

**Филиал федерального государственного бюджетного образовательного учреждения
высшего образования
«Национальный исследовательский университет «МЭИ»
в г. Смоленске**

**МЕТОДИЧЕСКИЕ МАТЕРИАЛЫ ДЛЯ ОБЕСПЕЧЕНИЯ
ОБРАЗОВАТЕЛЬНОГО ПРОЦЕССА**

Направление подготовки: **09.03.03 «Прикладная информатика»**

Профиль: **«Прикладная информатика в топливно-энергетическом комплексе»**

Уровень высшего образования: **бакалавриат**

Нормативный срок обучения: **4 года**

Форма обучения: **очная**

Год набора: **2023**

Методические материалы составил:

канд. техн. наук, доцент кафедры
информационных технологий в экономике и управлении _____ В.П. Фомченков



«20» _____ января _____ 2023 г.

Заведующий кафедрой информационных технологий в экономике и управлении:

_____ д-р техн. наук, профессор М.И. Дли
подпись _____ ФИО

«08» февраля 2023 г.

МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ЛЕКЦИЙ ПО ДИСЦИПЛИНЕ

Комплект слайдов к лекциям

1 | 1 | Понятие и разновидности автоматизированных информационных систем

**ИНФОРМАЦИОННАЯ СИСТЕМА (ИС)
 INFORMATION SYSTEM (IS)**

комплекс средств, обеспечивающих сбор, хранение и обработку информации в целях поддержки какого-либо вида деятельности.

**АВТОМАТИЗИРОВАННАЯ
 ИНФОРМАЦИОННАЯ СИСТЕМА (АИС)
 AUTOMATED INFORMATION SYSTEM (AIS)**

информационная система, функционирующая на основе ЭВМ и других технических средств информатики.

2 | 2 | Понятие и разновидности автоматизированных информационных систем

ОСНОВНЫЕ ФУНКЦИИ ИС

- хранения,
- обработка,
- поиск,
- распространение, передача и предоставление информации

3 | 3 | Понятие и разновидности автоматизированных информационных систем

ДОКУМЕНТАЛЬНЫЕ СИСТЕМЫ

служат для работы с документами на естественном языке – монографиями, публикациями в периодика, диссертациями, авторефератами, текстами законодательных актов и т.д.

**ИНФОРМАЦИОННО-ПОИСКОВЫЕ СИСТЕМЫ (ИПС)
 INFORMATION RETRIEVAL SYSTEM** – предназначены для накопления и поиска по различным критериям документов, записанных на естественном языке.

4 | 9 | Понятие и разновидности автоматизированных информационных систем

ФАКТОГРАФИЧЕСКИЕ СИСТЕМЫ – оперируют фактичными сведениями, представленными в виде специального образом организованных совокупностей формализованных данных.

БАНК ДАННЫХ (DATA BANK) - автоматизированная информационная система централизованного хранения и коллективного использования данных.

АВТОМАТИЗИРОВАННОЕ РАБОЧЕЕ МЕСТО (АРМ) - индивидуальный комплекс технических и программных средств, предназначенный для автоматизации профессионального труда специалиста и обеспечивающий подготовку, редактирование, поиск и выдачу на экран и печать необходимых ему документов и данных.

5 | 10 | Понятие базы данных и СУБД, классификация СУБД

СОСТАВ БАНКА ДАННЫХ

- одна или несколько баз данных,
- справочник баз данных,
- СУБД,
- библиотеки запросов и прикладных программ,
- технические средства,
- персонал.

6 | 11 | Понятие базы данных и СУБД, классификация СУБД

**БАЗА ДАННЫХ (БД) DATABASE;
 DATA BASE (DB)** – совокупность связанных данных, организованных по определенным правилам, предусматривающим общие принципы описания, хранения и манипулирования, независимая от прикладных программ.

7 | 12 | Понятие базы данных и СУБД, классификация СУБД

**СИСТЕМЫ УПРАВЛЕНИЯ
 БАЗАМИ ДАННЫХ (СУБД)** – программное обеспечение, которое управляет доступом к базе данных.

Наиболее употребительными системами являются
 Oracle, DB2, Sybase, Informix, Ingres, Progress, Microsoft SQL, PostgreSQL, MySQL, Microsoft Access и др.

8 | 13 | Понятие базы данных и СУБД, классификация СУБД

Настольные (персональные) СУБД предназначены для решения сравнительно небольших задач (небольшой объем обрабатываемых данных, малое количество пользователей).

Имеют относительно упрощенную архитектуру, в частности:

- как правило, функционируют в режиме файл-сервер,
- поддерживают не все возможные функции СУБД (например, не ведется журнал транзакций, отсутствует возможность автоматического восстановления базы данных после сбоя и т. п.).

9 14 Понятие базы данных и СУБД, классификация СУБД

Настольные СУБД:

- dBase III – PLUS (фирма Ashton-Tate)
- Clipper (фирма Nantucket Inc.)
- FoxBase+ (фирма Fox Software)
- FoxPro (фирма Fox Software)
- Visual FoxPro (фирма Microsoft)
- PARADOX (фирма Borland International)
- Microsoft Access (фирма Microsoft)

10 15 Понятие базы данных и СУБД, классификация СУБД

Промышленные СУБД – СУБД ориентированные на создание АИС, оперирующих большими объемами информации с повышенным требованием безопасности.

- Oracle
- DB2
- MS SQL Server
- Sybase
- Informix
- Ingres
- Progress

Работают по принципу серверов баз данных.

Среди СУБД выделяют СУБД (условно говоря) промежуточные между промышленными и настольными

- PostgreSQL
- MySQL

11 16 Понятие базы данных и СУБД, классификация СУБД

Локальная база данных - база данных, размещенная на одном или нескольких носителях на одном компьютере.
 (Не путать с автономными и сетевыми СУБД, где речь идет о однопользовательском и многопользовательском режимах работы).

Распределенная база данных - совокупность баз данных, физически распределенная по взаимосвязанным ресурсам вычислительной сети и доступная для совместного использования.

12 17 Понятие базы данных и СУБД, классификация СУБД

Файл-серверные СУБД
 Файлы данных располагаются централизованно на файл-сервере. СУБД располагается на каждом клиентском компьютере (рабочей станции). Доступ СУБД к данным осуществляется через локальную сеть. Синхронизация (чтение) и обновление осуществляется посредством файловых операций.
 Microsoft Access, Paradox, dBase, FoxPro, Visual FoxPro.

Клиент-серверные
 Клиент-серверные СУБД располагаются на сервере вместе с БД и осуществляют доступ к БД непосредственно, в локальном режиме. Все клиентские запросы на обработку данных обрабатываются клиент-серверной СУБД централизованно.
 Oracle, Ingres, IBM DB2, Informix, MS SQL Server, Sybase Adaptive Server Enterprise, PostgreSQL, MySQL, LinType

Встраиваемые СУБД – СУБД, которые могут поставляться как составная часть некоторого программного продукта, не требуя процедуры самостоятельной установки.
 OpenEdge, SQLite, BerkeleyDB, Firebird Embedded, Microsoft SQL Server Compact

13 18 Модели данных

Модель данных определяет принцип (или порядок) организации (представления) данных в базе и характер связей между ними.

Реляционная база данных (Relational database) использует реляционную модель представления данных.
 Реляционная модель данных - логическая модель данных в виде (математическое выражение) набора отношений.

Объектно-реляционная (постреляционная) база данных.
 Объектно-реляционная модель данных (ОРМД) реализована с помощью реляционных таблиц, но включает объекты, являющиеся по сути объектами в объектно-ориентированной программировании.

В ОРМД используются также объектно-ориентированные компоненты, как пользовательские типы данных, индексы, полиморфизм, наследование, перераспределение методов и т.д. Объектно-реляционные СУБД продолжают использовать стандартный язык запросов для реляционных БД – SQL, но с объектными расширениями.

14 19 Модели данных

Иерархическая база данных использует иерархическую модель представления данных.
 Information Management System (IMS) фирмы IBM, первая версия появилась в 1965 г.

Иерархическая модель данных - логическая модель данных в виде древовидной структуры.

Терминологической основой для иерархической и сетевой моделей являются понятия:

- атрибут;
- агрегат;
- запись.

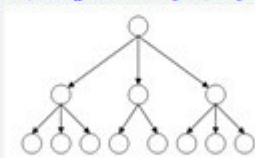
Атрибут (элемент данных) - наименьшая поименованная структурная единица данных.
 Поименованное множество атрибутов может образовывать агрегат данных.

Запись - составной агрегат, который не входит в состав других агрегатов.

15 20 Модели данных

В иерархической модели все записи, агрегаты и атрибуты базы данных образуют иерархический организованный набор - структуру, в которой все элементы связаны отношениями подчиненности, и при этом любой элемент может подчиняться только одному какому-нибудь другому элементу.

Такую форму зависимости удобно изображать с помощью древовидного графа (совокупности из точек и стрелок, которая связана и не имеет циклов).




16 21 Модели данных

Сетевая база данных использует сетевую модель представления данных.
 Integrated Database Management System (IDMS) компании Cullin Software Inc.

Сетевая модель данных - логическая модель данных в виде произвольного графа.

Сетевой подход к организации данных является расширением иерархического. В иерархической структуре запись-потомок должна иметь в точности одного предка, в сетевой структуре данных потомок может иметь любое число предков.

Сетевая БД состоит из набора записей и набора связей между этими записями.



17 22 Основные понятия и определения реляционных БД

РЕЛЯЦИОННАЯ БАЗА ДАННЫХ RELATIONAL DATABASE использует реляционную модель представления данных.

РЕЛЯЦИОННАЯ МОДЕЛЬ ДАННЫХ логическая модель данных в виде (изменяющегося во времени) набора отношений.

Данные в реляционной БД структурированы в таблицы. Собственно реляционная база данных представляет собой набор таблиц и связей между ними. Базе данных присваивается имя.

21 26 Индексы.

ИНДЕКС представляет собой таблицу, содержащую минимум два поля: номер записи в исходной таблице N_i и ее номер в упорядоченной последовательности N_j .

Вид индексов при упорядочивании в алфавитном порядке по полю *товар*:

Индекс <i>itovar</i>	
N_i	N_j
1	2
2	3
3	1

18 23 Основные понятия и определения реляционных БД

БАЗА ДАННЫХ FIRMA

1. ТАБЛИЦА SKLAD – содержит информацию об остатках товара на складе.
2. ТАБЛИЦА NAKLAD – содержит информацию о расходных операциях.

22 27 Индексы.

Механизм вывода записей по индексу *itovar*

19 24 Основные понятия и определения реляционных БД

ТАБЛИЦА SKLAD

Код товара	Товар	Фасовка	Цена	Остаток
<i>kodt</i>	<i>tovar</i>	<i>fасovka</i>	<i>цена</i>	<i>ostatok</i>
00108	Принтер HP 1200	шт.	750	9
00100	Дискеты Verbatim	кор.10шт.	82,6	125
00114	Монитор ViewSonic 17	шт.	4800	12

23 28 Связывание таблиц.

Ненормированная структура таблицы *naklad*

Номер наклад.	Дата продаж	Клиент	Код товара	Товар	Фасовка	Цена	Количество

Избыточные поля: сведения о наименовании товара, фасовка и цена.
 Эти данные есть в таблице *sklad*.

20 25 Основные понятия и определения реляционных БД

Основные понятия и определения реляционных БД

Термин	Синоним	Определение
Таблица	таблица	Двумерная таблица данных.
Связывание таблиц	связывание	Связывание таблиц, содержащих связанные данные. Каждое поле описывается своим именем (или именами) и типом данных, который ему присвоен.
Ключ	ключ	Специальная таблица, содержащая сведения о ключах, связанных с таблицей в реляционной БД. Каждый записи имеет номер записи, который присваивается ей в порядке ввода данных.
Связь	связь	Связь между таблицами (связи).
Связываемое поле	связываемое поле	Поле в одной из таблиц (или в таблице).
Данные	данные	Множество значений, имеющих, по крайней мере, значение для определенного атрибута определенного отношения.
Тип	тип	Набор из одного значения.
Типовой элемент	элемент	Элемент, принадлежащий множеству или типу данных.

24 29 Лекция 2. Обработка и хранение информации. Реляционные базы данных. Связывание таблиц.

Нормированная структура таблицы *naklad*

Ключевое поле	номер наклад	код товара	дата продажи	клиент	количество
<i>key</i>	<i>nomnakl</i>	<i>kodt</i>	<i>dat_pr</i>	<i>klent</i>	<i>kol_cho</i>
1	001	00100	05.10.99	ООО Метеор	6
2	001	00108	05.10.99	ООО Метеор	1
3	002	00106	12.10.99	ЧП Иванова	7
4	002	00104	12.10.99	ЧП Иванова	4
5	003	00104	20.11.99	АО Тройка	12
6	003	00108	20.11.99	АО Тройка	2
7	003	00103	20.11.99	АО Тройка	15

Избыточные поля заменены полем *кодт*.

1 **1** **Современные промышленные СУБД**



2 **2** **ORACLE Oracle Database**

Oracle (Oracle Corporation) — американская корпорация, второй по величине поставщик программного обеспечения (после Microsoft), крупнейший поставщик программного обеспечения для организаций, крупный поставщик серверного оборудования.

Количество основана в 1977 году. Ларри Деллоис — со-основатель, генеральный директор в период с 1977 года по 2014 год, крупнейший акционер (25 % по состоянию на 2014 год).

Подразделения корпорации, расположенные более чем в 145 странах. По состоянию на 2014 год насчитывает 122 тыс. сотрудников. Штаб-квартира корпорации расположена в США, в штате Калифорния, рядом с Сан-Франциско.

3 **3** **ORACLE Oracle Database**

Компания специализируется на выпуске систем управления базами данных, анализирует программное обеспечение и бизнес-приложений (ERP- и CRM-системы, специализированные отраслевые приложения).

Наиболее известный продукт компании — Oracle Database, который компания выпускает с момента своего основания. Последняя версия - Oracle Database 19c.



4 **4** **ORACLE Oracle Database**

Наибольшее распространение получила предыдущая версия - Oracle Database 12c.

Oracle Database 12c предлагает новую мультиверсионную архитектуру, которая упрощает быструю консолидацию большого числа баз данных и управление ими как облачными службами.


Предоставляет возможности обработки данных в памяти и беспроводную по эффективности аналитику.

Дополнительные инновации, реализованные в базе данных, обеспечивают эффективность, производительность, безопасность и доступность совершенно нового уровня.

5 **5** **ORACLE Oracle Database**

Oracle Database 12c поставляется в двух вариантах:

- Enterprise Edition
- Standard Edition Two.



Oracle Database Standard Edition Two

Лицензия на именованного пользователя (Named User Plus): Именованный пользователь — лицо, упомянутое в названии программы, установленной на одной или нескольких серверах, независимо от того, использует ли оно программу в данный момент времени или нет.

Максимальное количество лицензий для партии - 10 лицензий на каждый сервер.

Лицензия на именованного пользователя - 22 284,82 руб.

6 **6** **ORACLE Oracle Database**

Процессорная лицензия (Processor): Данный тип лицензирования применяется в том случае, когда нет возможности точно высчитать количество пользователей или их количество очень велико. С помощью этой схемы лицензируются все процессоры системы, на которой установлены или запускаются программы. Доступ к программам может предоставляться как внутренними пользователями организации, так и третьим лицам, что предполагает неограниченное число пользователей.

Максимально допустимое количество процессоров и ячеек (место под процессор) на сервере для данной редакции - 2.

Процессорная лицензия - 1 114 730,76 руб.

Developer — бесплатная интегрированная среда разработки программного обеспечения, разработанная корпорацией Oracle.

Предоставляет возможность для разработки на языках программирования Java, JavaScript, PHP, PHP, SQL, PL/SQL и на языках разметки HTML, XML.

7 **7** **Microsoft SQL Server**

Microsoft SQL Server — система управления реляционными базами данных, разработанная корпорацией Microsoft. Основой используемый язык запросов — Transact-SQL, создан совместно Microsoft и Sybase.


Microsoft SQL Server - это законченное предложение в области баз данных и анализа данных для быстрого создания масштабируемых решений электронной коммерции, бизнесприложений и хранилищ данных.

К настоящему времени разработано несколько версий систем: MS SQL Server 2000, MS SQL Server 2005, MS SQL Server 2008, Server 2012, Server 2014, Server 2016, Server 2017.

Последняя - MS SQL Server 2019

8 **8** **Microsoft SQL Server**

Достоинства SQL Server 2017



- Вы сами выбираете язык и платформу:** Поддержка современных технологий, поддержка как настраиваемых, так и готовых решений, поддержка как в облаке, так и в локальной среде, поддержка гибридных сред, поддержка гибридных сред, поддержка гибридных сред.
- Производительность на уровне лидера отрасли:** Инновационные технологии, оптимизация производительности, поддержка гибридных сред, поддержка гибридных сред, поддержка гибридных сред.
- Полноценное решение для баз данных:** Поддержка гибридных сред, поддержка гибридных сред, поддержка гибридных сред, поддержка гибридных сред, поддержка гибридных сред.
- Аналитика в реальном времени:** Поддержка гибридных сред, поддержка гибридных сред, поддержка гибридных сред, поддержка гибридных сред, поддержка гибридных сред.
- Самые высокие показатели безопасности:** Поддержка гибридных сред, поддержка гибридных сред, поддержка гибридных сред, поддержка гибридных сред, поддержка гибридных сред.

9

Microsoft SQL Server

MS SQL Server 2016 поставляется в двух выпусках:
 - Enterprise Edition
 - Standard Edition

Редакция Microsoft SQL Server Enterprise Edition является оптимальным решением для крупномасштабных хранилищ данных и масштабных приложений корпоративного класса. Выпуск Microsoft SQL Server Standard Edition является экономичным вариантом для организаций малого и среднего размера.

13

Microsoft SQL Server

Microsoft SQL Server Standard Edition

Лицензирование на базе вычислительных мощностей (Core-based) – зависит от мощности сервера, виртуальной и адрес.

Лицензии на основе числа ядер продаются в комплектах для двух ядер физических или виртуальных процессоров.

Чтобы правильно лицензировать физической сервер, необходимо получить лицензии на все ядра этого сервера. Минимально возможное число лицензий на каждый физический процессор сервера составляет 4 лицензии на ядро. Клиентовские лицензии не требуются.

Single License No Level Conlic (лицензия на 2-х лицензий на ядро) – около 285 000 руб.
 Т.е. чтобы выполнить требования минимально возможного числа лицензий (4 лицензий на ядро) – 1 140 000 руб. на двухядерный процессор.

10

Microsoft SQL Server

Преимущества версии 2016

- Повышение эффективности обработки в памяти, при которой транзакции обрабатываются в 30 раз, а запросы — в 100 раз быстрее, чем в дисковых реляционных. Базы данных и системы оперативной аналитики в реальном времени.
- Новая технология Always Encrypted (всегда зашифровано) помогает защитить данные при хранении и при передаче, в локальных системах и в облаке, с помощью основных ключей, расположенных в приложениях.
- Технология Stretch Database (расширение Базы данных) позволяет держать под рукой большую часть данных клиентских журналов, переводя эти данные в Microsoft Azure Безопасным способом.

Microsoft Azure SQL Database – это облачный сервис от корпорации Microsoft, предоставляющий возможность хранения и обработки реляционных данных, а также генерации отчетности (основан на MS SQL server).

14

Microsoft SQL Server

Microsoft SQL Server Standard Edition

Лицензирование на базе вычислительных мощностей (Core-based) – зависит от мощности сервера, виртуальной и адрес.

Лицензии на основе числа ядер продаются в комплектах для двух ядер физических или виртуальных процессоров.

Чтобы правильно лицензировать физической сервер, необходимо получить лицензии на все ядра этого сервера. Минимально возможное число лицензий на каждый физический процессор сервера составляет 4 лицензий на ядро. Клиентовские лицензии не требуются.

Single License No Level Conlic (лицензия на 2-х лицензий на ядро) – около 285 000 руб.
 Т.е. чтобы выполнить требования минимально возможного числа лицензий (4 лицензий на ядро) – 1 140 000 руб. на двухядерный процессор.

11

Microsoft SQL Server

Преимущества версии 2016

- Встроенные возможности расширенной аналитики обеспечивают масштабируемость и эффективность при создании и выполнении дополнительных аналитических алгоритмов непосредственно в основной транзакционной Базе данных SQL Server.
- Изучение Бизнес-данных посредством визуализации на мобильных устройствах с использованием собственных приложений для Windows, iOS и Android.
- Упрощенное управление реляционными и нереляционными данными с помощью запросов T-SQL.
- Развитые средства резервного копирования и репликации. Высокий уровень доступности и однократное аварийное восстановление для резервного копирования и восстановления локальных Баз данных в Microsoft Azure.
- Полные средства разработки приложений: уровень Бизнес-аналитик. С помощью SQL Server Data Tools разработать Базы данных интегрируется в среду Visual Studio и позволяет поддерживать создание новейших корпоративных, Бизнес-аналитических, дата-зависимых, мобильных и веб-приложений как в частной, так и в публичной облаке.

15


СУБД DB2 компании IBM

DB2 – семейство систем управления реляционными Базы данных, выпускаемых корпорацией IBM.

Развитие DB2 уходят корнями в начало 1970-х, когда доктор Эдвард Франк Кадд, работавший на IBM, разработал теорию реляционных Баз данных и в июне 1970 года опубликовал модель манипулирования данными. Для воплощения этой модели он разработал язык реляционных Баз данных и назвал его Alpha.

DB2 – первая СУБД, которая стала использовать SQL. С 1978 по 1982 год протокол DB2 разрабатывался в IBM под названием System R relational, или System R.

СУБД DB2 получила свое название в 1982 году, тогда был выпущен первый коммерческий релиз для VM под названием SQL/DS, и затем релиз для MVS под названием DB2.



Британский ученый Эдвард Франк Кадд

12


Microsoft SQL Server

Microsoft SQL Server Standard Edition

Лицензирование по числу пользователей/устройств – оптимальная модель для предприятий с небольшим числом пользователей Базы данных.

Предлагает покупку лицензий для каждого сервера Базы данных под управлением Microsoft SQL Server и приобретение клиентских лицензий Microsoft SQL Server CAL.

Лицензия на сервер Базы данных – около 190 000 руб.
 Клиентовские лицензии Microsoft SQL Server CAL – около 11 000 руб.



16

СУБД DB2 компании IBM

IBM DB2 – одна из наиболее высокопроизводительных и мощных СУБД в мире. На настоящий момент доступен версия продукта 11.5.


Новые возможности Db2

В июле 2015 г. компания IBM представила новую версию своего семейства баз данных DB2 для операционных систем Linux. Эта платформа теперь сама способна эффективно работать для хранения данных, обработки запросов, обеспечения безопасности, и более того, теперь, благодаря тому же семейству данных, возможно и для гибридных облаков.

Полностью обновленная СУБД IBM DB2 теперь работает на Linux, Oracle и платформе IBM Cloud.

Базы данных DB2 теперь можно применять для хранения и обработки данных в гибридных облаках. IBM предоставляет возможность, которая позволяет клиенту выбирать, где хранить данные, и как их обрабатывать.





17  СУБД DB2 компании IBM

Коммерческая версия DB2 10.5 ("Карнет") для Linux, UNIX and Windows IBM вышла в июне 2012.

Имена продуктов DB2


- DB2 Advanced Enterprise Server Edition
- DB2 Advanced Recovery Feature
- DB2 Advanced Workgroup Server Edition
- DB2 Developer Edition
- DB2 Enterprise Server Edition
- DB2 Express-C
- DB2 Workgroup Server Edition

21  СУБД DB2 компании IBM



10.1 (предпродвижение 9.2)* IBM DB2 v9.x

- Лицензия на 1 пользователя – 9 800 руб.
- Лицензия на 20 пользователей – 162 400 руб.
- Лицензия на 100 пользователей – 736 000 руб.
- Лицензия на сервер (055-04) – 77 500 руб.


18  СУБД DB2 компании IBM

IBM DB2 Workgroup Server Edition — это высокопроизводительная база данных для управления реляционными данными. Решение предназначено для отделов, рабочих групп и организаций малого и среднего размера.

Оно обеспечивает надежное управление данными, высокую готовность, автономные функции и функции защиты. Решение помогает обеспечить более высокую производительность среди исполнителей, надежность и низкую совокупную стоимость владения.


DB2 Workgroup Server Edition предоставляет следующие возможности и преимущества:

- Повышает производительность приложений.
- Предоставляет функции поддержки высокой готовности и восстановления после аварии.
- Предоставляет защищенную и гибкую среду.
- Взаимодействует с разнородными данными более эффективно.
- Повышает производительность и сокращает трудозатраты администраторов.

22  Объектно-реляционная СУБД PostgreSQL


PostgreSQL — свободная объектно-реляционная система управления базами данных (СУБД).

Разработка PostgreSQL в различных формах ведется с 1977 года. Работа началась с проекта Ingres в Калифорнийском университете (Беркли).



Майкл Стоунбрейкер — американский ученый в области информатики

- В 1986 году другая группа, которую возглавлял Майкл Стоунбрейкер (Michael Stonebraker) из Беркли, продолжила работу над Ingres и создала объектно-реляционную СУБД Postgres.
- В 1996 году из-за усовершенствований пакета и перехода на распространение с открытым исходным текстом было принято новое название — PostgreSQL (в течение многолетнего времени использовалось название Postgres95).
- В настоящее время над проектом PostgreSQL активно работает группа разработчиков со всего мира.


19  СУБД DB2 компании IBM

IBM DB2 Enterprise Server Edition — это масштабируемое программное обеспечение базы данных, предназначенное для компаний среднего размера и крупных предприятий.

Это решение включает в себя все функции DB2 Workgroup Server Edition.

Кроме того, оно предоставляет расширенные функции управления данными, поддержки высокой готовности, восстановления после аварии и надежной защиты, позволяя обеспечить высокую производительность и эффективность.

IBM DB2 Express-C — это бесплатная версия программного обеспечения базы данных DB2, предоставляющая основные возможности более функциональной версии DB2.

23  Объектно-реляционная СУБД PostgreSQL


PostgreSQL считается одной из лучших СУБД, распространенной на условиях открытого исходного текста.

В PostgreSQL реализованы многие возможности, традиционно встречающиеся только в масштабных коммерческих продуктах.

По данным многих тестов проводимых для СУБД PostgreSQL значительно уступает по своей производительности Oracle (около 15%).


Существует в реализации для множества UNIX-подобных платформ, включая AIX, различные BSD-системы, HP-UX, IRIX, Linux, Mac OS X, Solaris, Tru64, QNX, а также для Microsoft Windows.

Последняя версия – 12.2

20  СУБД DB2 компании IBM

Сотрудничество компании IC с IBM.


Программные продукты "IC/Предприятие 3 + IBM DB2 v9.x"



Начало продаж - апрель 2010.

Программные продукты "IC/Предприятие 3 + IBM DB2 v9.x" являются расширением продуктовой линейки "IC/Предприятие 3" за счет включения в поставку СУБД IBM DB2 Workgroup v9.x.

Основным преимуществом совместного использования IC/Предприятие и IBM DB2 является возможность построения системы автоматизации на платформе Linux, в то же время повышение отказоустойчивости такой системы за счет резервного сервера DB2 данных.

24  Объектно-реляционная СУБД PostgreSQL

PostgreSQL базируется на языке SQL и поддерживает многие из возможностей стандарта SQL:2011.

Для версии 9.5.3 в PostgreSQL имеются следующие ограничения:

Максимальный размер базы данных	Пят ограничен
Максимальный размер таблицы	32 Тбайт
Максимальный размер зписи	1,6 Тбайт
Максимальный размер поля	1 Гбайт
Максимум зписей в таблице	Пят ограничен
Максимум полей в зписи	250—1000, в зписи меньше, от типа поля
Максимум индексов в таблице	Пят ограничен

1 | **1** | **Задачи администрирования промышленных СУБД**

Основные задачи администрирования базы данных

Анализ предметной области	Проектирование структуры БД	Задание ограничений целостности
Первоначальная загрузка и ведение БД	Защита данных от несанкционированного доступа	Защита от потери данных
Обеспечение восстановления БД	Анализ обращений пользователей	Анализ эффективности работы БД и развитие
Работа с пользователями	Поддержка программных средств	Организационно-методическая работа

5 | **5** | **Задачи администрирования промышленных СУБД**

Анализ обращений пользователей к базе данных

- Сбор статистики обращений пользователей к базе данных.
- Хранение и анализ статистики обращений пользователей к БД.

Анализ эффективности функционирования БД

- Анализ показателей функционирования системы:
 - время обработки;
 - объем памяти;
 - стоимостные показатели.
- Реорганизация и реструктуризация баз данных.
- Развитие программных и технических средств.

Работа с пользователями

- Сбор информации об изменениях в предметной области.
- Сбор информации об оценке пользователями работы БД.
- Определение регламента работы пользователей с БД.
- Обучение и консультирование пользователей.

2 | **2** | **Задачи администрирования промышленных СУБД**

Анализ предметной области

Участие в построении внешней модели АИС, состоящей из описания (в термилах пользователей АИС) предметной области, бизнес-процессов, ресурсов и потоков данных, перечня требований и ограничений к технической реализации АИС.

Проектирование структуры БД

- Участие в построении логической модели АИС (ER-диаграмма).
- Разработка структуры БД на физическом уровне.
- Задание связей между элементами структуры БД.
- Выбор методов упорядочения данных.
- Выбор методов доступа к информации.
- Описание структуры БД.

6 | **6** | **Задачи администрирования промышленных СУБД**

Поддержка программных средств

- Сбор и анализ информации о современной обстановке СУБД и других прикладных программах.
- Поддержание и развитие имеющихся программных средств.
- Приобретение, установка и настройка новых программных средств.

Организационно-методическая работа

- Выбор или создание методики проектирования БД.
- Определение целей и направления развития системы.
- Разработка и выпуск организационно-методических материалов.

3 | **3** | **Задачи администрирования промышленных СУБД**

Задание ограничений целостности

- Определение ограничений целостности, вызванных особенностями предметной области и структурой БД.
- Разработка процедур обеспечения целостности БД при вводе и корректировке данных.
- Обеспечение ограничений целостности при параллельной работе пользователей в многопользовательном режиме.

Первоначальная загрузка и ведение БД

- Разработка технологии первоначальной загрузки и ведения (изменения, добавления, удаления записей).
- Проектирование форм ввода.
- Подготовка исходных данных.
- Ввод и контроль ввода.

7 | **7** | **Типы специалистов, участвующих в администрировании и их задачи**

Администратор БД

Специалист, обеспечивающий создание, функционирование и развитие базы данных.

Администратор БД:

- не может рассматриваться исключительно как технический специалист;
- имеет тесную связь с конечными пользователями на всех этапах жизненного цикла базы данных.

4 | **4** | **Задачи администрирования промышленных СУБД**

Защита данных от несанкционированного доступа

- Обеспечение парольного входа в систему.
- Обеспечение защиты конкретных данных.
- Тестирование средств защиты данных.
- Фиксация попыток несанкционированного доступа к информации.
- Исследование возникающих случаев нарушения защиты данных и проведение мероприятий по их предотвращению.

Защита баз от потери данных

Разработка механизмов защиты баз данных от потери данных.

- Резервирование баз данных.

Обеспечение восстановления БД

- Разработка программно-технологических средств восстановления базы данных.
- Организация и ведение системных журналов.

8 | **8** | **Типы специалистов, участвующих в администрировании и их задачи**

В зависимости от сложности и объема задач, особенностей конкретной СУБД, услуги администрирования БД могут различаться по составу и квалификации специалистов.

Типы специалистов, участвующих в администрировании БД

Системный администратор	Аналитик БД
Архитектор БД	Администратор хранилища данных

9 | **9** | Типы специалистов, участвующих в администрировании и их задачи

Системный администратор

- Отвечает за резервирование и восстановление данных.
- Осуществляет контроль производительности системы.
- Занимается поиском и устранением неполадок.

Дневной специалист обычно в курсе текущего состояния и будущей потребности БД в плане емкости.

Архитектор базы данных

- Занимается разработкой, построением и оптимизацией конфигурации базы данных.
- Разрабатывает интерфейс взаимодействия базы данных с различными внешними приложениями.
- Отвечает за разработку документации, необходимой для сопровождения и развития базы данных.

13 | **13** | Программные средства администрирования промышленных СУБД

Графическая оболочка разработки и администрирования PostgreSQL (PgAdmin III)

PgAdmin - графическая оболочка, предоставляющая сервис разработки и администрирования СУБД PostgreSQL для Unix и Windows.

Она находится в свободном доступе в соответствии с условиями лицензии The PostgreSQL.

Мы работаем с версией pgAdminIII, pgAdminIII написана на C ++, в настоящее время доступна версия pgAdmin IV.

PgAdminIII – это клиентское приложение, которое отправляет SQL-запросы и получает результат их выполнения с сервера PostgreSQL, отображая администратору результат своей работы.

Один клиент pgAdmin может иметь доступ к нескольким серверам PostgreSQL, так же как и один сервер PostgreSQL может принимать несколько клиентов pgAdmin.

10 | **10** | Типы специалистов, участвующих в администрировании и их задачи

Аналитик базы данных

- Занимается проведением исследований, составлением прогноза, анализом результативности использования базы данных и поиском путей ее повышения.
- В отдельных случаях может заниматься сбором и обработкой информации, загрузкой данных в БД.

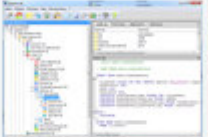
Администратор хранилища данных

- Осуществляет контроль корректности расчетов, производимых в хранилище данных.
- Предоставляет авторизованный доступ к хранилищу.
- Ведет работы по созданию историй запросов
- Консультирует сотрудников по вопросам работы хранилища данных

14 | **14** | Программные средства администрирования промышленных СУБД

По умолчанию, основное окно pgAdminIII содержит дерево объектов, вверху сверху их иерархия.

Слева внизу расположено панель SQL-запросов, связанной отношением к выделенному объекту.



PgAdminIII позволяет увидеть все множество данных, хранящихся PostgreSQL. Информацию подается в контексте, и ее можно быстро и легко найти. Динамическое обновление здесь нет, потому что при работе с приложениями следует помнить о необходимости обновления информации (F5).

11 | **11** | Программные средства администрирования промышленных СУБД

Каждая промышленная СУБД обладает программными средствами, предназначенными для выполнения целей администрирования СУБД.

Перечень программных средств меняется в зависимости от конкретной СУБД, ее версии и т.д. В рамках данной лекции рассматриваются основные программные средства, предназначенные для администрирования промышленной СУБД PostgreSQL.

В PostgreSQL много таких инструментов, но самыми популярными являются следующие:

- pgAdminIII;
- phpPgAdmin;
- терминальные утилиты PostgreSQL.

15 | **15** | Программные средства администрирования промышленных СУБД

PgAdminIII позволяет эффективно решать основные задачи администрирования базы данных PostgreSQL.

- Проверка структуры БД.
- Проверка ограничений целостности.
- Перенос данных из резерва и обратно БД.
- Защита данных от несанкционированного доступа.
- Защита Баз от потери данных.
- Обновление восстановления БД.
- Анализ обращений пользователей к Базе данных.
- Анализ эффективности функционирования БД.

12 | **12** | Программные средства администрирования промышленных СУБД

PgAdminIII – это приложение с открытым кодом, написанное на языке PHP и представляющее собой веб-интерфейс для администрирования СУБД PostgreSQL.

PhpPgAdmin позволяет через браузер осуществлять администрирование сервера PostgreSQL, запускать команды SQL и просматривать содержимое таблиц и баз данных.

16 | **16** | Программные средства администрирования промышленных СУБД

Для выполнения некоторых задач, решаемых SQL-запросами, PgAdminIII имеет достаточно развитые графические средства построения запросов.

Пример создания нескольких источников данных:

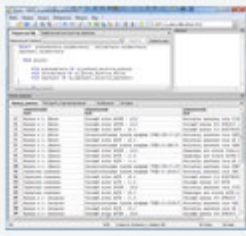
Получение списка из таблицы заводов, содержащего перечень подразделений с указанием оборудования, требующего ремонта, и значений для данного оборудования.

Задача может быть решена запросом:

```
SELECT podzdeleniya.naznachenie, oborudovaniya.naznachenie,
zapochast.naznachenie
FROM zavod/
JOIN podzdeleniya ON id_podzdel_zavod=id_podzdel
JOIN oborudovaniya ON id_oborud_zavod=id_oborud
JOIN zapochast ON id_zapochast_zavod=id_zapochast;
```


17 Программные средства администрирования промышленных СУБД

Запрос может быть набран вручную в редакторе SQL:



18 Программные средства администрирования промышленных СУБД

Другой способ решения задачи – использование графического конструктора построения запросов:



19 Программные средства администрирования промышленных СУБД

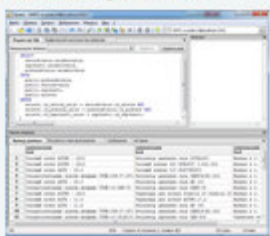
Замечание: ручной способ решения задачи оказывается гораздо более эффективный, чтобы убедиться в этом, достаточно сравнить представленный выше запрос с запросом, сгенерированным графическим конструктором.

```

SELECT
  obrudovaniya.naimenovanie,
  zapozhasti.naimenovanie,
  pozhazheniya.naimenovanie
FROM
  public.pozhazheniya,
  public.obrudovaniya,
  public.zapozhasti,
  public.zalivki
WHERE
  zalivki.id_obrud_zaliv = obrudovaniya.id_obrud AND
  zalivki.id_pozhaz_zaliv = pozhazheniya.id_pozhaz AND
  zalivki.id_zapozhast_zaliv = zapozhasti.id_zapozhast
    
```

20 Программные средства администрирования промышленных СУБД

Хотя результат выполнения запроса выключен:



21 Программные средства администрирования промышленных СУБД

Интерактивный терминал rpar!

В качестве примера терминальной утилиты RparSQL рассмотрим утилиту rpar.

rpar! — это терминальный клиент для работы с RparSQL. Позволяет интерактивно вводить запросы, отправлять их в RparSQL и смотреть результаты. rpar! может быть не только интерактивным, но и из файла.

Кроме того, предоставляется ряд навигационных и различных возможностей, подобные тем, что имеются у консольных оболочек, для обновления навигации, скриншотов и автоматизации широкого спектра задач.

В Windows rpar! создан как "консольное приложение".

Синтаксис

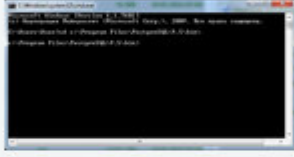
```
rpar! [параметры ...] [База_Данных] [или_пользователь!]
```

22 Программные средства администрирования промышленных СУБД

Чтобы в Windows запустить консольную строку, надо нажать сочетание клавиш Win+R и в появившемся окне выполнить ввод cmd. Сама утилита консольной строки лежит обычно в папке C:\Windows\System32\cmd.exe.


Файл утилиты rpar.exe находится в каталоге c:\Program Files\RparSQL\9.5\bin

Попытку перед запуском утилиты необходимо командой cd перейти в текущий каталог:



23 Программные средства администрирования промышленных СУБД

В качестве примера используем утилиту, приведенный вывод версий RparSQL:

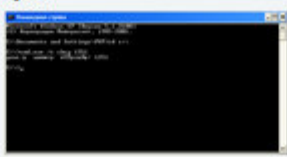


24 Программные средства администрирования промышленных СУБД

При работе с интерактивным терминалом rpar! возникает типичная проблема неправильного отображения символов кириллицы. Суть проблемы состоит в том, что консоль cmd.exe использует кодировку CP866, а сама Windows - WIN1251.

Чтобы решить проблему с кодировкой строчек (как советует документация по rpar!) надо:

1. Задать кодировку строчки (25), в консоли Windows выполнить команду cmd.exe /c echo 25!

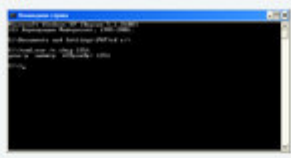


24 Программные средства администрирования промышленных СУБД

При работе с интерактивным терминалом rap возникает типичная проблема неправильного отображения символов кириллицы. Суть проблемы состоит в том, что консоль cmd.exe использует кодировку CP866, а сама Windows - WIN1251.

Чтобы решить проблему с кодировкой (как советует документация по rap) надо:


1. Задать кодировку строку 1251, в консоли Windows выполнить команду `cmd.exe /c chcp 1251`



28 Программные средства администрирования промышленных СУБД

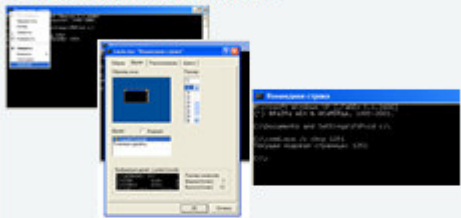
Выводим содержимое таблицы `розделенка`

DMTC» `SELECT * FROM розделенка;`




25 Программные средства администрирования промышленных СУБД

2. Установить шрифт консоли Lucida Console.



29 Программные средства администрирования промышленных СУБД

Возможности утилиты rap! можно также использовать, запустив ее в режиме командной строки SQL Shell (rap!).




26 Программные средства администрирования промышленных СУБД

Теперь поставим задачу просмотреть структуру и содержимое таблицы `розделенка` Базы DMTC.

Консоль cmd.exe имеет текущий каталог в `C:\Program Files\Parasoft\SQL Shell`.

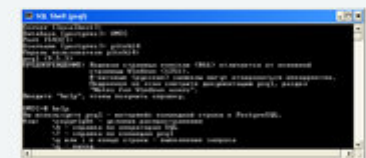
Получаем доступ к Базе DMTC rap! `> /? или базис > /? или пользователи`

Вводим логин пользователя.



30 Программные средства администрирования промышленных СУБД

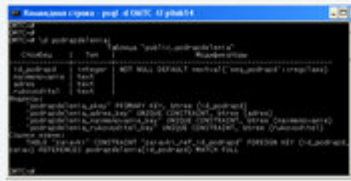
При запуске утилиты необходимо сразу подключиться к серверу и Базе данных.



27 Программные средства администрирования промышленных СУБД

Выводим структуру таблицы `розделенка`

DMTC» `id розделенка;`



31 Конфигурация кластера баз данных

Кластер баз данных представляет собой набор баз, управляемых одним экземпляром работающего сервера.

После инициализации кластер будет содержать базу данных с именем `розделка`, предназначенную для использования по умолчанию утилитами, пользователями и сторонними приложениями.

Сам сервер баз данных не требует наличия базы `розделка`, но она необходима для работы некоторых внешних вспомогательных программ.

При инициализации в каждом кластере создается еще одна база, с именем `template`. Эта база применяется в качестве шаблона создаваемых баз данных, использовать ее в качестве рабочей не следует.

*Направление подготовки 09.03.03 «Прикладная информатика»
Профиль «Прикладная информатика в топливно-энергетическом комплексе»
Методическое обеспечение РПД Б1.В.10 «Администрирование промышленных СУБД»*

Полный комплект лекций по дисциплине «Администрирование промышленных СУБД» в формате мультимедийных презентаций расположен на кафедральных ресурсах в аудитории 210.

МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ ЛАБОРАТОРНЫХ РАБОТ ПО ДИСЦИПЛИНЕ

Лабораторная работа 1. Установка промышленной СУБД

Цель работы: получить практические навыки установки промышленной СУБД и настройки роли суперпользователя.

Задания

1. Провести установку и настройку СУБД PostgreSQL.
2. Создать новую роль базы данных *pitek14*.
3. Создать новое подключение к серверу под пользователем *pitek14*.
4. Изменить привилегии у роли *pitek14*, добавив права суперпользователя.
5. Оформить отчет по проделанной работе.

Методические указания к выполнению заданий

Задание 1. Установка и настройка СУБД PostgreSQL.

Перейти на официальный сайт СУБД PostgreSQL: <http://www.postgresql.org/>. Выбрать в меню: Download, затем - последнюю версию программы (на момент написания методических указаний: PostgreSQL 9.5.4). Из предложенных вариантов выбрать операционную систему компьютера (Windows) и разрядность операционной системы (x86-32 или x86-64). В открывшемся окне нажать Download и загрузить файл на компьютер.

Запустить установочный файл. На предупреждение Windows об установке файла ответить утвердительно. Далее появится начальное диалоговое окно (рисунок 1). Нажать Next.

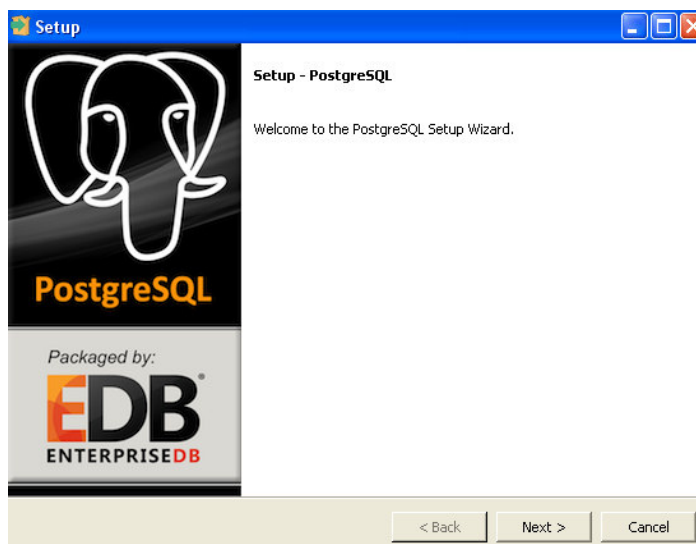


Рисунок 1 – Начальное диалоговое окно установки PostgreSQL

В процессе инсталляции не отмеченные ниже настройки оставлять по умолчанию.

В окне Password ввести пароль для суперпользователя (*superuser*) баз данных - *postgres*. Пароль уточнить у преподавателя.

После завершения установки пользователю предлагается возможность запуска установщика дополнительных компонентов СУБД. На данном этапе этого не требуется. В случае необходимости установщик можно запустить позже в любое время. Отмените установку дополнительных компонент.

По окончании инсталляции необходимо проверить результат установки, запустив среду администрирования pgAdminIII.

Задание 2. Создание новой роли базы данных *pitek14*.

Задать имя новой роли по шаблону *pitek_[N]*, где N – номер студента в журнале. В описании лабораторных работ будем использовать имя *pitek14*.

Подключиться к серверу PostgreSQL 9.5:

- в браузере объектов выделить сервер PostgreSQL 9.5;
- в контекстном меню выполнить «Подключение»;
- ввести пароль суперпользователя *postgres* (уточнить у преподавателя).

В контекстном меню объекта «Роли входа» выполнить «Новая роль».

На вкладке «Свойства» задать имя роли – *pitek14* (уточнить у преподавателя). На вкладке «Определение» задать пароль (уточнить у преподавателя). На вкладке «Привилегии роли» задать все привилегии кроме суперпользователя.

Выполнить «Ок».

Убедиться в появлении новой роли в браузере объектов.

Прокомментировать команду создания роли базы данных[‡].

```
CREATE ROLE pitek14 LOGIN  
ENCRYPTED PASSWORD 'md588269df70679bd55e9483c468a27707a'  
NOSUPERUSER INHERIT CREATEDB CREATEROLE REPLICATION;
```

Задание 3. Создание нового подключения к серверу под пользователем *pitek14*.

Задать имя нового сервера по шаблону *Pitek14_[N]*, где N – номер студента в журнале.

В описании лабораторных работ будем использовать имя *Pitek*.

Подключиться к серверу PostgreSQL 9.5.

Выполнить команду «Файл»/«Добавить сервер». На вкладке «Свойства» задать:

Имя сервера – *Pitek* (уточнить у преподавателя).

Хост – *localhost*. Порт – *5432*.

Обслуживание DB – *postgres*.

Имя пользователя – *pitek14*.

Пароль – пароль пользователя *pitek14*.

Отключить режим «Сохранять пароль». Выполнить «Ок».

Убедиться в появлении нового сервера *Pitek* в браузере объектов.

Сравнить состав объектов сервера «PostgreSQL 9.5» и нового сервера «*Pitek*».

Отсоединиться от сервера *Pitek*, выполнив одноименную команду в его контекстном меню.

Выполнить подключение к серверу *Pitek*, выполнив одноименную команду в его контекстном меню. Ввести пароль пользователя *pitek14*.

Обратите внимание, что при подключении к серверу *Pitek* система не спрашивает пользователя, а автоматически предлагает ввести пароль для пользователя *pitek*, который задан как пользователь при создании этого подключения.

Задание 4. Изменение привилегий у роли *pitek14*.

В следующей лабораторной работе мы отработаем технологию резервного копирования и восстановления базы данных.

Чтобы можно было выполнять операцию восстановления базы данных, требуется привилегия суперпользователя.

Добавьте привилегию «Суперпользователь» для роли *pitek14*.

Выполнить подключение к серверу «PostgreSQL 9.5», выполнив одноименную команду в его контекстном меню и введя пароль.

В браузере объектов выделить роль «*pitek14*» и открыть окно ее свойств. На вкладке

[‡] <https://postgrespro.ru/docs/postgresql/9.5/sql-createrole.html>

«Привилегии роли» включить привилегию «Суперпользователь».

Вопросы для подготовки к защите лабораторной работы

1. Какой сайт является официальным сайтом СУБД PostgreSQL?
2. Для каких операционных систем существуют дистрибутивы СУБД PostgreSQL?
3. Какие условия использования предусмотрены PostgreSQL License?
4. Какие настройки необходимо выполнить при установке СУБД PostgreSQL?
5. Каким образом после окончания процесса установки и начала работы с СУБД PostgreSQL можно установить дополнительные компоненты?
6. Что в PostgreSQL понимается под кластером базы данных?
7. Что такое роль базы данных?
8. Как создать новую роль?
9. Какие привилегии рекомендуется устанавливать для роли базы данных?
10. Какие атрибуты ролей вы знаете?
11. Можно ли после создания роли изменить значения ее атрибут?
12. Прокомментируйте команду создания новой роли.

Лабораторная работа 2. Конфигурирование и настройка операционной среды промышленной СУБД

Цель работы: получить практические навыки конфигурирования и настройки операционной среды промышленной СУБД, а также резервного копирования и восстановления базы данных.

Задания

1. Создать новое табличное пространство *Ptk14*.
2. Создать новую базу данных ОМТС.
3. Создать резервную копию базы данных ОМТС и выполнить операцию восстановления базы данных.
4. Оформить отчет по проделанной работе.

Методические указания к выполнению заданий

Задание 1. Создание нового табличного пространства *ptk14*.

На диске C:\ создайте каталог нового табличного пространства –
C:\Program Files\PostgreSQL\9.5\data\ptk14 (уточнить у преподавателя).

В контекстном меню объекта «Табличные пространства» выполнить «Новый tablespace...».

На вкладке «Свойства» задать:

- имя табличного пространства – *ptk14* (уточнить у преподавателя);
- владелец – *ptek14*.

На вкладке «Определение» задать каталог нового табличного пространства –
C:\Program Files\PostgreSQL\9.5\data\ptk14 (уточнить у преподавателя).

Выполнить «Ок».

Убедиться в появлении нового табличного пространства в браузере объектов (рисунок 2).

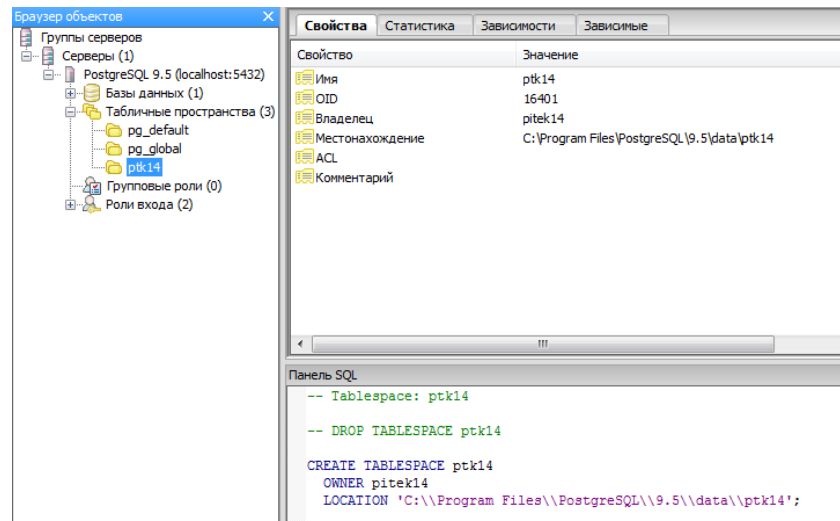


Рисунок 2 – Браузер объектов PostgreSQL

Прокомментировать команду создания табличного пространства§.

```
CREATE TABLESPACE ptk14
```

```
OWNER ptek14
```

```
LOCATION 'C:\Program Files\PostgreSQL\9.5\data\ptk14';
```

Задание 2. Создание новой базы данных *OMTC*.

Задать имя новой базы данных по шаблону *OMTC_[N]*,

где N – номер студента в журнале.

Имя набирайте латиницей. В описании лабораторных работ используется имя *OMTC* (буквы английского алфавита).

На лабораторном практикуме мы будем разрабатывать элементы автоматизированной информационной системы обеспечения процесса оказания услуг по поставкам природного газа, а именно подсистемы, обеспечивающей поддержку деятельности отдела материально-технического снабжения региональной газораспределительной компании по планированию текущего ремонта оборудования. Подробнее постановка задачи будет рассмотрена в лабораторной работе 5-6.

Для создания новой базы данных выполните следующую последовательность действий.

В контекстном меню объекта «Базы данных» выполнить «Новая база данных...».

На вкладке «Свойства» задать: имя базы данных – *OMTC*; владелец – *ptek14*.

На вкладке «Определение» задать: кодировка – оставьте по умолчанию; шаблон – *postgres*; табличное пространство – *ptk14*.

Выполнить «Ок».

Убедиться в появлении новой базы данных в браузере объектов.

Прокомментировать команду создания базы данных**.

```
CREATE DATABASE "OMTC"
```

```
WITH OWNER = ptek14
```

```
ENCODING = 'UTF8'
```

```
TABLESPACE = ptk14
```

```
LC_COLLATE = 'Russian_Russia.1251'
```

§ <https://postgrespro.ru/docs/postgresql/9.5/sql-createtablespace.html>

** <https://postgrespro.ru/docs/postgresql/9.5/sql-createdatabase.html>

`LC_CTYPE = 'Russian_Russia.1251'`
`CONNECTION LIMIT = -1;`

Задание 3. Создание резервной копии и восстановление базы данных.

Задание 3.1. Создать резервную копию базы данных ОМТС.

Подключиться к серверу *Pitek*, выполнив одноименную команду в его контекстном меню и введя пароль.

В браузере объектов выделить базу данных ОМТС и в контекстном меню выполнить команду «Резервная копия ...»

Указать путь и имя файла, в который будет записана копия (уточнить у преподавателя).

Установить степень сжатия в диапазоне 1-9. Остальные настройки оставить по умолчанию.

Выполнить команду «Резервная копия».

Убедиться, что операция резервного копирования выполнена без ошибок. В этом случае последним будет сообщение: «Процесс вернул код выхода 0».

Выполнить команду «Завершено». Убедиться в появлении файла резервной копии в заданном месте.

Задание 3.2. Восстановить базу данных ОМТС из резервной копии.

Алгоритм восстановления может быть следующий:

- удалить восстанавливаемую базу *ОМТС*;
- отключиться от сервера;
- подключиться к серверу с правами суперпользователя;
- создать «пустую» базу данных с тем же именем и теми же параметрами, что и восстанавливаемая база;
- выполнить операцию восстановления.

Замечание: Можно не удалять, а переименовать восстанавливаемую базу данных, изменив имя базы на вкладке «Свойства». А удалить переименованную базу уже после успешного восстановления.

В браузере объектов выделить базу данных *ОМТС* и в контекстном меню выполнить команду «Удалить»

Отключаться от сервера в данном случае нет необходимости, так как для пользователя *ritek14* мы задали привилегию «Суперпользователь».

Создать новую «пустую» базу данных *ОМТС*, как это было сделано при выполнении задания 2.

В браузере объектов выделить базу данных *ОМТС* и в контекстном меню выполнить команду «Восстановить»

Указать путь и имя файла, из которого будет восстановлена база данных. Остальные настройки оставить по умолчанию.

Выполнить команду «Восстановить».

Убедиться, что операция восстановления выполнена без ошибок. В этом случае последним будет сообщение: «Процесс вернул код выхода 0».

Выполнить команду «Завершено».

В контекстном меню сервера выполнить команду «Обновить». Проверить появление восстановленной базы.

Замечание:

В процессе выполнения лабораторных работ, курсовой работы вам потребуется переносить базы данных с одного компьютера на другой. Эта операция несколько отличается от просто операции резервного копирования/восстановления.

Алгоритм переноса базы данных с одного компьютера на другой может быть следующим.

На компьютере № 1:

- создать резервную копию базы данных на внешний носитель.

На компьютере №2:

- создать такие же как на компьютере 1:

- подключение к серверу (не обязательно);

- роль владельца базы данных;

- табличное пространство базы данных;

- «пустую» базу данных;

- подключиться к серверу с правами суперпользователя;

- выполнить процедуру восстановления базы данных с внешнего носителя.

Вопросы для подготовки к защите лабораторной работы

1. Для чего предназначены табличные пространства?
2. Как создать новое табличное пространство?
3. Какие параметры задаются при создании нового табличного пространства?
4. Прокомментируйте команду создания нового табличного пространства.
5. Что собой представляет база данных PostgreSQL?
6. Что такое схема, каково назначение схем в PostgreSQL?
7. Как создать новую базу данных?
8. Какие параметры задаются при создании новой базы данных?
9. Прокомментируйте команду создания базы данных.
10. Какова последовательность действий при создании резервной копии базы данных?
11. Какова последовательность действий при восстановлении базы данных из резервной копии?
12. Какие права должны быть у пользователя для создания резервной копии базы данных?
13. Какие права должны быть у пользователя для восстановления базы данных?
14. Вы сделали дома часть заданий лабораторной работы, какие действия вы должны выполнить, чтобы продемонстрировать преподавателю свою работу на компьютере в институтском классе?
15. Можно ли создать резервную копию базы данных сохранением на другом носителе файлов каталога табличного пространства базы данных?

Лабораторная работа 3. Конфигурирование структуры базы данных

Цель работы: получить практические навыки конфигурирования структуры базы данных промышленной СУБД.

Задания

1. Провести анализ предметной области. Конкретизировать постановку задачи.
2. Разработать структуру базы данных.
3. Построить схему данных.
4. Создать таблицу подразделений.
5. Заполнить таблицу подразделений данными.
6. Создать таблицу списка оборудования.
7. Заполнить таблицу списка оборудования методом импорта данных из файла.
8. Создать и заполнить таблицу запасных частей для ремонта.
9. Оформить отчет по проделанной работе.

Постановка задачи

Средствами СУБД PostgreSQL разработать элементы автоматизированной информационной системы обеспечения процесса оказания услуг по поставкам природного газа.

Наименование комплекса задач: планирование и получение необходимых ресурсов.
Реализуемая функция: приобретение материалов, комплектующих и запасных частей.
Реализуемая задача: планирование текущего ремонта.

Методические указания к выполнению заданий

Задание 1. Анализ предметной области. Конкретизация постановки задачи.

Средствами СУБД PostgreSQL разработать элементы автоматизированной информационной системы обеспечения процесса оказания услуг по поставкам природного газа, а именно подсистемы, обеспечивающей поддержку деятельности отдела материально-технического снабжения региональной газораспределительной компании по планированию текущего ремонта оборудования.

Подсистема должна обеспечивать выполнение следующих функций.

1. Учет заявок от подразделений на запасные части для ремонта.
2. Предоставление отчетов.

На основе анализа предметной области составить и привести в отчете перечень реализуемых действий по обработке информации, таких как вывод списка руководителей всей подразделений, вывод списка заявок на запчасти для ремонта от нескольких заданных подразделений, получение списка подразделений с указанием требуемой суммы на запчасти для ремонта за заданный диапазон дат ремонта с подведением итоговой суммы затрат на запчасти и др.

Задание 2. Разработка структуры базы данных.

База данных *ОМТС* представляет собой реляционную базу данных, состоящую из таблиц и связей между ними. Анализ предметной области показывает, что для решения поставленной задачи база данных должна содержать следующие таблицы.

Таблица 1 – Список таблиц базы данных

№ п/п	Имя таблицы	Назначение
1	<i>podrazdelenia</i>	Список подразделений
2	<i>oborudovanie</i>	Список газораспределительного и бытового газового оборудования
3	<i>zapchasti</i>	Список запчастей для ремонта оборудования
4	<i>zaiavki</i>	Учет заявок на запасные части от структурных подразделений на проведение ремонта

В терминах PostgreSQL поля таблицы называют колонками, а записи – строками.

Рекомендуемый перечень колонок таблиц представлен в таблице 2.

Требуется разработать и привести в отчете полную структуру таблиц базы данных, перечень необходимых сведений представлен в таблице 3.

Основные стандартные типы полей PostgreSQL подробно описаны в справочной поддержке СУБД^{††}.

Наиболее часто применяются следующие типы данных:

- *integer* для целых чисел,
- *numeric* для чисел, которые могут быть дробными,
- *text* для текстовых строк,
- *date* для дат,
- *time* для времени,
- *timestamp* для значений, включающих дату и время.

^{††} <https://postgrespro.ru/docs/postgresql/9.5/datatype>

Задание 3. Построение схемы данных.
Схему данных создайте самостоятельно.

Таблица 2 – Перечень полей (колонок) таблиц базы данных

№	Имя колонки	Назначение	№	Имя колонки	Назначение
Таблица <i>podrazdelenia</i>			Таблица <i>oborudovanie</i>		
1	<i>id_podrazd</i>	Идентификатор строки	1	<i>id_oborud</i>	Идентификатор строки
2	<i>naimenovanie</i>	Наименование подразделения	2	<i>naimenovanie</i>	Наименование оборудования
3	<i>adres</i>	Адрес	3	<i>tip</i>	Тип/марка оборудования
4	<i>rukovoditel</i>	Фамилия, инициалы руководителя	4	<i>srok</i>	Срок службы (лет)
Таблица <i>zaiavki</i>			Таблица <i>zapchasti</i>		
1	<i>id_zaiavki</i>	Идентификатор строки	1	<i>id_zapchasti</i>	Идентификатор строки
2	<i>nomer</i>	Номер заявки	2	<i>naimenovanie</i>	Наименование запчасти
3	<i>id_podrazd_zaiav</i>	Целевая колонка. Значение идентификатора строки таблицы <i>podrazdelenia</i>	3	<i>tip</i>	Тип/марка запчасти
4	<i>id_oborud_zaiav</i>	Целевая колонка. Значение идентификатора строки таблицы <i>oborudovanie</i>	4	<i>fasovka</i>	Единица поставки
5	<i>id_zapchasti_zaiav</i>	Целевая колонка. Значение идентификатора строки таблицы <i>zapchasti</i>			
6	<i>kol_vo</i>	Количество единиц поставки			
7	<i>cena_plan</i>	Планируемая цена запчасти			
8	<i>data_remonta</i>	Планируемая дата ремонта			

Таблица 3 – Перечень требуемых сведений по структуре таблиц базы данных

№	Имя колонки	Тип данных	Длина	Ограничения	Примечание (назначение колонки)
1					

Задание 4. Создание таблицы подразделений.

Порядок действий при создании и программировании структуры таблицы выглядит следующим образом:

- создать последовательность для заполнения уникальными значениями полей-идентификаторов строк типа первичный ключ (PRIMARY KEY);
- создать таблицу;
- добавить в таблицу колонки;
- задать ограничения для колонок.

Задание 4.1. Создать последовательность *seq_podrazd* для заполнения уникальными значениями колонки *id_podrazd*.

В дереве объектов базы данных *ОМТС* выделить объект *Последовательности* и в контекстном меню выполнить команду «Новая последовательность ...». Задать свойства и определить параметры последовательности.

Задание 4.2. Создать таблицу *podrazdelenia*.

В дереве объектов базы данных *ОМТС* выделить объект *Таблицы* и в контекстном меню выполнить команду «Новая таблица ...». Задать свойства таблицы. Задать табличное пространство (каталог, где будет храниться файл таблицы) - *ptk14* (уточнить у преподавателя).

Задание 4.3. Добавить в таблицу колонки.

Добавить в таблицу *podrazdelenia* колонку *id_podrazd* – идентификатор строки. Задать свойства колонки *id_podrazd*. На вкладке «Определение» установить «Значение по умолчанию» - *nextval('public.seq_podrazd')*.

nextval – одна из функций для работы с последовательностями, увеличивает текущее значение последовательности *seq_podrazd* и возвращает новое значение, обеспечивая таким образом наращивание значений колонки *id_podrazd* с использованием последовательности *seq_podrazd* в момент ввода новой строки.

Добавить колонки *naimenovanie*, *adres*, *rukovoditel*.

Задание 4.4. Задать ограничения для колонок.

Задать ограничение PRIMARY KEY для колонки *id_podrazd* – идентификатор строки. Открыть «Свойства» объекта таблица *podrazdelenia*. Перейти на вкладку «Ограничения». Выбрать «Первичный ключ» (PRIMARY KEY). Выполнить «Добавить».

Установить ограничение уникальности (UNIQUE) для полей *naimenovanie*, *adres* и *rukovoditel*.

Задание 5. Заполнение таблицы подразделений данными.

В этом задании мы отработаем две технологии ввода/редактирования данных в таблице:

- при помощи утилиты просмотра/редактирования данных, входящей в состав PgAdminIII;

- выполнением команды SQL в утилите «Инструмент запросов» PgAdminIII.

Примерное содержимое таблицы *podrazdelenia* получить у преподавателя.

Задание 5.1. Ввод данных в таблицу при помощи утилиты просмотра/редактирования данных PgAdminIII.

В дереве объектов таблицы *ОМТС* выделить таблицу *podrazdelenia* и в контекстном меню выполнить команду «Просмотр данных/Просмотр всех строк». Заполнение строк данными осуществить в окне «Редактирование данных ...».

Ожидаемый результат представлен на рисунке 3.

	id_podrazd [PK] integer	naimenovanie text	adres text	rukovoditel text
1	1	АО «Газ Область»	г. Область, ул. Тимирязева, д. 15.	Потапов И.А.
2	2	филиал в г. Двинск	г. Двинск, ул. Солнечная, д. 54	Коршунов А.П.
3	3	филиал в г. Ангарово	г. Ангарово, ул. Северная, д. 29	Дорохов А.В.
4	4	филиал в г. Дубровники	г. Дубровники, ул. Южная, д. 4	Филимонов О.Е.
5	5	филиал в г. Шклов	г. Шклов, ул. Восточная, д. 31	Краско И.И.
*				

Рисунок 3 – Заполнение данными строк таблицы *podrazdelenia*

Задание 5.2. Ввод/Редактирование данных в таблице выполнением команды SQL в утилите «Инструмент запросов» PgAdminIII.

Для добавления (вставки) записи в таблицу служит оператор INSERT:

INSERT INTO имяТаблицы (списокИменКолонок) VALUES (списокЗначений)

– вставляет пустую запись в таблицу *имяТаблицы* и вводит в столбцы *списокИменКолонок* значения из списка *списокЗначений*. При этом в первый столбец из *списокИменКолонок* вводится первое значение из *списокЗначений*, во второй столбец – второе значение и т.д.

Задание 6. Создание таблицы со списком оборудования.

Самостоятельно:

- создать последовательность *seq_oborud* для заполнения уникальными значениями колонки *id_oborud*;
- создать таблицу *oborudovanie*.

Задание 7. Заполнение таблицы списка оборудования методом импорта данных из файла.

Этот метод часто бывает очень удобным, но требует аккуратности и точности исполнения. Мы рассмотрим его для случая импорта из таблицы Excel.

Алгоритм импорта данных из файла выглядит следующим образом.

7.1. Подготовить таблицу Excel к импорту.

Требования к таблице на рабочем листе Excel:

- столбцы должны иметь те же заголовки, что и колонки таблицы PostgreSQL;
- типы данных в столбцах должны совпадать с типами данных в колонках;
- должны соблюдаться ограничения, установленные для колонок, например, столбец первичного ключа не должно содержать дублирующих значений;
- не должно быть пустых ячеек, например, для пустых текстовых ячеек можно ввести символ тире.

Примерное содержимое таблицы *oborudovanie* получить у преподавателя.

7.2. Сохранить рабочий лист Excel в формате текста *Unicod*.

7.3. С помощью редактора *notepad* преобразовать текст в формате *Unicod* в файл формата *csv* с кодировкой *UTF-8* и разделителем «;».

Кодировка *UTF-8* позволяет избежать проблемы с импортом специальных символов.

Главное требование к файлу *csv* – отсутствие лишних символов.

В принципе можно использовать в качестве разделителя символ табуляции[tab]. PgAdminIII позволяет импортировать данные из такого файла. Просто в случае в раздлителем «;» легче проследить отсутствие лишних символов в файле *csv*.

7.4. Осуществить импорт данных в таблицу.

В дереве объектов таблицы *OMTC* выделить таблицу *oborudovanie*. В контекстном меню выполнить команду «Импорт...»

Задание 8. Создание и заполнение таблицы запасных частей для ремонта.

Самостоятельно:

- создать последовательность *seq_zapchasti* для заполнения уникальными значениями колонки *id_zapchasti*;
- создать таблицу *zapchasti*;
- заполнить таблицу данными (примерное содержимое таблицы *zapchasti* получить у преподавателя)

Вопросы для подготовки к защите лабораторной работы

1. Что такое тип данных?
2. Назовите основные стандартные типы данных PostgreSQL.
3. Какие числовые типы данных вы знаете?

4. В чем особенности символьных типов character varying(n), varchar(n), character(n), char(n), text?
5. Каков порядок действий при создании и программировании структуры таблицы?
6. Что такое последовательность? Для каких целей используются последовательности?
7. Как создать последовательность? Команда CREATE SEQUENCE.
8. Как создать таблицу? Команда CREATE TABLE.
9. Как добавить в таблицу колонку? Команда ALTER TABLE.
10. Что такое ограничения на данные?
11. Каково назначение ограничения Первичный ключ (PRIMARY KEY)?
12. Каким образом устанавливаются ограничения для колонок таблицы?
13. Какие действия необходимо выполнить, чтобы создать колонку – идентификатор строки с уникальными значениями?
14. Какие способы ввода/редактирования строк таблицы вы знаете?
15. Каково назначение утилиты просмотра/редактирования данных PgAdminIII? Как с ее помощью вставить/отредактировать строку таблицы?
16. Каково назначение утилиты «Инструмент запросов» PgAdminIII? Как с ее помощью выполнить запрос?
17. Назначение и синтаксис команд SQL вставки и удаления строк в таблице.
18. Какие действия необходимо выполнить, чтобы осуществить импорт данных в таблицу из файла?
19. Какие требования предъявляют к подготовленной для импорта таблице Excel?
20. Какие требования предъявляют к подготовленному для импорта файлу csv? Что такое файл csv?
21. Как осуществить импорт данных в таблицу в PgAdminIII?

Лабораторная работа 4. Упорядочивание структуры базы данных

Цель работы: получить практические навыки создания и конфигурирования связанных таблиц, хранения и отображения дат и денежных единиц, создания индексов базы данных промышленной СУБД.

Задания

1. Создать таблицу учета заявок на запасные части для ремонта.
2. Заполнить таблицу учета заявок на запасные части для ремонта данными.
3. Вывести содержимое таблицы *zaiavki* с отображением денежных единиц и дат в российском стандарте.
4. Создать индекс для таблицы запасных частей в порядке возрастания наименования изделия.
5. Создать индекс для таблицы подразделений в порядке убывания наименования подразделения.
6. Оформить отчет по проделанной работе.

Методические указания к выполнению заданий

Задание 1. Создание таблицы *zaiavki* учета заявок на запасные части для ремонта.

Таблица имеет локальные (целевые) колонки *id_podrazd_zaiav*, *id_oborud_zaiav*, *id_zapchasti_zaiav*, зависимыми (ссылающимися) колонками для которых являются одноименные колонки *podrazdelenia.id_podrazd*, *oborudovanie.id_oborud*, *zapchasti.id_zapchasti*.

Схема связи таблиц представлена на рисунке 4.

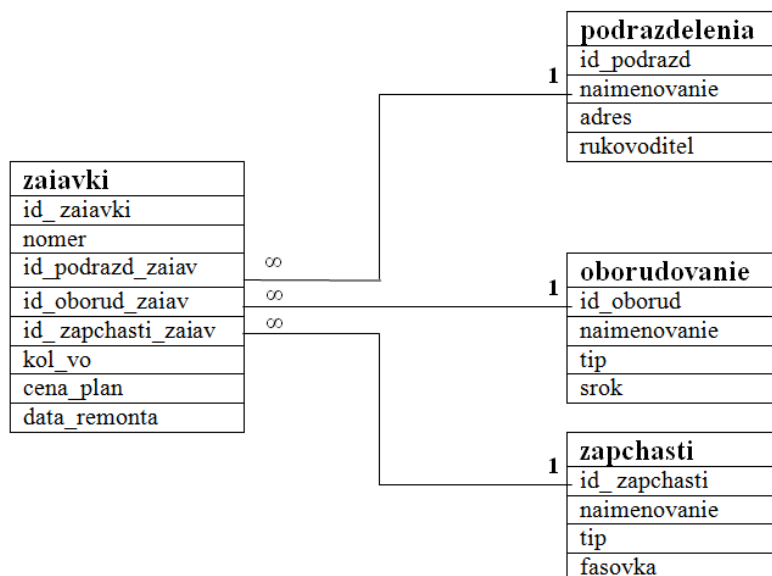


Рисунок 4 – Схема связи таблиц *zaiavki*, *podrazdelenia*, *oborudovanie*, *zapchasti* базы данных ОМТС

Локальные (целевые) колонки имеют ограничение внешнего ключа. Такое ограничение указывает, что значения колонки должны соответствовать значениям в некоторой строке другой таблицы. Это называется *ссылочной целостностью* двух связанных таблиц.

Т.е., например, в колонке *zaiavki.id_podrazd_zaiav* должны быть только такие значения, которые есть в колонке *podrazdelenia.id_podrazd*.

Самостоятельно:

- создать последовательность *seq_zaiavki* для заполнения уникальными значениями колонки *id_zaiavki*;

- создать таблицу *zaiavki*.

Новое в создании таблицы *zaiavki* состоит в задании ограничения типа Внешний ключ (FOREIGN KEY) для колонок *id_podrazd_zaiav*, *id_oborud_zaiav*, *id_zapchasti_zaiav*.

Рассмотрим порядок задания ограничения типа Внешний ключ (FOREIGN KEY) на примере колонки *id_podrazd_zaiav*.

В свойствах таблицы *zaiavki* перейти на вкладку «Ограничения». Для ограничения «Внешний ключ» выполнить команду «Добавить». Задать имя нового внешнего ключа *zaiavki_ref_id_podrazd*. На вкладке «Определение» включить режим «Совпадение полное». На вкладке «Колонки» установить: зависимая таблица – *public.podrazdelenia*; локальная колонка – *id_podrazd_zaiav*; зависимая колонка – *id_podrazd*. Выполнить «Добавить». Выполнить «Ок».

Аналогично выполнить установку ограничения типа Внешний ключ (FOREIGN KEY) для колонок *id_oborud_zaiav* и *id_zapchasti_zaiav*.

Имена внешних ключей задать следующие: *zaiavki_ref_id_oborud* и *zaiavki_ref_id_zapchasti*.

Задание 2. Заполнение данными таблицы *zaiavki* учета заявок на запасные части для ремонта.

Ввод данных в таблицу *zaiavki* имеет две новые особенности, ранее нами не рассматриваемые.

Первая особенность: в таблице есть колонка нового типа *date*, заполнение которой требует знание формата представления дат, который зависит от локализации даты и времени.

Изменение формата представления значений типа *date* иллюстрируется примером в таблице 4.

Таблица 4 – Изменение формата представления значений типа *date*

Тип данных	Вводимое значение		Формат представления в колонке
	вручную	из файла	
date	01/10/16	01.10.16	2016-10-01

При импорте данных из файла, значения в колонке *data_remonta* должны быть предварительно отформатированы (например, средствами форматирования ячеек Excel) соответствующим образом.

Вторая особенность – таблица имеет целевые колонки *id_podrazd_zaiav*, *id_oborud_zaiav*, *id_zapchasti_zaiav*, которые содержат идентификаторы из зависимых колонок *podrazdelenia.id_podrazd*, *oborudovanie.id_oborud*, *zapchasti.id_zapchasti*. Что затрудняет ее восприятие и требует аккуратности в подготовке данных.

Заполнить таблицу *zaiavki* методом импорта данных из файла *csv*.

1. Подготовить к импорту таблицу Excel с данными для таблицы *zaiavki*.

Данные по заявкам на запасные части для ремонта составить самостоятельно и согласовать с преподавателем.

2. Сохранить рабочий лист Excel в файл *zaiavki.txt* формате текста *Unicod*.

Для этого необходимо выполнить команду «Сохранить как» и в Тип файла» установить «Текст Юникод».

3. С помощью редактора *notepad* преобразовать текст в формате *Unicod* в файл формата *csv* с кодировкой *UTF-8* и разделителем «;».

4. Осуществить импорт данных в таблицу *zaiavki* из файла *zaiavki.scv*.

Задание 3. Вывод содержимого таблицы *zaiavki* с отображением денежных единиц и дат в российском стандарте.

PostgreSQL имеет встроенный тип *money* для колонок с денежными единицами. *Money* считается устаревшим типом данных и не рекомендуется к применению. Дело в том, что этот тип отображает денежную единицу в соответствии с настройками локализации, что может привести к проблемам при переносе базы с одного сервера на другой.

Вместо типа *money* следует использовать тип *numeric* с точностью, достаточной для представления максимальной необходимой величины (включая две цифры для дробной части).

Форматирование, аналогичное типу *money* можно выполнить при помощи функции *to_char()*^{‡‡}.

С помощью функции *to_char()* можно также выводить дату в удобном для нас формате *День.Месяц.Год*.

Для денежных единиц функция *to_char(cena_plan, '999999.99')||' p.'* преобразует цену из ячейки колонки *cena_plan* в текст формата '999999.99', а оператор конкатенации строк добавляет символ денежной единицы.

Для дат функция *to_char(data_remonta, 'DD.MM.YYYY')* преобразует дату из ячейки колонки *data_remonta* в текст формата 'DD.MM.YYYY'.

Запрос SQL на вывод всей таблицы *zaiavki* с измененными форматами вывода колонок *cena_plan* и *data_remonta* имеет следующий вид:

```
SELECT id_zaiavki, nomer, id_podrazd_zaiav, id_oborud_zaiav, id_zapchasti_zaiav, kol_vo,
to_char(cena_plan, '999999.99')|| ' p.' , to_char(data_remonta, 'DD.MM.YYYY') FROM public.zaiavki;
```

• ‡‡ <https://postgrespro.ru/docs/postgresql/9.5/functions-formatting>

Вывести содержимое таблицы *zaiavki* с отображением денежных единиц и дат в российском стандарте.

В контекстном меню таблицы *zaiavki* выполнить «Скрипты/Скрипт Select». В редакторе запросов заменить стандартный запрос на запрос, представленный выше. Выполнить запрос.

Задание 4. Создание индекса для таблицы запасных частей в порядке возрастания наименования изделия.

В дереве объектов таблицы *zapchasti* выделить «Индексы». В контекстном меню выполнить «Новый индекс». На вкладке «Свойства» задать имя индекса - *zapch_naimen_idx*.

На вкладке «Свойства» задать: табличное пространство – *ptk14*; метод доступа – *btree* (В-дерево, значение по умолчанию, можно не устанавливать).

На вкладке «Колонки» добавить колонку *naimenovanie*. Выполнить «Ok». Убедиться в появлении индекса в дереве объектов.

Прокомментировать команду SQL§§:

```
CREATE INDEX zapch_naimen_idx  
ON public.zapchasti  
USING btree  
(naimenovanie COLLATE pg_catalog."default");
```

Задание 5. Создание индекса для таблицы подразделений в порядке убывания наименования подразделения.

Выполнить данное задание самостоятельно. Прокомментировать команду SQL создания индекса.

Вопросы для подготовки к защите лабораторной работы

1. Каково назначение ограничения Внешний ключ (FOREIGN KEY)?
2. Что такое локальная (целевая) колонка?
3. Что такое зависимая (ссылающаяся) колонка?
4. Какие типы связи поддерживаются ограничениями Внешний ключ?
5. Каким образом устанавливается ограничение Внешний ключ для колонок таблицы?
6. Какие особенности имеет формат представления данных типа date?
7. Какие особенности имеет представления денежных единиц?
8. Какие особенности имеет подготовка файла с данными, имеющими столбцы типов дат и денежных типов, для импорта в таблицу?
9. Что позволяет сделать функция *to_char*?
10. Каким образом можно осуществить вывод денежных единиц и дат в российском стандарте?
11. Как запустить скрипт в PgAdminIII?
12. Какие задачи выполняют индексы в PostgreSQL?
13. Какие типы индексов вы знаете?
14. Как создать индекс? Как его удалить?

Лабораторная работа 5. Реализация действий по обработке информации базы данных

Цель работы: получить практические навыки простой выборки данных и отбора данных промышленной СУБД по условию.

Задания

Простая выборка данных

• §§ <https://postgrespro.ru/docs/postgresql/9.5/sql-createindex>

1. Вывести все записи из таблиц *podrazdelenia*, *oborudovanie*, *zapchasti*, *zaiavki*.
 2. Получить список руководителей всей подразделений и список сроков службы оборудования.
 3. Получить список идентификаторов подразделений, подававших заявки на запчасти для ремонта оборудования. Выполнить задание с заголовком столба, набранным кириллицей.
 4. Получить список идентификаторов запчастей из заявок без повторений с указанием количества, цены и суммы.
 5. Вывести в алфавитном порядке список подразделений. Вывести список оборудования в порядке убывания срока его службы.
 6. Вывести все записи таблицы *zaiavki* в следующем порядке:
 - сначала в порядке возрастания планируемых сроков ремонта,
 - затем в порядке возрастания идентификатора подразделений,
 - затем в порядке убывания суммы, требуемой для закупки запчастей.
- Отбор данных по условию*
7. Вывести список оборудования со сроком службы более 10 лет.
 8. Вывести список запчастей из заявок для ремонта с заданной планируемой датой ремонта и стоимостью более 5 000 рублей.
 9. Вывести список оборудования со сроком службы от 10 лет до 15 лет.
 10. Получить список заявок на запчасти для ремонта от трех заданных подразделений.
 11. Вывести список всех запчастей – газовых клапанов.
 12. Вывести в алфавитном порядке список всех подразделений – филиалов.
 13. Вывести список всех газовых котлов со сроком службы более 10 лет.
 14. Получить список запчастей для ремонта в заданном месяце стоимостью менее 10 000 рублей.
 15. Получить список подразделений, запланировавших ремонт на заданный месяц.
 16. Оформить отчет по проделанной работе.

Методические указания к выполнению заданий

Все запросы на выборку данных осуществляются при помощи команды *SELECT*, подробное описание которой представлено в справке PostgreSQL***.

Создание, редактирование и запуск пользовательский запросов осуществляется в «Редакторе SQL» среды администрирования pgAdminIII.

Задание 1. Вывод всех записей из таблиц *podrazdelenia*, *oborudovanie*, *zapchasti*, *zaiavki*.

Запрос, возвращающий все данные (все столбцы и все записи) из одной таблицы, формулируется следующим образом:

```
SELECT *  
FROM имяТаблицы;
```

В дереве объектов таблицы *ОМТС* выделить таблицу *podrazdelenia*. В панели инструментов щелкнуть по инструменту «Выполнить пользовательские SQL-запросы». В окне «Редактор SQL» ввести запрос, выполняющий вывод всей таблицы, и выполнить его. Убедиться, что запрос выполнен без ошибок.

Вывод всех записей из таблиц *oborudovanie*, *zapchasti*, *zaiavki* выполнить самостоятельно.

Задание 2. Получение списка руководителей всей подразделений и список сроков службы оборудования.

Если требуется выбрать не все столбцы таблицы, а только некоторые, то используется следующий SQL-запрос:

*** <https://postgrespro.ru/docs/postgresql/9.5/sql-select.html>

SELECT список Столбцов
FROM имя Таблицы

Задание 3. Получение списка идентификаторов подразделений, подававших заявки на запчасти для ремонта оборудования.

Для отбора без повторения или вывода всех записей используются ключевые слова *ALL* | *DISTINCT*, которые ставятся после *SELECT*

SELECT ALL | *DISTINCT ...*

ALL – все записи (по умолчанию); *DISTINCT* – в результатной таблице представляются только уникальные записи.

Заголовки столбцов в результатной таблице можно переопределить по своему усмотрению, назначив для них так называемые псевдонимы (синонимы). Для этого в списке столбцов после соответствующего столбца следует написать выражение вида:

AS заголовок Столбца

Задание 4. Получение списка идентификаторов запчастей из заявок без повторений с указанием количества, цены и суммы.

Задание выполнить самостоятельно.

Задание 5. Вывод в алфавитном порядке списка подразделений. Вывод списка оборудования в порядке убывания срока его службы.

Средством сортировки является конструкция *ORDER BY*.

Секции *ORDER BY* передается список полей, разделенный запятыми (или выражений, в которых используются поля). Переданный список задает критерий сортировки. Для каждого критерия сортировки могут дополнительно указываться ключевые слова *ASC* и *DESC*, управляющие типом сортировки.

ASC. Записи сортируются по возрастанию заданного критерия (то есть числа сортируются от меньших к большим, даты – от ранних к поздним, а текст – по алфавиту). По умолчанию выбирается именно этот способ сортировки, поэтому *ASC* используется лишь для наглядности.

DESC. Записи сортируются по убыванию заданного критерия (то есть числа сортируются от больших к меньшим, даты – от поздних к ранним, а текст – в порядке, обратном алфавитному).

При сортировке по нескольким выражениям в списке после *ORDER BY* сначала производится упорядочивание итогового набора по первому (левому) критерию, а дальнейшие критерии применяются лишь в том случае, если сортировка по первому критерию не обеспечивает однозначного результата.

При сортировке по нескольким критериям для каждого критерия можно задать свой тип упорядочивания.

Задание 6. Вывод всех записей таблицы *zaiavki* в следующем порядке:

- сначала в порядке возрастания планируемых сроков ремонта,
- затем в порядке возрастания идентификатора подразделений,
- затем в порядке убывания суммы, требуемой для закупки запчастей.

Задание выполнить самостоятельно.

Задание 7. Вывод списка оборудования со сроком службы более 10 лет.

Для выделения требуемых записей (строк) исходной таблицы используется выражение, следующее за ключевым словом *WHERE*. Условия поиска в операторе *WHERE* являются

логическими выражениями, т.е. принимающими одно из двух возможных значений – *true* (ИСТИНА) или *false* (ЛОЖЬ).

SELECT списокСтолбцов

FROM списокТаблиц

WHERE условие;

Задание 8. Вывод списка запчастей из заявок для ремонта с заданной планируемой датой ремонта и стоимостью более 5 000 рублей.

Секция *WHERE* может содержать несколько условий, объединенных логическими операторами (например, *AND* или *OR*) и возвращающими одно логическое значение.

В выражениях PostgreSQL дата записывается в одинарных кавычках '2016-10-23'. Об операторах и функциях даты/времени подробнее смотри в справке PostgreSQL^{†††}.

Задание 9. Вывод списка оборудования со сроком службы от 10 лет до 15 лет.

Для выполнения данного задания следует использовать предикат *BETWEEN* (ставится после конструкции *WHERE*), который позволяет задать выражение проверки вхождения какого-либо значения в диапазон, определяемый граничными значениями:

WHERE поле *between* значение1 *and* значение2

Задание 10. Получения списка заявок на запчасти от ремонта от трех заданных подразделений.

Для выполнения данного задания следует использовать предикаты *IN* или *NOT IN* (ставятся после конструкции *WHERE*), которые применяются для проверки вхождения какого-либо значения в заданный список значений.

WHERE поле *IN* (список значений)

Задание 11. Получения списка всех запчастей – газовых клапанов.

Для выполнения данного задания следует использовать предикаты *LIKE* или *NOT LIKE* (ставятся после конструкции *WHERE*), которые применяются для проверки частичного соответствия символьных строк.

WHERE поле *LIKE* 'критерий частичного соответствия';

Критерий частичного соответствия задается с помощью двух символов-масок: знака процента (%) и подчеркивания (_).

Знак процента означает любой набор символов, в том числе и пустой, а символ подчеркивания – любой одиночный символ.

Например:

'%EUROSIT%' – все записи, для которых поле содержит набор символов *EUROSIT* ;

'Крышка комфорки%' – все записи, для которых поле начинается с *Крышка комфорки*.

Задания 12-15.

Задания выполнить самостоятельно.

Вопросы для подготовки к защите лабораторной работы

1. Дайте определение запроса к базе данных?
2. Прокомментируйте понятия результатная таблица, виртуальная таблица?
3. Как выполнить запрос в PgAdminIII?
4. Какова структура команды *SELECT*?
5. Назначение операторов *WHERE* и *GROUP BY* команды *SELECT*.
6. Назначение операторов *HAVING* и *ORDER BY* команды *SELECT*.
7. Каким образом можно вывести на просмотр всю таблицу базы данных?

^{†††} <https://postgrespro.ru/docs/postgresql/9.5/functions-datetime>

8. Как оставить в списке определенные столбцы и задать им имя кириллицей?
9. Как получить список строк таблицы без повторов по содержимому столбца, имеющему повторяющиеся значения?
10. Как выполнить сортировку записей? Что такое сложная сортировка?
11. Назначение секции *WHERE* в команде *SELECT*.
12. Приведите пример использования условия отбора с логическими операторами *AND* и *OR*.
13. Что такое предикаты, какие предикаты вы знаете?
14. Что такое критерий частичного соответствия? Приведите пример его использования.
15. Как сравнивать даты?

Лабораторная работа 6. Реализация действий по сложной обработке и агрегированию информации базы данных

Цель работы: получить практические навыки создания многотабличных, итоговых и сложных запросов к базе данных промышленной СУБД.

Задания

Многотабличные запросы

1. Получить полное декартово произведение таблиц *zaiavki* и *podrazdelenia*. Пояснить недостаток такого способа соединения таблиц.
2. Используя условное соединение, получить список подразделений, подавших заявки на ремонт.
3. Используя сложное условное соединение, получить список из таблицы *zaiavki*, содержащий перечень подразделений с указанием оборудования, требующего ремонта, и запчастей для данного оборудования.

Итоговые запросы с использованием агрегатных функций

4. Определить количество заявок на ремонт с датой ремонта в заданном диапазоне.
5. Определить сумму, требующуюся на покупку запчастей по заявкам с датой ремонта в заданном диапазоне.
6. Рассчитать среднюю цену на запчасти по всем заявкам.
7. Получить список из таблицы *zaiavki*, содержащий перечень подразделений с указанием оборудования, требующего ремонта, запчастей для данного оборудования, суммы на закупку запчастей и итоговой суммы по списку.

Агрегирование с использованием группировки записей

8. Получить список, содержащий перечень подразделений с указанием сумм, необходимых на закупку запчастей для ремонта по всем заявкам от этих подразделений.
9. Получить список, содержащий перечень подразделений с указанием сумм, необходимых на закупку запчастей для ремонта по всем заявкам от этих подразделений, если сумма превышает 500 000 рублей.
10. Получить список, содержащий перечень подразделений с указанием сумм, необходимых на закупку запчастей для ремонта по всем заявкам от этих подразделений с датой ремонта в заданном диапазоне.
11. Получить список, содержащий перечень подразделений с указанием сумм, необходимых на закупку запчастей для ремонта по всем заявкам от этих подразделений с датой ремонта в заданном диапазоне, если сумма превышает 500 000 рублей.
12. Оформить отчет по проделанной работе.

Методические указания к выполнению заданий

Задание 1. Получение полного декартова произведения таблиц *zaiavki* и *podrazdelenia*. В результате выполнения команды *SELECT* для нескольких источников без секций *WHERE* и *JOIN*, уточняющих связи между источниками, возвращается полное декартово

произведение источников. Итоговый набор будет содержать все возможные комбинации записей из всех источников.

Полное декартово произведение может быть получено и с помощью секции CROSS JOIN. Между синтаксисом CROSS JOIN и простым перечислением таблиц через запятую нет никаких функциональных различий. Результат перекрестного соединения принципиально не отличается от перечисления источников через запятую.

Задание выполнить самостоятельно.

Задание 2. Получение списка подразделений, подавших заявки на ремонт, с использованием условного соединения. Устранить повторяющиеся записи.

Условное соединение позволяет соединить таблицы по произвольным столбцам, не обязательно имеющим одинаковые имена. Кроме того, в качестве условия соединения может выступать не только равенство, но вообще любое логическое выражение, которое записывается после ключевого слова ON.

Если условие выполняется для текущей записи декартового произведения исходных таблиц, то она входит в результирующую таблицу.

Пусть в базе данных имеются следующие две таблицы:

Сотрудники (Номер, Фамилия, Имя, Номер_отдела);
Отделы (Номер, Название).

Тогда эти таблицы можно соединить:

```
SELECT *  
FROM Сотрудники JOIN Клиенты  
ON (Номер_отдела = Отделы.Номер);  
Задание выполнить самостоятельно.
```

Примечание: не рекомендуем копировать содержимое SQL запросов непосредственно из текста методических указаний.

Задание 3. Получение списка из таблицы *zaiavki*, содержащего перечень подразделений с указанием оборудования, требующего ремонта, и запчастей для данного оборудования. Использовать сложное соединение.

Сложное соединение представляет собой конструкцию вида:

```
SELECT <список полей через запятую к выводу>  
FROM таблица 1  
JOIN таблица 2 ON условие связи таблиц 1 и 2  
JOIN таблица 3 ON условие связи таблиц 1 и 3  
JOIN таблица 4 ON условие связи таблиц 1 и 4
```

Решение поставленной задачи:

```
SELECT podrazdelenia.naimenovanie, oborudovanie.naimenovanie,  
zapchasti.naimenovanie  
FROM zaiavki  
JOIN podrazdelenia ON id_podrazd_zaiav=id_podrazd  
JOIN oborudovanie ON id_oborud_zaiav=id_oborud  
JOIN zapchasti ON id_zapchasti_zaiav=id_zapchasti;
```

Задание 4. Определение количества заявок на ремонт с датой ремонта в заданном диапазоне.

Когда требуется узнать, сколько записей соответствует тому или иному запросу, какова сумма значений некоторого числового столбца, его максимальное, минимальное и среднее значение, используются так называемые итоговые (статистические, агрегатные) функции.

Агрегатные функции – особый класс функций, применяемых сразу к нескольким записям набора данных, но возвращающим одно значение.

Основные агрегатные функции, поддерживаемые в PostgreSQL, представлены в таблице 5.

Таблица 5 – Основные агрегатные функции, поддерживаемые в PostgreSQL

Функция	Описание
avg(выражение)	Среднее арифметическое значений выражения для всех записей в группе
count(выражение)	Количество записей в группе, для которых значение выражения отлично от NULL
max(выражение)	Максимальное значение выражения в группе
min(выражение)	Минимальное значение выражения в группе
stddev(выражение)	Среднеквадратичное отклонение значений выражения в группе
sum(выражение)	Сумма значений выражения в группе
variance(выражение)	Дисперсия значений выражения в группе

Выражение означает любой столбец в итоговом наборе или любое выражение, выполняющее операцию с этим столбцом.

Полный список агрегатных функций выводится командой \da. Более подробно можно познакомиться с ними в справочной поддержке PostgreSQL^{†††}.

Определим количество заявок на ремонт с датой ремонта от 15.10.16 по 18.11.16. Сначала определим общее количество непустых строк по столбцу *nomer* таблицы *zaiavki*.

```
SELECT count(nomer)
FROM public.zaiavki;
```

Затем определим количество строк с неповторяющимися значениями столбца *nomer*.

```
SELECT count(DISTINCT nomer)
```

```
FROM public.zaiavki;
```

И наконец, решим поставленную задачу:

```
SELECT count(DISTINCT nomer)
FROM public.zaiavki
WHERE data_remonta >='2016-10-15' AND data_remonta <='2016-11-18';
```

Задания 5,6.

Задания выполнить самостоятельно.

Задание 7. Получение списка из таблицы *zaiavki*, содержащего перечень подразделений с указанием оборудования, требующего ремонта, запчастей для данного оборудования, суммы на закупку запчастей и итоговой суммы по списку.

Решим задачу поэтапно.

Получим список без связывания таблиц и без итоговой суммы.

```
SELECT id_podrazd_zaiav, id_oborud_zaiav, id_zapchasti_zaiav,
kol_vo*cena_plan As summa
FROM public.zaiavki;
```

Получим список со связыванием таблиц и без итоговой суммы.

```
SELECT podrazdelenia.naimenovanie, oborudovanie.naimenovanie,
zapchasti.naimenovanie,kol_vo*cena_plan As summa
FROM public.zaiavki
JOIN podrazdelenia ON id_podrazd_zaiav=id_podrazd
JOIN oborudovanie ON id_oborud_zaiav=id_oborud
JOIN zapchasti ON id_zapchasti_zaiav=id_zapchasti
```

^{†††} <https://postgrespro.ru/docs/postgresql/9.5/functions-aggregate>

Рассчитаем итоговую сумму.

```
SELECT sum(kol_vo*cena_plan) As Итого  
FROM public.zaiavki
```

Задание 8. Получение списка, содержащего перечень подразделений с указанием сумм, необходимых на закупку запчастей для ремонта по всем заявкам от этих подразделений.

Решение задачи без связи таблиц.

```
SELECT id_podrazd_zaiav, sum(kol_vo*cena_plan) As Сумма  
FROM public.zaiavki GROUP BY id_podrazd_zaiav;
```

Решение задачи со связыванием таблиц.

```
SELECT podrazdelenia.naimenovanie As Подразделение,  
sum(kol_vo*cena_plan) As Сумма  
FROM public.zaiavki  
JOIN podrazdelenia ON id_podrazd_zaiav=id_podrazd  
GROUP BY podrazdelenia.naimenovanie;
```

Секция GROUP BY указывает на то, что записи объединенного набора данных должны группироваться по наименованию подразделения.

Все записи с одинаковым наименованием подразделения группируются, после чего функция sum() подсчитывает в каждой группе количество непустых значений произведений полей *kol_vo* и *cena_plan* и возвращает результат – сумму этих произведений для объединенных в каждую группу записей для одного подразделения.

Задание 9. Получение списка, содержащего перечень подразделений с указанием сумм, необходимых на закупку запчастей для ремонта по всем заявкам от этих подразделений, если сумма превышает 500 000 рублей.

Результат запроса, сформированный при помощи GROUP BY может быть отфильтрован параметром HAVING.

Предложение HAVING, за которым следует условие отбора, определяет группы строк, которые включаются в результирующую таблицу. Условие отбора будет применяться к каждой из групп, сформированных с помощью секции GROUP BY. Задание выполнить самостоятельно.

Задания 10,11.

Задания выполнить самостоятельно.

Вопросы для подготовки к защите лабораторной работы

1. Что вы понимаете под декартовым произведением таблиц? Приведите примеры.
2. Что такое перекрестное соединение таблиц, как оно соотносится с декартовым произведением таблиц?
3. Можно ли для соединения таблиц *podrazdelenia*, *oborudovanie*, *zapchasti*, *zaiavki* использовать естественное соединения с использованием оператора NATURAL JOIN? Если да, приведите пример.
4. Можно ли для соединения таблиц *podrazdelenia*, *oborudovanie*, *zapchasti*, *zaiavki* использовать естественное соединение с использованием оператора NATURAL JOIN? Если да, приведите пример.
5. Можно ли для соединения таблиц *podrazdelenia*, *oborudovanie*, *zapchasti*, *zaiavki* использовать соединение по именам столбцов с использованием оператора USING? Если да, приведите пример.
6. В чем отличие реализации условного соединения с использованием оператора WHERE и с использованием оператора JOIN ON?
7. Можно ли для соединения таблиц *podrazdelenia*, *oborudovanie*, *zapchasti*, *zaiavki* использовать условное соединение с использованием оператора ON? Если да, приведите пример.

8. Что собой представляет сложное соединение более двух таблиц? Приведите пример использования сложного соединения.
9. Дайте определение агрегатных функций. Какие агрегатные функции вы знаете? Приведите пример их использования.
10. Как происходит подсчет с использованием агрегатной функции при группировке записей оператором GROUP BY? Приведите пример.
11. В чем состоит назначение оператора HAVING при его использовании совместно с оператором GROUP BY? Приведите пример. В чем разница между операторами HAVING и WHERE?

Лабораторная работа 7. Конфигурирование профилей пользователей и объектных привилегий в промышленной СУБД

Цель работы: получить практические навыки управления профилями пользователей и объектными привилегиями пользователей в промышленных СУБД.

Задания

1. Составить таблицы параметров конфигурирования профилей пользователей и объектных привилегий сервера *pitak* и базы данных *OMTC*. Выбрать метод аутентификации пользователей базы данных.
2. Создать групповые роли базы данных с требуемыми атрибутами.
3. Сконфигурировать объектные привилегии групповых ролей базы данных.
4. Создать и сконфигурировать несколько ролей пользователей базы данных.
5. Проверить разграничение объектных привилегий для пользователей разных групп на таблицу *oborudovanie*.
6. Внести изменения в созданную конфигурацию пользователей и объектных привилегий с использованием команд PostgreSQL с проверкой результата их выполнения в среде PgAdmin.
7. Оформить отчет по проделанной работе.

Методические указания к выполнению заданий

Задание 1. Составление таблиц параметров конфигурирования профилей пользователей и объектных привилегий сервера *pitak* и базы данных *OMTC*. Выбор метода аутентификации пользователей базы данных.

Для определения параметров конфигурирования профилей пользователей и объектных привилегий необходимо сформировать четыре таблицы: групповые роли базы данных; привилегии (атрибуты) ролей базы данных; объектные привилегии ролей базы данных; пользователи базы данных.

Примерный вид таблиц представлен ниже (таблицы 6-9). Параметры конфигурирования профилей пользователей и объектных привилегий определяются на основе анализа предметной области и документов, определяющих политику безопасности в области защиты информации в компании.

Таблица 6 – Групповые роли базы данных

№	Имя групповой роли	Описание
1	admin	Администраторы
2	boss	Руководители
3	spec1	Специалисты подразделений

Таблица 7 – Привилегии (атрибуты) групповых ролей базы данных

№	Имя групповой роли	SUPERUSER (Статус суперпользователя)	CREATEDB (Создание базы данных)	CREATEROLE (Создание роли)	REPLICATION LOGIN (Запуск репликации)	INHERIT (Наследование прав из родительских ролей)
1	admin	-	+	+	-	+
2	boss	-	-	-	-	+
3	spec1	-	-	-	-	+

Таблица 8 – Объектные привилегии ролей базы данных

№	Имя групповой роли	Таблица <i>podrazdelenia</i>	Таблица <i>oborudovanie</i>	Таблица <i>zapchasti</i>	Таблица <i>zaiavki</i>
1	admin	ALL PRIVILEGES	ALL PRIVILEGES	ALL PRIVILEGES	ALL PRIVILEGES
2	boss	SELECT	SELECT	SELECT	SELECT
3	spec1	SELECT	SELECT	SELECT	SELECT INSERT UPDATE DELETE

Таблица 9 – Пользователи базы данных

№	Пользователь	Имя роли	Имя групповой роли	Описание
1	Прохоров С.В.	admin1	admin	Администратор, инженер-программист ИВЦ
2	Потапов И.А.	potapov	boss	Директор компании
3	Игнатьев Б.С.	ignatjev	spec1	Специалист по текущему ремонту оборудования

Замечание: На практике пользователям базы данных, не входящим в группу *admin*, не стоит давать привилегию DELETE, а операцию удаления заменить на операцию установки метки «запись удалена».

С целью обеспечения защиты от возможности перехвата трафика выбираем метод аутентификации пользователей базы данных с помощью пароля *md5\$\$\$*.

Задание 2. Создание групповых ролей базы данных с требуемыми атрибутами.

Выполнить задание можно, используя интерфейс утилиты pgAdminIII, как это было сделано в лабораторной работе 1-2. Но если создается большое количество ролей с однотипными атрибутами удобнее выполнить задание с помощью запроса, имеющего следующий синтаксис****:

CREATEROLE <имя роли> WITH <список атрибутов>

Задание 2.1. Создать групповую роль *admin*.

Для выполнения задания в браузере объектов pgAdminIII выделить базу *OMTC*; вызвать редактор SQL и выполнить команду:

CREATE ROLE admin WITH CREATEDB CREATEROLE INHERIT

Убедиться, что групповая роль создана, выполнив команду:

\$\$\$ <https://postgrespro.ru/docs/postgresql/9.5/auth-methods.html>

**** <https://postgrespro.ru/docs/postgrespro/9.5/sql-createrole.html>

```
SELECT rolname FROM pg_roles;
```

Убедиться в правильности задания атрибутов групповой роли *admin* в браузере объектов pgAdminIII.

Задание 2.2. Самостоятельно создать групповые роли *boss* и *spec1*. Привилегии (атрибуты) групповых ролей базы данных определять по таблице 6.

Проверить наличие созданных групповых ролей и убедиться в правильности задания атрибутов.

Задание 3. Конфигурирование объектных привилегий групповых ролей базы данных.

Объектные привилегии устанавливаются для групповой роли с помощью команды GRANT – определителя прав доступа:††††

```
GRANT<список привилегий> ON <объект> TO <роль>;
```

Задание 3.1. Задать права доступа группы пользователей *spec1* на таблицу *zaiavki*.

Согласно распределению объектных привилегий по таблице 7 для выполнения задачи следует выполнить команду:

```
GRANT SELECT, INSERT, UPDATE, DELETE ON zaiavki TO spec1
```

Задание 3.2. Задать права доступа для всех групп пользователей на таблицу базы данных *zaiavki* согласно таблице 7.

Сделать это быстрее и удобнее, задавая привилегии для каждой таблицы в ее свойствах на панели «Привилегии».

Перейти на панель «Привилегии» диалогового окна свойств таблицы *zaiavki*. Выбрать групповую роль. Выбрать привилегии. Выполнить «Добавить/Изменить». Выполнить «Ок»

Задание 3.3. Аналогично задать права доступа для всех групп пользователей на таблицы базы данных *podrazdelenia*, *oborudovanie*, *zapchasti* согласно таблице 7.

Задание 4. Создание и конфигурирование нескольких ролей пользователей (ролей входа) базы данных.

Создать роли пользователей из таблицы 8. Для выполнения задачи используется запросы, имеющие следующие синтаксисы:

```
CREATE ROLE <имя роли> WITH LOGIN PASSWORD 'Пароль';
```

(создание роли с заданным паролем);

```
GRANT<Имя групповой роли>TO<список ролей для включения> ;
```

(включение роли в группу).

В качестве пароля используйте двойное имя роли.

Для создания роли Игнатьева Б.С. выполнить:

```
CREATE ROLE ignatjev WITH LOGIN PASSWORD 'ignatjevignatjev';
```

```
GRANT spec1 TO ignatjev;
```

Остальные роли создать самостоятельно, одну из ролей создать средствами pgAdminIII.

Задание 5. Проверка разграничения объектных привилегий для пользователей разных групп на таблицу *oborudovanie*.

Проверить, имеет ли возможность изменять данные в таблице *oborudovanie* директор Потапов И.А.

Для этого зайти на сервер *pitek* под логином *potapov*:

- отсоединиться от сервера *pitek*;

- в свойствах сервера *pitek* установить имя пользователя *potapov*;

- подключиться к серверу под пользователем *potapov*.

†††† <https://postgrespro.ru/docs/postgrespro/9.5/sql-grant>

Самостоятельно проверить работоспособность защиты от изменений пользователем *potarov* таблицы *oborudovanie*.

Самостоятельно проверить работоспособность защиты от изменений пользователем Круглова А.И. таблицы *zajavki* и работоспособность ее прав на внесение изменений в таблицу *oborudovanie*.

Задание 6. Внесение изменений в созданную конфигурацию пользователей и объектных привилегий с использованием команд PostgreSQL с проверкой результата их выполнения в среде pgAdminIII.

Задание 6.1. Изменить любой атрибут выбранной вами роли.

Для этого используйте следующий синтаксис запроса:

```
ALTERROLE <имя роли> SET <атрибут> TO <целевое значение>;
```

Задание 6.2. Включить или исключить членов в/из групповой роли.

Синтаксис запроса:

```
GRANT <Имя групповой роли> TO <список ролей для включения>;  
REVOKE <Имя групповой роли> FROM <список ролей для исключения>;
```

Задание 6.3. Удалить одну из ролей.

Синтаксис запроса:

```
DROP ROLE <имя роли>;
```

Задание 6.4. Аннулировать любую одну из привилегий пользователя на один из объектов.

Синтаксис запроса:

```
REVOKE <список привилегий> ON <объект> FROM <роль>;  
Результаты выполнения команд PostgreSQL проверить в среде pgAdminIII.
```

Вопросы для подготовки к защите лабораторной работы

1. Что вы понимаете под ролью базы данных? Какие роли базы данных вы знаете?
2. Какая роль предустановлена в СУБД PostgreSQL по умолчанию? Каково ее назначение?
3. Какие атрибуты роли СУБД PostgreSQL вам известны?
4. Что такое групповая роль? В чем ее отличие от просто роли базы данных?
5. Какой смысл вы вкладываете в термин объектные привилегии?
6. Какие параметры используются для описания объектных привилегий ролей базы данных?
7. Назовите основные методы аутентификации пользователей базы данных PostgreSQL.
8. Какие способы создания групповых ролей вы знаете?
9. Приведите пример конфигурирования объектных привилегий с помощью команды GRANT.
10. Как сконфигурировать объектные привилегии средствами pgAdminIII?
11. Какие способы создания ролей входа вы знаете?
12. Каким образом формируется состав групп ролей?
13. Каким образом осуществляется вход в базу данных под разными пользователями?
14. Как изменить атрибут роли?
15. Как можно удалить роль?
16. Как аннулировать привилегию пользователя на объект базы данных?

Лабораторная работа 8. Конфигурирование сетевых соединений и системный аудит в промышленной СУБД

Цель работы: получить практические навыки конфигурирования сетевых соединений и использования системного аудита в промышленных СУБД.

Задания

1. Разработать структуру средств аудита базы данных *ОМТС*.

2. Разработать структуру данных аудита. Для объекта аудита – таблицы *podrazdelenia* создать таблицу *podraz_audit* хранения данных аудита.

3. Запрограммировать функцию триггера аудита *proc_podraz_audit* для таблицы *podrazdelenia*.

4. Создать триггер *podraz_audit* выполнения функции *proc_podraz_audit* для аудита событий, связанных с изменением таблицы *podrazdelenia*.

5. Внести изменения в таблицу *podrazdelenia*, проверить их протоколирование в таблице *podraz_audit*.

6. Выполнить конфигурирование сетевых соединений СУБД PostgreSQL.

7. Оформить отчет по проделанной работе.

Методические указания к выполнению заданий

Задание 1. Разработка структуры средств аудита базы данных ОМТС.

В стандартной поставке PostgreSQL не предусмотрено встроенных средств системного аудита операций с базой данных. Однако создание инструментов системного аудита не является сложной и трудоемкой задачей. Ее выполнение по силам даже начинающему программисту. Одним из наиболее гибких и одновременно наиболее просто реализуемых инструментов, используемых для организации системного аудита в промышленной СУБД PostgreSQL, является триггер аудита.

Для организации системного аудита необходимо:

- определить список объектов базы данных, которые будут подвергаться аудиту;
- разработать структуру данных аудита и для каждого объекта аудита создать таблицу, которая будет хранить эти данные;
- для каждого объекта аудита запрограммировать функцию триггера аудита и создать триггер, который будет выполнять эту функцию всякий раз при наступлении события аудита (например, удаления строки в таблице).

Структура средств аудита базы данных ОМТС сведена в таблицу 10.

Таблица 10 – Структура средств аудита базы данных ОМТС

№	Объект аудита	Действия над объектом, подлежащие аудиту	Таблица для хранения данных аудита	Функция триггера аудита	Триггер аудита
1	Таблица <i>podrazdelenia</i>	DELETE UPDATE INSERT	<i>podraz_audit</i>	<i>proc_podraz_audit</i>	<i>podraz_audit</i>
2	Таблица <i>oborudovanie</i>	DELETE UPDATE INSERT	<i>oborud_audit</i>	<i>proc_oborud_audit</i>	<i>oborud_audit</i>
3	Таблица <i>zapchasti</i>	DELETE UPDATE INSERT	<i>zapch_audit</i>	<i>proc_zapch_audit</i>	<i>zapch_audit</i>
4	Таблица <i>zaiavki</i>	DELETE UPDATE INSERT	<i>zaiavki_audit</i>	<i>proc_zaiavki_audit</i>	<i>zaiavki_audit</i>

Здесь: DELETE – операция удаления строки таблицы; UPDATE – операция редактирования строки таблицы; INSERT – операция вставки строки таблицы.

Задание 2. Разработка структуры данных аудита и создание таблицы *podraz_audit* хранения данных аудита таблицы *podrazdelenia*.

Задание 2.1. Создать схему *audit*.

Создавать таблицы для хранения данных аудита целесообразно в отдельной схеме. Пусть это будет схема *audit*.

Для создания схемы *audit*: в контекстном меню объекта «Схемы» выполнить «Новая схема»; задать имя схемы - *audit*; задать владельца схемы – *pitek14*.

Задание 2.2. Разработать структуру таблицы *podraz_audit*

Структура таблицы *podraz_audit* для хранения данных аудита таблицы *podrazdelenia* представлена в таблице 11.

Обратите внимание на порядок чередования колонок. Для функции триггера аудита важно, чтобы колонка, которая является первичным ключом, стояла последней.

Создать таблицу *podraz_audit* можно так, как мы это делали в лабораторной работе 5-6, а можно быстрее, выполнив соответствующие SQL-запросы.

Таблица 11 – Структура таблицы *podraz_audit* для хранения данных аудита таблицы *podrazdelenia*

№	Имя колонки	Тип данных	Ограничения	Примечание
1	<i>operation</i>	character(1)	NOT NULL	Выполненная операция: D – удаление; U – редактирование; I – вставка.
2	<i>stamp</i>	timestamp without time zone	NOT NULL	Временная метка наступления события аудита.
3	<i>userid</i>	text	NOT NULL	Имя роли пользователя (входа), выполнившего операцию.
4	<i>data_text</i>	text	NOT NULL	Содержимое строки таблицы <i>podrazdelenia</i> , подвергшееся изменению
5	<i>id_podraz_audit</i>	integer	Первичный ключ PRIMARY KEY	Идентификатор строки. Имя последовательности: <i>seq_podraz_audit</i>

Задание 2.3. Создать последовательность *seq_podraz_audit*.

В редакторе SQL выполнить запрос:

```
CREATE SEQUENCE audit.seq_podraz_audit
INCREMENT 1
MINVALUE 1
MAXVALUE 100000
START 1
CACHE 1;
ALTER TABLE audit.seq_podraz_audit
OWNER TO pitek14;
```

Обратите внимание, что последовательность *seq_podraz_audit* создана в новой схеме *audit*, а не в схеме по умолчанию – *public*.

Теперь, чтобы PostgreSQL знал, где искать объекты новой схемы, мы будем использовать сложные имена типа

audit.<имя_объекта>

Задание 2.4. Создать таблицу *podraz_audit*.

В редакторе SQL выполнить запрос:

```
CREATE TABLE audit.podraz_audit
(
```

```
operation character(1) NOT NULL,  
stamp timestamp without time zone NOT NULL,  
userid text,  
data_text text,  
id_podraz_audit integer NOT NULL DEFAULT nextval('audit.seq_podraz_audit'::regclass),  
CONSTRAINT podraz_audit_pkey PRIMARY KEY (id_podraz_audit)  
)  
WITH (OIDS=FALSE);  
ALTER TABLE audit.podraz_audit  
OWNER TO pitek14;
```

Задание 3. Программирование функции триггера аудита для таблицы *podrazdelenia*.††††

Триггер является указанием, что база данных должна автоматически выполнить некоторую заданную функцию (*триггерную функцию, функцию триггера*), всякий раз, когда выполнен определённый тип операции.

Например, для таблиц базы данных можно определять триггеры, которые будут срабатывать до или после любой из команд INSERT, UPDATE или DELETE; либо один раз для каждой модифицируемой строки, либо один раз для оператора SQL.

Триггерная функция должна быть создана до триггера. Она должна быть объявлена без аргументов и возвращать тип *trigger*.

Триггерные функции могут быть написаны на большинстве доступных процедурных языков, включая PL/pgSQL, PL/Tcl, PL/Perl и PL/Python).

После создания триггерной функции создаётся триггер с помощью CREATE TRIGGER. Одна и та же триггерная функция может быть использована для нескольких триггеров.

Создать триггерную функцию *proc_podraz_audit*. В редакторе SQL выполнить скрипт: CREATE OR REPLACE FUNCTION audit.proc_podraz_audit()

```
RETURNS trigger AS  
$BODY$  
BEGIN  
-- Создаем строку в таблице podraz_audit схемы audit, которая отражает выполненную  
операцию.  
-- Воспользуемся переменной TG_OP для определения типа операции.  
IF (TG_OP = 'DELETE') THEN  
INSERT INTO audit.podraz_audit SELECT 'D', now(), user, ROW(OLD.*);  
RETURN OLD;  
ELSIF (TG_OP = 'UPDATE') THEN  
INSERT INTO audit.podraz_audit SELECT 'U', now(), user, ROW(NEW.*);  
RETURN NEW;  
ELSIF (TG_OP = 'INSERT') THEN  
INSERT INTO audit.podraz_audit SELECT 'I', now(), user, ROW(NEW.*);  
RETURN NEW;  
END IF;  
RETURN NULL; -- возвращаемое значение для триггера AFTER не имеет значения  
END;  
$BODY$  
LANGUAGE plpgsql VOLATILE  
COST 100;
```

†††† <https://postgrespro.ru/docs/postgrespro/9.5/triggers.html>

```
ALTER FUNCTION audit.proc_podraz_audit()
    OWNER TO pitek14;
```

Задание 4. Создание триггера *podraz_audit*.

Триггер *podraz_audit* является триггером типа AFTER.

Он должен запускать выполнения функции *proc_podraz_audit* каждый раз после того, как пользователь внес изменения в данные, хранящиеся в таблице *podrazdelenia*.

Создать триггер *podraz_audit*. В редакторе SQL выполнить запрос:

```
CREATE TRIGGER podraz_audit
    AFTER INSERT OR UPDATE OR DELETE
    ON public.podrazdelenia
    FOR EACH ROW
    EXECUTE PROCEDURE audit.proc_podraz_audit();
```

Задание 5. Проверка работоспособности аудита таблицы *podrazdelenia*.

В лабораторной работе 13-14 мы дали права на редактирование таблицы *podrazdelenia* администратору, инженеру-программисту ИВЦ Прохорову С.В. (имя роли- *admin1*, пароль – *admin1admin1*).

Создать роль входа *admin1* с правами групповой роли *admin* (см. лаб. раб. 13-14).

Для того чтобы при входе на сервер *pitek* под ролью *admin1* была возможность внесения изменений в таблицу аудита, необходимо сконфигурировать права пользователя *admin1* на объекты схемы *audit*.

Для схемы *audit* для групповой роли *admin* установить привилегии USAGE и CREATE. Для последовательности *seq_podraz_audit* для групповой роли *admin* установить привилегию UPDATE.

Для проверки работоспособности аудита таблицы *podrazdelenia* :

войти на сервер *pitek* под пользователем *admin1*; вставить новую строку с информацией о новом подразделении; внести изменения в фамилию и инициалы руководителя; удалить эту строку; проверить информацию, записанную в таблицу аудита *podraz_audit* (убедиться, что таких прав у пользователя *admin1* нет); проверить информацию, записанную в таблицу аудита *podraz_audit* под пользователем *pitek14*.

Задание 6. Конфигурирование сетевых соединений СУБД PostgreSQL.

Универсальным способом конфигурирования сетевых соединений в промышленных СУБД, в том числе и СУБД PostgreSQL, является модификация содержимого конфигурационных файлов§§§§. Изменяя содержимое конфигурационного файла PostgreSQL добиться сетевой конфигурации с параметрами, представленными в таблице 12*****. Перед внесением изменений в конфигурационный файл сохранить его резервную копию.

Таблица 12 – Конфигурация сетевых соединений

Параметр конфигурации	Требуемое значение
Прослушиваемый адрес соединения	Любые адреса протокола IPv6
TCP-порт, открываемый сервером	49001
Максимальное число одновременных подключений	70
Количество слотов, зарезервированных для суперпользователей	5
Максимальное время для завершения аутентификации	30 секунд

§§§§ <https://postgrespro.ru/docs/postgresql/9.5/config-setting.html#CONFIG-SETTING-CONFIGURATION-FILE>

**** <https://postgrespro.ru/docs/postgresql/9.5/runtime-config-connection.html#RUNTIME-CONFIG-CONNECTION-SETTINGS>

Использование SSL	Выключено
Шифрование пароля	Включено

После завершения конфигурации сетевого соединения проверить работоспособность сервера СУБД. Для этого перезапустить сервис. По окончании выполнения задания - вернуть настройки к значениям по умолчанию.

Вопросы для подготовки к защите лабораторной работы

1. Что вы понимаете под системным аудитом? Для каких целей он необходим?
2. Какие действия необходимо выполнить для организации системного аудита базы данных?
3. Что входит в состав средств системного аудита?
4. Что такое триггер? Триггерная функция?
5. Какие разновидности триггеров вы знаете?
6. Как создать триггерную функцию?
7. Какие действия выполняет триггерная функция?
8. Как создается триггер?
9. Каким образом можно проверить работоспособность системного аудита?
10. Каким образом следует сконфигурировать права пользователей на доступ к таблице, содержащей данные аудита?
11. Каким образом выполняется конфигурирование сетевых соединений PostgreSQL? Какие параметры конфигурации можно изменить?
12. Какие преимущества дает использование SSL? Какие действия необходимо предпринять для обеспечения возможности его использования?

Лабораторная работа 9. Основные команды языков расширения промышленных СУБД

Цель работы: получить практические навыки использования основных команд языков расширения промышленных СУБД, получить знания о возможностях администрирования промышленных СУБД с помощью языков расширения, научиться создавать резервные копии конфигурации сервера и восстанавливать конфигурацию из резервной копии.

Задания

1. Используя язык PL/pgSQL реализовать функцию, которая позволяет получить список оборудования с выдачей сообщения «Срок службы превышает N лет», если данное условие выполняется, или сообщение «Срок службы не превышает N лет», если данное условие не выполняется.
2. Реализовать на языке PL/pgSQL функцию, выполняющую одну из задач администрирования (по заданию преподавателя).
3. Создать резервную копию сервера *pitek*.
4. Восстановить конфигурацию сервера *pitek* из резервной копии.
5. Оформить отчет по проделанной работе.

Методические указания к выполнению заданий

Задание 1. Реализация функций на языке PL/pgSQL.

Для создания функции в редакторе SQL выполнить скрипт:

```
CREATE OR REPLACE FUNCTION public.spisok(integer,integer)
RETURNS text AS
$BODY$
DECLARE
    val_srok1 ALIAS FOR $1;
    val_srok2 ALIAS FOR $2;
```

```
val text;  
val1 text;  
val2 text;  
BEGIN  
Val1:='Срок службы превышает ' || to_char(val_srok2, '999') || ' лет';  
Val2:='Срок службы не превышает ' || to_char(val_srok2, '999') || ' лет';  
IF val_srok1 > val_srok2 THEN  
    val := val1 ;  
else  
    val := val2;  
END IF;  
RETURN val;  
END;  
$BODY$  
LANGUAGE 'plpgsql' VOLATILE
```

Убедиться, что функция создана (рисунок 5).

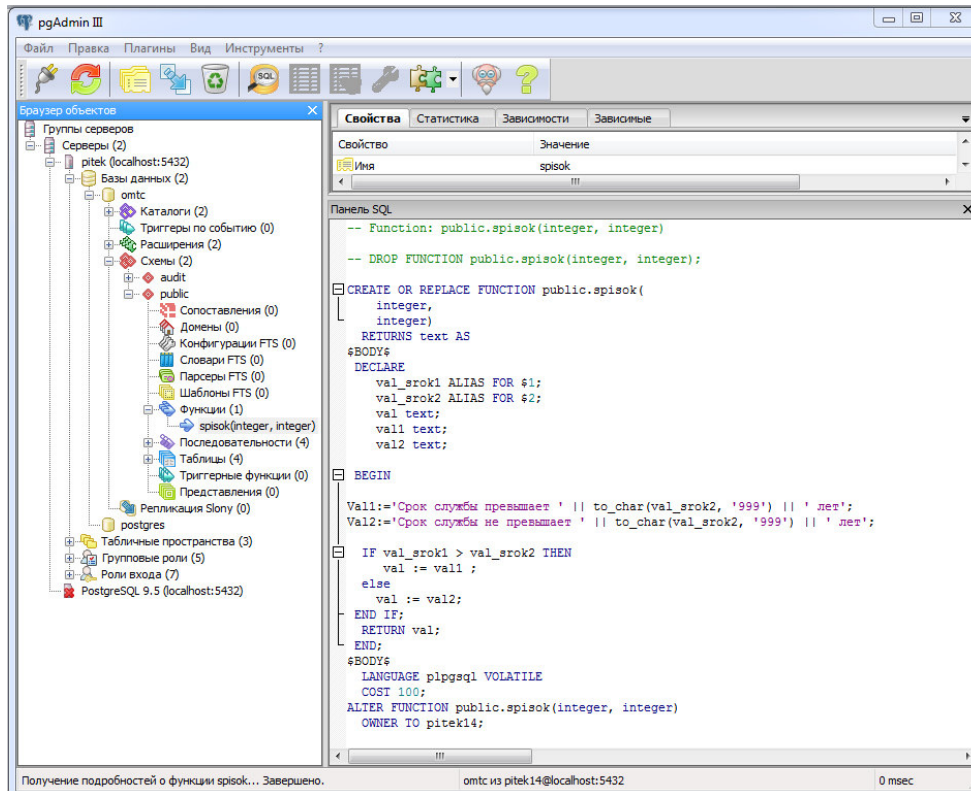


Рисунок 5 –Функция *spisok* в браузере объектов PostgreSQL

Запустить функцию на исполнение, выполнив запрос:

```
SELECT *, public.spisok(srok,12) as Comment  
FROM public.oborudovanie
```

Ожидаемый результат выполнения функции *spisok* представлен на рисунке 6.

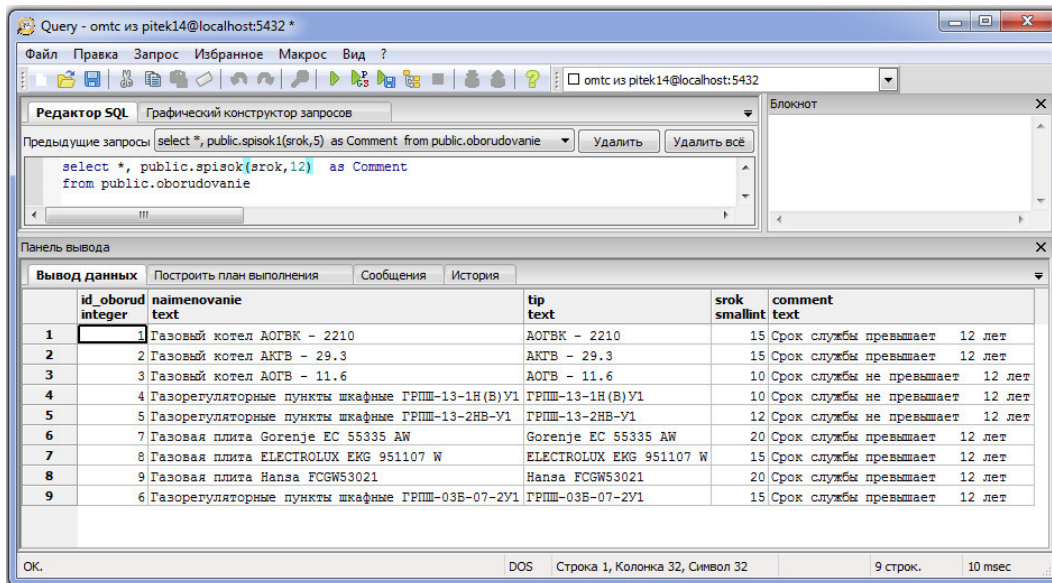


Рисунок 6 – Результат выполнения функции *spisok*

Задание 2. Реализация на языке PL/pgSQL функции, выполняющей одну из задач администрирования.

Задание выполнить самостоятельно.

Задание 3. Создание резервной копии сервера *pitek*.

Создать резервную копию сервера можно в pgAdminIII, для этого следует выполнить команду «Резервная копия» к контекстном меню сервера.

Команда «Резервная копия» запускает создание копии утилитой *pg_dumpall*†††††.

Утилита *pg_dumpall* предназначена для записи ("выгрузки") всех баз данных кластера PostgreSQL в один файл в формате скрипта с расширением *sql*. Этот скрипт содержит команды SQL. Выполнение такого скрипта позволяет восстановить все схемы, базы данных, роли, права доступа и табличные пространства.

Принято называть такого рода резервную копию *дампом* базы данных.

Дамп базы данных - это файл, содержащий инструкции на языке SQL, за счет которых создается точная копия базы данных, как по содержанию, так и по структуре.

Выполнить скрипт в среде pgAdminIII нельзя, восстановление осуществляется путем запуска скрипта утилитой *psql*. Для получения полного содержимого баз запускать процедуру создания резервного дампа нужно от имени суперпользователя.

Создать резервную копию сервера *pitek*.

Для этого: выделить сервер; в контекстном меню выполнить «Резервная копия сервера ...»; задать имя *sql*-скрипта и место его расположения; задать имя роли (роль должна обладать правами суперпользователя); выполнить «Резервная копия»

Задание 4. Восстановление конфигурации сервера *pitek* из резервной копии.

Восстановление резервной копии, как уже было отмечено выше, осуществляется путем запуска скрипт, полностью воспроизводящего кластер PostgreSQL, утилитой *psql*.

Перед восстановлением на сервере, куда будет восстанавливаться кластер, необходимо наличие следующих объектов конфигурации:

- роль суперпользователя, под которым создавался дамп (*pitek14*);
- табличные пространства и их каталоги на диске (*ptk*);
- «пустые» базы данных (*ОМТС*).

††††† <https://postgrespro.ru/docs/postgresql/9.5/app-pg-dumpall>

Восстановить резервную копию сервера *pitek* из дампа, созданного в пункте 3 задания. Запустить утилиту *psql*. Экранная форма утилиты *psql* представлена на рисунке 7.

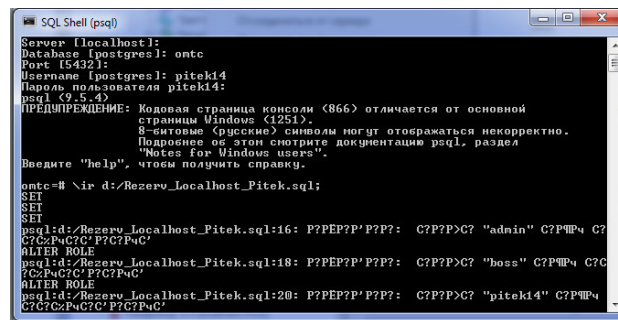


Рисунок 7 – Экранная форма утилиты *psql*

Зайти на сервер под ролью *pitek14*.

Запустить на исполнение скрипт восстановления сервера командой:

```
\ir d:/Rezerv_Localhost_Pitek.sql;
```

или

```
\i d:/Rezerv_Localhost_Pitek.sql;
```

После завершения восстановления убедиться в работоспособности восстановленного сервера.

Вопросы для подготовки к защите лабораторной работы

1. Для чего используются языки расширения промышленных СУБД? Какие языки расширения вы знаете?
2. Какие условия должны быть выполнены, чтобы иметь возможность использовать язык расширения?
3. Как создается функция на языке расширения?
4. Как осуществить вызов функции?
5. Каким образом осуществляется передача параметров и возвращение значений при вызове функции?
6. Каким образом можно создать резервную копию всего кластера PostgreSQL?
7. Что такое дамп?
8. Каким образом выполняется восстановление кластера PostgreSQL?
9. Можно ли восстановить кластер PostgreSQL в «пустую» конфигурацию, устанавливаемую при инсталляции PostgreSQL?

Полный комплект методического обеспечения лабораторных работ по дисциплине «Администрирование промышленных СУБД» в формате pdf-файлов расположен на кафедральных ресурсах в аудитории 210. Преподаватель, ведущий лабораторные работы, выдает раздаточный материал в начале семестра

МЕТОДИЧЕСКОЕ ОБЕСПЕЧЕНИЕ КУРСОВОЙ РАБОТЫ ПО ДИСЦИПЛИНЕ

1 Цели и задачи курсовой работы

Цель курсовой работы по дисциплине «Администрирование промышленных СУБД»: сформировать навыки применения на практике знаний в области администрирования промышленных СУБД.

Тема курсовой работы – «Разработка базы данных для информационной системы предприятия ТЭК».

Для достижения поставленной цели при выполнении курсовой работы студентами решаются следующие задачи:

- провести описание объекта информатизации – предприятия ТЭК и анализ предметной области;
- провести сравнительный анализ автоматизированных информационных систем и СУБД, на основе которых они функционируют, используемых в данной предметной области;
- конкретизировать постановку задачи, сформировать требования к разрабатываемой базе данных для рассматриваемой информационной системы;
- разработать модели, необходимые при проектировании базы данных для информационной системы;
- разработать базу данных для информационной системы предприятия ТЭК в соответствии со сформированными требованиями.

На выполнение курсовой работы выделяется 8 часов в 6 семестре профиля «Прикладная информатика в топливно-энергетическом комплексе» (направления «Прикладная информатика»).

2. Требования к содержанию курсовой работы

2.1 Объем и содержание курсовой работы

Общий объем пояснительной записки не должен превышать 25 – 30 страниц, в том числе введение – не более 1-2 страниц, заключение – 1 страница.

Пояснительная записка к курсовой работе должна давать достаточно полное представление о принципе решения задачи разработки базы данных для информационной системы с обоснованием правильности решения задачи на ЭВМ.

Типовая структура пояснительной записки курсовой работы:

- титульный лист;
- содержание;
- введение;
- разделы и подразделы основной части;
- заключение;
- список использованных источников;
- приложения (при необходимости);
- задание на курсовую работу.

2.2 Содержание основных разделов пояснительной записки

Титульный лист должен соответствовать установленному образцу.

Содержание должно включать наименование всех разделов курсовой работы, а также подразделов и пунктов, если они имеют наименование, с указанием номера страниц, на которых размещается начало материала разделов, подразделов и пунктов.

Введение должно содержать: постановку задачи, анализ актуальности и цели разработки информационной системы, описание структуры пояснительной записки.

Основная часть включает в себя выполнение заданий, представленных в таблице 1 и содержит два раздела.

Первый раздел посвящается описанию объекта информатизации (предприятия ТЭК) и анализу предметной области – деятельности структурного подразделения предприятия ТЭК, которое будет использовать разрабатываемую базу данных.

Рекомендуемая примерная тематика первого раздела.

- общие сведения о предприятии ТЭК (история, организационно-правовая форма, виды деятельности, организационная структура, руководство и т.д.);

- анализ предметной области на уровне абстракции внешнего моделирования - деятельности структурного подразделения предприятия ТЭК, которое будет использовать разрабатываемую базу данных (структура, задачи и функции, описание бизнес-процессов, схем передачи информации и т.д.);

- обзор и сравнительная оценка автоматизированных информационных систем и СУБД, на основе которых они функционируют, используемых в данной предметной области по критериям, важным для разработки базы данных для информационной системы рассматриваемого предприятия ТЭК.

Второй раздел посвящен вопросам реализации практической части курсовой работы по разработке базы данных для информационной системы предприятия ТЭК.

Рекомендуемая примерная тематика второго раздела.

- разработка структуры базы данных, построение схемы базы данных;
- реализация запросов по обработке информации;
- конфигурирование и настройка операционной среды СУБД;
- администрирование доступа (создание профилей пользователей, установка объектных привилегий пользователей);
- организация системного аудита;
- результаты тестирования.

Во второй раздел могут помещаться небольшие фрагменты программного кода (запросы, процедуры и т.д.).

В заключении необходимо отразить основные результаты, полученные в курсовой работе, сформулировать выводы по проделанной работе, зафиксировать степень достижения поставленных во введении цели и задач.

Список использованных источников должен содержать библиографическое описание директивных и нормативно-методических материалов, научных, учебных и периодических изданий, информационных ресурсов, используемых при написании работы. На все приводимые литературные источники в тексте должны быть ссылки. Список использованных источников должен быть оформлен в соответствии с ГОСТом.

Приложения оформляются как продолжение пояснительной записки. В качестве приложения рекомендуется подготовить глоссарий – словарь основных терминов и понятий (не более 20). В остальные приложения целесообразно вынести содержимое информационных баз, листинги больших фрагментов программного кода.

3 Задание на курсовую работу

В процессе выполнения курсовой работы студент должен разработать базу данных для информационной системы предприятия ТЭК, последовательно решив поставленные за-

дачи. Все задания взаимосвязаны между собой и образуют строгую логическую последовательность, т.е. выполнять каждое последующее задание следует с опорой на результаты предыдущего.

Таблица 1 – Задания на курсовую работу и сроки представления результатов их выполнения для промежуточной проверки

Задания	Срок сдачи	% выполнения
1. Выполнить анализ объекта информатизации и предметной области.	4 неделя	10
2. Провести сравнительный анализ автоматизированных информационных систем и СУБД, на основе которых они функционируют, используемых в данной предметной области.	6 неделя	20
3. Конкретизировать постановку задачи, сформировать требования к разрабатываемой базе данных для рассматриваемой информационной системы. Составить техническое задание на разработку базы данных для информационной системы предприятия ТЭК.	7 неделя	30
4. Разработать структуру и схему базы данных.	8 неделя	40
5. Реализовать запросы по обработке информации.	11 неделя	60
6. Настроить авторизованный доступ к базе данных и инструменты системного аудита. Провести тестирование разработанной базы данных.	13 неделя	80
7. Представить окончательный вариант пояснительной записки на проверку.	14 неделя	100

В зависимости от масштаба предприятия ТЭК допускается ограничиться

- на стадии концептуального моделирования – уровнем деятельности подразделения предприятия;
- на физическом уровне при разработке структуры и схемы базы данных – уровнем реализуемой задачи.

Информационная база должна содержать следующий минимальный набор объектов:

- вспомогательные таблицы-справочники (не менее трех);
- таблицы с целевыми полями (не менее одной).

Количество связей между таблицами – не менее трех.

СУБД – PostgreSQL.

Информация, хранящаяся в базе данных должна иметь отношение к предметной области. Объем данных в базе должен обеспечивать возможность проведения тестирования и проверки правильности формирования запросов и выполнения реализуемых действий по обработке информации.

К пояснительной записке (в конце) прилагается CD с электронным вариантом записки и файлами резервной копии разработанной базы данных.

По согласованию с преподавателем студент может выполнить индивидуальное задание, по тематике и сложности соответствующее предъявляемым требованиям.

4. Примерные темы курсовой работы

1. Разработка базы данных для информационной системы учёта кадров предприятия ТЭК.
2. Разработка базы данных для информационной системы учёта услуг, оказываемых предприятием ТЭК.
3. Разработка базы данных для информационной системы себестоимости производства на предприятии ТЭК.
4. Разработка базы данных для информационной системы учёта амортизации оборудования предприятия ТЭК.
5. Разработка базы данных для информационной системы расчёта заработной платы на предприятии ТЭК.
6. Разработка базы данных для информационной системы планирования потребностей в материалах на предприятиях ТЭК.
7. Разработка базы данных для информационной системы учёта запасов на предприятиях ТЭК.
8. Разработка базы данных для информационной системы бухгалтерского учёта для предприятия ТЭК.
9. Разработка базы данных для информационной системы управления сбытом на предприятиях ТЭК.
10. Разработка базы данных для информационной системы управления цепочками поставок предприятиями ТЭК.
11. Разработка базы данных для информационной системы управления сервисным обслуживанием на предприятиях ТЭК.
12. Разработка базы данных для информационной системы по учёту спецификаций производимой продукции на предприятиях ТЭК.
13. Разработка базы данных для информационной системы планирования ресурсов на предприятиях ТЭК.
14. Разработка базы данных для информационной системы планирования производственных мощностей на предприятиях ТЭК.
15. Разработка базы данных для информационной системы учёта заказов на предприятиях ТЭК.
16. Разработка базы данных для информационной системы учёт ремонта и профилактических работ оборудования на предприятиях ТЭК.
17. Разработка базы данных для информационной системы учёта сырья на предприятии ТЭК
18. Разработка базы данных для информационной системы учёта формирования и ведения штатного расписания на предприятиях ТЭК.
19. Разработка базы данных для информационной системы индивидуальных программ обучения и служебного продвижения на предприятиях ТЭК.
20. Разработка базы данных для информационной системы учёта основных технологических операций на предприятиях ТЭК.

5 Критерии оценивания курсовой работы

При оценивании курсовой работы принимаются во внимание следующие критерии:

- соблюдение сроков выполнения заданий;
- правильность выполнения заданий;
- качество оформления отчета.

6 Оформление пояснительной записки по курсовой работе

Курсовая работа оформляется в виде пояснительной записки с соответствующими расчетами, формулами, диаграммами, схемами, таблицами и другими материалами, выполняется полностью с применением печатающих и графических устройств вывода компьютера на одной стороне листа формата А4 по ГОСТ 9327-60 (297×210 мм) через полуторный межстрочный интервал шрифтом TimesNewRoman – 14, с полями: правое – 10 мм, верхнее и нижнее – 20 мм, левое – 30 мм.

Цвет шрифта должен быть черным, высота букв, цифр и других знаков - кегль 14. Полужирный шрифт не применяется. Заголовки таблиц и рисунков печатать через один интервал. Разрешается использовать компьютерные возможности акцентирования внимания на определенных терминах, формулах, теоремах, применяя шрифты разной гарнитуры. В тексте работы буквы, цифры, линии и знаки должны быть четкими. Повреждения листов работы, помарки и следы не полностью удаленного прежнего текста (графики) не допускаются.

Текст работы делится на абзацы, каждый из которых содержит законченную мысль. Не следует превращать текст в тезисы, начиная каждое предложение с абзаца. Как правило, на одной странице располагается 2-3 абзаца. Не следует злоупотреблять нумерованными и маркированными списками. Лучше использовать перечисления через запятую и конструкции: «во-первых,..., во-вторых...»; «с одной стороны..., с другой стороны...» и т.п.

Фамилии, названия экономических субъектов, торговые марки, название программных продуктов и другие имена собственные в работе приводят на языке оригинала. Допускается транслитерировать имена собственные и приводить названия экономических субъектов в переводе с добавлением (при первом упоминании) оригинального названия. Недопустимо использование без особой необходимости (например, при цитировании) разговорных выражений, подмены профессиональных терминов их бытовыми аналогами. При описании тех или иных процессов, явлений не стоит прибегать к приемам художественной речи, злоупотреблять метафорами. Как правило, при выполнении научных исследований повествование ведется от первого лица множественного числа («Мы полагаем», «По нашему мнению») или от имени третьего лица («Автор считает необходимым», «По мнению автора»).

Наименования структурных элементов работы «СОДЕРЖАНИЕ», «ВВЕДЕНИЕ», «ЗАКЛЮЧЕНИЕ», «СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ», «ПРИЛОЖЕНИЕ» необходимо располагать в середине строки без точки в конце и печатать прописными буквами. Каждый структурный элемент и раздел следует начинать с новой страницы, при этом предшествующая страница должна быть заполнена не менее чем наполовину. Заголовки разделов и подразделов основной части отчета по практике должны четко и кратко отражать их содержание. Если заголовок состоит из двух предложений, их разделяют точкой. Разделы должны иметь порядковую нумерацию арабскими цифрами. Их наименования располагают в середине строки без точки после цифры и в конце, печатают прописными буквами. Подразделы должны иметь нумерацию в пределах каждого раздела. Номер подраздела состоит из номеров раздела и подраздела, разделенных точкой: 1.1, 1.2, 1.3 и т.д. В конце номера подраздела точка не ставится. Подразделы следует записывать с абзацного отступа. Названия подразделов отделяются от названия разделов и от текста пустой строкой.

Страницы текста отчета следует нумеровать арабскими цифрами, соблюдая сквозную нумерацию. Номер страницы проставляют в центре нижней части листа без точки. Титульный лист включают в общую нумерацию страниц работы, при этом номер страницы на нем не проставляют. Иллюстрации и таблицы на листе формата А3 учитывают как одну страницу.

Ссылки на источники являются обязательным элементом любой работы, содержат точные сведения о заимствованных автором источниках. В тексте отчета необходимо сопровождать ссылками не только цитаты, но и любое заимствование из литературы, статистических сборников, справочников и иных источников информации. Ссылки на использованные источники следует указывать порядковым номером библиографического описания источника в

списке использованных источников. Порядковый номер ссылки заключают в квадратные скобки. Нумерация ссылок ведется арабскими цифрами в порядке приведения ссылок в тексте работы независимо от ее деления на разделы.

Цитата в тексте работы приводится в кавычках, а в скобках указывается порядковый номер источника в списке использованных источников и номер страницы в тексте источника, например, [32, с.3]. Если дается свободный пересказ принципиальных положений трудов тех или иных авторов, то в скобках указывается только порядковый номер источника по списку использованных источников без указания номера страницы. Если ссылаются на многотомное издание, то, кроме того, указывают номер тома: [12, Т.3, с.115]. При ссылке на работы одного автора или на работы нескольких авторов в квадратных скобках приводят порядковые номера этих работ из списка через запятую.

Примеры оформления ссылок:

А.Г. Грязнова [23] и Д.А. Ендовицкий [32] считают

В своей работе П. Друкер [56, с.11] пишет «...».

Ряд авторов [14,23,52] считает ...

Список использованных источников формируется в порядке появления ссылок в тексте отчета по практике, нумеруется арабскими цифрами без точки и печатается с абзацного отступа. Список должен содержать сведения об источниках, использованных при написании работы. Сведения об источниках приводятся в соответствии с требованиями Национального стандарта РФ «Библиографическая ссылка. Общие требования и правила составления» ГОСТ Р 7.0.5–2008. Основным источником данных для библиографического описания использованных книг (брошюр) являются сведения, указанные на обороте их титульного листа..

Уравнения и формулы следует выделять из текста в отдельную строку. Для оформления формул следует использовать редактор формул. Выше и ниже каждой формулы или уравнения должно быть оставлено не менее одной свободной строки. Пояснение значений символов и числовых коэффициентов следует приводить непосредственно под формулой в той же последовательности, в которой они даны в формуле. После формулы следует ставить запятую, затем с новой строки набрать слово «где» (без двоеточия) и далее располагать пояснения значений символов и числовых коэффициентов, отделяемых друг от друга точкой с запятой. Формулы следует нумеровать порядковой нумерацией в пределах всей работы арабскими цифрами в круглых скобках, например, (1), в крайнем правом положении на строке, на которой указана формула. Допускается нумерация формул в пределах раздела. В этом случае номер формулы состоит из номера раздела и порядкового номера формулы, разделенных точкой, например, (3.1).

Формулы, помещаемые в приложениях, должны нумероваться отдельной нумерацией арабскими цифрами в пределах каждого приложения с добавлением перед каждой цифрой обозначения приложения, *например*, «формула (В.1)».

Ссылки в тексте на порядковые номера формул дают в скобках.

Например, «...в формуле (1)».

Таблицы применяют для лучшей наглядности и удобства сравнения показателей. Таблицы сопровождают текстом, который должен предшествовать им, содержать анализ и не повторять приведенные в них данные. Таблицу следует располагать непосредственно после текста, в котором она упоминается впервые, или на следующей странице. На все таблицы должны быть ссылки. Ссылаться на таблицу нужно в том месте текста, где формируется положение, подтверждаемое ею. При ссылке следует писать слово «таблица» с указанием ее номера. Таблицы нумеруются в пределах раздела: номер таблицы состоит из номера раздела и порядкового номера таблицы, разделенных точкой, например, Таблица 2.1. Таблицы каждого приложения обозначают отдельной нумерацией арабскими цифрами с добавлением перед цифрой обозначения приложения, например, Таблица В.1.

Наименование таблицы следует помещать над таблицей слева, без абзацного отступа в

одну строку с ее номером через тире. В конце наименования точка не ставится. Наименование таблицы должно отражать ее содержание, быть точным, кратким. Текст после таблицы следует отделять от таблицы пустой строкой.

Целые числа в таблицах не должны быть многозначными. Для этого надо пользоваться соответствующими степенями размерности – тыс., млн., млрд. и т.д. Если цифровые или иные данные в какой-либо строке таблицы не приводят, то в ней ставят прочерк. Дробные числа в таблицах приводят в виде десятичных дробей. При этом числовые значения в пределах одной графы должны иметь одинаковое количество десятичных знаков (также в том случае, когда после целого числа следуют нули, например, 100,0). Показатели могут даваться через тире (30-40 и т.д.), со словами «свыше» (св.30), «от» (от 20), «до» (до 10) и т.п.

Таблицы следует размещать так, чтобы их можно было читать без поворота работы. Если это невозможно, таблицы располагают так, чтобы для их чтения надо было повернуть работу по часовой стрелке на 90 градусов. Допускается применять размер шрифта в таблице меньший, чем в тексте.

Таблицы следует разграничивать по объему. Громоздкие таблицы (более 1 страницы) должны быть вынесены в приложения. Если таблица не помещается целиком на одном листе (странице), ее можно перенести на следующий лист (страницу). При переносе части таблицы на другой лист (страницу) слово «Таблица», ее номер и наименование указывают один раз слева над первой частью таблицы, а над другими частями также слева пишут слова «Продолжение таблицы» и указывают номер таблицы, например, «Продолжение таблицы 1».

Таблицу с большим количеством граф допускается делить на части и помещать одну часть под другой в пределах одной страницы. Если строки и графы таблицы выходят за формат страницы, то в первом случае в каждой части таблицы повторяется головка, во втором случае - боковик. При делении таблицы на части допускается ее головку или боковик заменять соответственно номером граф и строк. При этом нумеруют арабскими цифрами графы и (или) строки первой части таблицы. Заголовки граф и строк таблицы следует писать с прописной буквы в единственном числе, а подзаголовки граф — со строчной буквы, если они составляют одно предложение с заголовком, или с прописной буквы, если они имеют самостоятельное значение. Заголовки граф, как правило, записывают параллельно строкам таблицы. При необходимости допускается перпендикулярное расположение заголовков граф. Заголовки и подзаголовки граф можно выполнять через один межстрочный интервал.

Все иллюстрации (чертежи, графики, схемы, компьютерные распечатки, диаграммы, фотоснимки) называются рисунками. Их количество определяется содержанием работы и должно быть достаточным для того, чтобы придать излагаемому тексту ясность и конкретность. Рисунки следует располагать непосредственно после текста, в котором они упоминаются впервые, или на следующей странице. На все рисунки должны быть даны ссылки в работе. При ссылках на рисунки следует писать «в соответствии с рисунком 2.1», «как следует из рисунка 3.2», «показано на рисунке 1.4».

Слово «рисунок» и его наименование располагают посередине строки, печатаются с прописной буквы размером шрифта 14 пунктов, через один межстрочный интервал. Рисунки нумеруются в пределах раздела: номер рисунка состоит из номера раздела и порядкового номера рисунка, разделенных точкой. Рисунки приложения обозначают отдельной нумерацией арабскими цифрами с добавлением перед цифрой обозначения приложения. Например, Рисунок Г.2.

Основные требования к графику – максимальное использование площади листа, минимум надписей, рациональный выбор масштаба по осям, использование множителей и приставок для кратных и дольных единиц. На осях указываются только принятые в тексте обозначения изображенных величин. Если обозначение отсутствует, вдоль осей пишут развернутое наименование величины (с прописной буквы), отделяя от единицы величины запятой. При наличии цифр обязательно указываются величины в соответствии с принятыми сокра-

щениями. Если на рисунке изображено семейство кривых, то буквенное обозначение параметра указывается на первой и последней кривых.

Размещать рисунки следует так, чтобы их можно было рассматривать без поворота работы. Если это невозможно, то рисунки располагают так, чтобы для их рассматривания надо было повернуть работу по часовой стрелке на 90 градусов.

Если страница не полностью занята таблицей или иллюстрацией, то на ней размещают, кроме того, соответствующее количество строк.

Приложения оформляются как продолжение отчета. В тексте работы должны быть ссылки на все приложения. Приложения располагаются в порядке ссылок на них. Каждое приложение следует начинать с новой страницы с указанием наверху посередине страницы слова «Приложение» с его обозначением. Приложения обозначают заглавными буквами русского алфавита, за исключением букв Ё, З, Й, О, Ч, Ъ, Ы, Ь. После слова «Приложение» следует буква, обозначающая его последовательность. Допускается обозначение приложений буквами латинского алфавита, за исключением букв I и O. Приложение должно иметь заголовок, который записывают посередине страницы с прописной буквы отдельной строкой.

Рисунки каждого приложения и таблицы обозначаются отдельной нумерацией арабскими цифрами с добавлением перед цифрой обозначения приложения. Например, Рисунок А.5 или Таблица Б.2.

Сокращения в тексте отчета возможны лишь в тех случаях, когда установлены соответствующим стандартом или правилом русской орфографии и пунктуации, например, и так далее – и т.д.; год (года) – г. (гг.); тысячи – тыс., миллионы – млн, миллиарды – млрд. Могут применяться узкоспециализированные сокращения. При этом необходимо один раз детально расшифровать их в скобках после первого упоминания, а в последующем эту расшифровку не повторять, например, ключевые показатели эффективности (КПЭ).

Сокращение слов в заголовках разделов, подразделов, пунктов, таблиц, приложений и в подписях под рисунками не допускается. Не допускается замена слов в тексте математическими знаками без цифр, например: \leq (меньше или равно), \geq (больше или равно), знак № (номер) и % (процент). Нельзя сокращать обозначения единиц физических и стоимостных величин, если они употребляются без цифр, за исключением единиц этих величин в таблицах и в расшифровках буквенных обозначений, входящих в формулы.

Единицы измерения необходимо указывать в соответствии с государственными стандартами и другими общепринятыми правилами. Например, принято обозначать сокращенно единицы измерения времени (секунда – с, минута – мин, час – ч); массы (грамм – г, килограмм – кг, центнер – ц, тонна – т); площади (квадратный метр – м² (кв. м), гектар – га); длины (миллиметр – мм, сантиметр – см, метр – м, километр – км); объема (кубический метр – м³ (куб. м)); скорости (метр в секунду – м/с, километр в час – км/ч) и т.д. После таких сокращений точку не ставят. Денежную единицу измерения обозначают с точкой: руб.

Оформление исходных текстов программ.

При необходимости проиллюстрировать в тексте документа алгоритм или важные особенности его реализации на конкретном языке программирования, фрагменты исходного текста программы, реализующей данный алгоритм, допускается включать в текст по ходу изложения. При этом следует стремиться к уменьшению объема включаемого фрагмента, заменяя несущественные строки комментариями на русском языке.

Включаемый в документ текст программы должен сохранять: синтаксическую корректность с точки зрения того языка программирования, на котором он написан (не допускается, к примеру, разбивать строчные комментарии на несколько строк и т. п.).

При оформлении текста программы (или его фрагментов) следует использовать моноширинный шрифт прямого начертания (например, Courier New). Допускается применять размер шрифта в листингах меньший, чем в тексте.

Фрагмент программы, включаемый в текст документа, должен быть выделен пустыми строками сверху и снизу. Пример оформления фрагмента исходного текста запроса на языке SQL приведен ниже.

Пример.

Получение списка из таблицы zaiavki, содержащего перечень подразделений с указанием оборудования, требующего ремонта, и запчастей для данного оборудования реализуется следующим запросом.

```
SELECT podrazdelenia.naimenovanie, oborudovanie.naimenovanie,  
zapchasti.naimenovanie  
FROM zaiavki  
JOIN podrazdelenia ON id_podrazd_zaiav=id_podrazd  
JOIN oborudovanie ON id_oborud_zaiav=id_oborud  
JOIN zapchasti ON id_zapchasti_zaiav=id_zapchasti;
```

7 Рекомендуемые источники

1. Осипов Д. Л. Технологии проектирования баз данных [электронный ресурс]: / Д. Л. Осипов. – Москва: ДМК Пресс, 2019. – 498 с. Режим доступа: <https://e.lanbook.com/book/131692>.

2. Шёниг Г. PostgreSQL 11. Мастерство разработки [электронный ресурс]: / Г. Шёниг; перевод с английского А. А. Слинкина. – Москва: ДМК Пресс, 2020. – 352 с. Режим доступа: <https://e.lanbook.com/book/131714>.

3. Джуба С. Изучаем PostgreSQL 10 [электронный ресурс]: / С. Джуба, А. Волков. – Москва: ДМК Пресс, 2018. – 400 с. Режим доступа: <https://e.lanbook.com/book/116125>.

4. Стасышин В.М. Практикум по языку SQL [электронный ресурс]: учебное пособие / В.М. Стасышин, Т.Л. Стасышина; Новосибирский государственный технический университет. – Новосибирск: Новосибирский государственный технический университет, 2016. – 60 с. Режим доступа: <https://biblioclub.ru/index.php?page=book&id=576764>.

5. Гуцин А.Н. Базы данных [электронный ресурс]: учебно-методическое пособие / А.Н. Гуцин. - Электронные текстовые данные. - М.: Директ-Медиа, 2015. - 311 с. Режим доступа: <http://biblioclub.ru/index.php?page=book&id=278093>

6. Баженова, И.Ю. SQL и процедурно-ориентированные языки [электронный ресурс]: / И.Ю. Баженова. - 2-е изд., испр. - Москва : Национальный Открытый Университет «ИНТУИТ», 2016. Режим доступа: <http://biblioclub.ru/index.php?page=book&id=428934> .

7. Нестеров С. А. Основы информационной безопасности [электронный ресурс]: учебное пособие / Нестеров С. А.- Электронные текстовые данные. - СПб: Издательство Политехнического университета, 2014. - 322 с. Режим доступа: http://biblioclub.ru/index.php?page=book_red&id=363040

8. Стасышин В.М. Проектирование информационных систем и баз данных [электронный ресурс]: учебное пособие / В.М. Стасышин. - Электронные текстовые данные. - Новосибирск: НГТУ, 2012. - 100 с. Режим доступа: <http://biblioclub.ru/index.php?page=book&id=228774>

9. Щелоков С.А. Базы данных [электронный ресурс]: учебное пособие/ С.А. Щелоков - Электронные текстовые данные. – Оренбург: Оренбургский Государственный университет, 2014.- 298 с. Режим доступа: <http://biblioclub.ru/index.php?page=book&id=260752>

10 Полякова Л. Основы SQL [электронный ресурс]: учебный курс // WWW.INTUIT.RU: официальный сайт Национального Открытого Университета «ИНТУИТ». Режим доступа: <https://www.intuit.ru/studies/courses/5/5/info>

11. Документация к PostgreSQL [электронный ресурс]: <https://postgrespro.ru/docs>