ZAIO INSTITUTE OF TECHNOLOGY

Assignment 3: SQL Analysis of NCR Ride Bookings

Total Marks: 50

Due: 21 September 2025 @ 23:59

Assignment Overview

This assignment focuses on **data loading, cleaning, exploration, and statistical analysis in SQL** using the NCR_Ride_bookings dataset. Students will import the CSV into a relational database (e.g., MySQL, PostgreSQL, SQLite, DuckDB), create an appropriate schema, and use SQL to analyse ride patterns, cancellations, fares, and operational metrics.

Students will demonstrate proficiency in handling missing data, feature engineering with SQL, aggregations, window functions, subqueries, and outlier detection.

Dataset Columns (from the uploaded CSV):

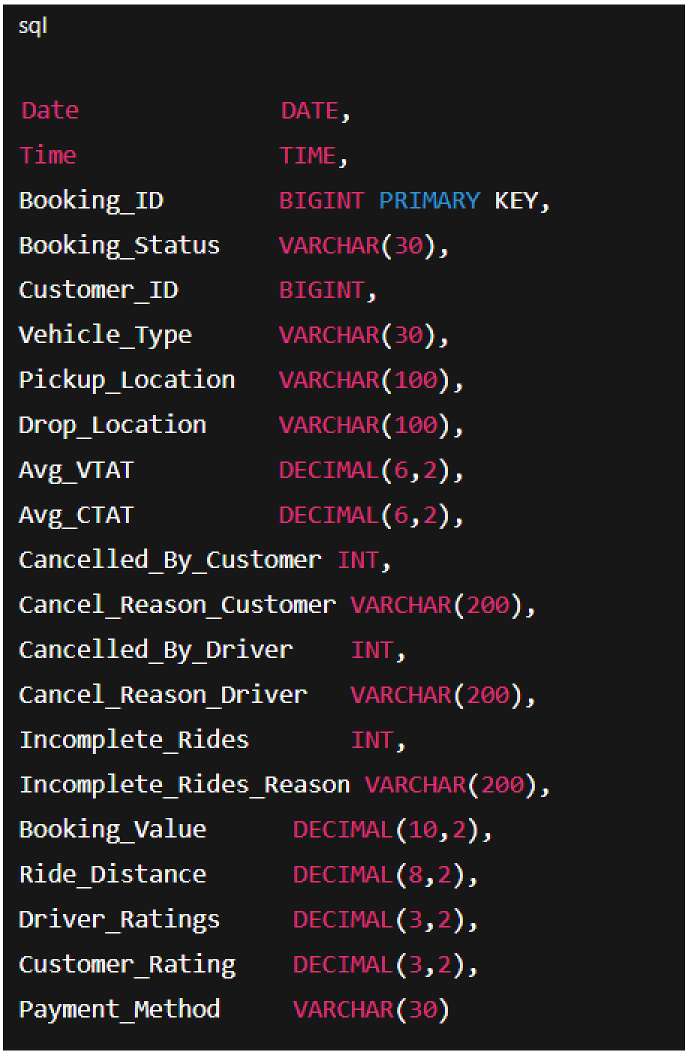
```
Date, Time, Booking ID, Booking Status, Customer ID, Vehicle Type, Pickup
Location, Drop Location, Avg VTAT, Avg CTAT, Cancelled Rides by Customer,
Reason for cancelling by Customer, Cancelled Rides by Driver, Driver
Cancellation Reason, Incomplete Rides, Incomplete Rides Reason, Booking
Value, Ride Distance, Driver Ratings, Customer Rating, Payment Method
```

Learning Objectives

1. Perform SQL-based data preprocessing (schema design, handling duplicates and missing values, SQL feature engineering).
2. Conduct exploratory data analysis (EDA) entirely in SQL using aggregations, grouping, filtering, and window functions.
3. Apply statistical methods (descriptive stats, correlation proxies, IQR outliers) using SQL.
4. Present findings in a structured report with clear SQL queries and interpretations.

Phase 1: Data Collection (5 Marks)

1. Import the CSV into your SQL database as a table named `bookings_raw`.
2. Create a cleaned, typed table `bookings` using an explicit schema (choose appropriate types):



sql

```
Date              DATE,
Time              TIME,
Booking_ID        BIGINT PRIMARY KEY,
Booking_Status    VARCHAR(30),
Customer_ID       BIGINT,
Vehicle_Type      VARCHAR(30),
Pickup_Location   VARCHAR(100),
Drop_Location     VARCHAR(100),
Avg_VTAT          DECIMAL(6,2),
Avg_CTAT          DECIMAL(6,2),
Cancelled_By_Customer INT,
Cancel_Reason_Customer VARCHAR(200),
Cancelled_By_Driver    INT,
Cancel_Reason_Driver   VARCHAR(200),
Incomplete_Rides       INT,
Incomplete_Rides_Reason VARCHAR(200),
Booking_Value      DECIMAL(10,2),
Ride_Distance      DECIMAL(8,2),
Driver_Ratings     DECIMAL(3,2),
Customer_Rating    DECIMAL(3,2),
Payment_Method     VARCHAR(30)
```

3. **Task:**
    - Write SQL to compute **missing value percentage** per column in `bookings_raw`.
    - Create and populate `bookings` with **rows where critical fields are not NULL**
    - Show the row count before vs after cleaning.

Phase 2: Data Preparation (10 Marks)

1. **Duplicate analysis:**
   - Detect duplicate `Booking ID` values and remove duplicates, **keeping the earliest (`Date,Time`)** occurrence. Show counts removed.

2. **Feature engineering (SQL):**
   - Create a `pickup_ts` timestamp from `Date` + `Time`.
   - Derive `Day_Of_Week` (Mon..Sun) and `Hour_Of_Day` (0–23) from `pickup_ts`.
   - Create `Route` as `Pickup_Location || ' -> ' || Drop_Location`.
   - Normalise `Payment_Method` to upper case and trim spaces (store as `Payment_Method_Norm`).

3. **Task:**
   - Document all preprocessing steps with the exact SQL used (CTEs or CREATE TABLE AS / INSERT…SELECT recommended).
   - Provide a **`bookings_clean`** table with the engineered columns.

Phase 3: Data Exploration (SQL) (10 Marks)

Create at least **4 exploration queries** (no Python plots required; results must be produced via SQL).

1. **Distribution analysis:**
   - Query to produce **bucketed fare distribution** (e.g., `<100`, `100-199.99`, `200-299.99`, `>=300`) with counts and percentages.

2. **Categorical analysis:**
   - Top 10 **Vehicle Type** × **Booking Status** counts with row-wise and column-wise percentages.

3. **Relationship analysis:**
   - **Ride Distance vs Booking Value**: compute Pearson-style components using SQL aggregates: `COUNT`, `SUM(x)`, `SUM(y)`, `SUM(x*y)`, `SUM(x^2)`, `SUM(y^2)` to enable correlation calculation (you may compute the final coefficient in SQL if your engine supports `CORR`, otherwise show the components).

4. **Comparative analysis:**

- **Booking Value by Payment Method**: show min, Q1 (25th pct), median (50th), Q3 (75th), max per `Payment_Method_Norm` using window functions or percentile functions supported by your DB (or approximate using NTILE). **Task:** For each query, add a **1–2 line interpretation** of the result (e.g., "High-value rides are more likely paid by CARD/UPI.").

Phase 4: Applied Statistical Analysis (5 Marks)

1. **Descriptive statistics (SQL):**
   - Compute **mean, median, stddev** for `Booking_Value` and `Ride_Distance`.
2. **Correlation analysis:**
   - Compute correlation (or correlation components) between `Booking_Value` and `Ride_Distance`. If your engine supports `CORR(x,y)`, use it; else, calculate the numerator/denominator terms and present the final value.
3. **Outlier detection (IQR):**
   - Using SQL, compute Q1 and Q3 for `Booking_Value`, derive IQR = Q3 − Q1, and flag outliers where `Booking_Value > Q3 + 1.5*IQR`. Return **top 20** outlier rows with `Booking_ID`, `Route`, `Booking_Value`.
4. **Task:**
   - Briefly explain your findings (e.g., "Weak positive correlation between distance and value; long but cheap trips appear as anomalies.").

Phase 5: Advanced Analysis (10 Marks)

Answer the following using **GROUP BY, window functions, and subqueries**:

1. **Operational performance:**
   - For each `Vehicle_Type`, compute **completion rate** = `Completed / (Completed + Cancelled + Incomplete)` using `Booking_Status`. Rank vehicle types by completion rate.
2. **Route profitability:**

- Identify the **top 10 Routes** by **total Booking_Value** and also show **average Booking_Value** and **ride count** per route.

3. **Cancellation forensics:**
   - Among cancelled bookings, show the **top 5 cancel reasons** from **both** `Cancel_Reason_Customer` and `Cancel_Reason_Driver` (treat NULL/blank as “Unspecified”). Include counts and percentages.

4. **Service levels (Time windows):**
   - By `Hour_Of_Day`, compute **average Avg_VTAT** and **average Avg_CTAT** and list the **3 busiest hours** by ride count, alongside those averages.

5. **Customer cohorts and churn (SQL only):**
   - Define cohorts by **first booking month** (YYYY-MM from `pickup_ts`). For each cohort month, compute **cohort size** and **retention into the next month** (customers who ride again in month+1).
   - List **customers who booked in 2023 but 0 bookings in 2024** (churn risk).

Task: Support each answer with the exact SQL you wrote and a one-line insight.

Phase 6: Conclusion and Report (10 Marks)

1. **Summary of findings:**
   - Key trends (e.g., “Auto rides dominate volume but have lower completion rate during evening peak hours,” “Top routes concentrate between A and B with higher average fares”).

2. **Reflection:**
   - Challenges faced (e.g., joining reasons from separate columns, handling NULL/blank reasons, computing percentiles/IQR in your SQL engine).

3. **Report structure:**
   - Sections:
     1. Data Collection
     2. Data Preparation
     3. Data Exploration (SQL)
     4. Statistical Analysis (SQL)
     5. Advanced Analysis (SQL)
     6. Conclusion

Submission Requirements

- **PDF Report** with your SQL listings and interpretations.
- **SQL scripts** (`.sql`) for DDL (table creation) and DML (queries).
- **Database dump** or file (e.g., `.db` for SQLite / `.sql` dump for MySQL/Postgres).
- **Cleaned CSV** exported from `bookings_clean` (optional but recommended).
- **ZIP YOUR WORK IN ONE FILE AND PUT YOUR NAME ON THE FOLDER**.

Rubric

Total Marks: 50

Criteria	Exceptional	Proficient	Needs Improvement	Unsatisfactory	Poor
1. Data Collection (5)	• Correct import and citation. • Clean table created with correct types.	• Minor type/constraint issues.	• Import done; types not ideal.	• Incomplete import; schema weak.	• No dataset / wrong table.
2. Data Preparation (10)	• 100% duplicates handled correctly by business rule. • Solid SQL feature engineering (timestamps, route, normalised payment). • Constraints/NULLs justified.	• Mostly correct; minor gaps.	• Minimal engineering; partial duplicate handling.	• Major formatting/logic errors.	• No cleaning performed.
3. Data Exploration (SQL) (10)	• 4+ strong queries, clear buckets/percentiles, insightful interpretations.	• 4 queries with minor issues.	• 3 basic queries; weak insight.	• 1–2 shallow queries; no insight.	• No exploration.
4. Statistical Analysis (5)	• Full stats (mean/median/std), correlation (or components), IQR outliers correctly flagged with justification.	• Small omissions or rounding errors.	• Partial stats; no outliers or correlation.	• Only basic averages.	• None.
5. Advanced Analysis (10)	• All tasks completed using window functions/CTEs/subqueries; insights tied to operations.	• 3–4 tasks done; minor insight.	• 2 tasks; superficial.	• 1 task; incorrect.	• None.
6. Report and Presentation (10)	Well-structured, academic tone, reproducible SQL, sensible conclusions.	Good report; minor omissions.	Basic report missing elements.	Poorly structured; major gaps.	No report submitted.

Detailed steps for submitting an assignment:

1. Go to Google Classroom: Access Google Classroom through your web browser and sign in with your Google account.
2. Navigate to the class and assignment: Click on the relevant class and then the "Classwork" tab. Locate the assignment you want to submit.
3. View the assignment: Click on the assignment to view the instructions and details.
4. Add or create an attachment: Under "Your work", click "Add or create".
5. Choose the attachment type: Select "File" to upload from your device or choose from Google Drive, a link, or create a new document (Docs, Slides, Sheets, Drawings, or PDF).
6. Attach the file: If selecting "File", browse your computer, select the file, and click "Open".
7. Upload the file: Click "Upload" to attach the file to your assignment.
8. Turn in the assignment: Once the file is attached, click "Turn In".
9. Confirm submission: Click "Turn In" again to finalise the submission.