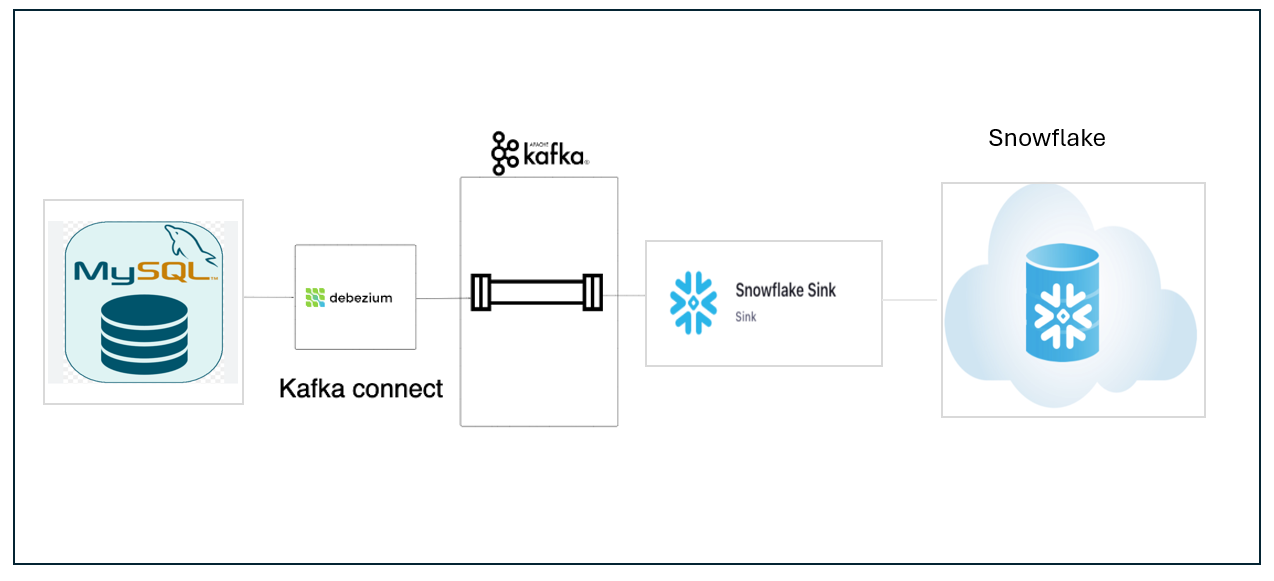
This document outlines the data ingestion and Change Data Capture (CDC) implementation performed for the ETL, highlighting the chosen tools, architecture diagram, and the steps taken.

# Architecture Diagram:

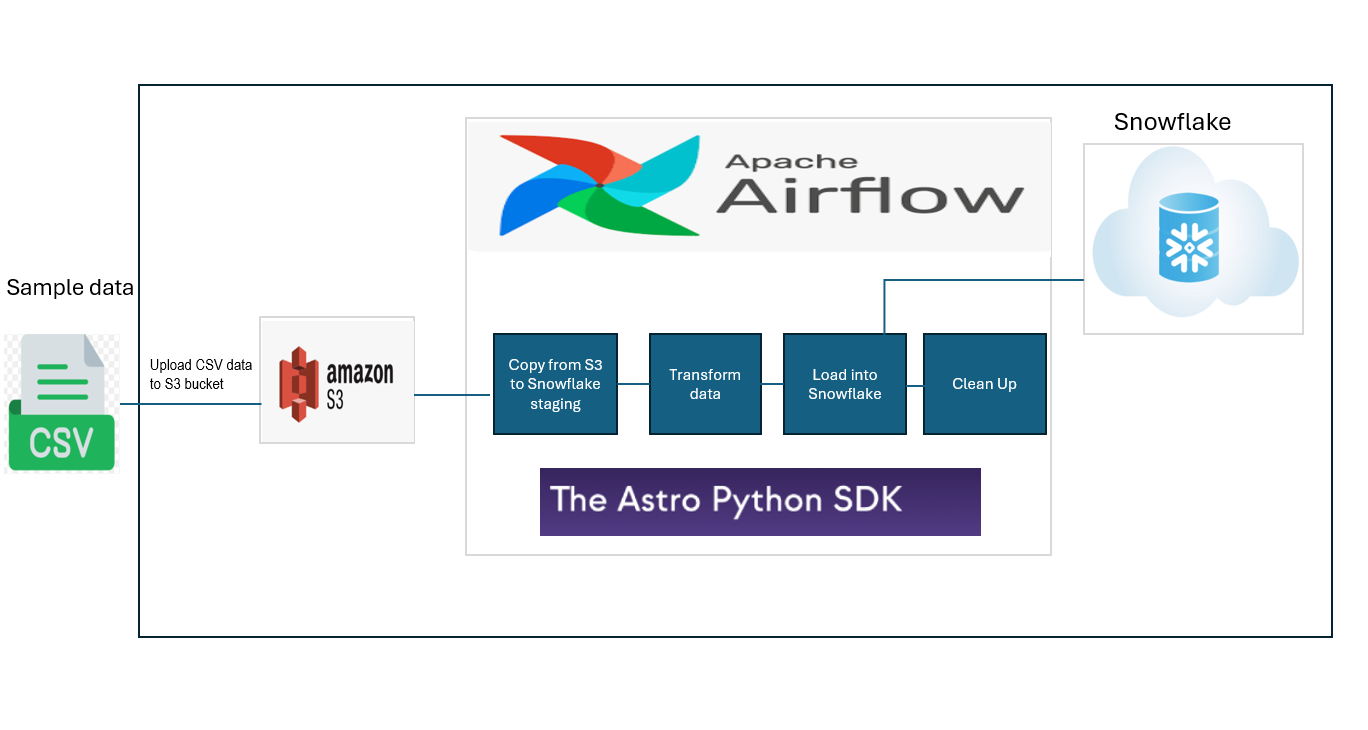
Here is the architecture diagram for my CDC process: I use an Aiven-hosted MySQL database for the source, a Kafka cluster as the real-time messaging queue, and Debezium connectors (source and target) to capture and apply changes to Snowflake.

**CDC process**



**Ingestion process**

Here is the architecture diagram for my ingestion process, using Astro SDK for ETL: I upload a CSV file to an S3 bucket, triggering an Airflow DAG that copies it to Snowflake staging, transforms the data, loads it into a final table, and cleans up.



# Step 1: Initial Data Ingestion

# Environment Setup:

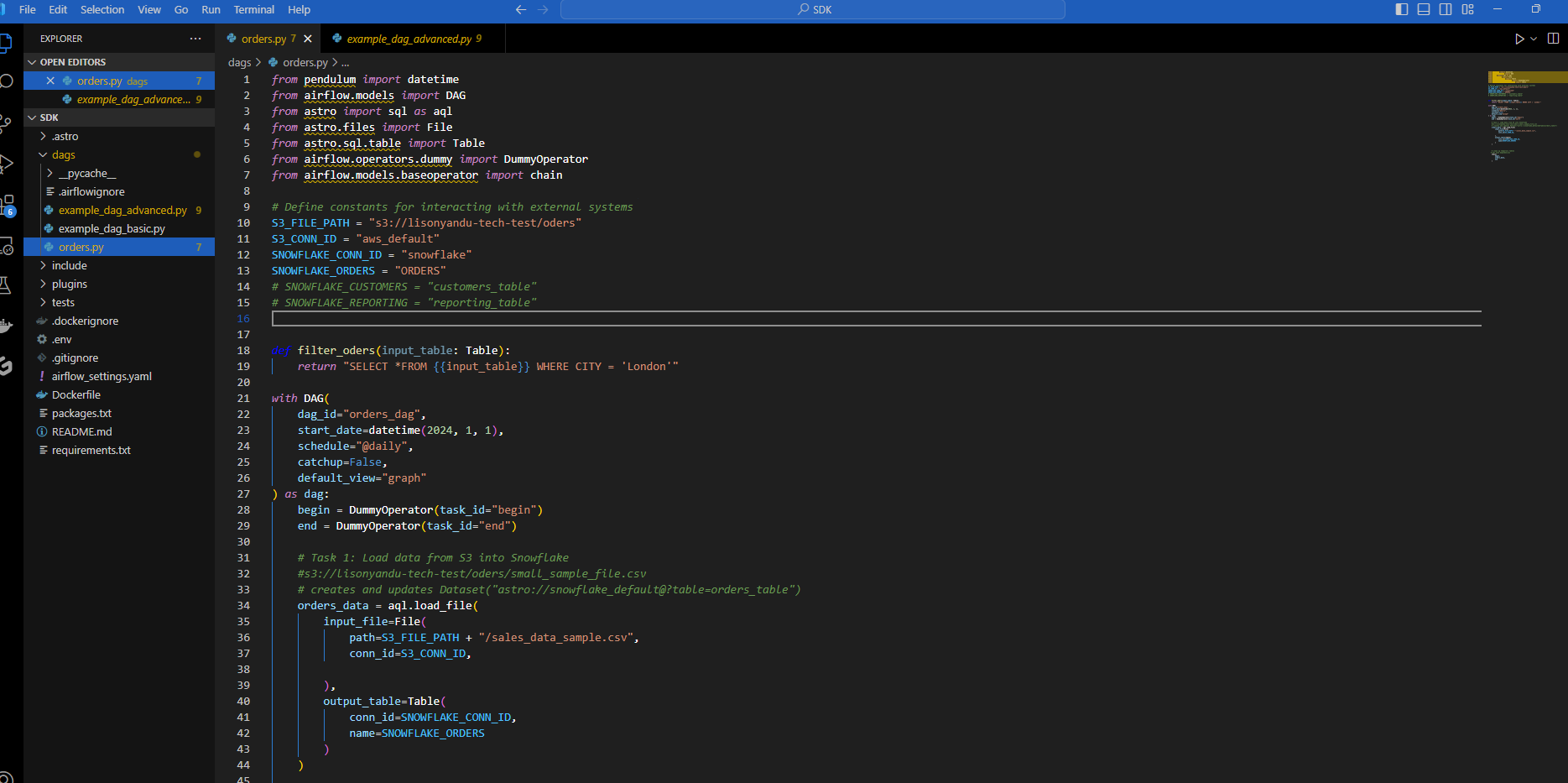
* **AWS Free Tier:**
  + Create an AWS free tier account.
  + Set up an S3 bucket for storing the csv sample data file.
  + Create access keys and secret access keys for programmatic access to S3. Store them somewhere safe.
* **Snowflake Trial Account:**
  + Create a Snowflake trial account.
  + Create a user and role with appropriate access privileges.
  + Create a database and schema for storing the ingested data.

# Create an ORDERS table with the desired schema (columns and data types).

# 2. Data Ingestion with Airflow:

Choose Airflow for orchestration.

Write an Airflow DAG using the Astro Python SDK to copy data from S3 to Snowflake.

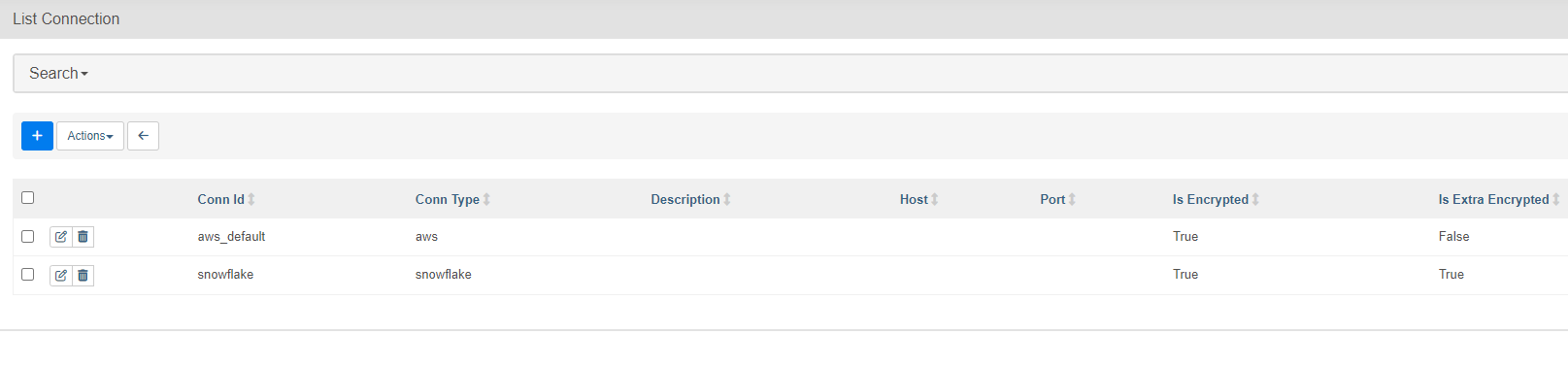


# 3. Airflow Execution and Monitoring:

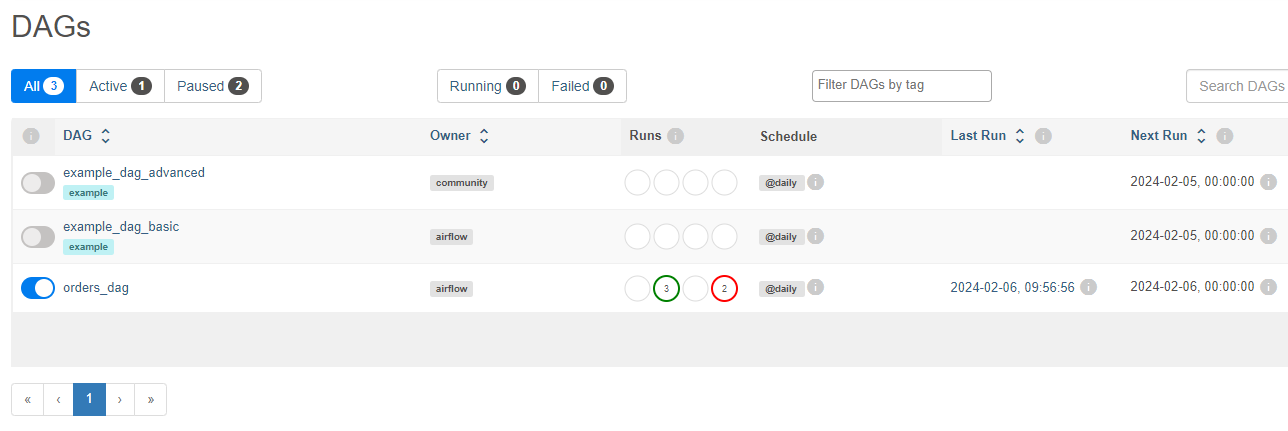
* Start Airflow Webserver by running “ *astro dev star*t” : Open Airflow in your browser at <http://localhost:8080>
* Set up Airflow connections for S3 and Snowflake:

**S3 Connection**: Use the access keys and secret access keys to create a connection to your S3 bucket.

**Snowflake Connection:** Use the Snowflake account credentials to create a connection to your Snowflake database.



* Trigger the DAG: Manually trigger the DAG to start the data ingestion process.
* Monitor DAG execution: Use the Airflow Web UI to track the progress and status of each task in the DAG

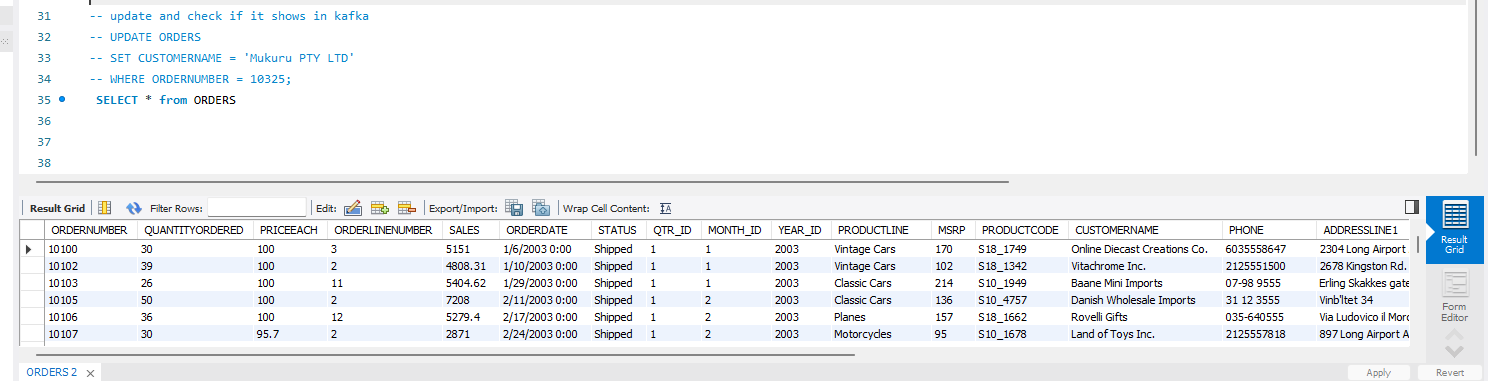


# Step 2: Implementing CDC with DML and DDL Updates with Kafka and Debezium

# 3.1 Set up source database using Aiven:

Aiven is a cloud platform that provides fully managed open-source infrastructure services, like databases, Kafka, Elasticsearch, and other tools.

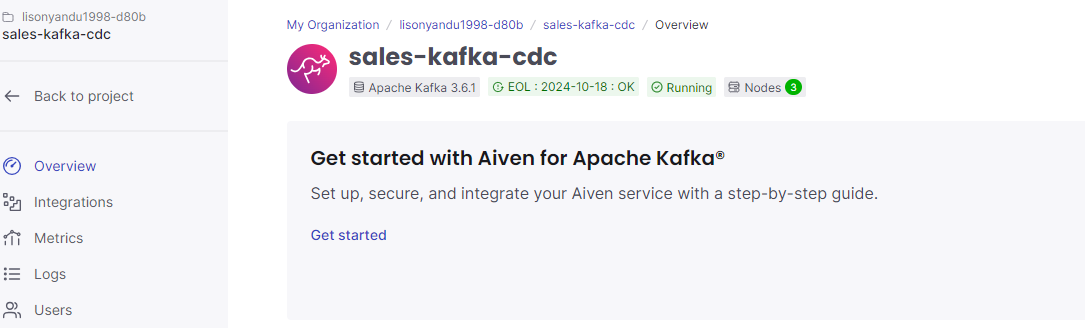
* Create a new database instance using Aiven.
* Connect it to MySQL Workbench.
* Define Table Structure and Populate Orders



# 

# 3.2. Apache Kafka Setup

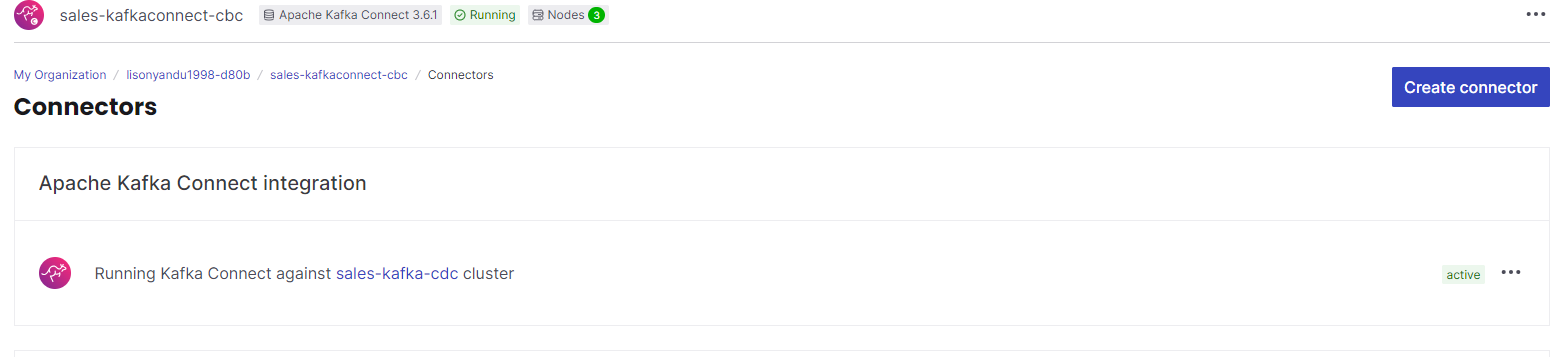
* Kafka Cluster: Create a Kafka cluster to serve as the real-time messaging queue for CDC data.



# 

# 3.3 Debezium Connect Configuration

* **Source Connector**: Configure a Debezium connector for MySQL as the source connector, pointing to the Aiven MySQL instance.



* **Target Connector:** Configured a Debezium connector for Snowflake as the target connector, pointing to the Snowflake orders database and table.

# CDC Capture:

Start the Debezium connectors to capture DML (INSERT, UPDATE, DELETE) changes on the source database and stream them to the Kafka topic.

**DML**

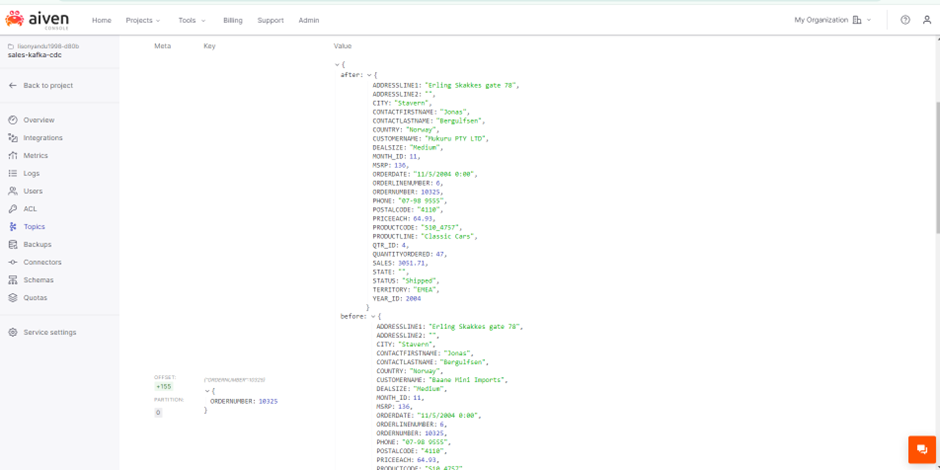
**Execute**  the following in MySQL workbench:

*UPDATE ORDERS*

*SET CUSTOMERNAME = 'Mukuru PTY LTD'*

*WHERE ORDERNUMBER = 10325;*

Here’s an example of what a change capture data point created looks like:



### **Inserting data**

INSERT INTO ORDERS (

ORDERNUMBER,

QUANTITYORDERED,

PRICEEACH,

ORDERLINENUMBER,

SALES,

ORDERDATE,

STATUS,

QTR\_ID,

MONTH\_ID,

YEAR\_ID,

PRODUCTLINE,

MSRP,

PRODUCTCODE,

CUSTOMERNAME,

PHONE,

ADDRESSLINE1,

CITY,

STATE,

POSTALCODE,

COUNTRY,

TERRITORY,

CONTACTLASTNAME,

CONTACTFIRSTNAME,

DEALSIZE

) VALUES (

21310,

3,

100.10,

6,

934.62,

'10/16/2004 0:00',

'Shipped',

4,

10,

2004,

'Classic Cars',

194,

'S12\_1099',

'Toms Spezialitten, Ltd',

'0221-5554327',

'Mehrheimerstr. 369',

'Koln',

NULL,

'50739',

'Germany',

'EMEA',

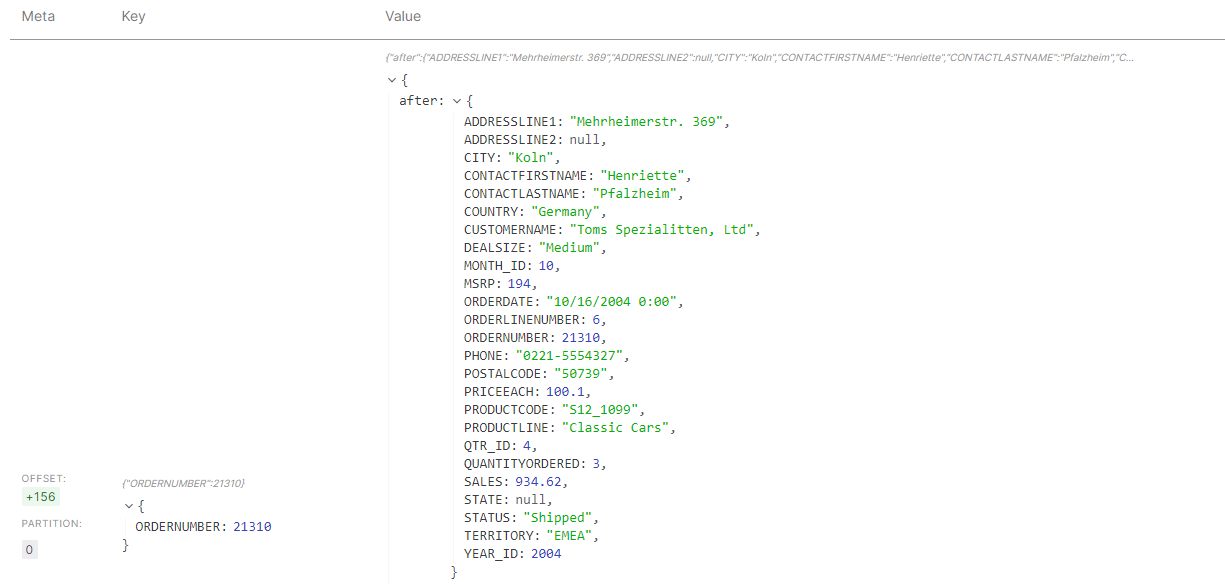
'Pfalzheim',

'Henriette',

'Medium'

);

Here’s an example of what a change capture data point created looks like:



## DDL

**Alter Table**

ALTER TABLE `ORDERS`

RENAME COLUMN `ORDERNUMBER` TO `ORDERNUM`;

Here’s an example of what a change capture data point created looks like:



# Challenges

Connecting Airflow and Snowflake initially proved challenging. The pipeline kept failing due to connection and version issues. Thankfully, Airflow and Snowflake documentation helped me navigate these issues. The biggest issue was configuring the Debezium connector. The configuration file contains many variables that I initially struggled to understand. Fortunately, the Aiven documentation proved to be extremely helpful, allowing me to create a connector that successfully captures changes made in my source database. Unfortunately, the extensive time I spent debugging the connector prevented me from seamlessly combining Step 1 and Step 2 and I also couldn't complete the optional BI Report/Dashboard, which I had originally intended to do. Nonetheless, it was a good exercise, and I enjoyed learning how the different tools fit together.