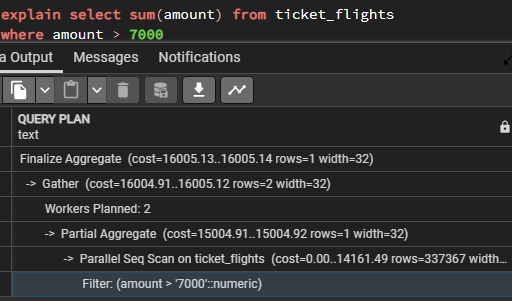
**Лабораторная работа № 12**

**Букин Николай**





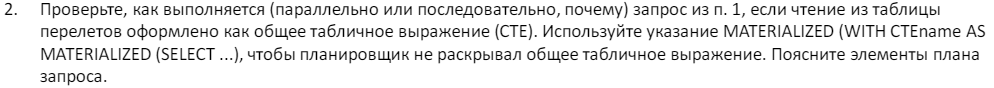
1. "Finalize Aggregate (cost=16005.13..16005.14 rows=1 width=32)": Это завершающий этап запроса, где происходит суммирование результатов всех рабочих процессов. Параметр "cost" указывает на оценочную стоимость выполнения этого шага, "rows" - на ожидаемое количество возвращаемых строк, а "width" - на средний размер строки в байтах.

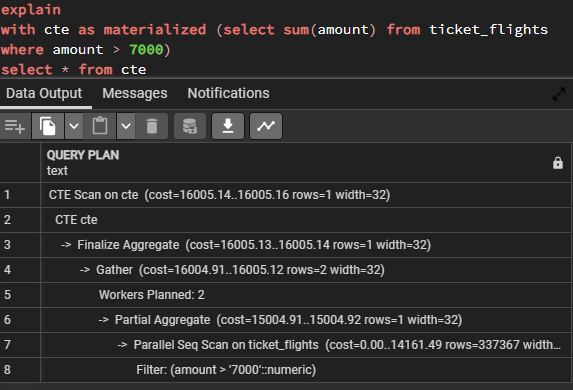
2. "-> Gather (cost=16004.91..16005.12 rows=2 width=32)": Этот шаг объединяет результаты всех рабочих процессов. "Workers Planned: 2" означает, что планируется использование двух рабочих процессов, что подтверждает параллельное выполнение запроса.

3. "-> Partial Aggregate (cost=15004.91..15004.92 rows=1 width=32)": На этом шаге каждый рабочий процесс выполняет частичное суммирование стоимости билетов.

4. "-> Parallel Seq Scan on ticket\_flights (cost=0.00..14161.49 rows=337367 width=6)": Этот шаг представляет собой параллельное сканирование таблицы ticket\_flights каждым рабочим процессом в поиске билетов, стоимость которых превышает 7000.

5. "Filter: (amount > '7000'::numeric)": Это условие фильтрации.





1. "CTE Scan on cte (cost=16005.14..16005.16 rows=1 width=32)": Этот этап выполняется после вычисления всех результатов CTE.

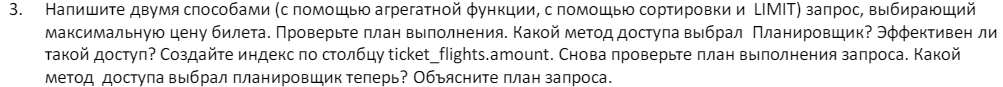
2. "CTE cte": Это определение общего табличного выражения (CTE), используемого в запросе.

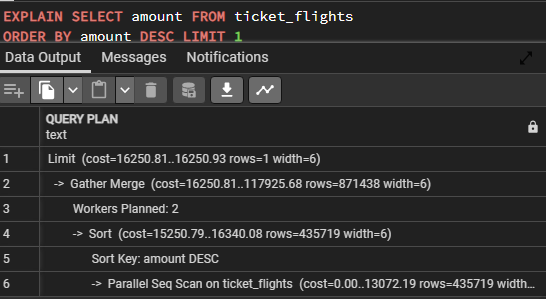
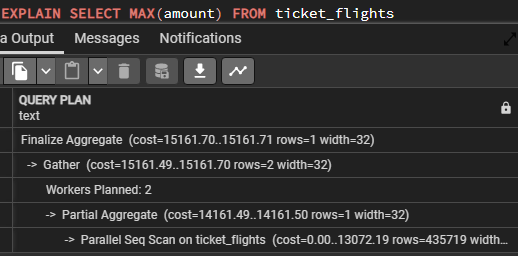
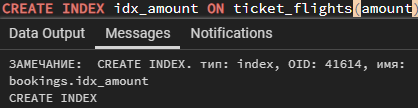
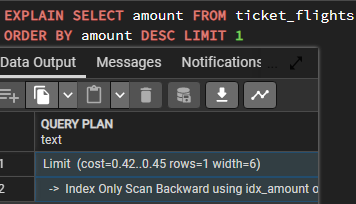
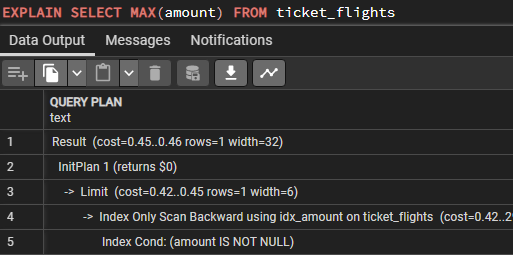
3. "-> Finalize Aggregate (cost=16005.13..16005.14 rows=1 width=32)": Завершающий шаг агрегирования, аналогичный первому пункту из предыдущего задания.

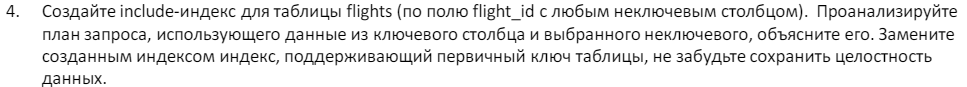
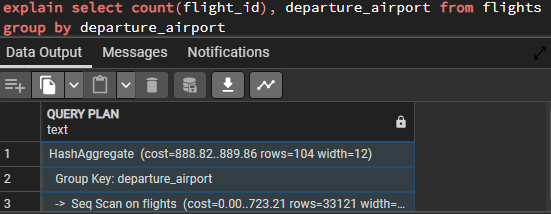
4. "-> Gather (cost=16004.91..16005.12 rows=2 width=32)": Шаг, объединяющий результаты всех рабочих процессов, аналогичный второму пункту из предыдущего задания, подтверждая параллельное выполнение.

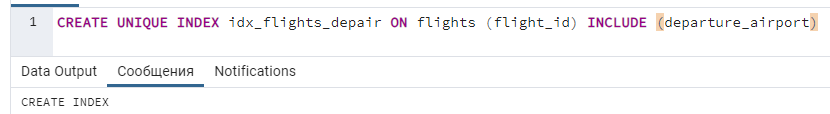
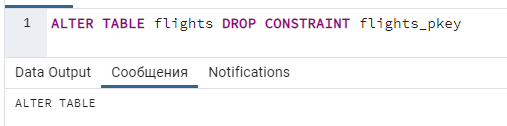
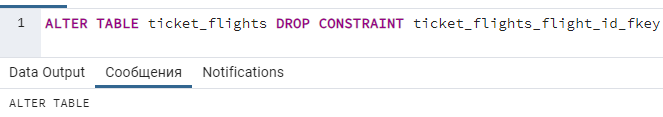
5. "-> Partial Aggregate (cost=15004.91..15004.92 rows=1 width=32)": Шаг частичного агрегирования, аналогичный третьему пункту из предыдущего задания.

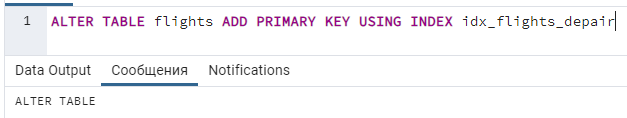
6. "-> Parallel Seq Scan on ticket\_flights (cost=0.00..14161.49 rows=337367 width=6)": Параллельное сканирование таблицы ticket\_flights, аналогичное четвертому пункту из предыдущего задания.



********

****

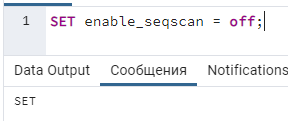
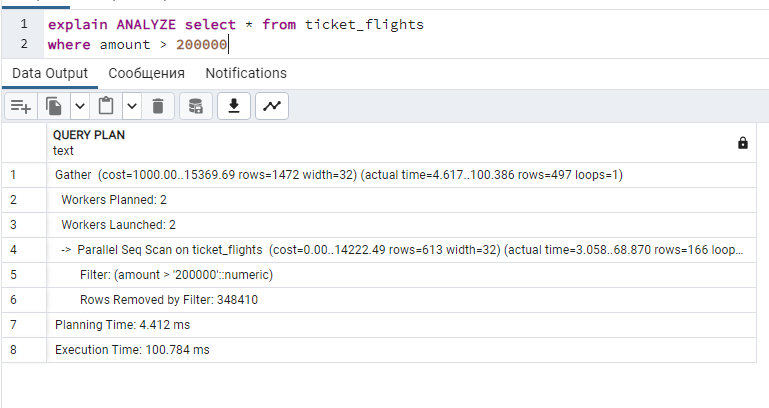
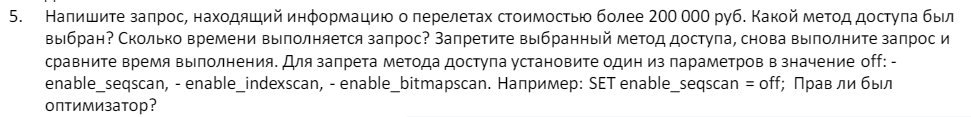


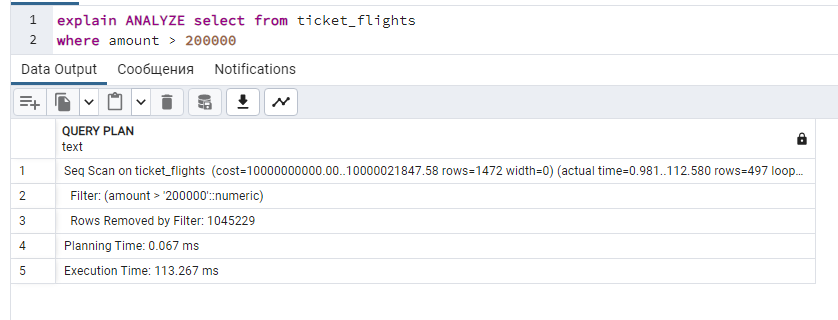


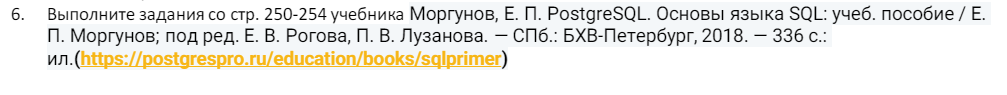
1. "HashAggregate (cost=888.82..889.86 rows=104 width=12)": Этот этап агрегирует данные с помощью хеш-таблицы. Параметр "cost" обозначает приблизительную стоимость выполнения, "rows" - предполагаемое количество возвращаемых строк, а "width" - средний размер строки в байтах. Указание "Group Key: departure\_airport" означает, что данные группируются по столбцу `departure\_airport`.

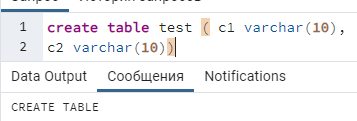
2. "-> Seq Scan on flights (cost=0.00..723.21 rows=33121 width=8)": Этот шаг выполняет полное последовательное сканирование таблицы flights, просматривая все строки без использования индексов.

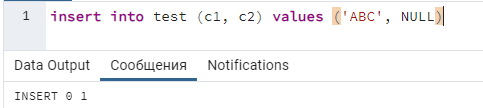
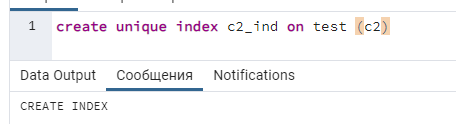
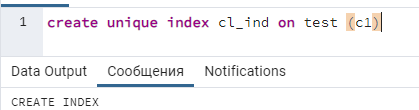
Для замены индекса сначала необходимо удалить все внешние ключи.









  
Повторное выполнение запроса может происходить быстрее благодаря кэшированию. После первого выполнения база данных может сохранить определенные данные в памяти, что ускоряет обработку аналогичных последующих запросов.

Когда выполняется запрос SELECT без использования индекса, база данных вынуждена проверять каждую строку в таблице, чтобы найти те, которые соответствуют условию WHERE. Этот процесс называется полным сканированием таблицы и может быть очень медленным при работе с большими таблицами.

Однако, если создать индекс, база данных сможет использовать его для быстрого поиска строк, которые соответствуют условию WHERE, минуя необходимость проверки каждой строки. Этот метод называется индексным сканированием и обычно значительно быстрее по сравнению с полным сканированием таблицы.









Создание индекса позволяет базе данных эффективно находить строки, соответствующие условию WHERE, обходя необходимость проверки каждой записи. Этот метод, известный как индексное сканирование, значительно ускоряет процесс по сравнению с полным сканированием таблицы. Индексное сканирование обычно обеспечивает гораздо более высокую производительность при поиске данных в больших таблицах.