МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ

Федеральное государственное бюджетное образовательное учреждение высшего образования

“Московский авиационный институт”

(национальный исследовательский университет)

Факультет №3 «Системы управления, информатика и электроэнергетика»

Отчет по домашней работе №9

по курсу «**Управление базами данных**»

Выполнили:

студенты группы 3О-218М-19:

Пономарев Роман

Принял:

Доцент, к.т.н. Моргунов Е.П.

Москва, 2020

№3)
Самостоятельно выполните команду EXPLAIN для запроса, содержащего общее табличное выражение (CTE). Посмотрите, на каком уровне находится узел плана, отвечающий за это выражение, как он оформляется. Учтите, что общие табличные выражения всегда материализуются, т. е. вычисляются однократно и результат их вычисления сохраняется в памяти, а затем все последующие обращения в рамках запроса направляются уже к этому материализованному результату.

Решение

```
EXPLAIN WITH ap1 AS
(
SELECT DISTINCT city FROM airports
),
ap2 AS
(
SELECT DISTINCT city FROM airports
)
SELECT count(*) FROM ap1 JOIN ap2 ON ap1.city<>ap2.city
```

```
demo=# EXPLAIN WITH ap1 AS
demo-# (
demo(# SELECT DISTINCT city FROM airports
demo(# ),
demo-# ap2 AS
demo-# (
demo(# SELECT DISTINCT city FROM airports
demo(# )
demo-# SELECT count(*) FROM ap1 JOIN ap2 ON ap1.city<>ap2.city;
                                          QUERY PLAN

------------------------------------------------------------------------------
--------------
 Aggregate  (cost=189.16..189.17 rows=1 width=8)
   ->  Nested Loop  (cost=6.60..163.91 rows=10100 width=0)
         Join Filter: (airports.city <> airports_1.city)
         ->  HashAggregate  (cost=3.30..4.31 rows=101 width=17)
               Group Key: airports.city
               ->  Seq Scan on airports  (cost=0.00..3.04 rows=104 width=17)
         ->  Materialize  (cost=3.30..5.82 rows=101 width=17)
               ->  HashAggregate  (cost=3.30..4.31 rows=101 width=17)
                     Group Key: airports_1.city
                     ->  Seq Scan on airports airports_1  (cost=0.00..3.04 rows=
104 width=17)
(10 строк)
```

№6)

Выполните команду EXPLAIN для запроса, в котором использована какаянибудь из оконных функций. Найдите в плане выполнения запроса узел с именем WindowAgg. Попробуйте объяснить, почему он занимает именно этот уровень в плане.

Решение

```
EXPLAIN SELECT airport_name,
city,
round( latitude::numeric, 2 ) AS ltd,
timezone,
rank() OVER (
PARTITION BY timezone
ORDER BY latitude DESC
)
FROM airports
WHERE timezone IN ( 'Asia/Irkutsk', 'Asia/Krasnoyarsk' )
ORDER BY timezone, rank;
```

```
demo=# EXPLAIN SELECT airport_name, city,
demo-# round( latitude::numeric, 2) AS ltd,
demo-# timezone,
demo-# rank() OVER (
demo(# PARTITION BY timezone
demo(# ORDER BY latitude DESC
demo(# )
demo-# FROM airports
demo-# WHERE timezone IN ( 'Asia/Irkutsk', 'Asia/Krasnoyarsk')
demo-# ORDER BY timezone, rank;
                                        QUERY PLAN
-------------------------------------------------------------------------------------------
 Sort  (cost=4.11..4.14 rows=13 width=97)
   Sort Key: timezone, (rank() OVER (?))
   ->  WindowAgg  (cost=3.54..3.87 rows=13 width=97)
         ->  Sort  (cost=3.54..3.57 rows=13 width=57)
               Sort Key: timezone, latitude DESC
               ->  Seq Scan on airports  (cost=0.00..3.30 rows=13 width=57)
                     Filter: (timezone = ANY ('{Asia/Irkutsk,Asia/Krasnoyarsk}'::text[]))
(7 строк)
```

Происходит сортировка записей из предложения ORDER BY главного запроса, после чего, полученные записи разбиваются на окна, далее происходит сортировка в пределах окна.

№8)

Замена коррелированного подзапроса соединением таблиц является одним из способов повышения производительности. Предположим, что

мы задались вопросом: сколько маршрутов обслуживают самолеты каждого типа? При этом нужно учитывать, что может иметь место такая ситуация, когда самолеты какого-либо типа не обслуживают ни одного маршрута. Поэтому необходимо использовать не только представление «Маршруты» (routes), но и таблицу «Самолеты» (aircrafts).
Это первый вариант запроса, в нем используется коррелированный подзапрос.

```
EXPLAIN ANALYZE
  SELECT a.aircraft_code AS a_code,
         a.model,
         ( SELECT count( r.aircraft_code )
             FROM routes r
             WHERE r.aircraft_code = a.aircraft_code
         ) AS num_routes
    FROM aircrafts a
    GROUP BY 1, 2
    ORDER BY 3 DESC;
```

А в этом варианте коррелированный подзапрос раскрыт и заменен внешним соединением:

```
EXPLAIN ANALYZE
  SELECT a.aircraft_code AS a_code,
         a.model,
         count( r.aircraft_code ) AS num_routes
    FROM aircrafts a
    LEFT OUTER JOIN routes r
      ON r.aircraft_code = a.aircraft_code
    GROUP BY 1, 2
    ORDER BY 3 DESC;
```

Причина использования внешнего соединения в том, что может найтись модель самолета, не обслуживающая ни одного маршрута, и если не использовать внешнее соединение, она вообще не попадет в результирующую выборку.
Исследуйте планы выполнения обоих запросов. Попытайтесь найти объяснение различиям в эффективности их выполнения. Чтобы получить усредненную картину, выполните каждый запрос несколько раз. Поскольку таблицы, участвующие в запросах, небольшие, то различие по абсолютным затратам времени выполнения будет незначительным. Но если бы число строк в таблицах было большим, то экономия ресурсов сервера могла оказаться заметной.
Предложите аналогичную пару запросов к базе данных «Авиаперевозки». Проведите необходимые эксперименты с вашими запросами.

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

```
EXPLAIN ANALYZE SELECT f.departure_city, f.arrival_city,
max( 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;

EXPLAIN ANALYZE SELECT f.departure_city, f.arrival_city,
(SELECT max( tf.amount ) FROM
ticket_flights tf WHERE tf.flight_id = f.flight_id)
FROM flights_v f
GROUP BY 1, 2, f.flight_id
ORDER BY 1, 2;
```

Второй запрос начинает выполняться и ничего не происходит.