```
USE WAREHOUSE tmg2546_WH_CAPSTONE;
USE SCHEMA ELT_STAGE;

-- Creates an external_stage in your ELT schema.
CREATE OR REPLACE STAGE ELT_STAGE.ELT_RAW_EXTERNAL_STAGE
COMMENT = 'Raw External Stage for the ELT Account on the RRC DataLake Blob
Container'
STORAGE_INTEGRATION = tmg2546_RAW_STORAGE_INTEGRATION
URL = 'azure://tylersblobstorage1.blob.core.windows.net/raw/';

-- Creating tables from Azure to Snowflake

-- (1) Create table to store Customers data

create OR replace transient table ELT_STAGE.Customers_dw (
    Customer_id                     varchar(50)             primary key,
    Customer_zip_code_prefix        int,
    Customer_city                   string,
    Customer_state                  string
);


-- (2) Create table to store Order Items data

create OR replace transient table ELT_STAGE.Order_Items_dw (
    Order_id                    varchar(50)             primary key,
    Product_id                  varchar(50),
    Seller_id                   varchar(50),
    Price                       float,
    Shipping_charges            float
);


-- (3) Create table to store Orders data

create OR replace transient table ELT_STAGE.Orders_dw (
    Order_id                            varchar(50)             primary
key,
    Customer_id                         varchar(50),
    Order_status                        string,
    Order_purchase_timestamp            datetime,
    Order_approved_at                   datetime,
    Order_delivered_timestamp           datetime,
    Order_estimated_delivery_date       date
);

-- (4) Create table to store Payments data

create OR replace transient table ELT_STAGE.Payments_dw (
    Order_id                    varchar(50),
    Payment_sequential          int,
    Payment_type                varchar(50),
    Payment_installments        int,
    Payment_value               float
);

--(5) Create table to store Products data

create OR replace transient table ELT_STAGE.Products_dw (
    Product_id                      varchar(50)             primary key,
    Product_category_name           varchar(50),
    Product_weight_g                int,
    Product_length_cm               int,
    Product_height_cm               int,
    Product_width_cm                int
);


-- Create a FILE FORMAT --> Example (CSV with headers)
CREATE OR REPLACE FILE FORMAT ELT_STAGE.ELT_CSV_COMMA_DELIMITED_HEADER
COMMENT = 'File Format for CSV comma delimited Column Header files'
COMPRESSION = 'NONE'
TYPE = CSV                                -- Set file tyle
FIELD_DELIMITER = ','                     -- Delimits columns by comma
RECORD_DELIMITER = '\n'                   -- Delimits rows by line break
SKIP_HEADER = 1                           -- Skip the first row and donâ□□t
treat as data
FIELD_OPTIONALLY_ENCLOSED_BY = '\042'
TRIM_SPACE = FALSE
ERROR_ON_COLUMN_COUNT_MISMATCH = FALSE
ESCAPE = '\134'
ESCAPE_UNENCLOSED_FIELD = 'NONE'
DATE_FORMAT = 'AUTO'
TIMESTAMP_FORMAT = 'AUTO'
EMPTY_FIELD_AS_NULL = TRUE;


--------------------------------------------------------------------------
--------------
-- COPY INTO statements to pull the 5 other files into your Snowflake
ELT_STAGE schema
--------------------------------------------------------------------------
--------------

-- CUSTOMERST_DW
TRUNCATE TABLE ELT_STAGE.CUSTOMERS_DW;

---copy from Customers raw file into CUSTOMERS_DW table
COPY INTO ELT_STAGE.CUSTOMERS_DW
FROM @ELT_STAGE.ELT_RAW_EXTERNAL_STAGE/raw_381npg_Customers_data.csv
FILE_FORMAT = ELT_STAGE.ELT_CSV_COMMA_DELIMITED_HEADER
ON_ERROR=CONTINUE;

-- SELECT *
-- FROM ELT_STAGE.Customers_dw;

----------------------------------------------------------------------
```

```
-- ORDERS_DW
TRUNCATE TABLE ELT_STAGE.ORDERS_DW;

---copy from orders raw file into ORDERS_DW table
COPY INTO ELT_STAGE.ORDERS_DW
FROM @ELT_STAGE.ELT_RAW_EXTERNAL_STAGE/raw_381npg_Orders_data.csv
FILE_FORMAT = ELT_STAGE.ELT_CSV_COMMA_DELIMITED_HEADER
ON_ERROR=CONTINUE;

-- SELECT *
-- FROM ELT_STAGE.ORDERS_DW;

------------------------------------------------------------------------

-- ORDER_ITEMS_DW
TRUNCATE TABLE ELT_STAGE.ORDER_ITEMS_DW;

---copy from employee raw file into employee_dw table
COPY INTO ELT_STAGE.ORDER_ITEMS_DW
FROM @ELT_STAGE.ELT_RAW_EXTERNAL_STAGE/raw_381npg_Order_Items_data.csv
FILE_FORMAT = ELT_STAGE.ELT_CSV_COMMA_DELIMITED_HEADER
ON_ERROR=CONTINUE;

-- SELECT *
-- FROM ELT_STAGE.ORDER_ITEMS_DW;

-------------------------------------------------------------------------

-- PAYMENTS_DW
TRUNCATE TABLE ELT_STAGE.PAYMENTS_DW;

---copy from payments raw file into PAYMENTS_DW table
COPY INTO ELT_STAGE.PAYMENTS_DW
FROM @ELT_STAGE.ELT_RAW_EXTERNAL_STAGE/raw_hrsys2022_Payments_data.csv
FILE_FORMAT = ELT_STAGE.ELT_CSV_COMMA_DELIMITED_HEADER
ON_ERROR=CONTINUE;

-- SELECT *
-- FROM ELT_STAGE.PAYMENTS_DW;

---------------------------------------------------------------------------

-- PRODUCTS_DW
TRUNCATE TABLE ELT_STAGE.PRODUCTS_DW;

---copy from products raw file into PRODUCTS_DW table
COPY INTO ELT_STAGE.PRODUCTS_DW
FROM @ELT_STAGE.ELT_RAW_EXTERNAL_STAGE/raw_hrsys2022_Products_data.csv
FILE_FORMAT = ELT_STAGE.ELT_CSV_COMMA_DELIMITED_HEADER
ON_ERROR=CONTINUE;

-- SELECT *
-- FROM ELT_STAGE.PRODUCTS_DW;
```

```
-------------------------------------------------------------------------
-
------------------- CREATION OF THE SILVER LAYER ------------------------
---
-------------------------------------------------------------------------
-

CREATE or REPLACE SCHEMA EDW_SILVER_LAYER;

use SCHEMA edw_silver_layer;

-- First merged table that focuses on customers + the orders they placed &
how they paid for those orders
CREATE OR REPLACE TABLE edw_silver_layer.Customer_Transactions as (
Select      c.CUSTOMER_ID,
            c.CUSTOMER_CITY || ', ' || c.CUSTOMER_STATE as ADDRESS,
            o.ORDER_ID,
            p.PAYMENT_TYPE,
            p.PAYMENT_INSTALLMENTS,
            p.PAYMENT_VALUE

from ELT_STAGE.CUSTOMERS_DW c
    inner join ELT_STAGE.ORDERS_DW o on c.CUSTOMER_ID = o.CUSTOMER_ID
    inner join ELT_STAGE.PAYMENTS_DW p on o.ORDER_ID = p.ORDER_ID
WHERE o.ORDER_ID IS NOT NULL -- this dedup query makes sure there are no
null order_ids in the dataset
);

-- Checks to make sure the table is created and outputs correctly --
select *
from Customer_Transactions;

-- Checking to make sure there are no duplicates with order numbers --

-- SELECT
--     CUSTOMER_ID,
--     ORDER_ID,
--     COUNT(*) AS order_count
-- FROM
--     Customer_Transactions
-- GROUP BY
--     CUSTOMER_ID,
--     ORDER_ID
-- HAVING
--     COUNT(ORDER_ID) > 1;


-- Second merged table that focuses on the information regarding orders
placed and for what products
CREATE OR REPLACE TABLE edw_silver_layer.Orders_Info_Table as (
Select Distinct   -- this dedup query makes sure only unique orders are
used for the table creation
            o.ORDER_ID,
            o.ORDER_STATUS,
            o.ORDER_PURCHASE_TIMESTAMP,
            o.ORDER_APPROVED_AT,
            o.ORDER_ESTIMATED_DELIVERY_DATE,
            o.ORDER_DELIVERED_TIMESTAMP,
            DATEDIFF('day', o.ORDER_PURCHASE_TIMESTAMP,
o.ORDER_DELIVERED_TIMESTAMP) as Days_to_Customer, -- this shows how long
it takes from customer purchasing
an item to when it is actually delivered
            oi.SELLER_ID,
            oi.PRICE,
            oi.SHIPPING_CHARGES,
            p.PRODUCT_ID,
            p.PRODUCT_CATEGORY_NAME,
            p.PRODUCT_WEIGHT_G as Item_Weight

from ELT_STAGE.ORDERS_DW o
    inner join ELT_STAGE.ORDER_ITEMS_DW oi on o.order_id = oi.order_id
    inner join ELT_STAGE.PRODUCTS_DW p on oi.product_id = p.product_id
);

-- Check to make sure the table is created and outputs correctly
Select *
from Orders_Info_Table;

---------------------------------------------------------------------------
-
------------------- CREATION OF THE GOLD LAYER ---------------------------
-
---------------------------------------------------------------------------
-


CREATE or REPLACE SCHEMA EDW_GOLD_LAYER
COMMENT = 'This schema is used to create the "Gold Layer"';

-- 2. Grant proper privileges on the Gold schema to SYSADMIN

USE ROLE SECURITYADMIN;

GRANT OWNERSHIP ON SCHEMA tmg2546_DW_CAPSTONE.EDW_GOLD_LAYER TO SYSADMIN;


CREATE OR REPLACE TABLE
edw_gold_layer.Customer_Transactions_and_Orders_Info as (
Select      ct.CUSTOMER_ID,
            ct.ADDRESS,
            ct.ORDER_ID,
            ct.PAYMENT_TYPE,
            ct.PAYMENT_INSTALLMENTS,
            ct.PAYMENT_VALUE,
            oit.DAYS_TO_CUSTOMER,
            oit.SELLER_ID,
            oit.PRODUCT_ID,
            oit.PRICE,
            oit.SHIPPING_CHARGES,
            oit.PRODUCT_CATEGORY_NAME,
            oit.ITEM_WEIGHT

from edw_silver_layer.CUSTOMER_TRANSACTIONS ct
    inner join edw_silver_layer.ORDERS_INFO_TABLE oit on ct.ORDER_ID =
oit.ORDER_ID
);

use ROLE SYSADMIN;
use SCHEMA edw_gold_layer;

-- Use this statement to make sure the table is created correctly --
-- select *
-- from Customer_Transactions_and_Orders_Info;

-- This statement will sort instances where the days_to_customer is over
10 - we might want to look at these and see if we can speed them up so
customers who are paying the most for our products are satisfied --

select
    customer_id,
    order_id,
    days_to_customer,
    price,
    product_category_name,
    shipping_charges
from customer_transactions_and_orders_info
where days_to_customer > 10
order by price desc;
```