Airline Database Project - Stage 2 **Yishai Dredzen and Asher Remer**

In this stage, we wrote several SQL statements and timed them using the methods we learnt in class.

Backups:

Backup Command with DROP, CREATE and INSERTS:

Run the following command in command line and input “postgres” user password:
pg_dump --file "backupPSQL.sql" --username "postgres" --format=c --large-objects --inserts --rows-per-insert "1000" --create --clean --if-exists --verbose "AirlineDB" > backupPSQL.log 2>&1

Restore Command:

To restore, run the following command in command line and input “postgres” user password:

pg_restore --username "postgres" --dbname "AirlineDB" --clean --if-exists --disable-triggers --verbose "backupPSQL.sql" > restorePSQL.log 2>&1

Backup Command:

Run the following command in command line and input “postgres” user password:

pg_dump --file "backupSQL.sql" --host "localhost" --port "5432" --username "postgres" --format=c --large-objects --verbose "AirlineDB" > backupSQL.log 2>&1

Restore Command:

To restore, run the following command in command line and input “postgres” user password:

pg_restore --host "localhost" --port "5432" --username "postgres" --dbname "AirlineDB" --clean --if-exists --disable-triggers --verbose "backupSQL.sql" > restoreSQL.log 2>&1

Here is a general overview of our queries that we wrote:

Select Queries:

1. List all flights departing from 'New York, USA' along with the number of available seats.
2. Calculate the average price of tickets in 'Business' class for each flight
3. Retrieve the contact information for passengers who have booked a flight to 'London, UK'
4. Sum the total cost of bookings for each passenger who has bookings in the 'Complete' status

Deletes:

5. Delete all bookings with status 'Cancelled' and return the count of deleted rows

6. Delete all tickets that 100 days have passed their flights

Updates:

7. Update the status of tickets to 'CheckedIn' for a flight departing on a specific date
8. Update the seat number for a specific ticket and ensure the seat is not already taken

Parameterized Queries:

9. Find flights departing on a specific date with available seats
10. Update ticket status based on user input and ensure the ticket exists
11. Delete bookings for a given passenger and return the count of deleted rows
12. Calculate the total cost of bookings within a date range for a specific passenger

The actual queries can be found in Queries.sql (1-8) and ParamQueries.sql (9-12) files.

Before indexing, the timing was as follows:

Query	Preparation Time (ms)	Execution Time (ms)
1	2.256	31.855
2	0.379	28.704
3	21.932	198.084
4	0.427	81.574
5	1.705	14.958
6	1.506	3.706
7	0.551	36.417
8	0.244	0.073

Query	Preparation Time (ms)	Execution Time (ms)
9	0.524	27.887
10	0.264	0.156
11	0.235	12.577
12	0.211	45.154

Indexes

We made the following indexes:

Booking Table:

Passengerid, status

Passengerid, cost

Flight Table:

ArrivalLocation

DepartureLocation

Ticket Table:

FlightNumber

Class, FlightNumber

FlightNumber, status

After indexing, timing was as follows:

Query	Preparation Time (ms)	Execution Time (ms)	Indexes Used
1	9.152	36.779	idx_flight_departurelocation
2	0.274	22.100	idx_ticket_class_flightnumber
3	16.124	74.296	idx_flight_arrivallocation
4	9.808	49.451	
5	0.118	7.690	idx_booking_passenger_status
6	0.074	3.157	
7	7.625	5.427	idx_ticket_flightnumber_status
8	0.419	0.152	idx_ticket_flightnumber_status

Checking Constraints:
Query: INSERT INTO SEAT VALUES(123, '19T');

ERROR: new row for relation "seat" violates check constraint "chk_seat_number"

DETAIL: Failing row contains (123, 19T).

Explanation: The constraint checks that the Seat is a possible seat which is checked using a regular expression. A seat is any two digit number followed by any of the letters A-K, excluding I.

QUERY: INSERT INTO Ticket (FlightNumber, SeatNumber, Price, Status, Class, PassengerID) VALUES (1, '12A', -100, 'Booked', 'Economy', 1);

ERROR: new row for relation "ticket" violates check constraint "chk_price"

DETAIL: Failing row contains (3, 1, 12A, -100, Booked, Economy, 1).

Explanation: The constraint checks that price is greater than zero. The error above occurred due to inputting -100 as the price.

QUERY: DELETE FROM Seat WHERE SeatNumber LIKE '%E';

ERROR: update or delete on table "seat" violates foreign key constraint "ticket_flightnumber_seatnumber_fkey" on table "ticket"

DETAIL: Key (flightnumber, seatnumber)=(484, 12E) is still referenced from table "ticket".

Explanataion: The delete doesn't work due to the seat number being referenced to from the Ticket table.

QUERY: INSERT INTO booking(PassengerID, BookingDate, status, cost, ticketnumber) VALUES (128, '12-12-2024', 'Pendin

g', -999.50, 43);

ERROR: New row for relation "booking" violates check constraint "chk_price"

DETAIL: Failing row contains (1, 128, 2024-12-12, Pending, -999.5, 43).

Explanation: The constraint ensures that the cost field is positive, the attempted input had a negative cost so an error was thrown.

QUERY: INSERT INTO Flight (DepartureLocation, ArrivalLocation, DepartureTime, ArrivalTime, Capacity) VALUES ('New York', 'Perth' , '2024-07-01 08:00:00', '2024-07-02 05:00:00', 250);

ERROR: new row for relation "flight" violates check constraint "chk_flight_duration"

DETAIL: Failing row contains (11, New York, Perth, 2024-07-01 08:00:00, 2024-07-02 05:00:00, 250).

Explanation: The constraint throws an error as after doing some research, the longest flight a plane can perform is 19 hours.