



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 "Программное обеспечение ЭВМ и информационные технологии"

КУРСОВАЯ РАБОТА

НА ТЕМУ:

«Сложные SQL-запросы»

Студент ИУ7-53Б(В)
(Группа)

(Подпись, дата)

Д.П. Косаревский
(И.О.Фамилия)

Преподаватель

(Подпись, дата)

А.А. Павлюк
(И.О.Фамилия)

Москва
2021 г.

Оглавление

Введение	3
1. Аналитический раздел	4
1.1. Постановка задачи	4
1.2. PostgreSQL	6
2. Конструкторский раздел	7
2.1. Проектирование БД	7
2.2. Проектирование веб-интерфейса	9
3. Технологический раздел	10
3.1. Сложные SQL-запросы	10
Заключение	12
Список использованных источников информации	13

Введение

Целью курсовой работы является реализация сложных запросов SQL в созданной реляционной базе данных.

Реляционная модель первой приходит на ум большинству разработчиков с опытом в области баз данных. Реляционные системы управления базами данных (РСУБД) основаны на теории множеств, в основе их реализации лежат двумерные таблицы, состоящие из строк и столбцов. Канонический способ взаимодействия с РСУБД – написание запросов на языке Structured Query Language (SQL). Значения данных типизированы, это могут быть числа, строки, даты, неструктурированные двоичные объекты (BLOB) и т. п. Тип данных контролируется системой. Существенно, что благодаря математическим основаниям реляционной модели (теории множеств) исходные таблицы можно соединять и трансформировать в новые, более сложные. Существует немало реляционных СУБД с открытым исходным кодом – MySQL, H2, HSQLDB, SQLite и многие другие, так что выбрать есть из чего. В данной работе будет использована СУБД PostgreSQL.

Запрос – это важнейший инструмент для извлечения информации из одной или нескольких таблиц БД. Посредством запроса можно вносить изменения в саму БД. Запрос может служить источником данных для форм, отчетов и страниц доступа к данным. Его результатом является новая таблица, которая может быть просмотрена, проанализирована, а затем сохранена или не сохранена.

1. Аналитический раздел

1.1. Постановка задачи

Для возможности реализации запросов была инициализирована тестовая база данных с использованием СУБД PostgreSQL.

Так как в работе будут использованы сложные запросы, важно понимать виды таковых запросов.

По видам запросы SQL чаще всего делятся на:

- запросы, предназначенные для работы со структурой данных - для создания, описания и модификации БД;
- запросы, используемые непосредственно в работе с данными, с помощью которых можно добавлять, обновлять, сохранять и удалять данные;
- запросы, применяемые для предоставления или отмены прав доступа к БД;

В свою очередь, каждый из видов SQL-запросов подразделяется на типы:

- команды, работающие со структурой БД. К ним относятся CREATE - «создать» (например, CREATE TABLE (создать таблицу), CREATE USER (создать пользователя)), ALTER - «модифицировать» (этот запрос используется при внесении изменений в саму БД или в ее часть), DROP - «удалить» (также относятся к БД и ее частям);
- команды, работающие с данными. К наиболее востребованным запросам относятся: SELECT (выборка данных), INSERT (вставка

новых данных), UPDATE (обновление данных), DELETE (удаление данных), MERGE (слияние данных);

- команды, работающие с правами доступа. В их список входят GRANT - разрешение пользователю на проведение определенных операций с БД или данными; REVOKE – отзыв выданного разрешения; DENY – установка запрета, имеющего приоритет над разрешением.

При составлении SQL-запроса для работы с базами данных в СУБД (MySQL, Microsoft SQL Server, PostgreSQL) вводятся следующие параметры отбора:

- названия таблиц, из которых необходимо извлечь данные;
- поля, значения которых требуется вернуть к исходным после внесения изменений в БД;
- связи между таблицами;
- условия выборки;
- вспомогательные критерии отбора (ограничения, способы представления информации, тип сортировки).

1.2. PostgreSQL

Испытанная в боях СУБД PostgreSQL – одна из самых старых и надежных. Совместимая со стандартом SQL, она покажется знакомой любому, кто раньше работал с реляционными базами данных, и послужит эталоном для сравнения с другими базами.

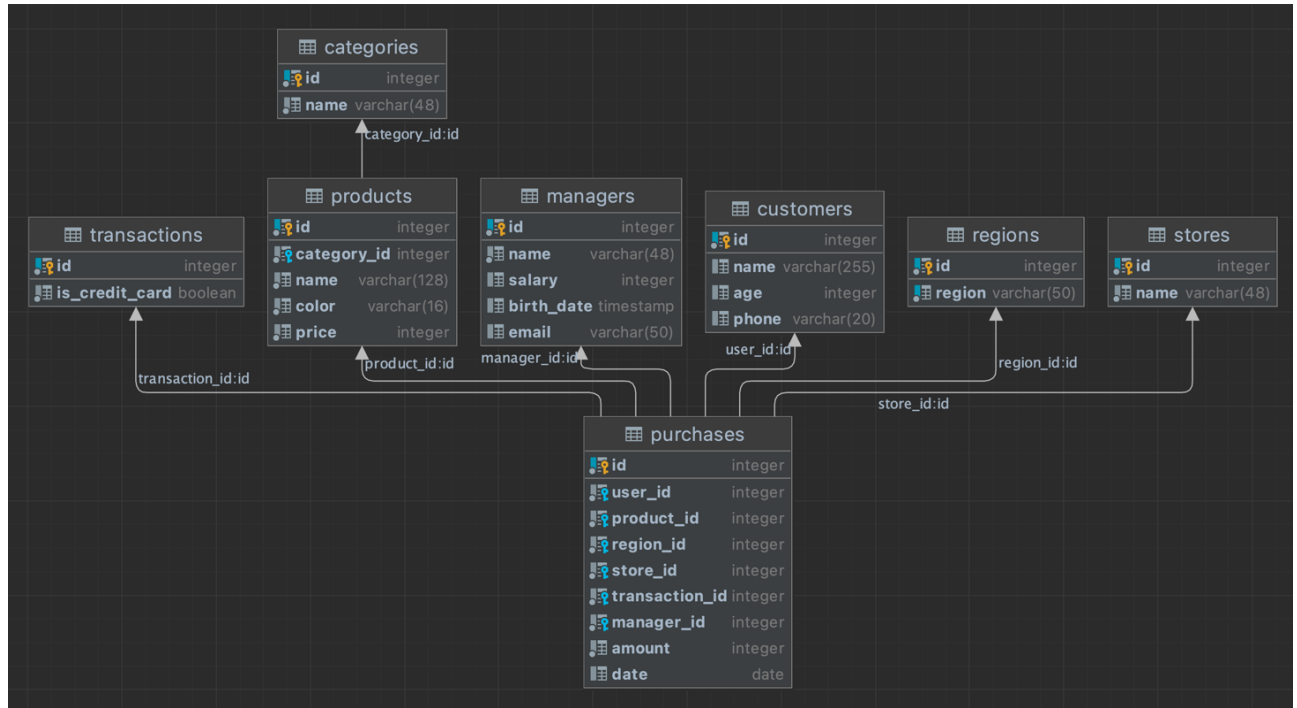
PostgreSQL – молоток в мире баз данных. Ее хорошо знают, она легко доступна, стабильна и при должном старании и умении способна решать на удивление разнообразные задачи. Нельзя рассчитывать стать опытным строителем, не овладев этим самым распространенным инструментом.

PostgreSQL – реляционная система управления базами данных, то есть основана на теории множеств, реализована в виде двумерных таблиц, где данные хранятся по строкам, и строго контролирует типы столбцов. Несмотря на растущий интерес к новым тенденциям в области баз данных, реляционный стиль является самым популярным и, вероятно, останется таким еще довольно долго.

Преобладание реляционных СУБД объясняется не только встроенным в них обширным инструментарием (триггеры, хранимые процедуры, развитые индексы), безопасностью данных (благодаря свойствам транзакционности ACID), количеством специалистов (многие программисты говорят и думают в реляционных терминах), но и гибкостью формулирования запросов. В отличие от некоторых других хранилищ данных, не требуется заранее планировать, как будут использоваться данные. Если реляционная схема нормализована, то можно предъявлять практически произвольные запросы. PostgreSQL – прекрасный пример системы с открытым исходным кодом, следующей традициям РСУБД.

2. Конструкторский раздел

2.1. Проектирование БД



На основе спроектированной UML-диаграммы была инициализирована БД содержащая информацию о покупках покупателей в различных регионах и различных магазинах, а также информацию о продуктах, менеджерах и транзакциях.

Для имитации работы с реальными данными были сгенерированы вымышленные данные с помощью специализированного ресурса, а также с применением реализованных функций. Функции, использованные для генерации случайных данных можно найти в открытом репозитории студента.

Скрипты инициализации БД и таблиц, а также наполнения таблиц данными можно найти репозитории.

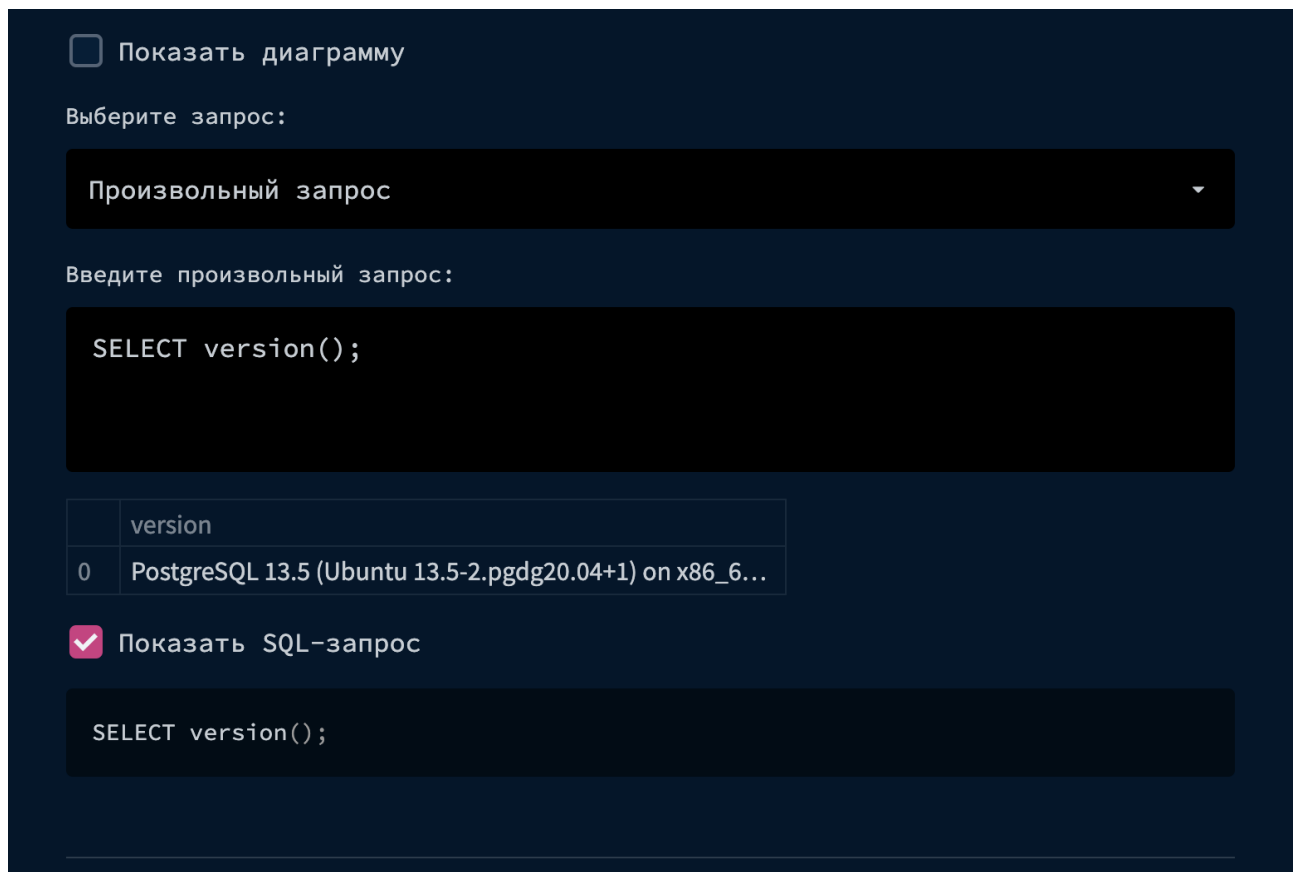
Пример функции для генерации случайной строки:

```
create or replace function random_string(length integer) returns text as
$$
declare
  chars text[] :=
    '{A,B,C,D,E,F,G,H,I,J,K,L,M,N,O,P,Q,R,S,T,U,V,W,X,Y,Z,a,b,c,d,e,f,g,h,i,j,k,l,m,n,o,p,q
    ,r,s,t,u,v,w,x,y,z}';
  result text := '';
  i integer := 0;
begin
  if length < 0 then
    raise exception 'Given length cannot be less than 0';
  end if;
  for i in 1..length loop
    result := result || chars[1+random()*(array_length(chars, 1)-1)];
  end loop;
  return result;
end;
$$ language plpgsql;
```


2.2. Проектирование веб-интерфейса

Для тестирования работы с базой данных был реализован веб-интерфейс. Для реализации был использован язык программирования Python.

Изображение веб-интерфейса:



The screenshot shows a web interface with a dark blue background. At the top, there is a checkbox labeled "Показать диаграмму" (Show diagram) which is currently unchecked. Below this, the text "Выберите запрос:" (Select query:) is followed by a dropdown menu showing "Произвольный запрос" (Arbitrary query). Underneath, the text "Введите произвольный запрос:" (Enter arbitrary query:) is followed by a text input field containing the SQL query "SELECT version();". Below the input field, a table displays the query results. The table has two columns: the first column contains the value "0", and the second column contains the text "PostgreSQL 13.5 (Ubuntu 13.5-2.pgdg20.04+1) on x86_6...". At the bottom, there is a checkbox labeled "Показать SQL-запрос" (Show SQL query) which is checked, followed by a text input field containing the same SQL query "SELECT version();".

☐ Показать диаграмму

Выберите запрос:

Произвольный запрос

Введите произвольный запрос:

```
SELECT version();
```

	version
0	PostgreSQL 13.5 (Ubuntu 13.5-2.pgdg20.04+1) on x86_6...

☒ Показать SQL-запрос

```
SELECT version();
```

3. Технологический раздел

3.1. Сложные SQL-запросы

В данной работе реализовано 11 (одиннадцать) сложных запросов, а также реализована возможность выполнять произвольные запросы через веб-интерфейс.

Примеры некоторых сложных запросов представлены ниже, остальные можно увидеть в веб-интерфейсе программы или в репозитории студента.

Запрос для отображения нарастающей суммы покупок (кумулятивная сумма):

```
SELECT date,  
       price,  
       SUM(price) OVER (ORDER BY date) as total  
FROM db_cp.products p  
     LEFT JOIN db_cp.purchases pu ON p.id = pu.product_id  
WHERE date is not null  
ORDER BY date;
```

Запрос для отображения покупателей и их покупок:

```
SELECT p.name as product_name,  
       p.price,  
       pu.date,  
       c.name as customer_name  
FROM db_cp.products p  
     LEFT JOIN db_cp.purchases pu ON p.id = pu.product_id  
     LEFT JOIN db_cp.customers c ON pu.user_id = c.id  
WHERE date is not null;
```

Запрос для отображения сумм покупок покупателей:

```
SELECT c.name      as customer_name,  
       SUM(p.price) as total_sales  
FROM db_cp.products p  
     LEFT JOIN db_cp.purchases pu ON p.id = pu.product_id  
     LEFT JOIN db_cp.customers c ON pu.user_id = c.id  
WHERE date is not null  
GROUP BY c.name  
ORDER BY total_sales desc;
```

Запрос для отображения покупок по месяцам:

```
SELECT SUM(price)                as total_sales,
       EXTRACT(month from date) as month
FROM db_cp.products p
     LEFT JOIN db_cp.purchases pu ON p.id = pu.product_id
     LEFT JOIN db_cp.customers c  ON pu.user_id = c.id
WHERE date is not null
GROUP BY EXTRACT(month from date)
ORDER BY EXTRACT(month from date);
```

Запрос для отображения дней недели с максимальными продажами:

```
WITH cte as (SELECT extract(isodow from pu.date) day_of_week,
                    SUM(p.price) as total_sales
              FROM db_cp.products p
                   LEFT JOIN db_cp.purchases pu ON p.id = pu.product_id
              WHERE date is not null
              GROUP BY day_of_week
              ORDER BY total_sales desc)
SELECT total_sales,
       CASE
         WHEN day_of_week = 1 THEN 'Понедельник'
         WHEN day_of_week = 2 THEN 'Вторник'
         WHEN day_of_week = 3 THEN 'Среда'
         WHEN day_of_week = 4 THEN 'Четверг'
         WHEN day_of_week = 5 THEN 'Пятница'
         WHEN day_of_week = 6 THEN 'Суббота'
         WHEN day_of_week = 7 THEN 'Воскресенье'
         ELSE 'Неизвестный день'
       END
FROM cte;
```

Запрос для отображения 20-ти лучших регионов по продажам:

```
WITH regional_sales as (
  SELECT region_id,
         SUM(amount) as total_sales
  FROM db_cp.purchases
  GROUP BY region_id
),
top_regions as (
  SELECT region_id
  FROM regional_sales
  ORDER BY total_sales desc
  LIMIT 20
)
SELECT region,
       region_id,
       SUM(amount) as product_sales
FROM db_cp.purchases pu
     LEFT JOIN db_cp.regions r ON pu.region_id = r.id
     LEFT JOIN db_cp.products p ON pu.product_id = p.id
WHERE region_id IN (SELECT region_id FROM top_regions)
GROUP BY region, region_id;
```

Заключение

В результате проведённой работы была достигнута поставленная цель:

- Инициализирована база данных, наполненная тестовыми вымышленными данными.
- Реализованы сложные запросы, демонстрирующие удивительный мир работы с данными.

В работе была использована СУБД PostgreSQL версии 13.5 развёрнутая на сервере (Ubuntu 13.5-2.pgdg20.04+1) on x86_64-pc-linux-gnu, compiled by gcc (Ubuntu 9.3.0-17ubuntu1~20.04) 9.3.0, 64-bit.

Для реализации веб-интерфейса использован ЯП Python версии 3.9.7, а также библиотеки streamlit и pandas.

Работающую программу можно увидеть и протестировать через web-интерфейс по следующей ссылке:

https://share.streamlit.io/dkosarevsky/db_cp/app.py

Исходный код программы и запросов находится в открытом репозитории по ссылке: https://github.com/dKosarevsky/db_cp

Список использованных источников информации

1. Эрик Редмонд, Джим. Р. Уилсон. Семь баз данных за семь недель. Введение в современные базы данных и идеологию NoSQL. Под редакцией Жаклин Картер, 2018
2. Документация Streamlit [Электронный ресурс].
URL: <https://docs.streamlit.io/knowledge-base/tutorials/databases>
3. Понимание SQL (Understanding SQL) Мартин Грабер (Martin Gruber).
[Электронный ресурс].
URL: https://www.sql.ru/docs/sql/u_sql/
4. PostgreSQL 13.5 Documentation. [Электронный ресурс].
URL: <https://www.postgresql.org/docs/13/>
5. Генерация тестовых данных. [Электронный ресурс].
URL: <https://generatedata.com/>
6. Павлюк А.А. Лекции по базам данных. МГТУ им. Баумана