

1 task

The screenshot shows the pgAdmin 4 interface. In the Object Explorer, under the 'MY DB' server's 'airport_2' database, the 'Tables' folder contains ten tables: airline, airport, baggage, baggage_check, boarding_pass, booking, booking_flight, flights, and passengers. The 'flights' table is currently selected. In the main query editor window, the following SQL command is run:

```
CREATE INDEX idx_act_dep ON flights (actual_departure);
```

The status bar at the bottom indicates "Query returned successfully in 64 msec." and "Query complete 00:00:00.064". A message box in the bottom right corner states: "You are currently running version 9.6 of pgAdmin 4, however the current version is 9.9." with a link to "Please click here for more information."

2 task

The screenshot shows the pgAdmin 4 interface. The 'flights' table is selected in the Object Explorer. In the main query editor window, the following SQL command is run:

```
CREATE UNIQUE INDEX idx_fsch ON flights (flight_no, scheduled_departure);
```

The status bar at the bottom indicates "Query complete 00:00:00.066". A message box in the bottom right corner states: "You are currently running version 9.6 of pgAdmin 4, however the current version is 9.9." with a link to "Please click here for more information.". Below the message box, an error message is displayed:

```
ERROR: создать уникальный индекс "idx_fsch" не удалось
Ключ (flight_no, scheduled_departure)=(US-KS, 2023-09-04) дублируется.

ОШИБКА: создать уникальный индекс "idx_fsch" не удалось
SQL state: 23565
Detail: Ключ (flight_no, scheduled_departure)=(US-KS, 2023-09-04) дублируется.
```

3 task

The screenshot shows the pgAdmin 4 interface with the Object Explorer on the left and a query editor on the right. The Object Explorer displays various database objects like Materialized Views, Operators, Procedures, Sequences, and Tables. The Tables section shows the 'flights' table with 14 columns. The query editor contains the following SQL code:

```
--2
CREATE UNIQUE INDEX idx_fsch ON flights (flight_no, scheduled_departure)
--3
CREATE INDEX idx_dep_arr ON flights (departure_airport_id, arrival_airport_id);
```

The status bar at the bottom indicates "Query returned successfully in 60 msec." and "Total rows: 3 Query complete 00:00:00.060". A message box in the bottom right corner states: "You are currently running version 9.6 of pgAdmin 4, however the current version is 9.9." and "Please click [here](#) for more information."

4 task

The screenshot shows the pgAdmin 4 interface with the Object Explorer on the left and a query editor on the right. The Object Explorer displays various database objects like MY DB, Servers, PostgreSQL 17, and Tables. The Tables section shows the 'flights' table with 14 columns. The query editor contains the following SQL code:

```
CREATE INDEX idx_dep_arr ON flights (departure_airport_id, arrival_airport_id);
SELECT * FROM flights WHERE actual_departure = '2023-09-04';
```

The results pane shows the following data:

flight_id	flight_no	scheduled_departure	scheduled_arrival	departure_airport_id	arrival_airport_id	departing_gate	arriving_gate	airline_id	status
1	78	2023-08-28	2023-08-02	20	11	355	515	1	Delayed
2	454	2023-06-10	2023-07-18	1	7	1987	45	42	Delayed
3	920	2023-05-15	2023-06-23	16	13	59	244	5	Delayed

The status bar at the bottom indicates "Successfully run. Total query runtime: 123 msec. 3 rows affected." and "Total rows: 3 Query complete 00:00:00.123".

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer

Servers (1)

Databases (6)

Tables (10)

Query Query History

```
--3
CREATE INDEX idx_dep_arr ON flights (departure_airport_id, arrival_airport_id);

--4
CREATE INDEX idx_ac_dep ON flights(actual_departure);
SELECT * FROM flights WHERE actual_departure = '2023-09-04';

5718
5719
5720
5721
5722
5723
5724
```

Data Output Messages Notifications

flight_id [PK] integer	flight_no character varying (50)	scheduled_departure_date	scheduled_arrival_date	departure_airport_id integer	arrival_airport_id integer	departing_gate character varying (50)	arriving_gate character varying (50)	airline_id integer	status character varying	
1	78	US-VT	2023-08-28	2023-08-02	20	11	335	515	1 Delayed	
2	454	US-WV		2023-06-10	2023-07-18	1	7	1987	45	42 Delayed
3	920	KR-11		2023-05-15	2023-06-23	16	13	59	244	5 Delayed

Total rows: 3 Query complete 00:00:00.118 ✓ Successfully run. Total query runtime: 118 msec. 3 rows affected. CRLF Ln 5718, Col 1

5 task

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer

Tables (10)

Columns (14)

Constraints (4)

Indexes (2)

Rules

Triggers

passengers

security_check

Trigger Functions

Types

Views

Subscriptions

lab10

lab11

mv airport

Query Query History

```
WHERE actual_departure = '2023-09-04';

EXPLAIN ANALYZE
SELECT * FROM flights
WHERE departure_airport_id = 15
AND arrival_airport_id = 18
```

Data Output Messages Notifications

QUERY PLAN

1 Bitmap Heap Scan on flights (cost=4.30..9.97 rows=2 width=61) (actual time=0.084..0.089 rows=5 loops=1)
2 Recheck Cond: ((departure_airport_id = 15) AND (arrival_airport_id = 18))
3 Heap Blocks: exact=5
4 -> Bitmap Index Scan on idx_dep_arr (cost=0.00..4.29 rows=2 width=0) (actual time=0.073..0.073 rows=5 loops=1)
5 Index Cond: ((departure_airport_id = 15) AND (arrival_airport_id = 18))
6 Planning Time: 0.122 ms
7 Execution Time: 0.112 ms

Total rows: 7 Query complete 00:00:00.142 ✓ CRLF Ln 5724, Col 1

6 task

```

File Object Tools Edit View Window Help
Object Explorer
    FTS Templates
    Foreign Tables
    Functions
    Materialized Views
    Operators
    Procedures
    Sequences
    Tables(10)
        airline
        airport
        baggage
        baggage_check
        boarding_pass
        booking
        booking_flight
        flights
        passengers
            Columns(10)
                passenger_id
                first_name
                last_name
                date_of_birth
                gender
                country_of_citizenship
                country_of_residence
                passport_number
                created_at
                update_at
            Constraints
            Indexes
            RLS Policies
            Rules
            Triggers
            security_check
            Trigger Functions
            Types
            Views
    Combinations
Query Query History
5724 --6
CREATE UNIQUE INDEX idx_pass_no ON passengers(passport_number);
5725
5726
5727
5728
5729
5730
5731
5732
5733
5734
5735
5736
5737
5738
5739
5740
5741
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
Data Output Messages Notifications
INSERT 0 1
Query returned successfully in 78 msec.
Total rows: 1 Query complete 00:00:00.078
CRLF Ln 5742, Col 1

```

When I add a UNIQUE index to the `passport_number` column in the `Passengers` table, the database guarantees that no two rows can have the same passport number. This means every passport number must be different. After creating the index, I check whether it was successfully created. Then I insert two new rows with different passport numbers, and the operation works without any problems. However, if I try to insert rows with the same passport number, the database returns an error. This error indicates that the uniqueness constraint has been violated because the passport number already exists in the table.

7 task

```

File Object Tools Edit View Window Help
Object Explorer
    FTS Templates
    Foreign Tables
    Functions
    Materialized Views
    Operators
    Procedures
    Sequences
    Tables(10)
        airline
        airport
        baggage
        baggage_check
        boarding_pass
        booking
        booking_flight
        flights
        passengers
            Columns(10)
                passenger_id
                first_name
                last_name
                date_of_birth
                gender
                country_of_citizenship
                country_of_residence
                passport_number
                created_at
                update_at
            Constraints
            Indexes
            RLS Policies
            Rules
            Triggers
            security_check
            Trigger Functions
            Types
            Views
    Combinations
Query Query History
5741 --7
CREATE INDEX idx_info ON passengers(first_name, last_name, date_of_birth, country_of_citizenship);
5742
5743
5744
5745
5746
5747
5748
5749
5750
5751
5752
5753
5754
5755
5756
5757
5758
5759
Data Output Messages Notifications
EXPLAIN ANALYZE
SELECT first_name||' '||last_name AS full_name, date_of_birth, country_of_citizenship
FROM passengers
WHERE country_of_citizenship = 'Philippines'
    AND date_of_birth >= '1984-01-01'
    AND date_of_birth < '1985-01-01';
Showing rows: 1 to 5 Page No: 1 of 1 < << > >> <<
Total rows: 5 Query complete 00:00:00.066
CRLF Ln 5750, Col 38

```

First, we created a composite index on the columns `country_of_citizenship`, `date_of_birth`, `first_name`, and `last_name`. These columns were chosen because the query filters by `country_of_citizenship` and `date_of_birth`, so placing them first in the index improves performance. The SQL query searches for passengers who are citizens of the Philippines and were born in 1984. To allow PostgreSQL to use the index efficiently, the date condition was written as a range:

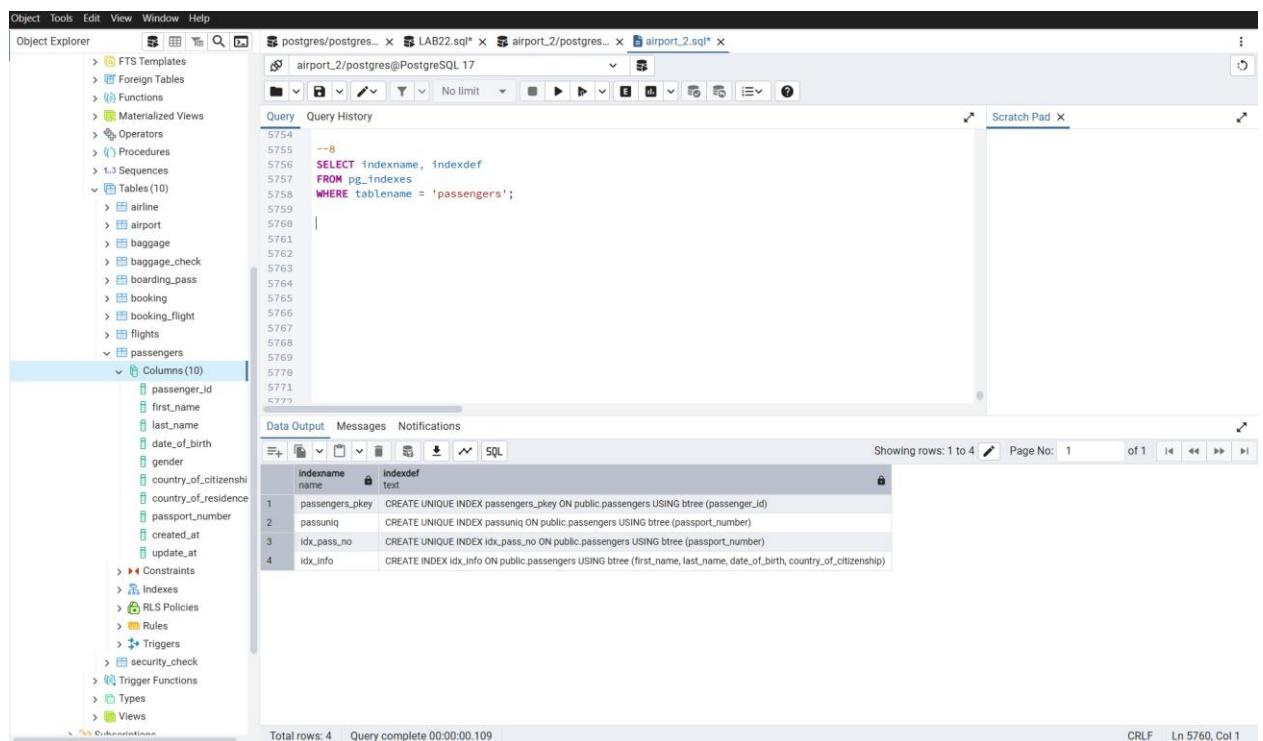
```
date_of_birth >= '1984-01-01'
```

```
AND date_of_birth < '1985-01-01'
```

When running the query with `EXPLAIN ANALYZE`, PostgreSQL performs an **Index Scan**, which confirms that the newly created index is actually used. If the date had been written using a function such as `EXTRACT(YEAR FROM date_of_birth)`, PostgreSQL would instead perform a **Sequential Scan**, because functions applied to indexed columns prevent index usage.

Therefore, the query with the date range and the composite index provides the most efficient execution plan.

8 task



The screenshot shows the pgAdmin 4 interface. The left pane is the Object Explorer, listing various database objects like FTS Templates, Foreign Tables, Functions, Materialized Views, Operators, Procedures, Sequences, and Tables (10). The 'passenger' table is selected, and its 'Columns (10)' are listed: passenger_id, first_name, last_name, date_of_birth, gender, country_of_citizenship, country_of_residence, passport_number, created_at, and update_at. The right pane contains a query editor with the following content:

```
--8
SELECT indexname, indexdef
FROM pg_indexes
WHERE tablename = 'passengers';
```

The results of the query are displayed in a table:

indexname	indexdef
passenger_pkey	CREATE UNIQUE INDEX passengers_pkey ON public.passengers USING btree (passenger_id)
passuniq	CREATE UNIQUE INDEX passuniq ON public.passengers USING btree (passport_number)
idx_pass_no	CREATE UNIQUE INDEX idx_pass_no ON public.passengers USING btree (passport_number)
idx_info	CREATE INDEX idx_info ON public.passengers USING btree (first_name, last_name, date_of_birth, country_of_citizenship)

Total rows: 4 Query complete 00:00:00.109 CRLF Ln 5760, Col 1

pgAdmin 4

File Object Tools Edit View Window Help

Object Explorer

FTS Templates
Foreign Tables
Materialized Views
Operators
Procedures
Sequences
Tables (10)
airline
airport
baggage
baggage_check
boarding_pass
booking
booking_flight
flights
passengers
Columns (10)
passenger_id
first_name
last_name
date_of_birth
gender
country_of_citizenship
country_of_residence
passport_number
created_at
update_at
Constraints
Indexes
RLS Policies
Rules
Triggers
security_check
Trigger Functions
Types
Views

Query History

```
--8
SELECT indexname, indexdef
FROM pg_indexes
WHERE tablename = 'passengers';
DROP INDEX idx_info, idx_pass_no;
```

Scratch Pad

5754
5755
5756
5757
5758
5759
5760
5761
5762
5763
5764
5765
5766
5767
5768
5769
5770
5771
5772

Data Output Messages Notifications

DROP INDEX

Query returned successfully in 92 msec.

Total rows: Query complete 00:00:00.092 CRLF Ln 5767, Col 1