Lecture 16. SQL

CREATE

1. Write SQL queries for table creation for a data model that you created for prev homework (Airbnb model)

CREATE TABLE IF NOT EXISTS Hosts (

host\_id SERIAL PRIMARY KEY,

**host\_name** VARCHAR(255)

);

CREATE TABLE IF NOT EXISTS Guests (

guest\_id SERIAL NOT NULL PRIMARY KEY,

**guest\_name** VARCHAR(255)

);

CREATE TABLE IF NOT EXISTS Rooms (

room\_id SERIAL PRIMARY KEY,

host\_id INT,

**price** DECIMAL(10, 2),

**capacity** INT,

**ac** BOOLEAN,

**refrigerator** BOOLEAN,

**wifi** BOOLEAN,

**private\_bathroom** BOOLEAN,

**pet\_friendly** BOOLEAN,

FOREIGN KEY (host\_id) REFERENCES Hosts(host\_id)

);

CREATE TABLE IF NOT EXISTS Reservations (

reservation\_id SERIAL PRIMARY KEY,

guest\_id INT,

room\_id INT,

**check\_in\_date** DATE,

**check\_out\_date** DATE,

**total\_price** DECIMAL(10, 2),

FOREIGN KEY (guest\_id) REFERENCES Guests(guest\_id),

FOREIGN KEY (room\_id) REFERENCES Rooms(room\_id)

);

CREATE TABLE IF NOT EXISTS Payments (

payment\_id SERIAL PRIMARY KEY,

guest\_id INT,

reservation\_id INT,

amount DECIMAL(10, 2),

payment\_date DATE,

FOREIGN KEY (guest\_id) REFERENCES Guests(guest\_id),

FOREIGN KEY (reservation\_id) REFERENCES Reservations(reservation\_id)

);

CREATE TABLE IF NOT EXISTS Reviews (

review\_id SERIAL PRIMARY KEY,

guest\_id INT,

host\_id INT,

room\_id INT,

rating INT,

comment TEXT,

FOREIGN KEY (guest\_id) REFERENCES Guests(guest\_id),

FOREIGN KEY (host\_id) REFERENCES Hosts(host\_id),

FOREIGN KEY (room\_id) REFERENCES Rooms(room\_id)

);

INSERT

2. Write 3 rows (using INSERT queries) for each table in the data model

INSERT INTO Hosts (host\_name) VALUES

('John Doe'),

('Jane Smith'),

('Michael Johnson');

INSERT INTO Guests (guest\_name) VALUES

('Alice Brown'),

('Bob Green'),

('Charlie White');

INSERT INTO Rooms (host\_id, price, capacity, ac, refrigerator, wifi, private\_bathroom, pet\_friendly) VALUES

(1, 100.00, 2, 1, 1, 1, 1, 0),

(1, 120.00, 3, 1, 1, 1, 1, 1),

(2, 80.00, 1, 0, 1, 1, 0, 0);

INSERT INTO Reservations (guest\_id, room\_id, check\_in\_date, check\_out\_date, total\_price) VALUES

(1, 1, '2024-03-10', '2024-03-15', 500.00),

(2, 2, '2024-03-12', '2024-03-14', 240.00),

(3, 3, '2024-03-08', '2024-03-11', 240.00);

INSERT INTO Reviews (guest\_id, host\_id, room\_id, rating, comment) VALUES

(1, 1, 1, 4, 'Great host and comfortable room.'),

(2, 1, 2, 5, 'Amazing stay! Highly recommended.'),

(3, 2, 3, 3, 'Room was clean but a bit small.');

INSERT INTO Payments (guest\_id, reservation\_id, amount, payment\_date) VALUES

(1, 1, 500.00, '2024-03-10'),

(2, 2, 240.00, '2024-03-12'),

(3, 3, 240.00, '2024-03-08');

SELECT

3. Create the next analytic queries:

      1. Find a user who had the biggest amount of reservations. Return user name and user\_id

SELECT g.guest\_id, g.guest\_name

FROM Guests g

JOIN (

SELECT guest\_id, COUNT(\*) AS reservation\_count

FROM Reservations

GROUP BY guest\_id

ORDER BY reservation\_count DESC

LIMIT 1

) AS max\_reservations ON g.guest\_id = max\_reservations.guest\_id;

      2. (Optional) Find a host who earned the biggest amount of money for the last month. Return hostname and host\_id

SELECT h.host\_id, h.host\_name

FROM Hosts h

JOIN (

SELECT r.host\_id, SUM(p.amount) AS total\_earnings

FROM Reservations res

JOIN Rooms r ON res.room\_id = r.room\_id

JOIN Payments p ON res.reservation\_id = p.reservation\_id

WHERE DATE\_TRUNC('month', p.payment\_date) = DATE\_TRUNC('month', CURRENT\_DATE - INTERVAL '1' MONTH)

GROUP BY r.host\_id

ORDER BY total\_earnings DESC

LIMIT 1

) AS max\_earnings ON h.host\_id = max\_earnings.host\_id;

      3. (Optional) Find a host with the best average rating. Return hostname and host\_id

SELECT h.host\_id, h.host\_name

FROM Hosts h

JOIN (

SELECT host\_id, AVG(rating) AS avg\_rating

FROM Reviews

GROUP BY host\_id

ORDER BY avg\_rating DESC

LIMIT 1

) AS best\_avg\_rating ON h.host\_id = best\_avg\_rating.host\_id;

My DRAFT for PostgreSQL

-- ====================================================================

-- SELECT

-- ====================================================================

-- SELECT \* from Hosts;

-- SELECT \* from Guests;

-- SELECT \* from Rooms;

-- SELECT \* from Reservations;

-- SELECT \* from Payments;

-- SELECT \* from Reviews;

-- ====================================================================

-- DROP

-- ====================================================================

-- DROP TABLE IF EXISTS Reviews;

-- DROP TABLE IF EXISTS Payments;

-- DROP TABLE IF EXISTS Reservations;

-- DROP TABLE IF EXISTS Rooms;

-- DROP TABLE IF EXISTS Guests;

-- DROP TABLE IF EXISTS Hosts;

-- ====================================================================

-- CREATE

-- ====================================================================

-- CREATE TABLE IF NOT EXISTS Hosts (

-- host\_id SERIAL PRIMARY KEY,

-- host\_name VARCHAR(255)

-- );

-- CREATE TABLE IF NOT EXISTS Guests (

-- guest\_id SERIAL NOT NULL PRIMARY KEY,

-- guest\_name VARCHAR(255)

-- );

-- CREATE TABLE IF NOT EXISTS Rooms (

-- room\_id SERIAL PRIMARY KEY,

-- host\_id INT,

-- price DECIMAL(10, 2),

-- capacity INT,

-- ac BOOLEAN,

-- refrigerator BOOLEAN,

-- wifi BOOLEAN,

-- private\_bathroom BOOLEAN,

-- pet\_friendly BOOLEAN,

-- FOREIGN KEY (host\_id) REFERENCES Hosts(host\_id)

-- );

-- CREATE TABLE IF NOT EXISTS Reservations (

-- reservation\_id SERIAL PRIMARY KEY,

-- guest\_id INT,

-- room\_id INT,

-- check\_in\_date DATE,

-- check\_out\_date DATE,

-- total\_price DECIMAL(10, 2),

-- FOREIGN KEY (guest\_id) REFERENCES Guests(guest\_id),

-- FOREIGN KEY (room\_id) REFERENCES Rooms(room\_id)

-- );

-- CREATE TABLE IF NOT EXISTS Payments (

-- payment\_id SERIAL PRIMARY KEY,

-- guest\_id INT,

-- reservation\_id INT,

-- amount DECIMAL(10, 2),

-- payment\_date DATE,

-- FOREIGN KEY (guest\_id) REFERENCES Guests(guest\_id),

-- FOREIGN KEY (reservation\_id) REFERENCES Reservations(reservation\_id)

-- );

-- CREATE TABLE IF NOT EXISTS Reviews (

-- review\_id SERIAL PRIMARY KEY,

-- guest\_id INT,

-- host\_id INT,

-- room\_id INT,

-- rating INT,

-- comment TEXT,

-- FOREIGN KEY (guest\_id) REFERENCES Guests(guest\_id),

-- FOREIGN KEY (host\_id) REFERENCES Hosts(host\_id),

-- FOREIGN KEY (room\_id) REFERENCES Rooms(room\_id)

-- );

-- ====================================================================

-- INSERT

-- ====================================================================

-- INSERT INTO Hosts (host\_name) VALUES

-- ('John Doe'),

-- ('Jane Smith'),

-- ('Michael Johnson');

-- INSERT INTO Guests (guest\_name) VALUES

-- ('Alice Brown'),

-- ('Bob Green'),

-- ('Charlie White');

-- INSERT INTO Rooms (host\_id, price, capacity, ac, refrigerator, wifi, private\_bathroom, pet\_friendly) VALUES

-- (1, 100.00, 2, TRUE, TRUE, TRUE, TRUE, FALSE),

-- (1, 120.00, 3, TRUE, TRUE, TRUE, TRUE, TRUE),

-- (2, 80.00, 1, FALSE, TRUE, TRUE, FALSE, FALSE);

-- INSERT INTO Reservations (guest\_id, room\_id, check\_in\_date, check\_out\_date, total\_price) VALUES

-- (1, 1, '2024-03-10', '2024-03-15', 500.00),

-- (2, 2, '2024-03-12', '2024-03-14', 240.00),

-- (3, 3, '2024-03-08', '2024-03-11', 240.00);

-- INSERT INTO Reviews (guest\_id, host\_id, room\_id, rating, comment) VALUES

-- (1, 1, 1, 4, 'Great host and comfortable room.'),

-- (2, 1, 2, 5, 'Amazing stay! Highly recommended.'),

-- (3, 2, 3, 3, 'Room was clean but a bit small.');

-- INSERT INTO Payments (guest\_id, reservation\_id, amount, payment\_date) VALUES

-- (1, 1, 500.00, '2024-03-10'),

-- (2, 2, 240.00, '2024-03-12'),

-- (3, 3, 240.00, '2024-03-08');