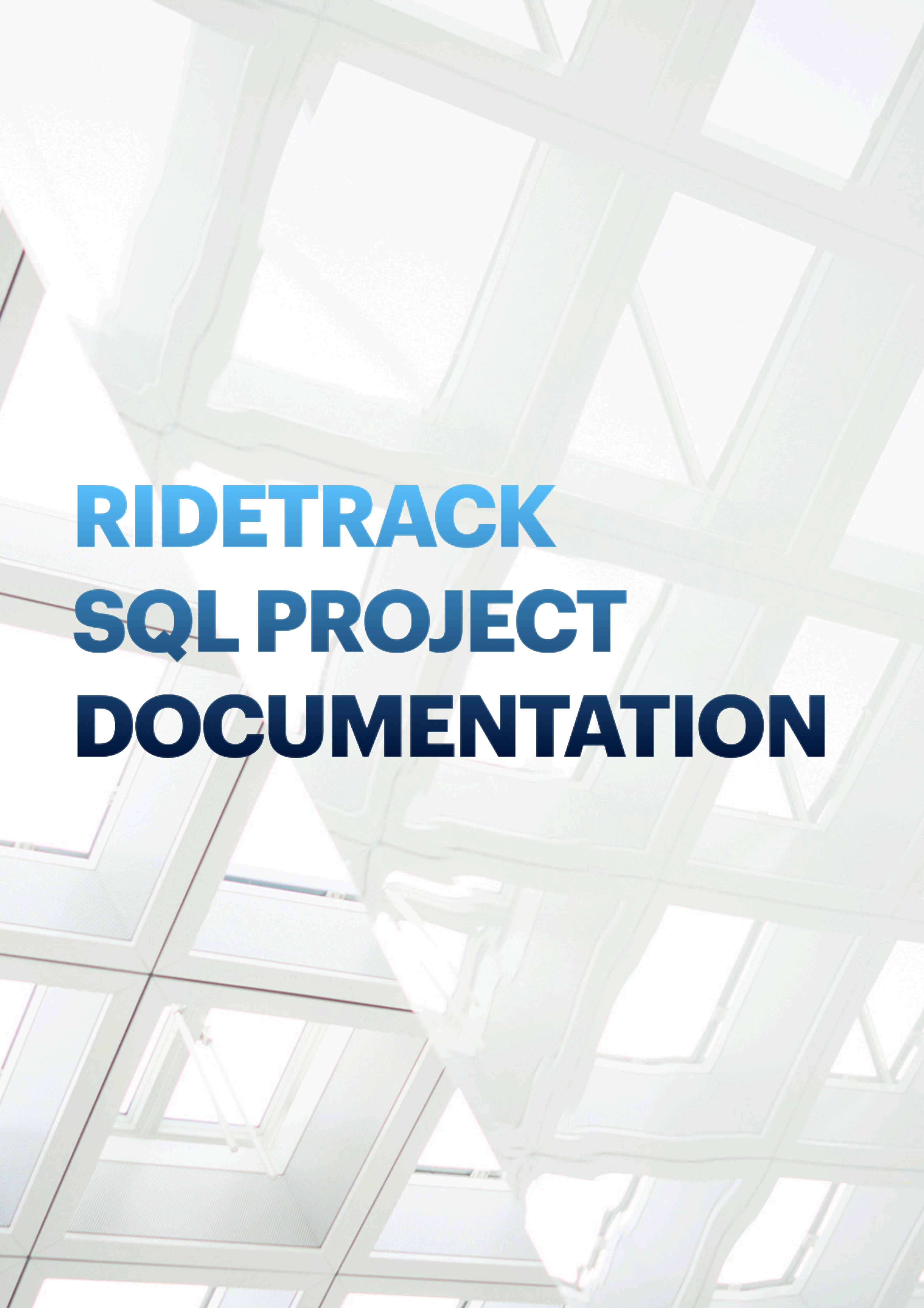


RIDETRACK SQL PROJECT DOCUMENTATION

PROJECT BY
ANIRUDHA PRADIP JOHARE

INTRODUCTION

RideTrack is a SQL-based ride-booking system designed to manage users (riders and drivers), vehicles, rides, payments, and ratings. It simulates a real-world ride-hailing application similar to Uber or Lyft, ensuring efficient ride management and business insights.

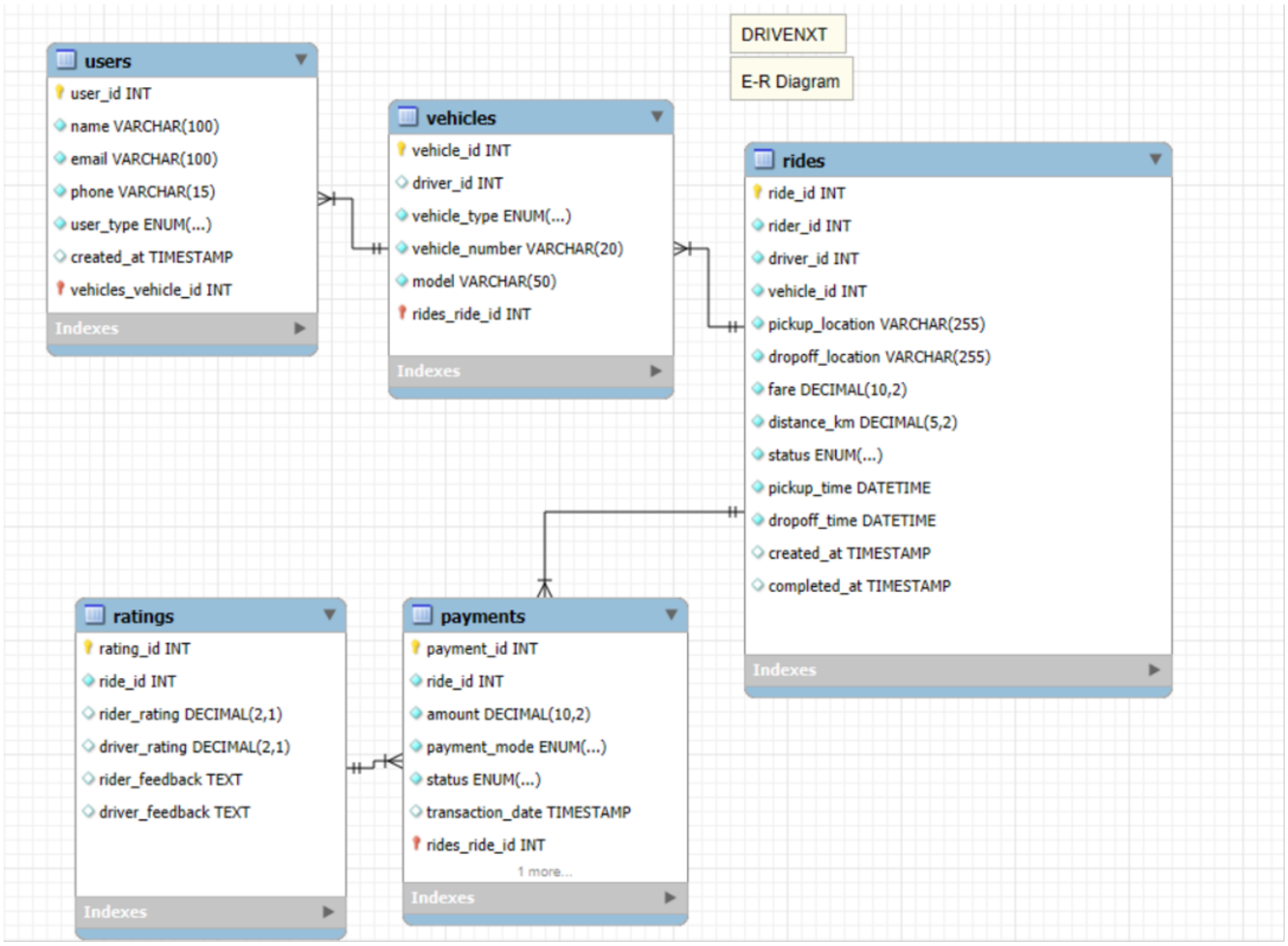
OBJECTIVE OF THIS PROJECT :

The primary objectives of this project are:

- **Database Management:** Efficiently store and retrieve ride-booking data.
- **Data Integrity & Optimization**: Ensure data consistency using foreign keys, constraints, and normalization.
- **Operational Analytics:** Analyze ride trends, payment methods, customer preferences, and driver performance.
- **Business Insights:** Provide key insights into ride demand, revenue, and customer feedback.
- **Decision-Making Support:** Assist stakeholders in making data-driven decisions for improving service quality and business efficiency.

SCHEMA DESIGN :

The RideTrack database consists of five core tables:



DATA INSIGHTS :

The RideTrack system provides key insights into business and operational performance, including:

- **User Distribution:** Understanding the ratio of riders to drivers helps in optimizing driver allocation and managing ride availability.
- **Most Popular Pickup Locations:** Identifying high-demand areas allows for better fleet management and service improvements.
- **Revenue Trends:** Analyzing earnings over time provides insight into business growth and profitability.
- **Driver Performance:** Tracking top-rated drivers ensures high service quality and helps in incentive planning.
- **Pending Payments:** Monitoring unpaid transactions helps in improving financial operations and reducing payment failures.

QUERIES FOR ANALYSIS :

1. Count the Total Number of Riders and Drivers

```
Select user_type, COUNT(*) As total_users

FROM Users

GROUP BY user_type;
```

2. List of All Drivers with Their Vehicles

```
SELECT u.user_id, u.name, u.phone, v.vehicle_type, v.vehicle_number, v.model

FROM Users u

LEFT JOIN Vehicles v ON u.user_id = v.driver_id

WHERE u.user_type = 'driver';
```

3. Find Riders Who Have Never Taken a Ride

```
SELECT u.user_id, u.name, u.email, u.phone

FROM Users u

LEFT JOIN Rides r ON u.user_id = r.rider_id

WHERE u.user_type = 'rider' AND r.ride_id IS NULL;
```

4.. Total Rides Completed vs. Cancelled

```
SELECT status, COUNT(*) AS total_rides

FROM Rides

GROUP BY status;
```

5. Average Fare and Distance of Completed Rides

```
SELECT ROUND(AVG(fare), 2) AS avg_fare, ROUND(AVG(distance_km), 2) AS
        avg_distance

FROM Rides

WHERE status = 'completed';
```

6. Driver Who Completed the Most Rides

```
SELECT driver_id, COUNT(*) AS total_rides

FROM Rides

WHERE status = 'completed'

GROUP BY driver_id

ORDER BY total_rides DESC

LIMIT 1;
```

7. Find the Most Popular Pickup Location

```
SELECT pickup_location, COUNT(*) AS total_rides

FROM Rides

GROUP BY pickup_location

ORDER BY total_rides DESC

LIMIT 1;
```

8. Total Revenue Generated

```
SELECT SUM(amount) AS total_revenue

FROM Payments
```

```
WHERE status = ‘completed';
```

9. Identify Pending Payments

```
SELECT p.payment_id, p.ride_id, u.name AS rider_name, p.amount, p.payment_mode

FROM Payments p

JOIN Rides r ON p.ride_id = r.ride_id

JOIN Users u ON r.rider_id = u.user_id

WHERE p.status = 'pending';
```

10. Find the Highest-Rated Driver

```
SELECT r.driver_id, u.name, ROUND(AVG(rt.driver_rating), 2) AS avg_rating

FROM Ratings rt

JOIN Rides r ON rt.ride_id = r.ride_id

JOIN Users u ON r.driver_id = u.user_id

GROUP BY r.driver_id, u.name

ORDER BY avg_rating DESC

LIMIT 1;
```

11. All Negative Feedback

```
SELECT r.ride_id, u.name AS rider_name, rt.rider_rating, rt.driver_rating,
        rt.rider_feedback, rt.driver_feedback

FROM Ratings rt

JOIN Rides r ON rt.ride_id = r.ride_id

JOIN Users u ON r.rider_id = u.user_id

WHERE rt.rider_rating < 3.5 OR rt.driver_rating < 3.5;
```

BUSINESS STRATEGIES :

Based on the data insights, we can formulate strategic business decisions:

1. Driver Incentives & Performance Improvement

- Reward drivers with high ride completion rates and top ratings.
- Provide training and assistance to drivers with low ratings.

2. Optimize Ride Allocation

- Identify high-demand locations and deploy more drivers in those areas.
- Implement dynamic pricing to encourage drivers to operate in peak areas.

3. Improve Customer Experience

- Identify common complaints in rider feedback and take corrective actions.
- Improve vehicle tracking and estimated arrival times.

4. Financial Growth & Fraud Detection

- Reduce pending payments by implementing automated reminders for users.
- Introduce cashback or loyalty programs to increase prepaid transactions.

5. Expansion & Scalability

- Expand to new cities based on ride demand analysis.
- Introduce subscription models for frequent riders.

CONCLUSION :

The RideTrack project effectively demonstrates database design, SQL querying, data analysis, and business strategy formulation. It showcases a real-world ride-booking system, making it an excellent addition to a data analytics or database management portfolio.