Мамонов Антон 2ИСиП-19-1

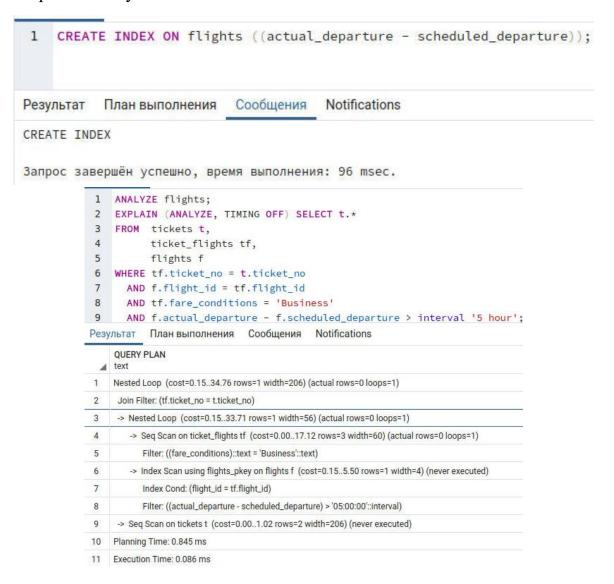
Приемы оптимизации

Отключение параллельного выполнения

```
1
    SET max_parallel_workers_per_gather = 0;
2
3
    EXPLAIN (ANALYZE, TIMING OFF) SELECT t.*
4
    FROM tickets t,
5
          ticket_flights tf,
6
          flights f
    WHERE tf.ticket_no = t.ticket_no
7
     AND f.flight_id = tf.flight_id
8
9
      AND tf.fare conditions = 'Business'
10
      AND f.actual_departure > f.scheduled_departure
```

```
QUERY PLAN
      text
     Nested Loop (cost=0.15..34.76 rows=1 width=206) (actual rows=0 loops=1)
      Join Filter: (tf.ticket_no = t.ticket_no)
      -> Nested Loop (cost=0.15..33.71 rows=1 width=56) (actual rows=0 loops=1)
         -> Seq Scan on ticket_flights tf (cost=0.00..17.12 rows=3 width=60) (actual rows=0 loops=1)
 5
             Filter: ((fare_conditions)::text = 'Business'::text)
6
         -> Index Scan using flights_pkey on flights f (cost=0.15..5.50 rows=1 width=4) (never executed)
7
            Index Cond: (flight_id = tf.flight_id)
8
            Filter: (actual_departure > (scheduled_departure + '05:00:00'::interval))
9 -> Seq Scan on tickets t (cost=0.00..1.02 rows=2 width=206) (never executed)
10 Planning Time: 0.936 ms
11 Execution Time: 0.084 ms
```

Постройка функционального индекса на разности двух столбцов и переписание условия



Теперь займемся таблицей ticket_flights, которая тоже сканируется полностью, хотя из нее читается незначительная часть строк.

Помог бы индекс по классам обслуживания fare_conditions, но лучше создать индекс по столбцу flight_id, что позволит эффективно выполнять соединение вложенным циклом с flights.

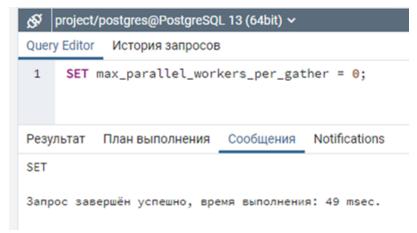
```
1 CREATE INDEX ON ticket_flights(flight_id);
   EXPLAIN (ANALYZE, TIMING OFF) SELECT t.*
2
  FROM tickets t,
3
         ticket_flights tf,
4
         flights f
5
  WHERE tf.ticket_no = t.ticket_no
6
     AND f.flight_id = tf.flight_id
7
     AND tf.fare_conditions = 'Business'
8
9
     AND f.actual_departure - f.scheduled_departure > interval '5 hour';
```

4	QUERY PLAN text
1	Nested Loop (cost=0.1534.76 rows=1 width=206) (actual rows=0 loops=1)
2	Join Filter: (tf.ticket_no = t.ticket_no)
3	-> Nested Loop (cost=0.1533.71 rows=1 width=56) (actual rows=0 loops=1)
4	-> Seq Scan on ticket_flights tf (cost=0.0017.12 rows=3 width=60) (actual rows=0 loops=1)
5	Filter: ((fare_conditions)::text = 'Business'::text)
6	-> Index Scan using flights_pkey on flights f (cost=0.155.50 rows=1 width=4) (never executed)
7	Index Cond: (flight_id = tf.flight_id)
8	Filter: ((actual_departure - scheduled_departure) > '05:00:00'::interval)
9	-> Seq Scan on tickets t (cost=0.001.02 rows=2 width=206) (never executed)
10	Planning Time: 0.423 ms
11	Execution Time: 0.047 ms

Время планирования и выполнения уменьшилось.

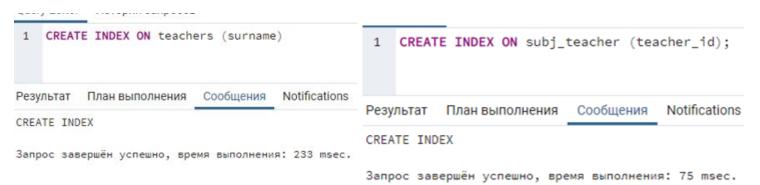
Проверка на собственной БД

Отключение параллельного выполнения





Можно построить функциональный индекс на разности двух столбцов и немного переписать условие



```
1 EXPLAIN (ANALYZE, TIMING OFF) SELECT t.*
2 FROM subjects t,
3
           subj_teacher tf,
4
           teachers f
5 WHERE tf.subject_id = t.subject_id
     AND f.teacher_id = tf.teacher_id
6
      AND tf.subject_id = 101
Результат План выполнения Сообщения Notifications
     OUERY PLAN
    text
    Hash Join (cost=1.24..39.89 rows=10 width=36) (actual rows=1 loops=1)
     Hash Cond: (tf.teacher_id = f.teacher_id)
     -> Nested Loop (cost=0.15..38.77 rows=10 width=40) (actual rows=1 loops=1)
4
       -> Index Scan using subjects_pkey on subjects t (cost=0.15..3.17 rows=1 width=36) (actual rows=1 loops=1)
5
           Index Cond: (subject_id = 101)
6
        -> Seq Scan on subj_teacher tf (cost=0.00..35.50 rows=10 width=8) (actual rows=1 loops=1)
7
          Filter: (subject_id = 101)
8
          Rows Removed by Filter: 3
    -> Hash (cost=1.04..1.04 rows=4 width=4) (actual rows=4 loops=1)
10
       Buckets: 1024 Batches: 1 Memory Usage: 9kB
11
        -> Seq Scan on teachers f (cost=0.00..1.04 rows=4 width=4) (actual rows=4 loops=1)
12 Planning Time: 0.811 ms
13 Execution Time: 0.127 ms
```

```
1 EXPLAIN (ANALYZE, TIMING OFF) SELECT t.*
 2 FROM subjects t,
 3
             subj_teacher tf,
 4
             teachers f
 5 WHERE tf.subject_id = t.subject_id
 6
       AND f.teacher_id = tf.teacher_id
 7
       AND tf.subject_id = 101
Результат План выполнения Сообщения Notifications
     QUERY PLAN

    dext

1
     Nested Loop (cost=0.15..5.32 rows=1 width=36) (actual rows=1 loops=1)
2
     Join Filter: (tf.teacher_id = f.teacher_id)
3
      Rows Removed by Join Filter: 1
4
     -> Nested Loop (cost=0.15..4.23 rows=1 width=40) (actual rows=1 loops=1)
5
       -> Index Scan using subjects_pkey on subjects t (cost=0.15..3.17 rows=1 width=36) (actual rows=1 loops=1)
6
           Index Cond: (subject_id = 101)
7
        -> Seq Scan on subj_teacher tf (cost=0.00..1.05 rows=1 width=8) (actual rows=1 loops=1)
8
           Filter: (subject_id = 101)
9
           Rows Removed by Filter: 3
     -> Seq Scan on teachers f (cost=0.00..1.04 rows=4 width=4) (actual rows=2 loops=1)
10
11
     Planning Time: 0.195 ms
12 Execution Time: 0.053 ms
```

Время планирования и выполнения уменьшилось.