# Coding Assignment - Small Ecommerce Shop Datawarehouse

Your datasources are CSV exports of the operational databases of the ecommerce shop microservices. You have the following tables:

## Customer Service

### Customer

\* \*\*customer\_id\*\* (type: string) uuid for every customer

\* \*\*customer\_name\*\* (type: string) name of the customer

\* \*\*customer\_email\*\* (type: string) email of the customer

\* \*\*customer\_phone\*\* (type: string) phonenumber of the customer

## Product Service

### Product

\* \*\*product\_id\*\* (type: string) uuid for every product

\* \*\*product\_name\*\* (type: string) name of the product

\* \*\*product\_customer\_price\*\* (type: float) price in USD the product is selled to the customer

\* \*\*product\_purchasing\_price\*\* (type: float) price in USD the product is bought from the supplier

## Inventory Service

### Inventory

\* \*\*inventory\_id\*\* (type: string) uuid for every inventory

\* \*\*inventory\_product\_id\*\* (type: string) uuid of the product which is in the inventory

\* \*\*inventory\_quantity\*\* (type: integer) number of products in the inventory

\* \*\*inventory\_location\*\* (type: string) location of the inventory

## Order Service

### Order

\* \*\*order\_id\*\* (type: string) uuid for every order

\* \*\*order\_customer\_id\*\* (type: string) uuid of the customer who placed the order

\* \*\*order\_total\_price\*\* (type: float) total price of the order in USD

\* \*\*order\_date\*\* (type: string) date when the order was placed

\* \*\*order\_status\*\* (type: string) status of the order (e.g. "processing", "shipped", "delivered")

\* \*\*order\_shipping\_address\*\* (type: string) address where the order should be shipped

\* \*\*order\_billing\_address\*\* (type: string) address where the bill should be sent

\* \*\*order\_delivery\_date\*\* (type: string) date when the order was delivered

\* \*\*order\_payment\_method\*\* (type: string) payment method used for the order

\* credit card, paypal, bank transfer, klarna, amazon pay

\* \*\*order\_payment\_status\*\* (type: string) payment status of the order (e.g. "pending", "paid", "refunded")

\* \*\*order\_payment\_date\*\* (type: string) date when the payment was made

\* \*\*order\_shipping\_method\*\* (type: string) shipping method used for the order

\* dhl, dpd, ups, fedex, hermes, gls, tnt, amazon

\* \*\*order\_shipping\_status\*\* (type: string) shipping status of the order (e.g. "pending", "shipped", "delivered")

\* \*\*order\_shipping\_date\*\* (type: string) date when the order was shipped

### Order Items

\* \*\*order\_id\*\* (type: string) uuid of the order which contains the product, reference to order table

\* \*\*order\_product\_id\*\* (type: string) uuid of the product which was ordered

\* \*\*order\_quantity\*\* (type: integer) number of products ordered

## Ask

1. Create a data model for the datawarehouse.

2. Create a data pipeline to load the data into the datawarehouse.

3. Create a SQL query to calculate the revenue per customer.

4. Create a SQL query to calculate the value of goods stored in the different inventory locations.

5. Create a SQL query to calculate the average order value per customer.

6. Run the data pipeline and execute the SQL queries on the datawarehouse.

7. Have some thought how to improve the usage of the datawarehouse.

As the requirement was not for historical data, current design is in the simplest form. However, can be extended by creating SCD-Type-2-dimension tables for Customer and product. Then it would have a surrogate key and start/end date columns.

Inventory data doesn’t have any date column. If it’s required to report historical data, it will need to have DATE/Month column as per the requirement and have a partition on that column.

Order table also can be partitioned based on which DATE column is used to report monthly Order details : order\_shipping\_date, order\_payment\_date, order\_delivery\_date.

Order Fact table is generally accumulated snapshot table wherein dates are updated as order progresses. However, in cloud DB, updates are very slow. So, we can model this table as periodic snapshot table as inserts are always fast in BQ.

Design Considerations / Assumptions:

1. For one time analysis and to fulfill current requirements, the simplest design for dimension tables as SCD Type-1 will suffice in DWH.
2. As there is no DATE mentioned in Inventory, it can not have historical data right now.
3. No CDC ( Change Data Capture ) or BAU scheduling is done as the ask was to only create the pipeline.
4. As there was no mention of Database/environment, I have chosen Google Cloud environment for the development : Google Cloud Storage, BigQuery

Coding Artifacts :