Project Report

Group 10:

Ankur Pandey - +1 (857) 398 5940
pandey.anku@northeastern.edu

Giridhar Babu - +1 (857) 398-5730 babu.gi@northeastern.edu

Percentage of Effort Contributed by Ankur Pandey: 50%

Percentage of Effort Contributed by Giridhar Babu: 50%

Signature of Student 1: *Ankur Pandey*

Signature of Student 2: *Giridhar Babu*

Submission Date: 12/07/ 2024

Table of Contents

Milestone 1

Problem Statement

Sail Foods needs to design and implement a comprehensive database system to manage the culinary operations for 5,000+ sailors on the USS Theodore Roosevelt, serving over 15,000 meals per day. The system must efficiently handle inventory management, meal planning, food preparation, serving logistics, and waste management while ensuring food safety, nutritional requirements, and operational efficiency in the unique environment of a Nimitz-class aircraft carrier.

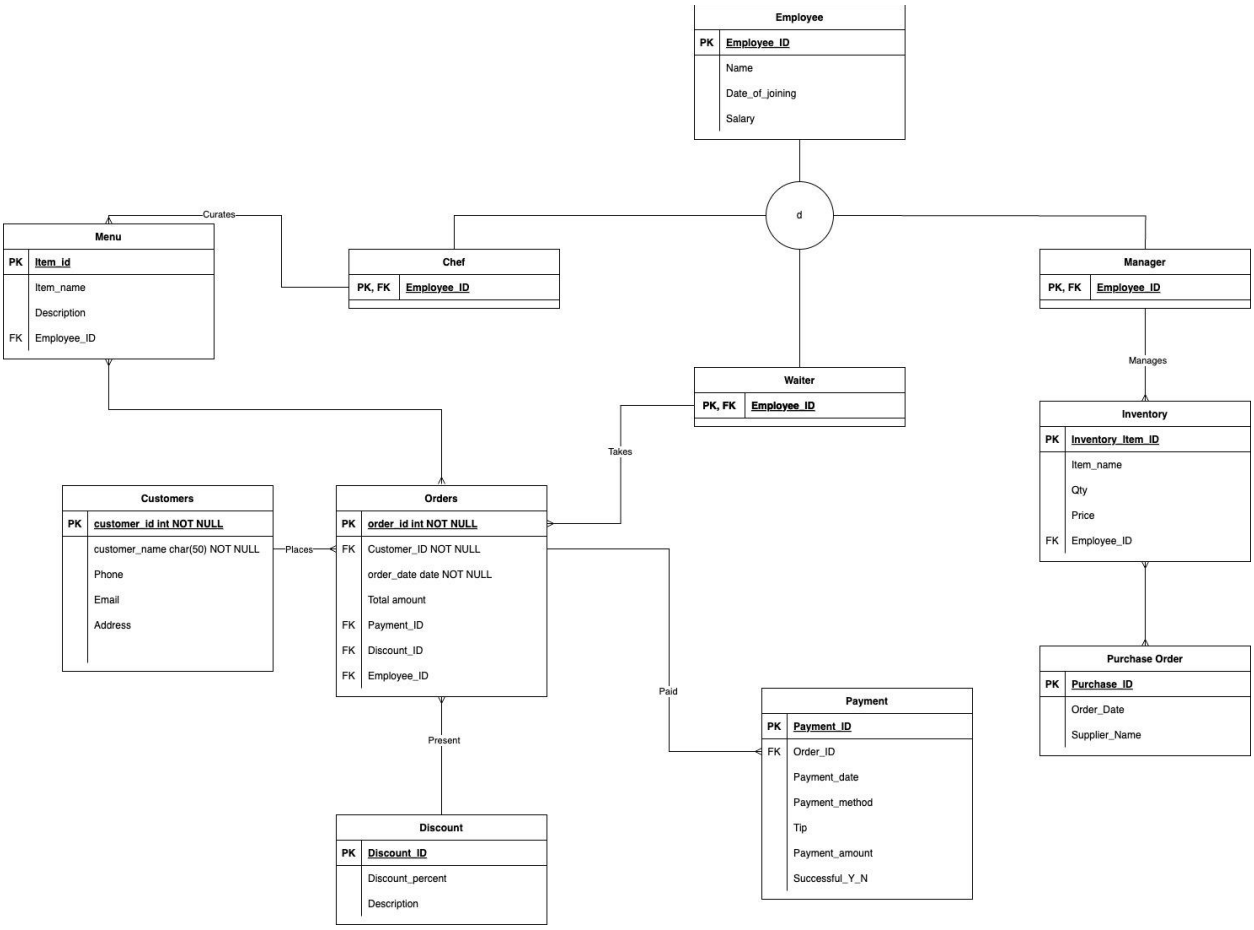
Database Definition

The database system for Sail Foods' operation on the USS Theodore Roosevelt should encompass the following key components:

1. Inventory Management:
    - Track all food items, ingredients, and supplies
    - Monitor stock levels and expiration dates
    - Manage procurement and restocking processes
2. Menu Planning:
    - Store recipes and nutritional information
    - Create and manage meal plans
    - Accommodate dietary restrictions and preferences
3. Food Preparation:
    - Schedule food preparation tasks
    - Track cooking equipment usage and maintenance
    - Manage food safety and quality control processes
4. Serving Logistics:
    - Manage meal serving schedules
    - Track meal consumption and popularity
    - Handle special meal requests and events
5. Personnel Management:
    - Store information on culinary staff and their schedules
    - Track staff certifications and training
6. Waste Management:
    - Monitor food waste
    - Manage disposal processes
7. Reporting and Analytics:
    - Generate reports on various aspects of the operation
    - Provide data for performance analysis and optimization

Milestone 2

ER Diagram



Entities:

1. **Customer**: Represents customers who place orders.
    - Attributes: customer_id (PK), customer_name, phone, email, address
2. **Order**: Represents food orders made by customers.
    - Attributes: order_id (PK), order_date, total_amount, customer_id (FK), payment_id, employee_id (FK), discount_id (FK)
3. **Payment**: Represents payments made for orders.
    - Attributes: payment_id (PK), order_id (FK), payment_date, payment_type, amount
4. **Discount**: Represents discounts available for orders.
    - Attributes: discount_id (PK), discount_percentage, description
5. **Menu**: Contains all food items available.
    - Attributes: item_id (PK), item_name, price, employee_id (FK)
6. **Employee**: Represents employees, which include Waitrs, Managers, and Chefs.
    - Attributes: employee_id (PK), name, Date_of_joining, salary
7. **Waiter** (Specializes Employee):
    - Attributes: employee_id (PK, FK)
8. **Chef** (Specializes Employee):
    - Attributes: employee_id (PK, FK), specialization, years_of_experience
9. **Manager** (Specializes Employee):
    - Attributes: employee_id (PK, FK)
10. **Inventory**: Represents the food inventory.
    - Attributes: inventory_item_id (PK), item_name, qty, unit_price, Employee_id
11. **Purchase Order**: Represents purchase orders for inventory.
    - Attributes: purchase _id (PK), inventory_id (FK), order_date, quantity_ordered, supplier_name

Relationships:

1. **Customer to Order:**
    - **A customer can place multiple orders, but each order is placed by a single customer.**
    - **Relationship: Customer (1) → Order (N)**
    - **Type: One-to-Many (1)**
    - **Foreign Key: customer_id in the Order table refers to customer_id in the Customer table.**
2. **Order to Payment:**
    - **Each order can have one or more payments, but each payment is for a specific order.**
    - **Relationship: Order (1) → Payment (N)**
    - **Type: One-to-Many (1)**
    - **Foreign Key: order_id in the Payment table refers to order_id in the Order table.**
3. **Order to Discount:**
    - **An order can apply a discount, but a discount can be used for multiple orders.**
    - **Relationship: Order (N) → Discount (1)**
    - **Type: Many-to-One (N:1)**
    - **Foreign Key: discount_id in the Order table refers to discount_id in the Discount table.**
4. **Order to Menu (via Junction Table Order_Menu):**
    - **An order can contain multiple menu items, and a menu item can be part of multiple orders (many-to-many relationship).**
    - **Relationship: Order (N) → Order_Menu ← Menu (N)**
    - **Type: Many-to-Many (N)**
    - **Junction Table: Order_Menu with order_id (FK) and menu_id (FK) links Order and Menu.**
5. **Employee to Order (for Waiters):**
    - **A waiter can serve multiple orders, but an order is served by a single employee (waiter).**
    - **Relationship: Employee (Waiter) (1) → Order (N)**
    - **Type: One-to-Many (1)**
    - **Foreign Key: employee_id in the Order table refers to employee_id in the Employee table for Waiters.**
6. **Employee to Menu (for Chefs):**
    - **A chef can create multiple menu items, but a menu item is associated with a single chef.**
    - **Relationship: Employee (Chef) (1) → Menu (N)**
    - **Type: One-to-Many (1)**
    - **Foreign Key: employee_id in the Menu table refers to employee_id in the Employee table for Chefs.**

7. **Employee to Inventory (for Managers):**
    - **A manager can oversee multiple inventory items, but each inventory item is associated with one manager.**
    - **Relationship: Employee (Manager) (1) → Inventory (N)**
    - **Type: One-to-Many (1)**
    - **Foreign Key: employee_id in the Inventory table refers to employee_id in the Employee table for Managers.**
8. **Purchase Order to Inventory:**
    - **Each purchase order is made for a single inventory item, but each inventory item can have multiple purchase orders over time.**
    - **Relationship: Inventory (1) → Purchase Order (N)**
    - **Type: One-to-Many (1)**
    - **Foreign Key: inventory_item_id in the Purchase Order table refers to inventory_item_id in the Inventory table.**
9. **Employee Specialization (Inheritance):**
    - **Employee generalizes Waiter, Chef, and Manager. Each employee can be specialized into one of these roles.**
    - **Type: One-to-One (1:1) Inheritance**
    - **Foreign Key: employee_id is shared across Employee, Waiter, Chef, and Manager.**

Diagram Structure:

- **Customer** → **Order** → **Payment**, **Discount**
- **Order** → **Menu** (N relationship)
- **Menu** → **Category**
- **Order** → **Waitress** (Handled by waitress from the Employee table)
- **Manager** → **Employee**
- **Inventory** → **Purchase Order**

Dimensions:

1. Customer Dimension
- **Attributes**: customer_id, customer_name, phone, email, address
- **Hierarchy**: None

2. Order Dimension
- **Attributes**: order_id, order_date, total_amount, customer_id, payment_id, employee_id, discount_id
- **Hierarchy**: order_date (can be split into year > month > day for time-based analysis)

3. Employee Dimension
- **Attributes**: employee_id, name, date_of_joining, salary
- **Hierarchy**: None

4. Menu Dimension
- **Attributes**: item_id, item_name, price, employee_id
- **Hierarchy**: Menu Type (Veg/Non-Veg) > Item Name

5. Payment Dimension
- **Attributes**: payment_id, order_id, payment_date, payment_type, amount
- **Hierarchy**: payment_date (can be split into year > month > day for time-based analysis)

6. Discount Dimension
- **Attributes**: discount_id, discount_percentage, description
- **Hierarchy**: None

7. Inventory Dimension
- **Attributes**: inventory_item_id, item_name, qty, unit_price, employee_id
- **Hierarchy**: None

8. Purchase Order Dimension
- **Attributes**: purchase_id, inventory_item_id, order_date, quantity_ordered, supplier_name
- **Hierarchy**: order_date (can be split into year > month > day for time-based analysis)

Measures

1. Total Bill Amount

- **Source**: Order.total_amount
- **Type**: Additive
- **Aggregation**: SUM
- **Description**: Represents the total value of orders placed by customers. This can be aggregated across various dimensions such as time, employee (waiter/chef), customer, and menu items.

2. Total Payment Amount

- **Source**: Payment.amount
- **Type**: Additive
- **Aggregation**: SUM
- **Description**: Represents the total payments made by customers. This measure provides insights into revenue collected across dimensions like time, payment type, and order.

3. Discount Amount

- **Source**: Discount.discount_percentage * Order.total_amount
- **Type**: Calculated
- **Aggregation**: SUM
- **Description**: Represents the total discount applied to orders. This helps analyze the impact of discounts on revenue.

4. Total Inventory Cost

- **Source**: Inventory.unit_price * Inventory.qty
- **Type**: Additive
- **Aggregation**: SUM
- **Description**: Represents the total cost of inventory items. It can be aggregated by item or supplier to understand inventory costs.

5. Total Items Ordered

- **Source**: Count of Order_items (assuming a table links each order to items)
- **Type**: Additive
- **Aggregation**: COUNT
- **Description**: This measure represents the total number of items ordered. It can be analyzed across time, customers, and menu items to understand purchasing trends.

6. Number of Orders

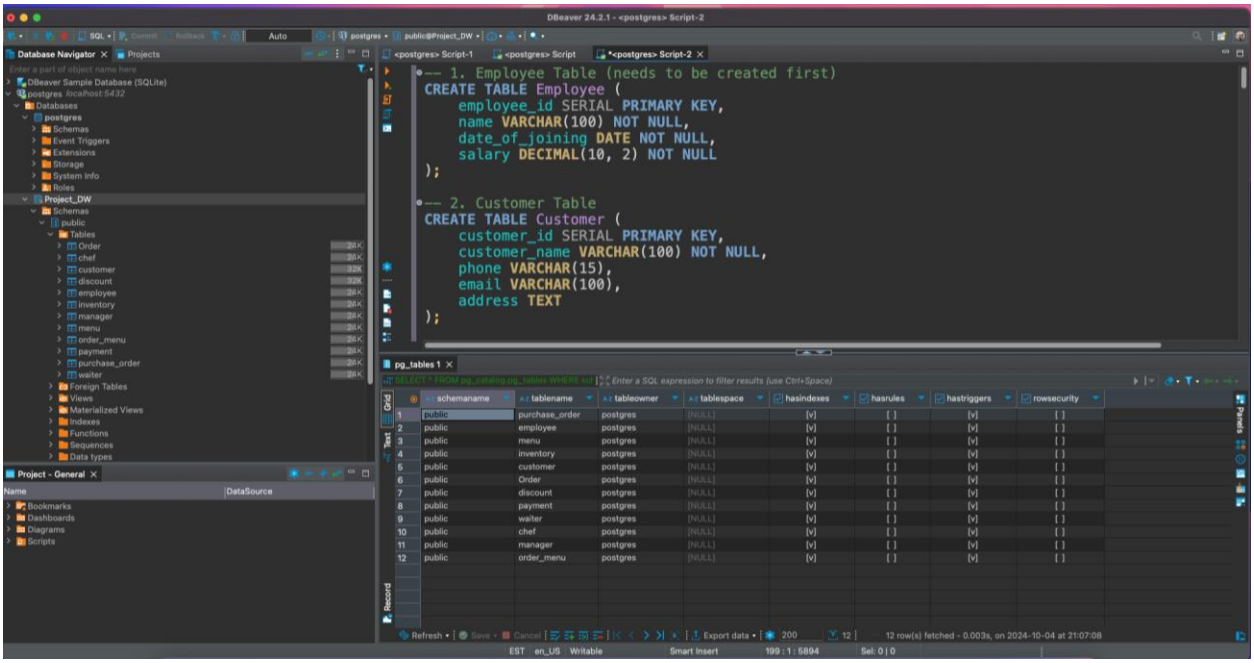
- **Source**: Order.order_id
- **Type**: Count
- **Aggregation**: COUNT
- **Description**: Represents the total number of orders placed. It can be broken down by time, customer, or employee (waiter/manager).

SQL Query Execution Order Summary:

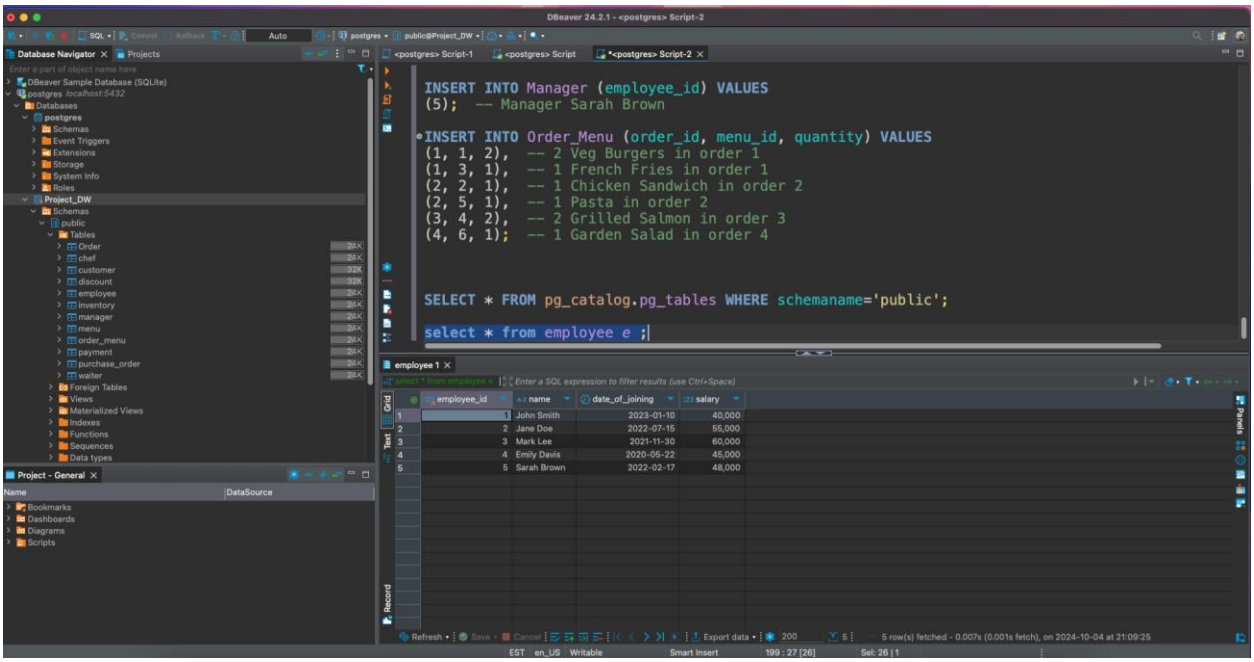
1. **Employee**
2. **Customer**
3. **Discount**
4. **Menu** (references **Employee**)
5. **Inventory** (references **Employee**)
6. **Purchase_Order** (references **Inventory**)
7. **Order** (references **Customer**, **Employee**, and **Discount**)
8. **Payment** (references **Order**)
9. **Add payment_id foreign key constraint to Order**
10. **Waiter** (specializes **Employee**)
11. **Chef** (specializes **Employee**)
12. **Manager** (specializes **Employee**)
13. **Order_Menu** (junction table between **Order** and **Menu**)

Implementation in SQL:

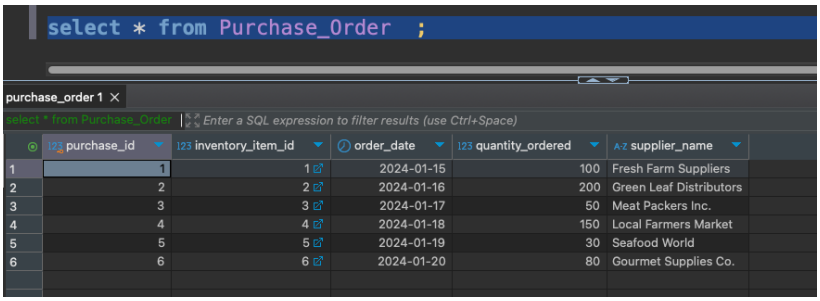
1. List of all the tables being created



2. List of values being populated in the Employee table



3. List of values being populated in the Purchase_order table



Similarly, we have populated all our tables. For now, we have inserted only 5 records in each table, but we aim to add up to 1,000 records by the next milestone. I'm attaching the SQL script in a .txt file format below.

https://drive.google.com/file/d/1GDqnbHKxYPCRflZLQZI4awitcco_hprH/view?usp=sharing

Data Population Methodology:

We generated our data in batches using various LLMs and first populated the static data, such as Employee, Customer, and Items. Using this static data, we then generated the transactional data, such as Amount paid, etc.

Milestone 3

1. Fact Tables

Order: This fact table captures detailed information about customer orders. Key measures in this table include:
Order Date: The date the order was placed.
Total Sales Amount: The total value of the order.
Total Items Ordered: The number of items in the order.
Number of Orders: The number of orders made by a customer.

Payment: This fact table captures details related to payments made for orders. Key measures include:
Payment Date: The date the payment was made.
Payment Type: The method of payment (e.g., cash, credit card).
Amount: The amount of money paid for the order.

Inventory Purchase: This fact table captures details about inventory purchases. Key measures include:
Order Date: The date when the inventory was purchased.
Quantity Ordered: The quantity of inventory items ordered.

2. Dimension Tables

Dimension tables provide descriptive data for the facts. In this model, the following dimensions are present:

Employee: Contains information about the employees involved in the business.
Attributes include:

Employee_ID, Name, Date_of_Joining, Salary, and Employee_Type (whether they are a Waiter, Chef, or Manager).
The specialization of employees (Chef, Waiter, Manager) is represented by different roles under the Employee_Type dimension.

Customer: Contains details about customers, such as:
Customer_ID, Customer_Name, Phone, and Address.

Menu: Represents the items on the menu, capturing details such as:
Item_ID, Item_Name, Description, and Price.
The menu is also linked to the chef responsible for specific items.

Discount: Contains information about discounts applied to orders, including:
Discount_ID, Discount_Percent, and Description.

Inventory: Stores details about inventory items, including:

Inventory_Item_ID, Item_Name, Qty (Quantity), Unit_Price, and Supplier_Name.

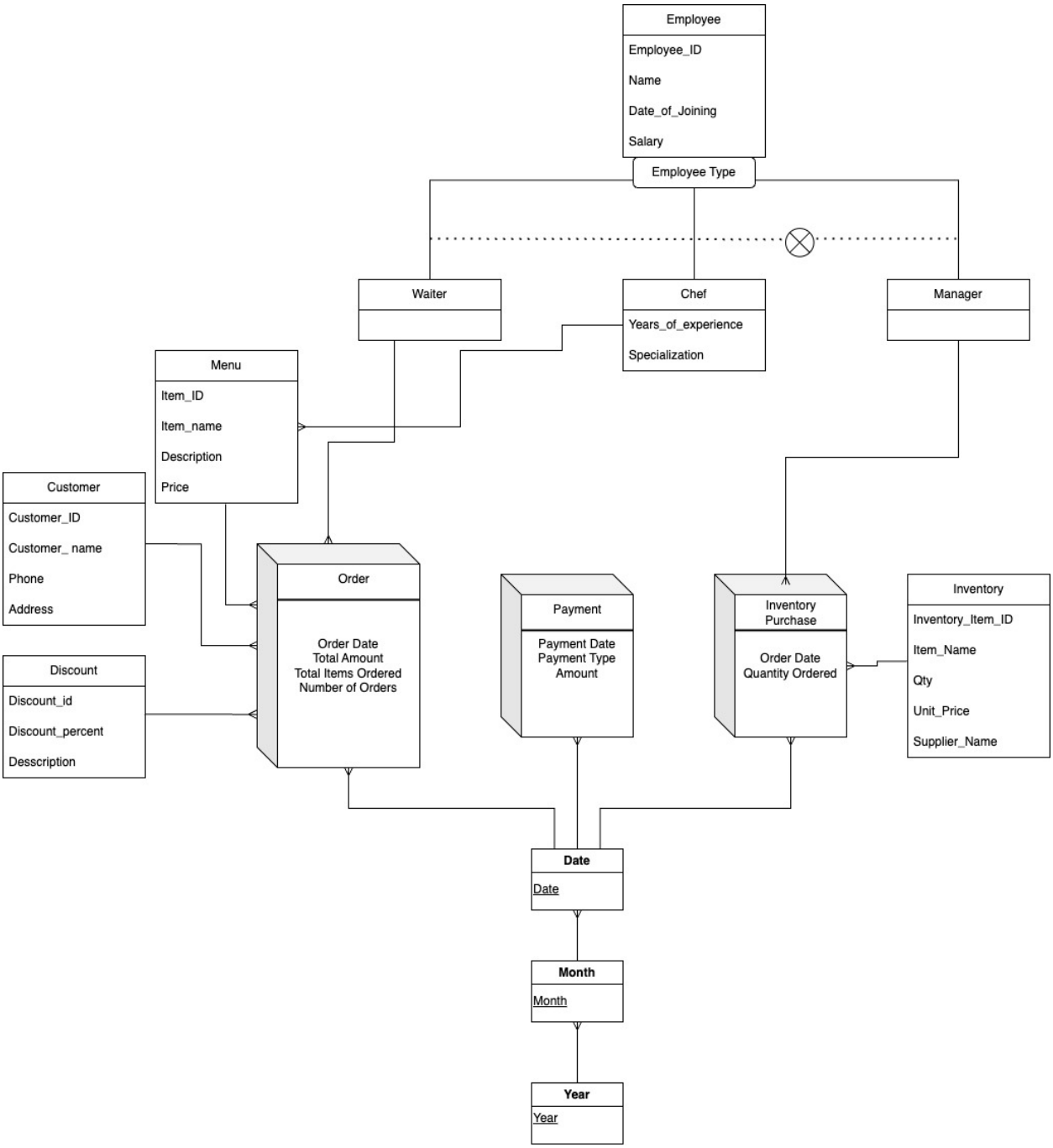
Date: This dimension captures the time-related data and is linked to multiple fact tables.
The date hierarchy is represented by:
Date, Month, and Year levels, allowing for time-based aggregation and analysis.

3. Hierarchies

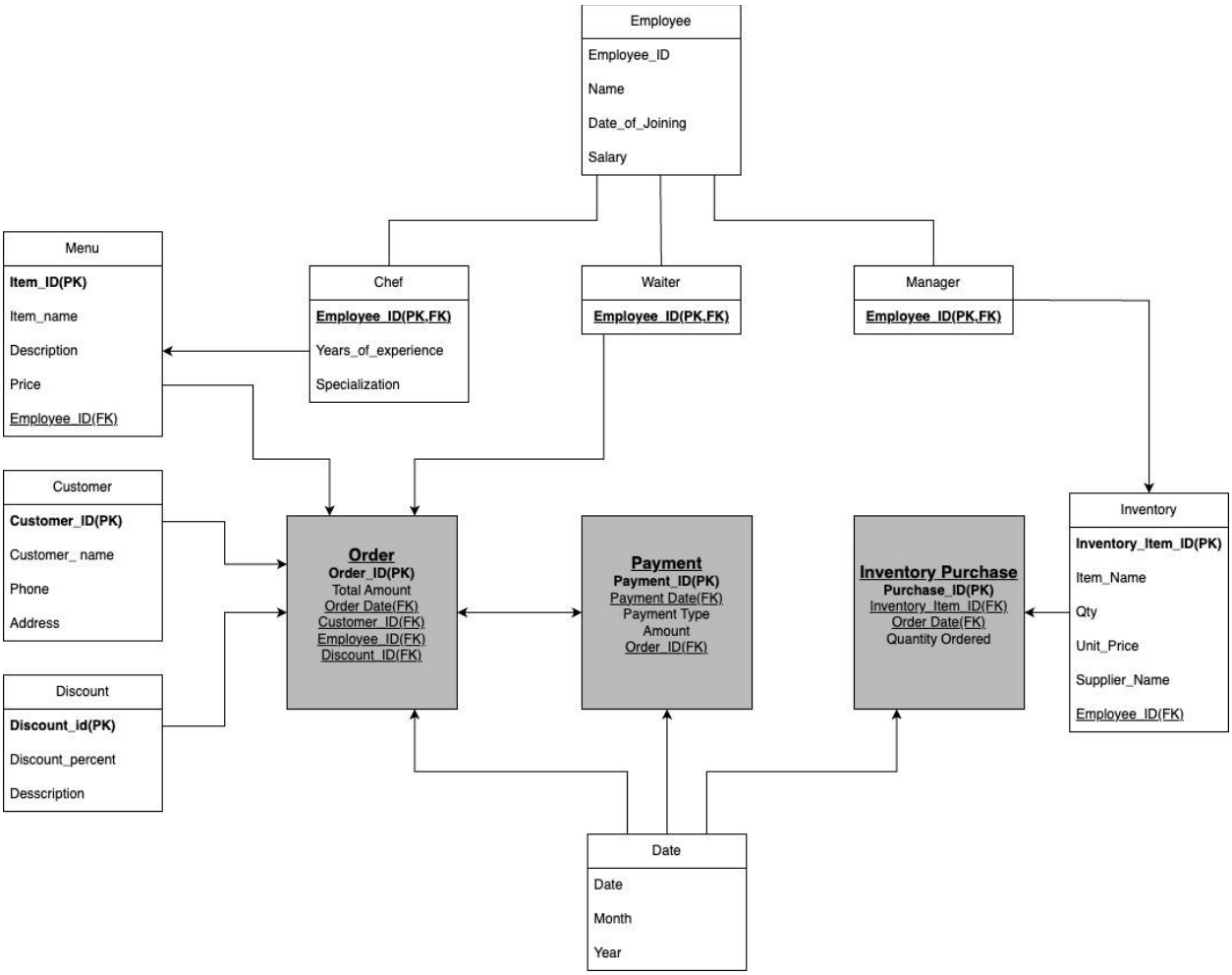
Date Dimension Hierarchy: The Date dimension contains a hierarchy that breaks down into Year, Month, and Day, which supports time-based aggregation (e.g., total sales per year, total payments per month).

Employee Role Hierarchy:
The Employee dimension includes a hierarchy for employee types, categorized as Waiter, Chef, and Manager.

Conceptual Model:

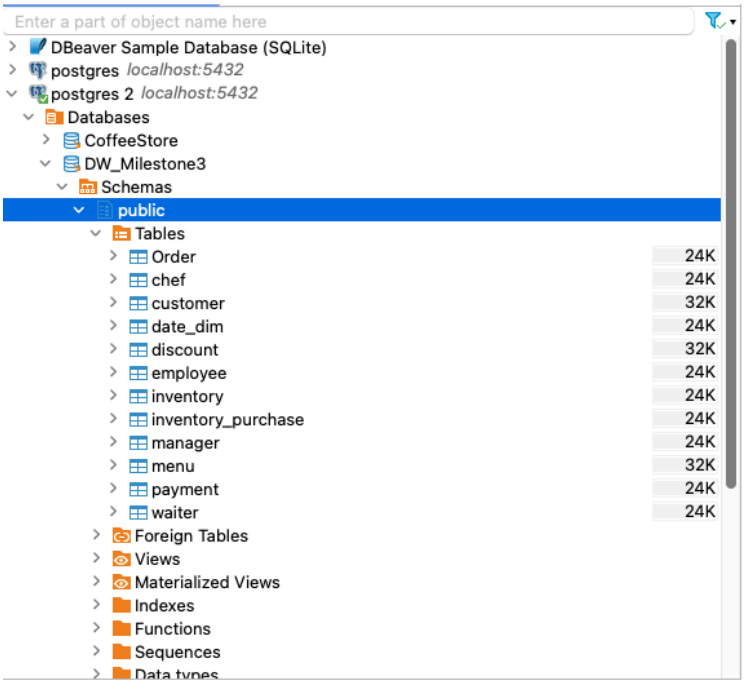


Logical Model:

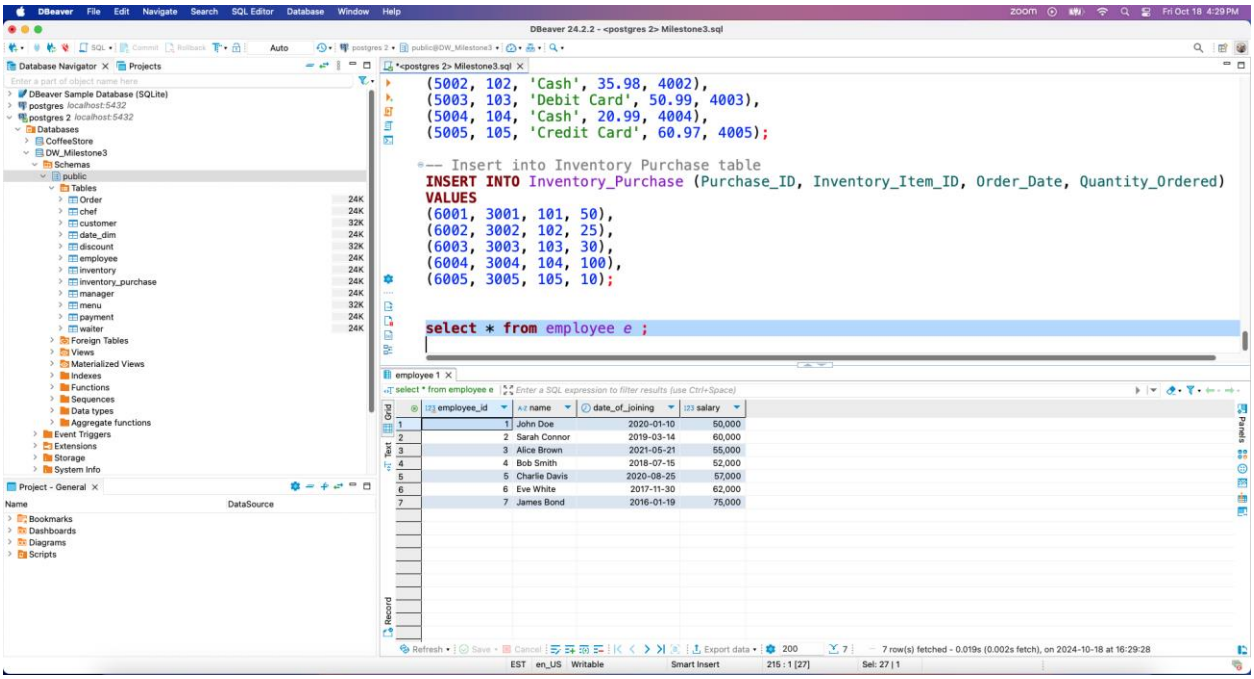


Created the Schema for the Warehouse in PostgresSQL:

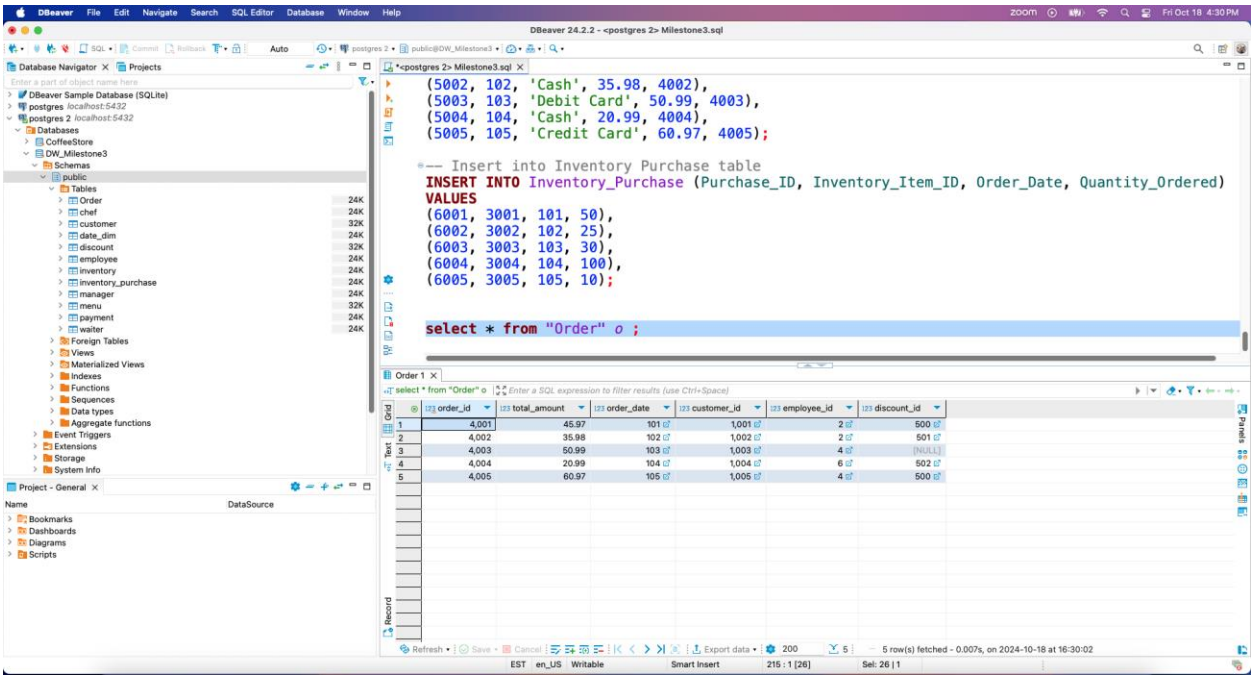
- The tables created:



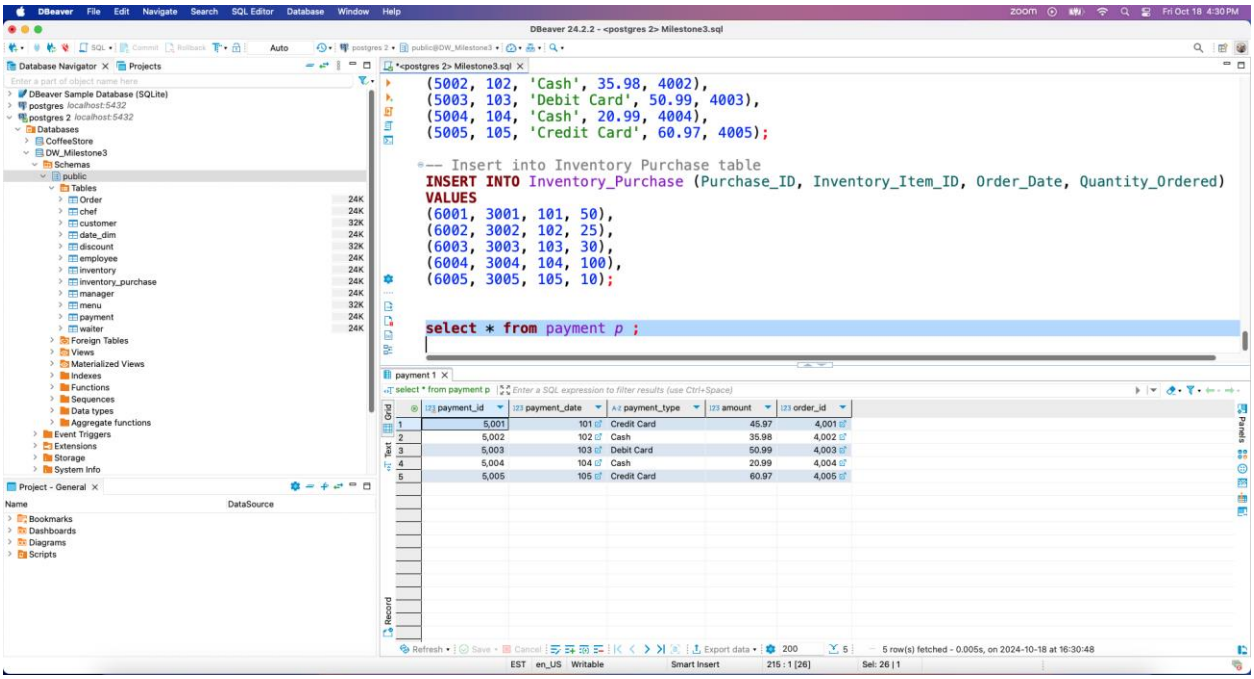
- The schema for Employee table, I have added dummy data.



- The fact table Order, which has the order details:



- The fact table Payment, which has the payment details:


DBeaver File Edit Navigate Search SQL Editor Database Window Help
DBeaver 24.2.2 - <postgres 2> Milestone3.sql
Database Navigator
Projects
DBeaver Sample Database (SQLite)
postgres localhost:5432
postgres 2 localhost:5432
Databases
CoffeeStore
DW_Milestone3
Schemas
public
Tables
Order
chef
customer
date_dim
discount
employee
inventory
inventory_purchase
manager
menu
payment
waiter
Foreign Tables
Views
Materialized Views
Indexes
Functions
Sequences
Data types
Aggregate functions
Event Triggers
Extensions
Storage
System Info
Project - General
Bookmarks
Dashboards
Diagrams
Scripts
(5002, 102, 'Cash', 35.98, 4002),
(5003, 103, 'Debit Card', 50.99, 4003),
(5004, 104, 'Cash', 20.99, 4004),
(5005, 105, 'Credit Card', 60.97, 4005);
-- Insert into Inventory Purchase table
INSERT INTO Inventory_Purchase (Purchase_ID, Inventory_Item_ID, Order_Date, Quantity_Ordered)
VALUES
(6001, 3001, 101, 50),
(6002, 3002, 102, 25),
(6003, 3003, 103, 30),
(6004, 3004, 104, 100),
(6005, 3005, 105, 10);
select * from payment p ;
payment_id
payment_date
payment_type
amount
order_id
5,001
101
Credit Card
45.97
4,001
5,002
102
Cash
35.98
4,002
5,003
103
Debit Card
50.99
4,003
5,004
104
Cash
20.99
4,004
5,005
105
Credit Card
60.97
4,005
5 row(s) fetched - 0.005s, on 2024-10-18 at 16:30:48

Primary Events (Fact Tables):

1. **Order Event** (Order table):
    - **Description**: This event occurs when a customer places an order. It captures information such as the total amount of the order, the customer who placed it, the waiter or manager who handled the order, and any discount applied.
    - **Key Attributes**:
        - Order_ID: The unique identifier for the order.
        - Total_Amount: The total value of the order.
        - Order_Date: The date the order was placed.
        - Customer_ID: The customer who placed the order.
        - Employee_ID: The waiter or manager who handled the order.
        - Discount_ID: Any discount applied to the order.

2. **Payment Event** (Payment table):
    - **Description**: This event occurs when a customer makes a payment for an order. It captures details such as the payment type, the amount paid, and the corresponding order.
    - **Key Attributes**:
        - Payment_ID: The unique identifier for the payment.
        - Payment_Date: The date the payment was made.
        - Payment_Type: The method of payment (credit card, cash, etc.).
        - Amount: The amount paid.
        - Order_ID: The order the payment corresponds to.

3. **Inventory Purchase Event** (Inventory_Purchase table):
    - **Description**: This event occurs when the restaurant or business purchases inventory items. It captures details like the item purchased, the quantity ordered, and the date of the purchase.
    - **Key Attributes**:
        - Purchase_ID: The unique identifier for the purchase.
        - Inventory_Item_ID: The inventory item being purchased.
        - Order_Date: The date of the inventory purchase.
        - Quantity_Ordered: The quantity of the inventory item purchased.

OLAP Operations:

- **Query 1:** Total Sales Amount by Customer, Year, and Month

ROLLUP*(Order, Customer → Customer_Name, Date → Year, Date → Month, SUM(Total_Amount))
Description: Aggregates sales handled by each employee.

- **Query 2:** Total Sales by Employee (Waiters)

ROLLUP*(Order, Employee → Employee_Name, SUM(Total_Amount))
Description: Aggregates sales handled by each employee.

- **Query 3:** Total Payments by Payment Type

AGGREGATE(Payment, Payment_Type, SUM(Amount))
Description: Aggregates total payments received by payment type.

- **Query 4:** Drill Down Sales from Year to Month to Day

DRILLDOWN(Order, Date → Year, Date → Month, Date → Day, SUM(Total_Amount))
Description: Drills down from yearly sales to daily sales.

- **Query 5:** Slice: Sales in the Year 2023

SLICE(Order, Date.Year = 2023, SUM(Total_Amount))
Description: Slices the sales data to show only the year 2023.

- **Query 6:** Inventory Purchases by Supplier and Item

ROLLUP*(Inventory_Purchase, Supplier → Supplier_Name, Inventory → Item_Name, SUM(Quantity_Ordered))
Description: Summarizes inventory purchases by supplier and item.

- **Query 7:** Payments by Date and Employee

ROLLUP*(Payment, Date → Year, Employee → Employee_Name, SUM(Amount))
Description: Aggregates payments received by each employee per year.

- **Query 8:** Top 5 Best-Selling Menu Items

ROLLUP*(Menu, Item_Name, SUM(Total_Amount))
ORDER BY SUM(Total_Amount) DESC LIMIT 5
Description: Finds the top 5 menu items based on total sales

- **Query 9:** Total Discount Given Per Customer by Year

ROLLUP*(Order, Customer → Customer_Name, Date → Year, Discount → Discount_percent, SUM(Total_Amount))
Description: Aggregates total discount per customer by year.

- **Query 10:** Inventory Purchases in 2023

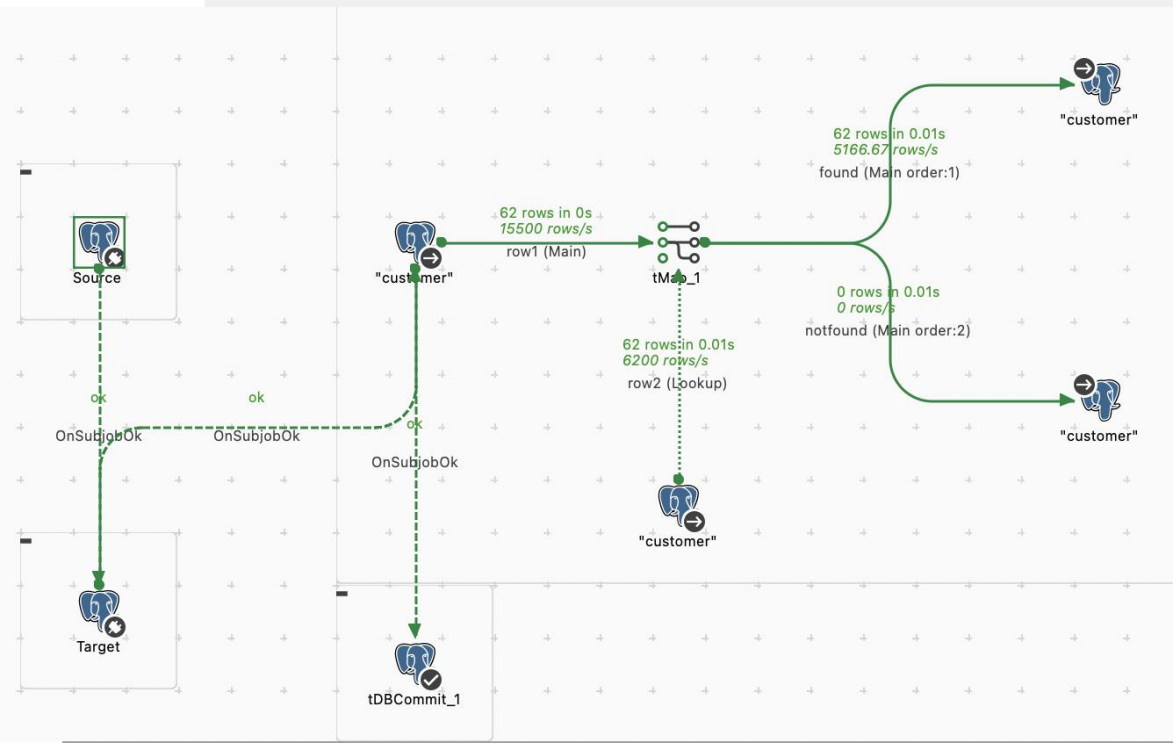
SLICE(Inventory_Purchase, Date.Year = 2023, SUM(Quantity_Ordered))
Description: Slices the inventory purchases to show data for the year 2023.

Milestone 4

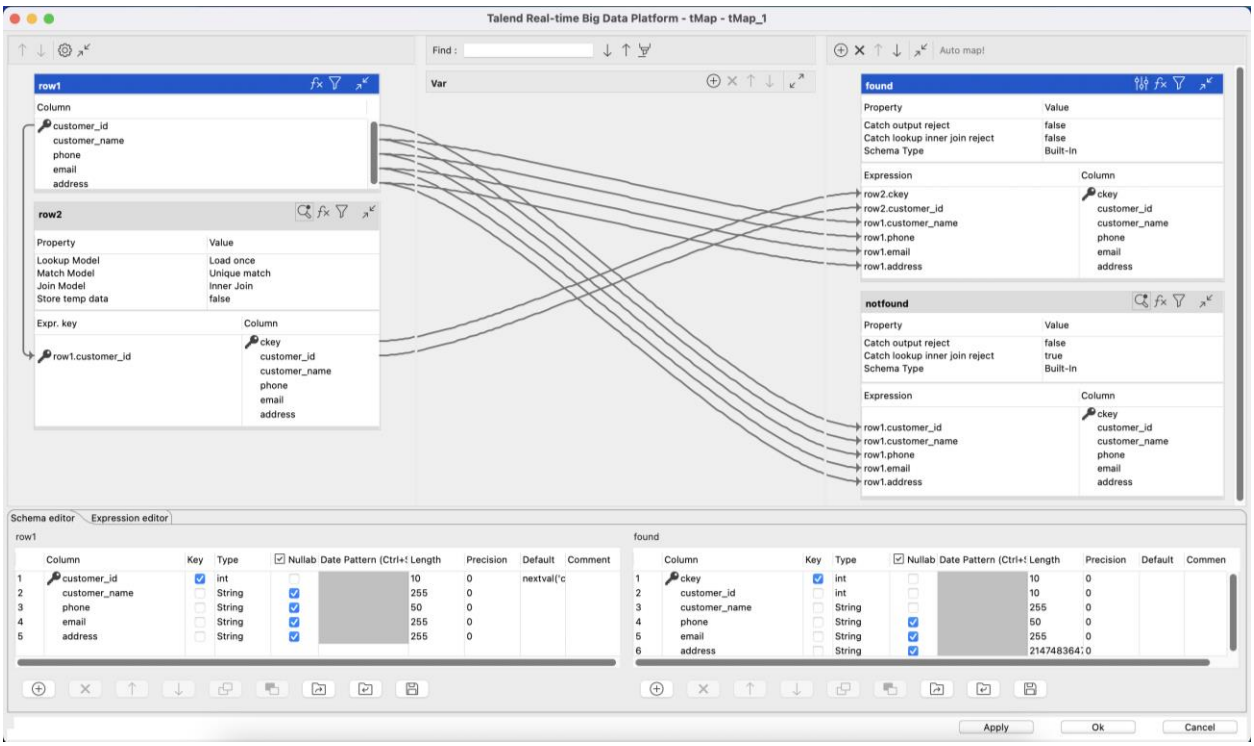
Implemented:

- **Insert/update data with surrogate key**
- **Control flow**
- **Implemented Type 3 SCD**
- **Implemented 2 calculated measures:**
  - **purchase_amount → inventorypurchase in Target**
  - **net_amount → order_details in Target**
- **Used 2 sources of data:**
  - **csv file for date**
  - **PostgresDB for the rest**
- **Performed transformations:**
  - **Date format**
  - **Manipulated the type of data for calculations**

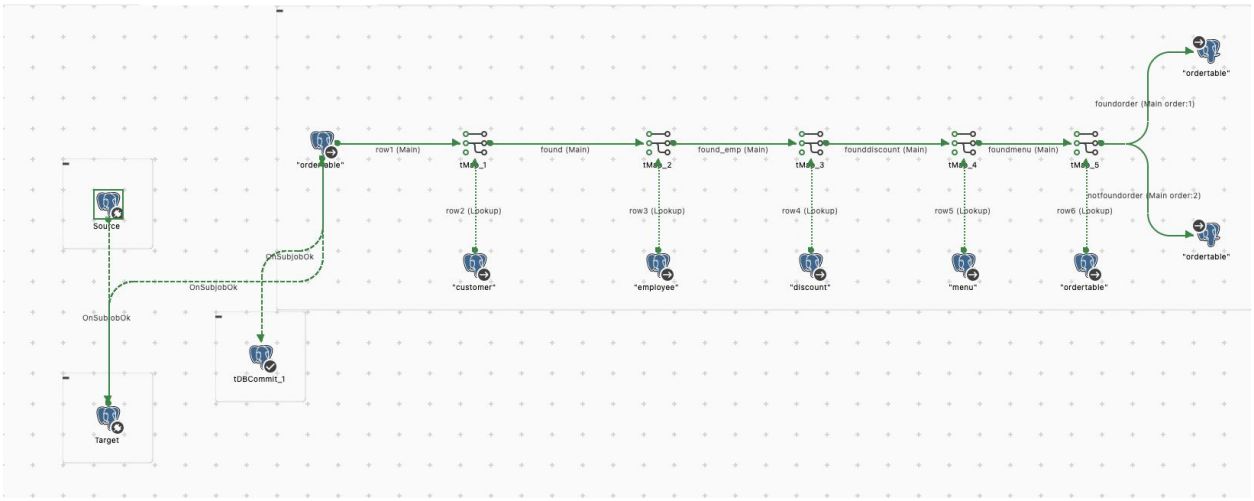
Insert/Update flow for Customer:



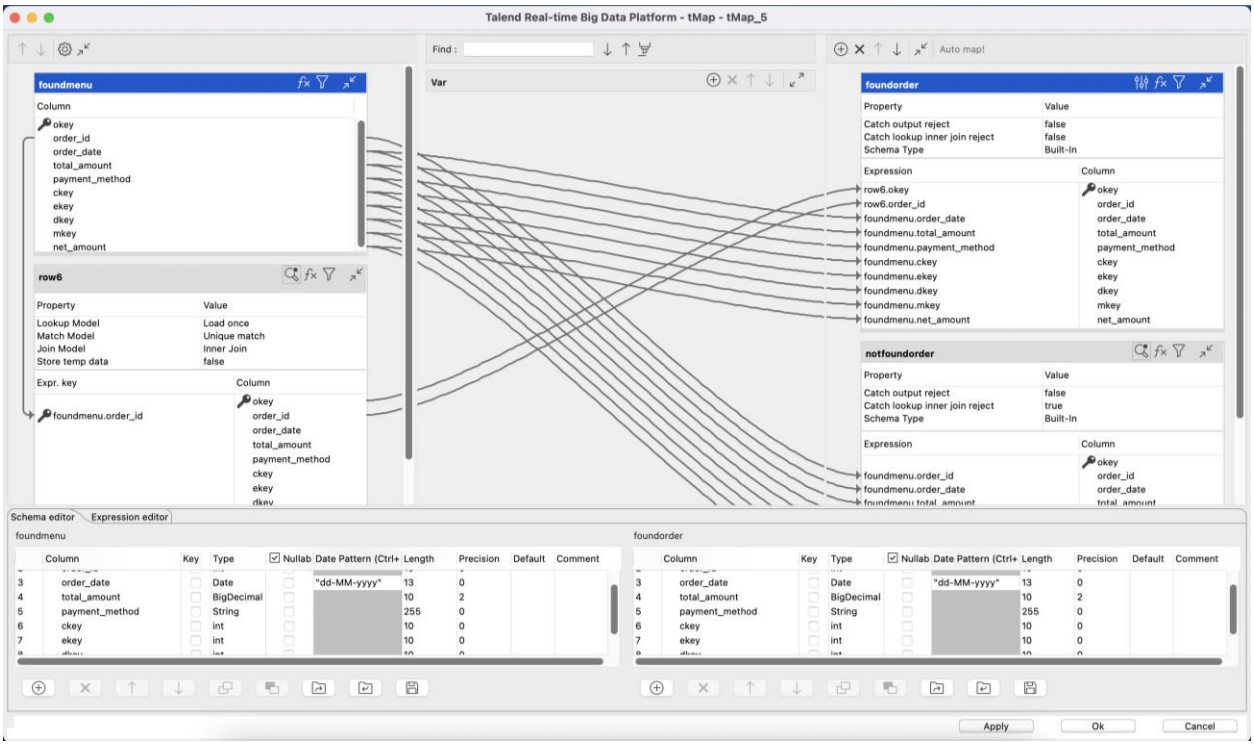
tMap for Customer:



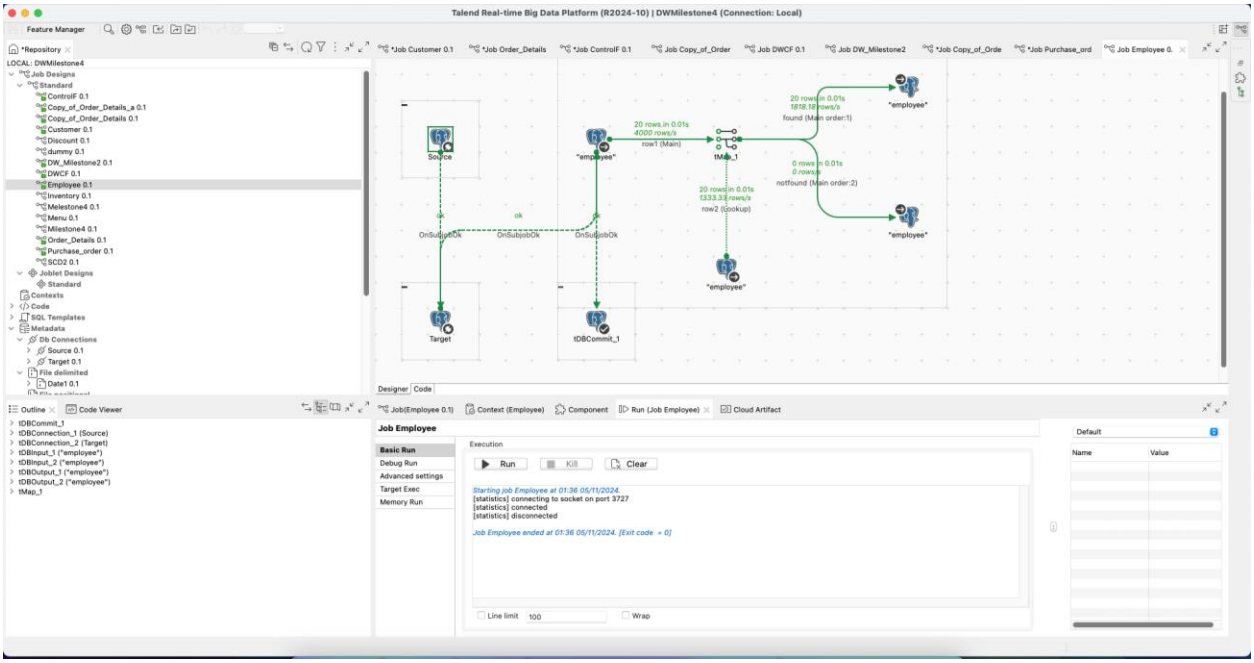
Insert/Update flow for Order_details:



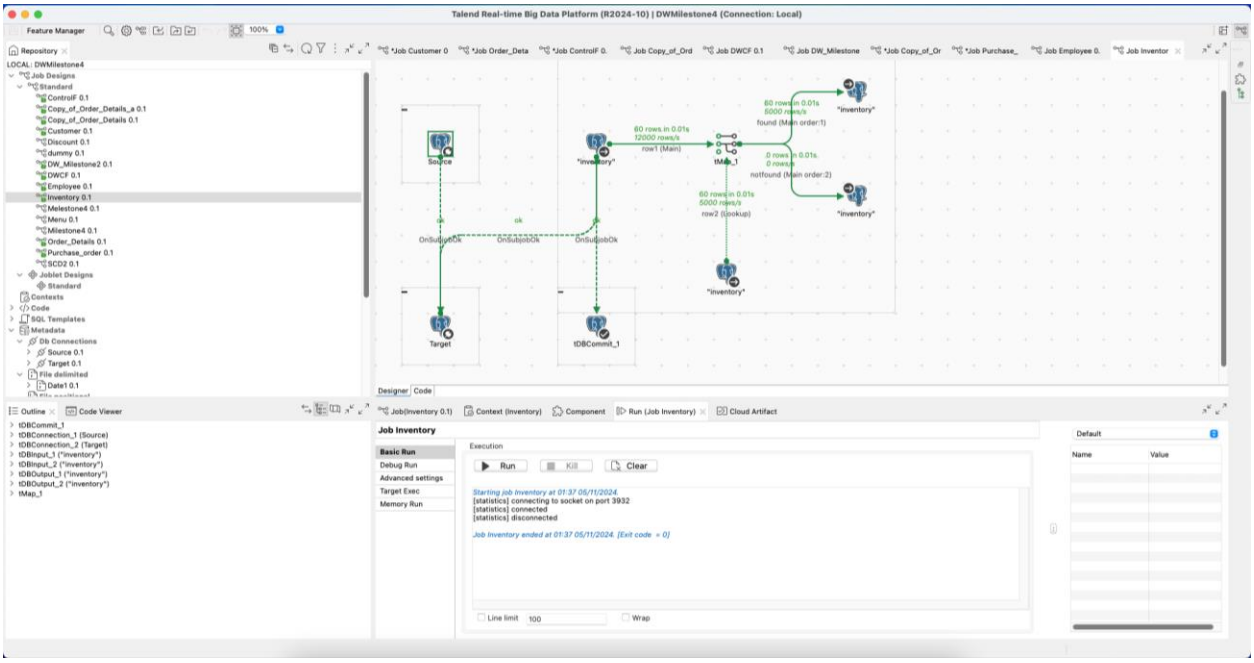
tMap for Order_details:



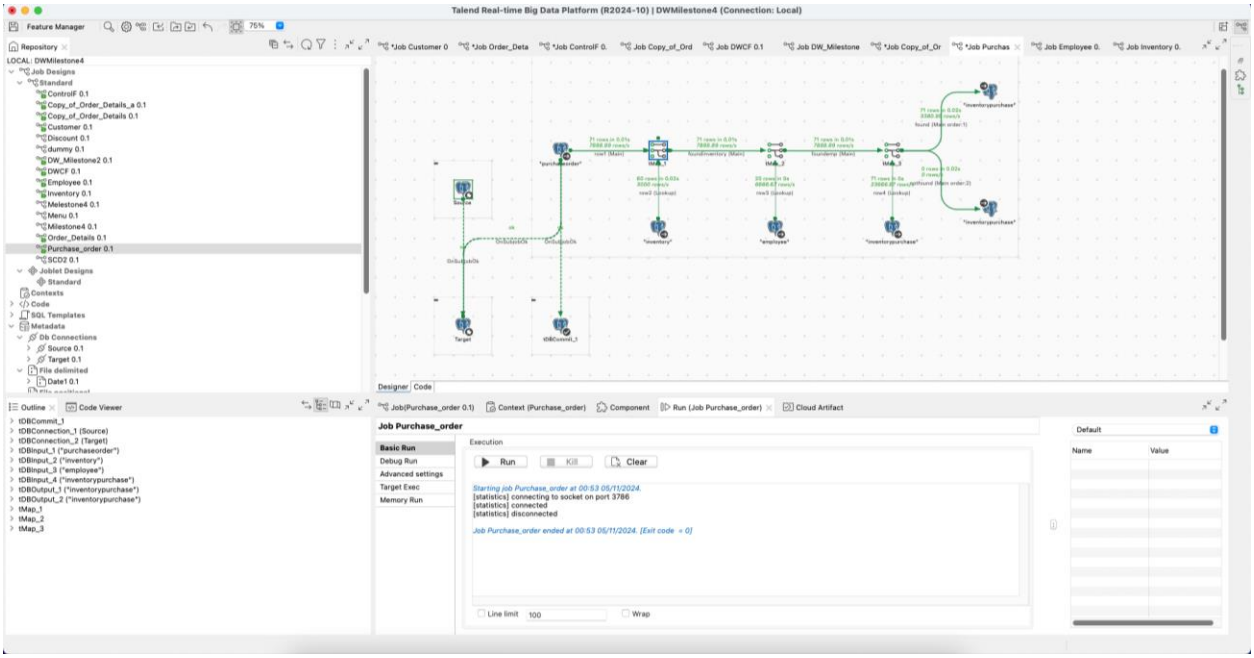
Insert and Update flow for Employee:



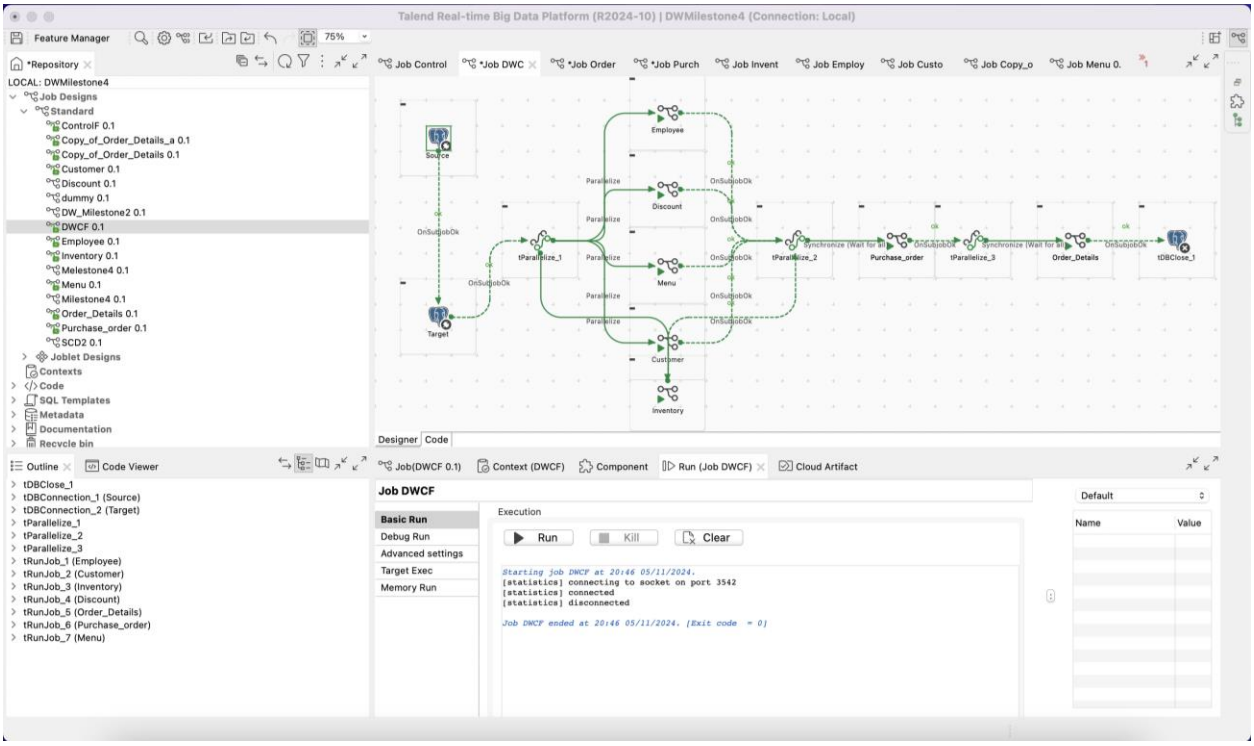
Insert and Update flow for Inventory:



Insert and Update flow for Purchase Order (Fact Table):

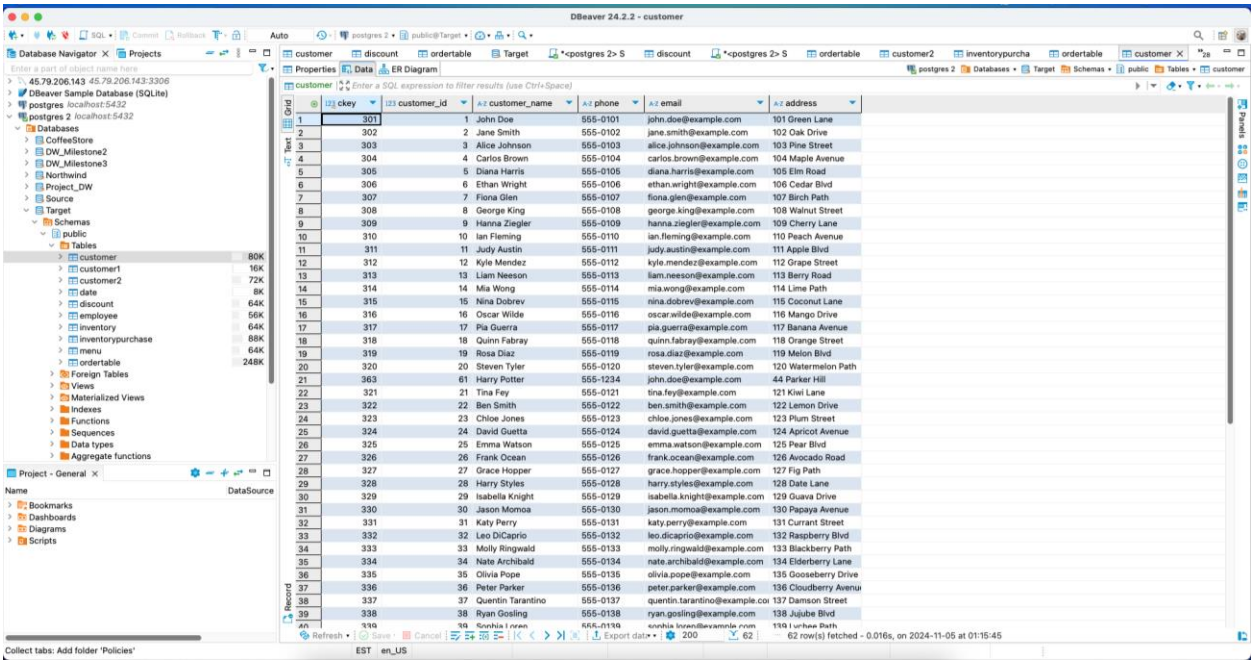


Control Flow:

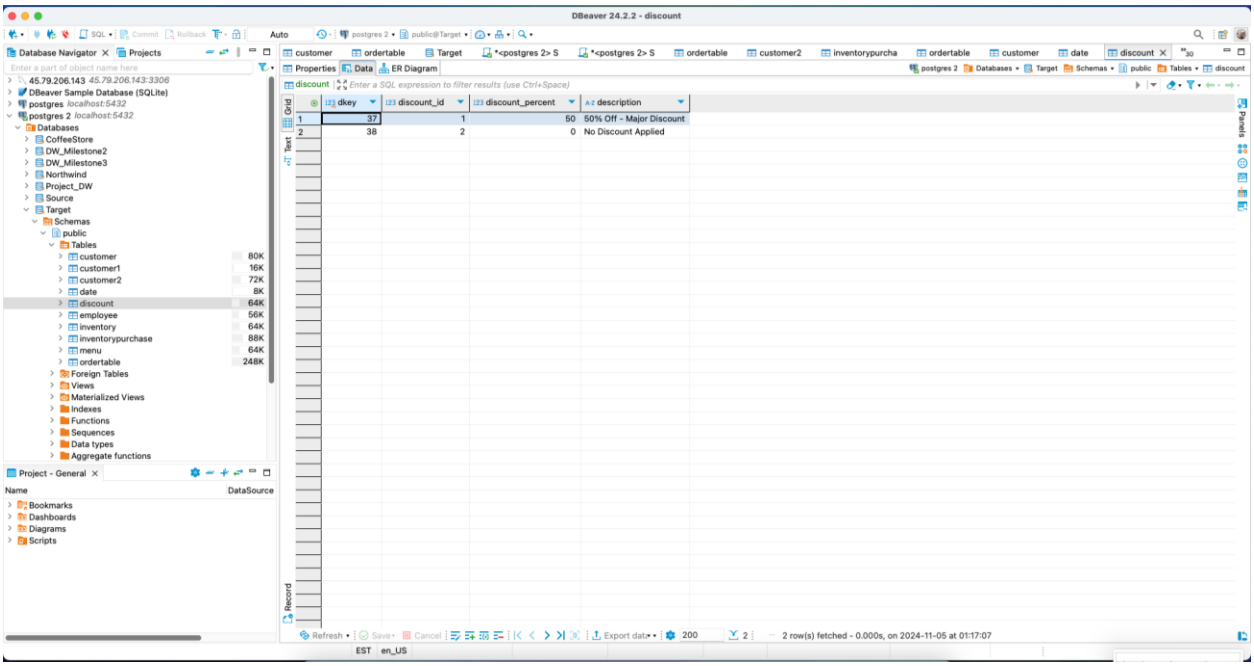


Data populated in the Data Warehouse with the Surrogate Keys:

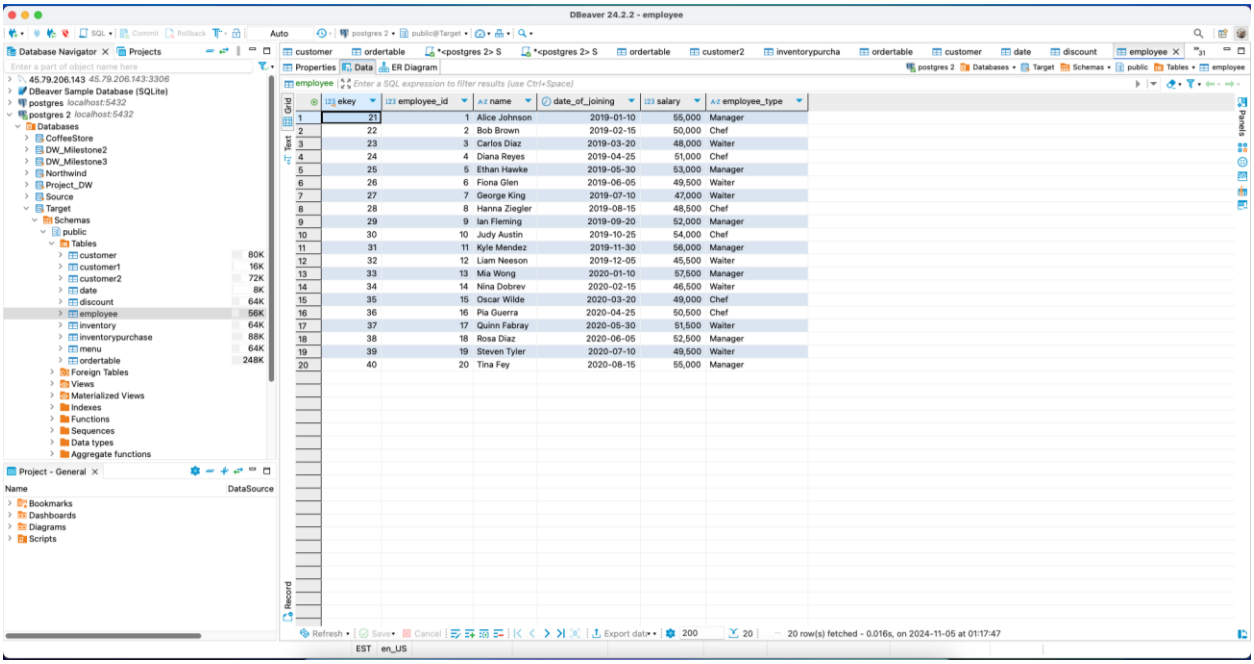
- **Customer: Surrogate Key → ckey**



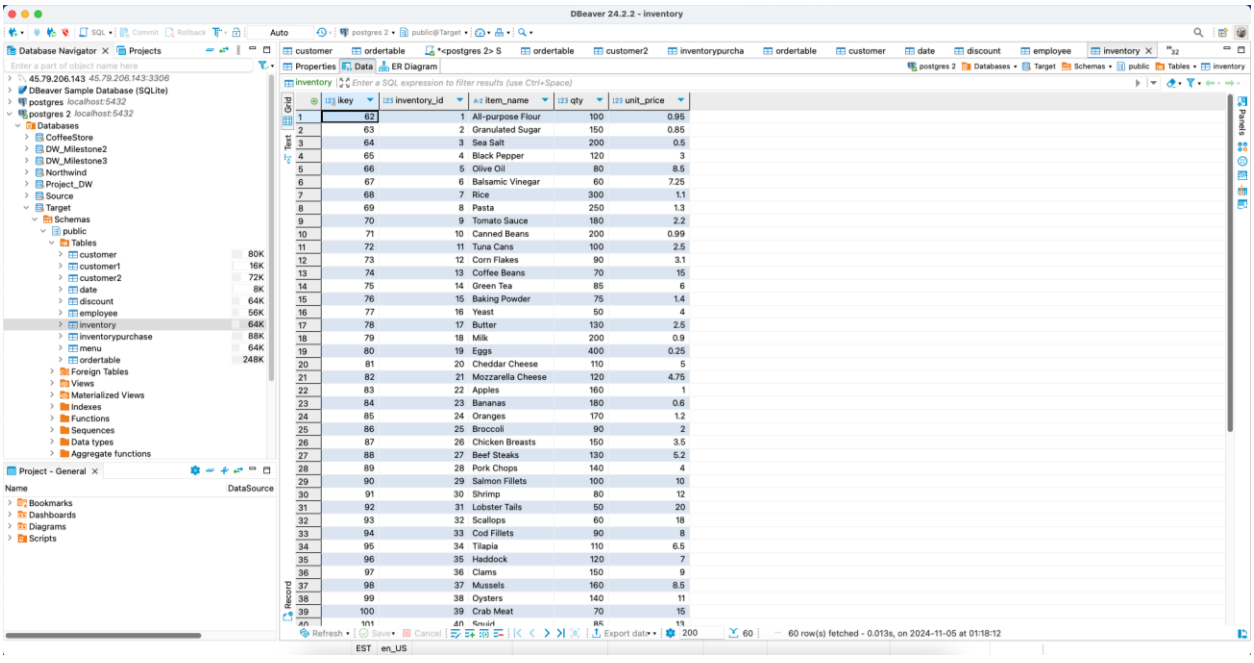
- **Discount: Surrogate Key → dkey**



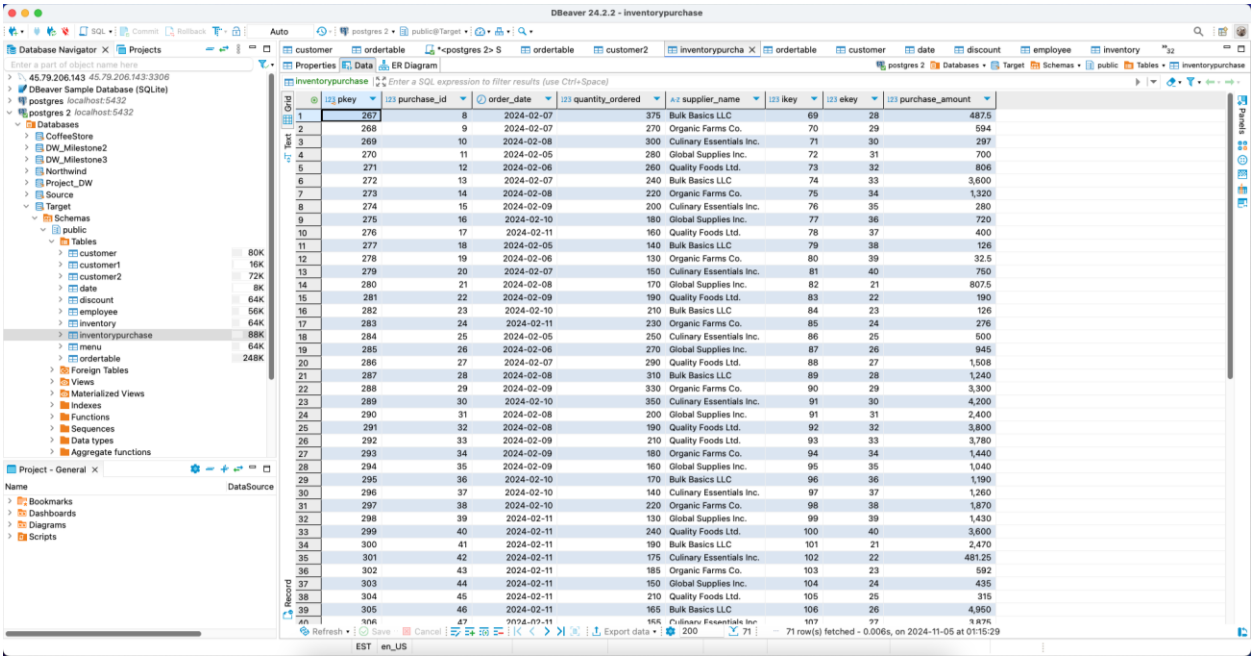
- **Employee: Surrogate Key → ekey**



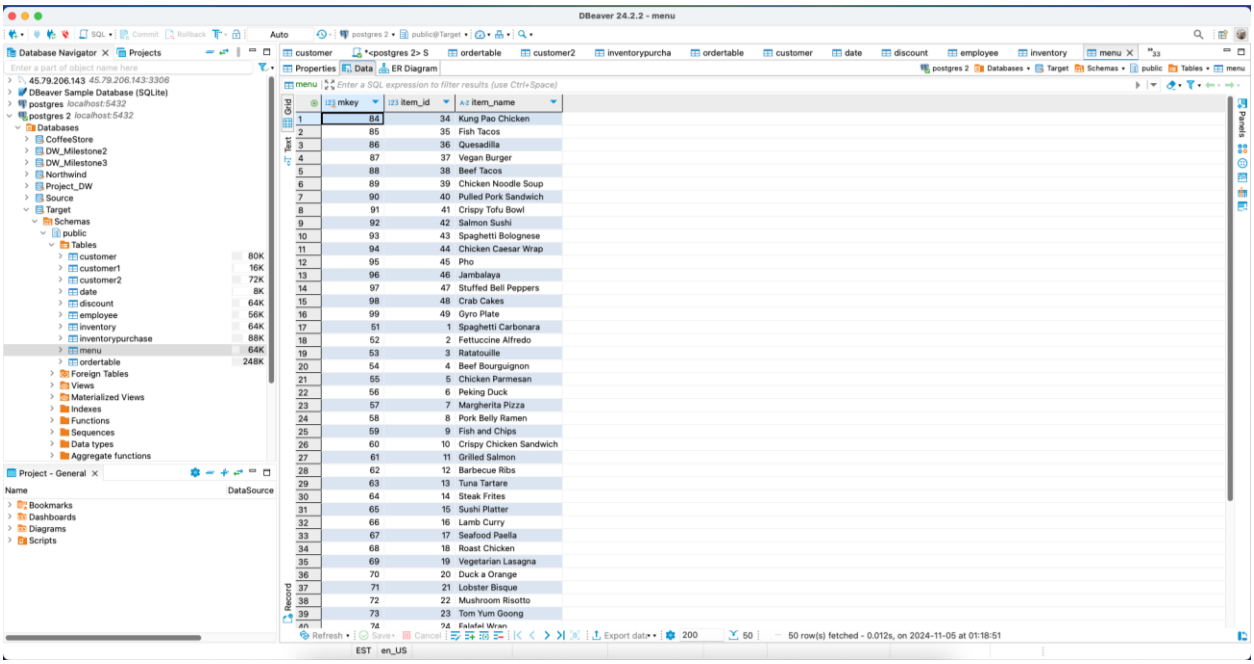
- **Inventory: Surrogate Key → ikey**



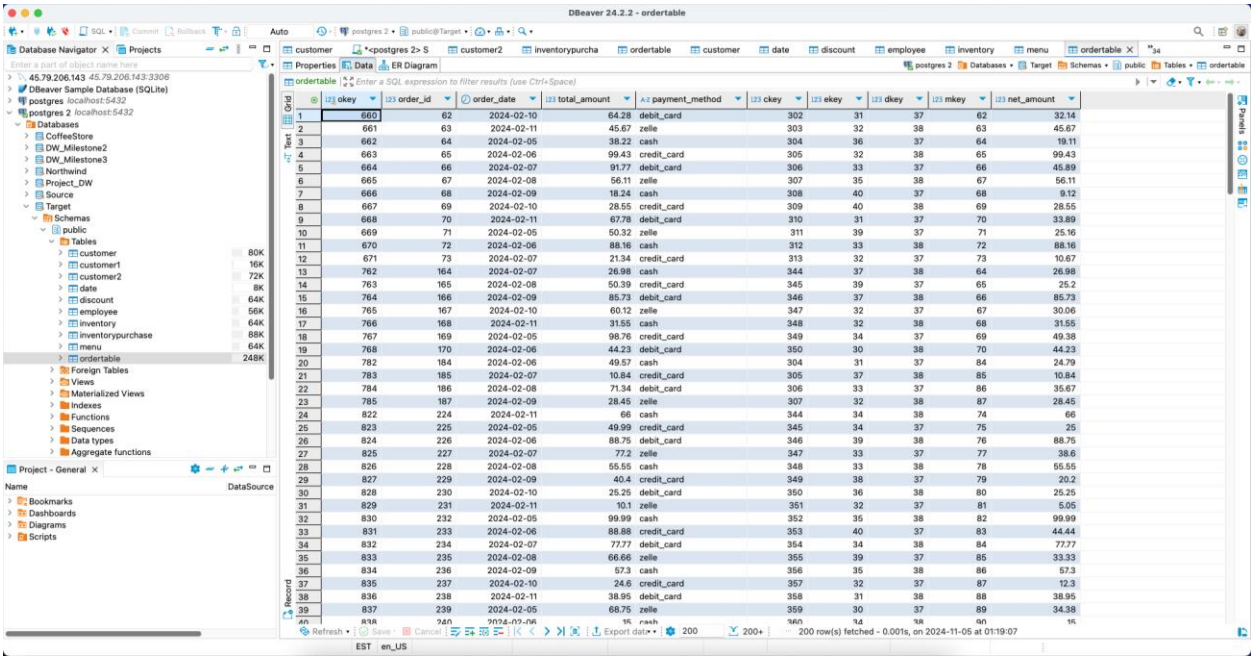
- **Inventory purchase: Surrogate Key → pkey**



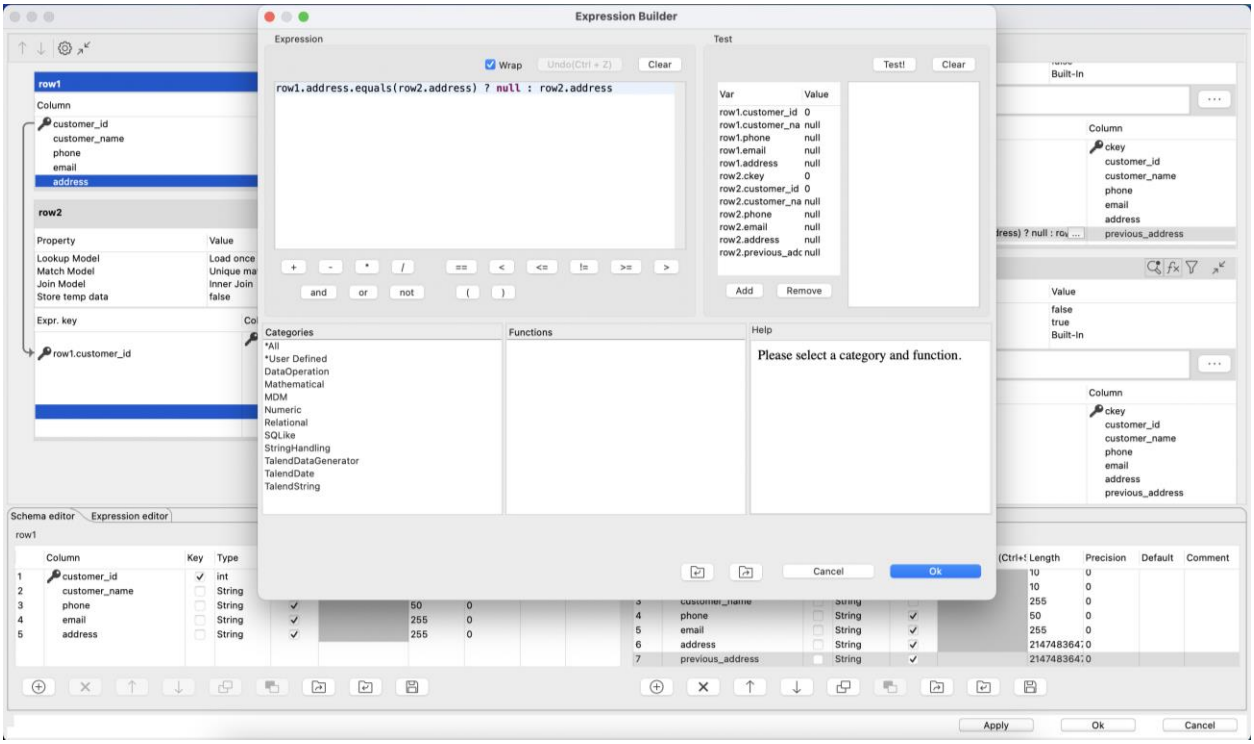
- **menu: Surrogate Key → mkey**

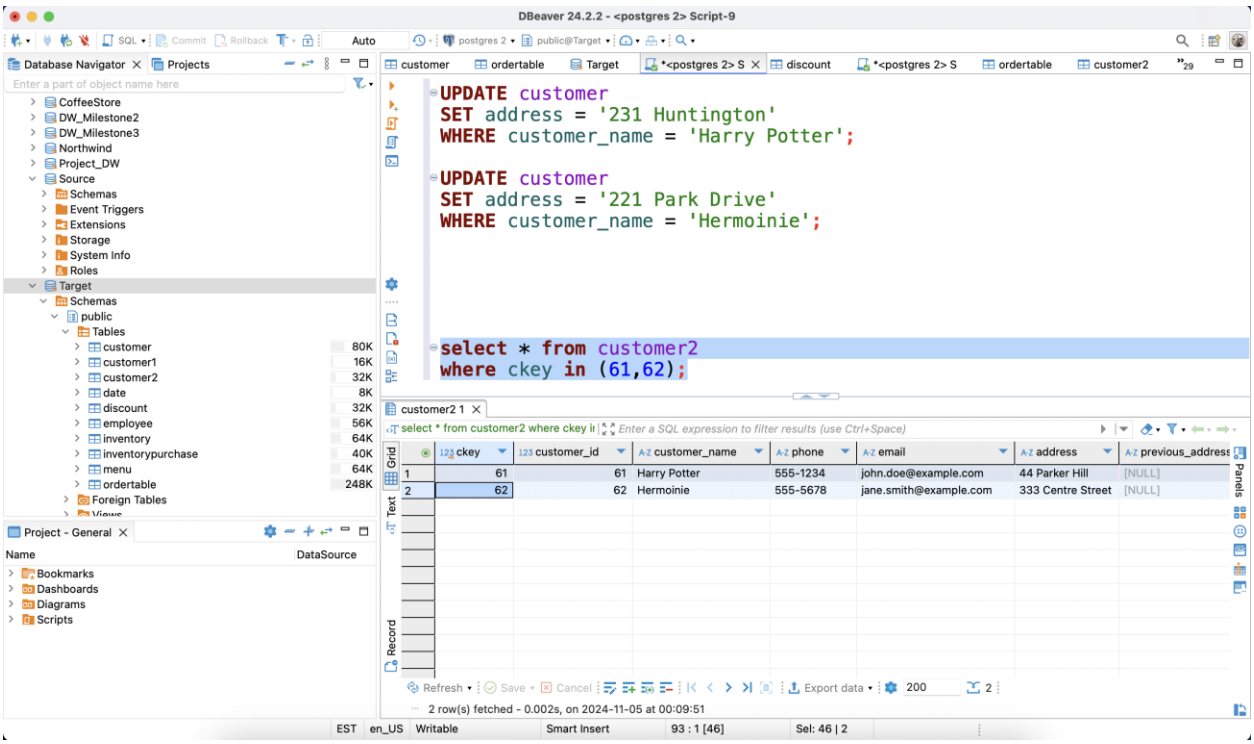


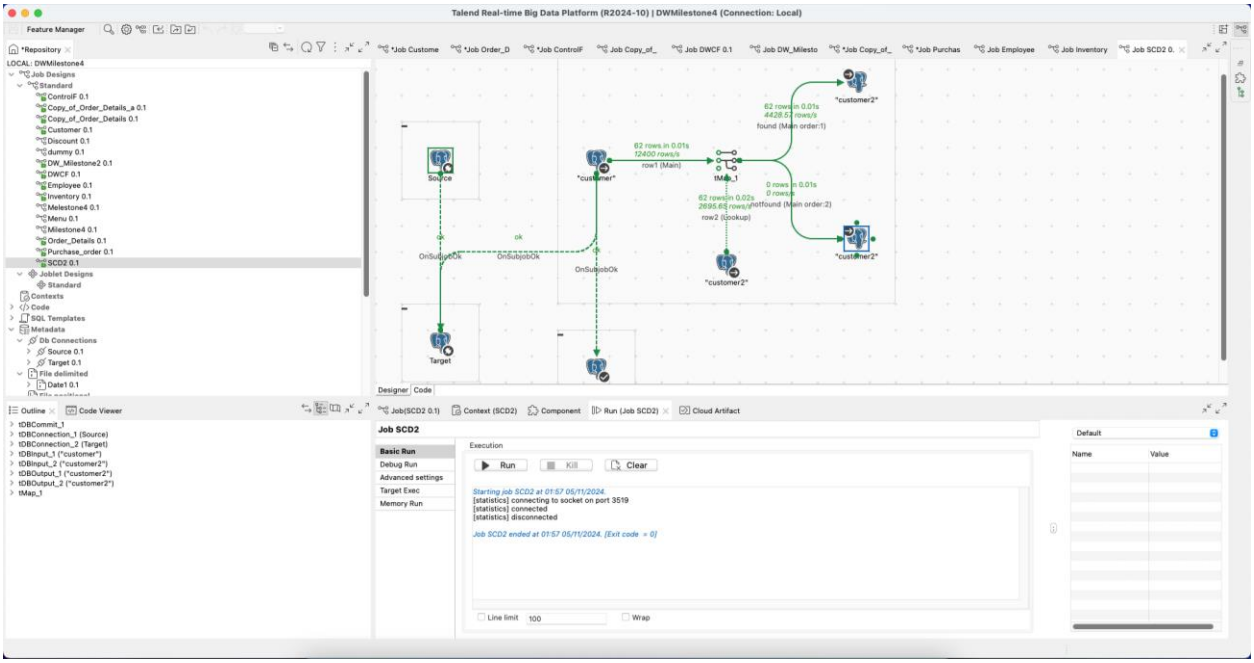
- **order: Surrogate Key → okey**

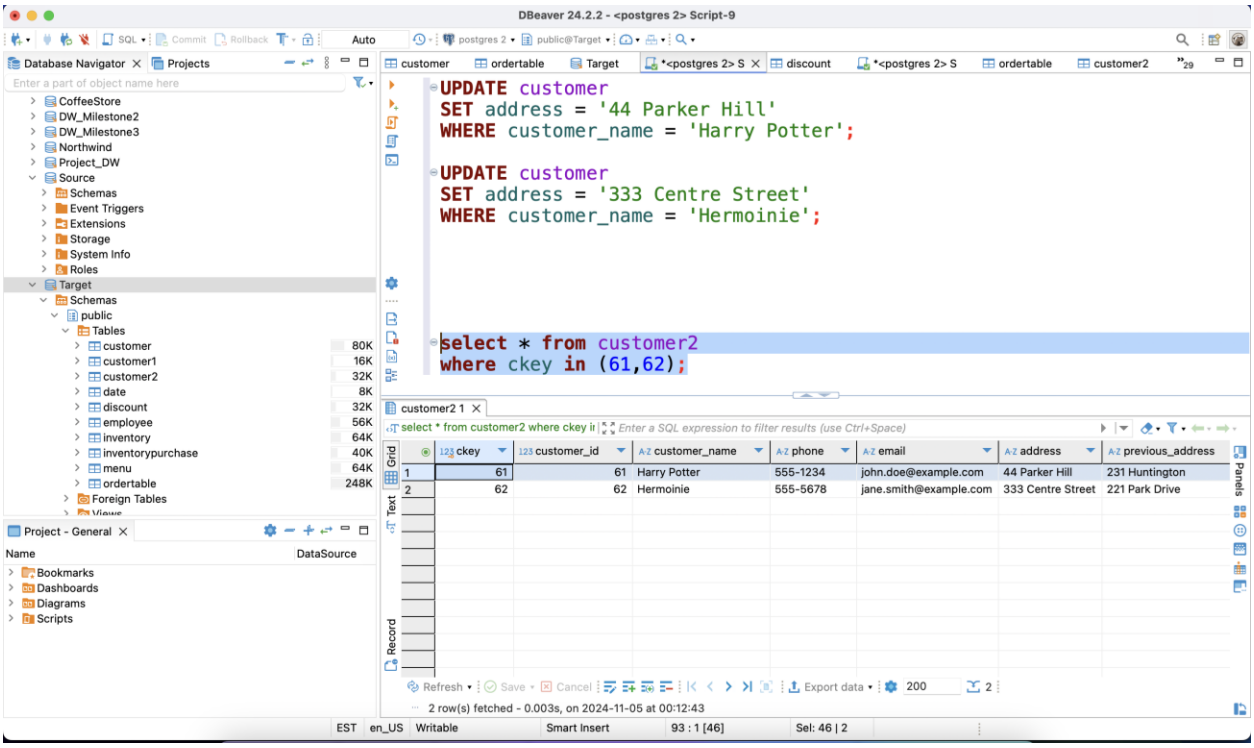


Implemented SCD type 3



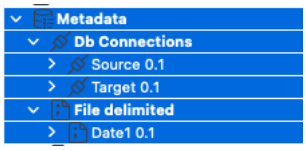




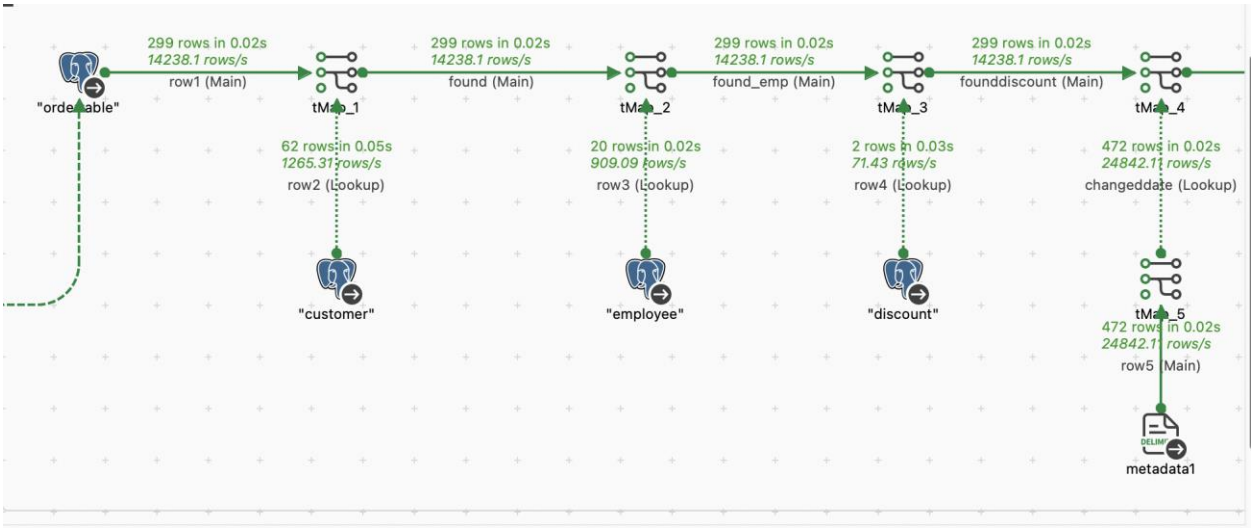


After updating the Address of customer and running the customer job, the **previous_address is being updated in the Data Warehouse**.

Used Multiple source of Data:

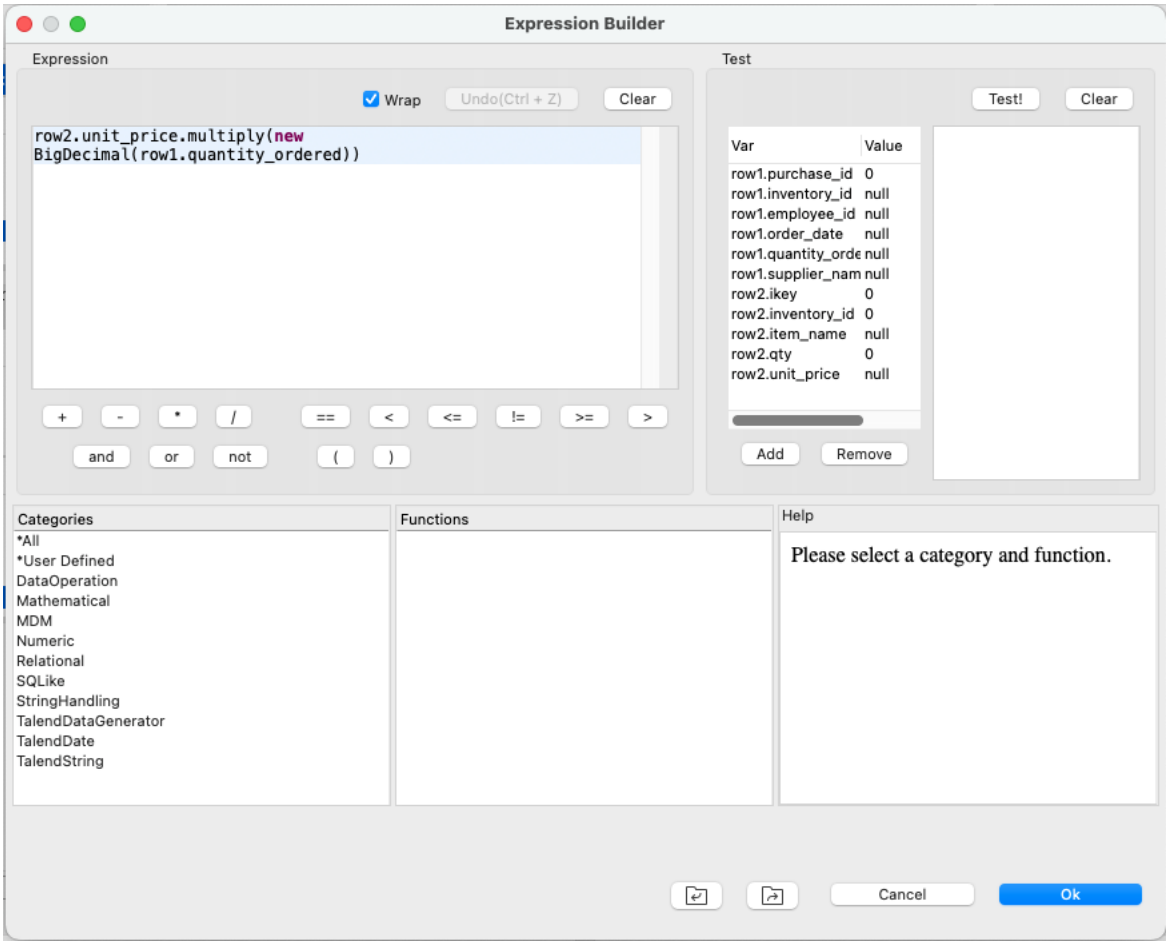


	A	B
1	Date	
2	1/17/24	
3	1/18/24	
4	1/19/24	
5	1/20/24	
6	1/21/24	
7	1/22/24	
8	1/23/24	
9	1/24/24	
10	1/25/24	
11	1/26/24	
12	1/27/24	
13	1/28/24	
14	1/29/24	
15	1/30/24	
16	1/31/24	
17	2/1/24	
18	2/2/24	

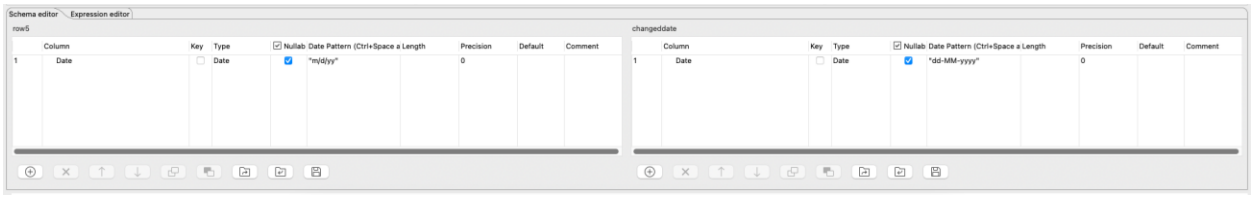


Transformations:

- While creating a calculated measure, type casted quantity ordered from integer to bigInteger to perform appropriate calculation



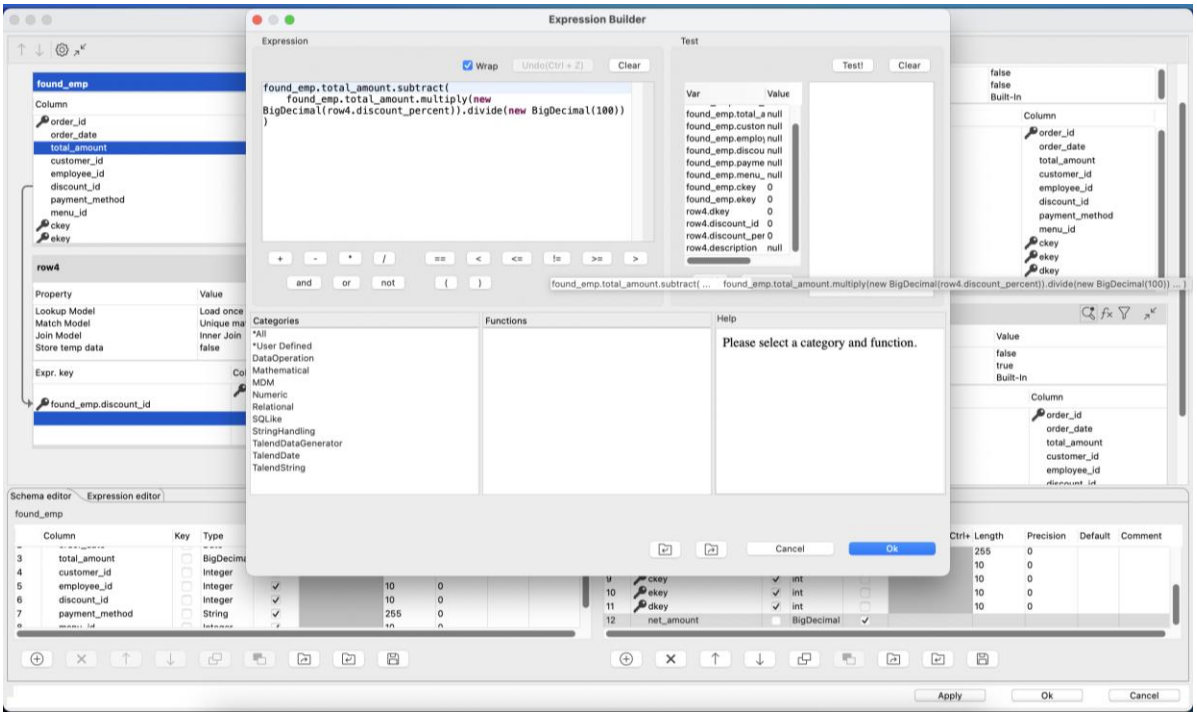
- Changed the format of date for lookup:



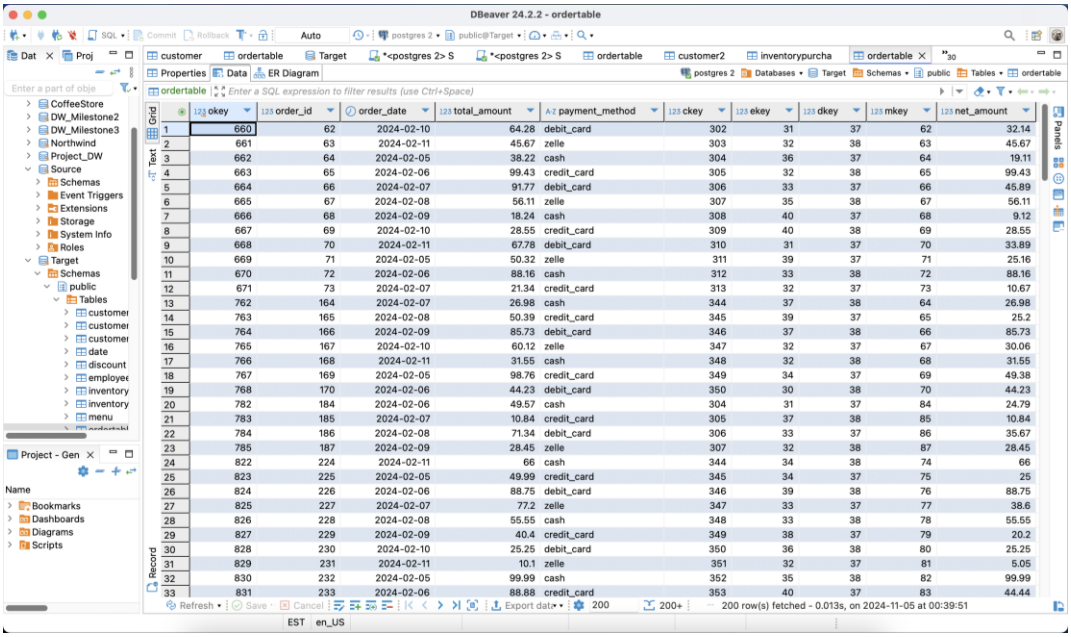
Calculated Measures:

- **net_amount**

This value is being calculated in the order_detail fact table. It is used to calculate the net bill amount once the discount is applied.

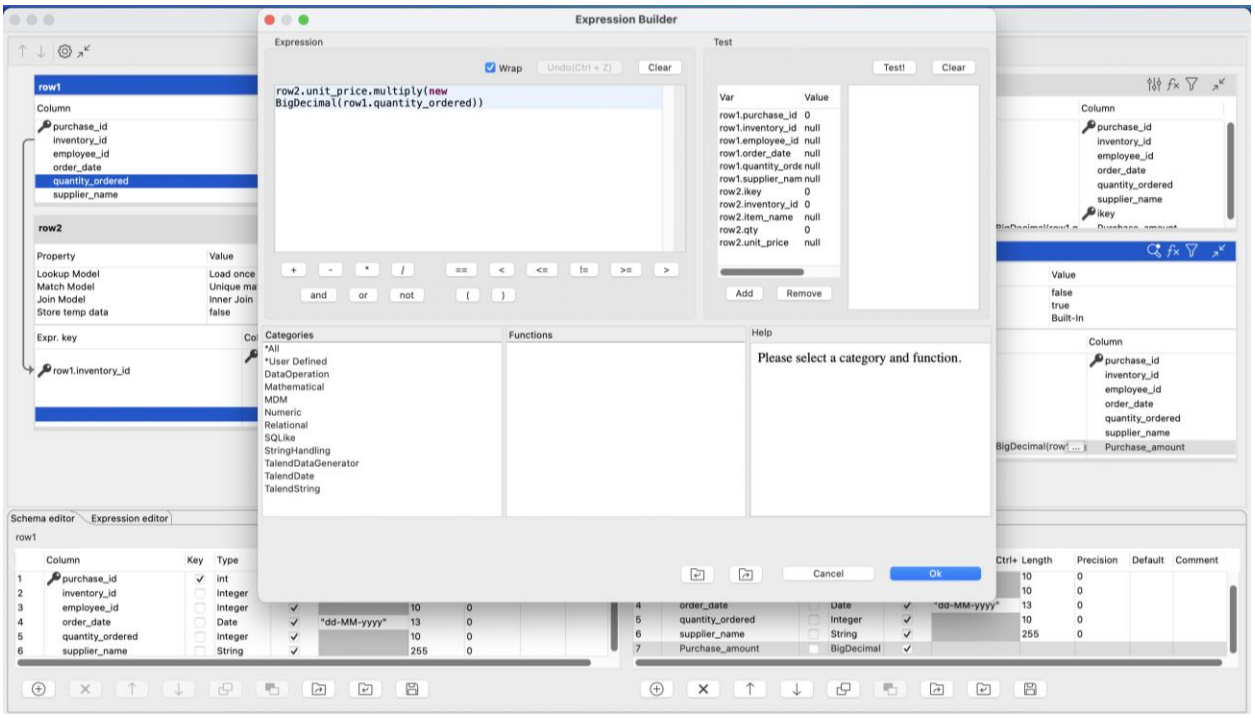


Net_amount being added to the order_details(Target Fact Table):

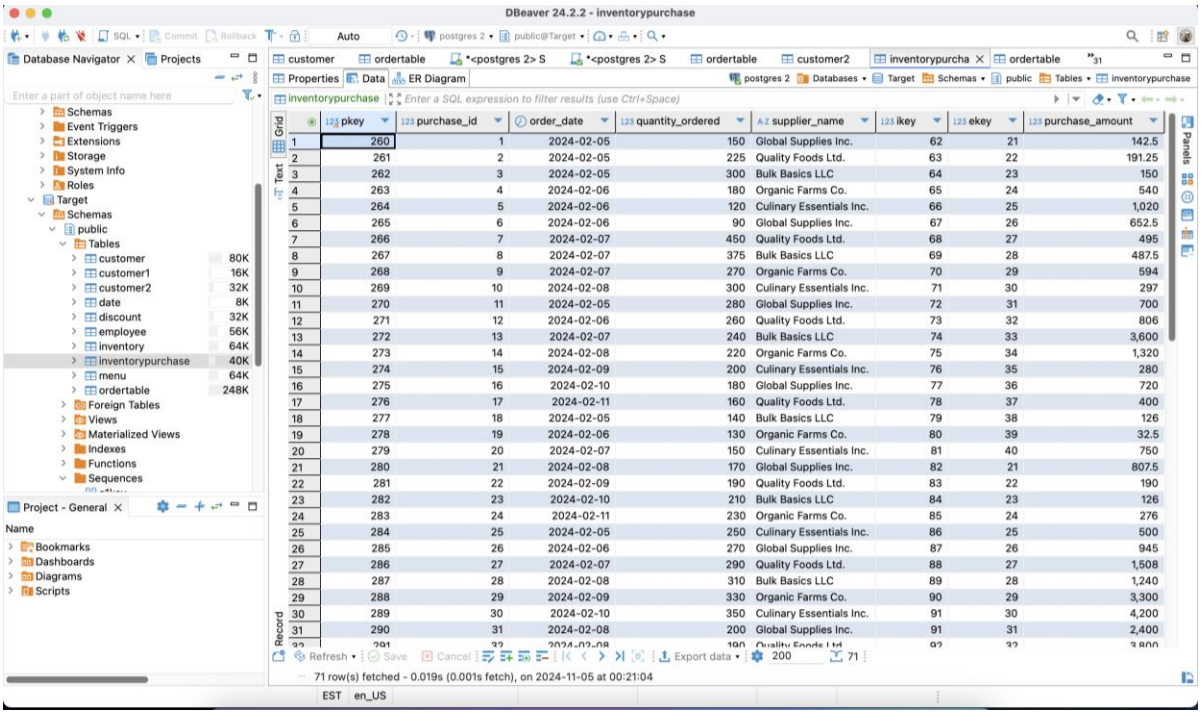


- **purchase_amount**

The purchase amount is being calculated in the inventory purchase table. This measure is being calculated using the unit price being referred from the inventory table and quantity ordered referred from the fact table. This measure signifies the amount being spent on each inventory order.



- purchase_amount being added to the purchase inventory table:



Milestone 5

Problem Statement

Sail Foods needs to design and implement a comprehensive database system to manage the culinary operations for 5,000+ sailors on the USS Theodore Roosevelt, serving over 15,000 meals per day. The system must efficiently handle inventory management, meal planning, food preparation, serving logistics, and waste management while ensuring food safety, nutritional requirements, and operational efficiency in the unique environment of a Nimitz-class aircraft carrier.

Database Definition:

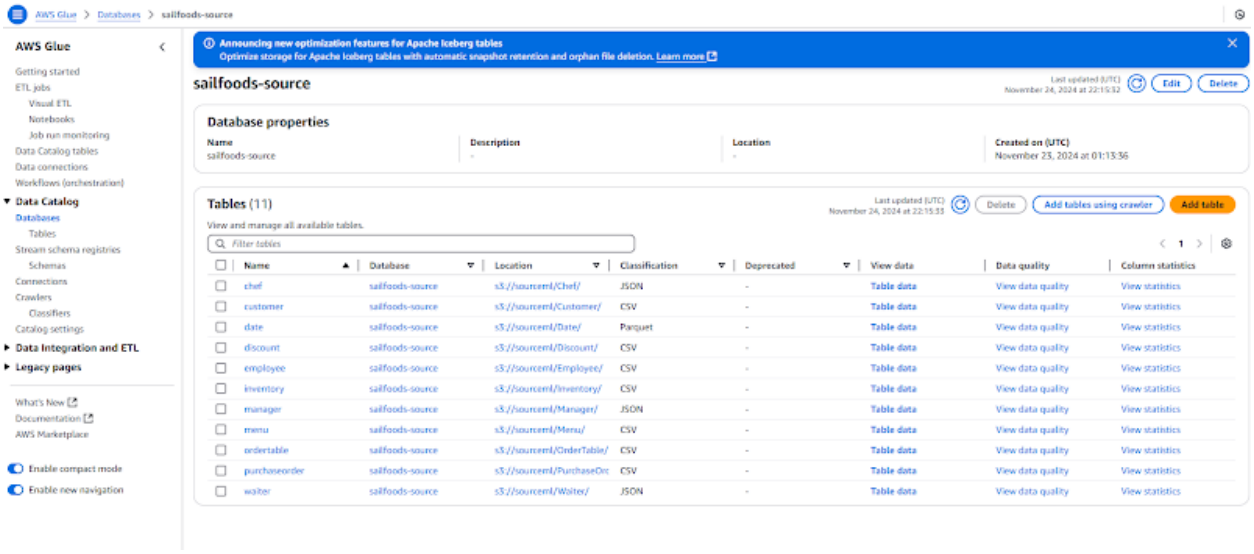
12. **Customer**: Represents customers who place orders.
    - Attributes: customer_id (PK), customer_name, phone, email, address
13. **Order**: Represents food orders made by customers.
    - Attributes: order_id (PK), order_date, total_amount, customer_id (FK), payment_id, employee_id (FK), discount_id (FK)
14. **Payment**: Represents payments made for orders.
    - Attributes: payment_id (PK), order_id (FK), payment_date, payment_type, amount
15. **Discount**: Represents discounts available for orders.
    - Attributes: discount_id (PK), discount_percentage, description
16. **Menu**: Contains all food items available.
    - Attributes: item_id (PK), item_name, price, employee_id (FK)
17. **Employee**: Represents employees, which include Waitrs, Managers, and Chefs.
    - Attributes: employee_id (PK), name, Date_of_joining, salary
18. **Inventory**: Represents the food inventory.
    - Attributes: inventory_item_id (PK), item_name, qty, unit_price, Employee_id
19. **Purchase Order**: Represents purchase orders for inventory.
    - Attributes: purchase _id (PK), inventory_id (FK), order_date, quantity_ordered, supplier_name

Data Population Methodology:

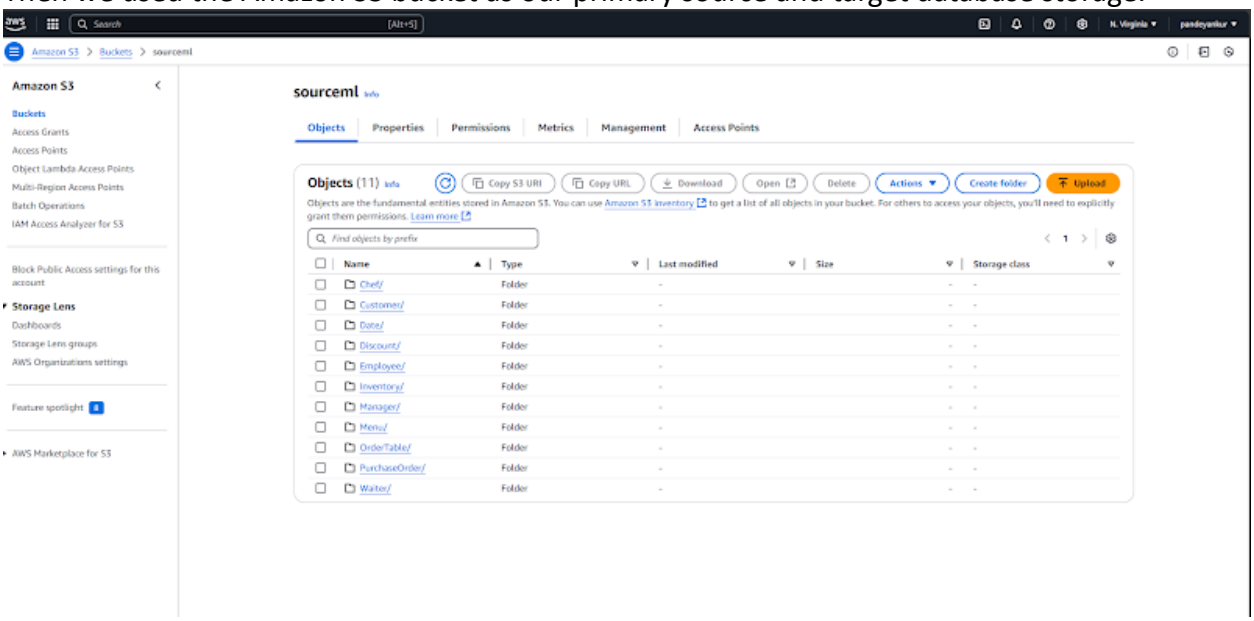
We generated our data in batches using various LLMs and first populated the static data, such as Employee, Customer, and Items. Using this static data, we then generated the transactional data, such as Amount paid, etc.

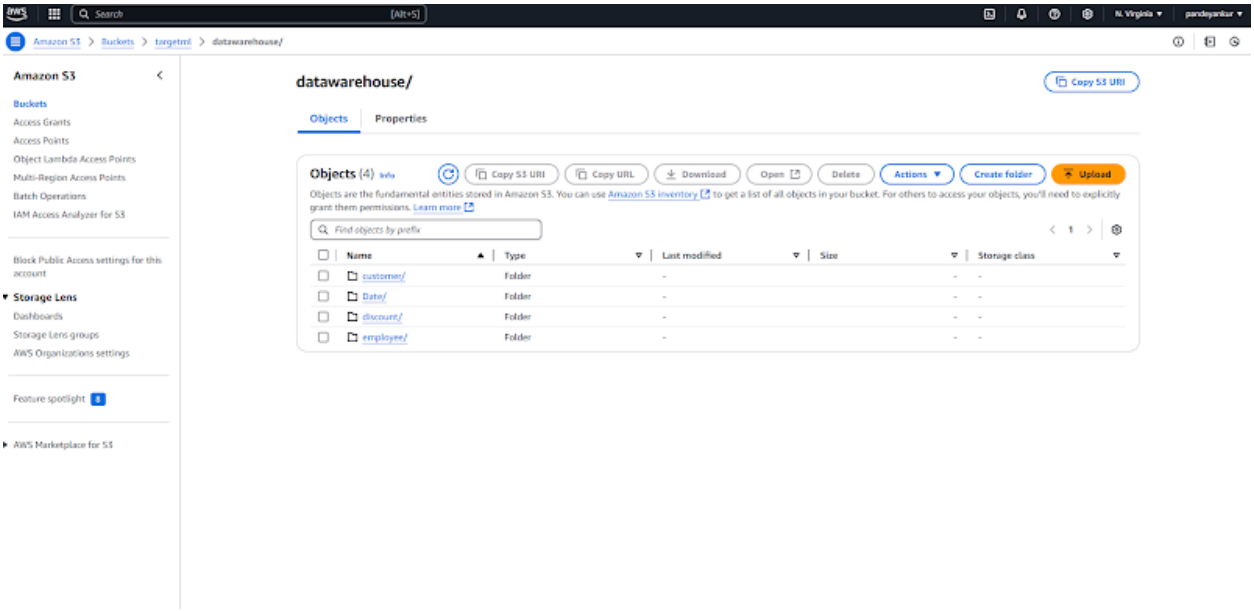
Milestone 6

Files and Data Sources.
We have used CSV, Parquet, and JSON files to populate our source data source on Amazon S3 via Crawler.

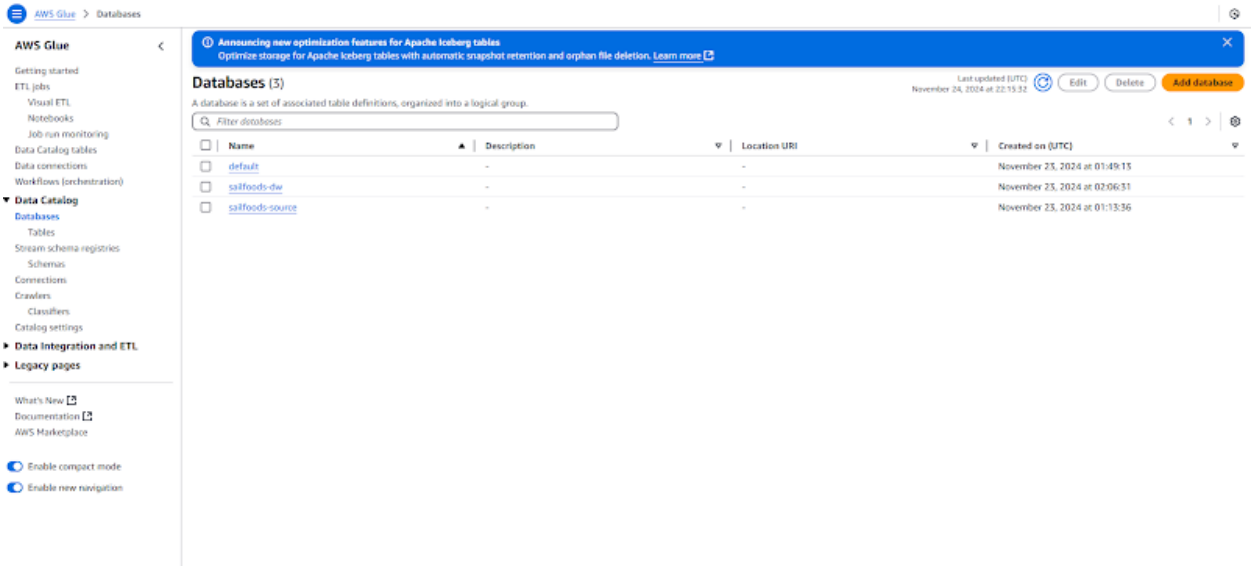


Then we used the Amazon S3 bucket as our primary source and target database storage.





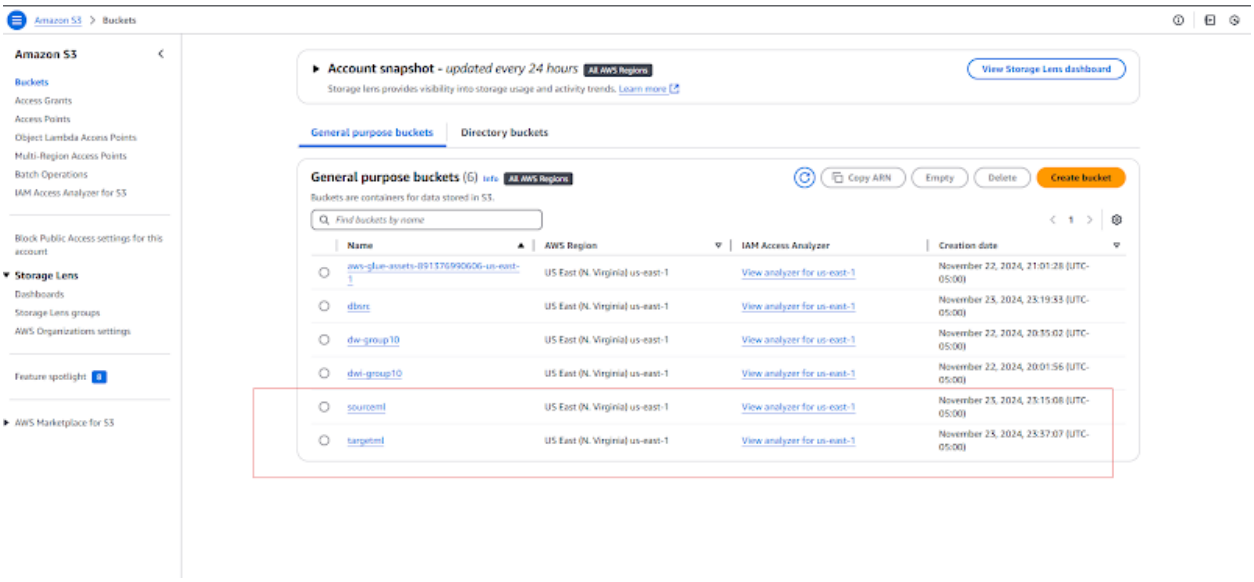
Finally, we are using AWS glue to host our source and target database.



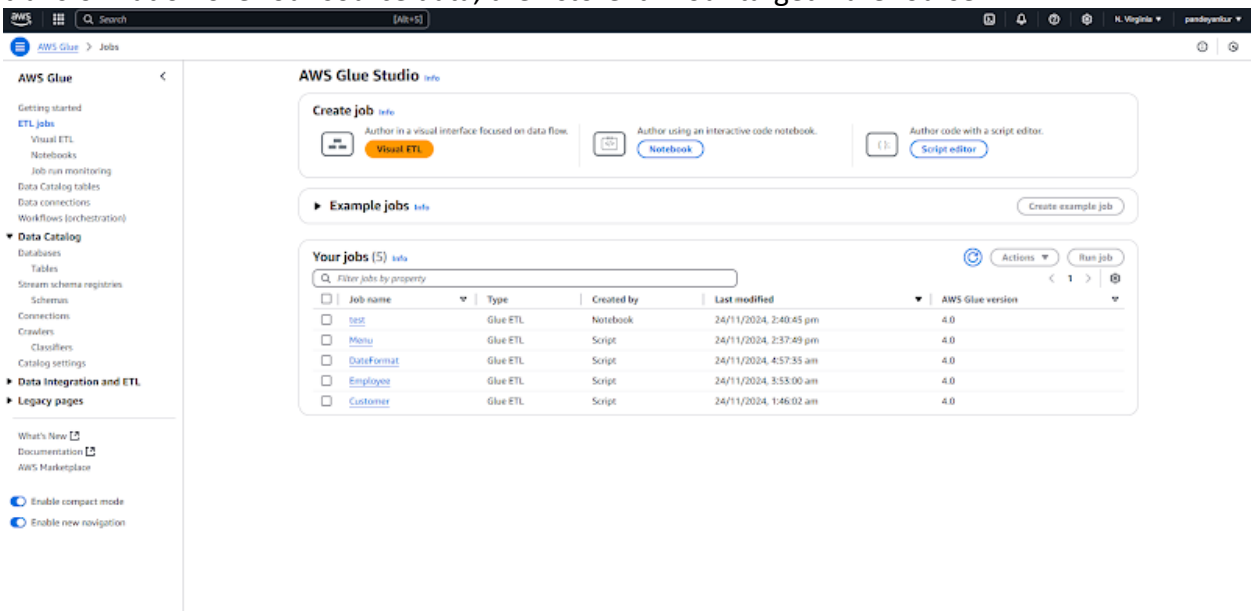
Pipeline Tools:

We are using the tools below to create, host, and orchestrate our data pipelines:

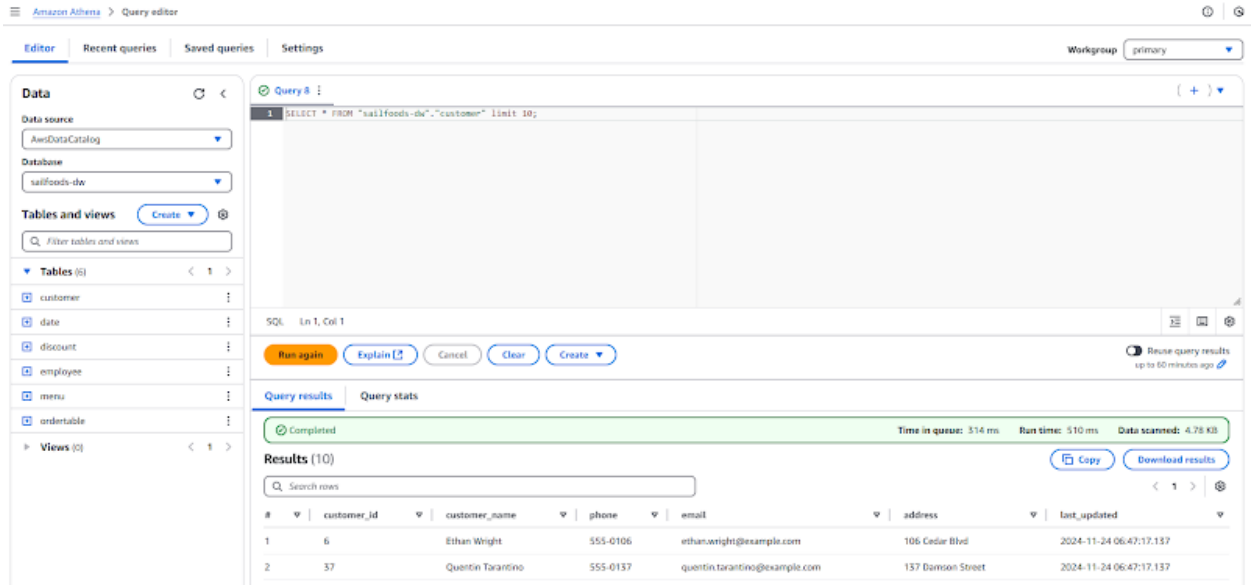
Amazon S3: We have used S3 buckets for our source and target database storage as well as storing file data, which we used to populate our source database.



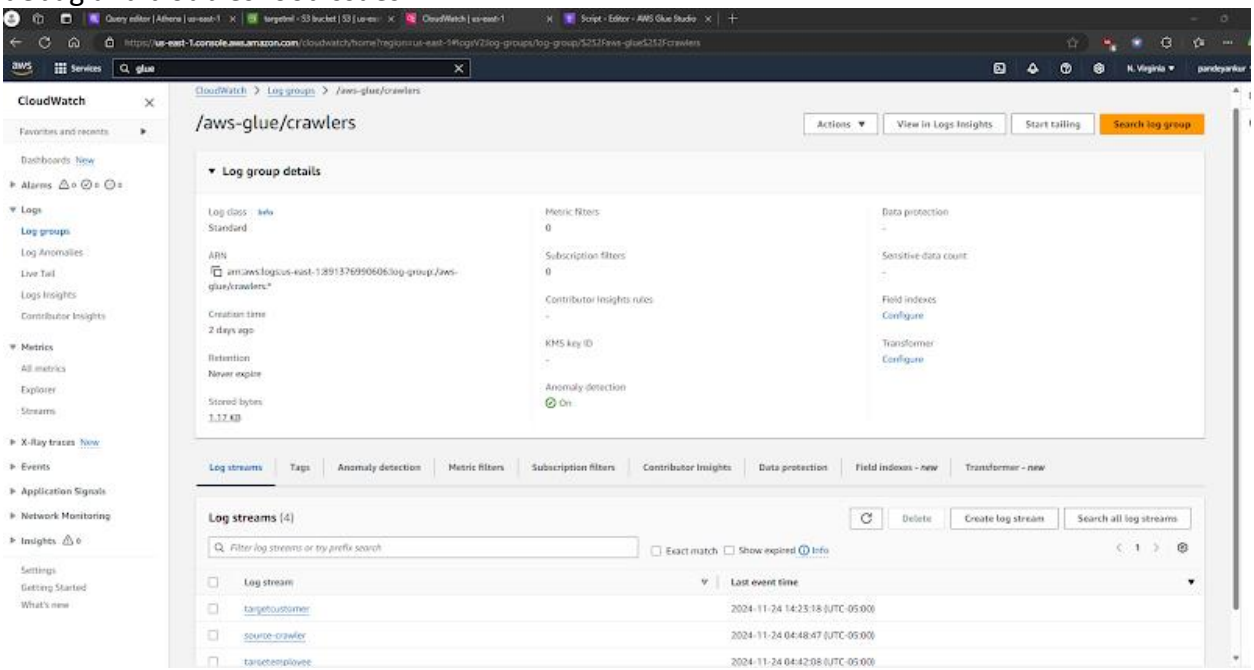
Amazon Glue: we are using Glue to host our databases and Glue Crawler to populate the database with CSV and JSON files. We are also, using glue to write ETL scripts and perform transformation over our source data, then store it in our target warehourse.



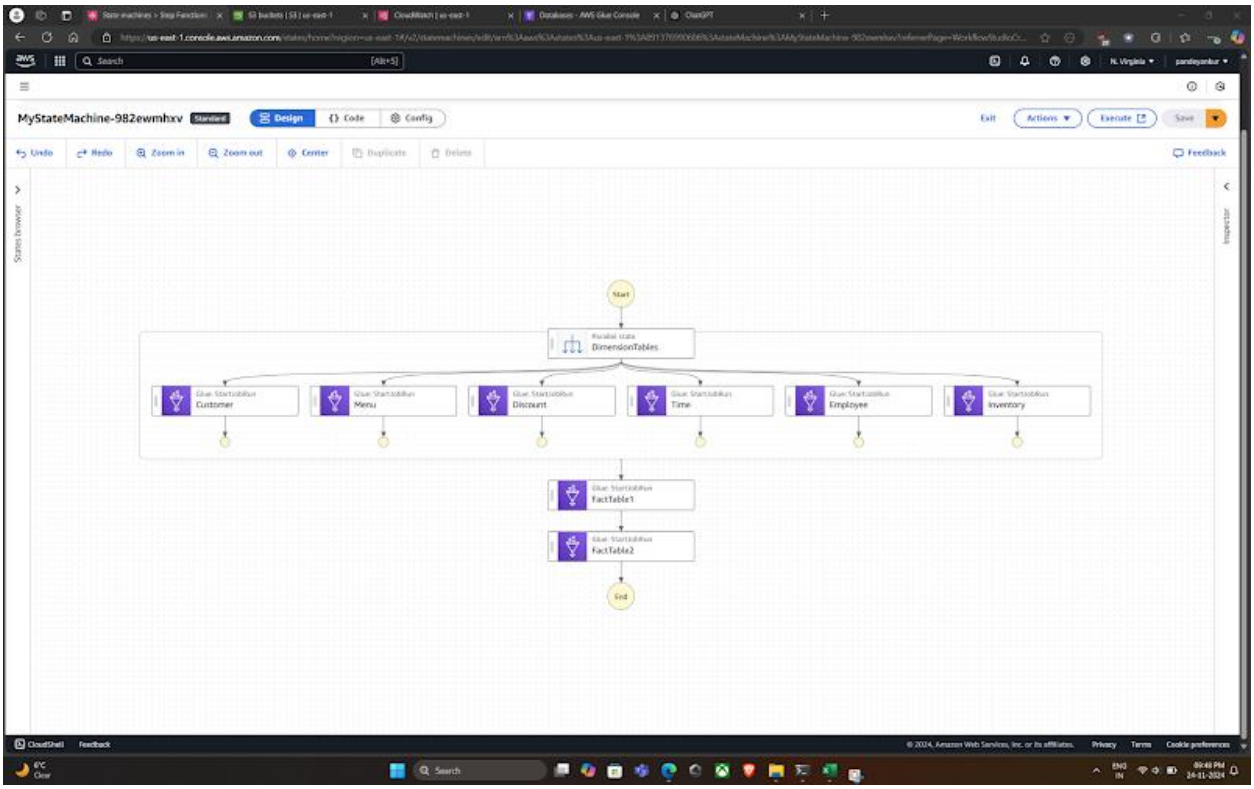
Amazon Athena: we have used Athena to create and modify our Database tables.



Amazon Cloudwatch: we are utilizing CloudWatch to monitor database, jobs, and query logs, to debug and troubleshoot issues.

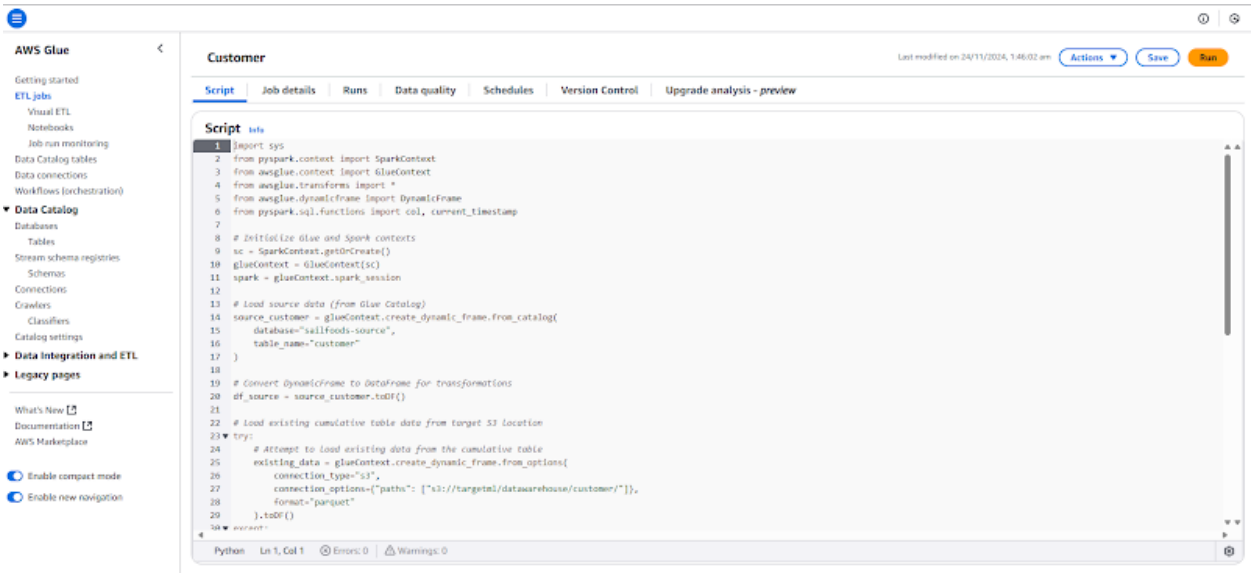


Amazon Step Function: we are using the Amazon step function to orchestrate our job workflow.

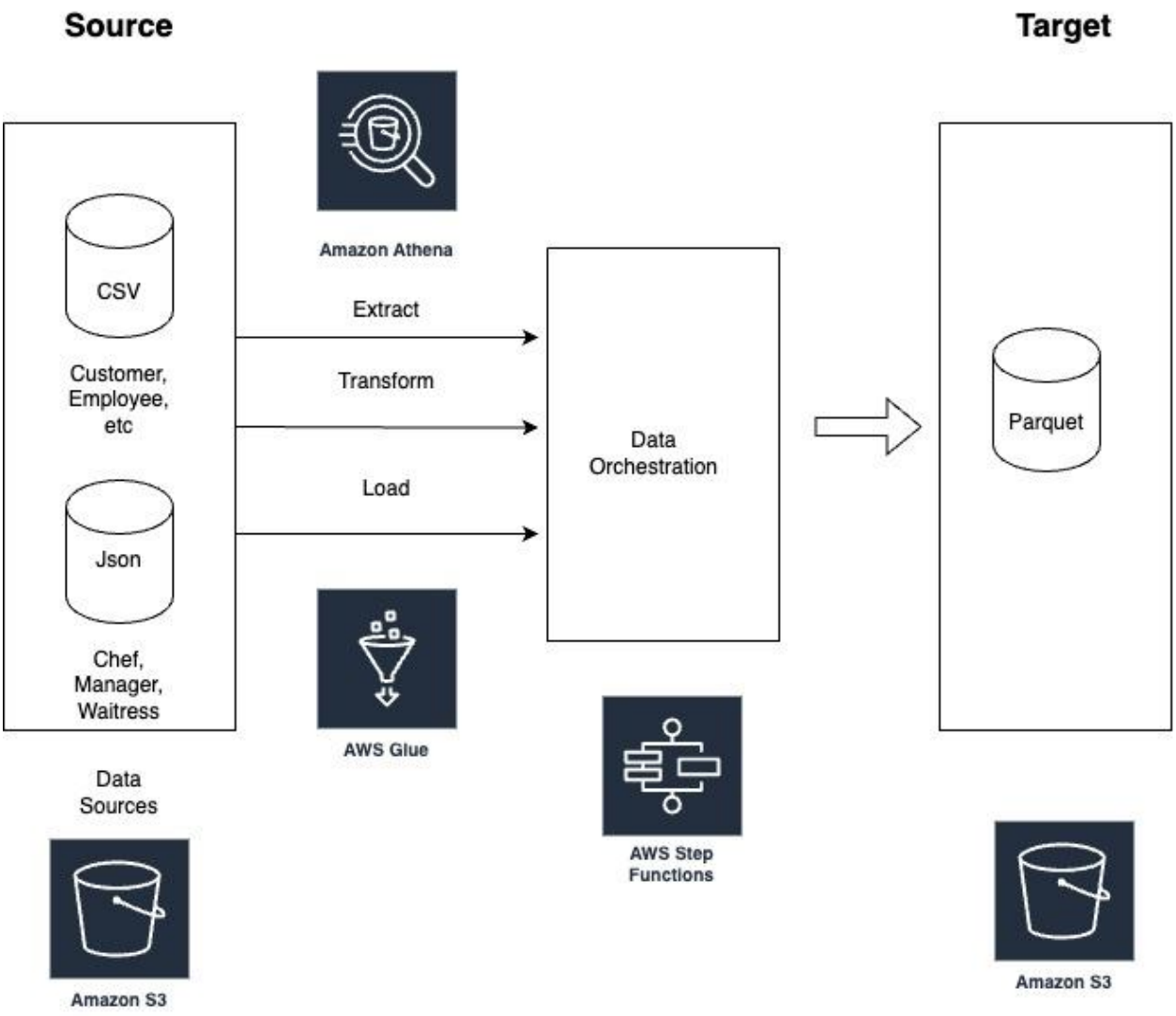


Programming Choice:

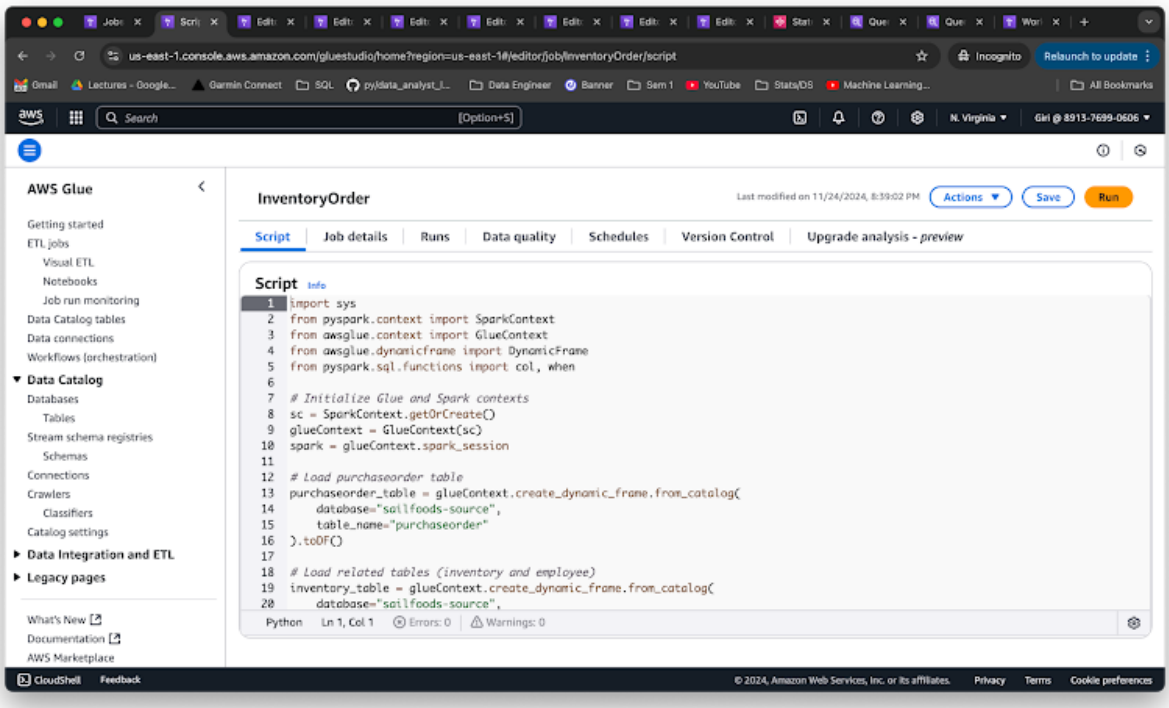
We are leveraging **Python** to write ETL scripts, which will read the data from our source tables, perform necessary transformations, and then populate the data onto our target data warehouse.



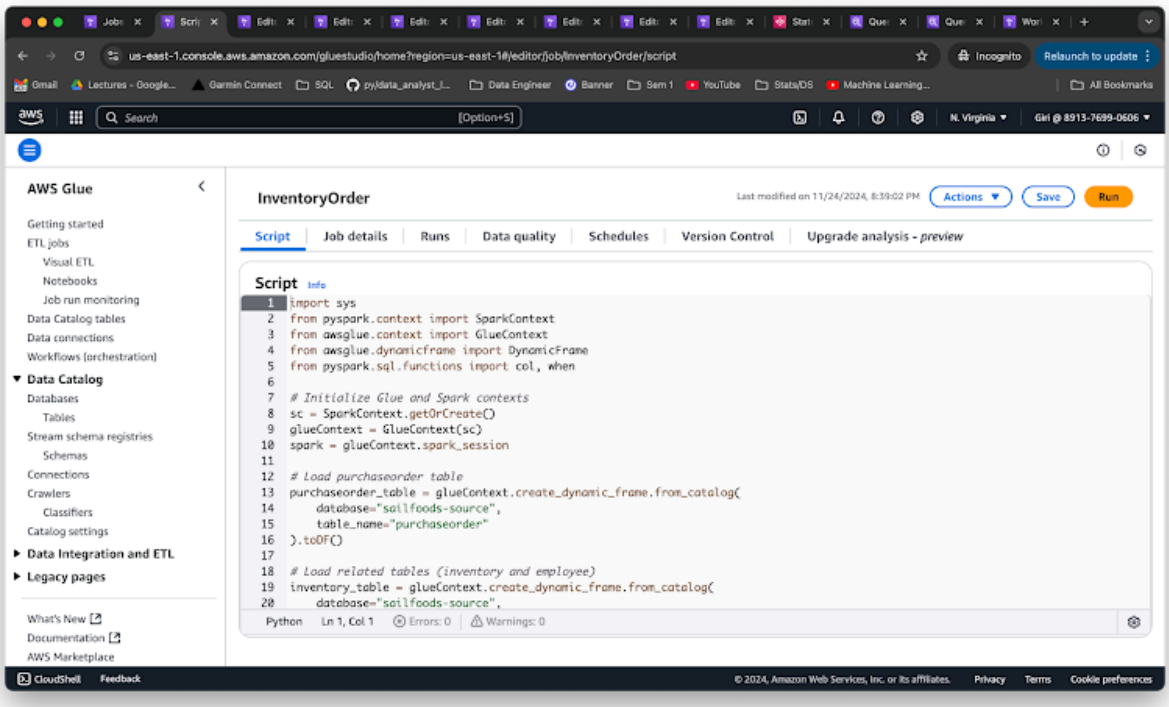
Pipeline Architecture:



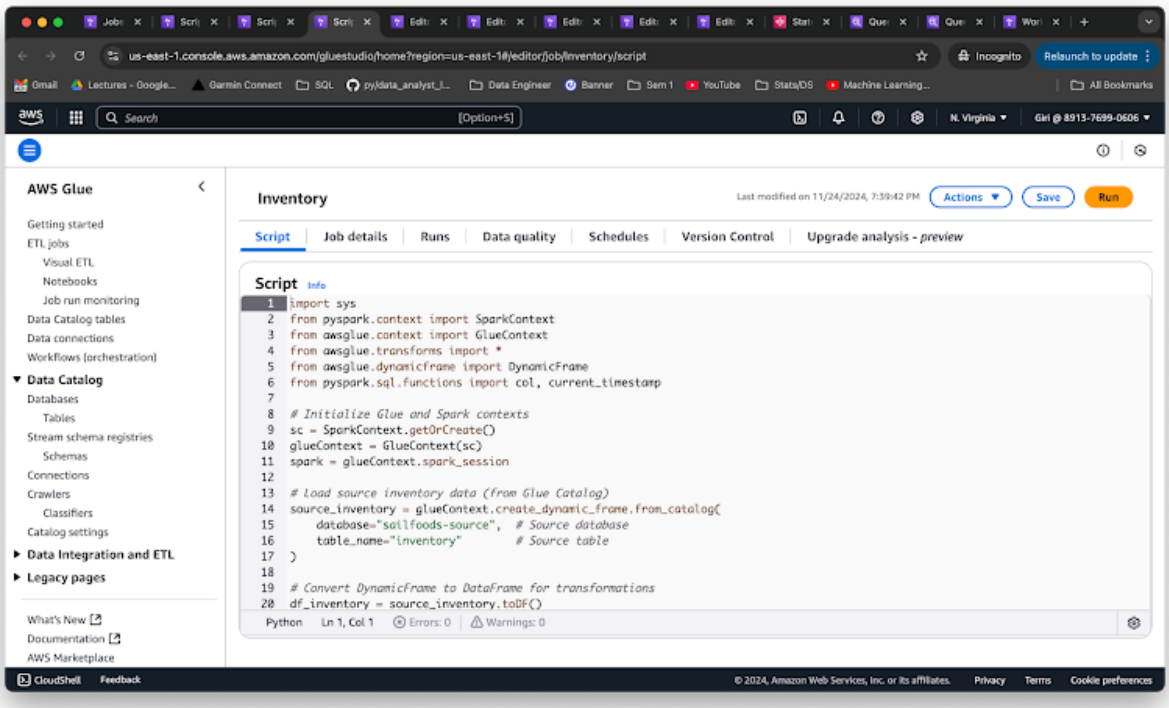
InventoryOrder Job:



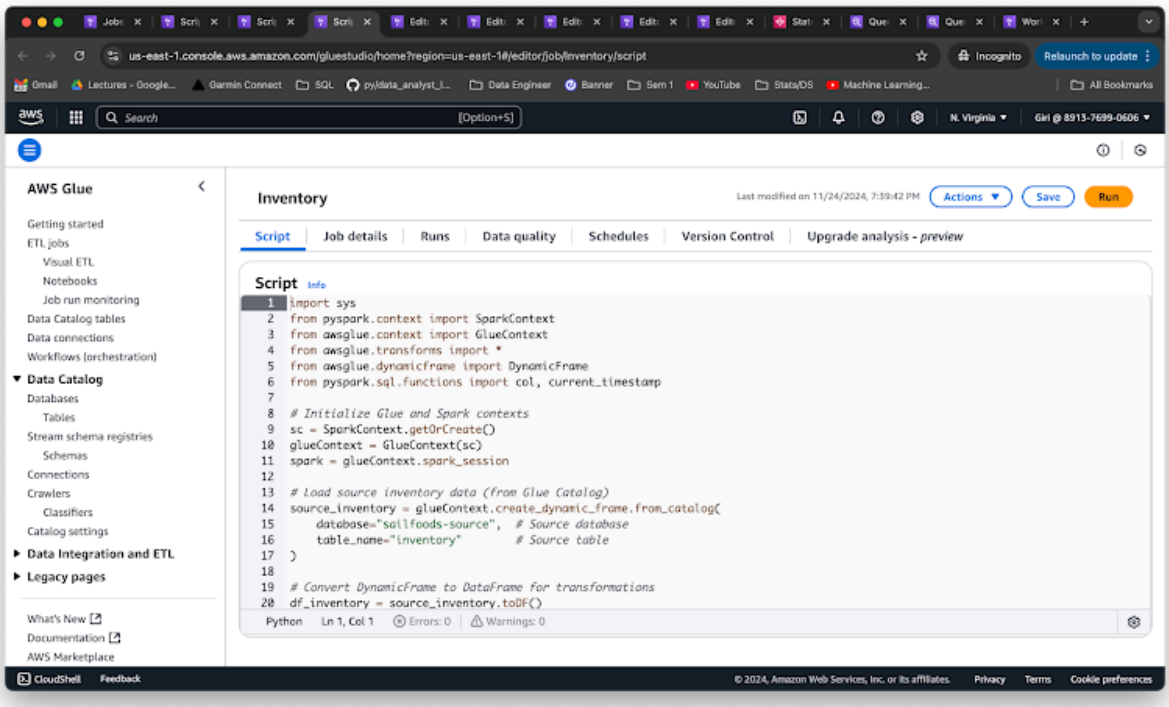
OrderTable:



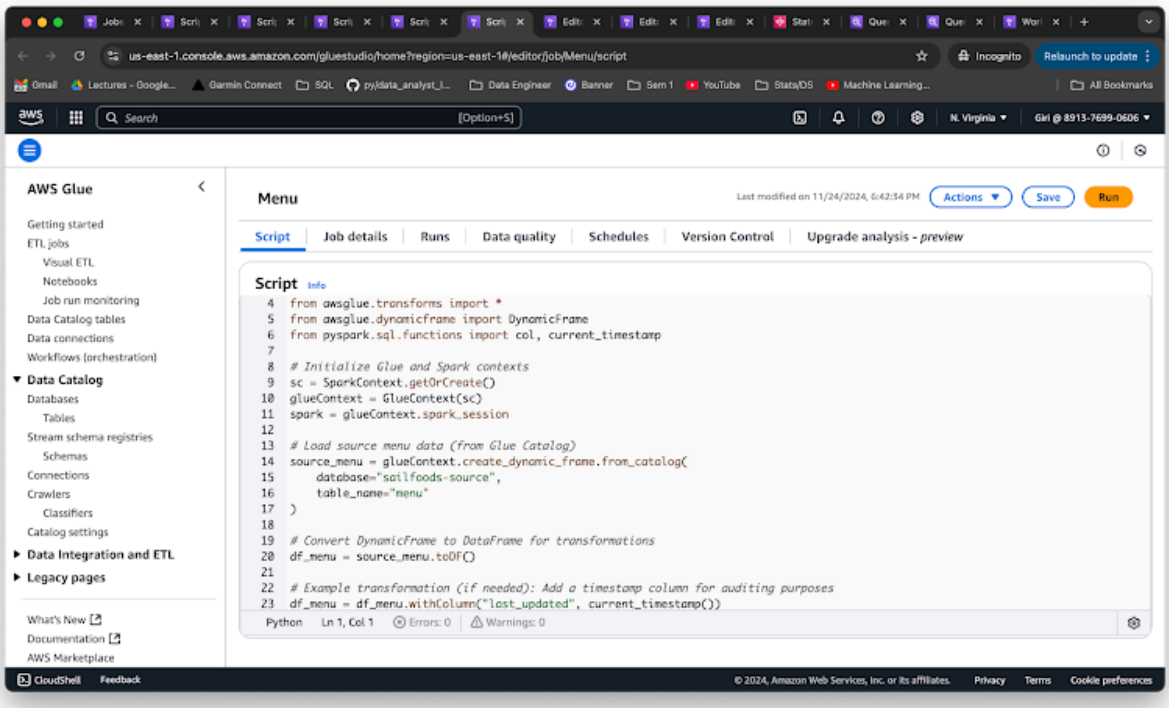
Inventory Job:



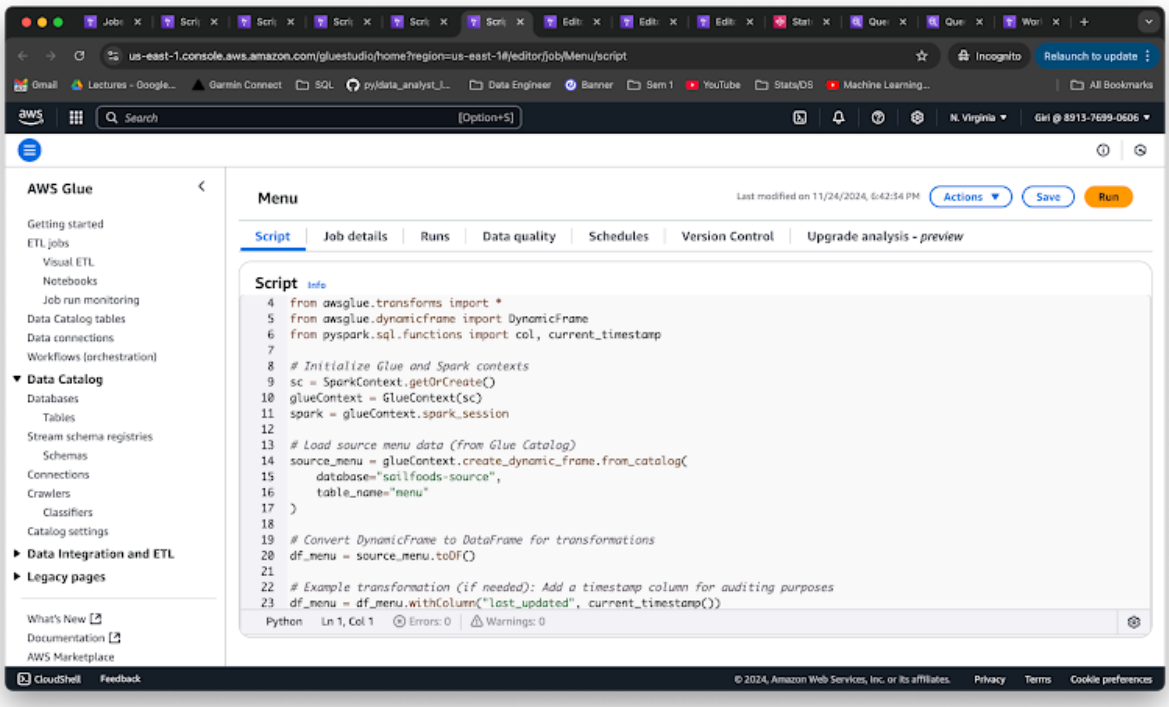
Discount Job:



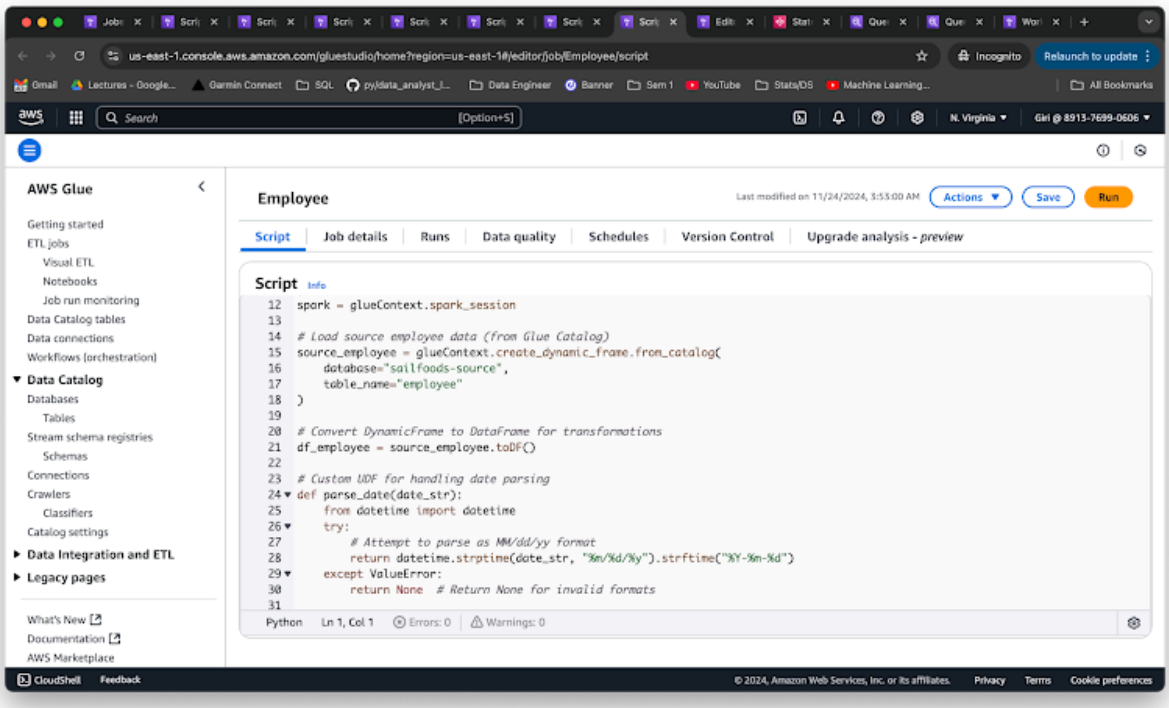
Menu Job:



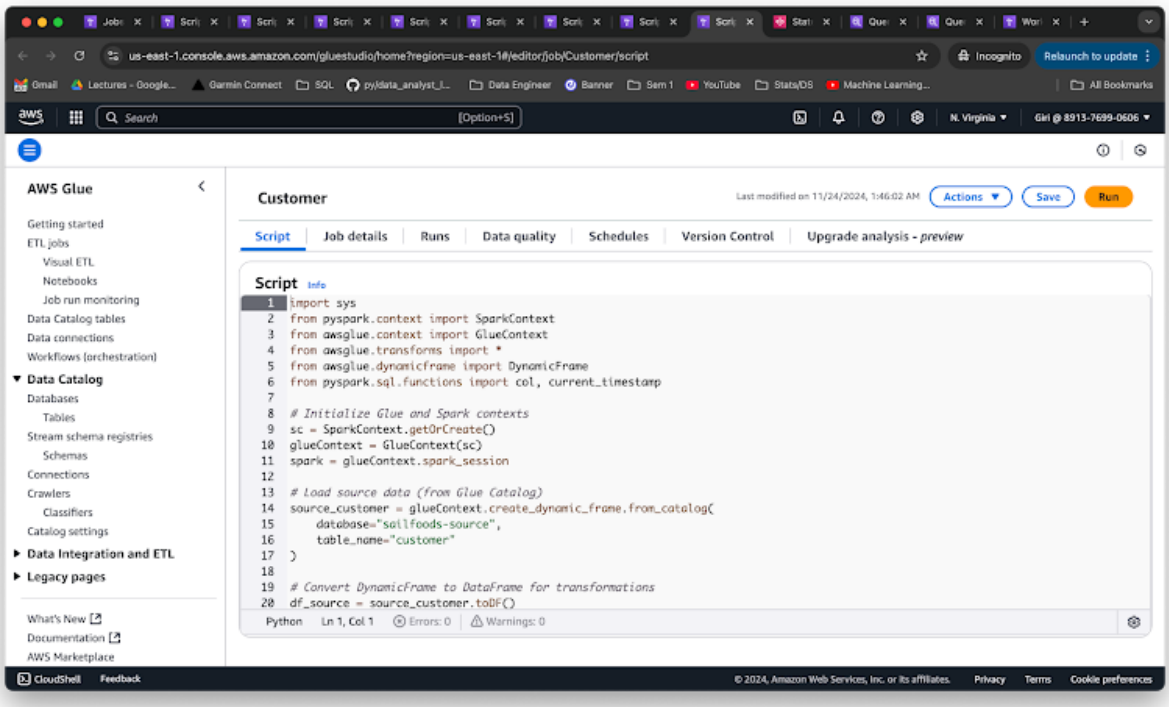
Date Job:



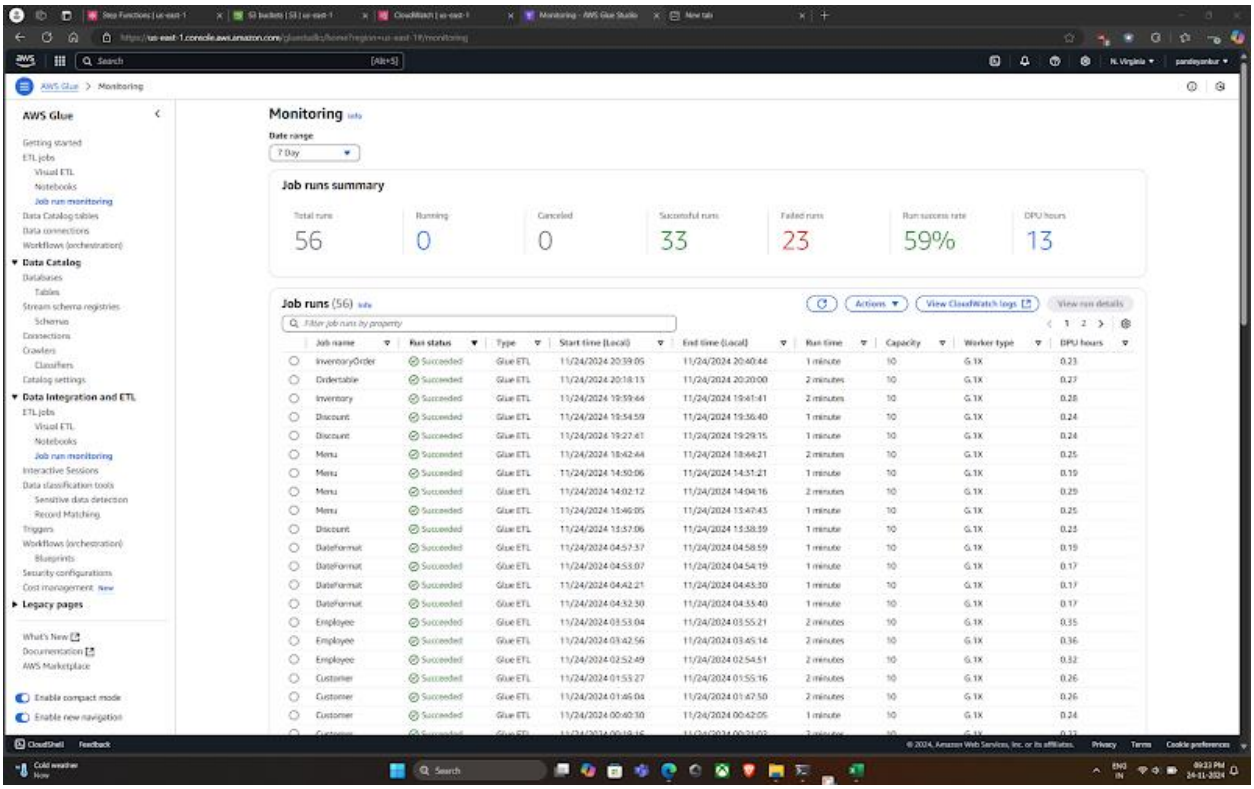
Employee Job:



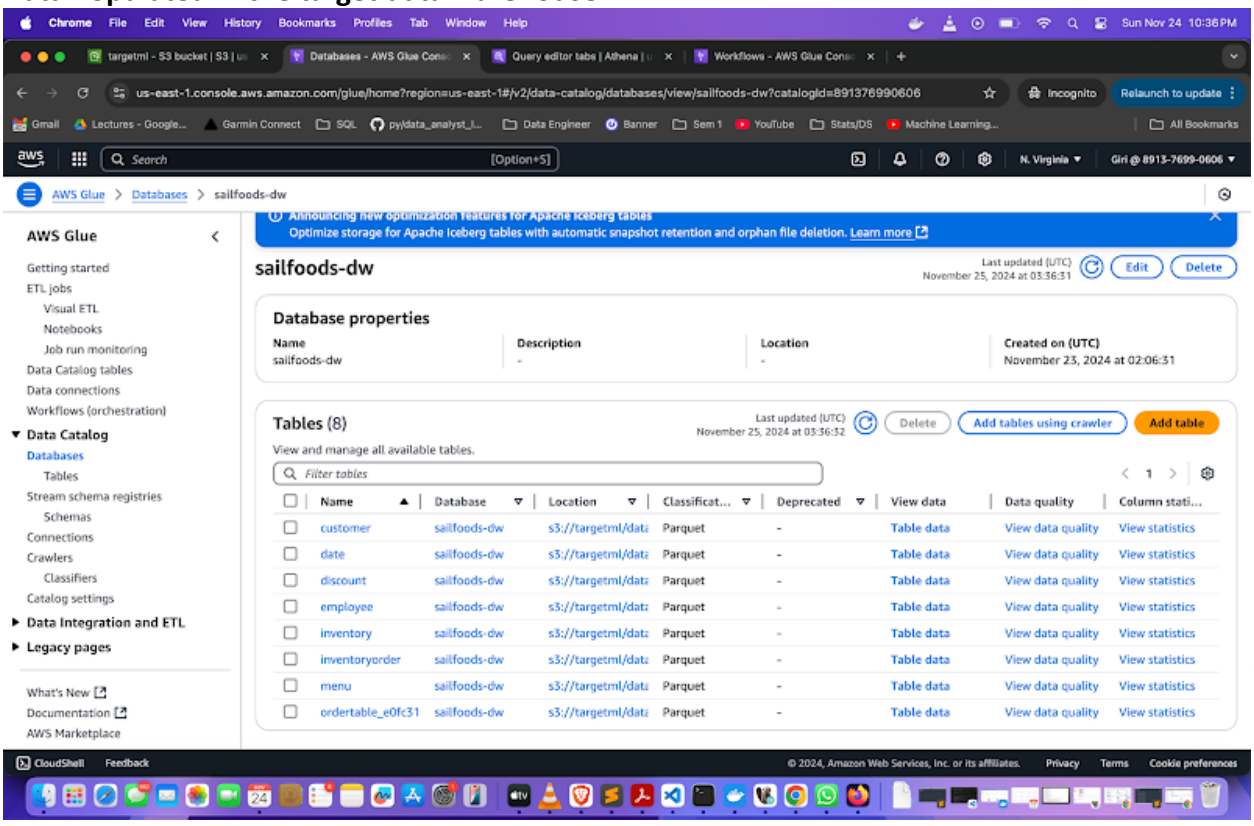
Customer Job:



Jobs executed successfully:



Data Populated in the target data warehouse:



Transformation implemented in the target employee table:

Employee

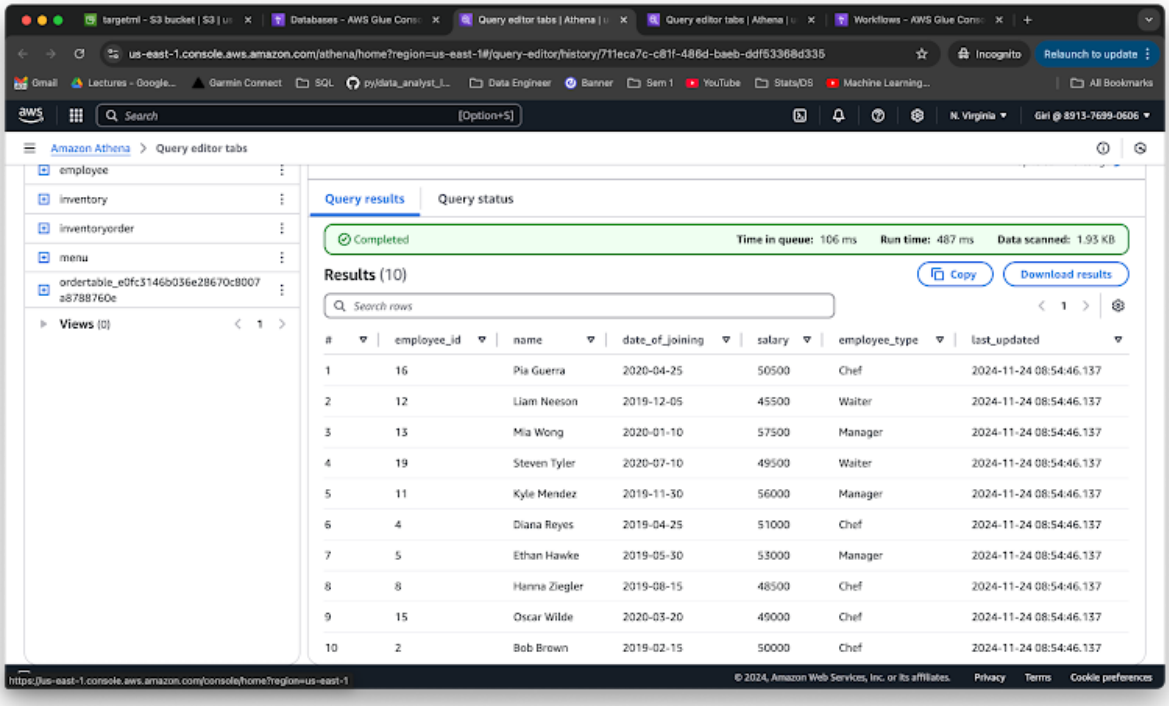
Last modified on 11/24/2024, 3:53:00 AM | Actions | Save | Run

Script | Job details | Runs | Data quality | Schedules | Version Control | Upgrade analysis - *preview*

Script Info

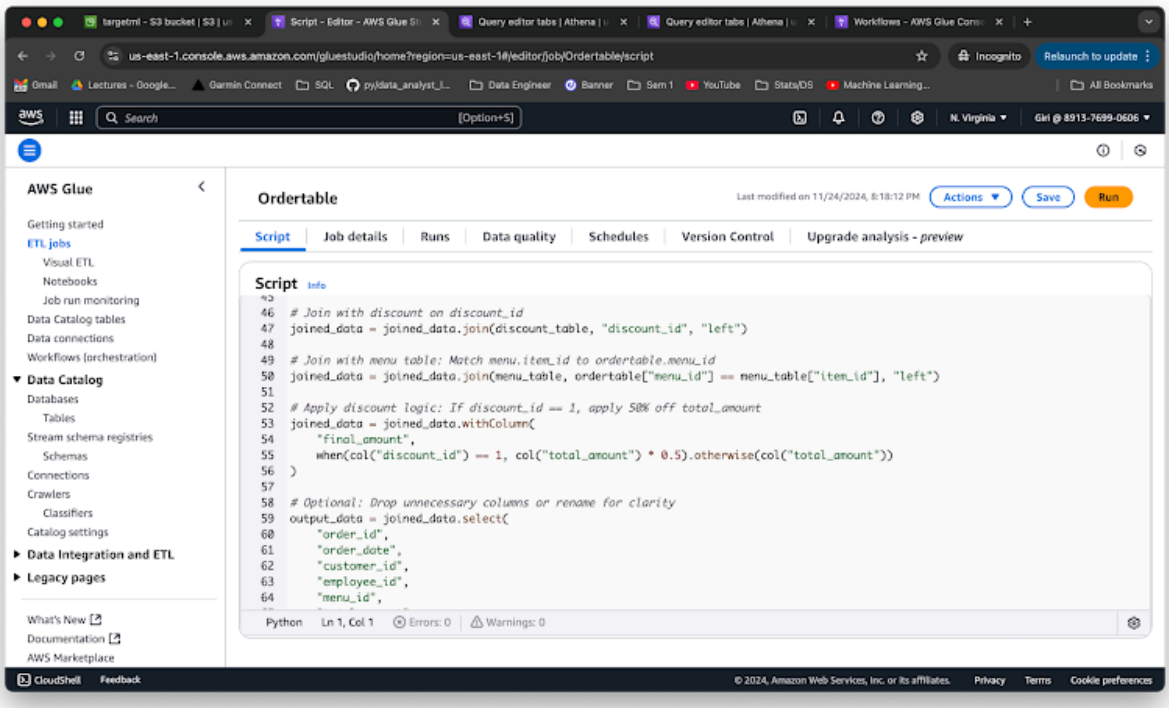
```
59  # Convert DynamicFrames to DataFrames for transformations
60  df_chef = source_chef.toDF()
61  df_manager = source_manager.toDF()
62  df_waiter = source_waiter.toDF()
63
64  # Add a specialization column to employees based on their presence in the specialization tables
65  df_employee_with_type = (
66      df_employee
67      .withColumn(
68          "Employee_Type",
69          when(col("employee_id").isin(df_chef.select("employee_id").rdd.flatMap(lambda x: x).collect()), "Chef")
70          .when(col("employee_id").isin(df_manager.select("employee_id").rdd.flatMap(lambda x: x).collect()), "Manager")
71          .when(col("employee_id").isin(df_waiter.select("employee_id").rdd.flatMap(lambda x: x).collect()), "Waiter")
72          .otherwise("General")  # Default to "General" if no specialization found
73      )
74  )
75
76  # Load existing cumulative employee table data from target S3 location
77  try:
78      # Attempt to load existing data from the cumulative table
```

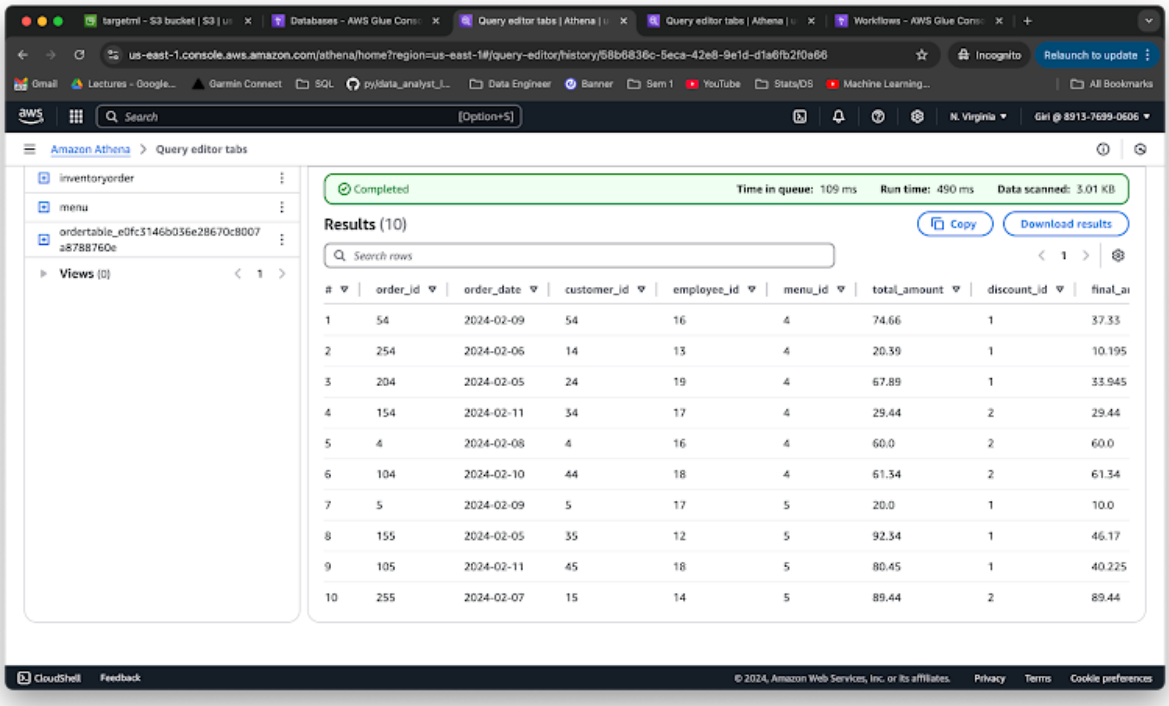
Python | Ln 1, Col 1 | Errors: 0 | Warnings: 0



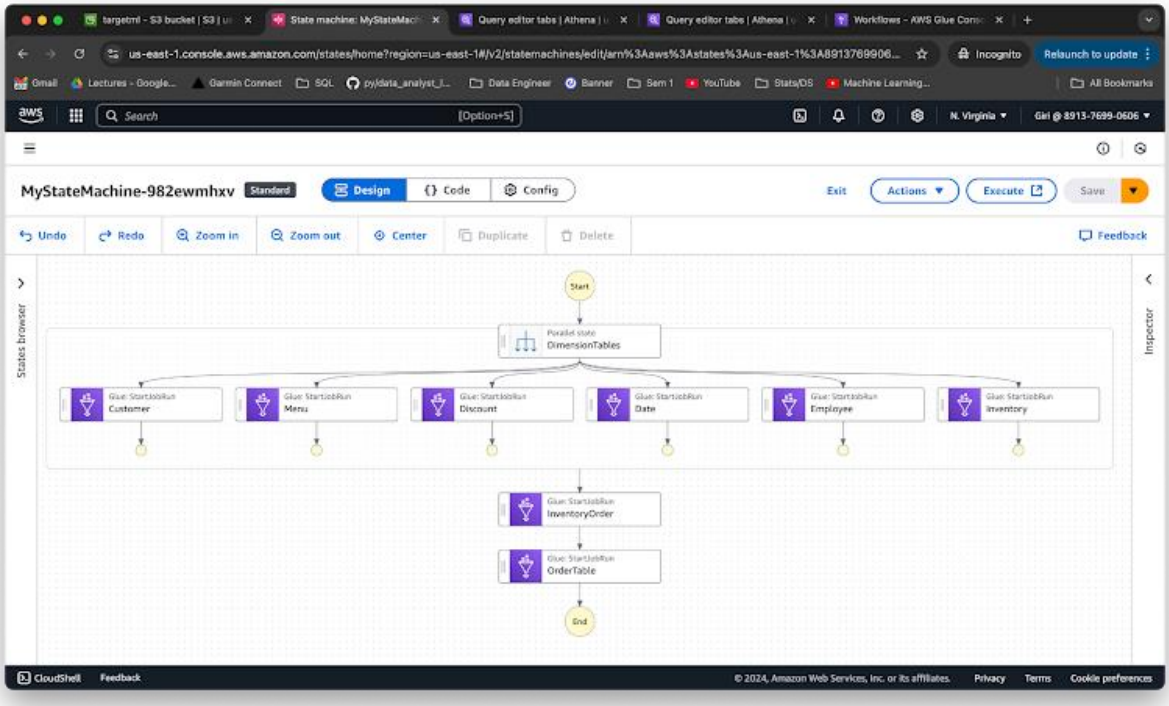
#	employee_id	name	date_of_joining	salary	employee_type	last_updated
1	16	Pia Guerra	2020-04-25	50500	Chef	2024-11-24 08:54:46.137
2	12	Liam Neeson	2019-12-05	45500	Waiter	2024-11-24 08:54:46.137
3	13	Mia Wong	2020-01-10	57500	Manager	2024-11-24 08:54:46.137
4	19	Steven Tyler	2020-07-10	49500	Waiter	2024-11-24 08:54:46.137
5	11	Kyle Mendez	2019-11-30	56000	Manager	2024-11-24 08:54:46.137
6	4	Diana Reyes	2019-04-25	51000	Chef	2024-11-24 08:54:46.137
7	5	Ethan Hawke	2019-05-30	53000	Manager	2024-11-24 08:54:46.137
8	8	Hanna Ziegler	2019-08-15	48500	Chef	2024-11-24 08:54:46.137
9	15	Oscar Wilde	2020-03-20	49000	Chef	2024-11-24 08:54:46.137
10	2	Bob Brown	2019-02-15	50000	Chef	2024-11-24 08:54:46.137

Measures implemented in the order table:



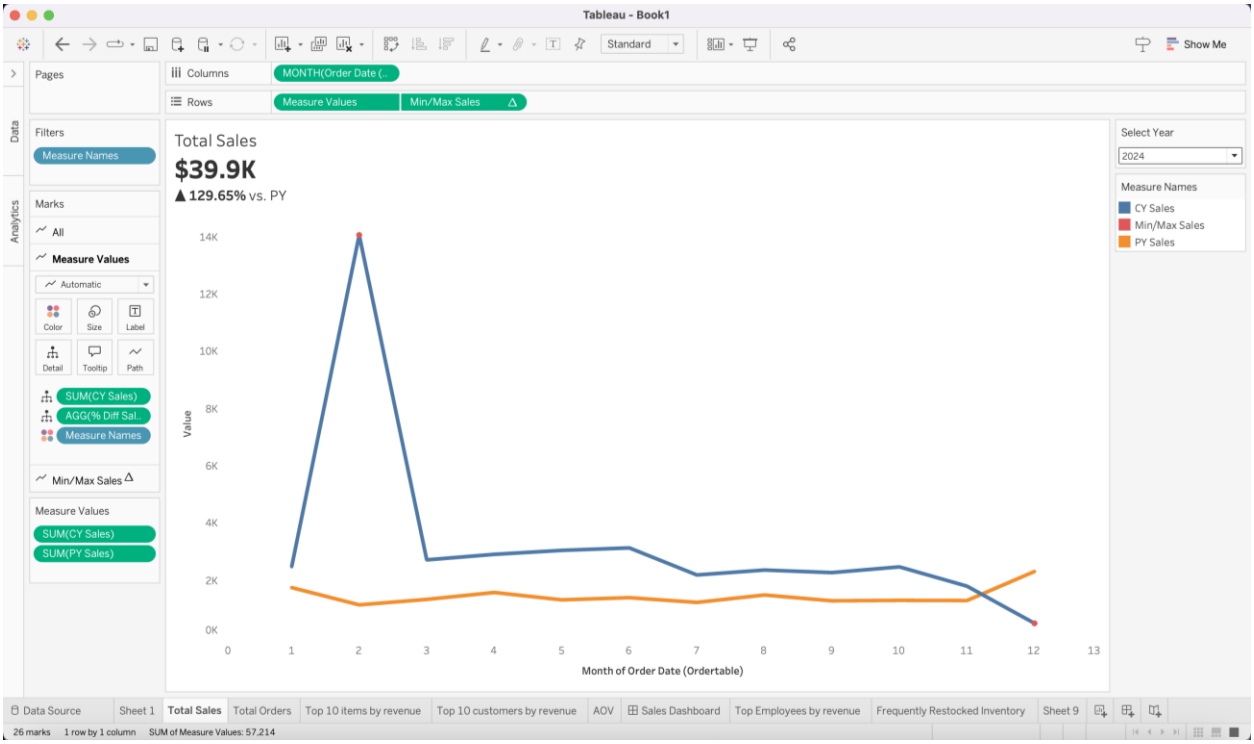


Control Flow:


MyStateMachine-982ewmhxv
Standard
Design
Code
Config
Exit
Actions
Execute
Save
Undo
Redo
Zoom in
Zoom out
Center
Duplicate
Delete
Feedback
States browser
Inspector
Start
Parallel state
DimensionTables
Glue: StartJobRun
Customer
Glue: StartJobRun
Menu
Glue: StartJobRun
Discount
Glue: StartJobRun
Date
Glue: StartJobRun
Employee
Glue: StartJobRun
Inventory
Glue: StartJobRun
InventoryOrder
Glue: StartJobRun
OrderTable
End
CloudShell
Feedback
© 2024, Amazon Web Services, Inc. or its affiliates.
Privacy
Terms
Cookie preferences

Milestone 7

Dashboard 1



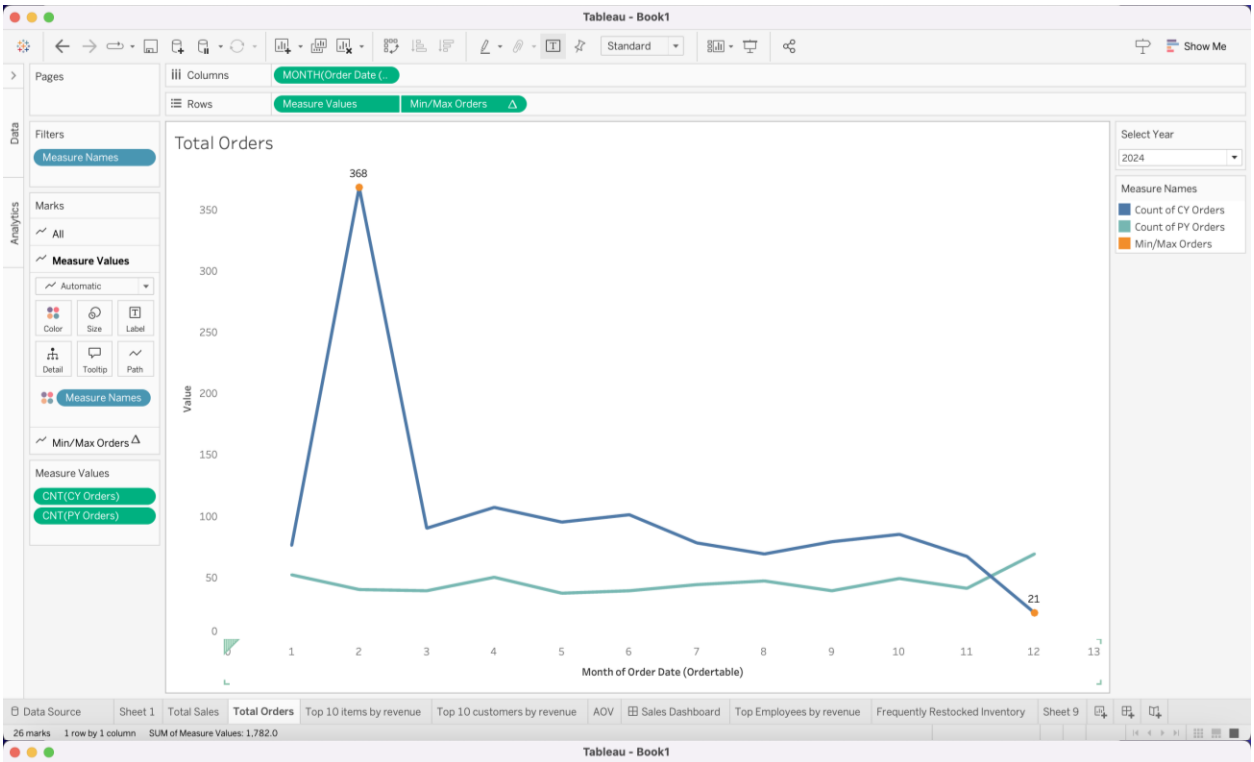


Tableau - Book1
Standard
Show Me
Pages
Columns
MONTH(Order Date (...
Rows
Measure Values
Min/Max Orders
Filters
Measure Names
Marks
All
Measure Values
Automatic
Color
Size
Label
Detail
Tooltip
Path
Measure Names
Min/Max Orders
Measure Values
CNT(CY Orders)
CNT(PY Orders)
Total Orders
368
21
Value
Month of Order Date (Ordertable)
Select Year
2024
Measure Names
Count of CY Orders
Count of PY Orders
Min/Max Orders
Data Source
Sheet 1
Total Sales
Total Orders
Top 10 items by revenue
Top 10 customers by revenue
AOV
Sales Dashboard
Top Employees by revenue
Frequently Restocked Inventory
Sheet 9
26 marks
1 row by 1 column
SUM of Measure Values: 1,782.0

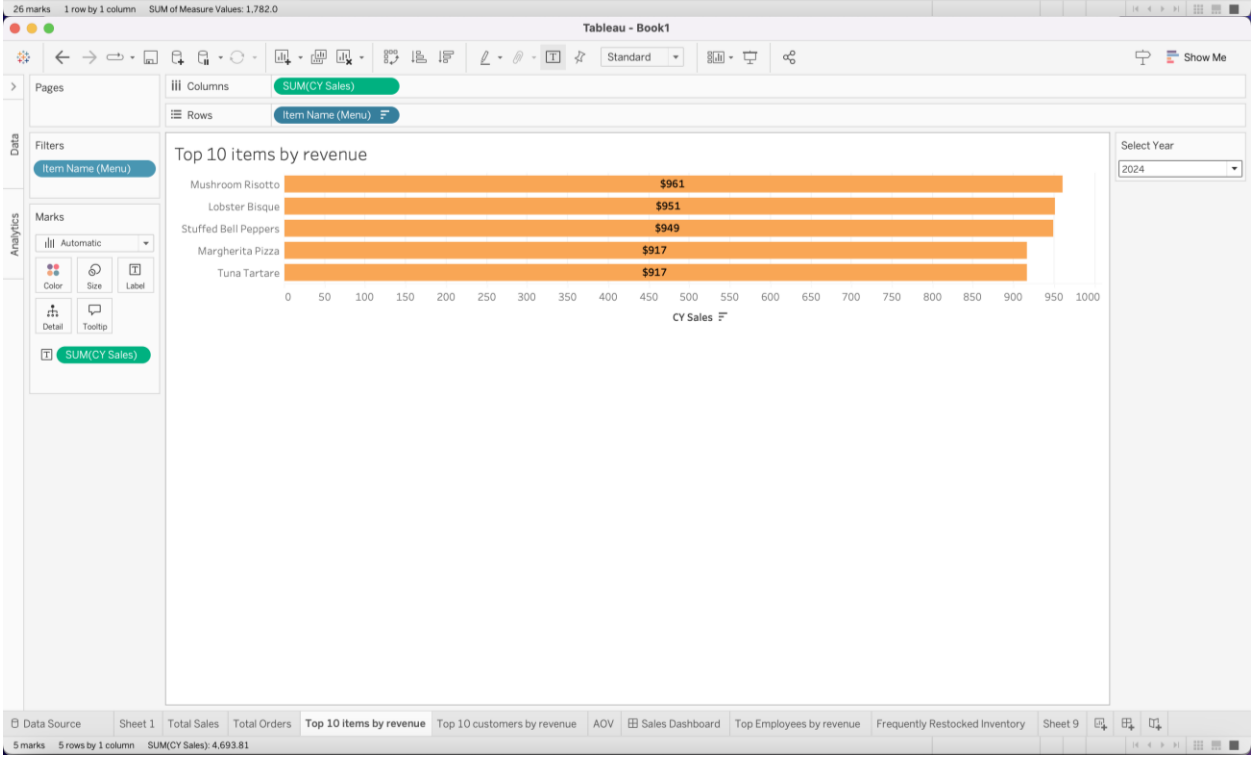


Tableau - Book1
Standard
Show Me
Pages
Columns
SUM(CY Sales)
Rows
Item Name (Menu)
Filters
Item Name (Menu)
Marks
Automatic
Color
Size
Label
Detail
Tooltip
SUM(CY Sales)
Top 10 items by revenue
Mushroom Risotto
Lobster Bisque
Stuffed Bell Peppers
Margherita Pizza
Tuna Tartare
$961
$951
$949
$917
$917
CY Sales
Select Year
2024
Data Source
Sheet 1
Total Sales
Total Orders
Top 10 items by revenue
Top 10 customers by revenue
AOV
Sales Dashboard
Top Employees by revenue
Frequently Restocked Inventory
Sheet 9
5 marks
5 rows by 1 column
SUM(CY Sales): 4,693.81

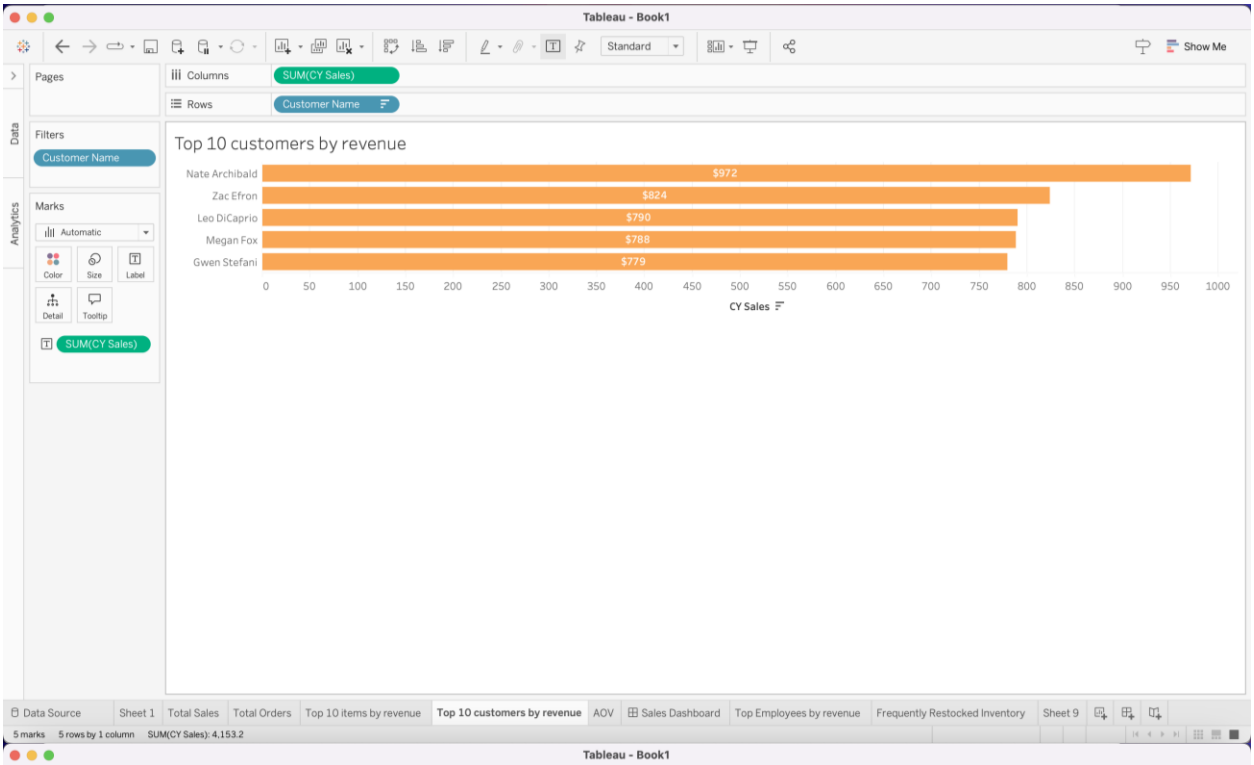


Tableau - Book1
Standard
Show Me
Pages
Columns
SUM(CY Sales)
Rows
Customer Name
Filters
Customer Name
Marks
Automatic
Color
Size
Label
Detail
Tooltip
SUM(CY Sales)
Top 10 customers by revenue
Nate Archibald
Zac Efron
Leo DiCaprio
Megan Fox
Gwen Stefani
$972
$824
$790
$788
$779
0 50 100 150 200 250 300 350 400 450 500 550 600 650 700 750 800 850 900 950 1000
CY Sales
Data Source
Sheet 1
Total Sales
Total Orders
Top 10 items by revenue
Top 10 customers by revenue
AOV
Sales Dashboard
Top Employees by revenue
Frequently Restocked Inventory
Sheet 9
5 marks
5 rows by 1 column
SUM(CY Sales): 4,153.2

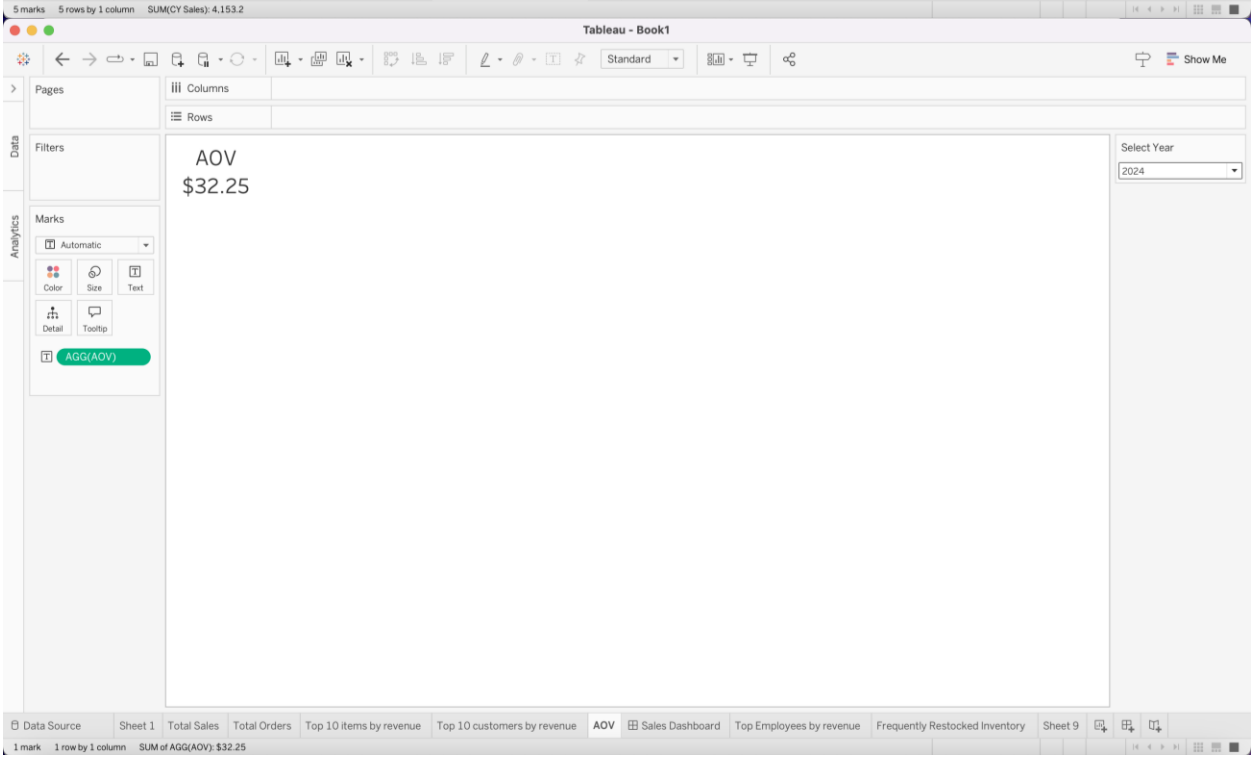
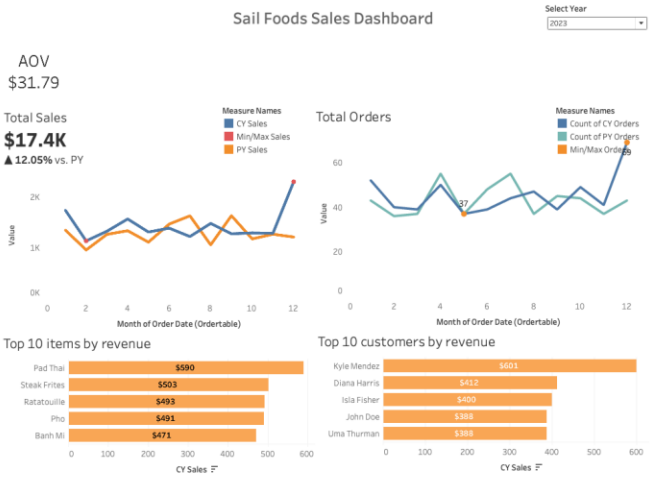
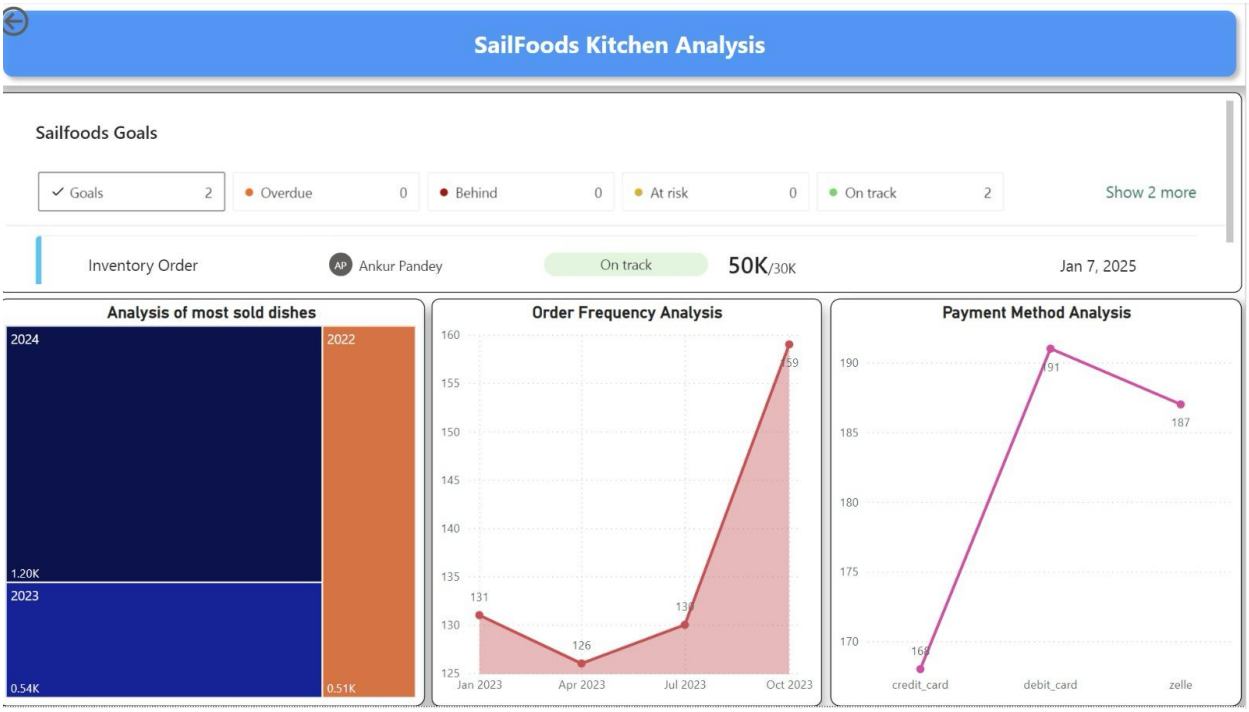


Tableau - Book1
Standard
Show Me
Pages
Columns
Rows
Filters
Marks
Automatic
Color
Size
Text
Detail
Tooltip
AGG(AOV)
AOV
$32.25
Select Year
2024
Data Source
Sheet 1
Total Sales
Total Orders
Top 10 items by revenue
Top 10 customers by revenue
AOV
Sales Dashboard
Top Employees by revenue
Frequently Restocked Inventory
Sheet 9
1 mark
1 row by 1 column
SUM of AGG(AOV): $32.25

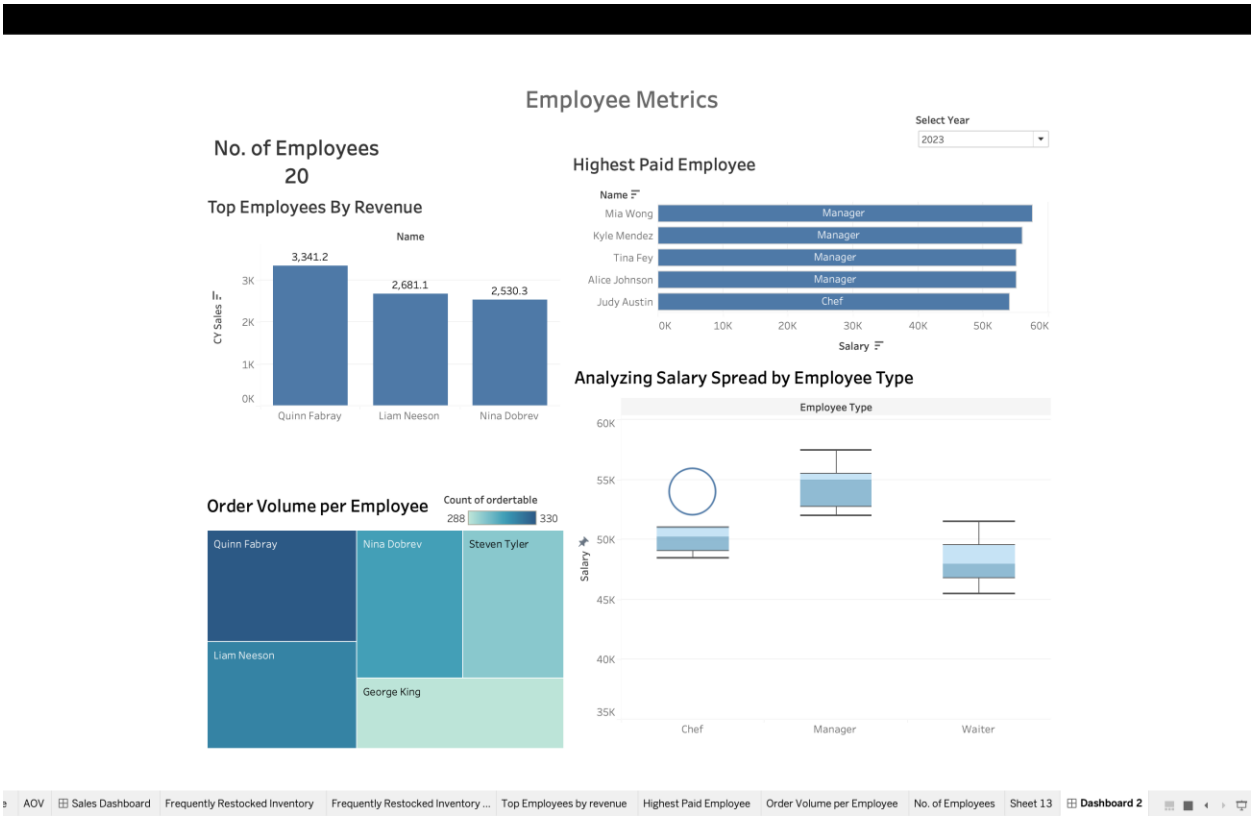


Sales | Total Orders | Top 10 items by revenue | Top 10 customers by revenue | AOV | **Sales Dashboard** | Frequently Restocked Inventory | Frequently Restocked Inventory ... | Top Employees by revenue | Highest Paid Employee | Order Volume per Employee | No. of Employees | Sheet 13 | Dashboard 2

Dashboard 2


SailFoods Kitchen Analysis
Sailfoods Goals
Goals 2
Overdue 0
Behind 0
At risk 0
On track 2
Show 2 more
Inventory Order
Ankur Pandey
On track
50K/30K
Jan 7, 2025
Analysis of most sold dishes
2024
1.20K
2023
0.54K
2022
0.51K
Order Frequency Analysis
160
155
150
145
140
135
130
125
131
126
159
Jan 2023
Apr 2023
Jul 2023
Oct 2023
Payment Method Analysis
190
185
180
175
170
191
187
credit_card
debit_card
zelle

Dashboard 3


Employee Metrics
Select Year
2023
No. of Employees
20
Top Employees By Revenue
Name
CY Sales
3,341.2
2,681.1
2,530.3
3K
2K
1K
0K
Quinn Fabray
Liam Neeson
Nina Dobrev
Highest Paid Employee
Name
Mia Wong
Kyle Mendez
Tina Fey
Alice Johnson
Judy Austin
Manager
Manager
Manager
Manager
Chef
0K
10K
20K
30K
40K
50K
60K
Salary
Analyzing Salary Spread by Employee Type
Employee Type
60K
55K
50K
45K
40K
35K
Salary
Chef
Manager
Waiter
Order Volume per Employee
Count of ordertable
288
330
Quinn Fabray
Nina Dobrev
Steven Tyler
Liam Neeson
George King
AOV
Sales Dashboard
Frequently Restocked Inventory
Frequently Restocked Inventory ...
Top Employees by revenue
Highest Paid Employee
Order Volume per Employee
No. of Employees
Sheet 13
Dashboard 2