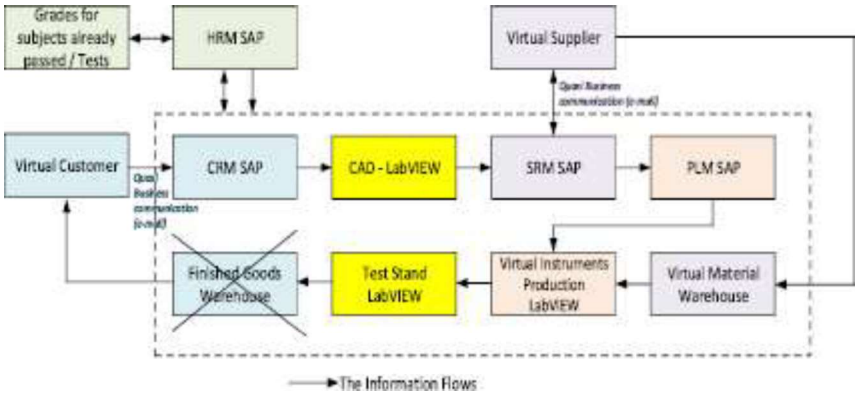

Рис. 2. Виртуализация приборостроительного предприятия

Восьмой этап – отгрузка готовой продукции заказчику (связь между блоками «Finished Goods Warehouse» и «Customer») и передача соответствующих документов о приемке в подразделение, ответственное за взаимодействие с клиентами (связь между блоками «Customer» и «Customer Relationship Management»).

Для создания условий обучения, которые максимально приближены к реальным условиям работы на современном приборостроительном предприятия, и предложена схема работы виртуальной организации (рис. 2).

Мы полагаем, что данная схема позволяет обеспечить внедрение принципов так называемого объектно-ориентированного подхода непосредственно к обучению, причем с ориентацией на заблаговременную адаптацию будущего специалиста к выполнению своих трудовых обязанностей.

**Список использованных источников**

1. Захарова И.Г. Информационные технологии в образовании : учебник для
2. студ. учреждений высш. проф. образования / И.Г.Захарова. —8-е изд., перераб. и доп. — М. : Издательский центр «Академия», 2013 — 208 с. — (Сер. Бакалавриат).ISBN 978-5-7695-9538-7
3. Шайхутдинов Д.В., Горбатенко Н.И., Коломиец А.В. Виртуальное предприятие – новый подход к организации учебного процесса в ВУЗе // Международный журнал экспериментального образования. – 2017. – № 10. – С. 43-48; URL: http://expeducation.ru/ru/article/view?id=11770
4. Шестак В.П., Весна Е.Б., Платонов В.Н. Сетевое образование: лучшие отечественные и зарубежные практики // Современные проблемы науки и образования. – 2013. – № 6.;
5. URL: http://science-education.ru/ru/article/view?id=10981 (дата обращения: 08.10.2019).

20.23.17

## СОЗДАНИЕ ПОЛЬЗОВАТЕЛЬСКИХ ФУНКЦИЙ НА СЕРВЕРЕ БАЗЫ ДАННЫХ ДЛЯ УПРАВЛЕНИЯ НОРМАТИВНО-СПРАВОЧНОЙ ИНФОРМАЦИЕЙ ФЕДЕРАЛЬНОЙ ТАМОЖЕННОЙ СЛУЖБЫ

**Автор:** Карпов Максим Александрович, студент 4 курса государственного университета «Дубна», филиал «Протвино»

**Научный руководитель:** Кульман Татьяна Николаевна, к.т.н., доцент кафедры университета «Дубна» филиала «Протвино»

**Аннотация**

В данной работе рассматривается создание хранимых пользовательских функций в базе данных PostgreSQL. Вызывая такие функции, с некоторыми параметрами, можно эффективно управлять данными в таблицах и самой базой данных. В разработке

программного обеспечения используется трёхзвенная технология, поэтому создание таких функций на сервере базы данных является актуальной темой.

**Annotation**

This paper discusses the creation of stored user-defined functions in a PostgreSQL database. By calling such functions, with some parameters, you can effectively manage data in tables and the database itself. Three-tier technology is used in software development; therefore, the creation of such functions on a database server is an urgent topic.

С появлением магнитных дисков началась история систем управления данными во внешней памяти. До этого каждая прикладная программа, которой требовалось хранить данные во внешней памяти, сама определяла расположение каждой порции данных на магнитной ленте или барабане и выполняла обмены между оперативной и внешней памятью с помощью программно-аппаратных средств низкого уровня (машинных команд или вызовов соответствующих программ операционной системы). Такой режим работы не позволяет или очень затрудняет поддержание на одном внешнем носителе нескольких архивов долговременно хранимой информации. Кроме того, каждой прикладной программе приходилось решать проблемы именования частей данных и структуризации данных во внешней памяти.

Пользовательские функции являются большим плюсом и их актуальность нельзя не отметить в разработке различных программных обеспечений, в основе которых лежат клиент-серверные принципы. При модернизации каких-либо хранимых функций пользователь не заметит каких-либо изменений в программе, но приложение изменится.

База данных (БД) – это набор, совокупность файлов, в которых находится информация. Программная система (приложение), обеспечивающая работу с базой данных (файлами данных) называется системой управления базой данных (СУБД). Информация, хранимая в базе данных, является структурированной и постоянно хранимой. Постоянность означает, что данные не уничтожаются по завершении программы или пользовательского сеанса, в котором они были созданы. В зависимости от расположения программы, которая использует данные, и самих данных, а также от способа разделения данных между несколькими пользователями различают локальные и удаленные базы данных.

PostgreSQL – свободная объектно-реляционная система управления базами данных.

PostgreSQL позволяет разрабатывать пользовательские функции не только на SQL и C, но и на других языках. Эти языки в целом называются процедурными языками (PL, Procedural Language). Если функция написана на процедурном языке, сервер баз данных сам по себе не знает, как интерпретировать её исходный текст. Вместо этого он передаёт эту задачу специальному обработчику, понимающему данный язык. Обработчик может либо выполнить всю работу по разбору, синтаксическому анализу, выполнению кода и т. д., либо действовать как «прослойка» между PostgreSQL и внешним исполнителем языка программирования. Сам обработчик представляет собой функцию на языке C, скомпилированную в виде разделяемого объекта и загружаемую по требованию, как и любая другая функция на C.

PL/pgSQL – это процедурный язык для СУБД PostgreSQL. Целью проектирования PL/pgSQL было создание загружаемого процедурного языка, который:

- используется для создания функций и триггеров;
- добавляет управляющие структуры к языку SQL;
- может выполнять сложные вычисления;
- наследует все пользовательские типы, функции и операторы;
- может быть определён как доверенный язык;

- прост в использовании;

Функции PL/pgSQL могут использоваться везде, где допустимы встроенные функции. Например, можно создать функции со сложными вычислениями и условной логикой, а затем использовать их при определении операторов или в индексных выражениях.

PostgreSQL и большинство других СУБД используют SQL в качестве языка запросов. SQL хорошо переносим и прост в изучении. Однако каждый оператор SQL выполняется индивидуально на сервере базы данных. Это значит, что ваше клиентское приложение должно каждый запрос отправлять на сервер, ждать пока он будет обработан, получать результат, делать некоторые вычисления, затем отправлять последующие запросы на сервер. Всё это требует межпроцессного взаимодействия, а также несёт нагрузку на сеть, если клиент и сервер базы данных расположены на разных компьютерах.

PL/pgSQL позволяет сгруппировать блок вычислений и последовательность запросов внутри сервера базы данных, таким образом, мы получаем силу процедурного языка и простоту использования SQL при значительной экономии накладных расходов на клиент-серверное взаимодействие.

В результате это приводит к значительному увеличению производительности по сравнению с приложением, которое не использует хранимых функций. PL/pgSQL позволяет использовать все типы данных, операторы и функции SQL.

Пользовательские функции во многом напоминают хранимые процедуры и представляют собой упорядоченное множество операторов T-SQL, которые заранее оптимизированы, откомпилированы и могут быть вызваны для выполнения работы в виде единого модуля.

Основное различие между пользовательскими функциями и хранимыми процедурами состоит в том, как в них осуществляется возврат полученных результатов.

В пользовательских функциях тип данных возвращаемого значения не ограничивается только целочисленным типом, как при использовании хранимых процедур. Значения, возвращаемые пользовательской функцией, могут относиться почти к любому типу данных PostgreSQL Server.

Функции на PL/pgSQL могут принимать в качестве аргументов все поддерживаемые сервером скалярные типы данных или массивы и возвращать в качестве результата любой из этих типов. Они могут принимать и возвращать именованные составные типы (строковый тип).

При отсутствии полезного возвращаемого значения функция на PL/pgSQL может возвращать void. Функции на PL/pgSQL можно объявить с выходными параметрами вместо явного заданного типа возвращаемого значения. Это не добавляет никаких фундаментальных возможностей языку, но часто бывает удобно, особенно для возвращения нескольких значений.

Функции на PL/pgSQL могут возвращать «множества» (или таблицы) любого типа, которые могут быть возвращены в виде одного объекта. Такие функции генерируют вывод, выполняя команду RETURN NEXT для каждого элемента результирующего набора или RETURN QUERY для вывода результата запроса.

Команда CREATE FUNCTION определяет новую функцию. CREATE OR REPLACE FUNCTION создаёт новую функцию, либо заменяет определение уже существующей. Чтобы определить функцию, необходимо иметь право USAGE для соответствующего языка.

Данный проект реализуется на операционной системе Ubuntu. Было принято решение, что версия будет PostgreSQL-10.10, так как данная версия одна из последних и наиболее стабильна. Серверная и клиентская часть написаны на языке программирования JS с использованием JSON.

В проекте используется трёхзвенная технология (рис. 1). Поэтому необходимо установить сервер PostgreSQL. После успешной установки можем приступать к написанию пользовательских функций.

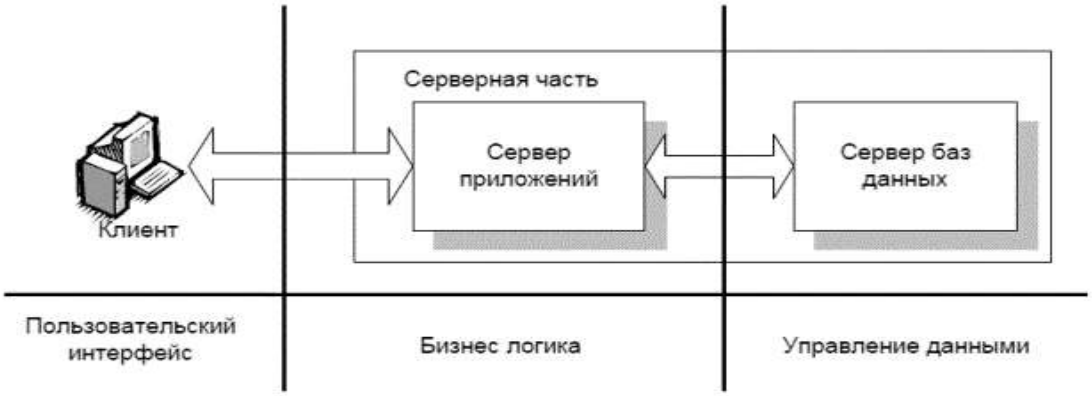


*Рисунок 4. Трёхзвенная технология*

Язык программирования JS, используемый на клиенте, понимает формат JSON и быстро работает с ним, поэтому данные, которые мы должны отправить на клиент, формируем в JSON формат.

Когда с клиента приходит запрос, сервер приложения вызывает определённые функции, находящиеся на сервере базы данных (рис. 3). После того, как сервер БД сформировал свой ответ в виде JSON-ов, он посылает его серверу приложения, а тот в свою очередь отправляет набор JSON-ов на клиент, где и происходит конечная обработка данных.

Одна из многих пользовательских функций представлена на рис. 2

```
-- Удаление пользователя p_login_del_user с паролем  p_password_del_user пользователем p_id_user
-- Пример:
-- select deleteUser(1, 'asd', 'asd123');
create or replace function pgnsi_comm.deleteUser(p_id_user integer, p_login_del_user varchar,
                                                 p_password_del_user varchar)
    returns void
    security definer as
$$
declare
    _id_user   int;
    _name_user varchar(50);
    _operation text;
begin
    if exists(select 1 from pgnsi_comm.users where login = p_login_del_user and password = p_password_del_user) then
        _name_user = (select login from pgnsi_comm.users where id_user = p_id_user);
        _operation = 'Пользователь ' || _name_user || ' удалил пользователя ' || p_login_del_user || ' с паролем ' ||
                     p_password_del_user;

        delete
        from pgnsi_comm.users
        where login = p_login_del_user
          and password = p_password_del_user returning id_user into _id_user;
        insert into pgnsi_comm.users_log (id_user, login, password, id_row_in_parent, operation, id_user_changed)
        values (_id_user, p_login_del_user, p_password_del_user, _id_user, _operation, p_id_user);

    end if;
end
$$ language plpgsql;
```

*Рисунок 5. Пользовательская функция на удаление пользователя*

```
// получение списка справочников
router.get('/:type/catalogs', async (req, res) => {
    const { type } = req.params;
    const query = `SELECT * FROM getDictList(${userId}, '${type}')`;
    console.log(query);
    const { rows } = await db.query(query);

    const data = rows.map(({ getdictlist }) => ({ ...getdictlist }));
    res.status(200).send(data);
});
```

*Рисунок 6. Запрос к БД и отправка данных на клиент*

**Вывод**

В работе рассмотрено создание пользовательских функций. Определено, в каком формате данные с сервера БД будут передаваться на сервер приложения и на клиент. Рассмотрено, как происходит запрос от клиента к серверу приложения и к серверу БД. Изучен процедурный язык PL/pgSQL с помощью которого написано большинство функций в данном программном обеспечении.

**Библиографический список**


1. Пирогов, В.Ю. Информационные системы и базы данных: организация и проектирование: Учебное пособие / В.Ю. Пирогов. - СПб.: БХВ-Петербург, 2009. - 528 с.
2. Преснякова, Г.В. Проектирование интегрированных реляционных баз данных: Учебное пособие / Г.В. Преснякова. - М.: КДУ , 2007. - 224 с.
3. Ригс, Саймон Администрирование PostgreSQL 9. Книга рецептов / Саймон Ригс , Ханну Кросинг. - М.: ДМК Пресс, 2015. - 364 с.



06.52.17


## ОБ ИНТЕРНЕТ-ИНДУСТРИИ СОВРЕМЕННОГО ОБЩЕСТВА


**Авторы:** Князева Наежда Михайловна, учащаяся 1 курса Серпуховского городского открытого колледжа

Щербаков Антон Павлович, учащаяся 2 курса Серпуховского городского открытого колледжа

**Научный руководитель**: Турицина Людмила Георгиевна, преподаватель Серпуховского городского открытого колледжа



**Аннотация**

В статье исследуется масштаб развития Интернет-технологий в различных отраслях экономической деятельности, оценивается методика прямого и косвенного их влияния на экономику, необходимость оценки масштабов с целью воздействия государства в фискальных интересах

**Annotetion.**

The article examines the scale of development of Internet technologies in various sectors of economic activity, assesses the method of direct and indirect impact on the economy, the need to assess the scale of the impact of the state in the fiscal interests



**Ключевые слова:** Интернет-процессы, интернет-потребители, производительность, интернет-аудитории, бизнес-процессы, ВВП, мировая экономика.

**Keywords:** Internet processes, Internet consumers, productivity, Internet audiences, business processes, GDP, world economy.