**МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ**

**РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное бюджетное образовательное учреждение**

**высшего образования**

**«Казанский государственный энергетический»**

**Отчет по лабораторной работе 6**

**Соединение хешированием**

Выполнил:

студент группы ПИ-1-22

Соловьёв Л.А.

Проверил:

Хамитов Р. М.

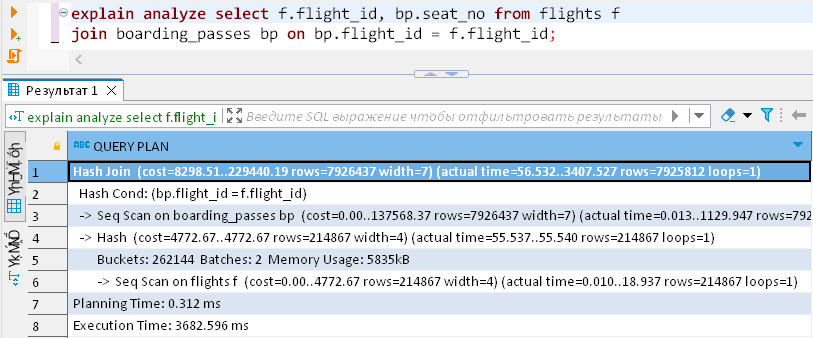
Казань 2024

**1. Напишите запрос, показывающий занятые места в салоне для всех рейсов.**

**Какой способ соединения выбрал планировщик? Проверьте, хватило ли оперативной памяти для размещения хеш-таблиц. Для этого достаточно соединить рейсы (flights) с посадочными талонами (boarding\_passes).**

**explain** **analyze** **select** f.flight\_id, bp.seat\_no **from** flights f

**join** boarding\_passes bp **on** bp.flight\_id = f.flight\_id;

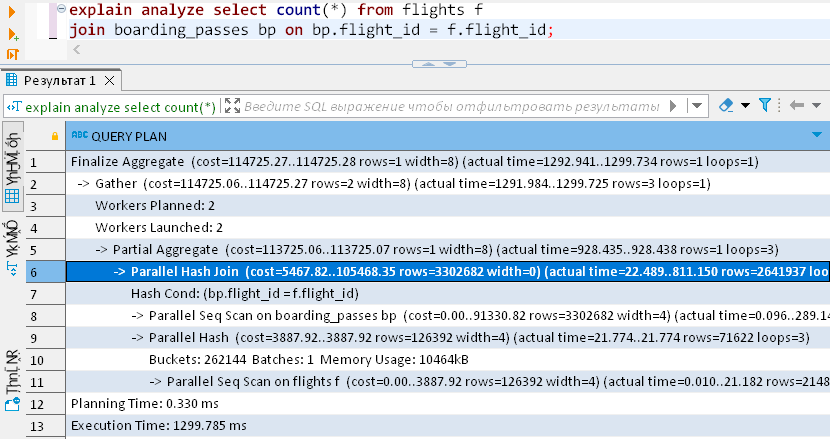
****

Планировщик выбрал соединение хэшированием, оперативной памяти не хватило, потребовалось 2 пакета.

**2. Измените запрос, чтобы он выводил только общее количество занятых мест. Как изменился план запроса? Почему планировщик не использовал аналогичный план для предыдущего запроса?**

**explain** **analyze** **select** **count**(\*) **from** flights f

**join** boarding\_passes bp **on** bp.flight\_id = f.flight\_id;



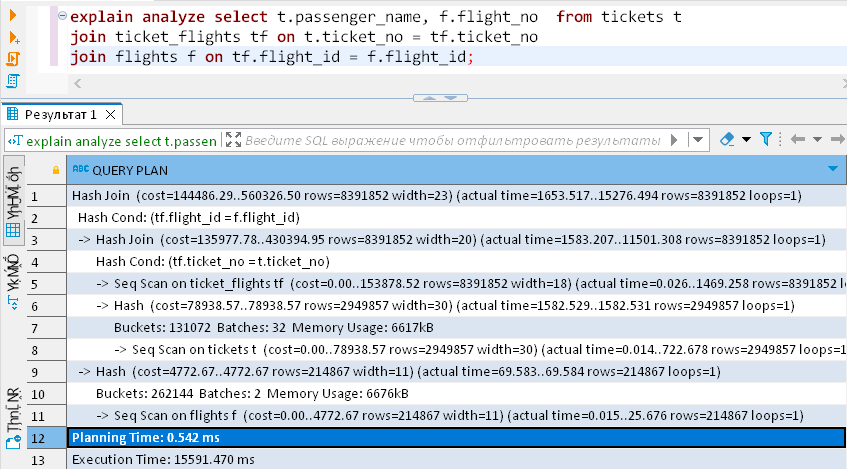
Планировщик использовал параллельное сканирование, потому что в результате этого запроса получается одно число, количество строк, соответствующих условию.

**3. Напишите запрос, показывающий имена пассажиров и номера рейсов, которыми они следуют. Разберитесь по плану запроса, в какой последовательности выполняются операции. Такой запрос должен соединять три таблицы: билеты (tickets), перелеты (ticket\_flights) и рейсы (flights)**

**explain** **analyze** **select** t.passenger\_name, f.flight\_no **from** tickets t

**join** ticket\_flights tf **on** t.ticket\_no = tf.ticket\_no

**join** flights f **on** tf.flight\_id = f.flight\_id;

****

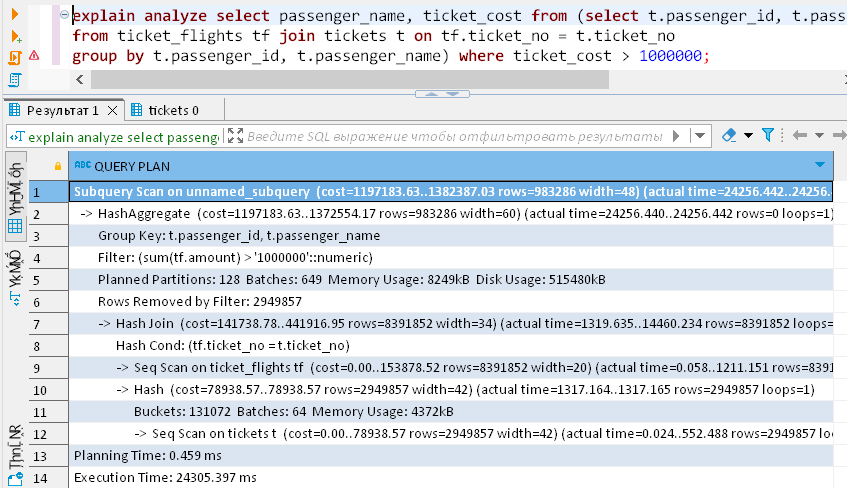
Сначала выполняется хэширование таблицы билетов, затем соединение хэшированием с перелетами по условию, потом выполняется хэширование таблицы рейсов и в конце соединяются хэшированием результат первого соединения и хэшированной таблицы рейсов.

**4.  Напишите запрос, показывающий имена пассажиров забронировавших билеты на сумму свыше 1 млн. рублей. Опишите план запроса. Как изменится план запрос если выбирать пассажиров бронировавших на сумму менее 100 тыс. рублей.**

**explain** **analyze** **select** passenger\_name, ticket\_cost **from** (**select** t.passenger\_id, t.passenger\_name, **sum**(tf.amount) **"ticket\_cost"**

**from** ticket\_flights tf **join** tickets t **on** tf.ticket\_no = t.ticket\_no

**group** **by** t.passenger\_id, t.passenger\_name) **where** ticket\_cost > 1000000;

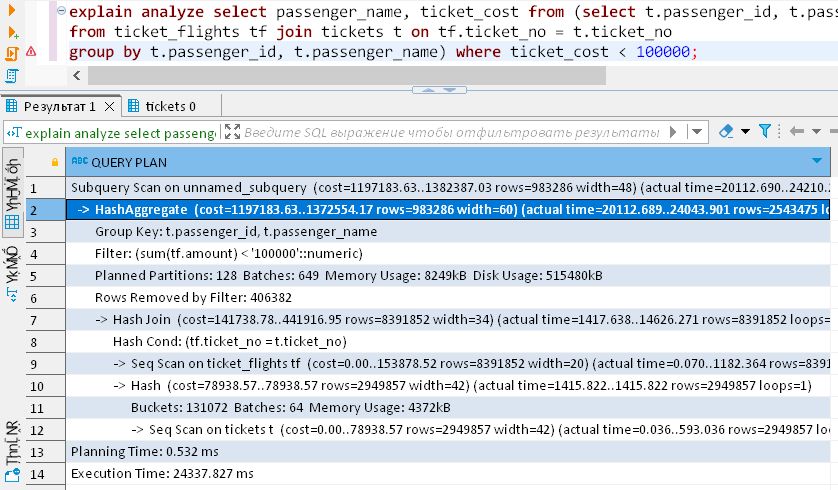
****

Сначала хэшируется таблица билетов, затем она соединяется хэшированием с таблицей перелетов, вычисляется общая сумма билетов каждого отдельного пассажира(id + name) и фильтруется по этой сумме

**explain** **analyze** **select** passenger\_name, ticket\_cost **from** (**select** t.passenger\_id, t.passenger\_name, **sum**(tf.amount) **"ticket\_cost"**

**from** ticket\_flights tf **join** tickets t **on** tf.ticket\_no = t.ticket\_no

**group** **by** t.passenger\_id, t.passenger\_name) **where** ticket\_cost < 100000

  
В сравнении с запросом для стоимости более 1 млн. ничего не изменилось.