

HOMWORK 4: CREATING AN ETL PIPELINE

In this assignment, we developed the foundational skill of creating pipelines and learned about the Extract, Transform, Load Process. The focus was to integrate the data into the Oracle Cloud based Data Warehouse. This assignment taught us to configure an Apache Hop ETL platform and also addressed key aspects of data integration, dimensional lookup and fact table loading. These tasks highlight how difficult it is to manage data in the real world and how important reliable ETL procedures are to preserving data accessibility and correctness.

Key Learning Outcomes

Data Warehouse Configuration and ETL Setup: Based on the requirements mentioned by the organization we dropped our earlier database and created a new database with required tables. Three of the tables included were Product, Customer and Date dimensions. Each of these tables were populated with data using the correct format of INSERT operations.

Next, we installed the Apache Hop ETL tool and configured it to connect to the Oracle Cloud Database using 'Connection String' which establishes an environment to conduct ETL operations.

ERROR Faced: While making the connection between the database and Apache Hop Initially there was an error which indicated that the location of the Wallet was not mentioned correctly. Once the exact file location was mentioned, the connection was successful.

Dimension Management

Once the connection to the cloud was established and the tables were populated, we updated the dimension tables based on the incoming data. We updated the dimension tables in accordance with the incoming data after the cloud connection was made and the tables were filled. We set up the tables to satisfy analytical needs and preserve historical modifications.

Understood the use of various Slowly Changing Dimension types and updated various dimension table fields to match current and correct warehouse data.

Next, we created a dimension lookup/update which on the basis of the input load, updates the slowly changing dimensional data to the database.

ERROR Faced: Initially while loading the Dimension table Product and Customer using the Microsoft excel input, there was an error for unknown file type as the files were in csv format.

Those files had to be manually saved as an Excel file before using it for the input.

Another unusual thing which was spotted was the entry of a null row while loading the Dimension table. Later, I figured out that it was expected, and Apache Hop does it intentionally for Data Integrity and Consistency,

Fact Table Loading:

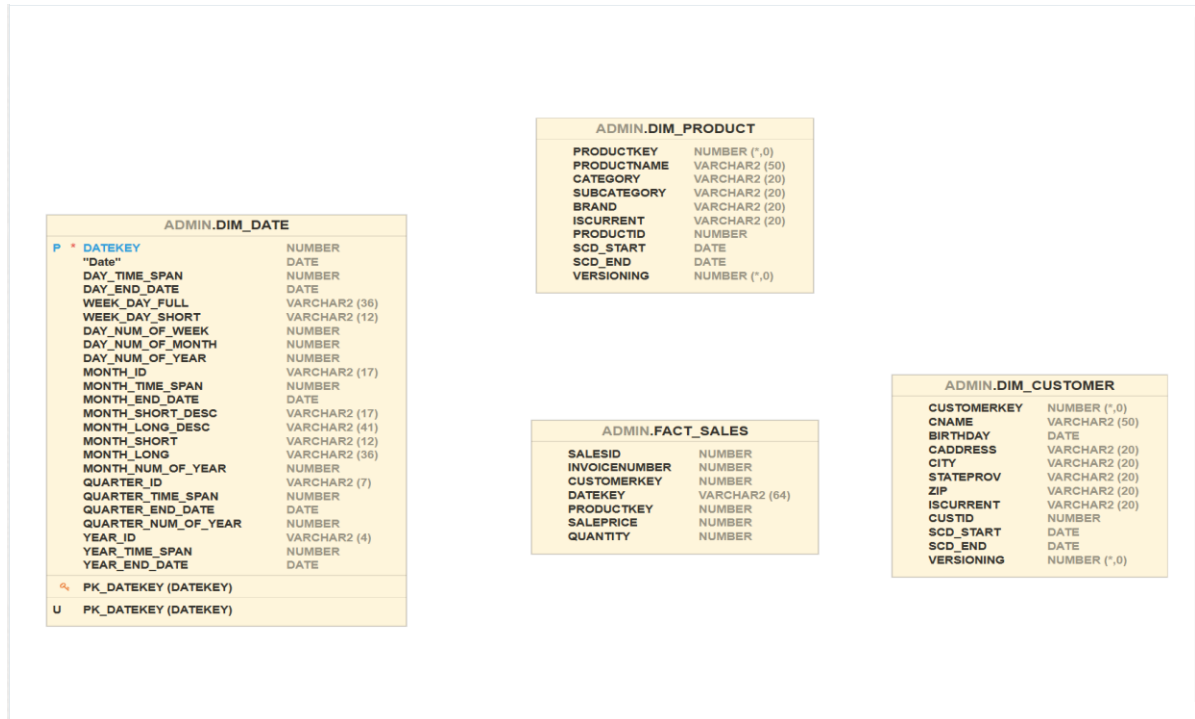
We constructed an Apache Hop ETL Pipeline to load the FACT_SALES table. We created a stream lookup to find the business key in the associated dimension tables where the records are marked as current, and return the surrogate key, which will become our foreign key in the fact

table. Then a filter was created to filter out the null values and finally the table output was used to load the data to the cloud database.

Overall, this assignment equipped us with an understanding of integrating and maintaining data pipelines. Moreover, Working with Apache Hop helped us learn ETL automation which is a critical skill in data engineering that enables us to manage and prepare complex analytical data.

APPENDIX

1. Database Diagram





The database diagram is missing the relationship between the tables. It is because no foreign key constraints have been defined between the tables. Foreign key constraints have been avoided here because it causes performance issues as the database has to work extra on CRUD operations to check foreign key consistency. Moreover, as data in the warehouses are entered in batch process there may be cases where some data arrives before the other. In this case if there is Foreign Key constraint then it may lead to the rejection of that record which may be useful. The check for referential integrity can be done in the ETL Process.

2. "SELECT * FROM Dim_Date"


[Worksheet]*									
Consumer group: LOW									
Data Load									
1 SELECT * FROM Dim_Date									
Query Result Script Output DBMS Output Explain Plan Autotrace SQL History									
Download Execution time: 0.033 seconds									
	DATEKEY	DATE	DAY_TIME_SPAN	DAY_END_DATE	WEEK_DAY_FULL	WEEK_DAY_SHORT	DAY_NUM_OF_WEE	DAY_NUM_OF_MOI	DA
1	20180101	1/1/2018, 12:00:00 AM	1	1/1/2018, 12:00:00 AM	Monday	MON	2	1	
2	20180102	1/2/2018, 12:00:00 AM	1	1/2/2018, 12:00:00 AM	Tuesday	TUE	3	2	
3	20180103	1/3/2018, 12:00:00 AM	1	1/3/2018, 12:00:00 AM	Wednesday	WED	4	3	
4	20180104	1/4/2018, 12:00:00 AM	1	1/4/2018, 12:00:00 AM	Thursday	THU	5	4	
5	20180105	1/5/2018, 12:00:00 AM	1	1/5/2018, 12:00:00 AM	Friday	FRI	6	5	
6	20180106	1/6/2018, 12:00:00 AM	1	1/6/2018, 12:00:00 AM	Saturday	SAT	7	6	
7	20180107	1/7/2018, 12:00:00 AM	1	1/7/2018, 12:00:00 AM	Sunday	SUN	1	7	
8	20180108	1/8/2018, 12:00:00 AM	1	1/8/2018, 12:00:00 AM	Monday	MON	2	8	
9	20180109	1/9/2018, 12:00:00 AM	1	1/9/2018, 12:00:00 AM	Tuesday	TUE	3	9	
10	20180110	1/10/2018, 12:00:00 AM	1	1/10/2018, 12:00:00 AM	Wednesday	WED	4	10	
11	20180111	1/11/2018, 12:00:00 AM	1	1/11/2018, 12:00:00 AM	Thursday	THU	5	11	
12	20180112	1/12/2018, 12:00:00 AM	1	1/12/2018, 12:00:00 AM	Friday	FRI	6	12	
13	20180113	1/13/2018, 12:00:00 AM	1	1/13/2018, 12:00:00 AM	Saturday	SAT	7	13	
14	20180114	1/14/2018, 12:00:00 AM	1	1/14/2018, 12:00:00 AM	Sunday	SUN	1	14	
15	20180115	1/15/2018, 12:00:00 AM	1	1/15/2018, 12:00:00 AM	Monday	MON	2	15	




3. Run "SELECT * FROM Dim_Date" again after Date Change

Start Date Changed

[Worksheet] *  Consumer group: LOW 

```
1 select * from DIM_DATE
```

Query Result Script Output DBMS Output Explain Plan Autotrace SQL History 

   Download Execution time: 0.036 seconds

	DATEKEY	DATE	DAY_TIME_SPAN	DAY_END_DATE	WEEK_DAY_FULL	WEEK_DAY_SHORT	DAY_NUM_OF_WEE	DAY_NUM_OF_MOI	DAY_N
1	20160101	1/1/2016, 12:00:00 AM	1	1/1/2016, 12:00:00 AM	Friday	FRI	6	1	
2	20160102	1/2/2016, 12:00:00 AM	1	1/2/2016, 12:00:00 AM	Saturday	SAT	7	2	
3	20160103	1/3/2016, 12:00:00 AM	1	1/3/2016, 12:00:00 AM	Sunday	SUN	1	3	
4	20160104	1/4/2016, 12:00:00 AM	1	1/4/2016, 12:00:00 AM	Monday	MON	2	4	
5	20160105	1/5/2016, 12:00:00 AM	1	1/5/2016, 12:00:00 AM	Tuesday	TUE	3	5	
6	20160106	1/6/2016, 12:00:00 AM	1	1/6/2016, 12:00:00 AM	Wednesday	WED	4	6	
7	20160107	1/7/2016, 12:00:00 AM	1	1/7/2016, 12:00:00 AM	Thursday	THU	5	7	
8	20160108	1/8/2016, 12:00:00 AM	1	1/8/2016, 12:00:00 AM	Friday	FRI	6	8	
9	20160109	1/9/2016, 12:00:00 AM	1	1/9/2016, 12:00:00 AM	Saturday	SAT	7	9	
10	20160110	1/10/2016, 12:00:00 AM	1	1/10/2016, 12:00:00 AM	Sunday	SUN	1	10	
11	20160111	1/11/2016, 12:00:00 AM	1	1/11/2016, 12:00:00 AM	Monday	MON	2	11	
12	20160112	1/12/2016, 12:00:00 AM	1	1/12/2016, 12:00:00 AM	Tuesday	TUE	3	12	
13	20160113	1/13/2016, 12:00:00 AM	1	1/13/2016, 12:00:00 AM	Wednesday	WED	4	13	
14	20160114	1/14/2016, 12:00:00 AM	1	1/14/2016, 12:00:00 AM	Thursday	THU	5	14	
15	20160115	1/15/2016, 12:00:00 AM	1	1/15/2016, 12:00:00 AM	Friday	FRI	6	15	

Updated Query:

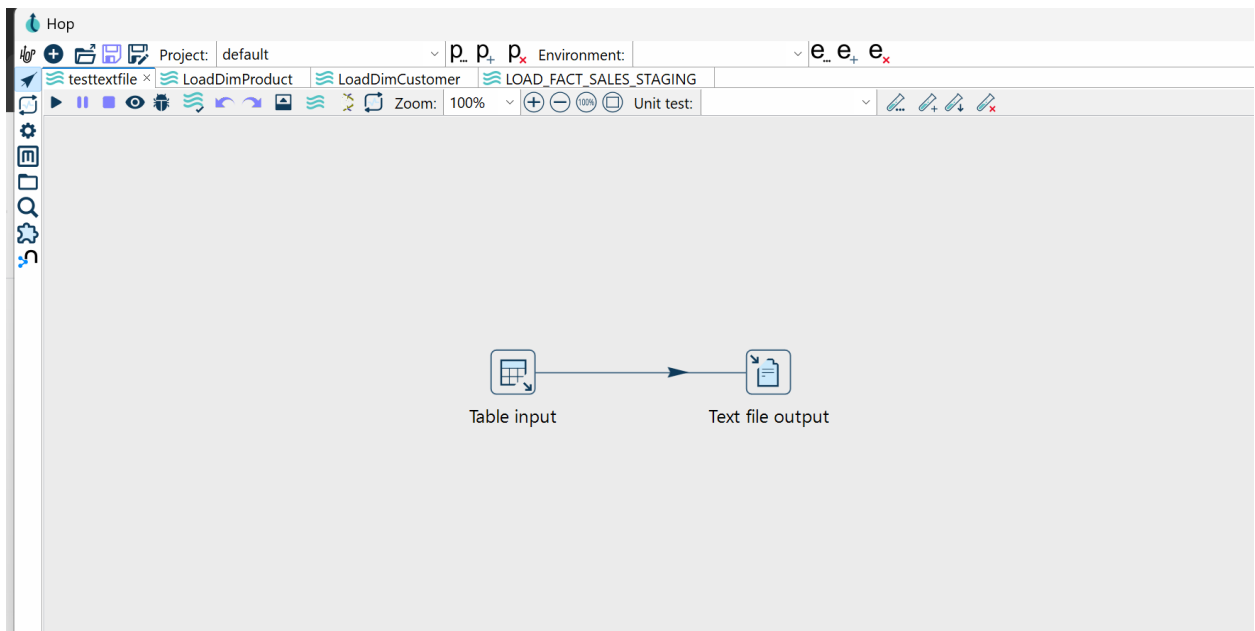
```
[Worksheet]*
2
3 CREATE TABLE DIM_DATE AS
4 SELECT
5 TO_NUMBER(TRIM(Leading '0' FROM TO_CHAR(CurrDate,'yyyymmdd'))) as DATEKEY,
6 CurrDate AS "Date",
7 1 AS Day_Time_Span,
8 CurrDate AS Day_End_Date,
9 TO_CHAR(CurrDate,'Day') AS Week_Day_Full,
10 TO_CHAR(CurrDate,'DY') AS Week_Day_Short,
11 TO_NUMBER(TRIM(Leading '0' FROM TO_CHAR(CurrDate,'D'))) AS Day_Num_of_Week,
12 TO_NUMBER(TRIM(Leading '0' FROM TO_CHAR(CurrDate,'DD'))) AS Day_Num_of_Month,
13 TO_NUMBER(TRIM(Leading '0' FROM TO_CHAR(CurrDate,'DDD'))) AS Day_Num_of_Year,
14 UPPER(TO_CHAR(CurrDate,'Mon')) || '-' || TO_CHAR(CurrDate,'YYYY') AS Month_ID,
15 -- 31 AS Month_Time_Span,
16 MAX(TO_NUMBER(TO_CHAR(CurrDate,'DD'))) OVER (PARTITION BY TO_CHAR(CurrDate,'Mon')) AS Month_Time_Span,
17 --to_date('31-JAN-2010','DD-MON-YYYY') AS Month_End_Date,
18 MAX(CurrDate) OVER (PARTITION BY TO_CHAR(CurrDate,'Mon')) as Month_End_Date,
19 TO_CHAR(CurrDate,'Mon') || '-' || TO_CHAR(CurrDate,'YYYY') AS Month_Short_Desc,
20 RTRIM(TO_CHAR(CurrDate,'Month')) || '-' || TO_CHAR(CurrDate,'YYYY') AS Month_Long_Desc,
21 TO_CHAR(CurrDate,'Mon') AS Month_Short,
22 TO_CHAR(CurrDate,'Month') AS Month_Long,
23 TO_NUMBER(TRIM(Leading '0' FROM TO_CHAR(CurrDate,'MM'))) AS Month_Num_of_Year,
24 'Q' || UPPER(TO_CHAR(CurrDate,'Q')) || '-' || TO_CHAR(CurrDate,'YYYY') AS Quarter_ID,
25 -- 31 AS Quarter_Time_Span,
26 COUNT(*) OVER (PARTITION BY TO_CHAR(CurrDate,'Q')) AS Quarter_Time_Span,
27 -- to_date('31-JAN-2010','DD-MON-YYYY') AS Quarter_End_Date,
28 MAX(CurrDate) OVER (PARTITION BY TO_CHAR(CurrDate,'Q')) AS Quarter_End_Date,
29 TO_NUMBER(TO_CHAR(CurrDate,'Q')) AS Quarter_Num_of_Year,
30 TO_CHAR(CurrDate,'YYYY') AS Year_ID,
31 --31 AS Year_Time_Span,
32 COUNT(*) OVER (PARTITION BY TO_CHAR(CurrDate,'YYYY')) AS Year_Time_Span,
33 -- to_date('31-JAN-2010','DD-MON-YYYY') AS Year_End_Date
34 MAX(CurrDate) OVER (PARTITION BY TO_CHAR(CurrDate,'YYYY')) Year_End_Date
35 FROM
36 (SELECT level n,
37 -- Calendar starts at the day after this date.
38 TO_DATE('31/12/2015','DD/MM/YYYY') + NUMTODSINTERVAL(level,'day') CurrDate
39 FROM dual
40 -- Change for the number of days to be added to the table.
41 CONNECT BY level <= 4018)
42 ORDER BY CurrDate
43 ;
44
45 ALTER TABLE DIM_DATE
46 ADD CONSTRAINT pk_datekey PRIMARY KEY (DATEKEY);
```

4. "Select * from Dim_Product"

Query Result									
Script Output DBMS Output Explain Plan Autotrace SQL History									
Download Execution time: 0.004 seconds									
	PRODUCTKEY	PRODUCTNAME	CATEGORY	SUBCATEGORY	BRAND	ISCURRENT	PRODUCTID	SCD_START	SCD_END
1	1	Cinnamon Bread	Wheat	Bread	Nothing Breader	Y	1	1/1/2024, 12:00:00 A	12/31/2099, 12:00:00 A
2	2	Milk	Dairy	Liquid	Buffalo Farms	Y	2	2/1/2024, 12:00:00 A	12/31/2099, 12:00:00 A
3	3	Chocolate Chip Cooki	Candy	Cookies	Nothing Breader	Y	3	3/1/2024, 12:00:00 A	12/31/2099, 12:00:00 A
4	4	Eggs	Dairy	Solid	Rochester Farms	Y	4	4/1/2024, 12:00:00 A	12/31/2099, 12:00:00 A
5	5	Rotini	Wheat	Pasta	Buffalo Farms	Y	5	5/1/2024, 12:00:00 A	12/31/2099, 12:00:00 A

5. Test text file copy and screenshot of the pipeline

CUSTOMERKEY,CNAME;BIRTHDAY;CADDRESS;CITY;STATEPROV;ZIP;ISCURRENT;CUSTID;SCD_START;SCD_END;VERSIONING										
1;Dominic Sellitto	;123 ABC St.	;Buffalo	;NY	;14222						;1.0;;1
2;Jeep Sellitto	;123 Cool St.	;Buffalo	;NY	;14222						;2.0;;1
3;Sally Sellitto	;415 Awesome Pl.	;Rochester	;NY	;54321						;3.0;;1



6. In Oracle Cloud, execute the query: "SELECT * FROM Dim_Product".

[Worksheet] *

Consumer group: LOW Data Load

```

1 select * from DIM_PRODUCT
    
```

Query Result
Script Output DBMS Output Explain Plan Autotrace SQL History

Download ▾ Execution time: 0.009 seconds

	PRODUCTKEY	PRODUCTNAME	CATEGORY	SUBCATEGORY	BRAND	ISCURRENT	PRODUCTID	SCD_START	SCD_END	VERSIONING
1	1	Cinnamon Bread Loaf	Wheat	Bread	Nothing Breader	Y	1	1/1/2024, 12:00:00 AM	12/31/2099, 12:00:00	1
2	2	Milk	Dairy	Liquid	Buffalo Farms	Y	2	2/1/2024, 12:00:00 AM	12/31/2099, 12:00:00	1
3	3	Chocolate Chip Cookie: Candy	Cookies	Cookies	Nothing Breader	Y	3	3/1/2024, 12:00:00 AM	12/31/2099, 12:00:00	1
4	4	Eggs	Dairy	Solid	Rochester Farms	N	4	4/1/2024, 12:00:00 AM	11/6/2024, 11:54:47 P	1
5	5	Rotini	Wheat	Pasta	Buffalo Farms	Y	5	5/1/2024, 12:00:00 AM	12/31/2099, 12:00:00	1
6	0	(null)	(null)	(null)	(null)	(null)	(null)	(null)	(null)	1
7	100	Eggs	Poultry	Solid	Rochester Farms	Y	4	11/6/2024, 11:54:45 P	12/31/2199, 11:59:59	2
8	101	Sugary Cereal	Wheat	Cereal	Food For You	Y	6	11/6/2024, 11:54:45 P	12/31/2199, 11:59:59	1

7. Screenshot of Apache Hop Pipeline

Run a *"SELECT * FROM Dim_Customer"*

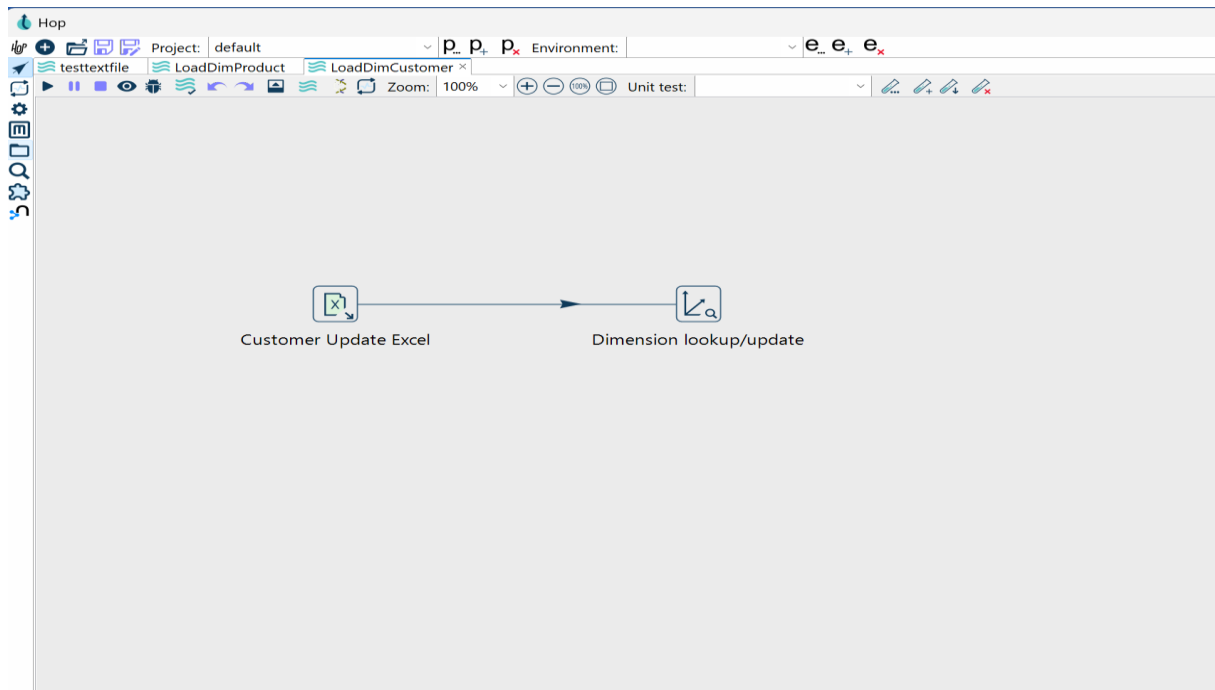


Table input

Transform name: DIM_CUSTOMER input

Connection: OCIDW

SQL: Get SQL select statement...

```
select * from ADMIN.DIM_CUSTOMER where ISCURRENT='Y'
```

Line 1 Column 1

Replace variables in script? ☐

Insert data from transform

Execute for each row? ☐

Limit size: 0

Help **OK** **Preview** **Cancel**

Stream lookup

Transform name: Stream lookup 2

Lookup transform: DIM_CUSTOMER input

The key(s) to look up the value(s):

#	Field	Lookup field
1	CustID	CUSTID

Specify the fields to retrieve :

#	Field	New name	Default	Type
1	CUSTOMERKEY			None

Preserve memory (costs CPU) ☒

Key and value are exactly one integer field ☐

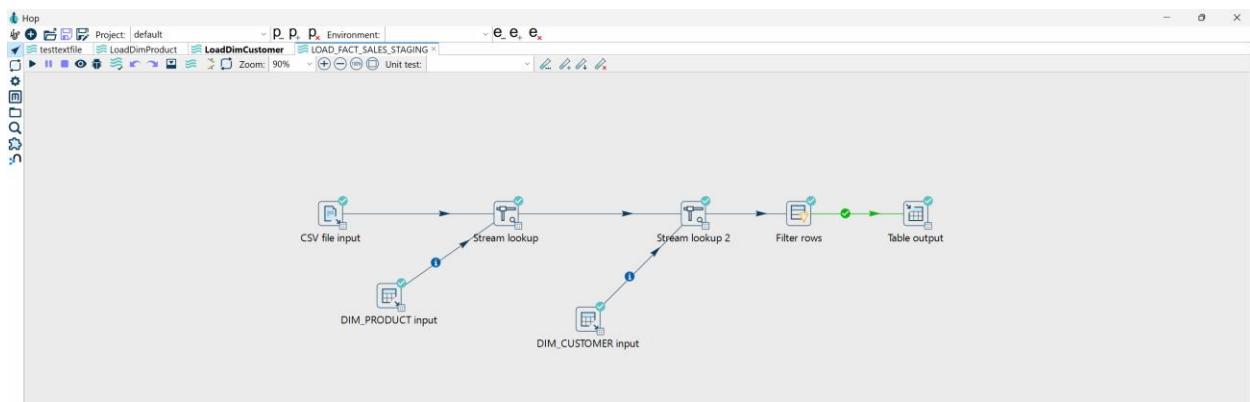
Use sorted list (i.s.o. hashtable) ☐

Help **OK** **Cancel** **Get key fields** **Get return fields**

10. Why is Date Dimension Lookup Not Needed?

A lookup for date dimension is not needed as date is already a unique identifier and there will be only one record for each date in the date dimension. The date from the input can be directly used to fetch the details from date dimension. However, the date format is different in both the CSV input file and the date dimension. This issue can be fixed by changing the date dimension input query. Additionally, while creating Visualization the tools interpret and update the date as needed.

11. Screenshot of your output from the select query you ran, a screenshot of your completed pipeline.



INTEGRATING WAREHOUSE WITH BI TOOLS

In this assignment, we learned how to connect our Oracle Data warehouse to a Business Intelligence tool and how we can use it to derive meaningful insights which can help the organization make data driven decisions to improve their sales.

Key Learning Outcomes:

1. Power BI Setup:

The initial steps included downloading and setting up Power BI Desktop. I downloaded the application from Windows Store. It was a smooth and seamless procedure. Signed in using the UB mail and everything was set up and ready to go without facing any issues.

2. Excel Connection:

Next step was to connect the Excel dataset 'NYS Employment Statistics' to power BI. The inbuild Navigator tool allowed easy connections with the excel file with just 2 clicks. Once the worksheet was connected the data along with the field names, data types etc. could be seen.

The data modeler tab shows relationships between the tables in the worksheet. For the Given worksheet there wasn't any relationship. Even the autodetect couldn't detect it.

Then, I modified the data field types where needed and created a calculated field to fix the rounding error associated with the 'Employment Numbers' Field.

3. Building Visualization:

In this step, we learnt about creating hierarchy with different fields and how we can add other fields to the created hierarchy. Also learnt that hierarchies for fields like 'date' are automatically detected and created by Power BI.

Next, we created 2 different line charts, learnt how to edit and change colors assigned to the charts and then used the forecast feature which is a built-in analytical tool to predict the future values for a given time-period.

Then Finally, the two charts were combined to form a dashboard.

4. Connecting Oracle Database to Power BI

For the connection the digital wallet was needed which was downloaded from the database connection option in Oracle. Updated the sqlnet.ora file with the correct directory to establish the connection. Next, the correct version of Oracle client was downloaded to set up the connection to Power BI.

Next, the connection was setup in Power BI by using the database credentials.

Once the connection was set up, we created the relationship Model.

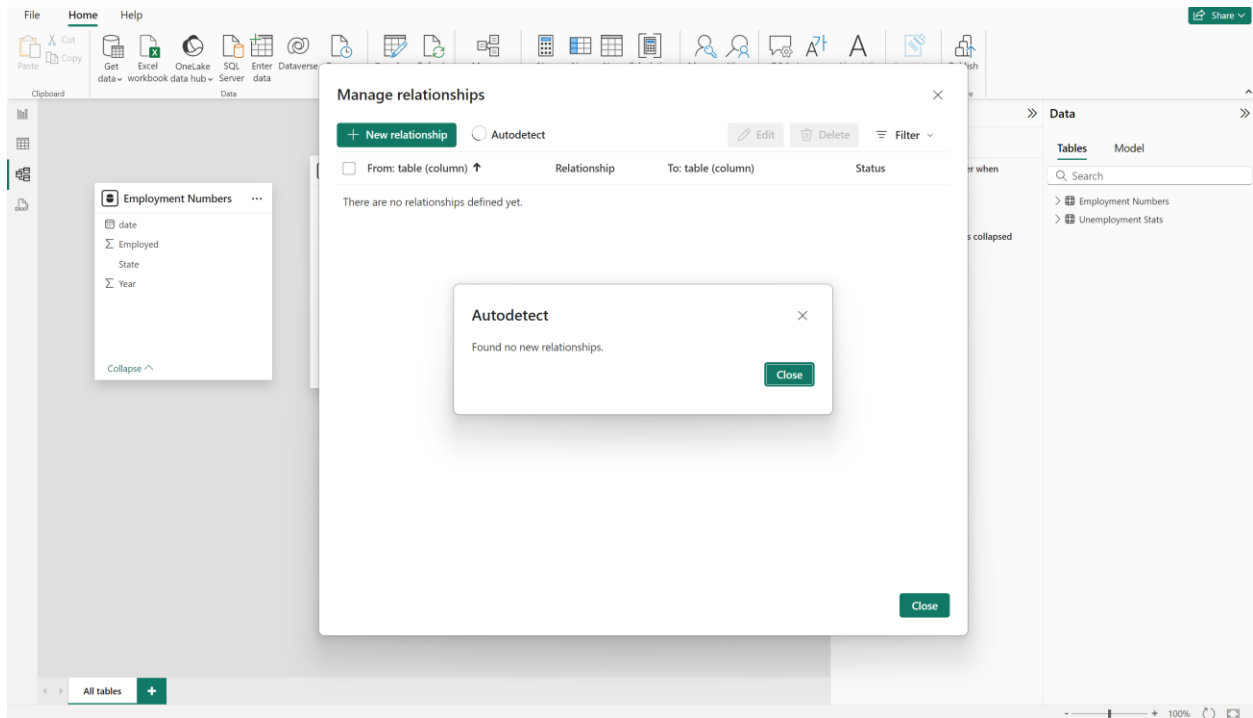
ERROR Faced: In the Relationship Model, I couldn't find a relationship between the DIM_Date and the FACT_Sales table. Even the Autodetect couldn't find a relationship. So, I manually created a relationship between the two tables on the DateKey Field.

5. Dashboard Creation:

Once the relationship model was created, the data was analyzed and understood to find insights. Then different visualizations like line chart, bar chart, pie chart etc. were used to create different visualizations. Finally, these charts were combined to form a meaningful dashboard which tells the Overall sales story to the Stakeholder.

APPENDIX:

Click the button and paste a screenshot into your report. Answer the following questions: Did autodetect find any relationships between the data? Why do you think this was the case?



No Relationship was detected by the autodetect functionality in Power BI. It was because there is no unique field that can identify each row in either of the tables. Further, the only field that is common between both the tables is State, but the State field only has one value that is 'NY' hence a relationship between both the tables does not make sense.

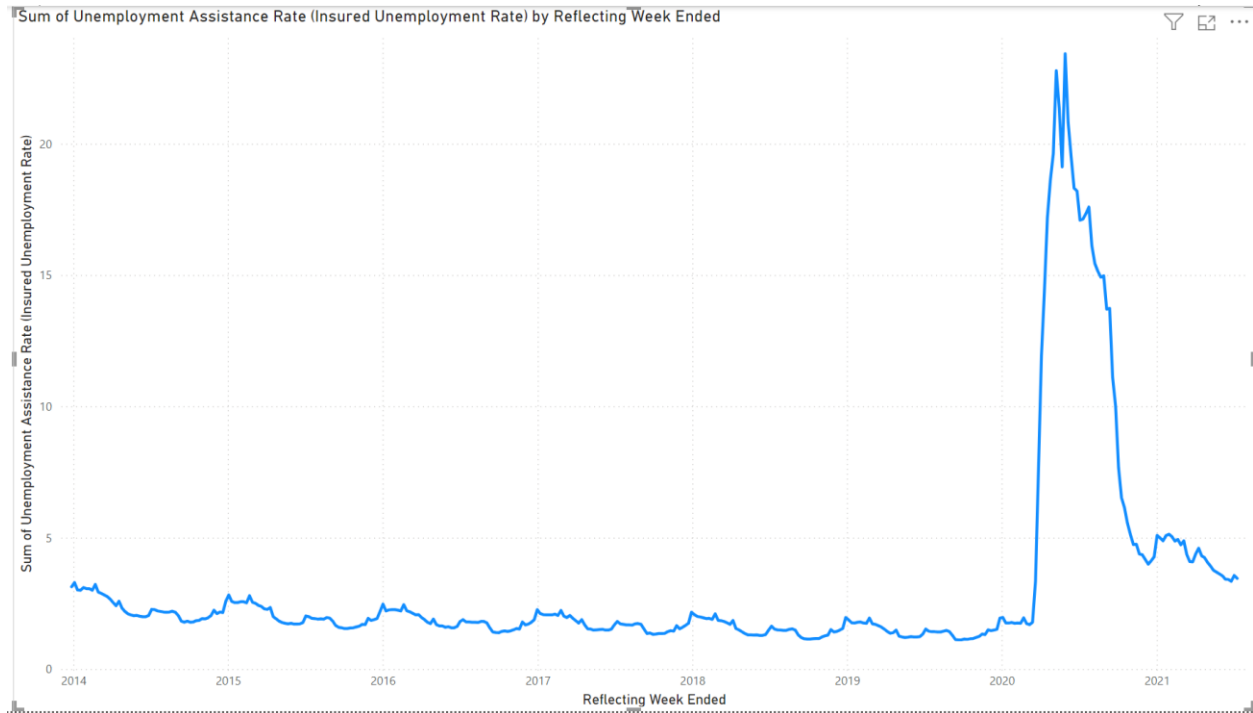
Paste a screenshot of the resulting screen in your report. Along with it, answer the following question: What was the TrueEmployed value for Saturday, February 1, 2014?

State	Year	date	Employed	TrueEmployed
NY	2021	Tuesday, June 1, 2021	8717	8717000
NY	2021	Saturday, May 1, 2021	8628	8628000
NY	2021	Thursday, April 1, 2021	8661	8661000
NY	2021	Monday, March 1, 2021	8640	8640000
NY	2021	Monday, February 1, 2021	8326	8326000
NY	2021	Friday, January 1, 2021	8322	8322000
NY	2020	Tuesday, December 1, 2020	8446	8446000
NY	2020	Sunday, November 1, 2020	8468	8468000
NY	2020	Thursday, October 1, 2020	8434	8434000
NY	2020	Tuesday, September 1, 2020	8387	8387000
NY	2020	Saturday, August 1, 2020	8381	8381000
NY	2020	Wednesday, July 1, 2020	8062	8062000
NY	2020	Monday, June 1, 2020	7955	7955000
NY	2020	Friday, May 1, 2020	7518	7518000
NY	2020	Wednesday, April 1, 2020	7373	7373000
NY	2020	Sunday, March 1, 2020	9044	9044000
NY	2020	Saturday, February 1, 2020	9130	9130000
NY	2020	Wednesday, January 1, 2020	9135	9135000
NY	2019	Sunday, December 1, 2019	9136	9136000
NY	2019	Friday, November 1, 2019	9159	9159000
NY	2019	Tuesday, October 1, 2019	9185	9185000
NY	2019	Sunday, September 1, 2019	9182	9182000
NY	2019	Thursday, August 1, 2019	9168	9168000
NY	2019	Monday, July 1, 2019	9209	9209000
NY	2019	Saturday, June 1, 2019	9178	9178000
NY	2019	Wednesday, May 1, 2019	9120	9120000
NY	2019	Monday, April 1, 2019	9119	9119000
NY	2019	Friday, March 1, 2019	9127	9127000
NY	2019	Friday, February 1, 2019	9072	9072000
NY	2019	Tuesday, January 1, 2019	9064	9064000
NY	2018	Saturday, December 1, 2018	9117	9117000

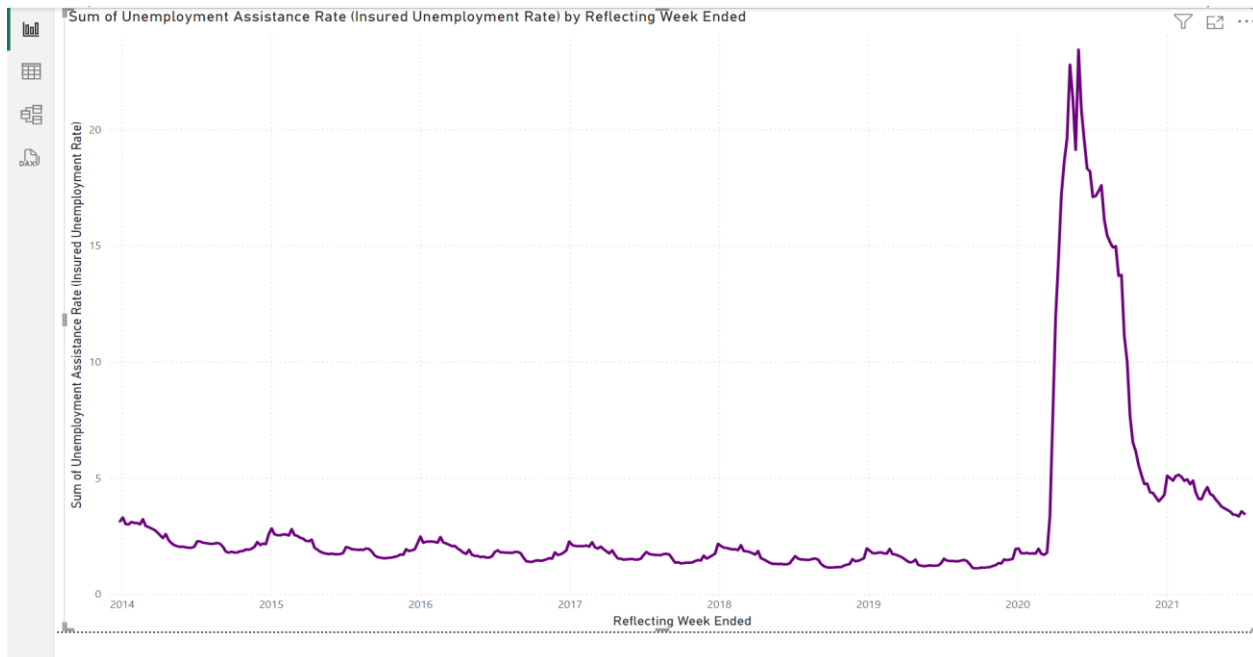
State	Year	date	Employed	TrueEmployed
NY	2014	Saturday, February 1, 2014	8818	8818000

TrueEmployed value for Saturday, February 1, 2014, is 8818000

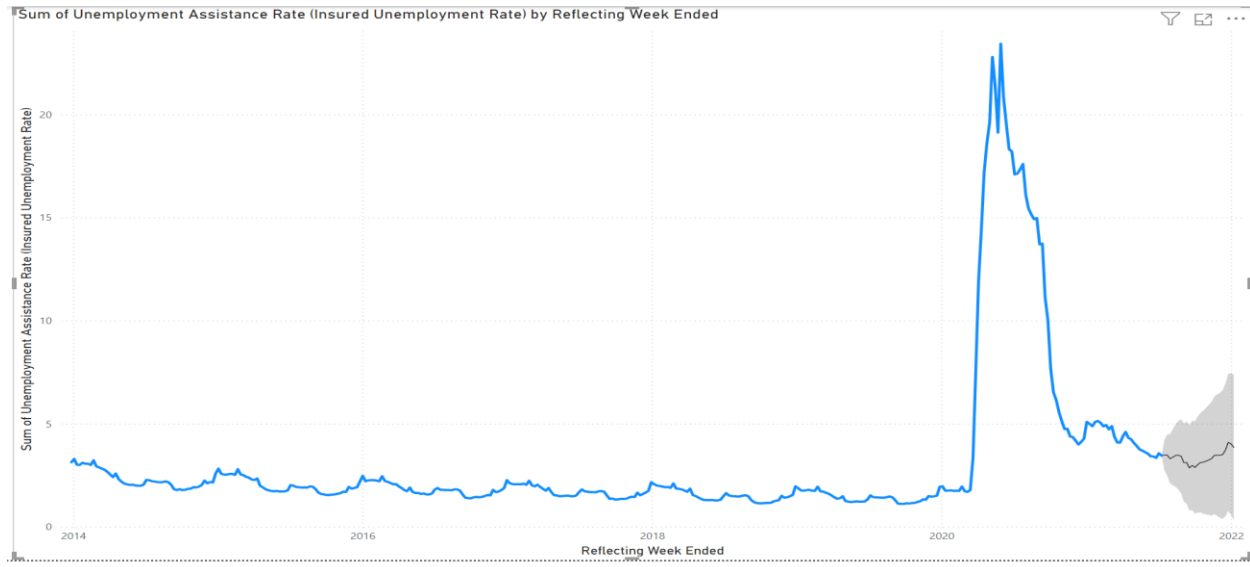
Paste a picture of the resulting line chart.



Change the color of the line chart to something other than blue. Paste a picture of the new line chart in your report.

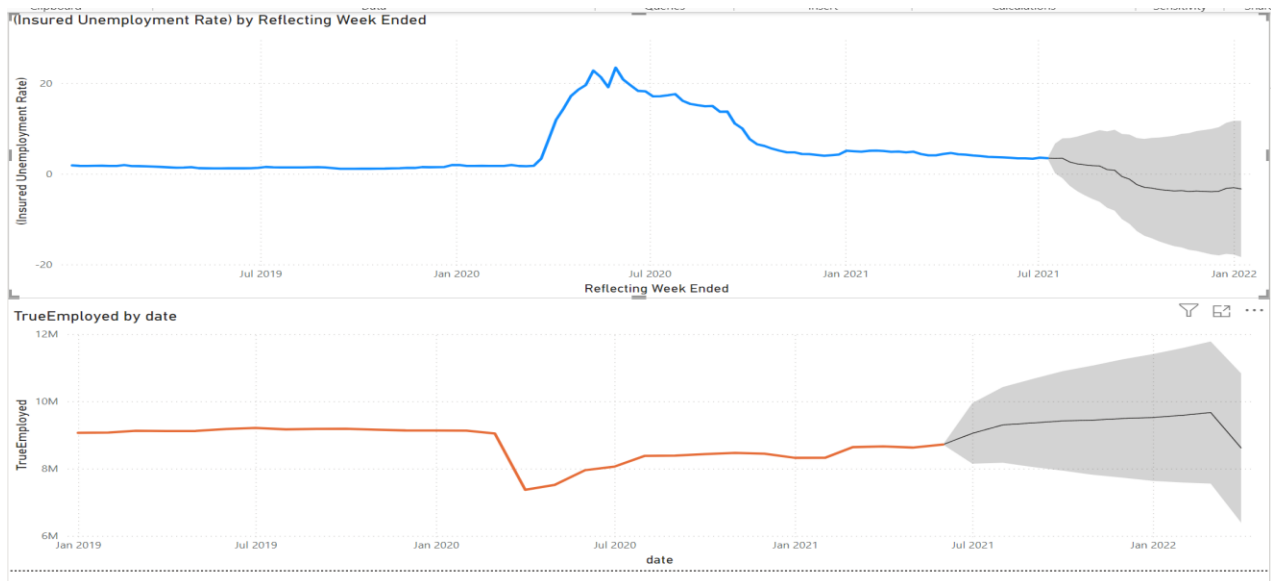


Create a forecast, setting the forecast length to 26 and the seasonality to 52 (this matches the data seasonality). Paste a picture of the line chart in your report.



Change your line chart to filter for a Relative date showing items in the last 3 calendar years.

Paste a picture of the full page (with both line charts) in your report.

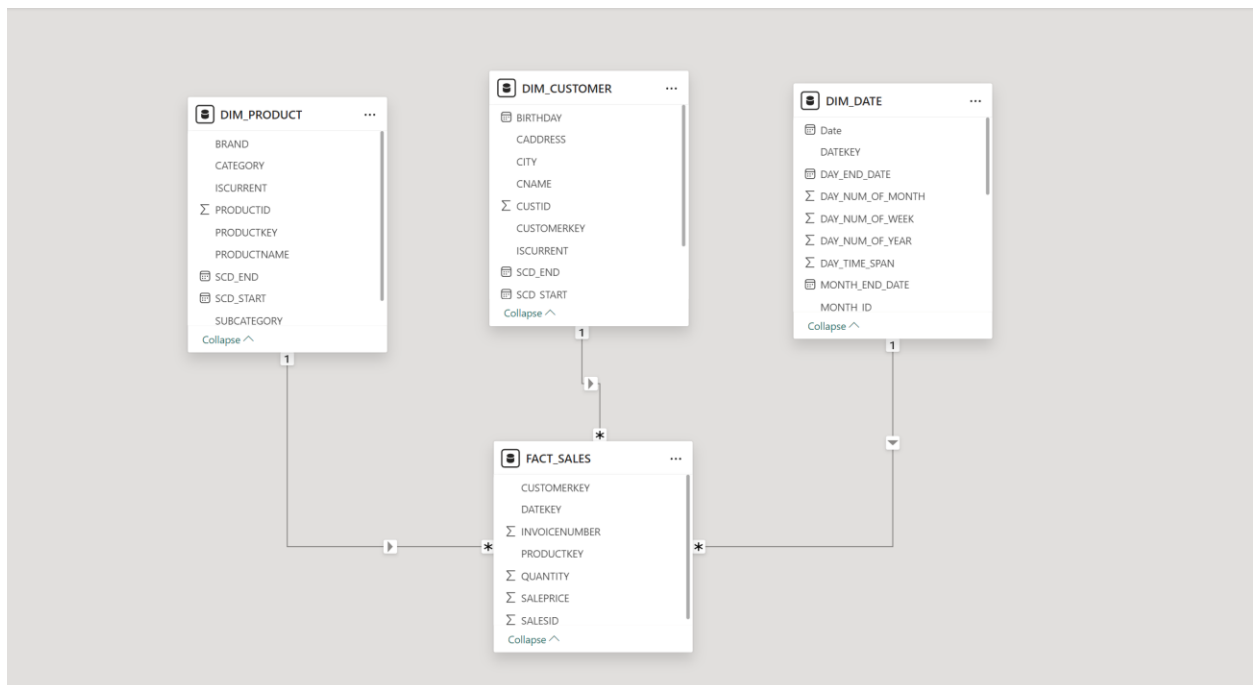


Write 2-4 sentences in your report detailing any insights you can draw from these two-line charts.

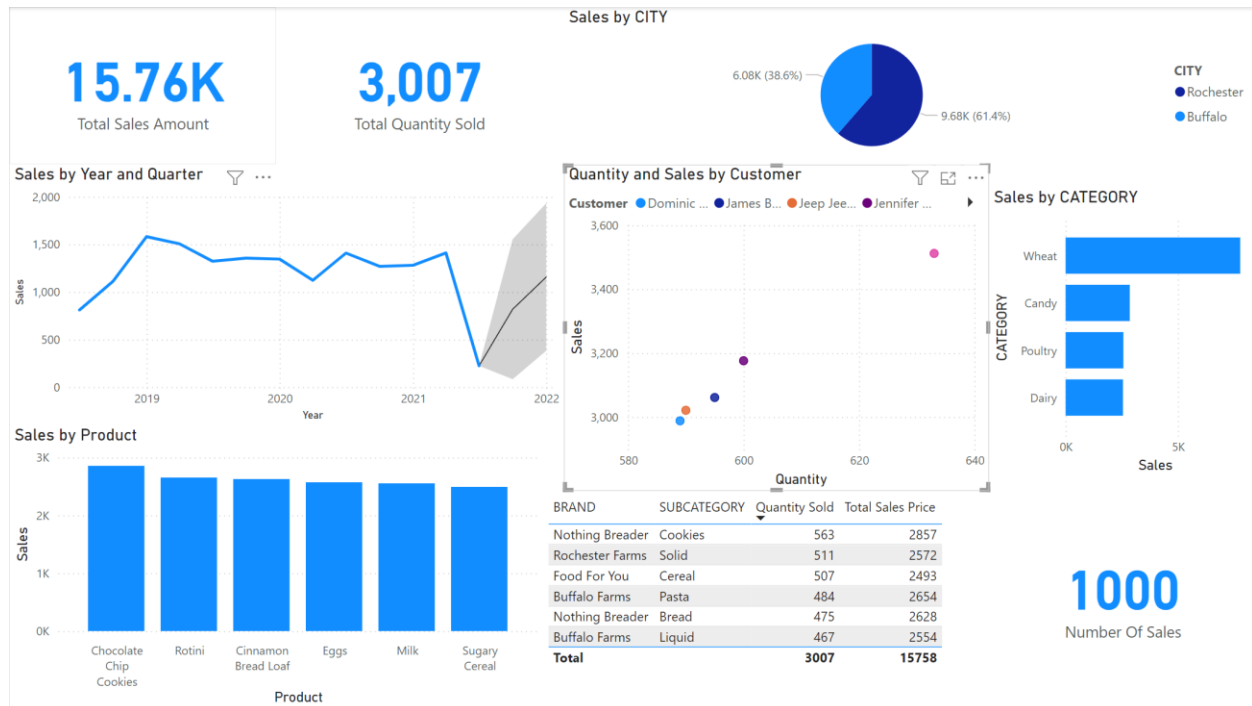
The 'TrueEmployed by Date' Chart indicates that the number of employees saw a huge dip from around 9 million to 7.3 million from March 2020 to April 2020. Which could be due to the COVID-19 Pandemic. Since this drop the number of TrueEmployed has gradually increased and is predicted to reach a higher peak of around 9.6 million in March 2022.

The second line chart shows a significant increase in the Insured Unemployment Rate from March 2020 to May 2020. The Rate has gradually decreased from 23.43 to 3.45 in July 2021 and is forecasted to decrease more in the next few months.

Take a screenshot of the resulting relationship diagram and attach it to your report.



THE GUAC STOP SALES DASHBOARD



This Dashboard gives an overall view of The Guac Stop's sales performance for a total of 1000 Sales. The Key metrics included the total sales amount of **15.76K** and a total of **3007** items sold. **The Chocolate chip cookies** from the **"Nothing Bread"** brand led the total sales with 563 units sold making a total sale of \$2857. Furthermore, the city of **Rochester** stands out as the primary sales driver, while customer **Sally Sallerson** ranks as the top customer in terms of sales and number of items bought. Overall, sales experienced a recent decline in Q2 2021, but they are projected to increase in the coming quarters.