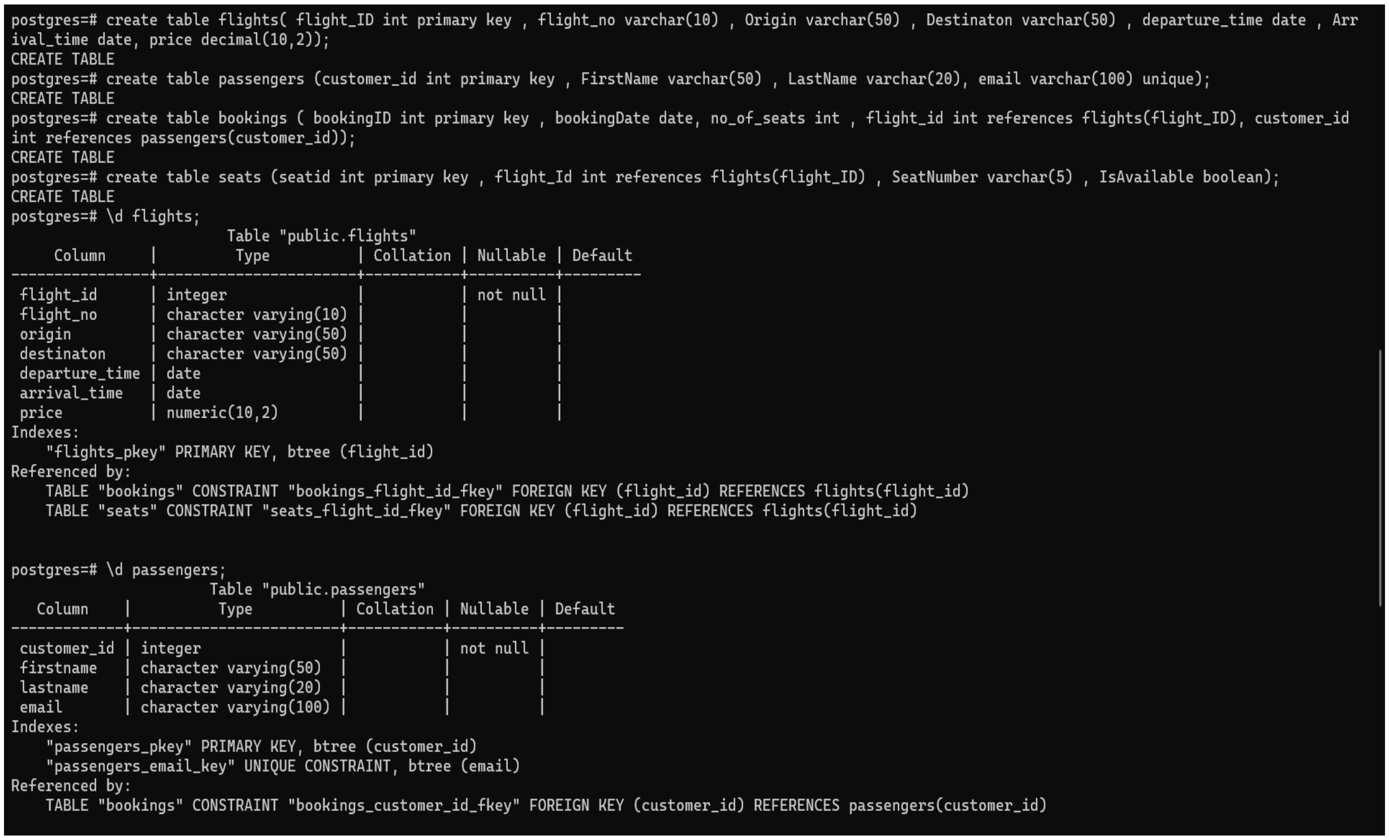
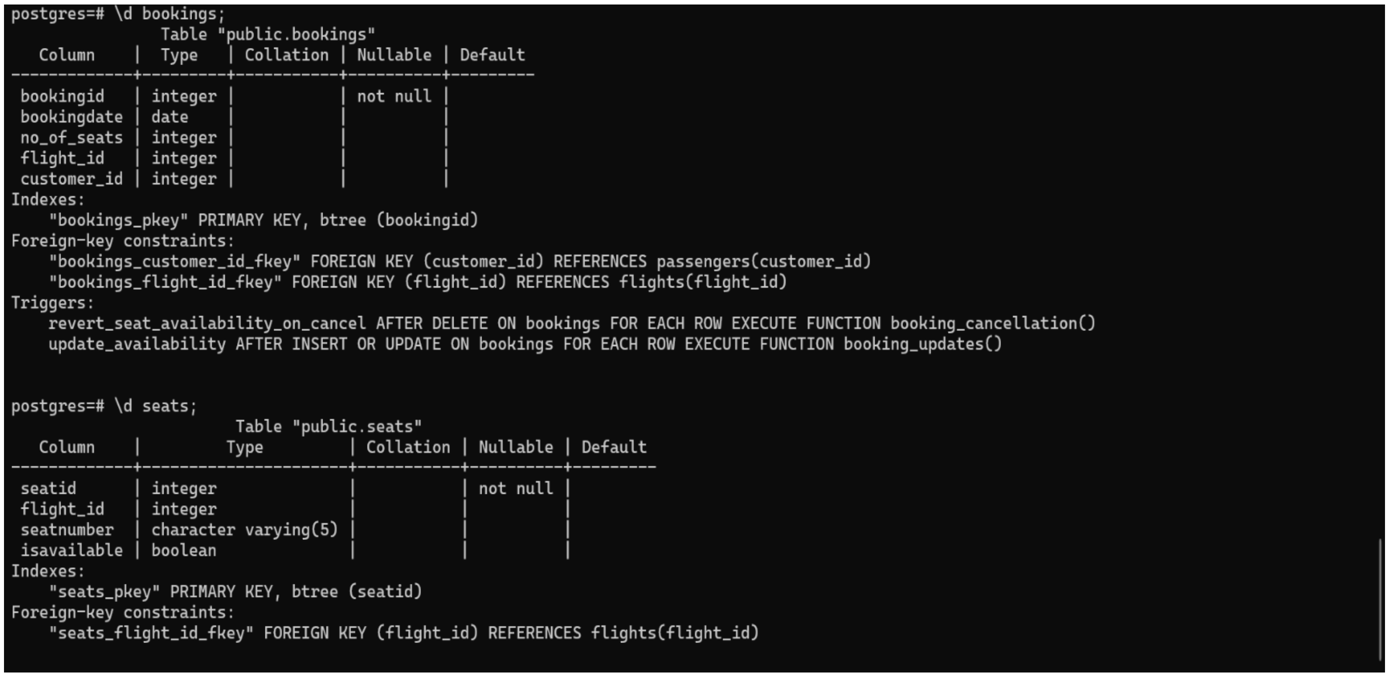
# AIRLINE RESERVATION SYSTEM

## 1)Design schema: Flights, Passengers, Bookings, Seats

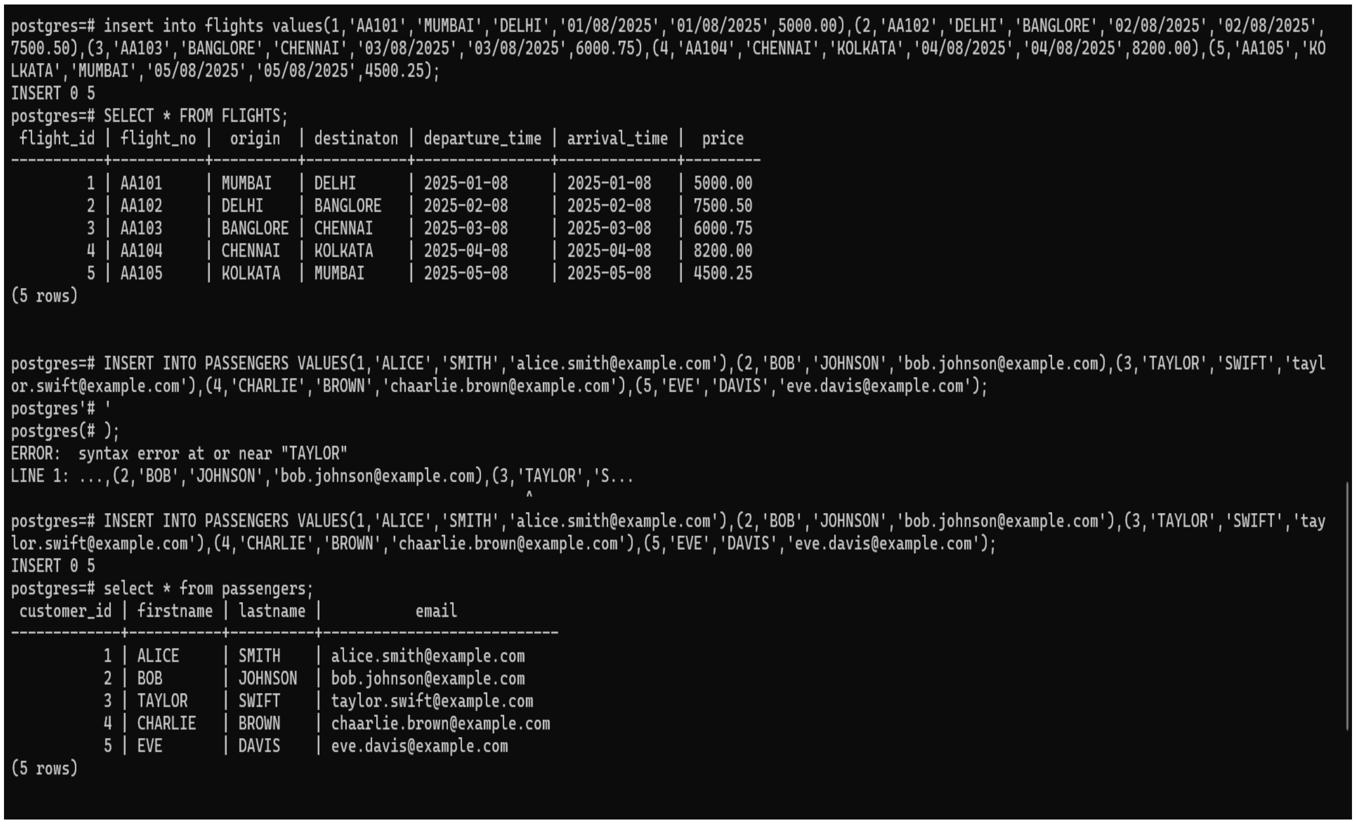


```
postgres=# create table flights( flight_ID int primary key , flight_no varchar(10) , Origin varchar(50) , Destinaton varchar(50) , departure_time date , Arr
ival_time date, price decimal(10,2));
CREATE TABLE
postgres=# create table passengers (customer_id int primary key , FirstName varchar(50) , LastName varchar(20), email varchar(100) unique);
CREATE TABLE
postgres=# create table bookings ( bookingID int primary key , bookingDate date, no_of_seats int , flight_id int references flights(flight_ID), customer_id
int references passengers(customer_id));
CREATE TABLE
postgres=# create table seats (seatid int primary key , flight_Id int references flights(flight_ID) , SeatNumber varchar(5) , IsAvailable boolean);
CREATE TABLE
postgres=# \d flights;
                        Table "public.flights"
     Column     |         Type          | Collation | Nullable | Default
----------------+-----------------------+-----------+----------+---------
 flight_id      | integer               |           | not null |
 flight_no      | character varying(10) |           |          |
 origin         | character varying(50) |           |          |
 destinaton     | character varying(50) |           |          |
 departure_time | date                  |           |          |
 arrival_time   | date                  |           |          |
 price          | numeric(10,2)         |           |          |
Indexes:
    "flights_pkey" PRIMARY KEY, btree (flight_id)
Referenced by:
    TABLE "bookings" CONSTRAINT "bookings_flight_id_fkey" FOREIGN KEY (flight_id) REFERENCES flights(flight_id)
    TABLE "seats" CONSTRAINT "seats_flight_id_fkey" FOREIGN KEY (flight_id) REFERENCES flights(flight_id)


postgres=# \d passengers;
                      Table "public.passengers"
   Column    |          Type          | Collation | Nullable | Default
-------------+------------------------+-----------+----------+---------
 customer_id | integer                |           | not null |
 firstname   | character varying(50)  |           |          |
 lastname    | character varying(20)  |           |          |
 email       | character varying(100) |           |          |
Indexes:
    "passengers_pkey" PRIMARY KEY, btree (customer_id)
    "passengers_email_key" UNIQUE CONSTRAINT, btree (email)
Referenced by:
    TABLE "bookings" CONSTRAINT "bookings_customer_id_fkey" FOREIGN KEY (customer_id) REFERENCES passengers(customer_id)
```

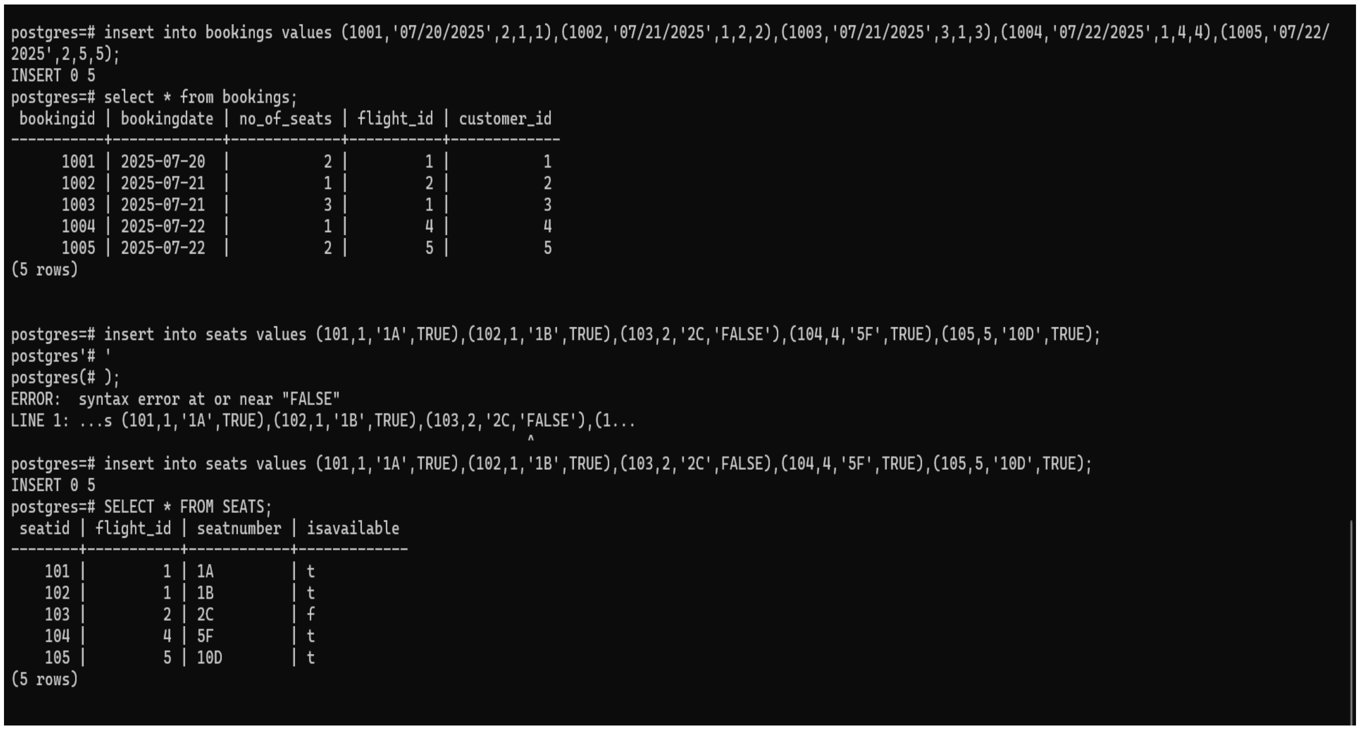


```
postgres=# \d bookings;
               Table "public.bookings"
   Column    |  Type   | Collation | Nullable | Default
-------------+---------+-----------+----------+---------
 bookingid   | integer |           | not null |
 bookingdate | date    |           |          |
 no_of_seats | integer |           |          |
 flight_id   | integer |           |          |
 customer_id | integer |           |          |
Indexes:
    "bookings_pkey" PRIMARY KEY, btree (bookingid)
Foreign-key constraints:
    "bookings_customer_id_fkey" FOREIGN KEY (customer_id) REFERENCES passengers(customer_id)
    "bookings_flight_id_fkey" FOREIGN KEY (flight_id) REFERENCES flights(flight_id)
Triggers:
    revert_seat_availability_on_cancel AFTER DELETE ON bookings FOR EACH ROW EXECUTE FUNCTION booking_cancellation()
    update_availability AFTER INSERT OR UPDATE ON bookings FOR EACH ROW EXECUTE FUNCTION booking_updates()


postgres=# \d seats;
                      Table "public.seats"
   Column    |         Type         | Collation | Nullable | Default
-------------+----------------------+-----------+----------+---------
 seatid      | integer              |           | not null |
 flight_id   | integer              |           |          |
 seatnumber  | character varying(5) |           |          |
 isavailable | boolean              |           |          |
Indexes:
    "seats_pkey" PRIMARY KEY, btree (seatid)
Foreign-key constraints:
    "seats_flight_id_fkey" FOREIGN KEY (flight_id) REFERENCES flights(flight_id)
```

## 2)Insert sample flight and booking records

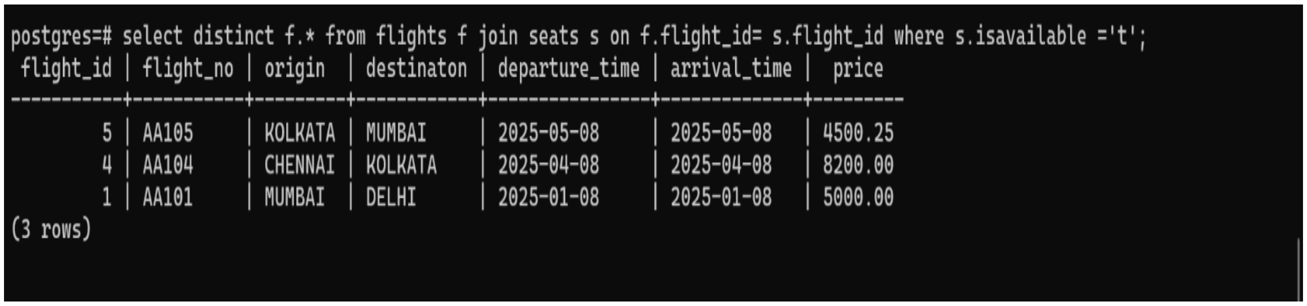


```
postgres=# insert into flights values(1,'AA101','MUMBAI','DELHI','01/08/2025','01/08/2025',5000.00),(2,'AA102','DELHI','BANGLORE','02/08/2025','02/08/2025',
7500.50),(3,'AA103','BANGLORE','CHENNAI','03/08/2025','03/08/2025',6000.75),(4,'AA104','CHENNAI','KOLKATA','04/08/2025','04/08/2025',8200.00),(5,'AA105','KO
LKATA','MUMBAI','05/08/2025','05/08/2025',4500.25);
INSERT 0 5
postgres=# SELECT * FROM FLIGHTS;
 flight_id | flight_no |  origin  | destinaton | departure_time | arrival_time |  price
-----------+-----------+----------+------------+----------------+--------------+---------
         1 | AA101     | MUMBAI   | DELHI      | 2025-01-08     | 2025-01-08   | 5000.00
         2 | AA102     | DELHI    | BANGLORE   | 2025-02-08     | 2025-02-08   | 7500.50
         3 | AA103     | BANGLORE | CHENNAI    | 2025-03-08     | 2025-03-08   | 6000.75
         4 | AA104     | CHENNAI  | KOLKATA    | 2025-04-08     | 2025-04-08   | 8200.00
         5 | AA105     | KOLKATA  | MUMBAI     | 2025-05-08     | 2025-05-08   | 4500.25
(5 rows)


postgres=# INSERT INTO PASSENGERS VALUES(1,'ALICE','SMITH','alice.smith@example.com'),(2,'BOB','JOHNSON','bob.johnson@example.com),(3,'TAYLOR','SWIFT','tayl
or.swift@example.com'),(4,'CHARLIE','BROWN','chaarlie.brown@example.com'),(5,'EVE','DAVIS','eve.davis@example.com');
postgres'# '
postgres(# );
ERROR:  syntax error at or near "TAYLOR"
LINE 1: ...,(2,'BOB','JOHNSON','bob.johnson@example.com),(3,'TAYLOR','S...
                                                            ^
postgres=# INSERT INTO PASSENGERS VALUES(1,'ALICE','SMITH','alice.smith@example.com'),(2,'BOB','JOHNSON','bob.johnson@example.com'),(3,'TAYLOR','SWIFT','tay
lor.swift@example.com'),(4,'CHARLIE','BROWN','chaarlie.brown@example.com'),(5,'EVE','DAVIS','eve.davis@example.com');
INSERT 0 5
postgres=# select * from passengers;
 customer_id | firstname | lastname |           email
-------------+-----------+----------+----------------------------
           1 | ALICE     | SMITH    | alice.smith@example.com
           2 | BOB       | JOHNSON  | bob.johnson@example.com
           3 | TAYLOR    | SWIFT    | taylor.swift@example.com
           4 | CHARLIE   | BROWN    | chaarlie.brown@example.com
           5 | EVE       | DAVIS    | eve.davis@example.com
(5 rows)
```

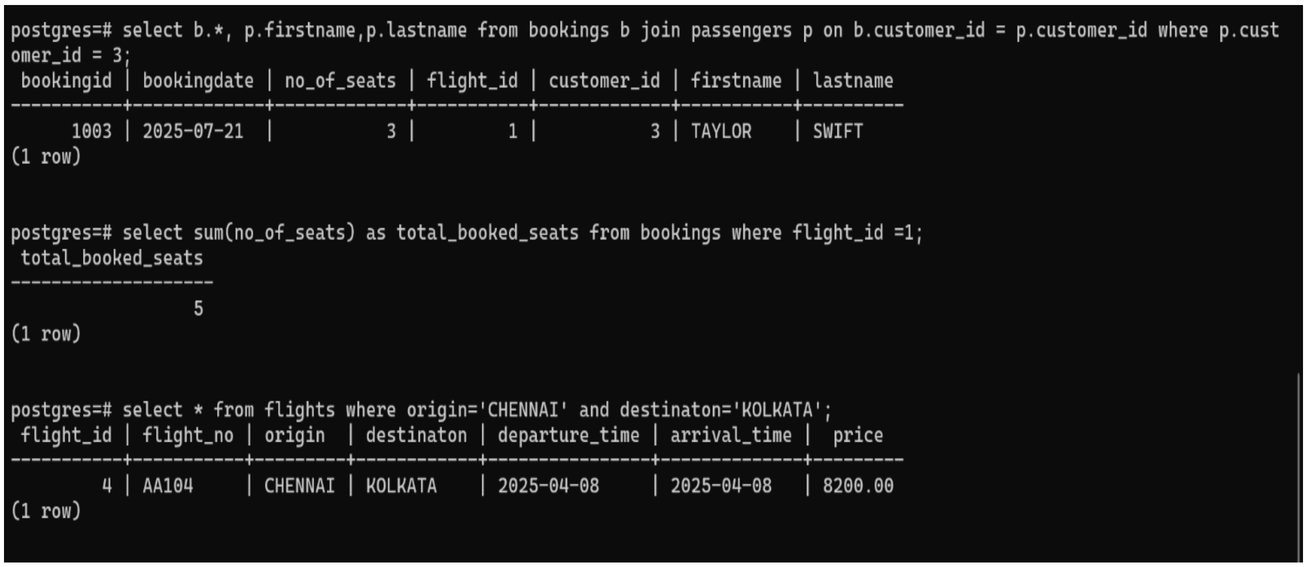


```
postgres=# insert into bookings values (1001,'07/20/2025',2,1,1),(1002,'07/21/2025',1,2,2),(1003,'07/21/2025',3,1,3),(1004,'07/22/2025',1,4,4),(1005,'07/22/
2025',2,5,5);
INSERT 0 5
postgres=# select * from bookings;
 bookingid | bookingdate | no_of_seats | flight_id | customer_id
-----------+-------------+-------------+-----------+-------------
      1001 | 2025-07-20  |           2 |         1 |           1
      1002 | 2025-07-21  |           1 |         2 |           2
      1003 | 2025-07-21  |           3 |         1 |           3
      1004 | 2025-07-22  |           1 |         4 |           4
      1005 | 2025-07-22  |           2 |         5 |           5
(5 rows)


postgres=# insert into seats values (101,1,'1A',TRUE),(102,1,'1B',TRUE),(103,2,'2C,'FALSE'),(104,4,'5F',TRUE),(105,5,'10D',TRUE);
postgres'# '
postgres(# );
ERROR:  syntax error at or near "FALSE"
LINE 1: ...s (101,1,'1A',TRUE),(102,1,'1B',TRUE),(103,2,'2C,'FALSE'),(1...
                                                            ^
postgres=# insert into seats values (101,1,'1A',TRUE),(102,1,'1B',TRUE),(103,2,'2C',FALSE),(104,4,'5F',TRUE),(105,5,'10D',TRUE);
INSERT 0 5
postgres=# SELECT * FROM SEATS;
 seatid | flight_id | seatnumber | isavailable
--------+-----------+------------+-------------
    101 |         1 | 1A         | t
    102 |         1 | 1B         | t
    103 |         2 | 2C         | f
    104 |         4 | 5F         | t
    105 |         5 | 10D        | t
(5 rows)
```

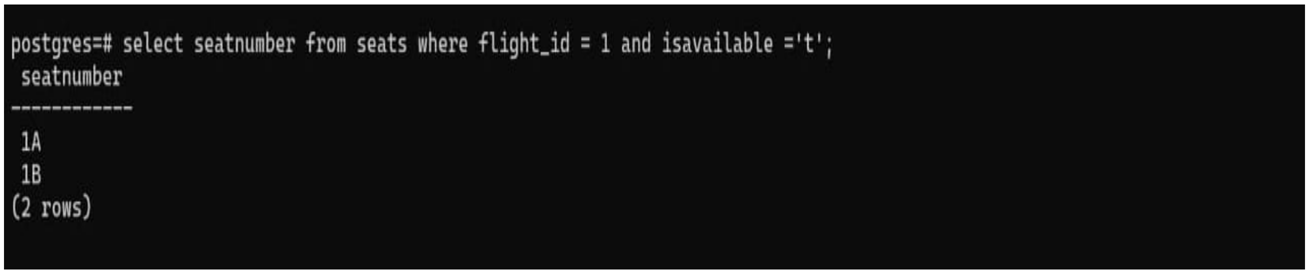
3)Write queries for available seats, flight search.



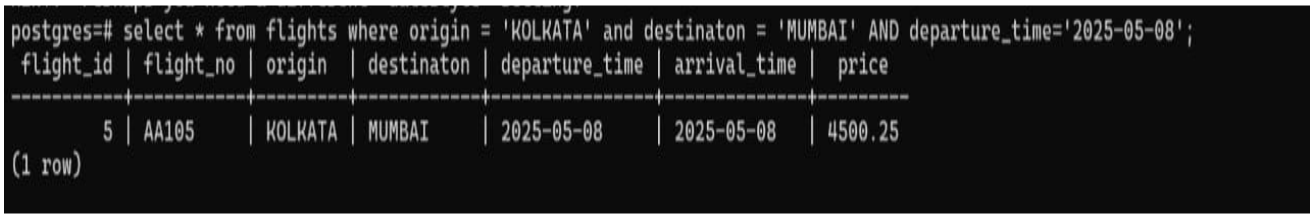
```
postgres=# select distinct f.* from flights f join seats s on f.flight_id= s.flight_id where s.isavailable ='t';
 flight_id | flight_no | origin  | destinaton | departure_time | arrival_time |  price
-----------+-----------+---------+------------+----------------+--------------+---------
         5 | AA105     | KOLKATA | MUMBAI     | 2025-05-08     | 2025-05-08   | 4500.25
         4 | AA104     | CHENNAI | KOLKATA    | 2025-04-08     | 2025-04-08   | 8200.00
         1 | AA101     | MUMBAI  | DELHI      | 2025-01-08     | 2025-01-08   | 5000.00
(3 rows)
```



```
postgres=# select b.*, p.firstname,p.lastname from bookings b join passengers p on b.customer_id = p.customer_id where p.cust
omer_id = 3;
 bookingid | bookingdate | no_of_seats | flight_id | customer_id | firstname | lastname
-----------+-------------+-------------+-----------+-------------+-----------+----------
      1003 | 2025-07-21  |           3 |         1 |           3 | TAYLOR    | SWIFT
(1 row)


postgres=# select sum(no_of_seats) as total_booked_seats from bookings where flight_id =1;
 total_booked_seats
--------------------
                  5
(1 row)


postgres=# select * from flights where origin='CHENNAI' and destinaton='KOLKATA';
 flight_id | flight_no | origin  | destinaton | departure_time | arrival_time |  price
-----------+-----------+---------+------------+----------------+--------------+---------
         4 | AA104     | CHENNAI | KOLKATA    | 2025-04-08     | 2025-04-08   | 8200.00
(1 row)
```



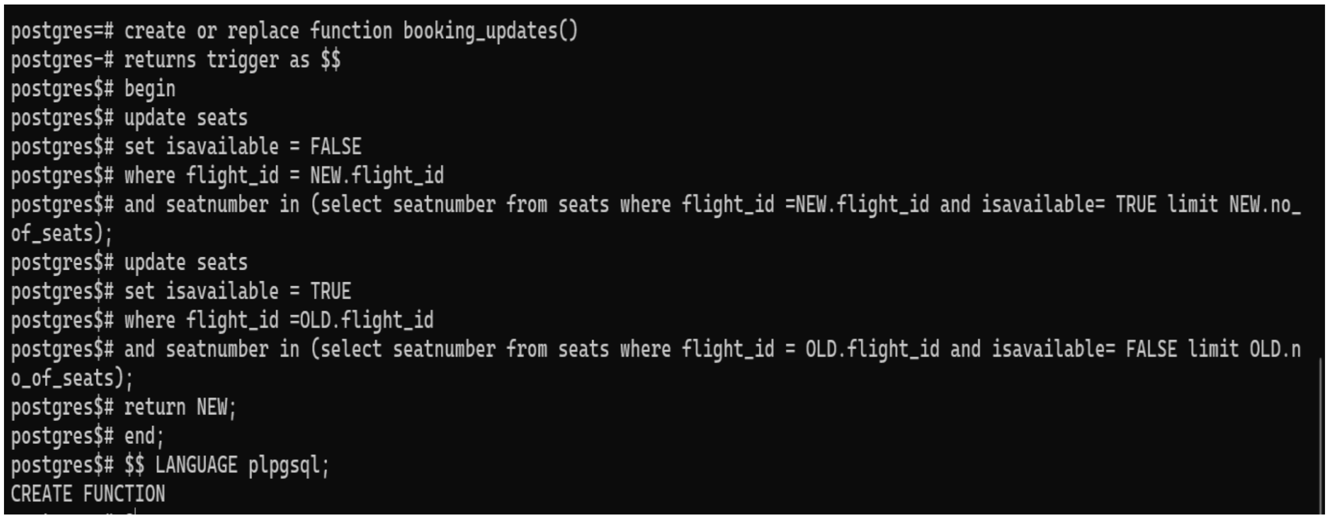
```
postgres=# select seatnumber from seats where flight_id = 1 and isavailable ='t';
 seatnumber
------------
 1A
 1B
(2 rows)
```



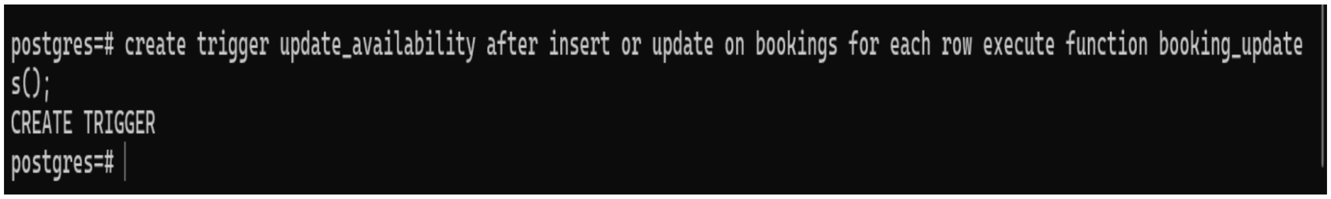
```
postgres=# select * from flights where origin = 'KOLKATA' and destinaton = 'MUMBAI' AND departure_time='2025-05-08';
 flight_id | flight_no | origin  | destinaton | departure_time | arrival_time |  price
-----------+-----------+---------+------------+----------------+--------------+---------
         5 | AA105     | KOLKATA | MUMBAI     | 2025-05-08     | 2025-05-08   | 4500.25
(1 row)
```

4)Write triggers for booking updates and cancellations.

- Booking updates

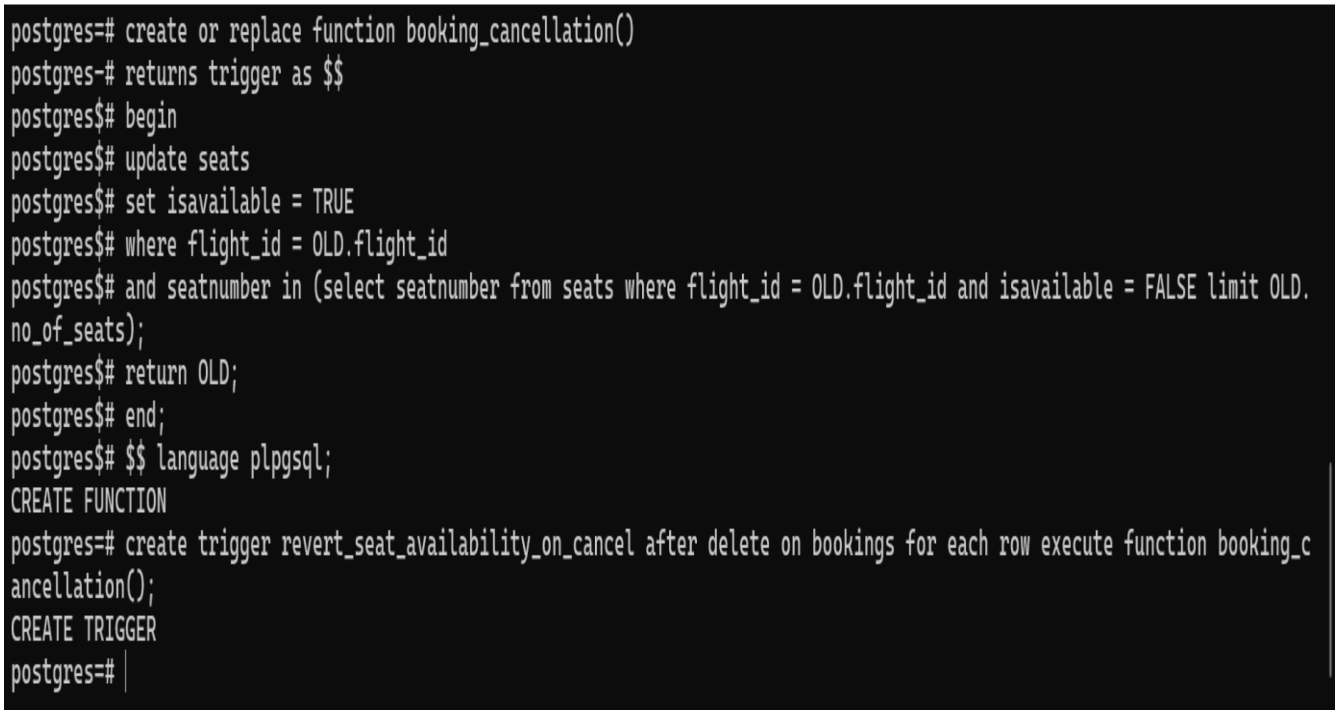


```
postgres=# create or replace function booking_updates()
postgres-# returns trigger as $$
postgres$# begin
postgres$# update seats
postgres$# set isavailable = FALSE
postgres$# where flight_id = NEW.flight_id
postgres$# and seatnumber in (select seatnumber from seats where flight_id =NEW.flight_id and isavailable= TRUE limit NEW.no_
of_seats);
postgres$# update seats
postgres$# set isavailable = TRUE
postgres$# where flight_id =OLD.flight_id
postgres$# and seatnumber in (select seatnumber from seats where flight_id = OLD.flight_id and isavailable= FALSE limit OLD.n
o_of_seats);
postgres$# return NEW;
postgres$# end;
postgres$# $$ LANGUAGE plpgsql;
CREATE FUNCTION
```



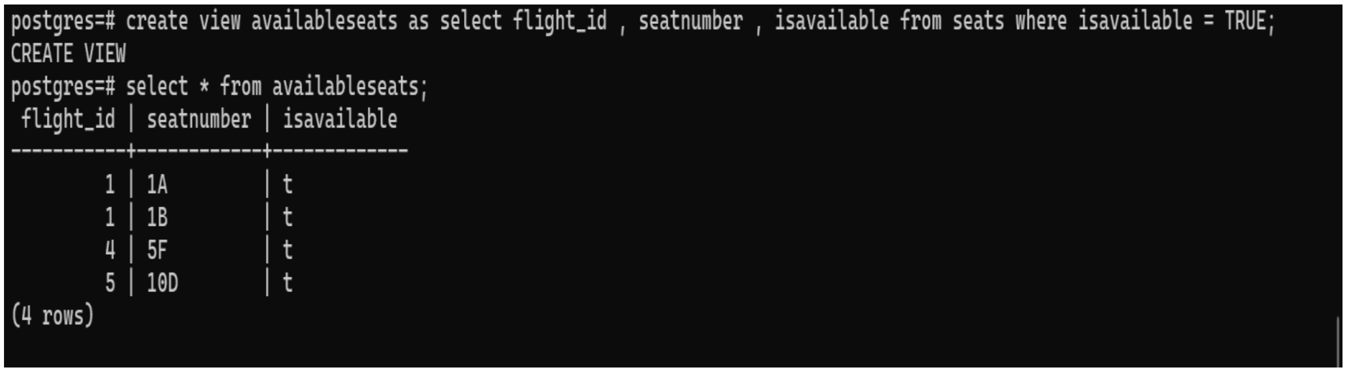
```
postgres=# create trigger update_availability after insert or update on bookings for each row execute function booking_update
s();
CREATE TRIGGER
postgres=#
```

- Booking cancellation

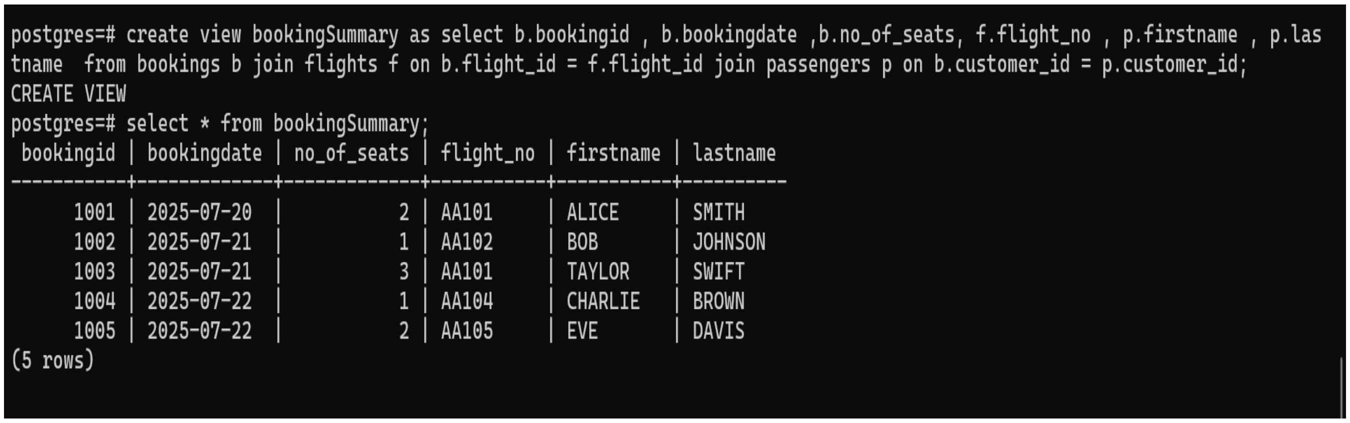


```
postgres=# create or replace function booking_cancellation()
postgres-# returns trigger as $$
postgres$# begin
postgres$# update seats
postgres$# set isavailable = TRUE
postgres$# where flight_id = OLD.flight_id
postgres$# and seatnumber in (select seatnumber from seats where flight_id = OLD.flight_id and isavailable = FALSE limit OLD.
no_of_seats);
postgres$# return OLD;
postgres$# end;
postgres$# $$ language plpgsql;
CREATE FUNCTION
postgres=# create trigger revert_seat_availability_on_cancel after delete on bookings for each row execute function booking_c
ancellation();
CREATE TRIGGER
postgres=#
```

## 5)Creating Views



```
postgres=# create view availableseats as select flight_id , seatnumber , isavailable from seats where isavailable = TRUE;
CREATE VIEW
postgres=# select * from availableseats;
 flight_id | seatnumber | isavailable
-----------+------------+-------------
         1 | 1A         | t
         1 | 1B         | t
         4 | 5F         | t
         5 | 10D        | t
(4 rows)
```



```
postgres=# create view bookingSummary as select b.bookingid , b.bookingdate ,b.no_of_seats, f.flight_no , p.firstname , p.las
tname  from bookings b join flights f on b.flight_id = f.flight_id join passengers p on b.customer_id = p.customer_id;
CREATE VIEW
postgres=# select * from bookingSummary;
 bookingid | bookingdate | no_of_seats | flight_no | firstname | lastname
-----------+-------------+-------------+-----------+-----------+----------
      1001 | 2025-07-20  |           2 | AA101     | ALICE     | SMITH
      1002 | 2025-07-21  |           1 | AA102     | BOB       | JOHNSON
      1003 | 2025-07-21  |           3 | AA101     | TAYLOR    | SWIFT
      1004 | 2025-07-22  |           1 | AA104     | CHARLIE   | BROWN
      1005 | 2025-07-22  |           2 | AA105     | EVE       | DAVIS
(5 rows)
```

## 6)Generate booking summary report

PostgreSQL function named booking_updates() designed to manage seat availability in a booking system. It updates the isavailable status of seats based on flight_id and seatnumber for both new and old bookings.