|  |  |
| --- | --- |
| **Gerb-BMSTU_01** | **Министерство науки и высшего образования Российской Федерации**  **Федеральное государственное бюджетное образовательное учреждение**  **высшего образования**  **«Московский государственный технический университет**  **имени Н.Э. Баумана**  **(национальный исследовательский университет)»**  **(МГТУ им. Н.Э. Баумана)** |

ФАКУЛЬТЕТ **Информатика и системы управления**

КАФЕДРА **Компьютерные системы и сети (ИУ6)**

НАПРАВЛЕНИЕ ПОДГОТОВКИ **09.03.01** Информатика и вычислительная техника

**Отчет**

|  |
| --- |
| **по Лабораторной работе № 1** |

**Название: Введение в SQL**

**Дисциплина:** Базы данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Студент | ИУ6-32Б |  |  | А. П. Плютто |
|  | (Группа) |  | (Подпись, дата) | (И.О. Фамилия) |
|  |  |  |  |  |
| Преподаватель |  |  |  | М. А. Скворцова |
|  |  |  | (Подпись, дата) | (И.О. Фамилия) |

Москва, 2023

**Текст задания:**

Первое практическое задание заключается в знакомстве со средой pgAdmin и написании SQL-запросов с использованием оператора SELECT. Для модельной базы данных должны быть составлены 4 произвольных SELECT-запроса, демонстрирующие полученные знания. Запросы должны охватывать проработанные темы. После составления запросов следует убедиться в их правильности при помощи более простых запросов. Дополнительные вопросы могут заключаться в построении более сложных запросов или объяснении работы подготовленных заданий.

**Запрос 1**

Напишем запрос, который группирует все билеты по классу комфорта и минимальная цена билета больше 5000. В таблице выведем все классы комфорта, которые соответствуют последнему условию и некоторую информацию о них: сколько билетов каждого класса, средняя стоимость и насколько дороже может быть максимальная стоимость.

1. select fare\_conditions, count(fare\_conditions), round(avg(amount)) as avg, round(max(amount) - avg(amount)) as delta from bookings.ticket\_flights group by fare\_conditions having min(amount) > 5000;

2.

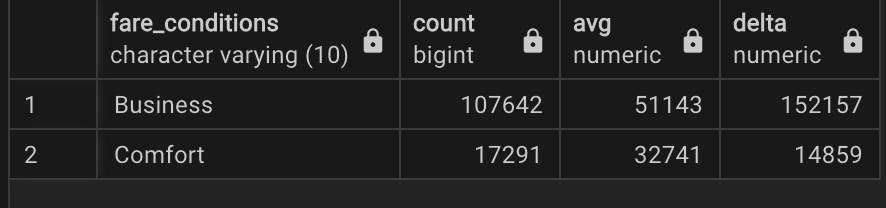


Рисунок 1. Результаты выполнения запроса 1

**Запрос 2**

Напишем запрос, который находит 10 самых больших покупок, не превышающих суммы в 800000 и выведем их стоимость и процент, на сколько их стоимость меньше от 800000.

1. select round(total\_amount, 0) as amount, round(total\_amount/800000\*100, 2) as persent from bookings.bookings where total\_amount < 800000 order by total\_amount desc limit 10;

2.

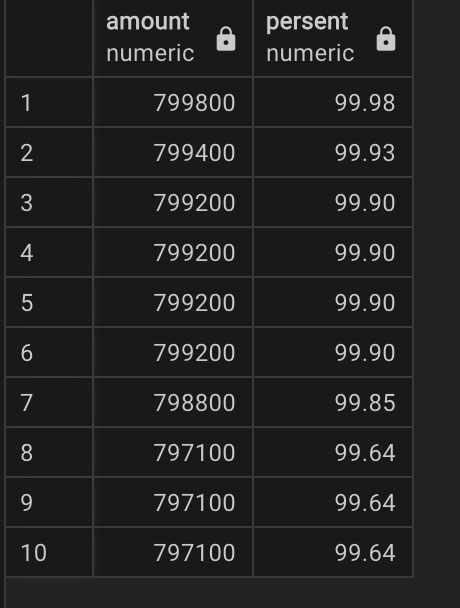


Рисунок 2. Результаты выполнения запроса 2

**Запрос 3**

Посмотрев на таблицу bookings можно увидеть, что несколько пользователей купило билеты в одну и ту же секунду, но их не очень много, поэтому группировка по времени все равно выдаст достаточно большую таблицу.

Найдем 10 самых поздних покупок, совершенных 4-6 пользователями в одну и ту же секунду и выведем на экран количество этих покупок и дату и время покупки.

1. select count(book\_date) as date\_count, book\_date from bookings.bookings group by book\_date having count(book\_date) between 4 and 6 order by book\_date desc limit 10;

2.

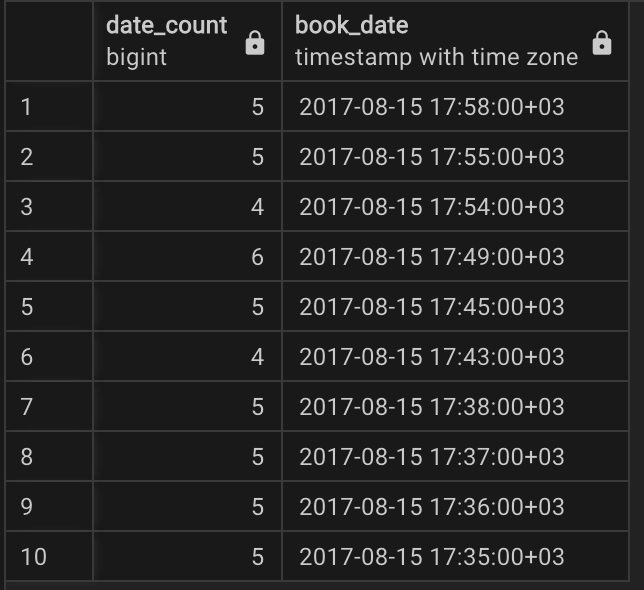


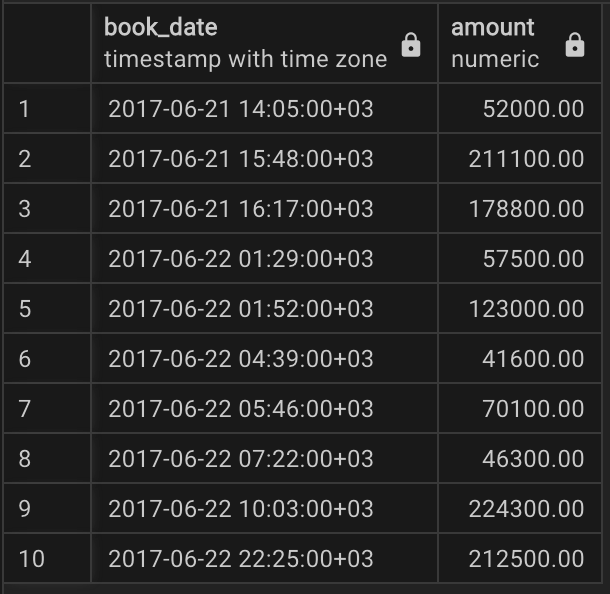
Рисунок 3. Результаты выполнения запроса 3

**Запрос 4**

Напишем еще один запрос, связанный со временем покупки. Теперь будем искать 10 самых ранних максимальных сумм покупки из единовременных и выводить ее вместе со временем совершения покупки.

1. select book\_date, max(total\_amount) over (partition by book\_date) as amount from bookings.bookings order by book\_date limit 10;

2.



**Ответы на контрольные вопросы**

1. Запрос 1:

Описание запроса:

С помощью ключевого слова SELECT выбираем данные, которые будут включены в итоговую таблицу. В данном запросе выбираем поле fare\_conditions. Дополнительно, вычисляем количество записей для каждого fare\_condition, округляем среднее значение столбца amount до целых (с использованием функции avg() для вычисления среднего и round() для округления), а также вычисляем разницу между максимальным значением amount и средним значением (с использованием функции max() и арифметических операций). Затем, с помощью ключевого слова FROM указываем таблицу ticket\_flights, из которой выполняется выборка данных. Словом GROUP указываем группировку строк по полю fare\_conditions. И, наконец, с помощью ключевого слова HAVING фильтруем результаты, оставляя только те группы, в которых минимальное значение amount больше 5000.

Запрос 2:

Описание запроса:

В данном запросе используется ключевое слово SELECT для выбора данных. Мы выбираем столбец total\_amount и округляем его до целых. Затем, вычисляем процентное отношение каждой записи total\_amount к числу 800000 с точностью до двух знаков после запятой. Затем, с помощью ключевого слова FROM, указываем таблицу bookings, из которой выполняется выборка. Словом WHERE фильтруем записи, оставляя только те, где total\_amount меньше 800000. Далее, сортируем результаты по убыванию total\_amount с использованием ключевого слова ORDER BY. Наконец, с помощью ключевого слова LIMIT ограничиваем результаты только первыми 10 записями.

Запрос 3:

Описание запроса:

В этом запросе также используется ключевое слово SELECT для выбора данных. Мы выбираем количество записей в столбце book\_date и сами даты (book\_date). Затем, с помощью ключевого слова FROM, указываем таблицу bookings, из которой выполняется выборка. Словом GROUP указываем группировку строк по полю book\_date. Далее, с помощью ключевого слова HAVING фильтруем результаты, оставляя только даты, в которых количество записей находится в диапазоне от 4 до 6. Словом ORDER BY сортируем результаты по убыванию book\_date. И, наконец, с помощью ключевого слова LIMIT ограничиваем результаты только первыми 10 датами.

Запрос 4:

Описание запроса:

В данном запросе используется ключевое слово SELECT для выбора данных. Мы выбираем столбец book\_date и применяем оконную функцию, которая вычисляет максимальное значение total\_amount для каждой даты (book\_date) без изменения порядка строк. Затем, с помощью ключевого слова FROM, указываем таблицу bookings, из которой выполняется выборка. Словом ORDER BY сортируем результаты по book\_date. Наконец, с помощью ключевого слова LIMIT ограничиваем результаты только первыми 10 записями, включая book\_date и максимальное total\_amount для каждой даты.

1. Из документации PostgrePro: JOIN - это оператор, который объединяет элементы списка FROM (таблицы) в определенном порядке. LEFT OUTER JOIN возвращает все строки из левой таблицы, дополненные значениями NULL из правой. RIGHT OUTER JOIN - аналогично, но с обратными ролями таблиц. FULL OUTER JOIN возвращает все строки и дополняет их значениями NULL из другой таблицы.
2. GROUP BY используется для группировки данных в SQL, часто с агрегатными функциями (COUNT, SUM, MAX, MIN, AVG). Оператор HAVING фильтрует группы.
3. Для удаления дубликатов используется оператор DISTINCT.
4. Для сортировки данных используется ORDER BY с ASC (по возрастанию) и DESC (по убыванию).
5. агрегатные функции в SQL игнорируют NULL значения по умолчанию
6. В SQL выполняются традиционные операции над множествами: объединение, пересечение, разность и декартово произведение.
7. См. ответ выше.
8. COUNT(\*) подсчитывает все записи в таблице, COUNT(field) - записи в столбце "field".
9. Для подсчета уникальных значений в столбце используется COUNT(DISTINCT field).
10. Предикат IS NULL используется для поиска пустых значений.
11. Для сравнения строк с образцом используется предикат LIKE с символами \_ (один символ) и % (любое количество символов).
12. Ограничение количества выводимых строк выполняется с помощью LIMIT. Если указано одно значение, выводится определенное количество строк; если два значения, выводится диапазон строк.
13. В SQL таблица - это физическое представление отношения, состоящее из строк и столбцов. Она имеет имя и структуру, определенные с помощью оператора CREATE TABLE. Одна таблица может представлять несколько отношений, и наоборот. Таблицы могут иметь ограничения целостности и индексы для улучшения производительности.
14. –
15. –
16. Для округления значения столбца используется ROUND(field, 3) (округление до 3 знаков после запятой).
17. Если требуется округлить значение без указания десятичных знаков, используется ROUND(field).
18. -
19. Для форматирования числа с разделителями тысяч используется функция FORMAT(). Например, FORMAT(field, 'd', 'en-US') форматирует число с разделителями тысяч в английском стиле.
20. –