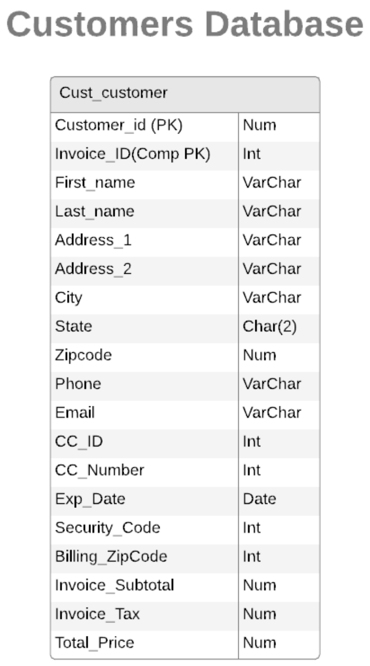
Debleena Das, Noah Placke, Mounika Tarigopula, Sarah Teng, Katherine Wroble
Data Management - Tej Anand - Fall 2020

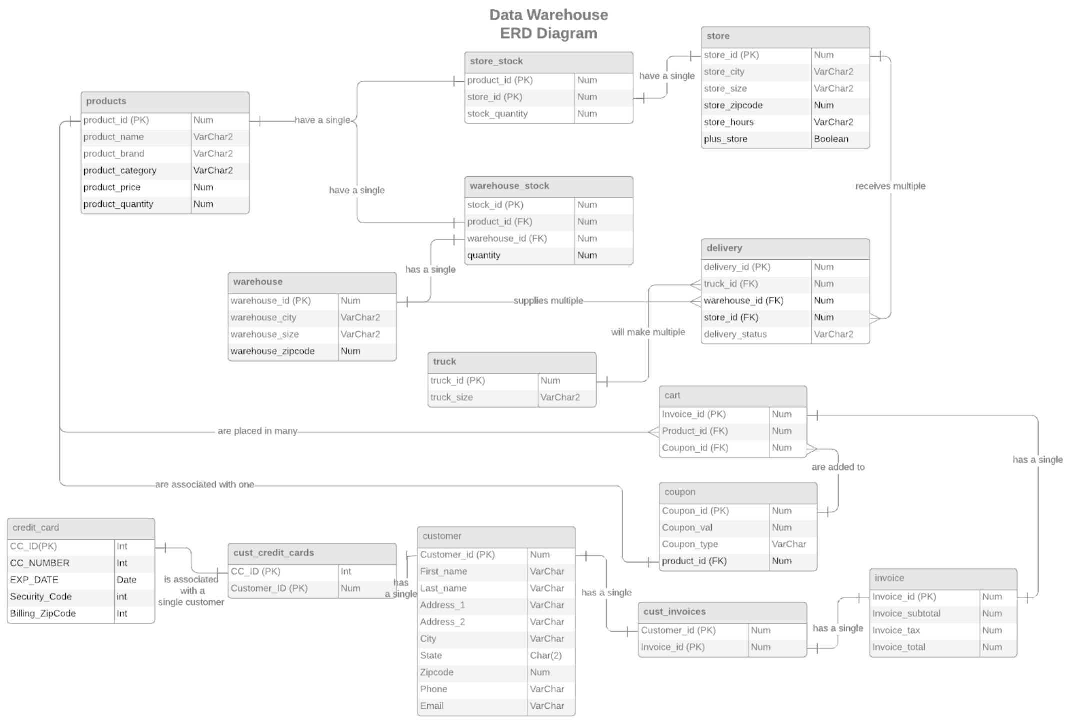
Data Management Project Final Deliverable

1. LucidChart Models



Customers Database

Cust_customer	
Customer_id (PK)	Num
Invoice_ID(Comp PK)	Int
First_name	VarChar
Last_name	VarChar
Address_1	VarChar
Address_2	VarChar
City	VarChar
State	Char(2)
Zipcode	Num
Phone	VarChar
Email	VarChar
CC_ID	Int
CC_Number	Int
Exp_Date	Date
Security_Code	Int
Billing_ZipCode	Int
Invoice_Subtotal	Num
Invoice_Tax	Num
Total_Price	Num



Data Warehouse ERD Diagram

store_stock	
product_id (PK)	Num
store_id (PK)	Num
stock_quantity	Num

store	
store_id (PK)	Num
store_city	VarChar2
store_size	VarChar2
store_zipcode	Num
store_hours	VarChar2
plus_store	Boolean

products	
product_id (PK)	Num
product_name	VarChar2
product_brand	VarChar2
product_category	VarChar2
product_price	Num
product_quantity	Num

warehouse_stock	
stock_id (PK)	Num
product_id (FK)	Num
warehouse_id (FK)	Num
quantity	Num

delivery	
delivery_id (PK)	Num
truck_id (FK)	Num
warehouse_id (FK)	Num
store_id (FK)	Num
delivery_status	VarChar2

warehouse	
warehouse_id (PK)	Num
warehouse_city	VarChar2
warehouse_size	VarChar2
warehouse_zipcode	Num

truck	
truck_id (PK)	Num
truck_size	VarChar2

cart	
Invoice_id (PK)	Num
Product_id (FK)	Num
Coupon_id (FK)	Num

coupon	
Coupon_id (PK)	Num
Coupon_val	Num
Coupon_type	VarChar
product_id (FK)	Num

credit_card	
CC_ID(PK)	Int
CC_NUMBER	Int
EXP_DATE	Date
Security_Code	int
Billing_ZipCode	Int

cust_credit_cards	
CC_ID (PK)	Int
Customer_ID (PK)	Num

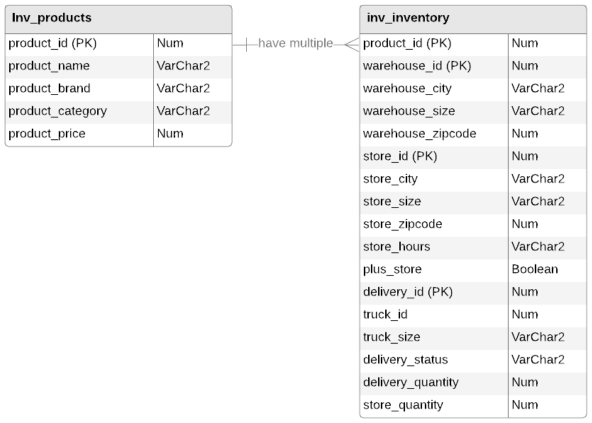
customer	
Customer_id (PK)	Num
First_name	VarChar
Last_name	VarChar
Address_1	VarChar
Address_2	VarChar
City	VarChar
State	Char(2)
Zipcode	Num
Phone	VarChar
Email	VarChar

cust_invoices	
Customer_id (PK)	Num
Invoice_id (PK)	Num

invoice	
Invoice_id (PK)	Num
Invoice_subtotal	Num
Invoice_tax	Num
Invoice_total	Num

Debleena Das, Noah Placke, Mounika Tarigopula, Sarah Teng, Katherine Wroble
Data Management - Tej Anand - Fall 2020

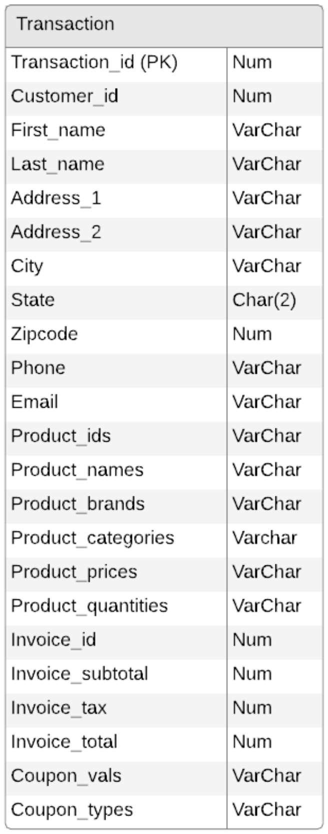
Inventory Database



Inv_products	
product_id (PK)	Num
product_name	VarChar2
product_brand	VarChar2
product_category	VarChar2
product_price	Num

inv_inventory	
product_id (PK)	Num
warehouse_id (PK)	Num
warehouse_city	VarChar2
warehouse_size	VarChar2
warehouse_zipcode	Num
store_id (PK)	Num
store_city	VarChar2
store_size	VarChar2
store_zipcode	Num
store_hours	VarChar2
plus_store	Boolean
delivery_id (PK)	Num
truck_id	Num
truck_size	VarChar2
delivery_status	VarChar2
delivery_quantity	Num
store_quantity	Num

Transactions Database



Transaction	
Transaction_id (PK)	Num
Customer_id	Num
First_name	VarChar
Last_name	VarChar
Address_1	VarChar
Address_2	VarChar
City	VarChar
State	Char(2)
Zipcode	Num
Phone	VarChar
Email	VarChar
Product_ids	VarChar
Product_names	VarChar
Product_brands	VarChar
Product_categories	Varchar
Product_prices	VarChar
Product_quantities	VarChar
Invoice_id	Num
Invoice_subtotal	Num
Invoice_tax	Num
Invoice_total	Num
Coupon_vals	VarChar
Coupon_types	VarChar

2. DDL for 4 Models

--TRANSACTION TABLE

```
DROP TABLE transaction;

CREATE TABLE transaction (
   transaction_id      NUMBER(10)      PRIMARY KEY,
   customer_id         NUMBER(10),
   first_name          VARCHAR(40),
   last_name           VARCHAR(40),
   address_line1       VARCHAR(50),
   address_line2       VARCHAR(50),
   city                VARCHAR(50),
   state               CHAR(2),
   zipcode             CHAR(5),
   phone               NUMBER(10),
   email               VARCHAR(75),
   product_ids         VARCHAR(100),
   product_names       VARCHAR(100),
   product_brands      VARCHAR(100),
   product_categories  VARCHAR(100),
   product_prices      VARCHAR(50),
   product_quantities  VARCHAR(10),
   invoice_id          NUMBER(10),
   invoice_subtotal    NUMBER(10),
   invoice_tax         NUMBER(10),
   invoice_total       NUMBER(10),
   coupon_vals         VARCHAR(50),
   coupon_types        VARCHAR(100)
);

--Inserting Data

INSERT INTO transaction
VALUES(00001, 1000, 'Katherine', 'Wroble', '7204 Balmoral Dr', null, 'Colleyville', 'TX', '76034',
8177218994, 'kathwrobs@gmail.com', '001, 002, 003', 'Bananas, Milk, Eggs', 'Dole, HEB, HEB','Fruit,
Dairy, Dairy', '1.00, 2.00, 1.50', '1, 1, 1', 1111, 5.00, .50, 5.50, null, null);

INSERT INTO transaction
VALUES(00002, 1001, 'Debleena', 'Das', '1234 Austin Dr', null, 'Pflugerville', 'TX', '78738', 5127484729,
'deb@gmail.com', '004, 002, 005', 'Orange Juice, Milk, Bread', 'Simply, HEB, Bimbo', 'Juice, Dairy, Bread',
'1.50, 2.00, 2.00', '2, 1, 3', 1112, 11.00, 1.00, 12.00, 1.00, 'Discount');

INSERT INTO transaction
```

```
VALUES(00003, 1002, 'Mounika', 'Tarigopula', '2029 Fun Ln', null, 'Dallas', 'TX', '75638', 5126373637,
'mounikaaa@gmail.com', '006, 100, 400', 'Chocolate Bar, Bananas, Orange Juice', 'Hersheys, Chiquita,
Sunny D', 'Candy, Fruit, Juice', '1.00, 1.00, 2.00', '4, 1, 2', 1113, 9.00, .75, 9.75, 2.00, 'BOGO');

INSERT INTO transaction
VALUES(00004, 1003, 'Noah', 'Placke', '904 Bevo Bucks Rd', null, 'Austin', 'TX', '78705', 8987487465,
'np@gmail.com', '008, 009, 010', 'Eggnog, Almonds, Salmon', 'Southern Comfort, HEB, Chicken of the
Sea', 'Dairy, Snack, Fish', '3.00, 5.00, 10.00', '2, 1, 1', 1114, 21.00, 3.00, 24.00, 5.00, 'Discount');

INSERT INTO transaction
VALUES(00005, 1004, 'Sarah', 'Teng', '12304 Longhorn Rd', null, 'Georgetown', 'TX', '23847',
5123787773, 'sarahteng@gmail.com', '009, 011, 034', 'Cookies, Blueberries, Salad', 'Oreo, Driscols,
HEB','Sweets, Fruit, Produce', '5.00, 3.00, 7.50', '1, 4, 1', 1115, 18.00, 8.50, 25.50, null, null);

INSERT INTO transaction
VALUES(00006, 1005, 'Han Yi', 'Jiang', '1909 Speedway Ln', null, 'Austin', 'TX', '78705', 5123787773,
'hanyiiiiii@gmail.com', '090, 190, 384', 'Red Solo Cups, Beer, Ping Pong Balls', 'Solo, Bud Light,
HEB','Plasticware, Alcohol, Games', '5.00, 20.00, 2.00', '1, 2, 1', 1116, 28.00, 3.00, 30.68, null, null);
```

--INVENTORY TABLES

```
DROP TABLE inv_products;
DROP TABLE inv_inventory;

CREATE TABLE inv_products (
   product_id      NUMBER(10) PRIMARY KEY,
   product_name    VARCHAR(40),
   product_brand   VARCHAR(40),
   product_category VARCHAR(50),
   product_price   NUMBER(38)
   );


CREATE TABLE inv_inventory (
   product_id     NUMBER(10),
   warehouse_id   NUMBER(10),
   store_id       NUMBER(38),
   delivery_id    NUMBER(10),
   truck_id       NUMBER(38),
   warehouse_city VARCHAR(40),
   warehouse_size VARCHAR(40),
   warehouse_zipcode NUMBER(10),
   store_city     VARCHAR(50),
   store_size     VARCHAR(50),
   store_zipcode  NUMBER(20),
   store_hours    VARCHAR(50),
   plus_store     NUMBER(1),
   truck_size     VARCHAR(50),
   delivery_status VARCHAR(50),
   delivery_quantity NUMBER(30),
   store_quantity  NUMBER(30)
  );

ALTER TABLE inv_inventory
   ADD CONSTRAINT inv_inventory_pk PRIMARY KEY (warehouse_id, store_id, delivery_id, truck_id);
ALTER TABLE inv_inventory
   ADD CONSTRAINT inv_delivery_fk1 FOREIGN KEY (product_id)
      REFERENCES inv_products (product_id);


INSERT INTO inv_products
VALUES(101, 'Dark Chocolate', 'Hershey', 'candy', 10);

INSERT INTO inv_products
VALUES(102, 'BBQ Lays', 'Lays', 'snacks', 5);

INSERT INTO inv_products
VALUES(103, 'Pepperoni Pizza', 'Digiornos', 'frozen', 12);

INSERT INTO inv_inventory
VALUES(101, 20001, 1, 801, 5001, 'Austin', 2000000, 78723, 'Austin', 10000, 78751, '6AM-11PM', 1,
50000, 'delivered', 1000, 500);

INSERT INTO inv_inventory
VALUES(102, 20002, 2, 802, 5002, 'Austin', 2000000, 78759, 'Austin', 10000, 78756, '6AM-11PM', 0,
40000, 'on the way', 700, 200);

INSERT INTO inv_inventory
VALUES(103, 20003, 3, 803, 5003, 'Austin', 1000000, 78705, 'Austin', 20000, 78723, '6AM-11PM', 1,
30000, 'not yet started', 100, 40);
```

--CUSTOMER TABLE

```
CREATE TABLE CUSTOMER_TDB(
CUSTOMER_ID INT,
INVOICE_ID Int,
FIRSTNAME VARCHAR(20) NOT NULL,
LASTNAME VARCHAR(20) NOT NULL,
PHONE INT NOT NULL,
EMAIL VARCHAR(30) NOT NULL,
HOME_ADDRESS VARCHAR(50) NOT NULL,
CC_ID INT,
CC_NUMBER INT NOT NULL,
EXP_DATE DATE NOT NULL,
SECURITY_CODE INT NOT NULL,
BILLING_ZIPCODE INT NOT NULL,
INVOICE_SUBTOTAL NUMBER NOT NULL,
INVOICE_TAX NUMBER NOT NULL,
TOTAL_PRICE NUMBER NOT NULL,
PRIMARY KEY(CUSTOMER_ID,INVOICE_ID)
);
```

-- DATA WAREHOUSE

```
CREATE TABLE customer(
customer_id             NUMBER          PRIMARY KEY,
first_name              VARCHAR2(30),
last_name               VARCHAR2(40),
address_1               VARCHAR2(40),
address_2               VARCHAR2(15),
city                    VARCHAR2(30),
state                   CHAR(2),
zipcode         NUMBER(5),
phone                   NUMBER,
email                   VARCHAR2(40)
);

CREATE TABLE credit_card(
cc_id                   NUMBER          PRIMARY KEY,
cc_number               NUMBER,
exp_date                DATE,
security_code           NUMBER,
billing_zipcode  NUMBER
);

CREATE TABLE cust_credit_cards(
cc_id                   NUMBER          REFERENCES credit_card(cc_id),
customer_id             NUMBER          REFERENCES customer(customer_id),
CONSTRAINT cc_pk PRIMARY KEY (cc_id, customer_id)
);

CREATE TABLE invoice(
invoice_id              NUMBER          PRIMARY KEY,
invoice_subtotal        NUMBER,
invoice_tax             DATE,
invoice_total           NUMBER
);

CREATE TABLE cust_invoices(
customer_id             NUMBER          REFERENCES customer(customer_id),
invoice_id              NUMBER          REFERENCES invoice(invoice_id),
CONSTRAINT cust_invoice_pk PRIMARY KEY (customer_id, invoice_id)
);

CREATE TABLE products(
product_id              NUMBER          PRIMARY KEY,
product_name            VARCHAR2(40),
product_brand           VARCHAR2(40),
product_category        VARCHAR2(30),
product_price           NUMBER,
product_quantity        NUMBER
);

CREATE TABLE coupon(
coupon_id               NUMBER          PRIMARY KEY,
coupon_val              NUMBER,
coupon_type             VARCHAR2(30),
product_id              NUMBER          REFERENCES products(product_id)
);


CREATE TABLE store(
store_id                NUMBER          PRIMARY KEY,
store_city              VARCHAR2(40),
store_size              VARCHAR2(10),
store_zipcode           NUMBER,
Store_hours             VARCHAR2(10)
Plus_store              BOOLEAN
);

CREATE TABLE store_stock(
product_id              NUMBER          REFERENCES products(product_id),
store_id                NUMBER          REFERENCES store(store_id),
stock_quantity          NUMBER,
CONSTRAINT store_stock_pk PRIMARY KEY (product_id, store_id)
);

CREATE TABLE warehouse(
warehouse_id            NUMBER          PRIMARY KEY,
warehouse_cityVARCHAR2(40),
warehouse_sizeVARCHAR2(10),
warehouse_zipcode       NUMBER
);

CREATE TABLE warehouse_stock(
stock_id                NUMBER          PRIMARY KEY,
product_id              NUMBER          REFERENCES products(product_id),
warehouse_id            NUMBER          REFERENCES warehouse(warehouse_id),
quantity                NUMBER
);
```

```
CREATE TABLE truck(
truck_id                NUMBER          PRIMARY KEY,
truck_size              VARCHAR2(10)
);

CREATE TABLE delivery(
delivery_id             NUMBER          PRIMARY KEY,
truck_id                NUMBER          REFERENCES truck(truck_id),
warehouse_id            NUMBER          REFERENCES warehouse(warehouse_id),
store_id                NUMBER          REFERENCES store(store_id),
delivery_status  VARCHAR2(20)
);
```

3. ETL to populate data models:

https://colab.research.google.com/drive/1_qdcMep94BoBrdFLUAf04V26nzyYr109?authuser=1

This is a google colab file.

```
# -*- coding: utf-8 -*-
"""DM.ipynb

Automatically generated by Colaboratory.

Original file is located at
   https://colab.research.google.com/drive/1_qdcMep94BoBrdFLUAf04V26nzyYr109
"""

! ls -l oracle-instantclient*-basiclite-*.rpm || wget
https://yum.oracle.com/repo/OracleLinux/OL7/oracle/instantclient/x86_64/getPackage/oracle-instantcl
ient19.3-basiclite-19.3.0.0.0-1.x86_64.rpm
! sudo apt-get install alien libaio1
! sudo alien -i oracle-instantclient19.3-basiclite-19.3.0.0.0-1.x86_64.rpm
!pip install cx_Oracle

import numpy
import pandas as pd
import cx_Oracle

con = cx_Oracle.connect("hj7627", "Andyellis888",
cx_Oracle.makedsn("msb-msitm.austin.utexas.edu",1521,'ORCL'))
cur = con.cursor()

#con.commit()

"""## LOAD DEFINITIONS"""

def insert_CREDITCARD(result):
   sql = ("INSERT INTO
CREDIT_CARD(CC_ID,CC_NUMBER,EXP_DATE,SECURITY_CODE,BILLING_ZIPCODE)values(:CC_ID,:CC_NU
MBER,:EXP_DATE,:SECURITY_CODE,:BILLING_ZIPCODE)")
   cur.executemany(sql,result)
   con.commit()

def insert_invoice_customer(result):
   sql = ("INSERT INTO
CUST_INVOICES(CUSTOMER_ID,INVOICE_ID)values(:CUSTOMER_ID,:INVOICE_ID)")
   cur.executemany(sql,result)
   con.commit()
```

```
def insert_credit_cust(result):
   sql = ("INSERT INTO CUST_CREDIT_CARDS(CUSTOMER_ID,CC_ID)values(:CUSTOMER_ID,:CC_ID)")
   cur.executemany(sql,result)
   con.commit()

def insert_invoice(result):
   sql = ("INSERT INTO
INVOICE(INVOICE_ID,INVOICE_SUBTOTAL,INVOICE_TAX,INVOICE_TOTAL)values(:INVOICE_ID,:INVOICE_S
UBTOTAL,:INVOICE_TAX,:INVOICE_TOTAL)")
   cur.executemany(sql,result)
   con.commit()

def insert_customer(result):
   sql = ("INSERT INTO
CUSTOMER(CUSTOMER_ID,FIRSTNAME,LASTNAME,PHONE,EMAIL,HOME_ADDRESS)values(:CUSTOMER_
ID,:FIRSTNAME,:LASTNAME,:PHONE,:EMAIL,:HOME_ADDRESS)")
   cur.executemany(sql,result)
   con.commit()

def insert_products(result):
   sql = ("INSERT INTO
PRODUCTS(PRODUCT_ID,PRODUCT_NAME,PRODUCT_BRAND,PRODUCT_CATEGORY,PRODUCT_PRICE)v
alues(:PRODUCT_ID,:PRODUCT_NAME,:PRODUCT_BRAND,:PRODUCT_CATEGORY,:PRODUCT_PRICE)")
   cur.executemany(sql,result)
   con.commit()

def insert_warehouse(result):
   sql = ("INSERT INTO
warehouse(WAREHOUSE_ID,WAREHOUSE_CITY,WAREHOUSE_SIZE,WAREHOUSE_ZIPCODE)values(:WAR
EHOUSE_ID,:WAREHOUSE_CITY,:WAREHOUSE_SIZE,:WAREHOUSE_ZIPCODE)")
   cur.executemany(sql,result)
   con.commit()

def insert_truck(result):
   sql = ("INSERT INTO truck(TRUCK_ID,TRUCK_SIZE)values(:TRUCK_ID,:TRUCK_SIZE)")
   cur.executemany(sql,result)
   con.commit()

def insert_delivery(result):
   sql = ("INSERT INTO
DELIVERY(DELIVERY_ID,TRUCK_ID,WAREHOUSE_ID,STORE_ID,DELIVERY_STATUS)values(:DELIVERY_ID,:T
RUCK_ID,:WAREHOUSE_ID,:STORE_ID,:DELIVERY_STATUS)")
   cur.executemany(sql,result)
   con.commit()

def insert_store(result):
    sql = ("INSERT INTO
STORE(STORE_ID,STORE_CITY,STORE_SIZE,STORE_ZIPCODE,STORE_HOURS,PLUS_STORE)values(:STORE_
ID,:STORE_CITY,:STORE_SIZE,:STORE_ZIPCODE,:STORE_HOURS,:PLUS_STORE)")
    cur.executemany(sql,result)
    con.commit()

def insert_store_stock(result):
    sql = ("INSERT INTO
STORE_STOCK(PRODUCT_ID,STORE_ID,STOCK_QUANTITY)values(:PRODUCT_ID,:STORE_ID,:STOCK_QUA
NTITY)")
    cur.executemany(sql,result)
    con.commit()

"""## EXTRACT and LOAD

###CUSTOMER_TDB
"""

creditcard_result = []
for row in cur.execute("SELECT CC_ID,CC_NUMBER,EXP_DATE,SECURITY_CODE,BILLING_ZIPCODE FROM
CUSTOMER_TDB"):
    creditcard_result.append(row)
creditcard_result

insert_CREDITCARD(creditcard_result)

customer_result = []
for row in cur.execute("SELECT CUSTOMER_ID,FIRSTNAME,LASTNAME,PHONE,EMAIL,HOME_ADDRESS
FROM CUSTOMER_TDB"):
    customer_result.append(row)

insert_customer(customer_result)

cust_credit_result = []
for row in cur.execute("SELECT CUSTOMER_ID,CC_ID FROM CUSTOMER_TDB"):
    cust_credit_result.append(row)
cust_credit_result

insert_credit_cust(cust_credit_result)

"""### TRANSACTION_TDB"""

tr_cust_result = []
for row in cur.execute("SELECT CUSTOMER_ID,FIRST_NAME,LAST_NAME,PHONE,EMAIL,ADDRESS_LINE1
FROM TRANSACTION_TDB"):
    tr_cust_result.append(row)

insert_customer(tr_cust_result)
```

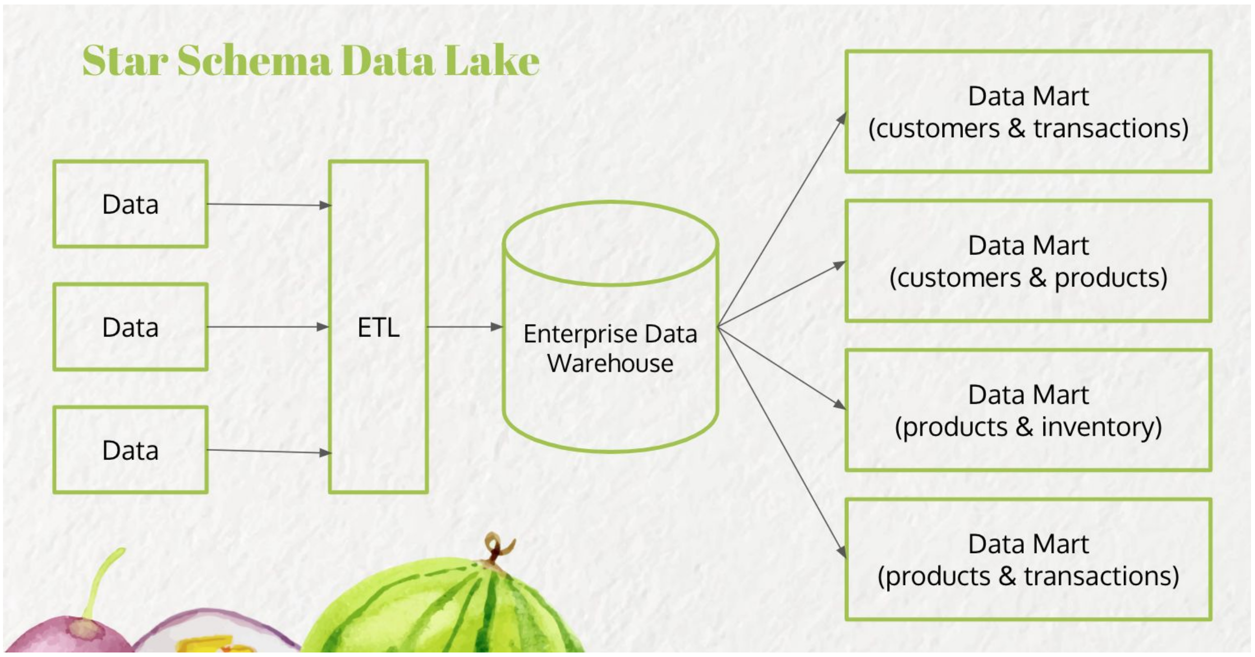
```
invoice_result = []
for row in cur.execute("SELECT INVOICE_ID,INVOICE_SUBTOTAL,INVOICE_TAX,INVOICE_TOTAL FROM 
TRANSACTION_TDB"):
   invoice_result.append(row)
invoice_result

insert_invoice(invoice_result)

invoice_customer_result = []
for row in cur.execute("SELECT CUSTOMER_ID,INVOICE_ID FROM TRANSACTION_TDB"):
   invoice_customer_result.append(row)
invoice_customer_result

insert_invoice_customer(invoice_customer_result)

tr_product_result = []
for row in cur.execute("SELECT 
PRODUCT_IDS,PRODUCT_NAMES,PRODUCT_BRANDS,PRODUCT_CATEGORIES,PRODUCT_PRICES FROM 
TRANSACTION_TDB"):
   tr_product_result.append(row)
tr_product_result

#Change strings in all tuples to list
final = []
for i in tr_product_result:
   new = []
   for j in range(len(i)):
      new.append(i[j].split(','))
   final.append(new)
final

#put the everything to tuple
final2 = []
for data in final:
   for i in range(len(data[0])):
      final2.append(tuple([int(data[0][i]),data[1][i],data[2][i],data[3][i],float(data[4][i])]))
print(final2)

def removeDuplicates(lst):
   return list(set([i for i in lst]))

#remove duplicates
tr_product_result2 = removeDuplicates(final2)

tr_product_result2

insert_products(tr_product_result2)
```

```
"""### INV_INVENTORY_TDB"""

# cur.execute("ALTER TABLE PRODUCTS DROP COLUMN PRODUCT_QUANTITY")
# con.commit

product_result = []
for row in cur.execute("SELECT * FROM INV_PRODUCTs_TDB"):
   product_result.append(row)
product_result

insert_products(product_result)

warehouse_result = []
for row in cur.execute("SELECT
WAREHOUSE_ID,WAREHOUSE_CITY,WAREHOUSE_SIZE,WAREHOUSE_ZIPCODE FROM
INV_INVENTORY_TDB"):
   warehouse_result.append(row)
warehouse_result

insert_warehouse(warehouse_result)

# DROP WAREHOUSE_STOCK

truck_result = []
for row in cur.execute("SELECT TRUCK_ID,TRUCK_SIZE FROM INV_INVENTORY_TDB"):
   truck_result.append(row)
truck_result

insert_truck(truck_result)

store_result = []
for row in cur.execute("SELECT
STORE_ID,STORE_CITY,STORE_SIZE,STORE_ZIPCODE,STORE_HOURS,PLUS_STORE FROM
INV_INVENTORY_TDB"):
   store_result.append(row)
store_result

insert_store(store_result)

delivery_result = []
for row in cur.execute("SELECT DELIVERY_ID,TRUCK_ID,WAREHOUSE_ID,STORE_ID,DELIVERY_STATUS
FROM INV_INVENTORY_TDB"):
   delivery_result.append(row)
delivery_result

insert_delivery(delivery_result)
```

```
store_stock_result = []
for row in cur.execute("SELECT PRODUCT_ID,STORE_ID,STORE_QUANTITY FROM
INV_INVENTORY_TDB"):
    store_stock_result.append(row)
store_stock_result

insert_store_stock(store_stock_result)
```

4. Model for data lake:

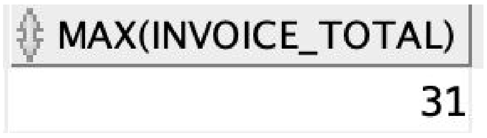
```
CREATE VIEW inv_cust_details_vw AS
  SELECT
    cust_invoices.invoice_id,
    invoice_subtotal,
    invoice_tax,
    invoice_total,
    cust_invoices.customer_id,
    firstname,
    lastname,
    phone,
    email,
    home_address,
    cust_credit_cards.cc_id,
    cc_number,
    exp_date,
    security_code,
    billing_zipcode
  FROM
    invoice
    LEFT JOIN cust_invoices ON invoice.invoice_id = cust_invoices.invoice_id
    LEFT JOIN customer ON customer.customer_id = cust_invoices.customer_id
    LEFT JOIN cust_credit_cards ON customer.customer_id = cust_credit_cards.customer_id
    LEFT JOIN credit_card ON cust_credit_cards.cc_id = credit_card.cc_id;
```


Star Schema Data Lake
Data
Data
Data
ETL
Enterprise Data Warehouse
Data Mart (customers & transactions)
Data Mart (customers & products)
Data Mart (products & inventory)
Data Mart (products & transactions)

5. Three types of analytics:

1) What is the maximum invoice total for customers at H-E-B?

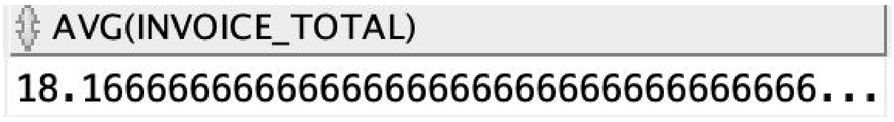
```
SELECT
    max(invoice.invoice_total)
  FROM
    invoice
    LEFT JOIN cust_invoices ON invoice.invoice_id = cust_invoices.invoice_id
    LEFT JOIN customer ON customer.customer_id = cust_invoices.customer_id
    LEFT JOIN cust_credit_cards ON customer.customer_id = cust_credit_cards.customer_id
    LEFT JOIN credit_card ON cust_credit_cards.cc_id = credit_card.cc_id;
```



```
MAX(INVOICE_TOTAL)
                31
```

2) **What is the average invoice total for customers at H-E-B?**

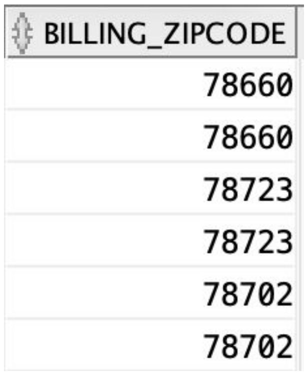
```
SELECT
    avg(invoice.invoice_total)
  FROM
    invoice
    LEFT JOIN cust_invoices ON invoice.invoice_id = cust_invoices.invoice_id
    LEFT JOIN customer ON customer.customer_id = cust_invoices.customer_id
    LEFT JOIN cust_credit_cards ON customer.customer_id = cust_credit_cards.customer_id
    LEFT JOIN credit_card ON cust_credit_cards.cc_id = credit_card.cc_id;
```



```
AVG(INVOICE_TOTAL)
18.16666666666666666666666666666666...
```

3) **What regions (or zipcodes) are H-E-B customers from?**

```
  SELECT
    billing_zipcode
  FROM
    invoice
    LEFT JOIN cust_invoices ON invoice.invoice_id = cust_invoices.invoice_id
    LEFT JOIN customer ON customer.customer_id = cust_invoices.customer_id
    LEFT JOIN cust_credit_cards ON customer.customer_id = cust_credit_cards.customer_id
    LEFT JOIN credit_card ON cust_credit_cards.cc_id = credit_card.cc_id;
```



BILLING_ZIPCODE
78660
78660
78723
78723
78702
78702