Executive Summary

The whole objective of this assignment was to implement a data-ware house using oracle cloud and Apachehop to connect my local database to Oracle Cloud

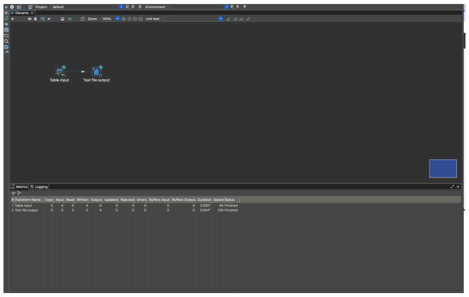
This proceeded in the following steps:

Database Creation on Oracle and Apache Hop Setup:
First I created a database in Oracle Cloud with Product_Dimension, Customer_Dimension, Date_Dimension, and FactSales tables. These tables were created using predefined scripts. Next, we populated the tables with data.

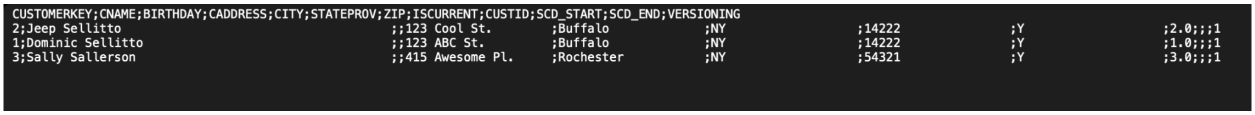
I setup Apache hop to automate data loading from local machine to the oracle data warehouse. This process involved installing javascript on the local machine, and then installing the oracle database's wallet into the java script files, furthermore, i integrated Apachehop to the oracle cloud using the wallet.
Once the connection was setup, Apachehop was launched and checked for successful data connection.

Building my First Dimensional Loader on Apache Hop:

I tested this setup by using a node on Apache hop to view data in our data warehouse. Using a table input field, i set up a connection to our data warehouse. From this connection, i was able to use SQL statements to view rows we previously added to the data warehouse. Next, I added a text file output which allows to create a text output to save the table on my local machine. I connected the text file output to the table input field using a hop, then ran the entire pipeline to call DIM_Customer to view and save on my local machine.



#	Transform Name	Copy	Input	Read	Written	Output	Updated	Rejected	Errors	Buffers Input	Buffers Output	Duration	Speed	Status
1	Table input	0	3	0	3	0	0	0	0	0	0	0.000"	48	Finished
2	Text file output	0	0	3	3	4	0	0	0	0	0	0.004"	129	Finished



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

Loading Slowly Changing Dimensions

The next task in the assignment involved loading slowly changing dimensions (SCDs) into the data warehouse. I updated the data warehouse with the `Product_Dimension` and `Customer_Dimension` tables, which included new fields.

To begin, I created an input node to read the source update file and transfer the data into the node. After verifying that all columns were correct and had the appropriate data types, the input node was connected to a Dimension Lookup node.

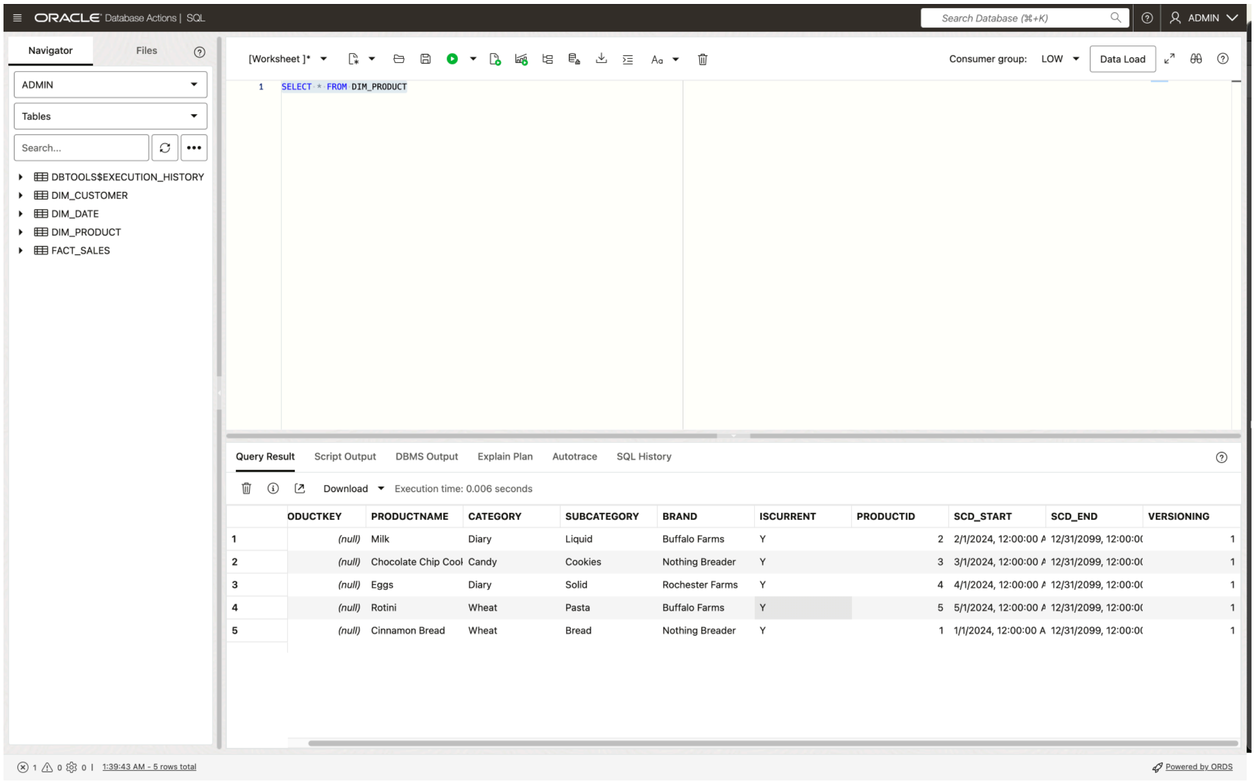
The Dimension Lookup node was configured to match the business key between the update data and the data warehouse. Additionally, we defined the type of slowly changing dimension operation to be performed for each row in the dimension table. Apache Hop provides various SCD options, such as:

- SCD Type 1 (Punch Through): Overwrites all records for the business key.
- SCD Type 2 (Insert): Adds a new row to represent the updated data.
- Update (Similar to SCD Type 1): Updates only the most recent record with the new data while leaving prior records unchanged.

A new field called `ISCURRENT` was also added to indicate whether a record is active or not.

Loading fact tables

After updating the dimension tables, I proceeded to load the Sales fact table. To accomplish this, I set up a pipeline that first verified a business key in the source sales update file. It then checked whether this record was current in the dimension tables. If the record was current, the pipeline retrieved the corresponding primary key and inserted it as a foreign key in the fact table.



```
SELECT * FROM DIM_PRODUCT
```

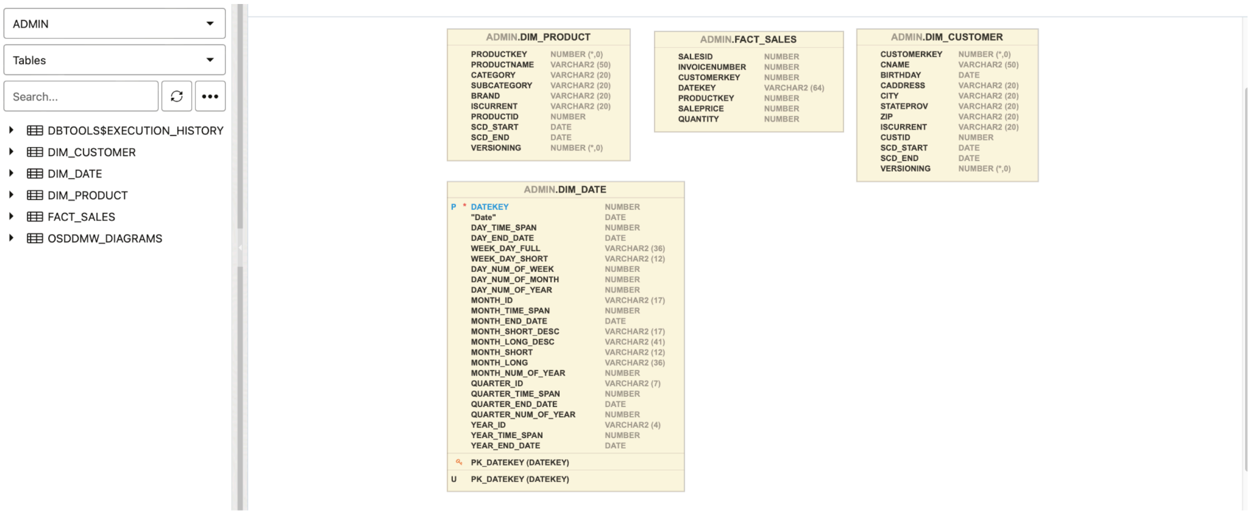
Execution time: 0.006 seconds

	ODUCTKEY	PRODUCTNAME	CATEGORY	SUBCATEGORY	BRAND	ISCURRENT	PRODUCTID	SCD_START	SCD_END	VERSIONING
1	(null)	Milk	Diary	Liquid	Buffalo Farms	Y	2	2/1/2024, 12:00:00 A	12/31/2099, 12:00:0(	1
2	(null)	Chocolate Chip Cool	Candy	Cookies	Nothing Breader	Y	3	3/1/2024, 12:00:00 A	12/31/2099, 12:00:0(	1
3	(null)	Eggs	Diary	Solid	Rochester Farms	Y	4	4/1/2024, 12:00:00 A	12/31/2099, 12:00:0(	1
4	(null)	Rotini	Wheat	Pasta	Buffalo Farms	Y	5	5/1/2024, 12:00:00 A	12/31/2099, 12:00:0(	1
5	(null)	Cinnamon Bread	Wheat	Bread	Nothing Breader	Y	1	1/1/2024, 12:00:00 A	12/31/2099, 12:00:0(	1

ⓧ 1 △ 0 ⚙ 0 | 1:39:43 AM - 5 rows total

Introduction:

Generate a diagram of the database you created and attach a screenshot to your report.



ADMIN.DIM_PRODUCT

PRODUCTKEY	NUMBER (*,0)
PRODUCTNAME	VARCHAR2 (50)
CATEGORY	VARCHAR2 (20)
SUBCATEGORY	VARCHAR2 (20)
BRAND	VARCHAR2 (20)
ISCURRENT	VARCHAR2 (20)
PRODUCTID	NUMBER
SCD_START	DATE
SCD_END	DATE
VERSIONING	NUMBER (*,0)

ADMIN.FACT_SALES

SALESID	NUMBER
INVOICENUMBER	NUMBER
CUSTOMERKEY	NUMBER
DATEKEY	VARCHAR2 (64)
PRODUCTKEY	NUMBER
SALEPRICE	NUMBER
QUANTITY	NUMBER

ADMIN.DIM_CUSTOMER

CUSTOMERKEY	NUMBER (*,0)
CNAME	VARCHAR2 (50)
BIRTHDAY	DATE
CADDRESS	VARCHAR2 (20)
CITY	VARCHAR2 (20)
STATEPROV	VARCHAR2 (20)
ZIP	VARCHAR2 (20)
ISCURRENT	VARCHAR2 (20)
CUSTID	NUMBER
SCD_START	DATE
SCD_END	DATE
VERSIONING	NUMBER (*,0)

ADMIN.DIM_DATE

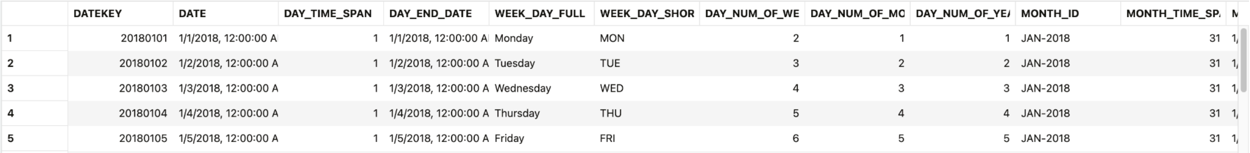
P *	DATEKEY	NUMBER
	"Date"	DATE
	DAY_TIME_SPAN	NUMBER
	DAY_END_DATE	DATE
	WEEK_DAY_FULL	VARCHAR2 (36)
	WEEK_DAY_SHORT	VARCHAR2 (12)
	DAY_NUM_OF_WEEK	NUMBER
	DAY_NUM_OF_MONTH	NUMBER
	DAY_NUM_OF_YEAR	NUMBER
	MONTH_ID	VARCHAR2 (17)
	MONTH_TIME_SPAN	NUMBER
	MONTH_END_DATE	DATE
	MONTH_SHORT_DESC	VARCHAR2 (17)
	MONTH_LONG_DESC	VARCHAR2 (41)
	MONTH_SHORT	VARCHAR2 (12)
	MONTH_LONG	VARCHAR2 (36)
	MONTH_NUM_OF_YEAR	NUMBER
	QUARTER_ID	VARCHAR2 (7)
	QUARTER_TIME_SPAN	NUMBER
	QUARTER_END_DATE	DATE
	QUARTER_NUM_OF_YEAR	NUMBER
	YEAR_ID	VARCHAR2 (4)
	YEAR_TIME_SPAN	NUMBER
	YEAR_END_DATE	DATE
	PK_DATEKEY (DATEKEY)	
U	PK_DATEKEY (DATEKEY)	

Based on the diagram generated, what is this database missing that you'd expect to see? Why might it be missing this component? Use materials we've discussed in class and research (citing sources) to write no more than 2-4 sentences in response.

There are a few more tables that would make the database complete such as the Store_Dimension table. This will be useful in determining which stores are performing interms of sales and profitability. We could also have tables on inventory which can help categorise items sales by categories. Structure-wise wise there are missing relationships or connections between primary and secondary keys between the tables and there are no pre-assigned Product keys.

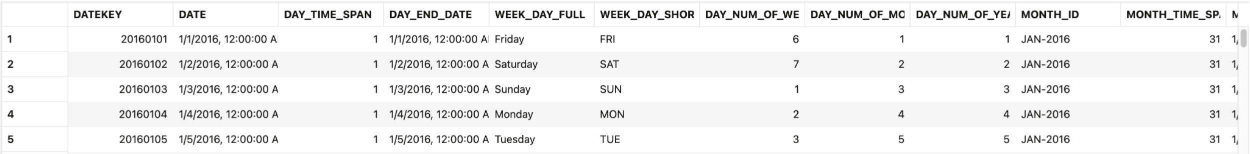
Loading your dimension tables:

Open a new query and type "SELECT * FROM Dim_Date" and click "Execute". Paste a screenshot of the result in your report (with a sample of rows).



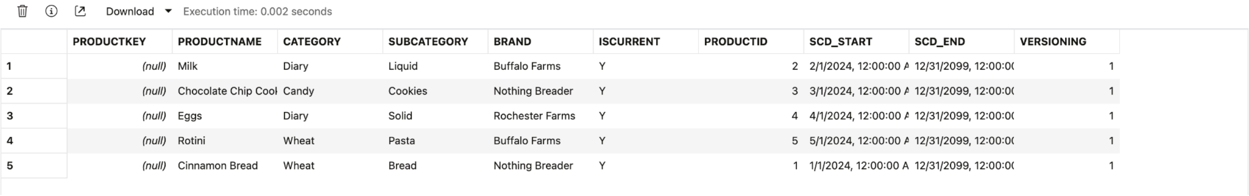
	DATEKEY	DATE	DAY_TIME_SPAN	DAY_END_DATE	WEEK_DAY_FULL	WEEK_DAY_SHOR	DAY_NUM_OF_WE	DAY_NUM_OF_MO	DAY_NUM_OF_YE	MONTH_ID	MONTH_TIME_SP	M
1	20180101	1/1/2018, 12:00:00 A	1	1/1/2018, 12:00:00 A	Monday	MON	2	1	1	JAN-2018	31	1/
2	20180102	1/2/2018, 12:00:00 A	1	1/2/2018, 12:00:00 A	Tuesday	TUE	3	2	2	JAN-2018	31	1/
3	20180103	1/3/2018, 12:00:00 A	1	1/3/2018, 12:00:00 A	Wednesday	WED	4	3	3	JAN-2018	31	1/
4	20180104	1/4/2018, 12:00:00 A	1	1/4/2018, 12:00:00 A	Thursday	THU	5	4	4	JAN-2018	31	1/
5	20180105	1/5/2018, 12:00:00 A	1	1/5/2018, 12:00:00 A	Friday	FRI	6	5	5	JAN-2018	31	1/

Run the "SELECT * FROM Dim_Date" query again and paste a screenshot in your report, along with a copy of the full updated script you ran.



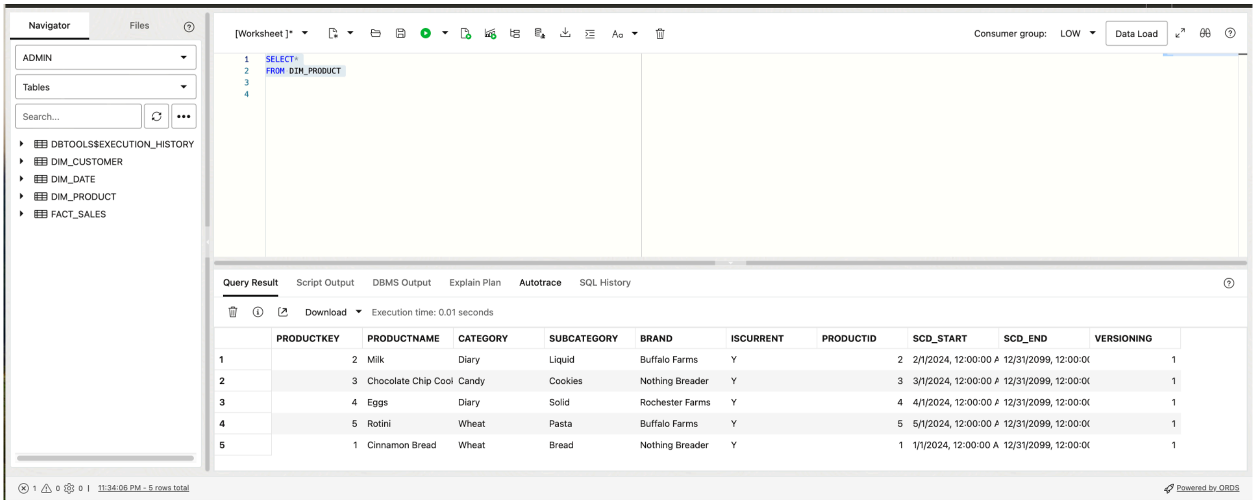
	DATEKEY	DATE	DAY_TIME_SPAN	DAY_END_DATE	WEEK_DAY_FULL	WEEK_DAY_SHOR	DAY_NUM_OF_WE	DAY_NUM_OF_MO	DAY_NUM_OF_YE	MONTH_ID	MONTH_TIME_SP	M
1	20160101	1/1/2016, 12:00:00 A	1	1/1/2016, 12:00:00 A	Friday	FRI	6	1	1	JAN-2016	31	1/
2	20160102	1/2/2016, 12:00:00 A	1	1/2/2016, 12:00:00 A	Saturday	SAT	7	2	2	JAN-2016	31	1/
3	20160103	1/3/2016, 12:00:00 A	1	1/3/2016, 12:00:00 A	Sunday	SUN	1	3	3	JAN-2016	31	1/
4	20160104	1/4/2016, 12:00:00 A	1	1/4/2016, 12:00:00 A	Monday	MON	2	4	4	JAN-2016	31	1/
5	20160105	1/5/2016, 12:00:00 A	1	1/5/2016, 12:00:00 A	Tuesday	TUE	3	5	5	JAN-2016	31	1/

Once complete, run a "Select * from Dim_Product" query, take a screenshot, and paste the result in your report.



	PRODUCTKEY	PRODUCTNAME	CATEGORY	SUBCATEGORY	BRAND	ISCURRENT	PRODUCTID	SCD_START	SCD_END	VERSIONING
1	*(null)*	Milk	Diary	Liquid	Buffalo Farms	Y	2	2/1/2024, 12:00:00 A	12/31/2099, 12:00:0	1
2	*(null)*	Chocolate Chip Cool	Candy	Cookies	Nothing Breader	Y	3	3/1/2024, 12:00:00 A	12/31/2099, 12:00:0	1
3	*(null)*	Eggs	Diary	Solid	Rochester Farms	Y	4	4/1/2024, 12:00:00 A	12/31/2099, 12:00:0	1
4	*(null)*	Rotini	Wheat	Pasta	Buffalo Farms	Y	5	5/1/2024, 12:00:00 A	12/31/2099, 12:00:0	1
5	*(null)*	Cinnamon Bread	Wheat	Bread	Nothing Breader	Y	1	1/1/2024, 12:00:00 A	12/31/2099, 12:00:0	1

Back in Oracle Cloud, execute the following query: "SELECT * FROM Dim_Product". Take a screenshot of the results and place it into your report.



```
SELECT*
FROM DIM_PRODUCT
```

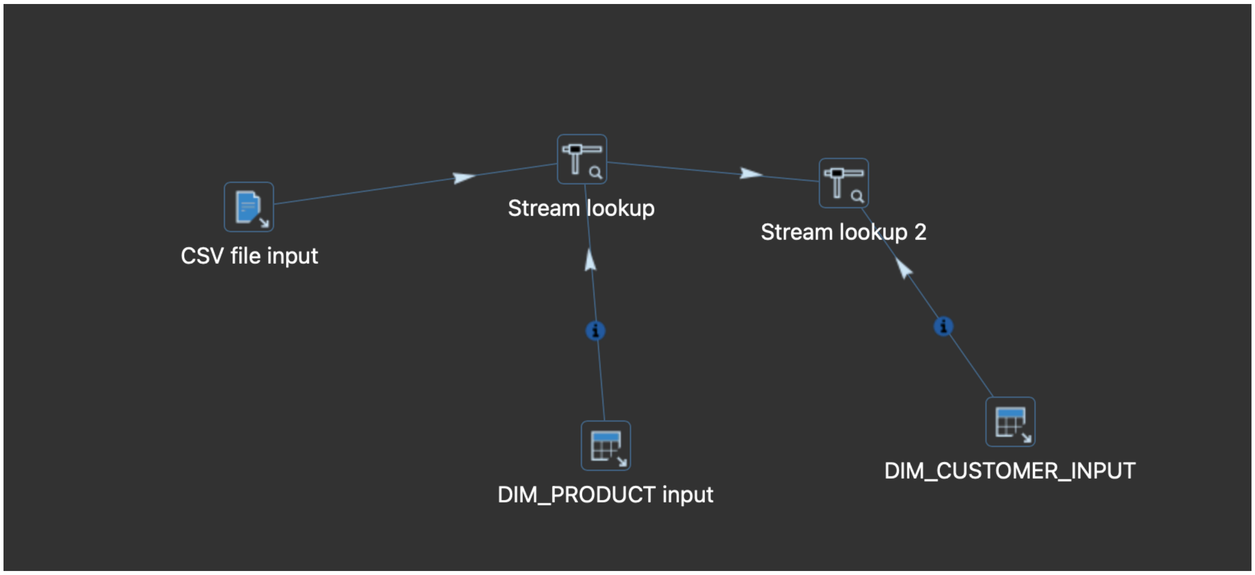
Execution time: 0.01 seconds

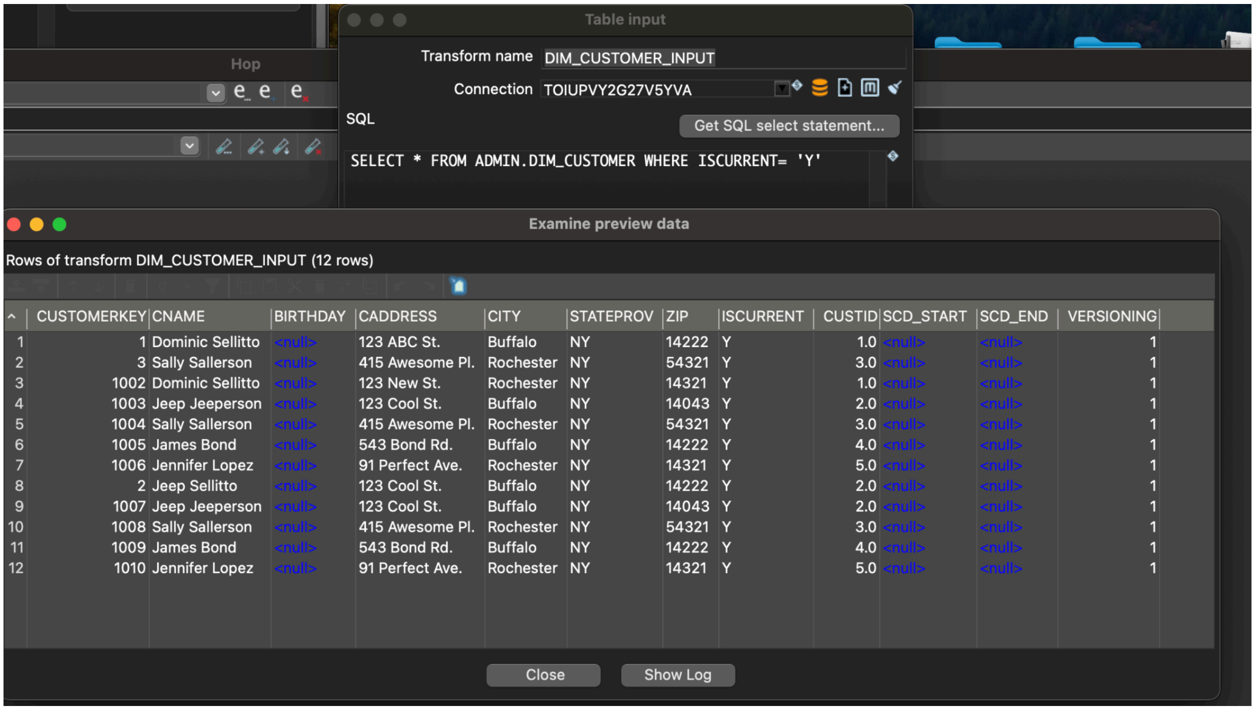
	PRODUCTKEY	PRODUCTNAME	CATEGORY	SUBCATEGORY	BRAND	ISCURRENT	PRODUCTID	SCD_START	SCD_END	VERSIONING
1	2	Milk	Diary	Liquid	Buffalo Farms	Y	2	2/1/2024, 12:00:00 A	12/31/2099, 12:00:0(	1
2	3	Chocolate Chip Cool	Candy	Cookies	Nothing Breader	Y	3	3/1/2024, 12:00:00 A	12/31/2099, 12:00:0(	1
3	4	Eggs	Diary	Solid	Rochester Farms	Y	4	4/1/2024, 12:00:00 A	12/31/2099, 12:00:0(	1
4	5	Rotini	Wheat	Pasta	Buffalo Farms	Y	5	5/1/2024, 12:00:00 A	12/31/2099, 12:00:0(	1
5	1	Cinnamon Bread	Wheat	Bread	Nothing Breader	Y	1	1/1/2024, 12:00:00 A	12/31/2099, 12:00:0(	1

ⓧ 1 △ 0 ⚙ 0 | 11:34:06 PM - 5 rows total

Loading your fact table:

Once you think you have the lookups done correctly, you'll need to take a screenshot of your entire flow, along with screenshots of your second table input and second stream input nodes (from edit mode in each) and append them to your report.





Transform name DIM_CUSTOMER_INPUT

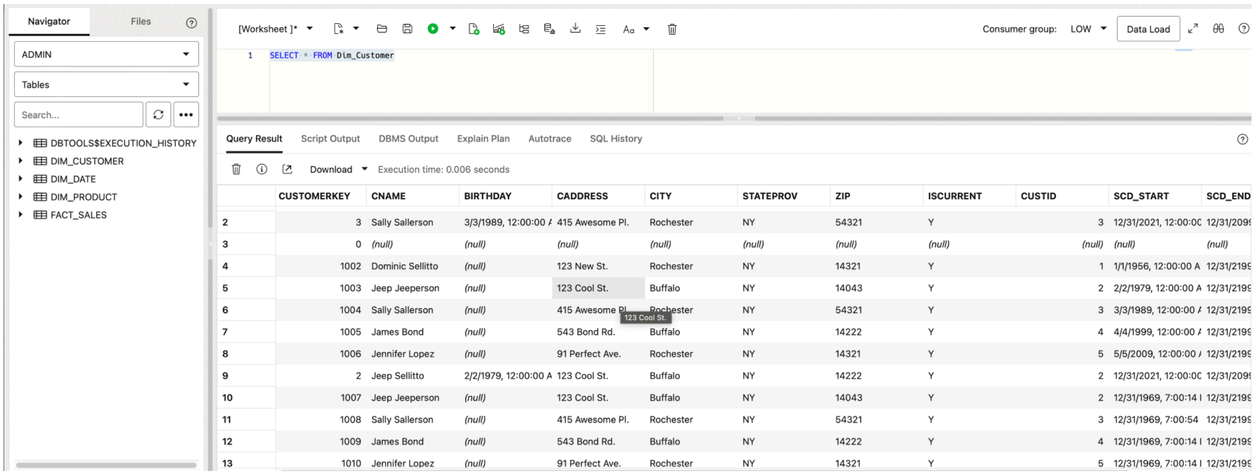
Connection TOIUPVY2G27V5YVA

SQL

```
SELECT * FROM ADMIN.DIM_CUSTOMER WHERE ISCURRENT= 'Y'
```

Rows of transform DIM_CUSTOMER_INPUT (12 rows)

^	CUSTOMERKEY	CNAME	BIRTHDAY	CADDRESS	CITY	STATEPROV	ZIP	ISCURRENT	CUSTID	SCD_START	SCD_END	VERSIONING
1	1	Dominic Sellitto	<null>	123 ABC St.	Buffalo	NY	14222	Y	1.0	<null>	<null>	1
2	3	Sally Sallerson	<null>	415 Awesome Pl.	Rochester	NY	54321	Y	3.0	<null>	<null>	1
3	1002	Dominic Sellitto	<null>	123 New St.	Rochester	NY	14321	Y	1.0	<null>	<null>	1
4	1003	Jeep Jeeperson	<null>	123 Cool St.	Buffalo	NY	14043	Y	2.0	<null>	<null>	1
5	1004	Sally Sallerson	<null>	415 Awesome Pl.	Rochester	NY	54321	Y	3.0	<null>	<null>	1
6	1005	James Bond	<null>	543 Bond Rd.	Buffalo	NY	14222	Y	4.0	<null>	<null>	1
7	1006	Jennifer Lopez	<null>	91 Perfect Ave.	Rochester	NY	14321	Y	5.0	<null>	<null>	1
8	2	Jeep Sellitto	<null>	123 Cool St.	Buffalo	NY	14222	Y	2.0	<null>	<null>	1
9	1007	Jeep Jeeperson	<null>	123 Cool St.	Buffalo	NY	14043	Y	2.0	<null>	<null>	1
10	1008	Sally Sallerson	<null>	415 Awesome Pl.	Rochester	NY	54321	Y	3.0	<null>	<null>	1
11	1009	James Bond	<null>	543 Bond Rd.	Buffalo	NY	14222	Y	4.0	<null>	<null>	1
12	1010	Jennifer Lopez	<null>	91 Perfect Ave.	Rochester	NY	14321	Y	5.0	<null>	<null>	1



```
SELECT * FROM Dim_Customer
```

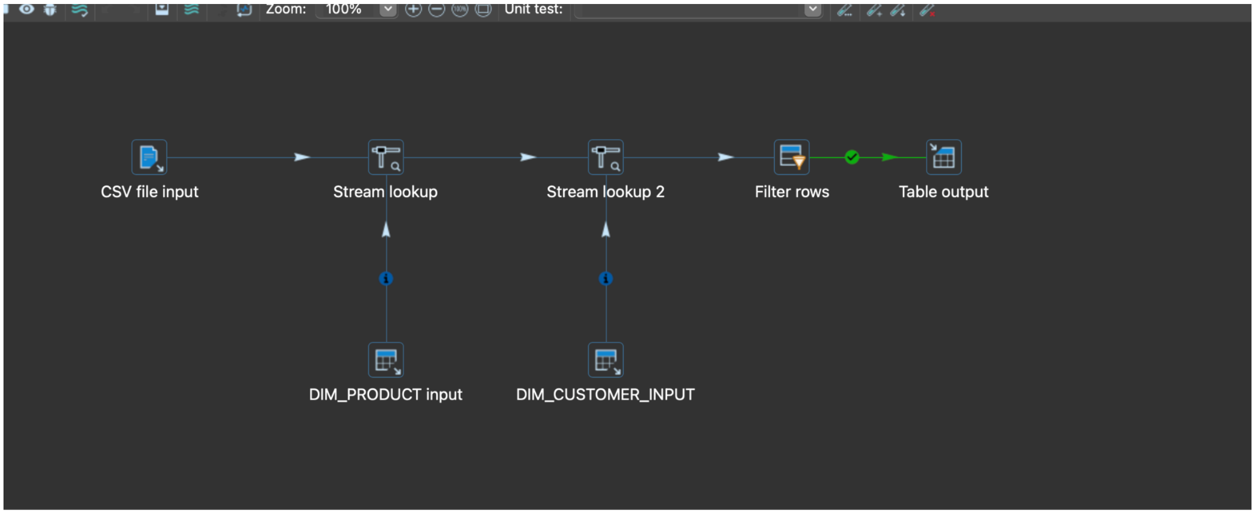
Execution time: 0.006 seconds

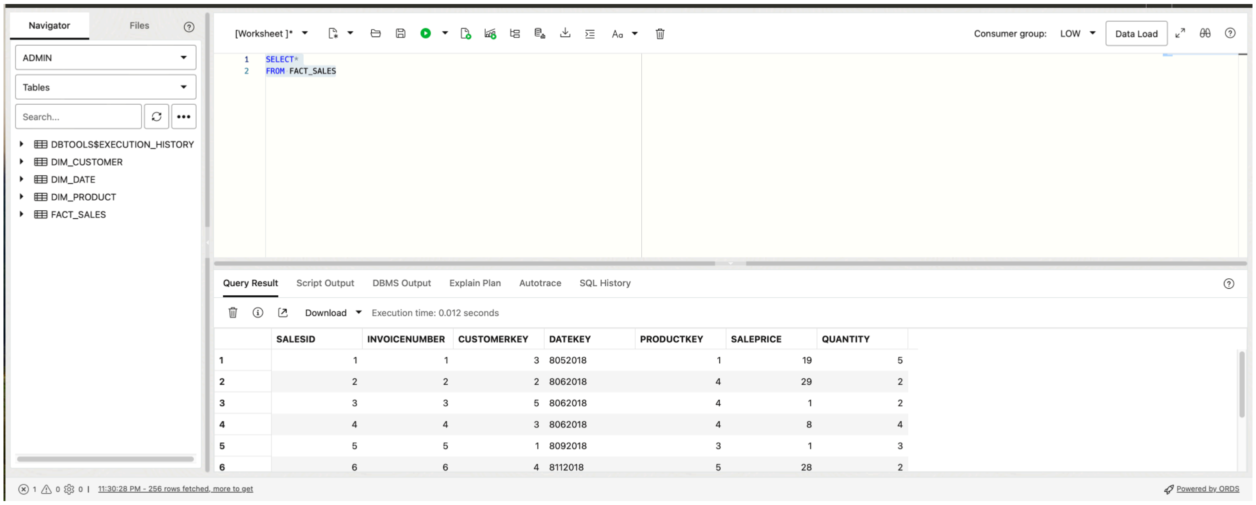
	CUSTOMERKEY	CNAME	BIRTHDAY	CADDRESS	CITY	STATEPROV	ZIP	ISCURRENT	CUSTID	SCD_START	SCD_END
2	3	Sally Sallerson	3/3/1989, 12:00:00 A	415 Awesome Pl.	Rochester	NY	54321	Y	3	12/31/2021, 12:00:0	12/31/209
3	0	(null)	(null)	(null)	(null)	(null)	(null)	(null)	(null)	(null)	(null)
4	1002	Dominic Sellitto	(null)	123 New St.	Rochester	NY	14321	Y	1	1/1/1956, 12:00:00 A	12/31/219
5	1003	Jeep Jeeperson	(null)	123 Cool St.	Buffalo	NY	14043	Y	2	2/2/1979, 12:00:00 A	12/31/219
6	1004	Sally Sallerson	(null)	415 Awesome Pl.	Rochester	NY	54321	Y	3	3/3/1989, 12:00:00 A	12/31/219
7	1005	James Bond	(null)	543 Bond Rd.	Buffalo	NY	14222	Y	4	4/4/1999, 12:00:00 A	12/31/219
8	1006	Jennifer Lopez	(null)	91 Perfect Ave.	Rochester	NY	14321	Y	5	5/5/2009, 12:00:00 A	12/31/219
9	2	Jeep Sellitto	2/2/1979, 12:00:00 A	123 Cool St.	Buffalo	NY	14222	Y	2	12/31/2021, 12:00:0	12/31/209
10	1007	Jeep Jeeperson	(null)	123 Cool St.	Buffalo	NY	14043	Y	2	12/31/1969, 7:00:14	12/31/219
11	1008	Sally Sallerson	(null)	415 Awesome Pl.	Rochester	NY	54321	Y	3	12/31/1969, 7:00:54	12/31/219
12	1009	James Bond	(null)	543 Bond Rd.	Buffalo	NY	14222	Y	4	12/31/1969, 7:00:14	12/31/219
13	1010	Jennifer Lopez	(null)	91 Perfect Ave.	Rochester	NY	14321	Y	5	12/31/1969, 7:00:14	12/31/219

You might have noticed we're not doing a lookup for the date dimension. Write 1-2 sentences detailing why we don't need to. You'll be able to figure this out likely by looking at the data in the fact CSV and the date dimension.

This is due to the format the data for the date_dimension table which will make it challenging to do lookup.

Finally, take a screenshot of your output from the select query you ran, a screenshot of your completed pipeline, and attach it to your submission.





```
SELECT*
FROM FACT_SALES
```

Execution time: 0.012 seconds

	SALESID	INVOICENUMBER	CUSTOMERKEY	DATEKEY	PRODUCTKEY	SALEPRICE	QUANTITY
1	1	1	3	8052018	1	19	5
2	2	2	2	8062018	4	29	2
3	3	3	5	8062018	4	1	2
4	4	4	3	8062018	4	8	4
5	5	5	1	8092018	3	1	3
6	6	6	4	8112018	5	28	2

1 0 0 | 11:30:28 PM - 256 rows fetched, more to get

Powered by ORDS