

МОСКОВСКИЙ АВИАЦИОННЫЙ ИНСТИТУТ
(НАЦИОНАЛЬНЫЙ ИССЛЕДОВАТЕЛЬСКИЙ УНИВЕРСИТЕТ)

Институт №8 «Компьютерные науки и прикладная математика»
Кафедра 806 «Вычислительная математика и программирование»

Лабораторная работа №5
по курсу «Проектирование баз данных»

Выполнила: Прудникова А. А.
Группа: М8О-114СВ-24
Преподаватель: Моргунов Е. П.

Москва, 2024

Задание 2

Этот запрос выбирает из таблицы «Билеты» (tickets) всех пассажиров с именами, состоящими из трех букв (в шаблоне присутствуют три символа «_»):

```
SELECT passenger_name
FROM tickets
WHERE passenger_name LIKE '___ %';
```

Предложите шаблон поиска в операторе LIKE для выбора из этой таблицы всех пассажиров с фамилиями, состоящими из пяти букв.

Запрос

```
SELECT passenger_name
FROM tickets
WHERE passenger_name LIKE '% _____';
```

Результат

passenger_name	
 Filter...	
ILYA POPOV	
VLADIMIR POPOV	
PAVEL GUSEV	
LEONID ORLOV	
EVGENIY GUSEV	
NIKOLAY FOMIN	
EKATERINA ILINA	
ANTON POPOV	
ARTEM BELOV	
VLADIMIR POPOV	
ALEKSEY ISAEV	
EMIL ISAEV	

Задание 7

Самые крупные самолеты в нашей авиакомпании — это Boeing 777-300. Выяснить, между какими парами городов они летают, поможет запрос:

```
SELECT DISTINCT departure_city, arrival_city
FROM routes r
JOIN aircrafts a ON r.aircraft_code = a.aircraft_code
```

```
WHERE a.model = 'Boeing 777-300'  
ORDER BY 1;
```



К сожалению, в этой выборке информация дублируется. Пары городов приведены по два раза: для рейса «туда» и для рейса «обратно». Модифицируйте запрос таким образом, чтобы каждая пара городов была выведена только один раз.

Запрос

Так как название в таблице хранится на различных языках, будем осуществлять поиск по коду самолета. Сначала выведем результат поиска с повторениями, затем устраним продублированные пары городов.

```
SELECT DISTINCT departure_city, arrival_city  
FROM routes r  
JOIN aircrafts a ON r.aircraft_code = a.aircraft_code  
WHERE a.aircraft_code = '773'  
ORDER BY 1;
```

Результат

departure_city	arrival_city
 Filter...	 Filter...
Екатеринбург	Москва
Москва	Екатеринбург
Москва	Новосибирск
Москва	Пермь
Москва	Сочи
Новосибирск	Москва
Пермь	Москва
Сочи	Москва

Запрос

```
SELECT DISTINCT departure_city, arrival_city  
FROM routes r  
JOIN aircrafts a ON r.aircraft_code = a.aircraft_code  
WHERE a.aircraft_code = '773'  
AND departure_city > arrival_city  
ORDER BY 1;
```

Результат

departure_city	arrival_city
abc Filter...	abc Filter...
Москва	Екатеринбург
Новосибирск	Москва
Пермь	Москва
Сочи	Москва

Задание 9

Для ответа на вопрос, сколько рейсов выполняется из Москвы в Санкт-Петербург, можно написать совсем простой запрос:

```
SELECT count( * )
      FROM routes
      WHERE departure_city = 'Москва'
            AND arrival_city = 'Санкт-Петербург';
```

count

12

(1 строка)

А с помощью какого запроса можно получить результат в таком виде?

```
departure_city | arrival_city | count
-----+-----+-----
      Москва   | Санкт-Петербург | 12
```

(1 строка)

Запрос

```
SELECT departure_city, arrival_city, COUNT(*) AS count
      FROM routes
      WHERE departure_city = 'Москва'
            AND arrival_city = 'Санкт-Петербург'
      GROUP BY departure_city, arrival_city;
```

Результат

departure_city	arrival_city	count
abc Filter...	abc Filter...	abc Filter...
Москва	Санкт-Петербург	12

Задание 13

Ответить на вопрос о том, каковы максимальные и минимальные цены билетов на все направления, может такой запрос:

```
SELECT f.departure_city, f.arrival_city,
       max( tf.amount ), min( tf.amount )
FROM flights_v f
JOIN ticket_flights tf ON f.flight_id = tf.flight_id
GROUP BY 1, 2
ORDER BY 1, 2;
```

А как выявить те направления, на которые не было продано ни одного билета?

Один из вариантов решения такой: если на рейсы, отправляющиеся по какому-то направлению, не было продано ни одного билета, то максимальная и минимальная цены будут равны NULL. Нужно получить выборку в таком виде:

departure_city	arrival_city	max	min
Абакан	Архангельск		
Абакан	Грозный		
Абакан	Кызыл		
Абакан	Москва	101000.00	33700.00
Абакан	Новосибирск	5800.00	5800.00
...			

Модифицируйте запрос, приведенный выше.

Запрос

```
SELECT f.departure_city, f.arrival_city,
       MAX(tf.amount) AS max_price, MIN(tf.amount) AS
min_price
FROM flights_v f
LEFT OUTER JOIN ticket_flights tf ON f.flight_id =
tf.flight_id
GROUP BY 1, 2
ORDER BY 1, 2;
```

Результат

departure_city	arrival_city	max_price	min_price
а Filter...	а Filter...	а Filter...	а Filter...
Абакан	Архангельск	NULL	NULL
Абакан	Грозный	NULL	NULL
Абакан	Кызыл	NULL	NULL
Абакан	Москва	101000.00	33700.00
Абакан	Новосибирск	5800.00	5800.00
Абакан	Томск	4900.00	4900.00
Абакан	Москва	105300.00	51000.00

Задание 19

В разделе 6.4 мы использовали рекурсивный алгоритм в общем табличном выражении. Изучите этот пример, чтобы лучше понять работу рекурсивного алгоритма:

```
WITH RECURSIVE ranges ( min_sum, max_sum )
AS (
    VALUES( 0,          100000 ),
           ( 100000, 200000 ),
           ( 200000, 300000 )
    UNION ALL
    SELECT min_sum + 100000, max_sum + 100000
    FROM ranges
    WHERE max_sum < ( SELECT max( total_amount ) FROM bookings )
)
SELECT * FROM ranges;
```

min_sum		max_sum	
-----+-----			
0		100000	исходные строки
100000		200000	
200000		300000	
100000		200000	результат первой итерации
200000		300000	
300000		400000	
200000		300000	результат второй итерации
300000		400000	
400000		500000	
300000		400000	
400000		500000	
500000		600000	
...			
1000000		1100000	результат (n-3)-й итерации
1100000		1200000	
1200000		1300000	
1100000		1200000	результат (n-2)-й итерации
1200000		1300000	
1200000		1300000	результат (n-1)-й итерации (предпоследней)

(36 строк)

Здесь мы с помощью предложения VALUES специально создали виртуальную таблицу из трех строк, хотя для получения требуемого результата достаточно только одной строки (0, 100000). Еще важно то, что предложение UNION ALL не удаляет строки-дубликаты, поэтому мы можем видеть весь рекурсивный процесс порождения новых строк.

При рекурсивном выполнении запроса

```
SELECT min_sum + 100000, max_sum + 100000
FROM ranges
WHERE max_sum < ( SELECT max( total_amount ) FROM bookings )
```

каждый раз выполняется проверка в условии WHERE. И на $(n - 2)$ -й итерации это условие отсеивает одну строку, т. к. после $(n - 3)$ -й итерации значение атрибута max_sum в третьей строке было равно 1 300 000.

Ведь запрос

```
SELECT max( total_amount ) FROM bookings;
```

выдаст значение

```
      max
-----
1204500.00
(1 строка)
```

Таким образом, после $(n - 2)$ -й итерации во временной области остается всего две строки, после $(n - 1)$ -й итерации во временной области остается только одна строка.

Заключительная итерация уже не добавляет строк в результирующую таблицу, поскольку единственная строка, поданная на вход команде SELECT, будет отклонена условием WHERE. Работа алгоритма завершается.

Задание 1. Модифицируйте запрос, добавив в него столбец level (можно назвать его и iteration). Этот столбец должен содержать номер текущей итерации, поэтому нужно увеличивать его значение на единицу на каждом шаге. Не забудьте задать начальное значение для добавленного столбца в предложении VALUES.

Задание 2. Для завершения экспериментов замените UNION ALL на UNION и выполните запрос. Сравните этот результат с предыдущим, когда мы использовали UNION ALL.

Запрос

```
WITH RECURSIVE ranges ( min_sum, max_sum, level )
AS (
VALUES ( 0, 100000, 1 ),
```

```

        ( 100000, 200000, 2 ),
        ( 200000, 300000, 3 )
    UNION ALL
    SELECT min_sum + 100000, max_sum + 100000, level + 1
        FROM ranges
        WHERE max_sum < ( SELECT max( total_amount ) FROM
bookings )
    )
SELECT * FROM ranges;




```




```

WITH RECURSIVE ranges ( min_sum, max_sum, level )
AS (
    VALUES( 0, 100000, 1 ),
        ( 100000, 200000, 2 ),
        ( 200000, 300000, 3 )
    UNION
    SELECT min_sum + 100000, max_sum + 100000, level + 1
        FROM ranges
        WHERE max_sum < ( SELECT max( total_amount ) FROM
bookings )
    )
SELECT * FROM ranges;

```

Результат

min_sum	max_sum	level
 Filter...	 Filter...	 Filter...
0	100000	1
100000	200000	2
200000	300000	3
100000	200000	2
200000	300000	3
300000	400000	4
200000	300000	3
300000	400000	4
400000	500000	5

min_sum	max_sum	level
 Filter...	 Filter...	 Filter...
0	100000	1
100000	200000	2
200000	300000	3
300000	400000	4
400000	500000	5
500000	600000	6
600000	700000	7
700000	800000	8
800000	900000	9
900000	1000000	10
1000000	1100000	11
1100000	1200000	12
1200000	1300000	13

Задание 21

Вместо знака «?» поставьте в приведенном ниже запросе нужное ключевое слово — UNION, INTERSECT или EXCEPT — и обоснуйте ваше решение.

```
SELECT city
  FROM airports
 WHERE city <> 'Москва'
?
SELECT arrival_city
  FROM routes
 WHERE departure_city = 'Москва'
ORDER BY city;
```

Решение

Так как нашей задачей является исключение Москвы из списка городов, из которых совершаются рейсы, используем EXCEPT и выделим ту часть маршрутов, в которых Москва является отправочным городом.

Запрос

```
SELECT city
  FROM airports
 WHERE city <> 'Москва'
EXCEPT
SELECT arrival_city
  FROM routes
 WHERE departure_city = 'Москва'
ORDER BY city;
```

Результат

city	
Filter...	
Благовещенск	
Иваново	
Иркутск	
Калуга	
Когалым	
Комсомольск-на-Аму...	
Кызыл	
Магадан	
Нижнекамск	
Новокузнецк	
Стрежевой	
Сургут	
Удачный	
Усть-Илимск	
Усть-Кут	
Ухта	
Череповец	
Чита	
Якутск	
Ярославль	

Задание 23

```
SELECT count( * )
  FROM ( SELECT DISTINCT city FROM airports ) AS a1
  JOIN ( SELECT DISTINCT city FROM airports ) AS a2
    ON a1.city <> a2.city;
```

Перепишите этот запрос с общим табличным выражением.

Запрос

```
WITH f AS ( SELECT DISTINCT city FROM airports )
SELECT count( * )
  FROM f AS a1
  JOIN f AS a2
    ON a1.city <> a2.city;
```

Результат

count
Filter...
10100