egypt	Clothes	2016-08-09	C	Middle East and North Africa	Online	2016-08-24	42721.28	87540.48	130261.76	
29	467399013	Monaco	Beverages	2016-05-29	M	Europe	Offline	2016-06-14	82940.11	40856.94	123797.05	
30	156530129	Tanzania	Fruits	2014-10-20	H	Sub-Saharan Africa	Online	2014-11-29	66425.08	23133.59	89558.67	
31	678896129	Indonesia	Personal Care	2012-10-22	L	Asia	Offline	2012-10-31	527371.02	233208.36	760579.38	
32	485070693	Haiti	Office Supplies	2010-12-31	C	Central America and the Caribbean	Online	2011-01-31	1077217.92	259065.0	1336282.92	52
33	955894076	Malaysia	Beverages	2014-11-13	M	Asia	Online	2014-12-28	291005.66	143351.64	434357.3	
34	659878194	Tanzania	Beverages	2016-11-30	L	Sub-Saharan Africa	Online	2017-01-16	46922.04	23114.16	70036.2	
35	704205024	Lebanon	Meat	2017-03-08	L	Middle East and North Africa	Online	2017-04-18	3198331.3	501644.0	3699975.3	36
36	116205585	Turkmenistan	Vegetables	2015-04-13	M	Asia	Online	2015-06-02	606503.1	421077.1	1027580.2	
37	937431466	Samoa	Household	2016-12-05	L	Australia and Oceania	Online	2016-12-08	2842868.78	937534.61	3780403.39	50
38	747796285	Japan	Household	2014-07-16	M	Asia	Offline	2014-07-24	4217818.22	1390971.89	5608790.11	50
39	731120240	Saint Lucia	Fruits	2015-12-12	L	Central America and the Caribbean	Offline	2016-01-12	24780.52	8630.21	33410.73	
40	297876536	South Korea	Meat	2016-03-16	L	Asia	Offline	2016-04-20	2604251.29	408465.2	3012716.49	36
41	563681733	Togo	Cosmetics	2015-09-08	M	Sub-Saharan Africa	Online	2015-09-09	1265563.98	835619.22	2101183.2	26
42	265081918	Sweden	Beverages	2012-09-13	H	Europe	Online	2012-10-12	78998.15	38915.1	117913.25	
43	839094388	Tonga	Baby Food	2016-05-11	L	Australia and Oceania	Online	2016-05-31	881752.02	530201.66	1411953.68	15
44	739008080	Tanzania	Cosmetics	2016-05-23	L	Sub-Saharan Africa	Online	2016-05-24	2045547.44	1350622.16	3396169.6	26
45	710296428	Kazakhstan	Snacks	2013-09-08	L	Asia	Online	2013-10-25	131738.88	74549.28	206288.16	
46	880811536	Djibouti	Clothes	2017-05-17	H	Sub-Saharan Africa	Online	2017-07-02	20142.08	41273.28	61415.36	

51	425793445	Seychelles	Beverages	2013-01-18	M	Sub-Saharan Africa	Online	2013-02-16	18978.63	9349.02	28327.65	
52	498071897	Taiwan	Cereal	2010-04-11	H	Asia	Online	2010-05-26	1100482.67	832480.23	1932962.9	11
53	368977391	Canada	Cosmetics	2011-05-09	H	North America	Online	2011-06-02	1965495.12	1297765.68	3263260.8	26
54	494570004	Madagascar	Clothes	2016-10-17	L	Sub-Saharan Africa	Online	2016-10-26	111462.4	228398.4	339860.8	
55	761723172	Algeria	Cosmetics	2011-02-18	M	Middle East and North Africa	Online	2011-02-24	2546137.77	1681149.03	4227286.8	26
56	757257401	Finland	Household	2016-01-15	H	Europe	Offline	2016-01-31	4094695.92	1350368.04	5445063.96	50
57	363086831	Mali	Household	2010-08-19	H	Sub-Saharan Africa	Offline	2010-09-07	2169465.18	715456.41	2884921.59	50
58	343752610	Nicaragua	Baby Food	2016-11-11	L	Central America and the Caribbean	Online	2016-11-16	460883.22	277131.26	738014.48	15
59	584356629	Democratic Republic of the Congo	Cosmetics	2012-10-07	M	Sub-Saharan Africa	Online	2012-10-25	781300.11	515872.29	1297172.4	26
60	752525556	Albania	Baby Food	2012-07-19	H	Europe	Offline	2012-08-13	1257823.8	756335.4	2014159.2	15
61	862861335	The Gambia	Fruits	2011-11-20	C	Sub-Saharan Africa	Online	2011-11-22	60197.08	20964.59	81161.67	
62	314505374	Vietnam	Personal Care	2010-04-04	M	Asia	Online	2010-05-06	452453.28	200079.04	652532.32	
63	791778934	Ghana	Clothes	2011-11-02	M	Sub-Saharan Africa	Online	2011-12-01	137482.24	281715.84	419198.08	
64	135178029	United Kingdom	Cosmetics	2015-05-01	L	Europe	Online	2015-05-16	273336.54	180477.06	453813.6	26
65	180908620	Moldova	Fruits	2013-01-27	C	Europe	Online	2013-03-03	25853.12	9003.76	34856.88	
66	887409770	Italy	Cereal	2015-11-30	C	Europe	Online	2015-12-05	366085.86	276932.34	643018.2	11
67	450544869	Malawi	Meat	2016-03-01	H	Sub-Saharan Africa	Offline	2016-03-12	1857730.86	291376.8	2149107.66	36
68	156295812	East Timor	Cereal	2015-12-07	M	Australia and Oceania	Offline	2015-12-30	30331.49	22944.81	53276.3	11
69	601245963	Ghana	Office Supplies	2017-03-23	L	Sub-Saharan Africa	Online	2017-04-15	470364.16	113120.0	583484.16	52
70	732588374	Taiwan	Fruits	2014-02-09	H	Asia	Offline	2014-02-23	55595.28	19361.94	74957.22	
71	246248090	The Bahamas	Personal Care	2011-01-19	C	Central America and the Caribbean	Online	2011-02-21	517793.79	228973.22	746767.01	
72	647164094	Papua New Guinea	Clothes	2011-06-20	L	Australia and Oceania	Offline	2011-07-14	325857.28	667716.48	993573.76	
73	548299157	Sao Tome and Principe	Clothes	2015-11-11	L	Sub-Saharan Africa	Offline	2015-12-28	98918.4	202694.4	301612.8	
74	153842341	Brunei	Cereal	2013-05-28	L	Asia	Online	2013-07-16	494438.42	374026.98	868465.4	11
75	940995585	Papua New Guinea	Meat	2015-05-15	M	Australia and Oceania	Offline	2015-06-04	131288.4	20592.0	151880.4	36
76	797385394	Italy	Office Supplies	2011-03-29	H	Europe	Online	2011-04-11	1529208.48	367766.25	1896974.73	52
77	479969346	Tunisia	Cosmetics	2012-07-01	H	Middle East and North Africa	Offline	2012-07-20	645158.5	425981.5	1071140.0	26
78	807785928	Ethiopia	Cosmetics	2011-07-07	M	Sub-Saharan Africa	Online	2011-07-25	174324.46	115101.94	289426.4	26
79	500371730	Pakistan	Meat	2013-12-28	M	Middle East and North Africa	Online	2014-02-15	3635594.61	570226.8	4205821.41	36
80	324669444	France	Cosmetics	2015-12-07	M	Europe	Online	2016-01-18	1516254.14	1001143.46	2517397.6	26
81	425418365	United Arab Emirates	Office Supplies	2012-06-23	H	Middle East and North Africa	Online	2012-07-06	5041190.88	1212378.75	6253569.63	52
82	756839835	Romania	Beverages	2012-02-19	M	Europe	Online	2012-02-23	145629.99	71738.46	217368.45	
83	459386289	Kuwait	Household	2011-06-13	C	Middle East and North Africa	Offline	2011-07-21	736723.64	242960.18	979683.82	50
84	101328551	Solomon Islands	Household	2010-12-17	M	Australia and Oceania	Online	2010-12-28	2123231.5	700209.25	2823440.75	50
85	642134416	Finland	Vegetables	2012-10-09	C	Europe	Offline	2012-11-27	63469.14	44064.74	107533.88	
86	887124383	Greece	Cereal	2015-08-22	H	Europe	Online	2015-10-08	1015812.14	768429.66	1784241.8	11
87	222504317	Belgium	Personal Care	2011-11-01	H	Europe	Online	2011-11-20	160206.09	70844.62	231050.71	
88	633134210	Romania	Cereal	2015-04-18	M	Europe	Online	2015-05-06	859236.07	649984.83	1509220.9	11
89	371502530	Israel	Beverages	2013-09-08	H	Middle East and North Africa	Online	2013-10-13	149699.11	73742.94	223442.05	
90	496523940	Ghana	Fruits	2016-05-09	M	Sub-Saharan Africa	Offline	2016-06-04	9155.16	3188.43	12343.59	
91	174590194	Slovakia	Beverages	2016-10-26	L	Europe	Online	2016-12-04	126301.67	62217.18	188518.85	
92	438011872	Dominica	Beverages	2012-06-12	L	Central America and the Caribbean	Online	2012-07-18	200308.79	98673.66	298982.45	
93	699160754	Rwanda	Fruits	2012-09-05	L	Sub-Saharan Africa	Online	2012-09-15	41928.28	14602.19	56530.47	
94	252889239	Thailand	Meat	2015-02-04	C	Asia	Online	2015-02-19	1605000.69	251737.2	1856737.89	36
95	830192887	Sri Lanka	Fruits	2011-11-07	L	Asia	Online	2011-12-18	9542.68	3323.39	12866.07	
96	925136649	Serbia	Clothes	2016-07-06	L	Europe	Offline	2016-07-13	263352.32	539637.12	802989.44	
97	824714744	Dominican Republic	Baby Food	2011-08-25	H	Central America and the Caribbean	Offline	2011-09-24	43681.08	26265.64	69946.72	15
98	572335612	Vanuatu	Cereal	2014-06-20	C	Australia and Oceania	Offline	2014-08-05	665301.91	503279.79	1168581.7	11

GOLD LEVEL TABLE:-

(i) Gold Table 1 (Show aggregated sales by region):-

```python
#Table to show sales by region
cloud_config= {
    'secure_connect_bundle': 'secure-connect-architecture.zip'
}

# This token JSON file is autogenerated when you download your token
with open("architecture-token.json") as f:
    secrets = json.load(f)

CLIENT_ID = secrets["clientId"]
CLIENT_SECRET = secrets["secret"]

auth_provider = PlainTextAuthProvider(CLIENT_ID, CLIENT_SECRET)
cluster = Cluster(cloud=cloud_config, auth_provider=auth_provider)
session = cluster.connect()

session.execute("""CREATE TABLE IF NOT EXISTS architecture.sales_by_region
(
        Region TEXT PRIMARY KEY,
        Total_Revenue DECIMAL,
        Total_Cost DECIMAL,
        Total_Profit DECIMAL
    )
    """)

import pandas as pd
from cassandra.cluster import Cluster

query = """
    SELECT Region, Total_Revenue, Total_Cost, Total_Profit
    FROM architecture.silver
"""
rows = session.execute(query)
```

```python
Load the data into a Pandas DataFrame
df = pd.DataFrame(rows)

Aggregate data by Region
aggregated_df = df.groupby('region').agg(
    Total_Revenue=('total_revenue', 'sum'),
    Total_Cost=('total_cost', 'sum'),
    Total_Profit=('total_profit', 'sum')
).reset_index()

Insert aggregated data into 'sales_by_region' table
for _, row in aggregated_df.iterrows():
    insert_query = """
        INSERT INTO architecture.sales_by_region (Region, Total_Revenue,
Total_Cost, Total_Profit)
        VALUES (%s, %s, %s, %s)
    """
    session.execute(insert_query, (row['region'], row['Total_Revenue'],
row['Total_Cost'], row['Total_Profit']))

#Print Data
rows = session.execute("SELECT * FROM architecture.sales_by_region")
data = [row._asdict() for row in rows]
df = pd.DataFrame(data)

# Print the DataFrame
print(df.to_string(index=True))
```

OUTPUT:-

	region	total_cost	total_profit	total_revenue
0	Australia and Oceania	7224318.1	3486940.03	10711258.13
1	Europe	23697468.8	11267281.03	34964749.83
2	Middle East and North Africa	18250865.54	6514261.71	24765127.25
3	Central America and the Caribbean	13318535.25	4252300.17	17570835.42
4	Asia	22090915.95	6749896.24	28840812.19
5	Sub-Saharan Africa	16573545.57	7651891.85	24225437.42
6	North America	2207136.0	1404621.52	3611757.52

(ii) Gold Table 2 (Show aggregated sales by item type):-

```python
#Table to show sales by item type
cloud_config= {
   'secure_connect_bundle': 'secure-connect-architecture.zip'
}

# This token JSON file is autogenerated when you download your token
with open("architecture-token.json") as f:
    secrets = json.load(f)

CLIENT_ID = secrets["clientId"]
CLIENT_SECRET = secrets["secret"]

auth_provider = PlainTextAuthProvider(CLIENT_ID, CLIENT_SECRET)
cluster = Cluster(cloud=cloud_config, auth_provider=auth_provider)
session = cluster.connect()

session.execute("""CREATE TABLE IF NOT EXISTS
architecture.sales_by_product (
        Item_Type TEXT PRIMARY KEY,
        Total_Units_Sold INT,
        Total_Revenue DECIMAL,
        Total_Cost DECIMAL,
        Total_Profit DECIMAL
    )
    """)



import pandas as pd
from cassandra.cluster import Cluster



query = """
    SELECT Item_Type, Units_Sold, Total_Revenue, Total_Cost, Total_Profit
    FROM architecture.silver
"""
rows = session.execute(query)
```

```
Load the data into a Pandas DataFrame
df = pd.DataFrame(rows)

 Aggregate data by Item_Type
aggregated_df = df.groupby('item_type').agg(
    Total_Units_Sold=('units_sold', 'sum'),
    Total_Revenue=('total_revenue', 'sum'),
    Total_Cost=('total_cost', 'sum'),
    Total_Profit=('total_profit', 'sum')
).reset_index()

# Step 4: Insert aggregated data into 'sales_by_product' table
for _, row in aggregated_df.iterrows():
    insert_query = """
        INSERT INTO architecture.sales_by_product (Item_Type,
Total_Units_Sold, Total_Revenue, Total_Cost, Total_Profit)
        VALUES (%s, %s, %s, %s, %s)
    """
    session.execute(insert_query, (row['item_type'],
row['Total_Units_Sold'], row['Total_Revenue'], row['Total_Cost'],
row['Total_Profit']))

#Print Data
rows = session.execute("SELECT * FROM architecture.sales_by_product")
data = [row._asdict() for row in rows]
df = pd.DataFrame(data)

# Print the DataFrame
print(df.to_string(index=True))
```

OUTPUT:-

```
        item_type   total_cost total_profit total_revenue  total_units_sold
0        Household   28966405.6    9552677.2    38519082.8             57640
1   Office Supplies  22475637.44    5405267.5   27880904.94             42814
2       Vegetables    669972.24    465141.84    1135114.08              7368
3           Snacks   1400894.88    792747.78    2193642.66             14377
4    Personal Care   2212680.15     978467.7    3191147.85             39045
5             Meat  18393869.53    2884996.4   21278865.93             50437
6           Fruits     456166.4     158867.2      615033.6             65920
7         Beverages   1437098.74    707925.96     2145024.7             45206
8           Cereal   5360827.36   4055295.84     9416123.2             45776
9         Cosmetics  17302624.31  11424476.09    28727100.4             65707
10       Baby Food   3247704.24   1952859.92    5200564.16             20372
11          Clothes   1438904.32   2948469.12    4387373.44             40148
```

(iii) Gold Table 2 (Show aggregated sales by order priority):-

```python
#Table to show sales by order priority
cloud_config= {
  'secure_connect_bundle': 'secure-connect-architecture.zip'
}

# This token JSON file is autogenerated when you download your token
with open("architecture-token.json") as f:
    secrets = json.load(f)

CLIENT_ID = secrets["clientId"]
CLIENT_SECRET = secrets["secret"]

auth_provider = PlainTextAuthProvider(CLIENT_ID, CLIENT_SECRET)
cluster = Cluster(cloud=cloud_config, auth_provider=auth_provider)
session = cluster.connect()

session.execute("""CREATE TABLE IF NOT EXISTS
architecture.sales_by_priority (
        Order_Priority TEXT PRIMARY KEY,
        Total_Revenue DECIMAL,
        Total_Cost DECIMAL,
```

```python
        Total_Profit DECIMAL
    )
    """)



import pandas as pd
from cassandra.cluster import Cluster

query = """
    SELECT Order_Priority, Units_Sold, Total_Revenue, Total_Cost,
Total_Profit
    FROM architecture.silver
"""
rows = session.execute(query)

Load the data into a Pandas DataFrame
df = pd.DataFrame(rows)

Aggregate data by Order_Priority
aggregated_df = df.groupby('order_priority').agg(
    Total_Revenue=('total_revenue', 'sum'),
    Total_Cost=('total_cost', 'sum'),
    Total_Profit=('total_profit', 'sum')
).reset_index()

Insert the aggregated data into the 'sales_by_priority' table
for _, row in aggregated_df.iterrows():
    insert_query = """
        INSERT INTO architecture.sales_by_priority (Order_Priority,
Total_Revenue, Total_Cost, Total_Profit)
        VALUES (%s, %s, %s, %s)
    """
    session.execute(insert_query, (row['order_priority'],
row['Total_Revenue'], row['Total_Cost'], row['Total_Profit']))

#Print Data
rows = session.execute("SELECT * FROM architecture.sales_by_priority")
data = [row._asdict() for row in rows]
df = pd.DataFrame(data)
```

```
# Print the DataFrame
print(df.to_string(index=True))
```

OUTPUT:-

```
  order_priority    total_cost total_profit total_revenue
0              C   25044636.63   7649022.93   32693659.56
1              M   34786866.21   14607042.5   49393908.71
2              H   27728599.52  11910944.43   39639543.95
3              L   15802682.85   7160182.69   22962865.54
```