-- Phase 1: MySQL Schema Design

# Create Database ---> DriveNXT

CREATE DATABASE If Not Exists DriveNXT;

# Use DriveNXT Database

USE DriveNXT;

/\*

----------------------------------------------------------------------------------------------------------------

Flow Structure of the Project:

1. Users Table ---> Stores rider and driver details with unique identifiers.

2. Vehicles Table ---> Links drivers to their assigned vehicles for accurate tracking.

3. Rides Table: ---> Core transactional entity capturing trip details, fare, distance, status, and timestamps

(pickup\_time, dropoff\_time) for duration analysis.

4. Payments Table: ---> Manages ride-linked financial transactions, tracking amount, mode, and status.

5. Ratings Table: ----> Enables service quality assessment through rider and driver feedback.

6. Database Design: ---> Ensures referential integrity, optimized query performance,

and supports business analytics for operational efficieny

----------------------------------------------------------------------------------------------------------------

\*/

-- [RDBMS] Flow Structure : Create Tables in DriveNXT Database

-- Create Users Table

CREATE TABLE Users (

user\_id INT PRIMARY KEY AUTO\_INCREMENT,

name VARCHAR(100) NOT NULL,

email VARCHAR(100) UNIQUE NOT NULL,

phone VARCHAR(15) UNIQUE NOT NULL,

user\_type ENUM('rider', 'driver') NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Create Vehicles Table

CREATE TABLE Vehicles (

vehicle\_id INT PRIMARY KEY AUTO\_INCREMENT,

driver\_id INT, -- No foreign key here

vehicle\_type ENUM('sedan', 'suv', 'hatchback', 'bike') NOT NULL,

vehicle\_number VARCHAR(20) UNIQUE NOT NULL,

model VARCHAR(50) NOT NULL

);

-- Create Rides Table

CREATE TABLE Rides (

ride\_id INT PRIMARY KEY AUTO\_INCREMENT,

rider\_id INT NOT NULL,

driver\_id INT NOT NULL,

vehicle\_id INT NOT NULL,

pickup\_location VARCHAR(255) NOT NULL,

dropoff\_location VARCHAR(255) NOT NULL,

fare DECIMAL(10,2) NOT NULL,

distance\_km DECIMAL(5,2) NOT NULL,

status ENUM('requested', 'ongoing', 'completed', 'cancelled') NOT NULL,

pickup\_time DATETIME NOT NULL,

dropoff\_time DATETIME NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

completed\_at TIMESTAMP NULL

);

-- Create Payments Table

CREATE TABLE Payments (

payment\_id INT PRIMARY KEY AUTO\_INCREMENT,

ride\_id INT UNIQUE NOT NULL,

amount DECIMAL(10,2) NOT NULL,

payment\_mode ENUM('cash', 'credit\_card', 'debit\_card', 'wallet') NOT NULL,

status ENUM('pending', 'completed', 'failed') NOT NULL,

transaction\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Create Ratings Table

CREATE TABLE Ratings (

rating\_id INT PRIMARY KEY AUTO\_INCREMENT,

ride\_id INT UNIQUE NOT NULL,

rider\_rating DECIMAL(2,1) CHECK (rider\_rating BETWEEN 1 AND 5),

driver\_rating DECIMAL(2,1) CHECK (driver\_rating BETWEEN 1 AND 5),

rider\_feedback TEXT,

driver\_feedback TEXT

);

INSERT INTO Users (name, email, phone, user\_type) VALUES

('Aarav Sharma', 'aarav.sharma@example.com', '9876543210', 'rider'),

('Isha Patel', 'isha.patel@example.com', '9876543211', 'rider'),

('Ravi Kumar', 'ravi.kumar@example.com', '9876543212', 'driver'),

('Priya Mehta', 'priya.mehta@example.com', '9876543213', 'rider'),

('Rahul Verma', 'rahul.verma@example.com', '9876543214', 'driver'),

('Ananya Gupta', 'ananya.gupta@example.com', '9876543215', 'rider'),

('Aditya Joshi', 'aditya.joshi@example.com', '9876543216', 'driver'),

('Neha Singh', 'neha.singh@example.com', '9876543217', 'rider'),

('Vishal Reddy', 'vishal.reddy@example.com', '9876543218', 'driver'),

('Sanya Desai', 'sanya.desai@example.com', '9876543219', 'rider');

INSERT INTO Vehicles (driver\_id, vehicle\_type, vehicle\_number, model) VALUES

(3, 'sedan', 'MH-1234-A', 'Toyota Corolla'),

(5, 'suv', 'DL-5678-B', 'Honda Civic'),

(7, 'hatchback', 'KA-2345-C', 'Maruti Swift'),

(9, 'bike', 'TN-3456-D', 'Honda Activa'),

(11, 'sedan', 'UP-4567-E', 'BMW X5'),

(13, 'suv', 'MH-8765-F', 'Tata Nexon'),

(15, 'hatchback', 'DL-9876-G', 'Suzuki Swift'),

(17, 'bike', 'KA-6543-H', 'Royal Enfield Classic'),

(18, 'sedan', 'TN-7654-I', 'Honda Civic'),

(20, 'suv', 'UP-4321-J', 'Mahindra Thar');

INSERT INTO Rides (rider\_id, driver\_id, vehicle\_id, pickup\_location, dropoff\_location, fare, distance\_km, status, pickup\_time, dropoff\_time) VALUES

(1, 3, 5, 'Andheri, Mumbai', 'Bandra, Mumbai', 150.00, 10.50, 'completed', '2025-02-01 08:30:00', '2025-02-01 08:50:00'),

(2, 5, 9, 'Vile Parle, Mumbai', 'Santacruz, Mumbai', 200.00, 15.00, 'completed', '2025-02-01 09:00:00', '2025-02-01 09:25:00'),

(3, 7, 10, 'Kandivali, Mumbai', 'Malad, Mumbai', 120.00, 8.00, 'completed', '2025-02-01 10:00:00', '2025-02-01 10:20:00'),

(4, 9, 3, 'Borivali, Mumbai', 'Dadar, Mumbai', 180.00, 12.50, 'completed', '2025-02-01 11:00:00', '2025-02-01 11:30:00'),

(5, 11, 8, 'Kalyan, Mumbai', 'Vasai, Mumbai', 220.00, 18.00, 'completed', '2025-02-01 12:00:00', '2025-02-01 12:45:00'),

(6, 13, 7, 'Kurla, Mumbai', 'Chembur, Mumbai', 160.00, 9.50, 'completed', '2025-02-01 13:00:00', '2025-02-01 13:20:00'),

(7, 15, 2, 'Mulund, Mumbai', 'Goregaon, Mumbai', 140.00, 10.00, 'completed', '2025-02-01 14:00:00', '2025-02-01 14:20:00'),

(8, 17, 6, 'Bhayandar, Mumbai', 'Mira Road, Mumbai', 110.00, 7.50, 'completed', '2025-02-01 15:00:00', '2025-02-01 15:15:00'),

(9, 18, 1, 'Lower Parel, Mumbai', 'Worli, Mumbai', 130.00, 6.50, 'completed', '2025-02-01 16:00:00', '2025-02-01 16:20:00'),

(10, 20, 4, 'Malad, Mumbai', 'Goregaon, Mumbai', 140.00, 8.00, 'completed', '2025-02-01 17:00:00', '2025-02-01 17:20:00');

INSERT INTO Payments (ride\_id, amount, payment\_mode, status) VALUES

(1, 150.00, 'cash', 'completed'),

(2, 200.00, 'credit\_card', 'completed'),

(3, 120.00, 'debit\_card', 'completed'),

(4, 180.00, 'wallet', 'completed'),

(5, 220.00, 'cash', 'completed'),

(6, 160.00, 'credit\_card', 'completed'),

(7, 140.00, 'debit\_card', 'completed'),

(8, 110.00, 'wallet', 'completed'),

(9, 130.00, 'cash', 'completed'),

(10, 140.00, 'credit\_card', 'completed');

INSERT INTO Ratings (ride\_id, rider\_rating, driver\_rating, rider\_feedback, driver\_feedback) VALUES

(1, 4.5, 4.7, 'Great ride!', 'Smooth driving, would recommend.'),

(2, 4.0, 4.3, 'Comfortable but a bit slow.', 'Good route choice, needs to improve speed.'),

(3, 3.8, 4.0, 'It was okay, but not fast enough.', 'The ride was good, but the AC was too cold.'),

(4, 4.5, 4.5, 'Loved it!', 'The ride was very comfortable.'),

(5, 4.2, 4.0, 'Good ride overall.', 'Driver was courteous but needs to work on navigation skills.'),

(6, 4.0, 3.8, 'The ride was decent.', 'The car could be cleaned better.'),

(7, 4.5, 4.8, 'Very enjoyable!', 'Friendly driver, good route.'),

(8, 3.7, 4.1, 'Okay, but the ride was bumpy.', 'Need to avoid potholes more carefully.'),

(9, 4.2, 4.3, 'Nice ride.', 'Driver was polite and quick.'),

(10, 4.8, 4.9, 'Amazing, will ride again!', 'Great experience all around.');

# QUESTION : Count the Total Number of Riders and Drivers

SELECT user\_type, COUNT(\*) AS total\_users

FROM Users

GROUP BY user\_type;

# QUESTION : 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';

# QUESTION : 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;

# QUESTION : Total Rides Completed vs. Cancelled

SELECT status, COUNT(\*) AS total\_rides

FROM Rides

GROUP BY status;

# QUESTION :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';

# QUESTION : 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;

# QUESTION : 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;

# QUESTION : Total Revenue Generated

SELECT SUM(amount) AS total\_revenue

FROM Payments

WHERE status = 'completed';

# QUESTION : 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';

# 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;

# 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;