**ETL Implementation**

**Complete ETL Code & Workflows**

**ETL Process Overview**

The **ETL (Extract, Transform, Load)** process involves:

1. **Extracting** raw data from the staging area.
2. **Transforming** the data into a structured and clean format.
3. **Loading** the transformed data into the **dimensional model** (fact & dimension tables).

**ETL Code: Extract, Transform & Load**

**Extract - Loading Data from Staged Files**

-- Load customer data into staging

COPY INTO stage\_customer

FROM @coffee\_shop\_stage/customer.csv

FILE\_FORMAT = (TYPE = 'CSV', SKIP\_HEADER = 1)

ON\_ERROR = 'CONTINUE';

-- Load product data into staging

COPY INTO stage\_product

FROM @coffee\_shop\_stage/product.csv

FILE\_FORMAT = (TYPE = 'CSV', SKIP\_HEADER = 1)

ON\_ERROR = 'CONTINUE';

-- Load store data into staging

COPY INTO stage\_store

FROM @coffee\_shop\_stage/sales\_outlet.csv

FILE\_FORMAT = (TYPE = 'CSV', SKIP\_HEADER = 1)

ON\_ERROR = 'CONTINUE';

-- Load staff data into staging

COPY INTO stage\_staff

FROM @coffee\_shop\_stage/staff.csv

FILE\_FORMAT = (TYPE = 'CSV', SKIP\_HEADER = 1)

ON\_ERROR = 'CONTINUE';

-- Load sales transaction data into staging

COPY INTO stage\_sales

FROM @coffee\_shop\_stage/sales\_receipts.csv

FILE\_FORMAT = (TYPE = 'CSV', SKIP\_HEADER = 1)

ON\_ERROR = 'CONTINUE';

-- Load date data into staging

COPY INTO stage\_date

FROM @coffee\_shop\_stage/Dates.csv

FILE\_FORMAT = (TYPE = 'CSV', SKIP\_HEADER = 1)

ON\_ERROR = 'CONTINUE';

**Transform - Cleaning & Structuring Data**

**Transforming dim\_customer**

INSERT INTO dim\_customer (customer\_id, first\_name, last\_name, email, phone\_number, loyalty\_member\_yn, birth\_date, gender)

SELECT DISTINCT

customer\_id, first\_name, last\_name, email, phone\_number, loyalty\_member\_yn, birth\_date, gender

FROM stage\_customer;

**Transforming dim\_product**

INSERT INTO dim\_product (product\_id, product\_group, product\_category, product\_type, product, product\_description, unit\_of\_measure,

current\_wholesale\_price, current\_retail\_price, tax\_exempt\_yn, promo\_yn, new\_product\_yn)

SELECT DISTINCT

product\_id, product\_group, product\_category, product\_type, product, product\_description, unit\_of\_measure,

TO\_NUMBER(REPLACE(current\_wholesale\_price, '$', '')),

TO\_NUMBER(REPLACE(current\_retail\_price, '$', '')),

tax\_exempt\_yn, promo\_yn, new\_product\_yn

FROM stage\_product;

**Transforming dim\_store**

INSERT INTO dim\_store (store\_id, store\_name, location, store\_manager)

SELECT DISTINCT

store\_id, store\_name, location, store\_manager

FROM stage\_store;

**Transforming dim\_staff**

INSERT INTO dim\_staff (staff\_id, first\_name, last\_name, position, start\_date, assigned\_store\_id)

SELECT DISTINCT

staff\_id, first\_name, last\_name, position, start\_date,

CASE

WHEN assigned\_store\_id = 'HQ' THEN 999

WHEN assigned\_store\_id = 'WH' THEN 888

ELSE assigned\_store\_id::BIGINT

END

FROM stage\_staff;

**Transforming dim\_date**

INSERT INTO dim\_date (date\_id, full\_date, year, month, day, weekday)

SELECT

TO\_NUMBER(TO\_CHAR(full\_date, 'YYYYMMDD')) AS date\_id,

full\_date,

YEAR(full\_date),

MONTH(full\_date),

DAY(full\_date),

DAYNAME(full\_date)

FROM stage\_date;

**Transforming fact\_sales**

INSERT INTO fact\_sales (transaction\_id, customer\_id, product\_id, store\_id, date\_id, staff\_id, quantity\_sold, total\_sales\_amount)

SELECT DISTINCT

transaction\_id, customer\_id, product\_id, store\_id,

TO\_NUMBER(TO\_CHAR(transaction\_date, 'YYYYMMDD')) AS date\_id,

staff\_id, quantity AS quantity\_sold, line\_item\_amount AS total\_sales\_amount

FROM stage\_sales;

**Handling ETL Challenges (Data Quality, SCD, etc.)**

**Handling Data Type Mismatches**

* **Issue:** Currency fields in product.csv were stored as **text** (e.g., "$18.00").
* **Solution:** Used REPLACE() and TO\_NUMBER() to convert them into **decimal values**.

SELECT TO\_NUMBER(REPLACE(current\_wholesale\_price, '$', '')) AS wholesale\_price

FROM stage\_product;

**Handling Missing & Invalid Data**

* **Issue:** Some staff.csv records had HQ and WH instead of store IDs.
* **Solution:** Mapped these to **999 (Headquarters)** and **888 (Warehouse)**.

CASE

WHEN assigned\_store\_id = 'HQ' THEN 999

WHEN assigned\_store\_id = 'WH' THEN 888

ELSE assigned\_store\_id::BIGINT

END

**Ensuring Referential Integrity (Foreign Key Checks)**

* **Issue:** Transactions in fact\_sales referenced **non-existent products or customers**.
* **Solution:** Checked for missing foreign keys using:

SELECT DISTINCT product\_id

FROM fact\_sales

WHERE product\_id NOT IN (SELECT product\_id FROM dim\_product);

**Execution Logs/Screenshots**

A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.A screenshot of a computer

AI-generated content may be incorrect.