Cyclistic SQL Workflow Walkthrough

Executive Summary

This document provides a clear and structured walkthrough of the SQL work I carried out as part of my Cyclistic portfolio project. The goal was to prepare, clean, and analyze a large dataset of bike-share rides using MySQL. While some data preparation was handled in Excel, this workflow focuses on the database-side processes that supported my analysis and dashboard creation in Tableau.

What follows is a breakdown of how I imported the data, organized it by rider type, removed duplicates, engineered time-based fields, and combined the data into a single clean table ready for analysis. Each step includes actual SQL snippets and short explanations to walk the reader through what was done and why. This was an important project for me in demonstrating my ability to manage real-world datasets and apply best-practice logic to support business insights.

1. Data Loading

The first step involved loading CSV files into the `ride_data` table using the `LOAD DATA INFILE` statement. Files were imported month-by-month with proper parsing of date and time formats.

Key actions:

- Truncate existing data

- Format `ride_length_mins` as `DECIMAL(7,2)`

- Allow null values for latitude/longitude columns

- Parse and convert date/time fields

Example SQL:

```
TRUNCATE TABLE ride_data;

ALTER TABLE ride_data MODIFY ride_length_mins DECIMAL(7,2);

LOAD DATA INFILE '...'
INTO TABLE ride_data
FIELDS TERMINATED BY ','
ENCLOSED BY '"'
LINES TERMINATED BY '\n'
IGNORE 1 ROWS
(ride_id, rideable_type, @started_at, @ended_at, ..., member_casual, ride_length_mins, ...)
```

```
SET
  started_at = STR_TO_DATE(@started_at, '%d-%m-%Y %H:%i:%s'),
  ended_at = STR_TO_DATE(@ended_at, '%d-%m-%Y %H:%i:%s');
```

2. Data Segmentation

The dataset was split into two subsets based on the `member_casual` column.

Example SQL:

```
CREATE TABLE casual_rides AS
SELECT * FROM ride_data
WHERE member_casual = 'casual';

CREATE TABLE member_rides AS
SELECT * FROM ride_data
WHERE member_casual = 'member';
```

3. Duplicate Detection & Removal

To identify duplicate rides (based on `ride_id`), a surrogate key (`id`) was added. Then, self-joins helped isolate and remove duplicate rows.

Example SQL:

```
ALTER TABLE member_rides ADD id INT AUTO_INCREMENT PRIMARY KEY;

DELETE r1
FROM member_rides r1
JOIN member_rides r2
  ON r1.ride_id = r2.ride_id AND r1.id > r2.id;
```

4. Cleaned Subsets

Both datasets were deduplicated using a `GROUP BY ride_id` and keeping only the row with the smallest `id`.

Example SQL:

```
CREATE TABLE member_rides_clean AS
SELECT * FROM member_rides
WHERE id IN (
  SELECT MIN(id)
  FROM member_rides
  GROUP BY ride_id
);
```

5. Feature Engineering

New time-based columns were added to support seasonal and monthly trend analysis.

Example SQL:

```
ALTER TABLE member_rides_clean ADD COLUMN month_name VARCHAR(15);
UPDATE member_rides_clean SET month_name = MONTHNAME(date);

ALTER TABLE member_rides_clean ADD COLUMN month_num TINYINT;
UPDATE member_rides_clean SET month_num = MONTH(date);
```

6. Unified Clean Table

A single combined table, `all_rides_clean`, was created to unify member and casual data.

Example SQL:

```
CREATE TABLE all_rides_clean AS
SELECT
  member_casual AS user_type,
  rideable_type,
  ride_length_mins,
  date,
  month_name,
  month_num,
  day_of_week,
  hour_block
FROM member_rides_clean

UNION ALL

SELECT
  member_casual AS user_type,
  rideable_type,
  ride_length_mins,
  date,
  month_name,
  month_num,
  day_of_week,
  hour_block
FROM casual_rides_clean;
```

7. Validation & Summary Queries

To support dashboard development in Tableau, a few high-level checks and summaries were done:

Example SQL:

```
SELECT user_type, COUNT(*) AS total_rides
FROM all_rides_clean
GROUP BY user_type;

SELECT COUNT(ride_length_mins) AS under_two_mins
FROM member_rides_clean
WHERE ride_length_mins < 2;
```

Note: Rides shorter than two minutes were identified as likely data anomalies — such as accidental scans, false starts, or quick undock-and-dock events. These records were excluded from the Tableau visualizations to maintain data integrity.

6.1 Table Schema for `all_rides_clean`

Below is a quick overview of the structure and purpose of the columns included in the final unified dataset.

Column	Description
user_type	Indicates whether the rider was a 'member' or 'casual' user
rideable_type	Type of bike used for the ride
ride_length_mins	Total ride duration in minutes
date	Date of the ride
month_name	Month name derived from the `date` column
month_num	Numeric representation of the month (1–12)
day_of_week	Day of the week the ride took place (0=Sunday)
hour_block	Hour of day during which the ride began

8. Additional Summary Queries for Business Insights

These queries helped provide context for business questions and supported dashboard creation in Tableau.

Average ride duration by user type:

```
SELECT user_type, ROUND(AVG(ride_length_mins), 2) AS avg_ride_duration
FROM all_rides_clean
```

```
GROUP BY user_type;
```

Top 5 peak usage hours for casual riders:

```
SELECT hour_block, COUNT(*) AS ride_count
FROM all_rides_clean
WHERE user_type = 'casual'
GROUP BY hour_block
ORDER BY ride_count DESC
LIMIT 5;
```