Experiment 2: Horizontal Fragmentation

- Name: Srikanth Iyengar
- UID: 2020400062
- Branch: IT
- Batch: G

- Subject: Advanced Database Management System

Aim:

Design a distributed database by applying the concept of horizontal fragmentation

Implementation

Flight Table

Flight Id	Flight Name	Source	Destination	Tickets AVailable
1	Transatlantic Express	New York	London	100
2	City Hopper	London	Paris	90
3	European Voyager	Paris	Berlin	80
4	Continent Cruiser	Berlin	Rome	70
5	Mediterranean Flyer	Rome	Madrid	60
6	Iberian Explorer	Madrid	Barcelona	50
7	Nordic Adventurer	Barcelona	Amsterdam	40
8	Oceanic Odyssey	Amsterdam	New York	30
9	Atlantic Ascension	New York	London	20
10	Skyline Shuttle	London	Paris	10
11	Continental Commuter	Paris	Berlin	20
12	Bella Italia	Berlin	Rome	30
13	España Express	Rome	Madrid	40
14	Dutch Delight	Madrid	Barcelona	50
15	Viking Venture	Barcelona	Amsterdam	60
16	Big Apple Bounce	Amsterdam	New York	70
17	British Airways	New York	London	80
18	French Connection	London	Paris	90
19	German Getaway	Paris	Berlin	100
20	Italian Holiday	Berlin	Rome	110

App user Table

Username	Password	Tickets Booked	Location
johndoe123	5f4dcc3b5aa765d61d8327deb882cf99	2	New York
janedoe456	5e884898da28047151d0e56f8dc62927	1	London
bobsmith789	21232f297a57a5a743894a0e4a801fc3	4	Paris
alicebrown101	ee11cbb19052e40b07aac0ca060c23ee	0	Berlin
tomgreen202	25f9e794323b453885f5181f1b624d0b	3	Madrid
jennypink303	a665a45920422f9d417e4867efdc4fb8	5	Rome
chrisblack404	5d41402abc4b2a76b9719d911017c592	6	Moscow
kimwhite505	bdbafc1b5f5dc8ab2f0196b5cd543a33	7	Toronto
paulred606	1a1dc91c907325c69271ddf0c944bc72	8	Sydney
lucyyellow707	ac0eafeed417f0ffa7befa0d1c01a7a6	9	Seoul
mikeblue808	098f6bcd4621d373cade4e832627b4f6	10	Tokyo
sarahpurple909	0b7c75b0f4b7e585a1c73e72409f3b7c	11	Shanghai
davidorange010	76a2173be6393254e72ffa4d6df1030a	12	Beijing
emmablack112	c1dfd96eea8cc2b62785275bca38ac26	13	Mexico City
danielgreen213	98f13708210194c475687be6106a3bce	14	Rio de Janeiro
elizabethpurple314	f0d2f1e2dc9abceb877557a483e07c3a	15	Sao Paulo
richardblue415	5ab5dbc27120680fa6f0b6ed7b6c0098	16	Buenos Aires
jenniferred516	7c222fb2927d828af22f592134e89324	17	Lima
robertyellow617	7a74f16c3e7d20a03b356f7bcf8c8f7a	18	Bogota

Query for creation of table

```
drop table if exists app_user;

CREATE TABLE IF NOT EXISTS public.app_user
(
    username character varying(100) COLLATE pg_catalog."default" NOT NULL,
    user_password character varying(100) COLLATE pg_catalog."default",
    tickets_booked integer,
    address varchar(100),
    CONSTRAINT app_user_pkey PRIMARY KEY (username)
);

INSERT INTO public.app_user (username, user_password, tickets_booked, address)
```

```
VALUES
('johndoe123', '5f4dcc3b5aa765d61d8327deb882cf99', 2, 'New York'),
('janedoe456', '5e884898da28047151d0e56f8dc62927', 1, 'London'),
('bobsmith789', '21232f297a57a5a743894a0e4a801fc3', 4, 'Paris'),
('alicebrown101', 'ee11cbb19052e40b07aac0ca060c23ee', 0, 'Berlin'),
('tomgreen202', '25f9e794323b453885f5181f1b624d0b', 3, 'Madrid'),
('jennypink303', 'a665a45920422f9d417e4867efdc4fb8', 5, 'Rome'),
('chrisblack404', '5d41402abc4b2a76b9719d911017c592', 6, 'Moscow'),
('kimwhite505', 'bdbafc1b5f5dc8ab2f0196b5cd543a33', 7, 'Toronto'),
('paulred606', '1a1dc91c907325c69271ddf0c944bc72', 8, 'Sydney'),
('lucyyellow707', 'ac0eafeed417f0ffa7befa0d1c01a7a6', 9, 'Seoul'),
('mikeblue808', '098f6bcd4621d373cade4e832627b4f6', 10, 'Tokyo'),
('sarahpurple909', '0b7c75b0f4b7e585a1c73e72409f3b7c', 11, 'Shanghai'),
('davidorange010', '76a2173be6393254e72ffa4d6df1030a', 12, 'Beijing'),
('emmablack112', 'c1dfd96eea8cc2b62785275bca38ac26', 13, 'Mexico City'),
('danielgreen213', '98f13708210194c475687be6106a3bce', 14, 'Rio de Janeiro'),
('elizabethpurple314', 'f0d2f1e2dc9abceb877557a483e07c3a', 15, 'Sao Paulo'),
('richardblue415', '5ab5dbc27120680fa6f0b6ed7b6c0098', 16, 'Buenos Aires'),
('jenniferred516', '7c222fb2927d828af22f592134e89324', 17, 'Lima'),
('robertyellow617', '7a74f16c3e7d20a03b356f7bcf8c8f7a', 18, 'Bogota');

drop table if exists flight;

CREATE TABLE IF NOT EXISTS public.flight
(
    flight_id integer NOT NULL,
    flight_name character varying(100) COLLATE pg_catalog."default" NOT NULL,
    flight_source character varying(100) COLLATE pg_catalog."default" NOT NULL,
    flight_destination character varying(100) COLLATE pg_catalog."default" NOT NULL,
    tickets_available integer NOT NULL,
    CONSTRAINT flight_pkey PRIMARY KEY (flight_id)
);
-------------------------------------------------------------------------------
---


insert into flight values
(1, 'Transatlantic Express', 'New York', 'London', 100),
(2, 'City Hopper', 'London', 'Paris', 90),
(3, 'European Voyager', 'Paris', 'Berlin', 80),
(4, 'Continent Cruiser', 'Berlin', 'Rome', 70),
(5, 'Mediterranean Flyer', 'Rome', 'Madrid', 60),
(6, 'Iberian Explorer', 'Madrid', 'Barcelona', 50),
(7, 'Nordic Adventurer', 'Barcelona', 'Amsterdam', 40),
(8, 'Oceanic Odyssey', 'Amsterdam', 'New York', 30),
(9, 'Atlantic Ascension', 'New York', 'London', 20),
(10, 'Skyline Shuttle', 'London', 'Paris', 10),
(11, 'Continental Commuter', 'Paris', 'Berlin', 20),
(12, 'Bella Italia', 'Berlin', 'Rome', 30),
(13, 'España Express', 'Rome', 'Madrid', 40),
(14, 'Dutch Delight', 'Madrid', 'Barcelona', 50),
(15, 'Viking Venture', 'Barcelona', 'Amsterdam', 60),
```

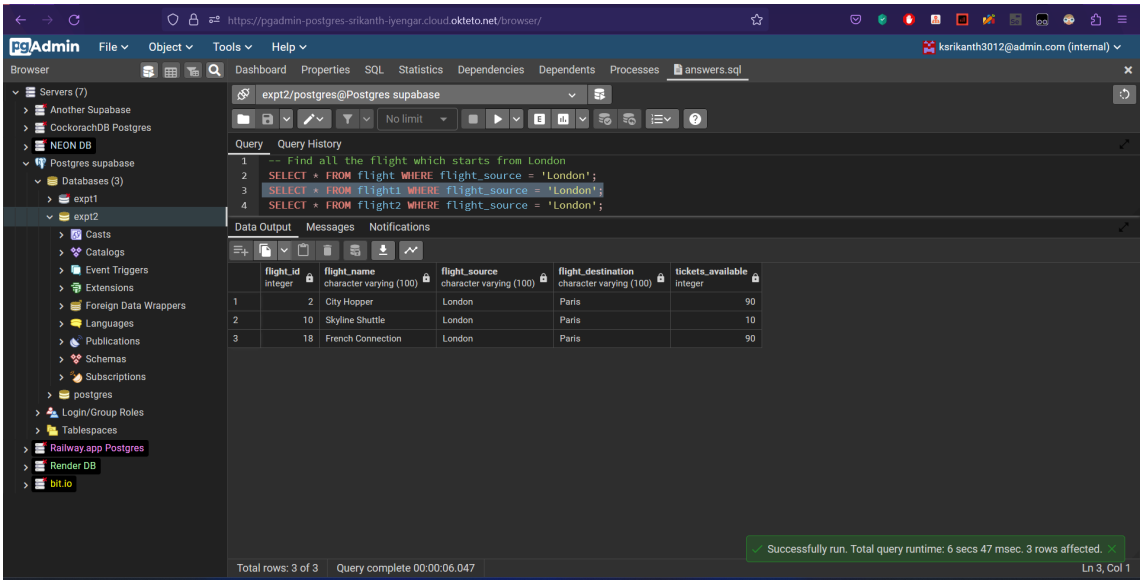
```
(16, 'Big Apple Bounce', 'Amsterdam', 'New York', 70),
(17, 'British Airways', 'New York', 'London', 80),
(18, 'French Connection', 'London', 'Paris', 90),
(19, 'German Getaway', 'Paris', 'Berlin', 100),
(20, 'Italian Holiday', 'Berlin', 'Rome', 110);
;

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

drop table if exists flight1, flight2, app_user1, app_user2;


CREATE TABLE flight1 (LIKE flight);
CREATE TABLE flight2 (LIKE flight);


CREATE TABLE app_user1 (LIKE app_user);
CREATE TABLE app_user2 (LIKE app_user);




insert into flight1 (SELECT * FROM flight where flight_id % 2 = 0);
insert into flight2 (SELECT * FROM flight WHERE flight_id % 2 = 1);


insert into app_user1 (SELECT * FROM app_user WHERE tickets_booked > 10);
insert into app_user2 (SELECT * FROM app_user WHERE tickets_booked <= 10);
```

PART A

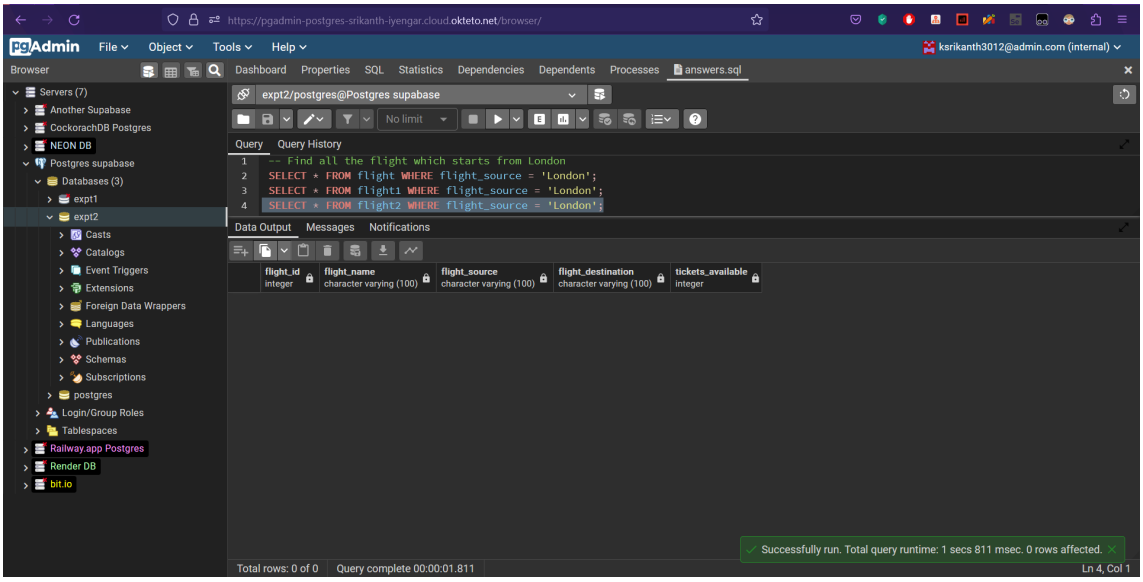
Query 1

```
-- Find all the flight which starts from London
SELECT * FROM flight WHERE flight_source = 'London';
SELECT * FROM flight1 WHERE flight_source = 'London';
SELECT * FROM flight2 WHERE flight_source = 'London';
```

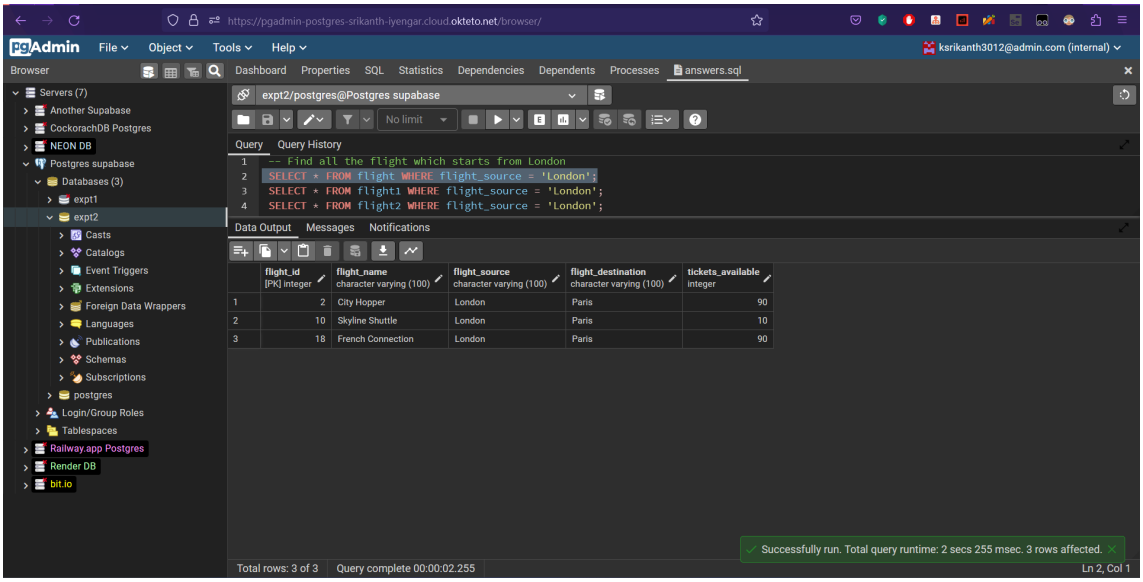
Result From table 1



Result From table 2



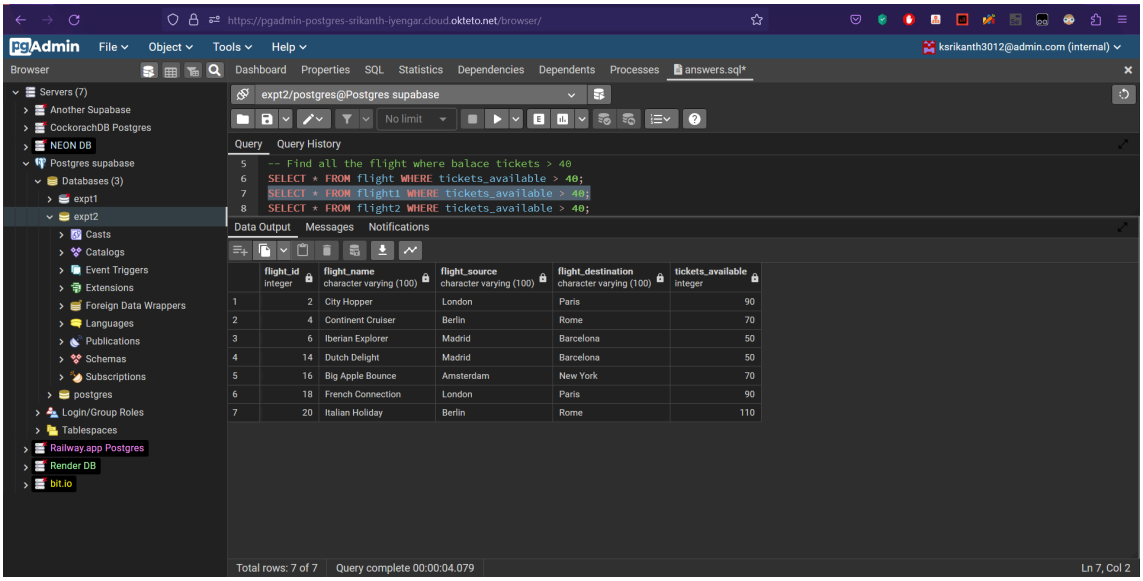
Result From original Table



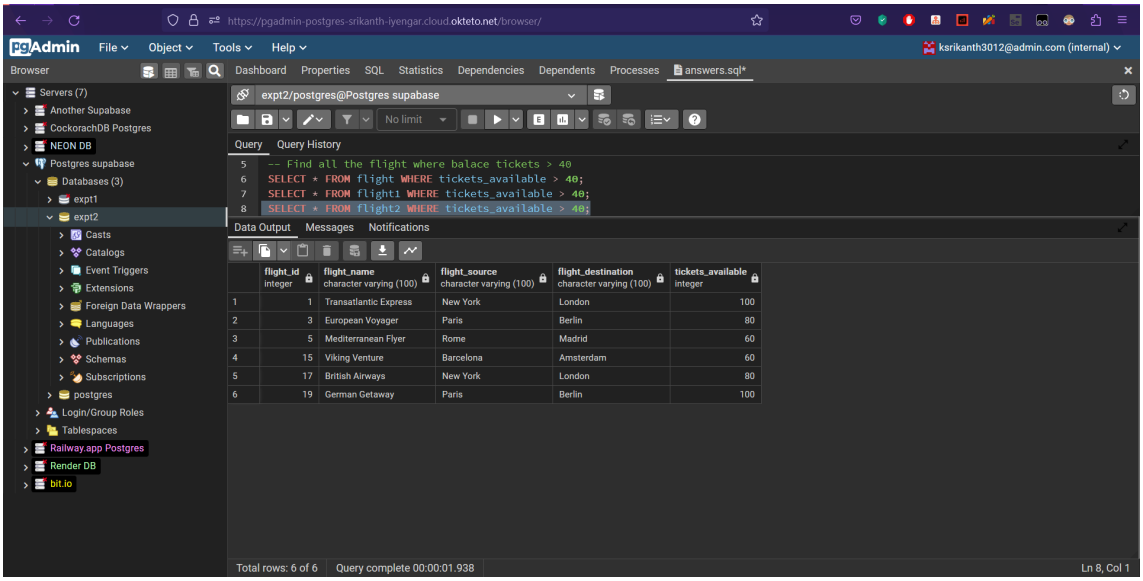
Query 2

```sql
-- Find all the flight where balace tickets > 40
SELECT * FROM flight WHERE tickets_available > 40;
SELECT * FROM flight1 WHERE tickets_available > 40;
SELECT * FROM flight2 WHERE tickets_available > 40;
```

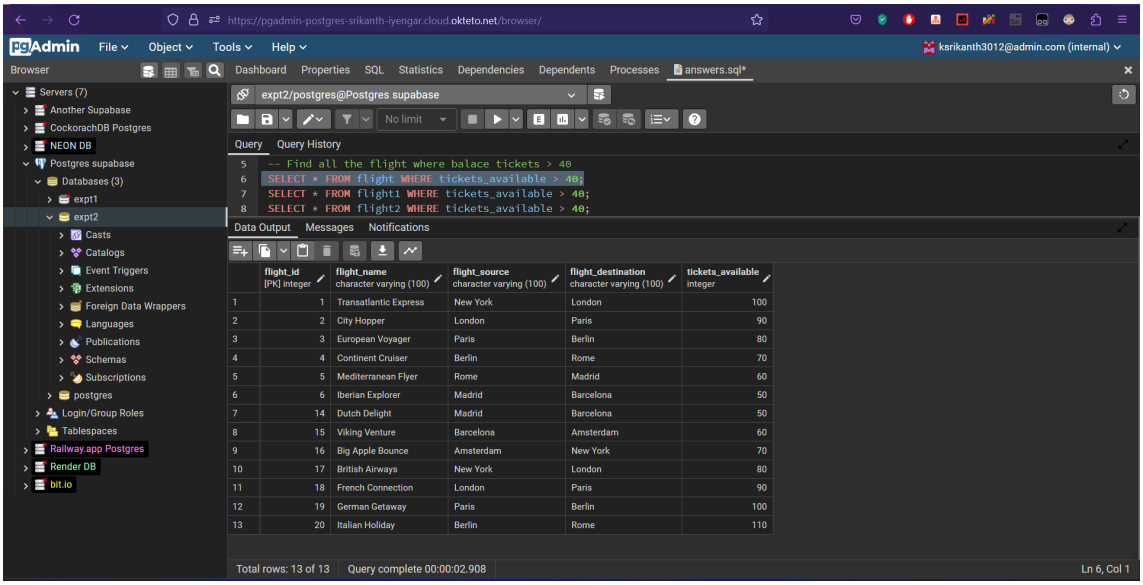
Result From table 1



Result From table 2



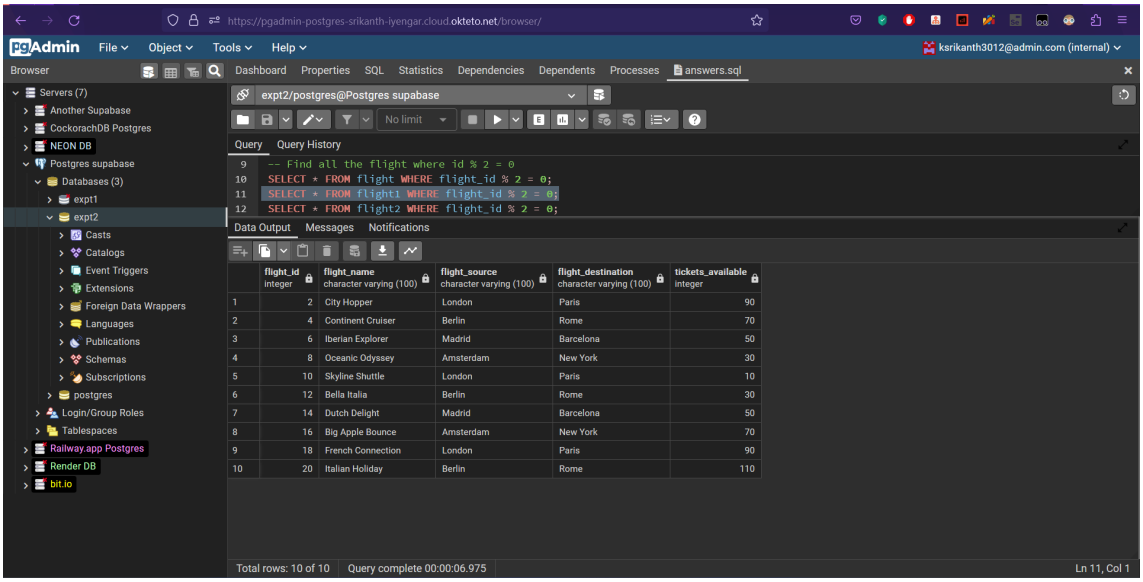
Result From original table



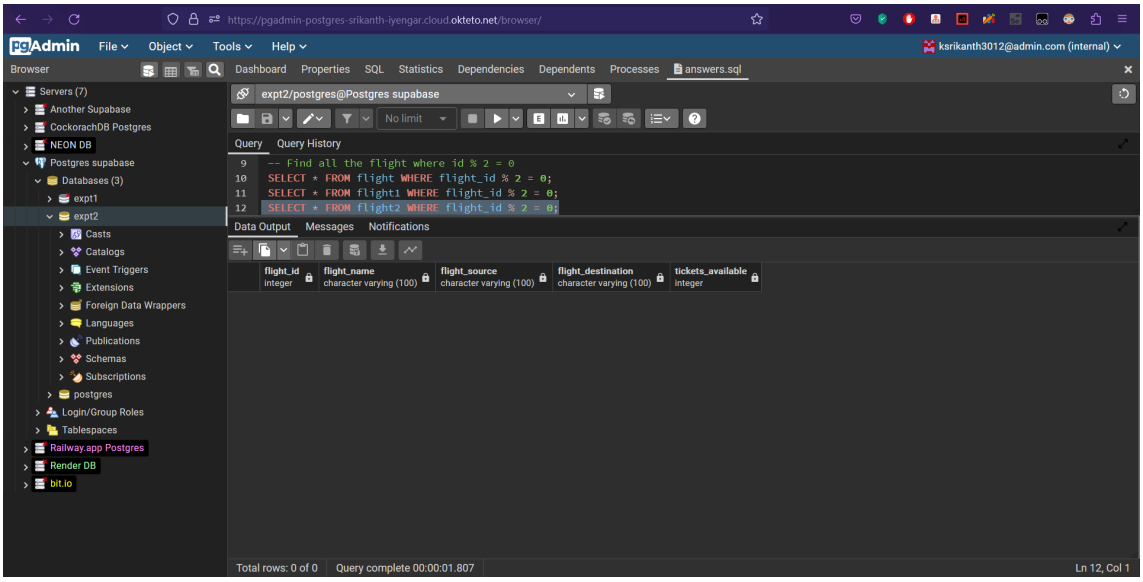
Query 3

```sql
-- Find all the flight where id % 2 = 0
SELECT * FROM flight WHERE flight_id % 2 = 0;
SELECT * FROM flight1 WHERE flight_id % 2 = 0;
SELECT * FROM flight2 WHERE flight_id % 2 = 0;
```

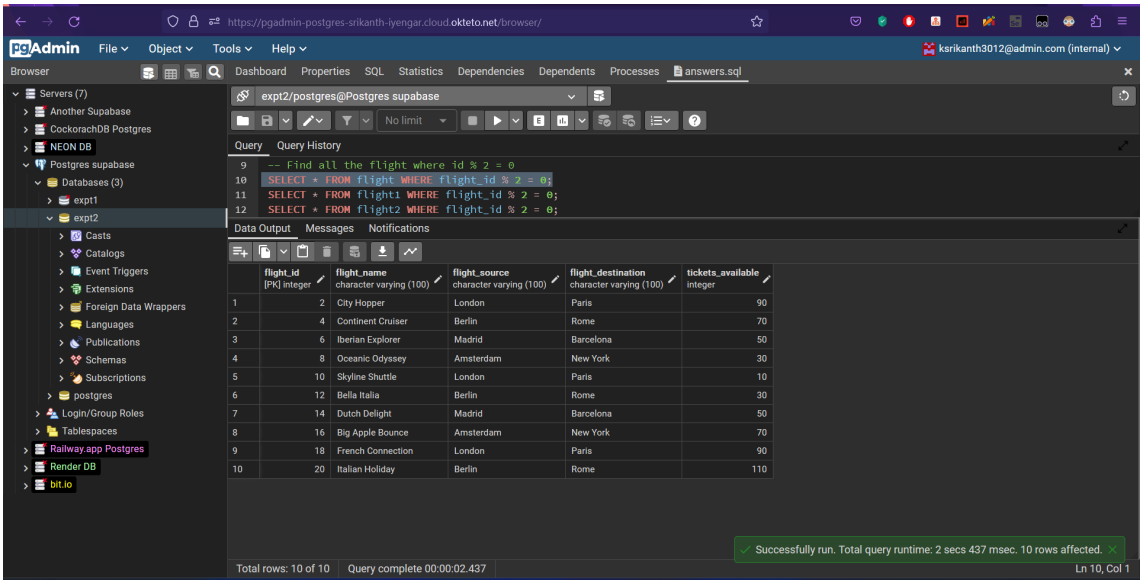
Result From table 1



Result From table 2



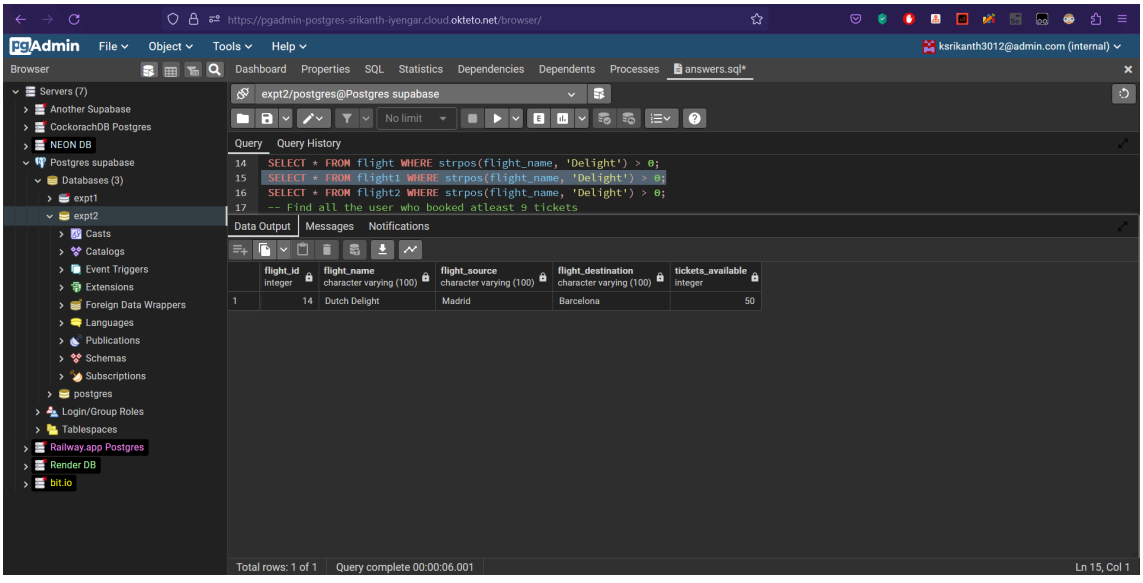
Result From original table



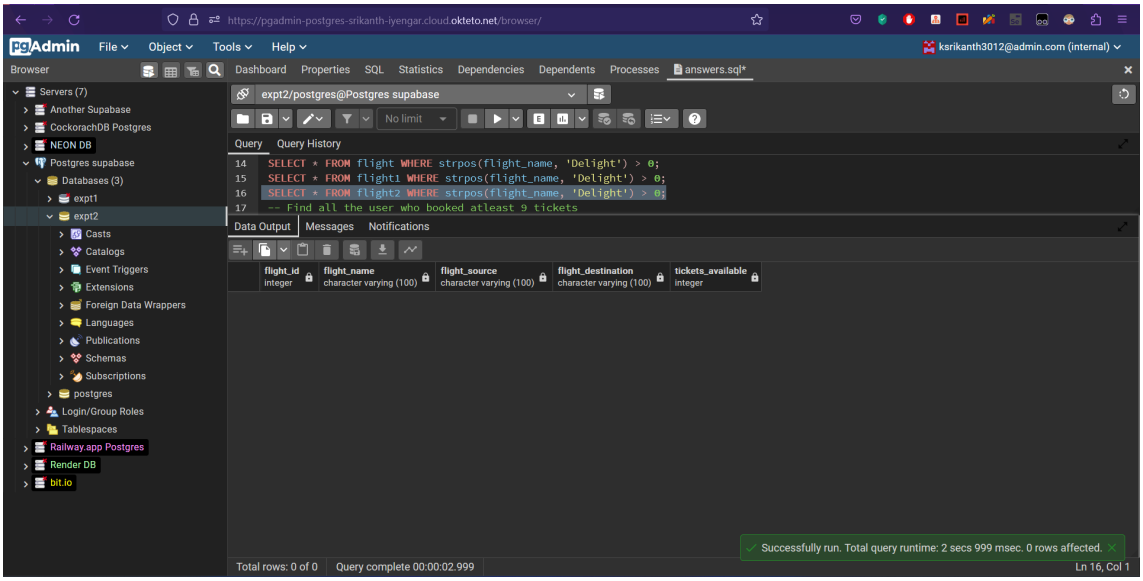
Query 4

```sql
-- Find all the flights where flight name contains express
SELECT * FROM flight WHERE strpos(flight_name, 'Delight') > 0;
SELECT * FROM flight1 WHERE strpos(flight_name, 'Delight') > 0;
SELECT * FROM flight2 WHERE strpos(flight_name, 'Delight') > 0;
```

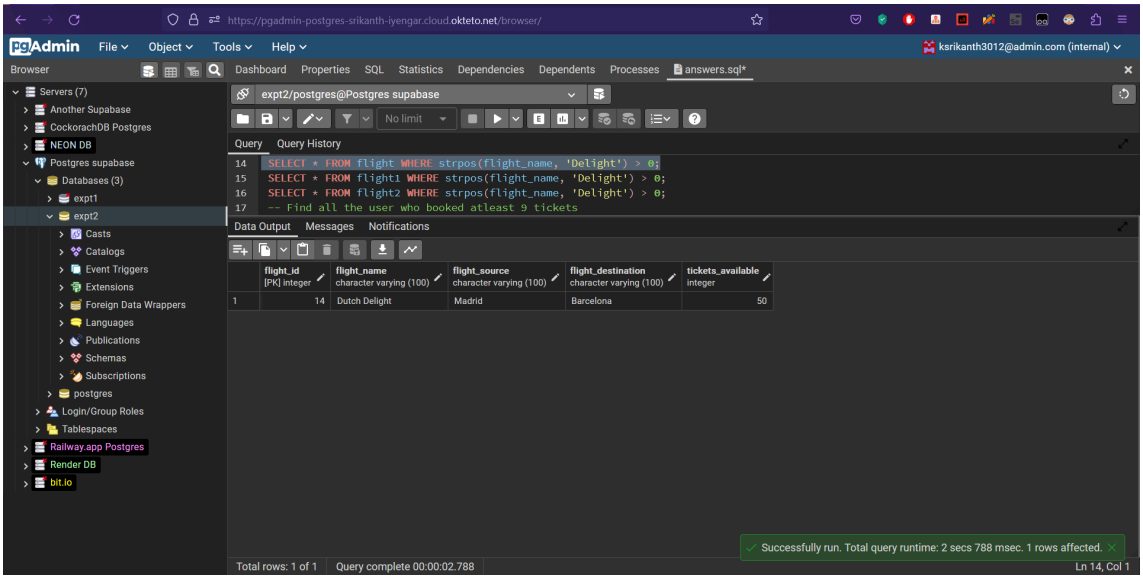
Result From table 1



Result From table 2



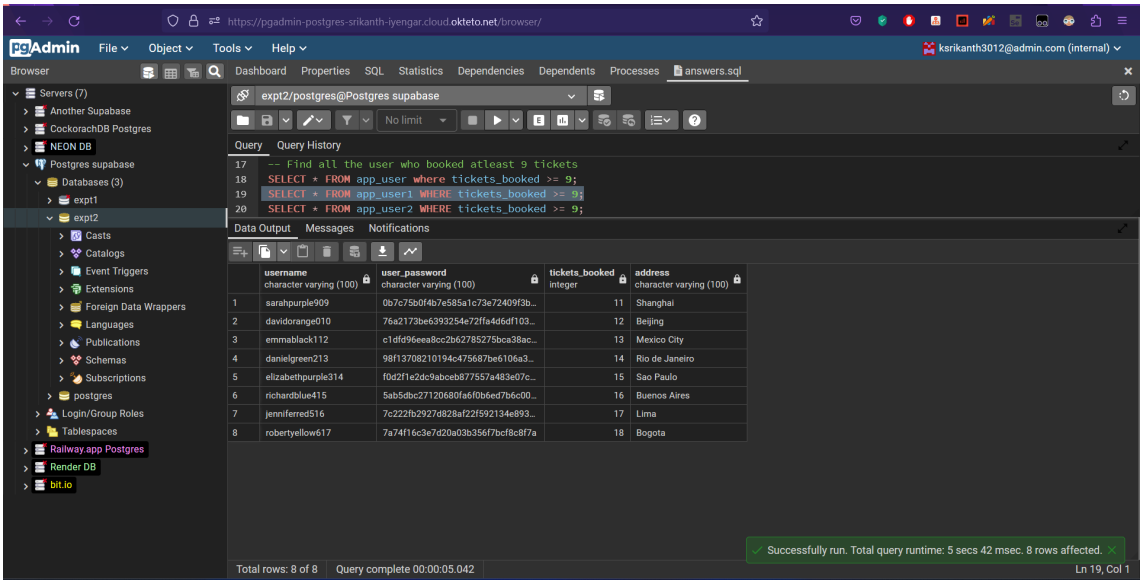
Result From original table



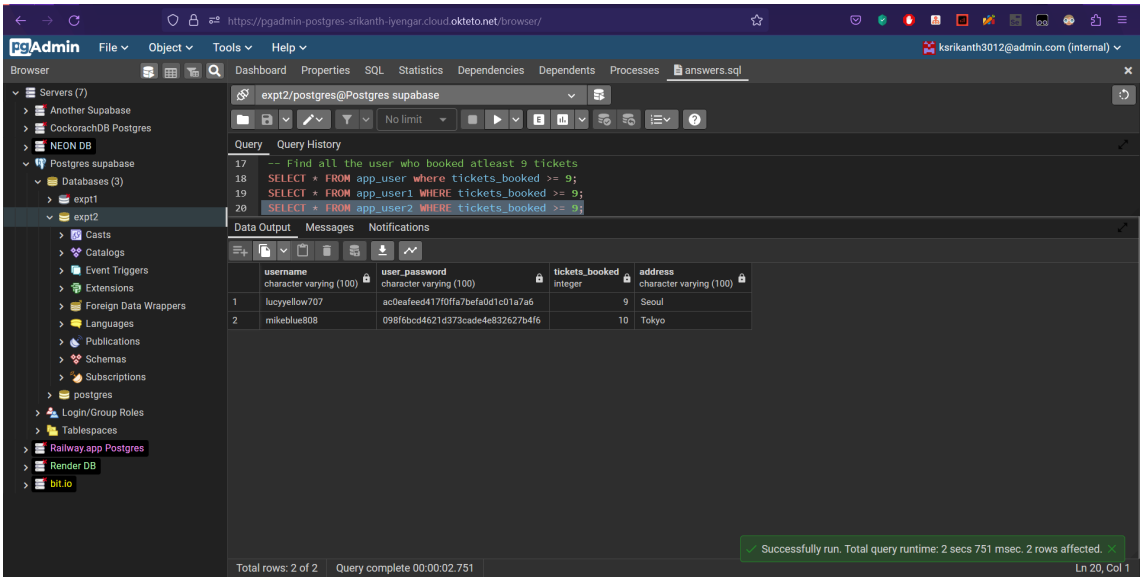
Query 5

```
-- Find all the user who booked atleast 9 tickets
SELECT * FROM app_user where tickets_booked >= 9;
SELECT * FROM app_user1 WHERE tickets_booked >= 9;
SELECT * FROM app_user2 WHERE tickets_booked >= 9;
```

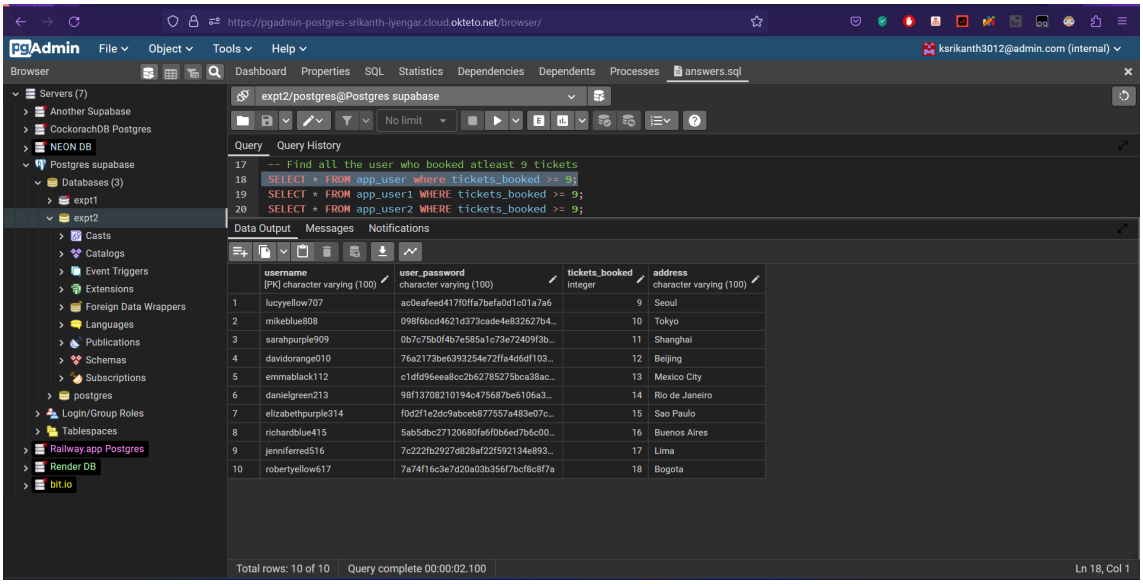
Result From table 1



Result From table 2



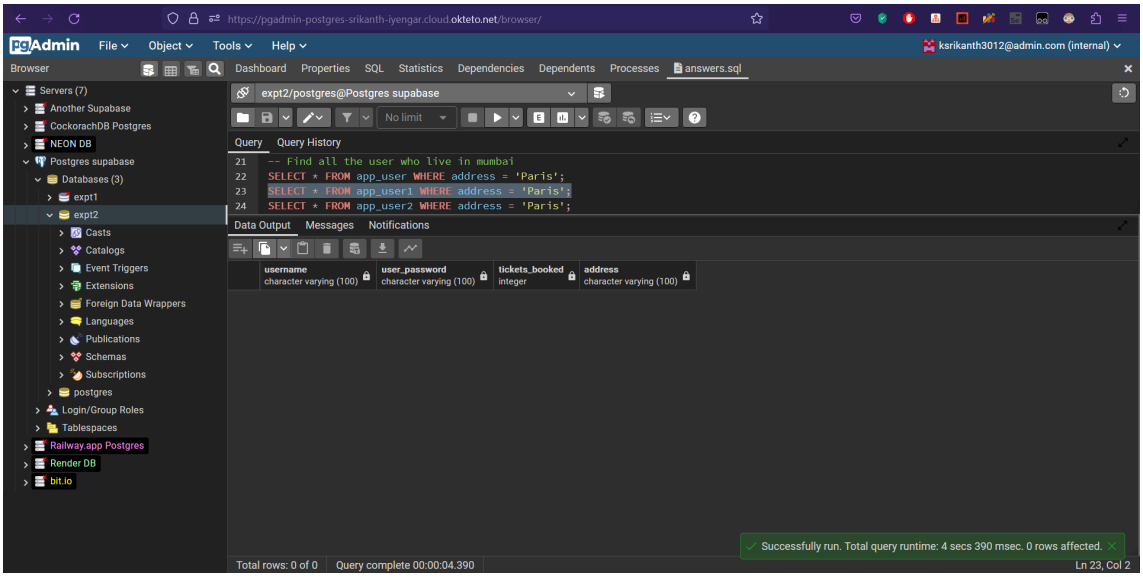
Result From original table



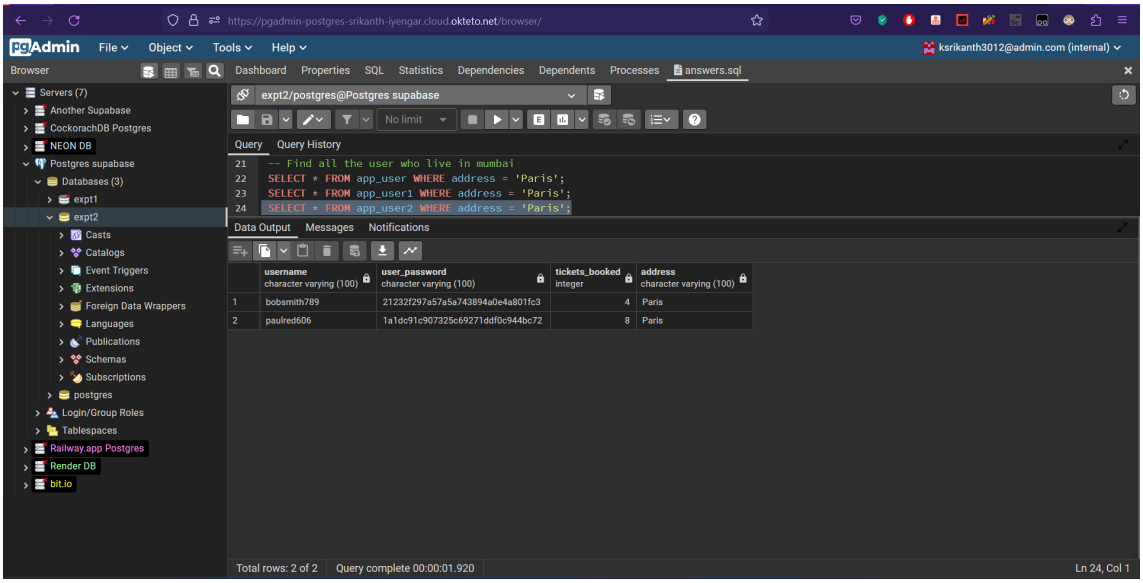
Query 6

```sql
-- Find all the user who live in mumbai
SELECT * FROM app_user WHERE address = 'Paris';
SELECT * FROM app_user1 WHERE address = 'Paris';
SELECT * FROM app_user2 WHERE address = 'Paris';
```

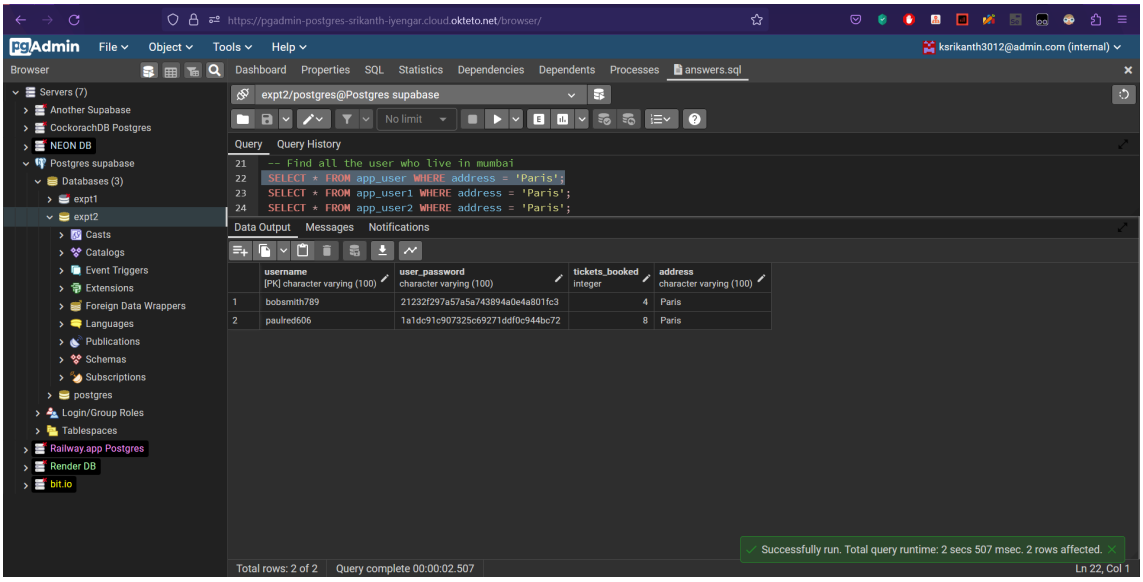
Result From table 1



Result From table 2



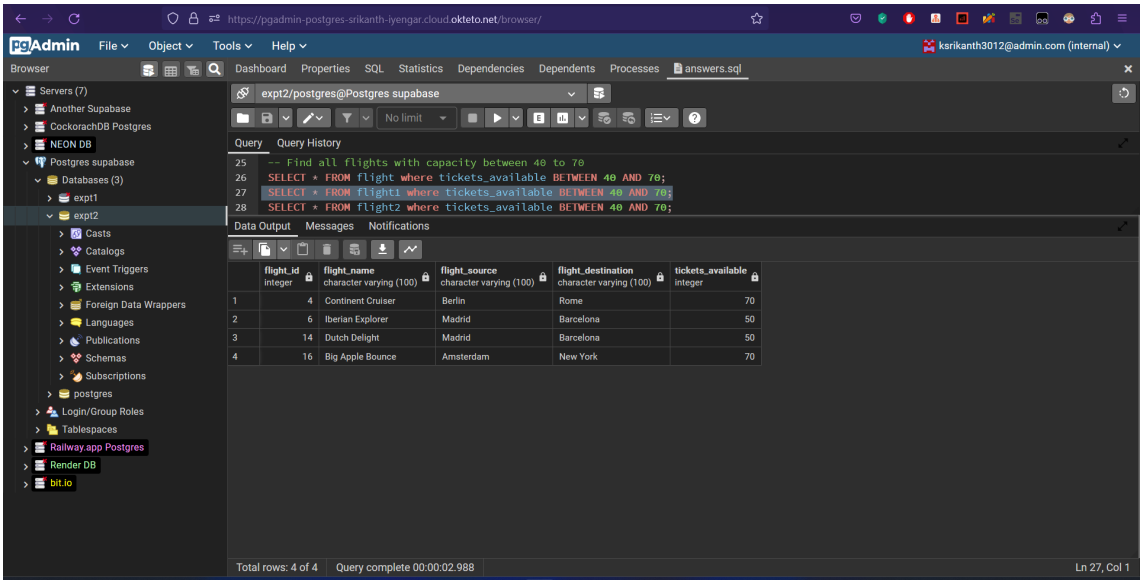
Result From original table



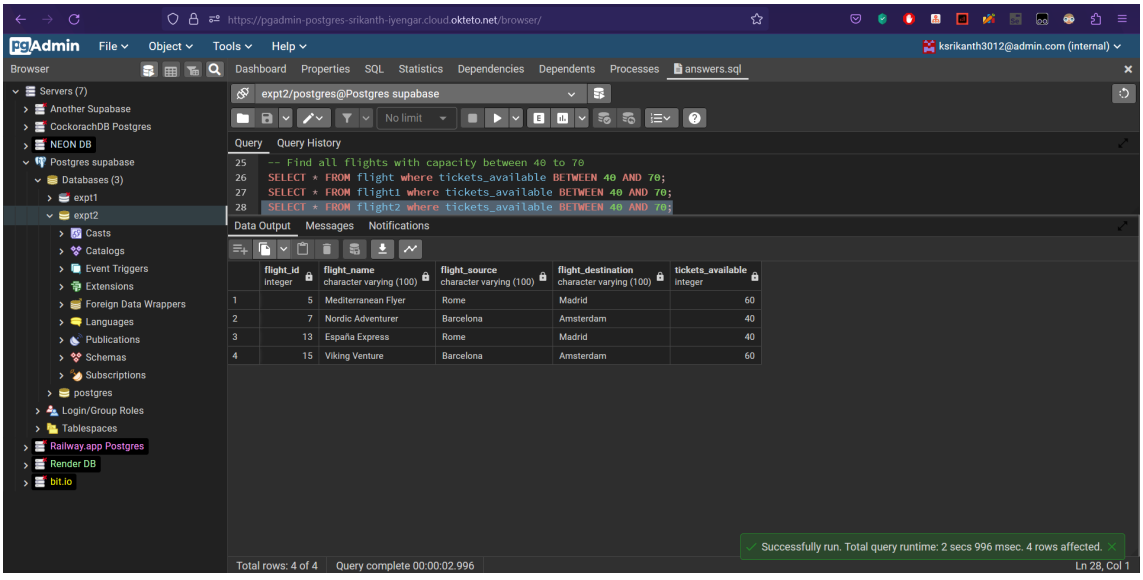
Query 7

```
-- Find all flights with capacity between 40 to 70
SELECT * FROM flight where tickets_available BETWEEN 40 AND 70;
SELECT * FROM flight1 where tickets_available BETWEEN 40 AND 70;
SELECT * FROM flight2 where tickets_available BETWEEN 40 AND 70;
```

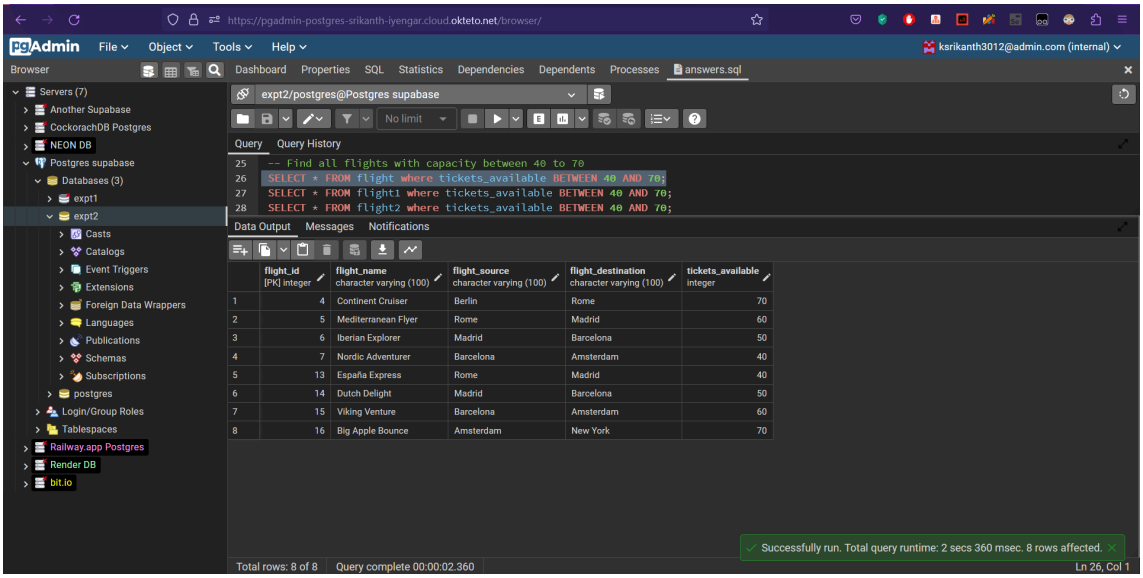
Result From table 1



Result From table 2



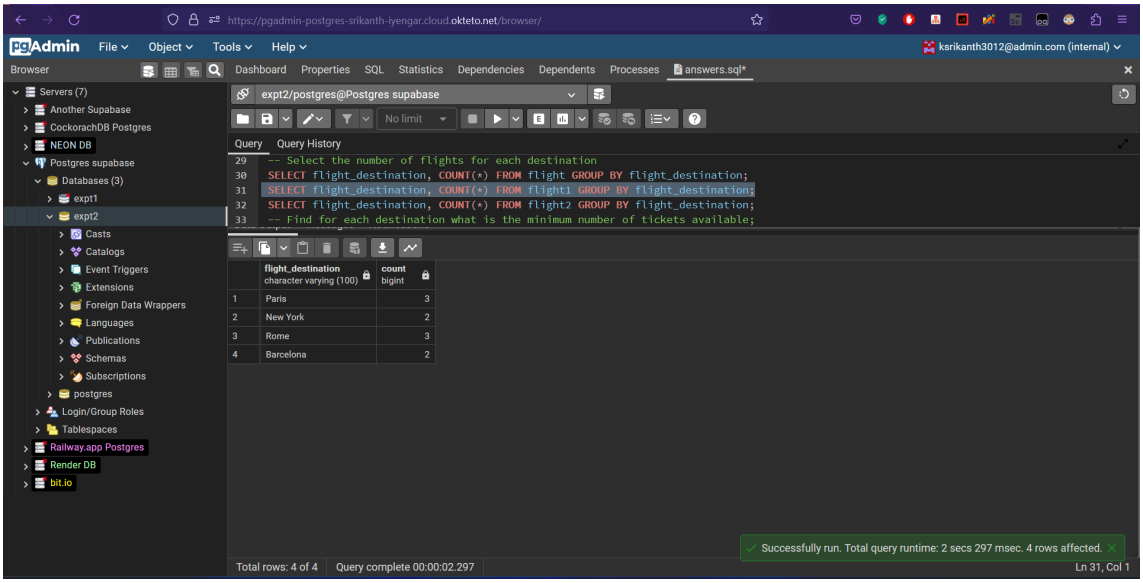
Result From original table



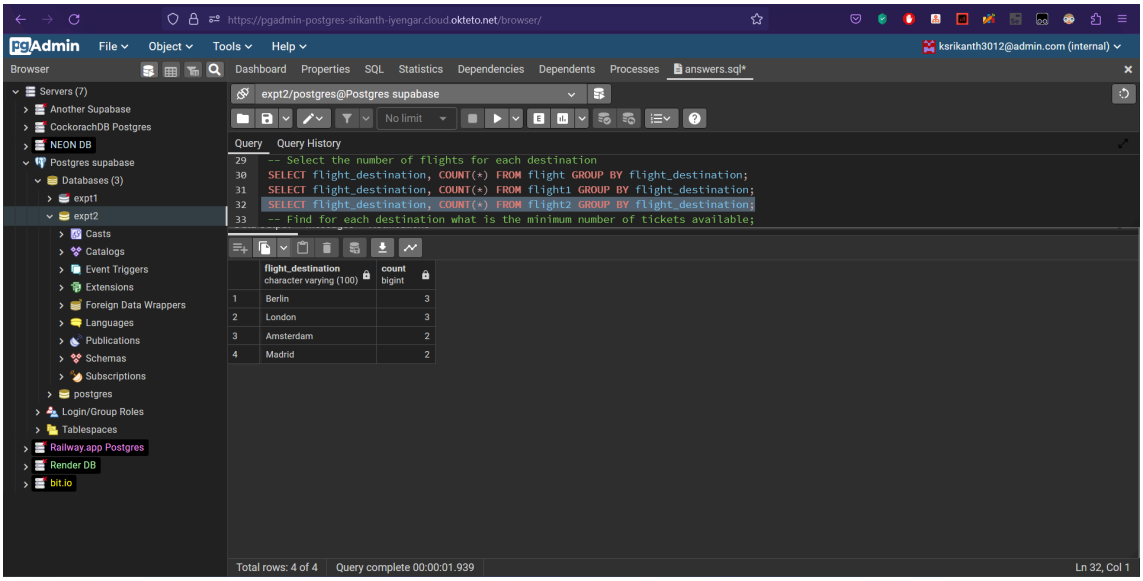
Query 8

```sql
-- Select the number of flights for each destination
SELECT flight_destination, COUNT(*) FROM flight GROUP BY flight_destination;
SELECT flight_destination, COUNT(*) FROM flight1 GROUP BY flight_destination;
SELECT flight_destination, COUNT(*) FROM flight2 GROUP BY flight_destination;
```

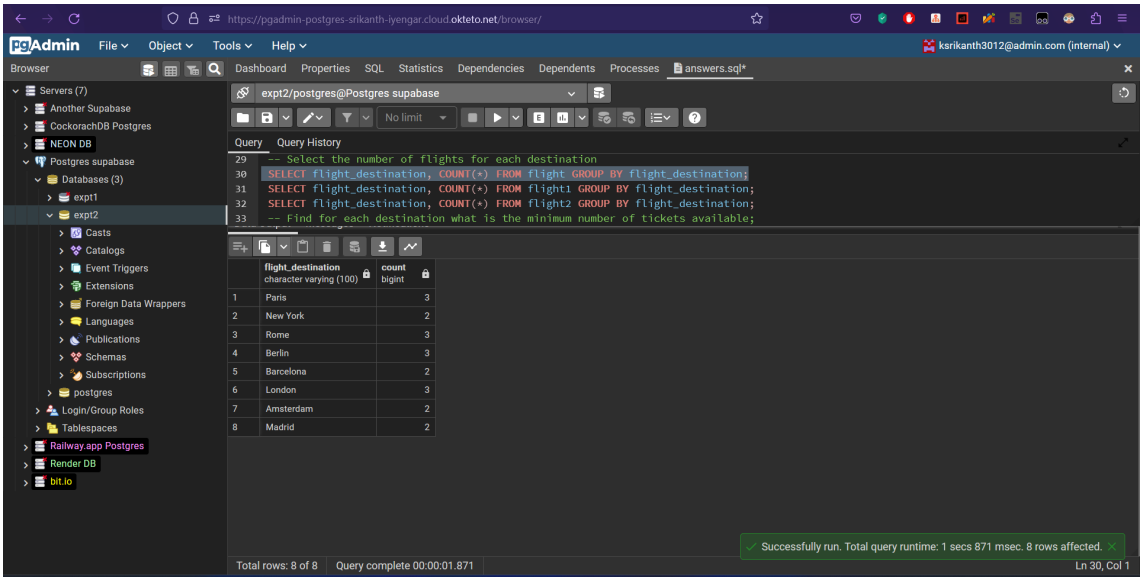
Result From table 1



Result From table 2



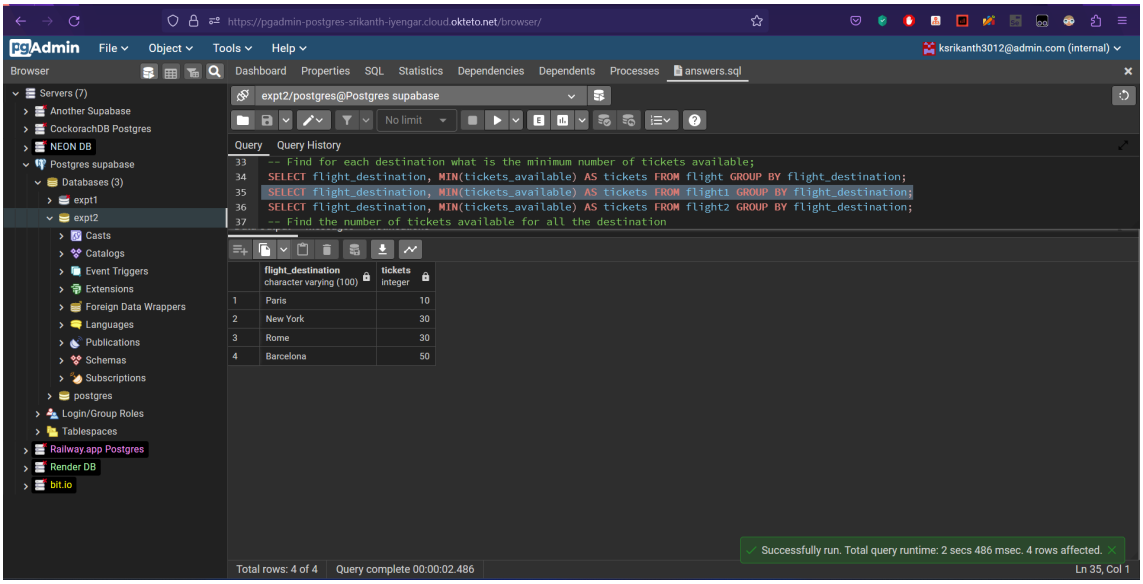
Result From original table



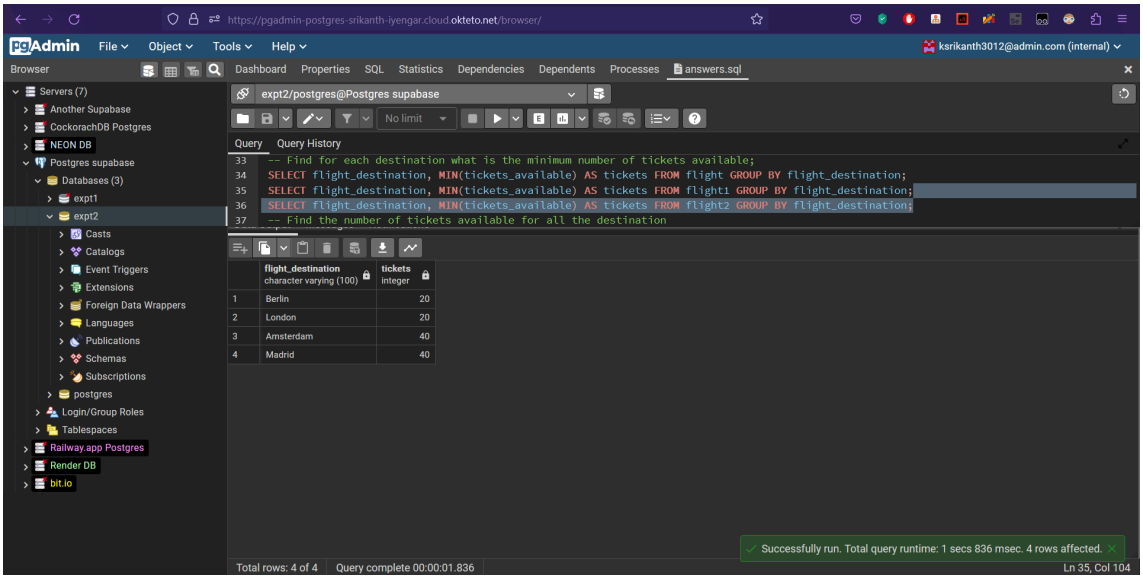
Query 9

```
 -- Find for each destination what is the minimum number of tickets available;
 SELECT flight_destination, MIN(tickets_available) AS tickets FROM flight GROUP BY
flight_destination;
 SELECT flight_destination, MIN(tickets_available) AS tickets FROM flight1 GROUP BY
flight_destination;
 SELECT flight_destination, MIN(tickets_available) AS tickets FROM flight2 GROUP BY
flight_destination;
```

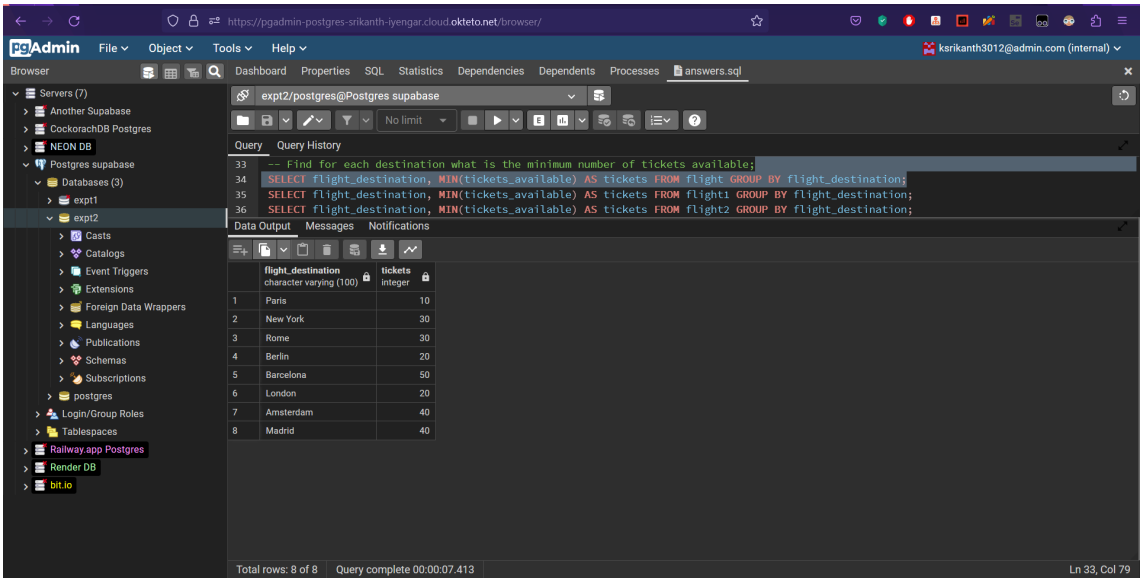
Result From table 1



Result From table 2



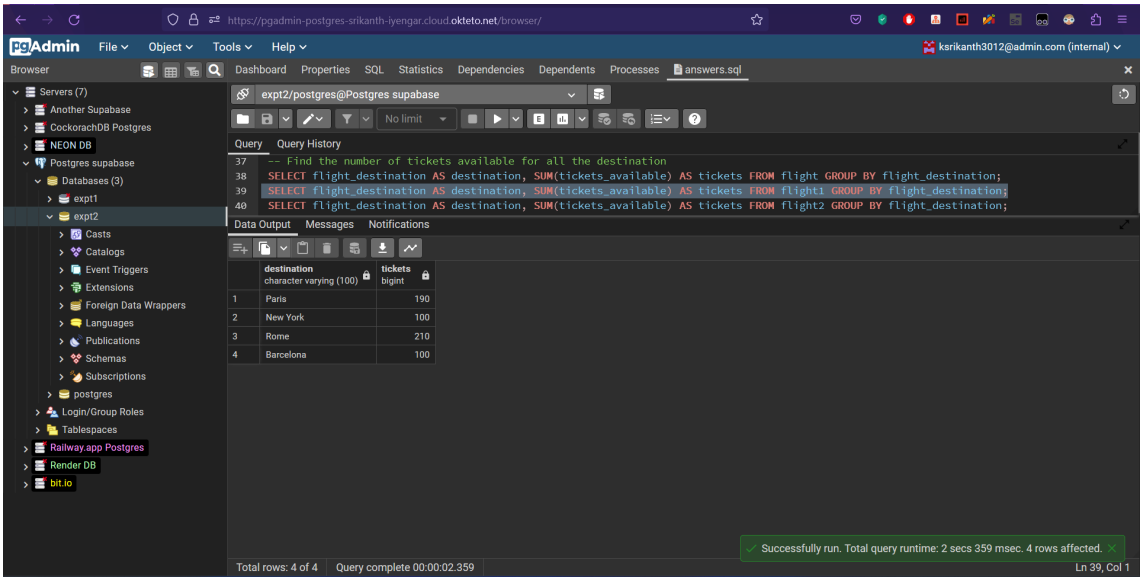
Result From original table



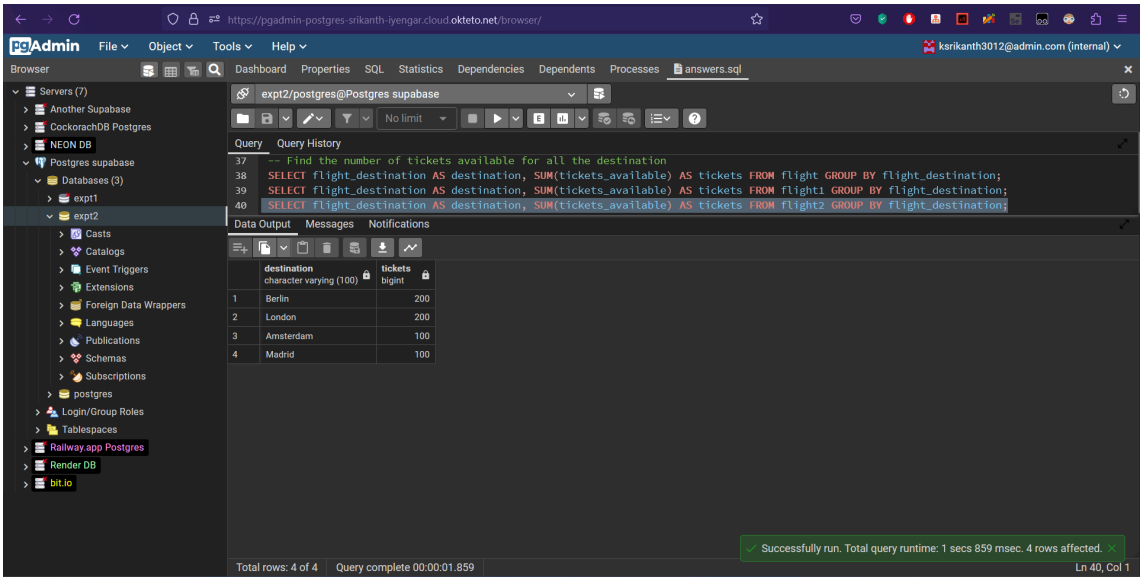
Query 10

```
 -- Find the number of tickets available for all the destination
 SELECT flight_destination AS destination, SUM(tickets_available) AS tickets FROM
flight GROUP BY flight_destination;
 SELECT flight_destination AS destination, SUM(tickets_available) AS tickets FROM
flight1 GROUP BY flight_destination;
 SELECT flight_destination AS destination, SUM(tickets_available) AS tickets FROM
flight2 GROUP BY flight_destination;
```

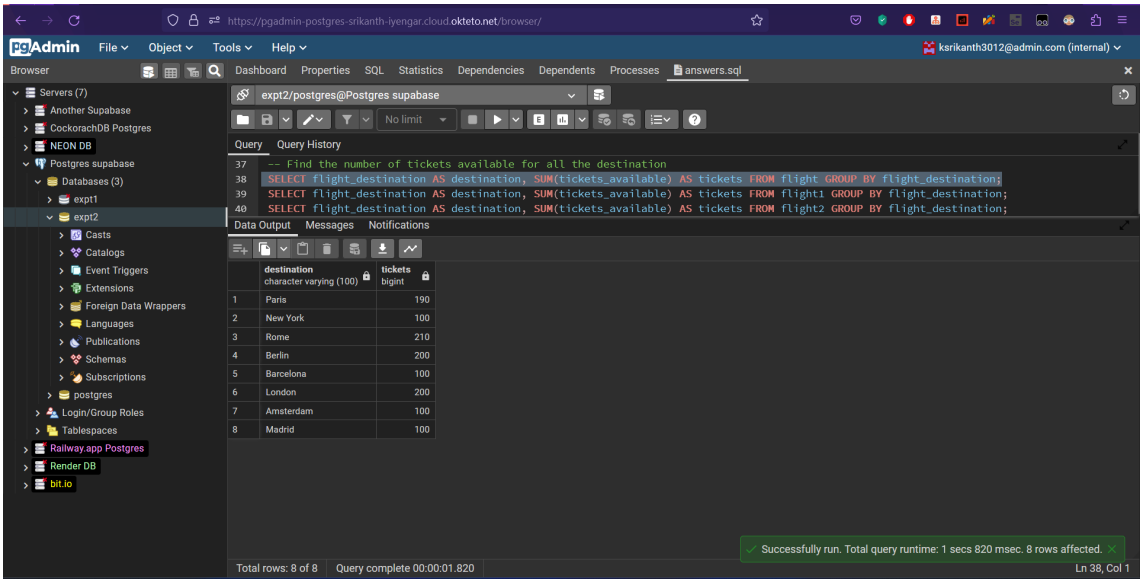
Result From table 1



Result From table 2



Result From original table



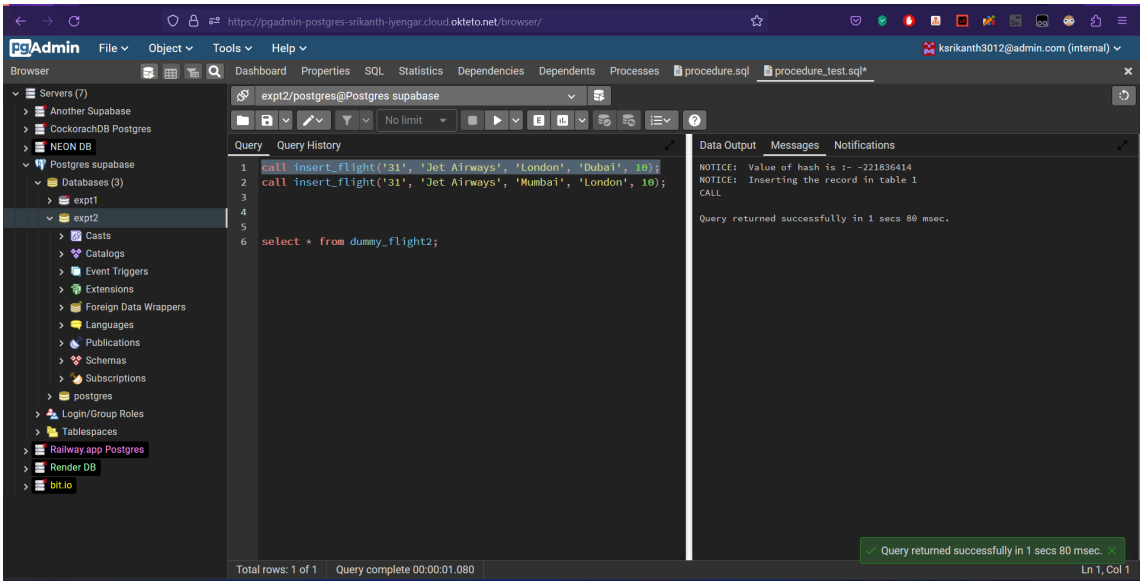
PART B

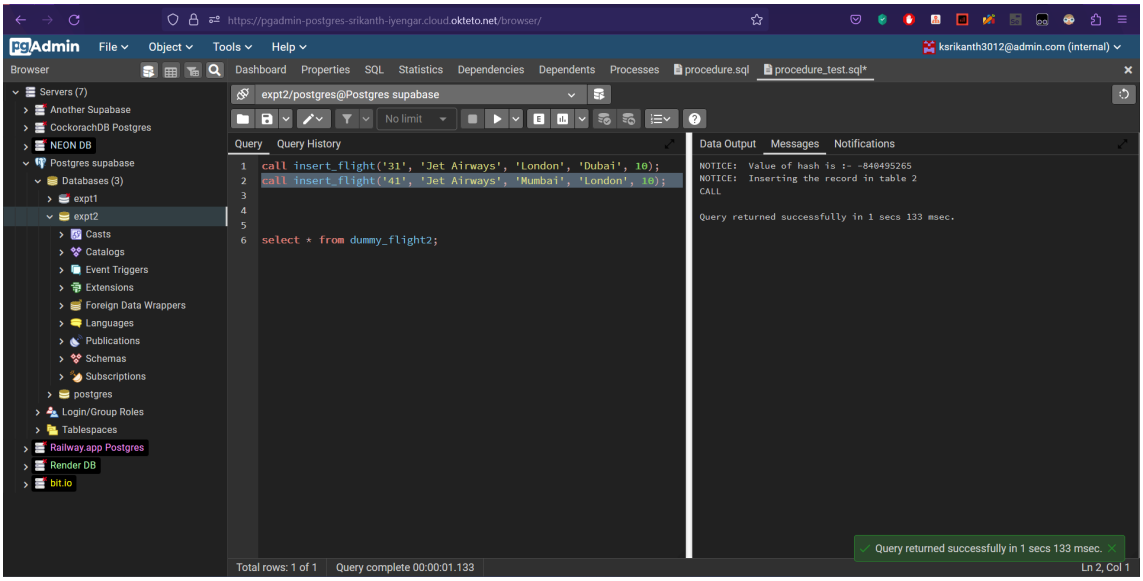
- Creating a procedure which will insert the table if the hash value if even or odd

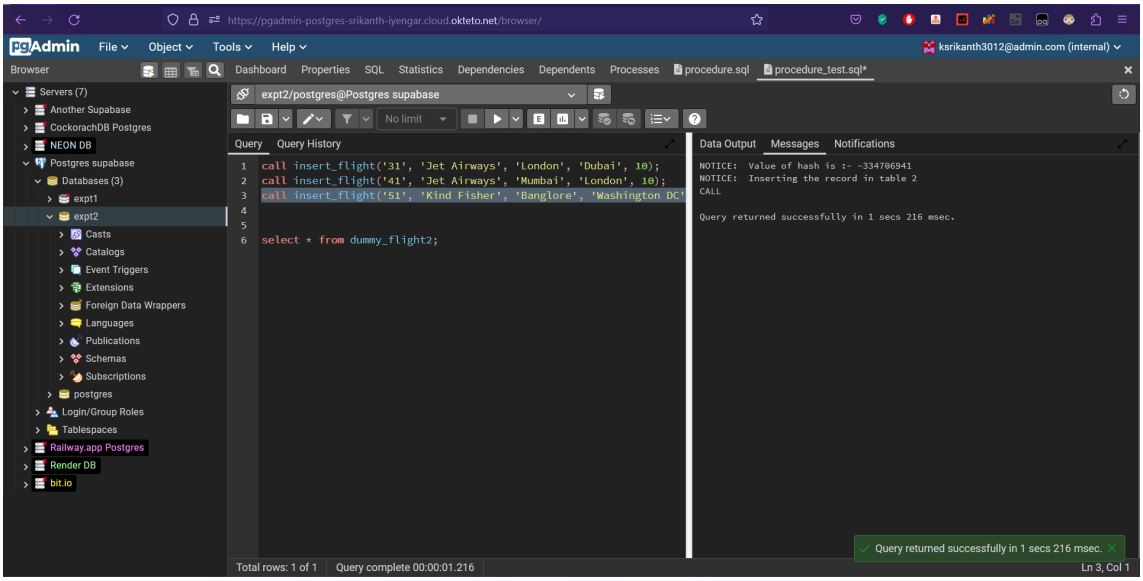
```
CREATE OR REPLACE PROCEDURE insert_flight (
    flight_id integer,
    flight_name varchar(100),
    flight_source varchar(100),
    flight_destination varchar(100),
    tickets_available integer
)
LANGUAGE plpgsql AS $$
DECLARE
```

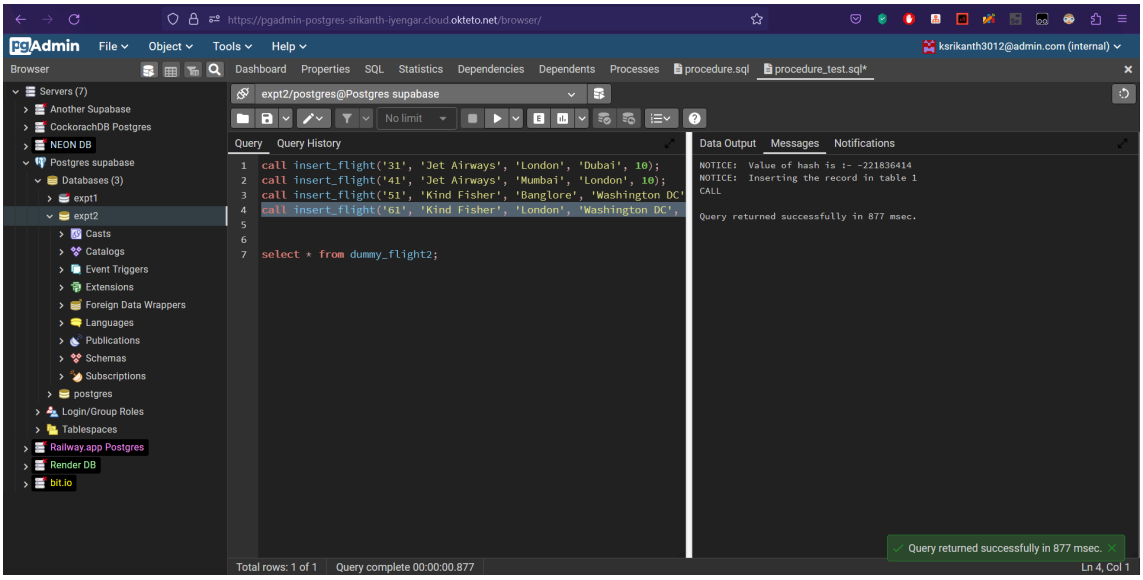
```
    hash int;
BEGIN
    hash := hashtext(flight_source);
    RAISE NOTICE 'Value of hash is :- %', hash;
    IF MOD(ABS(hash), 2) = 0 THEN
        RAISE NOTICE 'Inserting the record in table 1';
        INSERT INTO dummy_flight1 VALUES(flight_id, flight_name, flight_source,
flight_destination, tickets_available);
    END IF;
    IF MOD(ABS(hash), 2) = 1 THEN
        RAISE NOTICE 'Inserting the record in table 2';
        INSERT INTO dummy_flight2 VALUES(flight_id, flight_name, flight_source,
flight_destination, tickets_available);
    END IF;
END;$$
```

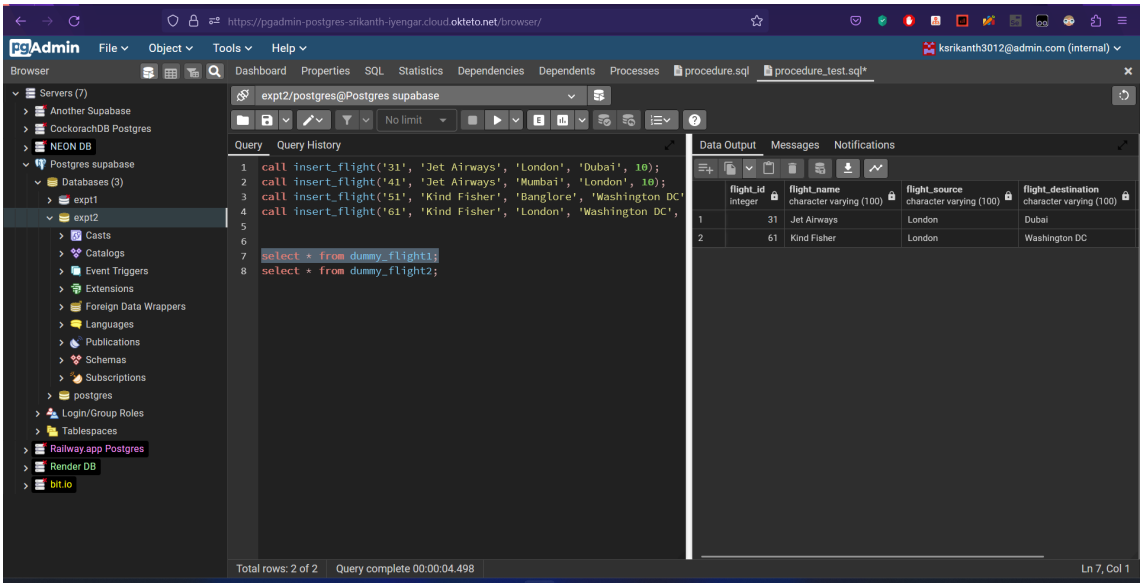
Inserting 4 records and it gets automatically added to the respective table

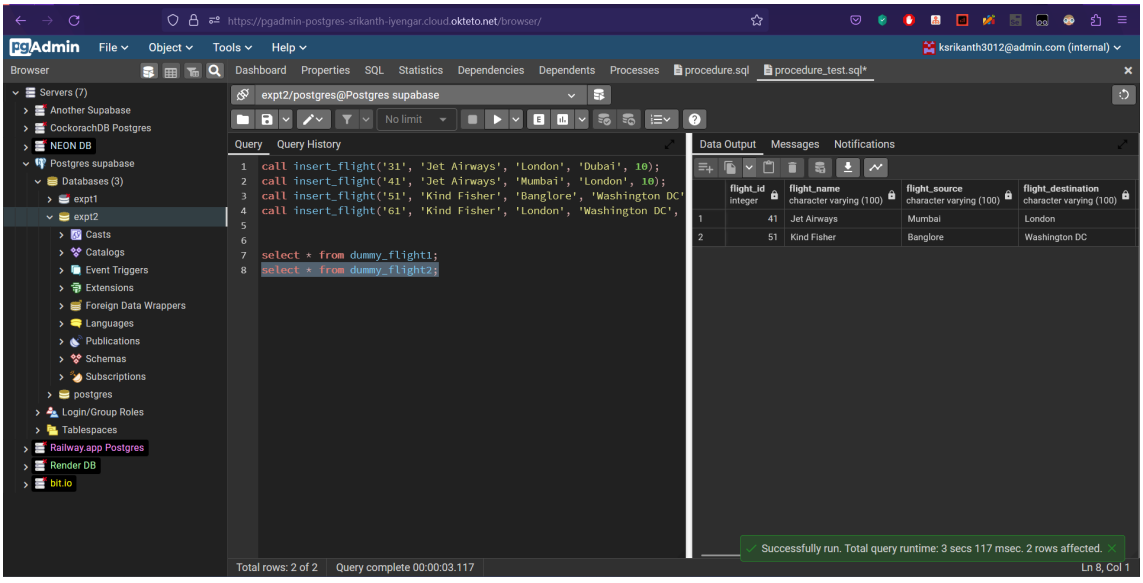

https://pgadmin-postgres-srikanth-iyengar.cloud.okteto.net/browser/
PgAdmin
File
Object
Tools
Help
ksrikanth3012@admin.com (internal)
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
procedure.sql
procedure_test.sql*
Servers (7)
Another Supabase
CockorachDB Postgres
NEON DB
Postgres supabase
Databases (3)
expt1
expt2
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas
Subscriptions
postgres
Login/Group Roles
Tablespaces
Railway.app Postgres
Render DB
bit.io
expt2/postgres@Postgres supabase
No limit
Query
Query History
1 call insert_flight('31', 'Jet Airways', 'London', 'Dubai', 10);
2 call insert_flight('31', 'Jet Airways', 'Mumbai', 'London', 10);
3
4
5
6 select * from dummy_flight2;
Data Output
Messages
Notifications
NOTICE: Value of hash is :- -221836414
NOTICE: Inserting the record in table 1
CALL
Query returned successfully in 1 secs 80 msec.
Query returned successfully in 1 secs 80 msec.
Total rows: 1 of 1
Query complete 00:00:01.080
Ln 1, Col 1


https://pgadmin-postgres-srikanth-iyengar.cloud.okteto.net/browser/
PgAdmin
File
Object
Tools
Help
ksrikanth3012@admin.com (internal)
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
procedure.sql
procedure_test.sql*
Servers (7)
Another Supabase
CockorachDB Postgres
NEON DB
Postgres supabase
Databases (3)
expt1
expt2
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas
Subscriptions
postgres
Login/Group Roles
Tablespaces
Railway.app Postgres
Render DB
bit.io
expt2/postgres@Postgres supabase
No limit
Query
Query History
1 call insert_flight('31', 'Jet Airways', 'London', 'Dubai', 10);
2 call insert_flight('41', 'Jet Airways', 'Mumbai', 'London', 10);
3
4
5
6 select * from dummy_flight2;
Data Output
Messages
Notifications
NOTICE: Value of hash is :- -840495265
NOTICE: Inserting the record in table 2
CALL
Query returned successfully in 1 secs 133 msec.
Query returned successfully in 1 secs 133 msec.
Total rows: 1 of 1
Query complete 00:00:01.133
Ln 2, Col 1


https://pgadmin-postgres-srikanth-iyengar.cloud.okteto.net/browser/
pgAdmin
File
Object
Tools
Help
ksrikanth3012@admin.com (internal)
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
procedure.sql
procedure_test.sql*
Servers (7)
Another Supabase
CockorachDB Postgres
NEON DB
Postgres supabase
Databases (3)
expt1
expt2
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas
Subscriptions
postgres
Login/Group Roles
Tablespaces
Railway.app Postgres
Render DB
bit.io
expt2/postgres@Postgres supabase
No limit
Query
Query History
1 call insert_flight('31', 'Jet Airways', 'London', 'Dubai', 10);
2 call insert_flight('41', 'Jet Airways', 'Mumbai', 'London', 10);
3 call insert_flight('51', 'Kind Fisher', 'Banglore', 'Washington DC'
4
5
6 select * from dummy_flight2;
Data Output
Messages
Notifications
NOTICE: Value of hash is :- -334706941
NOTICE: Inserting the record in table 2
CALL
Query returned successfully in 1 secs 216 msec.
Query returned successfully in 1 secs 216 msec.
Total rows: 1 of 1
Query complete 00:00:01.216
Ln 3, Col 1


https://pgadmin-postgres-srikanth-iyengar.cloud.okteto.net/browser/
pgAdmin
File
Object
Tools
Help
ksrikanth3012@admin.com (internal)
Browser
Dashboard
Properties
SQL
Statistics
Dependencies
Dependents
Processes
procedure.sql
procedure_test.sql*
Servers (7)
Another Supabase
CockorachDB Postgres
NEON DB
Postgres supabase
Databases (3)
expt1
expt2
Casts
Catalogs
Event Triggers
Extensions
Foreign Data Wrappers
Languages
Publications
Schemas
Subscriptions
postgres
Login/Group Roles
Tablespaces
Railway.app Postgres
Render DB
bit.io
expt2/postgres@Postgres supabase
No limit
Query
Query History
1 call insert_flight('31', 'Jet Airways', 'London', 'Dubai', 10);
2 call insert_flight('41', 'Jet Airways', 'Mumbai', 'London', 10);
3 call insert_flight('51', 'Kind Fisher', 'Banglore', 'Washington DC'
4 call insert_flight('61', 'Kind Fisher', 'London', 'Washington DC',
5
6
7 select * from dummy_flight2;
Data Output
Messages
Notifications
NOTICE: Value of hash is :- -221836414
NOTICE: Inserting the record in table 1
CALL
Query returned successfully in 877 msec.
Query returned successfully in 877 msec.
Total rows: 1 of 1
Query complete 00:00:00.877
Ln 4, Col 1





Conclusion

Learned how to design a distributed database by applying the concept of horizontal fragmentation in PostgreSQL.