





Лабораторная работа 4

Выполнил: Немков Даниил, КМБО-05-23



Размер БД: Большой

```
-- 1. Выясните, на каких маршрутах используются самолеты
-- компании Boeing.
-- В выборке вместо кода модели должно выводиться её
-- наименование
-- (например, вместо кода 733 должно выводиться Boeing 737-300).
-- Указание: можно работать с routes, aircrafts
SELECT t2.flight_id,
       t2.departure_airport,
       t2.arrival_airport,
       t1.model->>'en' as model
FROM (
    aircrafts_data t1
  JOIN
    flights t2
  ON t1.aircraft_code = t2.aircraft_code
)
WHERE t1.model->>'en' LIKE 'Boeing%';
```

| | flight_id [PK] integer  | departure_airport character (3)  | arrival_airport character (3)  | model text  |
|---|---|--|--|---|
| 1 | 2880 | DME | KUF | Boeing 767-300 |
| 2 | 9994 | DME | MRV | Boeing 737-300 |
| 3 | 50198 | SVO | AER | Boeing 777-300 |
| 4 | 60843 | LED | KHV | Boeing 767-300 |
| 5 | 64201 | OVB | SVO | Boeing 777-300 |
| Total rows: 1000 of 20143 Query complete 00:00:00.373 Ln 3, Col 1 | | | | |

```
-- 2. Выяснить, между какими парами городов летают самолеты Boeing 777-300.
```

```
SELECT DISTINCT
    r.departure_city,
    r.arrival_city
FROM
    routes as r
JOIN
    aircrafts_data as a
ON
    r.aircraft_code = a.aircraft_code
WHERE
    a.model->>'en' = 'Boeing 777-300'
```

| | departure_city text  | arrival_city text  | |
|--------------------|--|--|-----------------------------|
| 1 | Екатеринбург | Москва | |
| 2 | Москва | Екатеринбург | |
| 3 | Москва | Новосибирск | |
| 4 | Москва | Пермь | |
| 5 | Москва | Сочи | |
| Total rows: 8 of 8 | | | Query complete 00:00:00.474 |

```
-- 3. Модифицировать прошлый запрос так,  
-- чтобы каждая пара городов была выведена только один раз.
```

```
SELECT DISTINCT
    LEAST(r.departure_city, r.arrival_city),
    GREATEST(r.departure_city, r.arrival_city)
FROM
    routes as r
JOIN
    aircrafts_data as a
```

```
ON
r.aircraft_code = a.aircraft_code
WHERE
a.model ->> 'en' = 'Boeing 777-300'
```

| | least text | greatest text |
|---|---------------|------------------|
| 1 | Екатеринбург | Москва |
| 2 | Москва | Новосибирск |
| 3 | Москва | Пермь |
| 4 | Москва | Сочи |

Total rows: 4 of 4 Query complete 00:00:00.406

```
-- 4. Выяснить, сколько рейсов выполняется из Москвы в Санкт-
Петербург.
-- Вывести: город отправления, город прибытия, количество
рейсов.
SELECT departure_city as "город отправления",
        arrival_city as "город прибытия",
        COUNT(DISTINCT routes.flight_no) as "количество рейсов"
FROM
        routes
JOIN
        flights
ON routes.aircraft_code = flights.aircraft_code
WHERE
        status = 'Departed'
        AND departure_city = 'Москва'
        AND arrival_city = 'Санкт-Петербург'
GROUP BY departure_city, arrival_city
```

| | город отправления text | город прибытия text | количество рейсов bigint |
|---|---------------------------|------------------------|-----------------------------|
| 1 | Москва | Санкт-Петербург | 12 |

```
-- 5. Вычислить минимальные и максимальные цены билетов на все
направления.
-- Вывести: город отправления, город прибытия, максимальную
стоимость билета,
-- минимальную стоимость билета.
```

```
SELECT
    r.departure_city as "город отправления",
    r.arrival_city as "город прибытия",
    MAX(t.amount) as "максимальная стоимость билета",
    MIN(t.amount) as "минимальная стоимость билета"
FROM
    ticket_flights t
JOIN
    flights f
ON(t.flight_id = f.flight_id)
JOIN
    routes r
ON(f.flight_no = r.flight_no )
GROUP BY r.departure_city, r.arrival_city
```

| | город отправления text | город прибытия text | максимальная стоимость билета numeric | минимальная стоимость билета numeric |
|--|---------------------------|------------------------|--|---|
| 1 | Череповец | Новокузнецк | 89600.00 | 29900.00 |
| 2 | Курск | Москва | 12600.00 | 4200.00 |
| 3 | Самара | Сочи | 40300.00 | 13400.00 |
| 4 | Тюмень | Урай | 9800.00 | 3300.00 |
| 5 | Челябинск | Ставрополь | 19500.00 | 17800.00 |
| 6 | Хабаровск | Анадырь | 92200.00 | 30700.00 |
| 7 | Москва | Волгоград | 27800.00 | 9000.00 |
| Total rows: 367 of 367 Query complete 00:00:16.807 Ln 81, Col 59 | | | | |

```
-- 6. Беря за основу прошлый запрос, выяснить, вывести все
направления,
-- включая те, на которые не было продано ни одного билета.
```

```
SELECT
    r.departure_city,
    r.arrival_city,
```

```

        COALESCE(MAX(t.amount)::VARCHAR, '') AS max_price,
        COALESCE(MIN(t.amount)::VARCHAR, '') AS min_price
FROM
    ticket_flights t
    RIGHT JOIN
    flights f
    ON(t.flight_id = f.flight_id)
    JOIN routes r
    ON(f.flight_no = r.flight_no )
GROUP BY
    r.departure_city,
    r.arrival_city
ORDER BY
    departure_city

```


| | departure_city text | arrival_city text | max_price character varying | min_price character varying |
|---|------------------------|----------------------|--------------------------------|--------------------------------|
| 1 | Абакан | Новосибирск | 5800.00 | 5800.00 |
| 2 | Абакан | Томск | 4900.00 | 4900.00 |
| 3 | Абакан | Кызыл | | |
| 4 | Абакан | Москва | 101000.00 | 33700.00 |
| Total rows: 516 of 516 Query complete 00:00:08.010 Ln 82, Col 1 | | | | |

```

-- 7. Написать запрос, выводящий список городов, в которые нет
рейсов из Москвы
-- а. С использованием EXISTS
SELECT
    DISTINCT a.city ->> 'ru' AS city
FROM
    airports_data a
WHERE
    NOT EXISTS (
        SELECT 1
        FROM flights f
        JOIN airports_data a
        ON f.departure_airport = a.airport_code
        WHERE a.city ->> 'ru' = 'Москва'
            AND      f.arrival_airport =

```

```
a.airport_code
);
```

| | city text |  |
|---|----------------------|---|
| 1 | Благовещенск | |
| 2 | Иваново | |
| 3 | Иркутск | |
| 4 | Калуга | |
| 5 | Когалым | |
| 6 | Комсомольск-на-Амуре | |
| 7 | Кызыл | |

Total rows: 21 of 21 Query complete 00:00:00.082


```
]]
```

```
-- б. С использованием UNION, INTERSECT или EXCEPT
SELECT DISTINCT a.city ->> 'ru' AS city
FROM airports_data a
WHERE a.airport_code NOT IN (
    SELECT f.arrival_airport
    FROM flights f
    JOIN airports_data ai ON f.departure_airport =
ai.airport_code
    WHERE ai.city ->> 'ru' = 'Москва'
)
UNION
SELECT DISTINCT a.city ->> 'ru' AS city
FROM airports_data a
WHERE a.airport_code NOT IN (
    SELECT f.departure_airport
    FROM flights f
    JOIN airports_data ai ON f.arrival_airport =
ai.airport_code
```

```

        WHERE ai.city ->> 'ru' = 'Москва'
    )
ORDER BY city

```





| | city text  | |
|----------------------|--|-----------------------------|
| 1 | Благовещенск | |
| 2 | Иваново | |
| 3 | Иркутск | |
| 4 | Калуга | |
| Total rows: 21 of 21 | | Query complete 00:00:00.078 |

--8. Написать запрос для получения перечня аэропортов в тех городах, в которых больше
-- одного аэропорта

```

SELECT
    f.count,
    f.city,
    aa.airport_code,
    aa.airport_name ->> 'ru'
FROM (
    SELECT
        COUNT(DISTINCT a.airport_code) AS count,
        a.city ->> 'ru' AS city
    FROM
        airports_data a
    GROUP BY a.city ->> 'ru'
    ORDER BY count DESC
) f
JOIN
    airports_data aa
    ON(aa.city ->>'ru' = f.city)
WHERE f.count > 1

```

| | count bigint  | city text  | airport_code [PK] character (3)  | ?column? text  |
|--|---|--|--|--|
| 1 | 3 | Москва | SVO | Шереметьево |
| 2 | 3 | Москва | VKO | Внуково |
| 3 | 3 | Москва | DME | Домодедово |
| 4 | 2 | Ульяновск | ULY | Ульяновск-Восточный |
| 5 | 2 | Ульяновск | ULV | Баратаевка |
| Total rows: 5 of 5 Query complete 00:00:00.164 Ln 154, Col 1 | | | | |

-- 9. (2 балла) При планировании новых маршрутов и оценке экономической эффективности уже существующих может потребоваться информация о том,

- какова усредненная степень заполнения самолетов на всех направлениях.
- Учитывать будем только прибывшие рейсы. Запрос для решения такой задачи:


```

WITH tickets_seats
AS (
    SELECT f.flight_id,
           f.flight_no,
           f.departure_city,
           f.arrival_city,
           f.aircraft_code,
           count( tf.ticket_no ) AS fact_passengers,
           ( SELECT count( s.seat_no )
             FROM seats s
             WHERE s.aircraft_code = f.aircraft_code
           ) AS total_seats
    FROM flights_v f
    JOIN ticket_flights tf ON f.flight_id = tf.flight_id
    WHERE f.status = 'Arrived'
    GROUP BY 1, 2, 3, 4, 5
)
SELECT ts.departure_city,
       ts.arrival_city,
       sum( ts.fact_passengers ) AS sum_pass,
       sum( ts.total_seats ) AS sum_seats,
       round( sum( ts.fact_passengers )::numeric /
              sum( ts.total_seats )::numeric, 2 ) AS frac
FROM tickets_seats ts
GROUP BY ts.departure_city, ts.arrival_city
ORDER BY ts.departure_city;

```

Вывод:

| departure_city | arrival_city | sum_pass | sum_seats | frac |
|----------------|-----------------|----------|-----------|------|
| Абакан | Tomsk | 258 | 360 | 0.72 |
| Абакан | Novosibirsk | 217 | 348 | 0.62 |
| Абакан | Moscow | 466 | 1044 | 0.45 |
| ... | | | | |
| Якутск | Санкт-Петербург | 352 | 3596 | 0.10 |

(361 строка)

1) Как Вы считаете, равносильно ли в данном запросе `SELECT COUNT(s.seat_no)` и `SELECT COUNT(s.*)`? Ответ объяснить

Думаю, что это равносильные запросы, поскольку в данном столбце нет пустых ячеек. "Места определяют схему салона каждой модели. Каждое место определяется своим номером (`seat_no`) и имеет

закреплённый за ним класс обслуживания (`fare_conditions`)".
Если бы место не имело своего номера, то нарушилась бы логика базы данных, ведь нумерованное место в салоне - это не совсем корректная ситуация. Тогда не понятно, как это место вообще используется и где находится

Мои суждения верны, поскольку в приложении к используемой БД, есть следующая таблица, из которой видно, что столбец `seat_no` имеет модификатор NOT NULL:

| Столбец | Тип | Модификаторы | Описание |
|-------------|------------|--------------|--------------------------|
| ticket_no | char(13) | NOT NULL | Номер билета |
| flight_id | integer | NOT NULL | Идентификатор рейса |
| boarding_no | integer | NOT NULL | Номер посадочного талона |
| seat_no | varchar(4) | NOT NULL | Номер места |

2. Опишите, что делает каждый подзапрос

```
-- Подзапрос 1
SELECT COUNT(s.seat_no), COUNT( * )
FROM seats s
```

Считает количество мест для каждого самолета

```
-- Подзапрос 2
WITH tickets_seats AS
    (SELECT f.flight_id, f.flight_no, f.departure_city,
     f.arrival_city, f.aircraft_code,
     COUNT(tf.ticket_no) AS fact_passengers,
     (SELECT COUNT(s.seat_no)
      FROM seats s
      WHERE s.aircraft_code = f.aircraft_code ) AS
total_seats
    FROM flights_v f
    JOIN ticket_flights tf
    ON (f.flight_id = tf.flight_id)
```

```
WHERE f.status = 'Arrived'  
GROUP BY 1,2,3,4,5 )
```

1. **CTE (Common Table Expression):** `WITH tickets_seats AS (...)`

определяет временную таблицу, которая может быть использована в основном запросе.

2. **Фактические пассажиры:**

- `COUNT(tf.ticket_no) AS fact_passengers` : Считаются фактические пассажиры для каждого рейса, путем подсчета номеров билетов из таблицы `ticket_flights (tf)`, связанной с рейсами.

3. **Общее количество мест:**

- `(SELECT COUNT(s.seat_no) FROM seats s WHERE s.aircraft_code = f.aircraft_code) AS total_seats` :

Используется подзапрос для получения общего числа мест для конкретного типа самолета. Этот подзапрос считает количество мест (`seat_no`) в таблице `seats`, где код самолета совпадает с кодом, полученным из основного запроса.

4. **Объединение таблиц:** `JOIN ticket_flights tf ON (f.flight_id = tf.flight_id)` : Делает соединение между таблицами `flights_v` и `ticket_flights` по идентификатору рейса для получения информации о билетах, связанных с каждым рейсом.

5. **Фильтрация по статусу:** `WHERE f.status = 'Arrived'` : Фильтрует результаты, чтобы оставить только рейсы, которые уже прибыли.

6. **Группировка результатов:** `GROUP BY 1,2,3,4,5` : Группирует результаты по указанным полям, что позволяет применять агрегатные функции, такие как `COUNT`, для подсчета фактических пассажиров по каждому рейсу.

3) Модифицируйте этот запрос так, чтобы он выводил те же отчетные данные, но с учётом классов обслуживания (Business, Comfort и Economy)

```

WITH tickets_seats AS (
    SELECT
        f.flight_id,
        f.flight_no,
        f.departure_city,
        f.arrival_city,
        f.aircraft_code,
        tf.fare_conditions,
        COUNT(tf.ticket_no) AS fact_passengers,
        (SELECT COUNT(s.seat_no)
         FROM seats s
         WHERE s.aircraft_code = f.aircraft_code) AS total_seats
    FROM
        flights_v f
    JOIN
        ticket_flights tf ON f.flight_id = tf.flight_id
    WHERE
        f.status = 'Arrived'
    GROUP BY
        f.flight_id, f.flight_no, f.departure_city, f.arrival_city,
        f.aircraft_code, tf.fare_conditions)
SELECT
    ts.departure_city,
    ts.arrival_city,
    ts.fare_conditions,
    SUM(ts.fact_passengers) AS sum_pass,
    SUM(ts.total_seats) AS sum_seats,
    ROUND(SUM(ts.fact_passengers)::numeric /
    NULLIF(SUM(ts.total_seats)::numeric, 0), 2) AS frac
FROM
    tickets_seats ts
GROUP BY
    ts.departure_city,
    ts.arrival_city,
    ts.fare_conditions
ORDER BY
    ts.departure_city

```

