ORES
Научно-практический электронный журнал
Оригинальные исследования (ОРИС), *ISSN 2222-0402*
www.ores.su
original-research.ru

СОЗДАНИЕ ПРЕДСТАВЛЕНИЙ В POSTGRESQL

Свищёв Андрей Владимирович
Старший преподаватель кафедры практической и прикладной информатики
МИРЭА-Российский технологический университет (РТУ МИРЭА), г. Москва.

Беликов Илья Владиславович
Студент магистратуры, 1 курс
МИРЭА-Российский технологический университет (РТУ МИРЭА), г. Москва

Аннотация

В данной статье рассматривается создание представлений в PostgreSQL. Проанализированы достоинства и недостатки при работе с ними, которые могут быть использованы в разработке приложений.

Ключевые слова: PostgreSQL, запросы, ПО, разработка, представления.

MODERN EPISTEMOLOGY: FEATURES AND PROBLEMS

Andrey V. Svishchev
Senior Lecturer at the Department of Practical and Applied Informatics
MIREA - Russian Technological University (RTU MIREA), Moscow.

Ilya V. Belikov
Master's student, 1st year
MIREA - Russian Technological University (RTU MIREA), Moscow

ABSTRACT

It is concluded that the problems of epistemology and currently does not lose its relevance. This article discusses creating views in PostgreSQL. The advantages and disadvantages of working with them, which can be used in application development, are analyzed.

Keywords: PostgreSQL, queries, software, development, views.

Представления (views) в PostgreSQL и других СУБД играют важную роль в организации баз данных и предоставляют несколько преимуществ:

1. Абстракция сложных запросов: любые view дают возможность абстрагироваться от SQL-запросов. Для того чтобы создать сложный запрос, включающий, к примеру работу с разными таблицами, вы можете создать

представление, которое инкапсулирует логику запроса [1]. Это делает код более читаемым, что облегчает как разработку, так и дальнейшую поддержку проекта.

2. Безопасность: еще одним преимуществом будет то, что при помощи представлений мы модем ограничивать доступ к данным, предоставляя пользователям доступ только к представлению. Таким образом скрываются сложные части вашей БД или любые другие данные. Это особенно полезно при управлении правами доступа.
3. Создание виртуальных таблиц: view дают способ создания виртуальных таблиц на основе данных из любого множества таблиц. Таким образом можно создать нечто, что будет включать в себя огромное множество данных из разных уголков БД, что может быть полезно при разработке [2].
4. Обновление базы данных: представления очень сильно упрощают процесс обновления БД, представим, что структура бд изменяется, тогда представление применяется в приложении. Вы можете обновить представление, чтобы оно продолжало работать с новой структурой, не затрагивая код приложения.
5. Использование в качестве заглушек: представления часто применяются, как логические заглушки для обеспечения совместимости с изменениями в базе данных [3]. Например, если вы изменяете структуру таблицы, вы можете создать представление с прежней структурой, чтобы не нарушать работу существующих запросов и приложений.
6. Повышение производительности: зачастую view могут повысить время обработки запроса. Представление часто создаются так, что они содержат результаты сложного запросов. Именно поэтому запросы приложений могут обращаться к этому представлению. Получается избежать необходимости повторного выполнения сложных запросов.

В PostgreSQL реализация представлений является довольно частая и не сложная зачада. К примеру, создание представления в PostgreSQL, в данном примере my_view объединяет данные из таблиц users и orders, предоставляя их в виде виртуальной таблицы с колонками user_id, username, order_number и order_date:

```
-- Пример представления, объединяющего данные из двух таблиц
CREATE VIEW my_view AS
SELECT
   users.id AS user_id,
   users.username,
   orders.order_number,
   orders.order_date
FROM
   users
JOIN
   orders ON users.id = orders.user_id;
```

Если простых представлений недостаточно для проекта, то можно создавать параметризованные представления (parametrized views). В PostgreSQL они обеспечивают гибкость и динамичность при работе с данными в базе данных [4]. Их использование может быть полезным, они позволяют создавать динамические отчеты и запросы. Вы можете использовать параметры для фильтрации данных, выбирая только те записи, которые соответствуют заданным критериям. Это особенно полезно, если вам нужно извлекать данные с различными условиями, не создавая каждый раз новое представление. Так же параметризованные представления позволяют переиспользовать одно и то же представление с различными параметрами. Например, если у вас есть запрос для выборки данных за определенный период времени, вы можете создать параметризованное представление и использовать его для различных временных интервалов без необходимости создания дополнительных представлений [5]. Самым главным достоинством будет, что в некоторых случаях могут улучшить производительность. Если запросы к базе данных часто выполняются с различными наборами параметров, параметризованные представления могут уменьшить накладные расходы на обработку запросов.

В PostgreSQL также можно создавать параметризованные представления, используя функции и «WITH»:

```
-- Пример параметризованного представления
CREATE OR REPLACE FUNCTION get_orders_by_user(user_id INT)
RETURNS TABLE (
   order_number INT,
   order_date DATE
) AS $$
BEGIN
   RETURN QUERY
   SELECT
      order_number,
      order_date
   FROM
      orders
   WHERE
      user_id = get_orders_by_user.user_id;
END;
$$ LANGUAGE plpgsql;

-- Использование параметризованного представления
SELECT * FROM get_orders_by_user(1);
```

Самое важное при работе с представлениями, это помнить, что представления не хранят данные физически, они предоставляют виртуальный доступ к данным, определенным в исходных запросах.

Список литературы:

1. Токмаков, Г. П. Информационное и лингвистическое обеспечение локальных и распределенных автоматизированных систем: учебное пособие / Г. П. Токмаков. — Ульяновск: УлГТУ, 2022. — 333 с. — ISBN 978-5-9795-2230-2.

2. Инструментальное программное обеспечение разработки и проектирования информационных систем: учебное пособие / А. А. Куликов, В. Т. Матчин, А. В. Синицын, В. В. Литвинов. — Москва: РТУ МИРЭА, 2022.
3. Волков, М. Ю. Разработка серверных частей интернет-ресурсов: учебное пособие / М. Ю. Волков, В. В. Литвинов, А. А. Лобанов. — Москва: РТУ МИРЭА, 2021. — 188 с.
4. Архитектурные решения информационных систем / А. И. Водяхо, Л. С. Выговский, В. А. Дубенецкий, В. В. Цехановский. — 3-е изд., стер. — Санкт-Петербург: Лань, 2023. — 356 с. — ISBN 978-5-507-46063-2.
5. Токмаков, Г. П. Основы XML-технологий: учебное пособие / Г. П. Токмаков. — Ульяновск: УлГТУ, 2017. — 229 с. — ISBN 978-5-9795-1701-8.

References:

1. Tokmakov, G. P. Information and linguistic support of local and distributed automated systems: textbook / G. P. Tokmakov. - Ulyanovsk: UlSTU, 2022. - 333 p. — ISBN 978-5-9795-2230-2.
2. Tool software for the development and design of information systems: textbook / A. A. Kulikov, V. T. Matchin, A. V. Sinitsyn, V. V. Litvinov. — Moscow: RTU MIREA, 2022.
3. Volkov, M. Yu. Development of server parts of Internet resources: textbook / M. Yu. Volkov, V. V. Litvinov, A. A. Lobanov. — Moscow: RTU MIREA, 2021. — 188 p.
4. Architectural solutions of information systems / A. I. Vodyakho, L. S. Vygovsky, V. A. Dubenetsky, V. V. Tsekhanovsky. — 3rd ed., erased. - St. Petersburg: Lan, 2023. - 356 p. — ISBN 978-5-507-46063-2.
5. Tokmakov, G. P. Fundamentals of XML technologies: textbook / G. P. Tokmakov. - Ulyanovsk: Ulyanovsk State Technical University, 2017. - 229 p. — ISBN 978-5-9795-1701-8.