-- Create and select database

CREATE DATABASE IF NOT EXISTS platform\_data;

USE platform\_data;

-- 1. Gaming Platform: user\_play\_sessions

CREATE TABLE IF NOT EXISTS user\_play\_sessions (

user\_id INT,

game\_id INT,

hours\_played DECIMAL(5,2),

play\_date DATE

);

INSERT INTO user\_play\_sessions VALUES

(1, 101, 2.5, '2025-07-01'),

(1, 102, 1.0, '2025-07-15'),

(2, 101, 3.5, '2025-07-20'),

(3, 103, 4.0, '2025-08-01'),

(1, 101, 1.5, '2025-08-05');

-- 2. Cab Booking System: rides

CREATE TABLE IF NOT EXISTS rides (

ride\_id INT AUTO\_INCREMENT PRIMARY KEY,

city VARCHAR(50),

fare DECIMAL(8,2),

trip\_distance DECIMAL(5,2),

ride\_date DATE,

ride\_status VARCHAR(20),

driver\_id INT

);

INSERT INTO rides (city, fare, trip\_distance, ride\_date, ride\_status, driver\_id) VALUES

('New York', 25.50, 10.5, '2025-07-20', 'completed', 101),

('Los Angeles', 30.00, 12.0, '2025-07-22', 'completed', 102),

('New York', 15.00, 5.0, '2025-08-01', 'cancelled', 101),

('Chicago', 40.00, 20.0, '2025-08-02', 'completed', 103),

('Los Angeles', 22.00, 9.0, '2025-08-05', 'completed', 102);

-- 3. Hotel Booking System: guest\_stays

CREATE TABLE IF NOT EXISTS guest\_stays (

guest\_id INT,

room\_id INT,

room\_type VARCHAR(50),

booking\_date DATE,

check\_in\_date DATE,

check\_out\_date DATE

);

INSERT INTO guest\_stays VALUES

(1, 201, 'Deluxe', '2025-07-01', '2025-07-10', '2025-07-15'),

(2, 202, 'Standard', '2025-07-20', '2025-07-21', '2025-07-23'),

(1, 203, 'Suite', '2025-08-01', '2025-08-05', '2025-08-10'),

(3, 201, 'Deluxe', '2025-08-03', '2025-08-04', '2025-08-06');

-- 4. Logistics & Delivery Platform: package\_tracking

CREATE TABLE IF NOT EXISTS package\_tracking (

package\_id INT,

region VARCHAR(50),

shipment\_date DATE,

delivery\_date DATE,

dispatch\_date DATE,

status VARCHAR(20),

shipment\_type VARCHAR(20),

warehouse\_id INT

);

INSERT INTO package\_tracking VALUES

(1001, 'East', '2025-06-15', '2025-06-20', '2025-06-14', 'delivered', 'domestic', 1),

(1002, 'West', '2025-07-01', '2025-07-10', '2025-06-30', 'delivered', 'international', 2),

(1003, 'East', '2025-07-05', '2025-07-15', '2025-07-04', 'delivered', 'international', 1),

(1004, 'North', '2025-08-01', '2025-08-05', '2025-07-31', 'pending', 'domestic', 3),

(1005, 'East', '2025-08-01', '2025-08-07', '2025-08-01', 'delivered', 'domestic', 1);

-- 5. Streaming Service: viewership

CREATE TABLE IF NOT EXISTS viewership (

user\_id INT,

show\_id INT,

genre VARCHAR(50),

watch\_time INT,

view\_date DATE,

subscriber\_type VARCHAR(20)

);

INSERT INTO viewership VALUES

(1, 501, 'Drama', 120, '2025-07-01', 'premium'),

(2, 502, 'Comedy', 90, '2025-07-15', 'free'),

(1, 503, 'Drama', 60, '2025-07-20', 'premium'),

(3, 504, 'Action', 150, '2025-08-01', 'premium'),

(2, 505, 'Comedy', 80, '2025-08-05', 'premium');

USE platform\_data;

-- 1. Total number of hours played by each user in the past month

SELECT user\_id,

SUM(hours\_played) AS total\_hours\_played

FROM user\_play\_sessions

WHERE play\_date >= CURDATE() - INTERVAL 1 MONTH

GROUP BY user\_id;

-- 2. Average number of games played per user last week

SELECT AVG(games\_played) AS avg\_games\_per\_user

FROM (

SELECT user\_id, COUNT(DISTINCT game\_id) AS games\_played

FROM user\_play\_sessions

WHERE play\_date >= CURDATE() - INTERVAL 1 WEEK

GROUP BY user\_id

) AS sub;

-- 3. Most played game and total hours logged in the current quarter

SELECT game\_id,

SUM(hours\_played) AS total\_hours

FROM user\_play\_sessions

WHERE play\_date >= MAKEDATE(YEAR(CURDATE()), 1)

+ INTERVAL QUARTER(CURDATE()) QUARTER - INTERVAL 1 QUARTER

AND play\_date < MAKEDATE(YEAR(CURDATE()), 1)

+ INTERVAL QUARTER(CURDATE()) QUARTER

GROUP BY game\_id

ORDER BY total\_hours DESC

LIMIT 1;

-- 4. Total fare collected from each city in the last 30 days

SELECT city,

SUM(fare) AS total\_fare\_collected

FROM rides

WHERE ride\_date >= CURDATE() - INTERVAL 30 DAY

GROUP BY city;

-- 5. Average trip distance for all completed rides this week

SELECT AVG(trip\_distance) AS avg\_trip\_distance

FROM rides

WHERE ride\_status = 'completed'

AND ride\_date >= CURDATE() - INTERVAL WEEKDAY(CURDATE()) DAY;

-- 6. Driver with the highest total ride earnings in the last year

SELECT driver\_id,

SUM(fare) AS total\_earnings

FROM rides

WHERE ride\_date >= CURDATE() - INTERVAL 1 YEAR

GROUP BY driver\_id

ORDER BY total\_earnings DESC

LIMIT 1;

-- 7. Count the number of nights stayed by guests in the last season (last 3 months)

SELECT guest\_id,

SUM(DATEDIFF(check\_out\_date, check\_in\_date)) AS total\_nights\_stayed

FROM guest\_stays

WHERE check\_in\_date >= CURDATE() - INTERVAL 3 MONTH

GROUP BY guest\_id;

-- 8. Average number of rooms booked per guest this month

SELECT AVG(room\_count) AS avg\_rooms\_per\_guest

FROM (

SELECT guest\_id, COUNT(room\_id) AS room\_count

FROM guest\_stays

WHERE booking\_date >= CURDATE() - INTERVAL 1 MONTH

GROUP BY guest\_id

) AS sub;

-- 9. Room type with the highest occupancy in the current year

SELECT room\_type,

SUM(DATEDIFF(check\_out\_date, check\_in\_date)) AS total\_occupied\_nights

FROM guest\_stays

WHERE check\_in\_date >= DATE\_FORMAT(CURDATE(), '%Y-01-01')

GROUP BY room\_type

ORDER BY total\_occupied\_nights DESC

LIMIT 1;

-- 10. Number of packages delivered per region in the past quarter

SELECT region,

COUNT(\*) AS packages\_delivered

FROM package\_tracking

WHERE delivery\_date >= CURDATE() - INTERVAL 3 MONTH

AND status = 'delivered'

GROUP BY region;

-- 11. Average delivery time for international shipments (in hours)

SELECT AVG(TIMESTAMPDIFF(HOUR, shipment\_date, delivery\_date)) AS avg\_delivery\_time\_hours

FROM package\_tracking

WHERE shipment\_type = 'international'

AND status = 'delivered';

-- 12. Warehouse with the highest number of dispatches this year

SELECT warehouse\_id,

COUNT(\*) AS dispatch\_count

FROM package\_tracking

WHERE dispatch\_date >= DATE\_FORMAT(CURDATE(), '%Y-01-01')

GROUP BY warehouse\_id

ORDER BY dispatch\_count DESC

LIMIT 1;

-- 13. Total views per genre in the last 90 days

SELECT genre,

COUNT(\*) AS total\_views

FROM viewership

WHERE view\_date >= CURDATE() - INTERVAL 90 DAY

GROUP BY genre;

-- 14. Average watch time per user for premium subscribers

SELECT user\_id, AVG(watch\_time) AS avg\_watch\_time

FROM viewership

WHERE subscriber\_type = 'premium'

GROUP BY user\_id;

-- 15. Show or movie with the highest total view time this month

SELECT show\_id,

SUM(watch\_time) AS total\_view\_time

FROM viewership

WHERE view\_date >= CURDATE() - INTERVAL 1 MONTH

GROUP BY show\_id

ORDER BY total\_view\_time DESC

LIMIT 1;