**Part 2 SEIS 630: Database Project**

**Introduction:**

This project focuses on a database of logistic company, which provides services like carrying goods from one place to another, in this database I have taken the data of 20 drivers who have completed 1000 trips in a period of one month. The reason to choose this topic was, by considering the logistic company which runs in India, I wanted to see how the companies calculate incentives at once for the group of vehicles running, The minimal requirements of my project was to create the required tables and obtain data which is required for querying the database for calculating incentives, my optimistic goal was to query the same database in snowflake.

**Exploring:**

Initially, I didn’t had any experience with databases, everything was new for me, learnt everything from scratch with the help of class and other external sources. The topic which I have selected (Logistic Database) was my initial idea, this was the first thing which popped up in the back of my head when I started thinking about final project, after considering I wondered whether to choose a different topic which might have more scope, but after careful consideration, I finalized my initial topic as my final topic. I used my personal experience where I have seen the company previously as my source of information for selecting the topic.

**Building:**

While I started building my database, I wrote down every possible table that I wanted to create with their respective columns, I used mockaroo to generate my data where I learnt a lot on how to generate random data by configuring the python like functions in mockaroo. I tried running queries on incentives for drivers. But I encountered one of the biggest challenges, that is creation of trigger, which I did because I forgot the basics of normalization in a hurry, once I got clarified with those things everything was good.

**Discovering:**

As mentioned earlier, I started from very basic after starting this course, but now I am very much confident about myself that I have explored much in the way, I could create my own database with the knowledge acquired, I had learnt how to query the data using Oracle Data Modeler, but in this project as I am using snowflake to query the data, before going to demonstrate this in front the class I wanted to learn snowflake, so that is one of the important thing I learnt from this project.

**Topics from class**

1. **Normalization:**

As mentioned earlier, I made one of the major errors in the process of inserting data into tables where I found the problem of duplication of primary keys, but after clarifying these, I practically learnt the normalization process.

1. **Querying the data:**

After the creation of tables and once database is well structured, how we get things that is required from the database is important, I learnt querying the data from class which helped me here to get the incentives from the database created by querying it.

1. **Data types:**

Creating tables with required columns where we must assign their respective types, plays an important role, identifying them and assigning them was a challenging task, I learnt the difference between them which resulted in creation of proper database design.

1. **Primary Keys and Foreign Keys:**

Assigning the primary keys and foreign keys could be a difficult task where there should be no duplication of values in primary keys and primary key of one table should be referred as foreign key in another table, finding and assigning these could be challenging, I learnt and has applied these in my database

1. **Joins:**

The joins which are used in joining tables which result in combining 2 or more tables in the query for obtaining what we need from database is must, because if we are building a relationship between tables, to query them we must join them. Here in my project, I have joined tables in my query to find the incentive which is the important for my project

**Correctness and Completeness**:

The project meets the expectations of correctness and completeness. Mockaroo's realistic data generation was a critical resource in achieving this. Overall, this experience solidified my knowledge and skills in database design and querying.

create table vehicle (

vehicle\_id CHAR(7) NOT NULL PRIMARY KEY,

license\_plate CHAR(7) NOT NULL,

vehicle\_type VARCHAR(15) NOT NULL,

capacity\_in\_ton NUMERIC(2) NOT NULL,

model\_year CHAR(4) NOT NULL

);

create table customer (

cust\_id CHAR(7) NOT NULL PRIMARY KEY,

cust\_firstname VARCHAR(20) NOT NULL,

cust\_lastname VARCHAR(20) NOT NULL,

email\_id VARCHAR(50) NOT NULL,

cust\_phno VARCHAR(15) NOT NULL

);

create table driver (

driver\_id CHAR(8) NOT NULL PRIMARY KEY,

driver\_firstname VARCHAR(30) NOT NULL,

driver\_lastname VARCHAR(30) NOT NULL,

driver\_phno VARCHAR(15) NOT NULL,

driver\_address VARCHAR(100),

vehicle\_id CHAR(7) NOT NULL,

CONSTRAINT fk\_vehicle\_id FOREIGN KEY(vehicle\_id) REFERENCES vehicle(vehicle\_id)

);

create table trip (

trip\_id CHAR(11) NOT NULL PRIMARY KEY,

trip\_date DATE,

start\_point VARCHAR(50) NOT NULL,

end\_point VARCHAR(50) NOT NULL,

distance\_in\_miles VARCHAR(5) NOT NULL,

vehicle\_id CHAR(7) NOT NULL,

trip\_status VARCHAR(30),

driver\_id CHAR(8) NOT NULL,

cust\_id CHAR(7) NOT NULL,

promotion\_id VARCHAR(6),

CONSTRAINT fk\_vehicle\_id2 FOREIGN KEY(vehicle\_id) REFERENCES vehicle(vehicle\_id),

CONSTRAINT fk\_driver\_id FOREIGN KEY(driver\_id) REFERENCES driver(driver\_id),

CONSTRAINT fk\_cust\_id FOREIGN KEY(cust\_id) REFERENCES customer(cust\_id),

CONSTRAINT fk\_promotion\_id FOREIGN KEY(promotion\_id) REFERENCES promotion(promotion\_id)

);

create table payment (

payment\_id CHAR(6) NOT NULL PRIMARY KEY,

payment\_method VARCHAR(10),

payment\_status VARCHAR(10),

trip\_id CHAR(11) NOT NULL,

fare DECIMAL(5,2),

dicount\_type VARCHAR(10),

discount\_value DECIMAL(5, 2),

final\_fare DECIMAL(5, 2),

CONSTRAINT fk\_trip\_id FOREIGN KEY(trip\_id) REFERENCES trip(trip\_id)

);

create table feedback (

feedback\_id CHAR(10) NOT NULL PRIMARY KEY,

rating VARCHAR(4),

trip\_id CHAR(11) NOT NULL,

cust\_id CHAR(7) NOT NULL,

CONSTRAINT fk\_trip\_id2 FOREIGN KEY(trip\_id) REFERENCES trip(trip\_id),

CONSTRAINT fk\_cust\_id2 FOREIGN KEY(cust\_id) REFERENCES customer(cust\_id)

);

create table promotion (

promotion\_id VARCHAR(6)PRIMARY KEY,

discount\_type VARCHAR(10)

);

INSERT INTO VEHICLE (vehicle\_ID, license\_plate, vehicle\_type, capacity\_in\_ton, model\_year)

VALUES

('VH00001', 'XYZ5678', 'ACE', 1, 2015),

('VH00002', 'DEF2345', 'LEYLAND', 2, 2018),

('VH00003', 'GHI8765', 'BOLERO', 3, 2017),

('VH00004', 'JKL4321', 'INTRA', 4, 2020),

('VH00005', 'MNO6543', '407', 7, 2016),

('VH00006', 'PQR9876', 'CANTER', 10, 2019),

('VH00007', 'STU3456', 'ACE', 1, 2015),

('VH00008', 'VWX8901', 'LEYLAND', 2, 2018),

('VH00009', 'YZA2468', 'BOLERO', 3, 2017),

('VH00010', 'BCD5671', 'INTRA', 4, 2020),

('VH00011', 'EFG8902', '407', 7, 2016),

('VH00012', 'HIJ1357', 'CANTER', 10, 2019),

('VH00013', 'KLM2469', 'ACE', 1, 2015),

('VH00014', 'NOP7531', 'LEYLAND', 2, 2018),

('VH00015', 'QRS8642', 'BOLERO', 3, 2017),

('VH00016', 'TUV3790', 'INTRA', 4, 2020),

('VH00017', 'WXY8024', '407', 7, 2016),

('VH00018', 'ZAB9135', 'CANTER', 10, 2019),

('VH00019', 'CDE4702', 'ACE', 1, 2015),

('VH00020', 'STU5419', 'LEYLAND', 2, 2018);

INSERT INTO driver (driver\_id, driver\_firstname, driver\_lastname, driver\_phno, driver\_address, vehicle\_id)

VALUES

('DR00001', 'James', 'Smith', '452-679-8391', '123 Main St, Springfield, IL 62701', 'VH00001'),

('DR00002', 'Emma', 'Johnson', '379-041-5612', '456 Oak St, Chicago, IL 60601', 'VH00002'),

('DR00003', 'Michael', 'Brown', '918-273-6490', '789 Pine St, Madison, WI 53703', 'VH00003'),

('DR00004', 'Olivia', 'White', '653-298-1740', '101 Maple Ave, Des Moines, IA 50309', 'VH00004'),

('DR00005', 'Liam', 'Green', '204-785-3920', '202 Birch Rd, Omaha, NE 68102', 'VH00005'),

('DR00006', 'Sophia', 'Black', '578-246-9087', '303 Cedar Dr, St. Louis, MO 63103', 'VH00006'),

('DR00007', 'James', 'Smith', '364-728-9073', '404 Elm St, Kansas City, MO 64108', 'VH00007'),

('DR00008', 'David', 'Wilson', '483-921-7521', '505 Redwood Blvd, Indianapolis, IN 46204', 'VH00008'),

('DR00009', 'Grace', 'Martinez', '825-761-0473', '606 Willow Ln, Columbus, OH 43215', 'VH00009'),

('DR00010', 'Ethan', 'Clark', '671-042-5923', '707 Cedar Ln, Louisville, KY 40202', 'VH00010'),

('DR00011', 'Charlotte', 'Hall', '137-849-6274', '808 Maple St, Nashville, TN 37203', 'VH00011'),

('DR00012', 'Joshua', 'Allen', '839-570-2481', '909 Pine Blvd, Denver, CO 80202', 'VH00012'),

('DR00013', 'Liam', 'Green', '568-492-0193', '1010 Oak St, Phoenix, AZ 85004', 'VH00013'),

('DR00014', 'Lucas', 'Young', '976-283-4580', '1111 Birchwood Ave, Dallas, TX 75201', 'VH00014'),

('DR00015', 'Isabella', 'Lewis', '658-304-9142', '1212 Elm Blvd, Houston, TX 77002', 'VH00015'),

('DR00016', 'Amelia', 'Scott', '704-185-2945', '1313 Maple Rd, Miami, FL 33101', 'VH00016'),

('DR00017', 'Benjamin', 'Adams', '362-794-1580', '1414 Pine Dr, Atlanta, GA 30303', 'VH00017'),

('DR00018', 'Michael', 'Brown', '517-963-8472', '1515 Cedar Blvd, Charlotte, NC 28202', 'VH00018'),

('DR00019', 'Sophia', 'Black', '492-763-8120', '1616 Redwood Rd, Seattle, WA 98101', 'VH00019'),

('DR00020', 'Ava', 'Davis', '240-971-6357', '1717 Birch St, Boston, MA 02108', 'VH00020');

INSERT INTO promotion (promotion\_id, discount\_type)

VALUES

('MKSMPK', 'FLAT20%'),

('PRKMNU', 'FLAT5%'),

('RVKMRG', '20OFF'),

('LKHISD', 'FLAT10%'),

('PRMKMR', '15OFF'),

('MNUKMR', 'FLAT25%'),

('NEWUSR', 'FLAT50%'),

('N/A', 'null');

SELECT

d.driver\_id, d.driver\_lastname,

COUNT(t.trip\_id) AS trips\_completed, -- Count of completed trips

CASE

WHEN COUNT(t.trip\_id) >= 10 THEN 500

ELSE 0

END AS weekly\_incentive --calculate incentive only if 20+ trips are completed

FROM

driver d

JOIN

trip t ON d.driver\_id = t.driver\_id

WHERE

t.trip\_status = 'Completed' -- Only count completed trips

AND TO\_CHAR(t.trip\_date, 'YYYY-MM-DD') Between '2024-09-01' and '2024-09-07'

GROUP BY

d.driver\_id, d.driver\_lastname -- Group by driver to calculate total trips

HAVING

COUNT(t.trip\_id) >= 2;

--This is just one of the query I tried to get the incentive based on their trips.

**--Tier based incentive**

SELECT

d.driver\_id, d.driver\_lastname,

COUNT(t.trip\_id) AS trips\_completed, -- Count of completed trips

CASE

WHEN COUNT(t.trip\_id) >= 20 THEN 80 -- Incentive for 20 or more trips

WHEN COUNT(t.trip\_id) >= 15 THEN 45 -- Incentive for 15-19 trips

WHEN COUNT(t.trip\_id) >= 10 THEN 30 -- Incentive for 10-14 trips

WHEN COUNT(t.trip\_id) >= 8 THEN 20 -- Incentive for 8-9 trips

ELSE 0 -- No incentive for fewer than 8 trips

END AS weekly\_incentive -- Calculate weekly incentive based on trips completed

FROM

driver d

JOIN

trip t ON d.driver\_id = t.driver\_id

WHERE

t.trip\_status = 'Completed' -- Only count completed trips

AND TO\_CHAR(t.trip\_date, 'YYYY-MM-DD') BETWEEN '2024-09-01' AND '2024-09-07'

GROUP BY

d.driver\_id, d.driver\_lastname -- Group by driver to calculate total trips

HAVING

COUNT(t.trip\_id) > 1; -- Ensure at least 2 trips are completed

**--Trips per Vehicle**

SELECT

v.vehicle\_id, v.license\_plate,

COUNT(t.trip\_id) AS total\_trips

FROM

vehicle v

JOIN

trip t ON v.vehicle\_id = t.vehicle\_id

WHERE

t.trip\_status = 'Completed'

AND TO\_CHAR(t.trip\_date, 'YYYY-MM') = '2024-09'

GROUP BY

v.vehicle\_id, v.license\_plate

ORDER BY

total\_trips DESC;

**--Failed trips**

SELECT

t.trip\_id, t.trip\_date, d.driver\_id, d.driver\_lastname, v.vehicle\_id, v.license\_plate

FROM

trip t

JOIN

driver d ON t.driver\_id = d.driver\_id

JOIN

vehicle v ON t.vehicle\_id = v.vehicle\_id

WHERE

t.trip\_status = 'Trip Failed'

AND TO\_CHAR(t.trip\_date, 'YYYY-MM') = '2024-09';

**-- Promotion Usage Analysis**

SELECT

p.promotion\_id,

p.discount\_type,

COUNT(t.trip\_id) AS trips\_with\_promotion

FROM

promotion p

LEFT JOIN

trip t ON p.promotion\_id = t.promotion\_id

WHERE

TO\_CHAR(t.trip\_date, 'YYYY-MM') = '2024-09'

GROUP BY

p.promotion\_id, p.discount\_type

ORDER BY

trips\_with\_promotion DESC;

**--Top-Performing Drivers by Completed Trips**

SELECT

d.driver\_id,

d.driver\_lastname,

COUNT(t.trip\_id) AS total\_trips

FROM

driver d

JOIN

trip t ON d.driver\_id = t.driver\_id

WHERE

t.trip\_status = 'Completed'

AND TO\_CHAR(t.trip\_date, 'YYYY-MM') = '2024-09'

GROUP BY

d.driver\_id, d.driver\_lastname

ORDER BY

total\_trips DESC

FETCH FIRST 5 ROWS ONLY;