Приложение 1.2.2 к ОПОП по специальности
09.02.13 Интеграция решений с применением
технологий искусственного интеллекта

Министерство науки и высшего образования Российской Федерации
Федеральное государственное бюджетное образовательное учреждение
высшего образования
«Магнитогорский государственный технический университет им. Г.И. Носова»

Многопрофильный колледж

МЕТОДИЧЕСКИЕ УКАЗАНИЯ ДЛЯ ЛАБОРАТОРНЫХ ЗАНЯТИЙ по ПМ. 02 АДМИНИСТРИРОВАНИЕ БАЗ ДАННЫХ

МДК.02.02 Управление и автоматизация баз данных

для обучающихся специальности
09.02.13 Интеграция решений с применением технологий искусственного интеллекта

СОДЕРЖАНИЕ

1 ВВЕДЕНИЕ

Важную часть теоретической и профессиональной практической подготовки обучающихся составляют лабораторные занятия.

Состав и содержание лабораторных занятий направлены на реализацию Федерального государственного образовательного стандарта среднего профессионального образования.

Ведущей дидактической целью лабораторных занятий является экспериментальное подтверждение и проверка существенных теоретических положений (законов, зависимостей).

В соответствии с рабочей программой профессионального модуля ПМ.02 «Администрирование баз данных» МДК 02.02 «Управление и автоматизация баз данных» предусмотрено проведение лабораторных занятий.

В результате их выполнения, обучающийся должен:

уметь:

У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;

У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;

У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

У 2.2.1 Осуществлять основные функции по администрированию баз данных;

У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных

У 2.3.1 Дать независимую оценку уровня безопасности

У 2.3.2 Производить регламентное обновление программного обеспечения;

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;

У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Содержание практических и лабораторных занятий ориентировано на освоение вида деятельности программы подготовки специалистов среднего звена по специальности и овладению ***профессиональными компетенциями:***

ПК 2.1 Выявлять проблемы, возникающие в процессе эксплуатации баз данных ПК 2.2 Осуществлять процедуры администрирования баз данных.

ПК 2.3 Проводить аудит систем безопасности баз данных с использованием регламентов по защите информации.

ПК 2.4 Формировать требования хранилищ банка данных для обучения.

А также формированию общих компетенций:

ОК 01 Выбирать способы решения задач профессиональной деятельности применительно к различным контекстам.

ОК 02 Использовать современные средства поиска, анализа и интерпретации информации и информационные технологии для выполнения задач профессиональной деятельности.

ОК 09. Пользоваться профессиональной документацией на государственном и иностранном языках.

Лабораторные занятия проводятся в рамках соответствующей темы, после освоения дидактических единиц, которые обеспечивают наличие знаний, необходимых для ее выполнения.

2 МЕТОДИЧЕСКИЕ УКАЗАНИЯ

Тема 2.1 Установка и настройка программного обеспечения для администрирования баз данных

Лабораторное занятие №30
Установка СУБД MySQL и настройка службы на локальном сервере

Цель: освоить процесс установки и базовой настройки СУБД MySQL на локальном сервере.

Выполнив работу, вы будете уметь:
У 2.3.1 Дать независимую оценку уровня безопасности
У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
MySQL

Задание:
1. Установить MySQL на локальную машину (Windows/Linux).
2. Настроить автозапуск службы MySQL.
3. Задать пароль для root и создать пользователя с ограниченными правами.
4. Создать тестовую базу данных и таблицу.
5. Выполнить простые SQL-запросы (SELECT, INSERT, UPDATE, DELETE).

Порядок выполнения работы:
Для Windows
1. Скачать MySQL Installer с официального сайта.
2. Запустить установку, выбрать тип Developer Default или Server only.
3. В процессе настройки:
– Выбрать тип конфигурации Development Computer.
– Задать пароль для root.
– Настроить запуск службы автоматически.
4. Проверить службу: services.msc → MySQL → тип запуска Автоматически.
5. Подключиться через MySQL Command Line Client или Workbench.

Для Linux
```
sudo apt update
sudo apt install mysql-server
sudo systemctl enable mysql
sudo systemctl start mysql
sudo mysql_secure_installation
sudo mysql -u root -p
```
Общие шаги после установки
1. Создать пользователя:
```
CREATE USER 'user1'@'localhost' IDENTIFIED BY 'password';
GRANT SELECT, INSERT ON *.* TO 'user1'@'localhost';
FLUSH PRIVILEGES;
```
2. Создать базу данных и таблицу:
```
CREATE DATABASE lab_db;
USE lab_db;
CREATE TABLE users (id INT AUTO_INCREMENT PRIMARY KEY, name VARCHAR(50));
```
3. Выполнить запросы:
```
INSERT INTO users (name) VALUES ('Иванов');
```

```
SELECT * FROM users;
UPDATE users SET name = 'Петров' WHERE id = 1;
DELETE FROM users WHERE id = 1;
```

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.1 Установка и настройка программного обеспечения для администрирования баз данных

Лабораторное занятие №31
Установка PostgreSQL и настройка параметров конфигурации (порт, логирование)

Цель: освоить установку СУБД PostgreSQL, настройку основных параметров конфигурации (порт, логирование) и проверку работоспособности службы.

Выполнив работу, вы будете уметь:
У 2.3.1 Дать независимую оценку уровня безопасности
У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
PostgreSQL

Задание:
1. Установить PostgreSQL на локальную машину (Windows/Linux).
2. Настроить порт подключения (отличный от стандартного 5432).
3. Настроить параметры логирования (включить логи, задать формат и уровень детализации).
4. Проверить работу службы и подключение к БД с новыми параметрами.
5. Проанализировать лог-файлы.

Порядок выполнения работы:
Для Windows
1. Скачать установщик PostgreSQL с официального сайта.
2. Запустить установку, указать:
   - Пароль суперпользователя postgres
   - Порт по умолчанию: 5432
3. После установки открыть postgresql.conf.
4. Изменить порт
5. Настроить логирование:

```
logging_collector = on
log_directory = 'log'
log_filename = 'postgresql-%Y-%m-%d_%H%M%S.log'
log_statement = 'all'
log_line_prefix = '%t [%p]: [%l-1] user=%u,db=%d '
```

6. Перезапустить службу PostgreSQL через services.msc.
7. Подключиться с новым портом:

```
psql -h localhost -p 5433 -U postgres
```

Для Linux (Ubuntu/Debian)

```
sudo apt update
sudo apt install postgresql postgresql-contrib
sudo systemctl enable postgresql
sudo systemctl start postgresql
```

Редактирование конфигурации:

```
sudo nano /etc/postgresql/*/main/postgresql.conf
```

Изменить параметры (аналогично Windows):

```
port = 5433
logging_collector = on
log_directory = 'pg_log'
log_filename = 'postgresql-%Y-%m-%d.log'
log_statement = 'ddl'   # none, ddl, mod, all
```

Перезапуск:

```
sudo systemctl restart postgresql
```

Подключение:

```
sudo -u postgres psql -p 5433
```

Проверка логирования

Выполнить несколько SQL-запросов:

```
CREATE TABLE test (id int);
INSERT INTO test VALUES (1);
SELECT * FROM test;
DROP TABLE test;
```

Найти и открыть лог-файл (в log_directory).

Убедиться, что запросы залогированы.

Форма представления результата:

Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.1 Установка и настройка программного обеспечения для администрирования баз данных

Лабораторное занятие №32
Установка MongoDB и настройка репликации для отказоустойчивости

Цель: Освоить установку MongoDB и настройку репликации для обеспечения отказоустойчивости базы данных.

Выполнив работу, вы будете уметь:

У 2.3.1 Дать независимую оценку уровня безопасности

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;

У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:

MongoDB

Задание:

1. Установить MongoDB на несколько серверов (или виртуальных машин).
2. Настроить репликацию, создав replica set.
3. Проверить работу репликации и отказоустойчивость системы.

Порядок выполнения работы:

1. Установить MongoDB на каждом узле.
2. Настроить конфигурационные файлы для запуска в режиме репликации.
3. Инициализировать replica set, добавив в него все узлы.
4. Проверить статус репликации с помощью специальных команд.
5. Провести тест: остановить один из узлов и убедиться, что база остаётся доступной.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.1 Установка и настройка программного обеспечения для администрирования баз данных

Лабораторное занятие №33
Установка Microsoft SQL Server и настройка параметров аутентификации

Цель: освоить установку Microsoft SQL Server, настройку режима аутентификации (смешанного режима), управление учетной записью sa и проверку подключений к серверу.

Выполнив работу, вы будете уметь:
У 2.3.1 Дать независимую оценку уровня безопасности
У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
Microsoft SQL Server

Задание:

1. Установить Microsoft SQL Server (Developer или Express Edition).
2. Настроить смешанный режим аутентификации (Mixed Mode) во время или после установки.
3. Задать надежный пароль для учетной записи sa.
4. Настроить разрешения для входа через SQL Server Authentication и Windows Authentication.
5. Проверить подключение к серверу с использованием разных способов аутентификации.

Порядок выполнения работы:

1. Установка Microsoft SQL Server

Скачивание и запуск установщика

- Скачать установщик SQL Server с официального сайта Microsoft (Developer Edition — бесплатна для разработки).
- Запустить установку, выбрать Basic или Custom тип установки.

Выбор режима аутентификации при установке

На этапе Database Engine Configuration:

- Выбрать Mixed Mode (Windows Authentication and SQL Server Authentication)
- Ввести и подтвердить надежный пароль для учетной записи sa
- Нажать Add Current User для добавления текущего пользователя Windows в качестве администратора SQL Server
- Завершить установку

Требования к надежному паролю sa:

- Длина не менее 8 символов
- Содержать прописные и строчные буквы
- Содержать цифры
- Содержать небуквенно-цифровые символы (#, %, ^ и др.)

2. Настройка режима аутентификации (если не был выбран при установке)

Если был выбран только режим Windows Authentication, его можно изменить после установки:

- Запустить SQL Server Management Studio (SSMS)
- Подключиться к экземпляру SQL Server
- В Object Explorer щелкнуть правой кнопкой мыши по имени сервера → Properties
- Перейти на страницу Security
- В разделе Server authentication выбрать:
- SQL Server and Windows Authentication mode (Mixed Mode)
- Нажать OK
- Перезапустить службу SQL Server

3. Управление учетной записью sa

Включение учетной записи sa (если отключена)

```
ALTER LOGIN sa ENABLE;
GO
```

Изменение пароля sa

```
ALTER LOGIN sa WITH PASSWORD = 'НовыйНадежныйПароль!23';
GO
```

Проверка состояния учетной записи sa

```
SELECT name, is_disabled FROM sys.sql_logins WHERE name = 'sa';
GO
```

4. Создание дополнительных учетных записей

Создание SQL Server логина

```
CREATE LOGIN test_user WITH PASSWORD = 'UserPass123!';
GO
```

Создание пользователя в базе данных

```
USE master;
GO
CREATE USER test_user FOR LOGIN test_user;
GO
```

Назначение роли (например, sysadmin)

```
ALTER SERVER ROLE sysadmin ADD MEMBER test_user;
GO
```

5. Проверка подключений

Подключение через Windows Authentication

- В SSMS выбрать Windows Authentication
- Нажать Connect (пароль не требуется — используются учетные данные Windows)

Подключение через SQL Server Authentication

- В SSMS выбрать SQL Server Authentication
- Ввести Login: sa (или test_user)
- Ввести пароль
- Нажать Connect

Проверка через командную строку (sqlcmd)

```
sqlcmd -S localhost -U sa -P "Пароль" -Q "SELECT @@VERSION"
```

6. Дополнительные настройки безопасности

Принуждение политики паролей для SQL логина

```
CREATE LOGIN app_user WITH PASSWORD = 'AppPass123!'
MUST_CHANGE, CHECK_EXPIRATION = ON, CHECK_POLICY = ON;
GO
```

Форма представления результата:

Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.2. Установка и настройка программного обеспечения (ПО) для обеспечения работы пользователей с базами данных

Лабораторное занятие №34
Установка и настройка клиента SQL Workbench для работы с базой данных MySQL

Цель: освоить навыки установки и настройки клиента SQL Workbench для подключения к базе данных MySQL, а также выполнить базовые операции по работе с базой данных через графический интерфейс.

Выполнив работу, вы будете уметь:

У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;

У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;

У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

Материальное обеспечение:

MySQL, Workbench

Задание:

1. Установить SQL Workbench/J на компьютер.
2. Настроить подключение к серверу MySQL.
3. Проверить соединение с базой данных.
4. Выполнить простые SQL-запросы с помощью SQL Workbench.

Порядок выполнения работы:

1. Установка SQL Workbench/J
   - Скачайте последнюю версию SQL Workbench/J с официального сайта.
   - Распакуйте архив в удобную папку.
   - Запустите приложение, выполнив файл SQLWorkbench.jar (может потребоваться Java Runtime Environment).
2. Настройка подключения к MySQL
   - В меню выберите «Workbench» → «Manage Drivers».
   - Добавьте новый драйвер:
   - укажите имя;
   - в поле «Library List» добавьте файл драйвера MySQL (mysql-connector-java-*.jar);
   - в поле «URL Class» укажите: com.mysql.cj.jdbc.Driver;
   - в поле «Template» задайте шаблон URL: jdbc:mysql://<host>:<port>/<database>?useSSL=false&serverTimezone=UTC.
   - Сохраните настройки драйвера.
3. Создание профиля подключения
   - В меню выберите «File» → «Connect Window».
   - Нажмите «Create a new profile».
   - Укажите: имя профиля, выберите драйвер MySQL, введите адрес сервера, порт, имя базы данных, имя пользователя и пароль.
   - Проверьте соединение с помощью кнопки «Check».
4. Выполнение SQL-запросов
   - После успешного подключения откройте редактор запросов.
   - Введите и выполните простые SQL-запросы, например:

```
SHOW DATABASES;
USE <имя_базы_данных>;
SHOW TABLES;
SELECT * FROM <имя_таблицы> LIMIT 10;
```

   - Проанализируйте результаты выполнения запросов.

Форма представления результата:

Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.2. Установка и настройка программного обеспечения (ПО) для обеспечения работы пользователей с базами данных

Лабораторное занятие №35
Настройка пользователей и прав доступа через pgAdmin для PostgreSQL

Цель: освоить навыки создания пользователей и настройки прав доступа к объектам базы данных PostgreSQL с помощью графического интерфейса pgAdmin.

Выполнив работу, вы будете уметь:

У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;

У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;

У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

Материальное обеспечение:
PostgreSQL, pgAdmin

Задание:

1. Установить и запустить pgAdmin.
2. Подключиться к серверу PostgreSQL.
3. Создать нового пользователя (роль) с заданными параметрами.
4. Настроить права доступа для созданного пользователя к базе данных и отдельным таблицам.
5. Проверить работу прав доступа, выполнив тестовые SQL-запросы от имени нового пользователя.

Порядок выполнения работы:

1. Запуск pgAdmin и подключение к серверу
   - Откройте pgAdmin.
   - В разделе «Servers» создайте новое подключение к серверу PostgreSQL, указав имя сервера, порт, имя пользователя (например, postgres) и пароль.
   - Убедитесь, что соединение с сервером установлено успешно.
2. Создание нового пользователя (роли)
   - В дереве объектов выберите «Login/Group Roles».
   - Нажмите правой кнопкой мыши и выберите «Create» → «Login/Group Role».
   - В открывшемся окне заполните поля:
     - Name — имя пользователя;
     - Password — пароль;
     - Account Expiry — срок действия учётной записи (при необходимости);
     - Privileges — установите необходимые флаги (например, Can login?, Create databases?, Create roles?).
   - Сохраните изменения.
3. Настройка прав доступа к базе данных
   - В дереве объектов найдите нужную базу данных, нажмите правой кнопкой мыши и выберите «Properties».
   - Перейдите на вкладку «Privileges».
   - Добавьте нового пользователя и назначьте ему права:
     - CONNECT — право на подключение к базе;
     - CREATE — право создавать объекты в схеме public (или другой).
   - Примените изменения.
4. Настройка прав доступа к таблицам
   - Раскройте нужную базу данных, затем раздел «Schemas» → «Tables».
   - Выберите таблицу, нажмите правой кнопкой мыши и выберите «Properties» → вкладка «Privileges».
   - Добавьте пользователя и назначьте права:
     - SELECT, INSERT, UPDATE, DELETE — для работы с данными;
     - REFERENCES, TRIGGER — для дополнительных операций.

- Сохраните изменения.

5. Проверка прав доступа
   - Подключитесь к серверу от имени нового пользователя (создайте новое подключение в pgAdmin с его учётными данными).
   - Выполните тестовые SQL-запросы:
     - SELECT * FROM <имя_таблицы> LIMIT 5;
     - INSERT INTO <имя_таблицы> ...; (если есть право на вставку)
   - Зафиксируйте результаты: успешное выполнение или ошибки доступа.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.2. Установка и настройка программного обеспечения (ПО) для обеспечения работы пользователей с базами данных

Лабораторное занятие №36
Установка и настройка DBeaver для подключения к различным типам баз данных

Цель: освоить навыки установки и настройки универсального клиента DBeaver для подключения к различным типам систем управления базами данных (СУБД), а также выполнить базовые операции по работе с базами данных через графический интерфейс.

Выполнив работу, вы будете уметь:
У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;
У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;
У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

Материальное обеспечение:
MySQL, PostgreSQL, SQLite, DBeaver

Задание:
1. Установить DBeaver на компьютер.
2. Настроить драйверы для подключения к различным СУБД (MySQL, PostgreSQL, SQLite).
3. Создать и протестировать подключения к базам данных разных типов.
4. Выполнить простые SQL-запросы в каждой из подключённых баз данных.
5. Оформить результаты работы в виде отчёта.

Порядок выполнения работы:
1. Установка DBeaver
   - Скачайте последнюю версию DBeaver с официального сайта.

- Запустите установочный файл и следуйте инструкциям мастера установки.
- После завершения установки запустите DBeaver.

2. Настройка драйверов СУБД
   - Откройте «База данных» → «Менеджер драйверов».
   - В открывшемся окне ознакомьтесь со списком предустановленных драйверов.
   - При необходимости добавьте новый драйвер (например, для MySQL, PostgreSQL или другой СУБД):
     - Нажмите «Создать» → «Новый драйвер».
     - Укажите имя драйвера.
     - Загрузите или укажите путь к JAR-файлу драйвера (например, mysql-connector-java-*.jar для MySQL).
     - Проверьте, что класс драйвера указан верно (например, com.mysql.cj.jdbc.Driver для MySQL).
   - Сохраните изменения.
3. Создание подключения к базе данных
   - В главном окне DBeaver нажмите «Новое соединение» (значок плюса).
   - Выберите тип СУБД (например, MySQL, PostgreSQL, SQLite).
   - Заполните параметры подключения:
     - Хост (адрес сервера);
     - Порт;
     - Имя базы данных;
     - Имя пользователя;
     - Пароль.
   - Нажмите «Проверить соединение». Если всё настроено верно, появится сообщение об успешном подключении.
   - Сохраните подключение.
4. Выполнение SQL-запросов
   - Разверните созданное подключение в навигаторе баз данных.
   - Откройте редактор SQL, дважды щёлкнув по базе данных или выбрав «Создать» → «Скрипт SQL».
   - Выполните простые SQL-запросы для каждой СУБД, например:
     - Для MySQL/PostgreSQL:
       ```
       SHOW DATABASES;
       USE <имя_базы_данных>;
       SHOW TABLES;
       SELECT * FROM <имя_таблицы> LIMIT 5;
       ```
     - Для SQLite:
       ```
       .databases
       .tables
       SELECT * FROM <имя_таблицы> LIMIT 5;
       ```
   - Проанализируйте результаты выполнения запросов.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.2. Установка и настройка программного обеспечения (ПО) для обеспечения работы пользователей с базами данных

Лабораторное занятие №37
Установка и настройка Microsoft Management Studio (SSMS) для работы с SQL Server

Цель: освоить навыки установки и настройки Microsoft SQL Server Management Studio (SSMS) для подключения к серверу баз данных Microsoft SQL Server, а также выполнить базовые операции по работе с базами данных через графический интерфейс.

Выполнив работу, вы будете уметь:

У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;

У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;

У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

Материальное обеспечение:

SQL Server, Microsoft Management Studio

Задание:

1. Установить SSMS на компьютер.
2. Настроить подключение к экземпляру SQL Server.
3. Выполнить базовые SQL-запросы для работы с объектами базы данных.
4. Изучить основные элементы интерфейса SSMS.
5. Оформить результаты работы в виде отчёта.

Порядок выполнения работы:

1. Установка Microsoft SQL Server Management Studio (SSMS)
   - Скачайте установочный файл SSMS с официального сайта Microsoft.
   - Запустите скачанный установщик и следуйте инструкциям мастера установки.
   - Дождитесь завершения установки и запустите SSMS.
2. Подключение к серверу SQL Server
   - При первом запуске SSMS появится окно «Соединение с сервером».
   - В поле «Имя сервера» укажите имя или IP-адрес экземпляра SQL Server (например, localhost или .\SQLEXPRESS для локального сервера).
   - В поле «Проверка подлинности» выберите «Проверка подлинности Windows» (если используется доменная учётная запись) или «Проверка подлинности SQL Server» (если заданы логин и пароль SQL).
   - Нажмите кнопку «Соединить».
   - Если подключение выполнено успешно, в обозревателе объектов (Object Explorer) появится дерево объектов сервера.
3. Изучение интерфейса SSMS
   - Ознакомьтесь с основными элементами окна SSMS:
     - обозреватель объектов (Object Explorer);
     - редактор запросов (Query Editor);
     - панели результатов (Results), сообщений (Messages).
   - Разверните узлы «Базы данных», «Безопасность», «Серверные объекты», чтобы изучить структуру сервера.
4. Выполнение базовых SQL-запросов

- В обозревателе объектов выберите нужную базу данных, нажмите правой кнопкой мыши и выберите «Новая вкладка запроса».
- В открывшемся редакторе введите и выполните следующие SQL-запросы:
  - просмотр списка баз данных:
    ```
    SELECT name FROM sys.databases;
    ```
  - просмотр таблиц в текущей базе данных:
    ```
    SELECT * FROM sys.tables;
    ```
  - просмотр первых строк таблицы (например, Employees):
    ```
    SELECT TOP 10 * FROM Employees;
    ```
- Проанализируйте результаты выполнения запросов в панелях «Результаты» и «Сообщения».

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.2. Установка и настройка программного обеспечения (ПО) для обеспечения работы пользователей с базами данных

Лабораторное занятие №38
Установка и настройка библиотек Python для взаимодействия с базами данных (pymysql, psycopg2)

Цель: освоить навыки установки и настройки библиотек Python для взаимодействия с базами данных MySQL и PostgreSQL (pymysql, psycopg2), а также научиться выполнять базовые SQL-запросы из Python-скриптов.

Выполнив работу, вы будете уметь:
У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;
У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;
У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

Материальное обеспечение:
MySQL, PostgreSQL, Python

Задание:
1. Установить интерпретатор Python (если не установлен).
2. Установить библиотеки pymysql и psycopg2 для работы с MySQL и PostgreSQL.
3. Настроить подключение к базам данных из Python.
4. Написать и выполнить скрипты для выполнения базовых SQL-запросов.
5. Оформить результаты работы в виде отчёта.

Порядок выполнения работы:

1. Подготовка среды
   - Убедитесь, что на компьютере установлен Python (рекомендуется версия 3.8 и выше).
   - Откройте терминал или командную строку.
   - Установите необходимые библиотеки с помощью pip:
   ```
   pip install pymysql psycopg2-binary
   ```
   - Проверьте успешность установки, выполнив:
   ```
   python -c "import pymysql, psycopg2; print('pymysql:', pymysql.__version__, 'psycopg2:', psycopg2.__version__)"
   ```
2. Подключение к MySQL с помощью pymysql
   - Создайте Python-скрипт для подключения к MySQL:
   ```
   import pymysql

   # Параметры подключения
   connection = pymysql.connect(
       host='localhost',
       user='your_user',
       password='your_password',
       database='your_database',
       charset='utf8mb4'
   )

   try:
       with connection.cursor() as cursor:
           # Выполнение запроса
           sql = "SELECT VERSION();"
           cursor.execute(sql)
           result = cursor.fetchone()
           print("Версия MySQL:", result)
   finally:
       connection.close()
   ```
   - Запустите скрипт и убедитесь, что соединение установлено успешно.
3. Подключение к PostgreSQL с помощью psycopg2
   - Создайте Python-скрипт для подключения к PostgreSQL:
   ```
   import psycopg2

   # Параметры подключения
   connection = psycopg2.connect(
       host='localhost',
       user='your_user',
       password='your_password',
       dbname='your_database'
   )

   try:
       with connection.cursor() as cursor:
           # Выполнение запроса
           sql = "SELECT version();"
           cursor.execute(sql)
           result = cursor.fetchone()
           print("Версия PostgreSQL:", result)
   finally:
       connection.close()
   ```
   - Запустите скрипт и проверьте результат.
4. Выполнение базовых SQL-запросов
   - Для каждой базы данных напишите скрипты, выполняющие:
     - просмотр списка таблиц;
     - выборку первых строк из выбранной таблицы;
     - вставку новой записи (если есть соответствующие права).
   - Запустите скрипты и проанализируйте результаты.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.2. Установка и настройка программного обеспечения (ПО) для обеспечения работы пользователей с базами данных

Лабораторное занятие № 39
Настройка интеграции баз данных с клиентским ПО через ODBC-драйверы

Цель: освоить навыки настройки интеграции между клиентским программным обеспечением и базами данных с помощью ODBC-драйверов, научиться создавать источники данных (DSN) и выполнять запросы к различным СУБД через универсальный интерфейс.

Выполнив работу, вы будете уметь:
У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;
У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;
У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

Материальное обеспечение:
MySQL, PostgreSQL, SQL Server, MyODBC, psqlodbc, ODBC Driver for SQL Server

Задание:
1. Изучить назначение и принцип работы ODBC.
2. Установить необходимые ODBC-драйверы для выбранных СУБД (MySQL, PostgreSQL, SQL Server).
3. Настроить системные или пользовательские источники данных (DSN) для подключения к базам данных.
4. Проверить подключение с помощью тестовых утилит или клиентского ПО.
5. Выполнить простые SQL-запросы через ODBC-подключение.
6. Оформить результаты работы в виде отчёта.

Порядок выполнения работы:
1. Изучение ODBC и подготовка среды
   - Ознакомьтесь с теоретическими основами ODBC: назначение, архитектура, преимущества использования универсального интерфейса для доступа к различным СУБД.
   - Убедитесь, что на компьютере установлен клиентский инструмент для работы с ODBC (например, «Администратор источников данных ODBC» в Windows или аналогичные утилиты в Linux).
2. Установка ODBC-драйверов

- Скачайте и установите ODBC-драйверы для выбранных СУБД:
  - для MySQL — MySQL ODBC Driver (MyODBC);
  - для PostgreSQL — psqlodbc;
  - для SQL Server — ODBC Driver for SQL Server (от Microsoft).
- Следуйте инструкциям установщика, при необходимости перезагрузите компьютер.

3. Создание источника данных (DSN)
   - Откройте «Администратор источников данных ODBC» (в Windows: «Панель управления» → «Администрирование» → «Источники данных (ODBC)»).
   - Перейдите на вкладку «Пользовательский DSN» или «Системный DSN».
   - Нажмите «Добавить» и выберите установленный драйвер.
   - Введите имя источника данных (DSN Name), описание, а также параметры подключения:
     - адрес сервера (Host);
     - порт;
     - имя базы данных;
     - имя пользователя и пароль (при необходимости).
   - Сохраните настройки и протестируйте соединение с помощью кнопки «Проверить источник данных» или «Проверить».
4. Проверка подключения из клиентского ПО
   - Откройте клиентское приложение, поддерживающее ODBC (например, Microsoft Excel, Tableau, LibreOffice Base, Python с модулем pyodbc).
   - Создайте новое подключение, выбрав соответствующий DSN.
   - Проверьте, что приложение успешно подключается к базе данных.
5. Выполнение SQL-запросов через ODBC
   - С помощью клиентского ПО или тестовой утилиты выполните простые SQL-запросы:

```
SELECT * FROM <имя_таблицы> LIMIT 10;
SHOW TABLES;
SELECT COUNT(*) FROM <имя_таблицы>;
```

   - Проанализируйте результаты выполнения запросов.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.3. Управление доступом к базам данных

Лабораторное занятие № 40
Создание пользователей и групп в MySQL и назначение прав доступа (GRANT, REVOKE)

Цель: освоить навыки создания пользователей и групп в MySQL, а также научиться управлять правами доступа к базам данных и их объектам с помощью команд GRANT и REVOKE.

Выполнив работу, вы будете уметь:
У 2.2.1 Осуществлять основные функции по администрированию баз данных;

У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
MySQL

Задание:

1. Изучить структуру управления пользователями и привилегиями в MySQL.
2. Создать новых пользователей с различными правами доступа.
3. Создать роль (группу) и назначить ей набор привилегий.
4. Назначить роль пользователям.
5. Отработать команды GRANT и REVOKE для назначения и отзыва прав.
6. Проверить работу прав доступа, выполнив тестовые SQL-запросы от имени разных пользователей.
7. Оформить результаты работы в виде отчёта.

Порядок выполнения работы:

1. Подготовка и теоретическая часть
   - Подключитесь к серверу MySQL с правами администратора (например, под пользователем root).
   - Ознакомьтесь с основными понятиями: пользователь, хост, привилегии, роли (группы).
   - Изучите синтаксис команд: CREATE USER, GRANT, REVOKE, CREATE ROLE (для MySQL 8.0 и выше), SHOW GRANTS.
2. Создание пользователей
   - Создайте нового пользователя для локального подключения:

```
CREATE USER 'user1'@'localhost' IDENTIFIED BY 'password1';
```

   - Создайте пользователя для удалённого подключения (если требуется):

```
CREATE USER 'user2'@'%' IDENTIFIED BY 'password2';
```

   - Проверьте список пользователей:

```
SELECT User, Host FROM mysql.user;
```

3. Назначение прав доступа (GRANT)
   - Назначьте пользователю право только на подключение и чтение данных из определённой базы данных:

```
GRANT SELECT, INSERT ON mydatabase.* TO 'user1'@'localhost';
```

   - Назначьте пользователю все права на конкретную таблицу:

```
GRANT ALL PRIVILEGES ON mydatabase.employees TO 'user2'@'%';
```

   - Примените изменения:

```
FLUSH PRIVILEGES;
```

   - Проверьте назначенные права:

```
SHOW GRANTS FOR 'user1'@'localhost';
```

4. Создание роли (группы) и назначение прав
   - Создайте роль:
   - sql
   - Копировать

```
CREATE ROLE 'role_developer';
```

   - Назначьте роли необходимые привилегии:

```
GRANT SELECT, INSERT, UPDATE, DELETE ON mydatabase.* TO 'role_developer';
```

   - Назначьте роль пользователю:

```
GRANT 'role_developer' TO 'user1'@'localhost';
```

   - Активируйте роль для текущей сессии пользователя (при необходимости):

```
SET DEFAULT ROLE ALL TO 'user1'@'localhost';
```

5. Отзыв прав доступа (REVOKE)

- Отзовите у пользователя право на вставку данных:

```
REVOKE INSERT ON mydatabase.* FROM 'user1'@'localhost';
```

- Отзовите у пользователя роль:

```
REVOKE 'role_developer' FROM 'user1'@'localhost';
```

- Проверьте изменения командой SHOW GRANTS.

6. Проверка работы прав доступа
   - Подключитесь к серверу MySQL под именем созданного пользователя.
   - Попробуйте выполнить различные SQL-запросы (например, SELECT, INSERT, UPDATE, DELETE) и зафиксируйте, какие из них выполняются успешно, а какие — с ошибкой доступа.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.3. Управление доступом к базам данных

Лабораторное занятие № 41
Настройка ролей и прав доступа в PostgreSQL для различных пользователей

Цель: освоить навыки создания ролей, настройки прав доступа и управления пользователями в PostgreSQL с помощью команд SQL и инструментов администрирования, а также научиться применять принцип наименьших привилегий для обеспечения безопасности базы данных.

Выполнив работу, вы будете уметь:
У 2.2.1 Осуществлять основные функции по администрированию баз данных;
У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
PostgreSQL, pgAdmin, DBeaver

Задание:
1. Изучить систему ролей и привилегий в PostgreSQL.
2. Создать роли с различными наборами прав (пользователи, группы).
3. Назначить роли пользователям и сгруппировать их.
4. Настроить права доступа к базе данных, схемам и таблицам.
5. Отработать команды GRANT и REVOKE для назначения и отзыва прав.
6. Проверить работу прав доступа, выполнив тестовые SQL-запросы от имени разных пользователей.
7. Оформить результаты работы в виде отчёта.

Порядок выполнения работы:
1. Подготовка и теоретическая часть

- Подключитесь к серверу PostgreSQL с правами суперпользователя (например, postgres).
- Ознакомьтесь с понятиями: роль, пользователь, группа, привилегия, владелец объекта.
- Изучите синтаксис основных команд: CREATE ROLE, CREATE USER (сокращение для CREATE ROLE ... LOGIN), GRANT, REVOKE, \du (просмотр ролей в psql), \dp (просмотр прав на объекты).

2. Создание ролей и пользователей
   - Создайте роль-группу для аналитиков:
   ```
   CREATE ROLE analysts NOLOGIN;
   ```
   - Создайте пользователей, которые смогут входить в систему:
   ```
   CREATE USER analyst1 WITH PASSWORD 'password1';
   CREATE USER analyst2 WITH PASSWORD 'password2';
   ```
   - Добавьте пользователей в группу:
   ```
   GRANT analysts TO analyst1, analyst2;
   ```
   - Проверьте список ролей:
   ```
   \du
   ```
3. Назначение прав доступа (GRANT)
   - Назначьте группе право на подключение к базе данных и использование схемы:
   ```
   GRANT CONNECT ON DATABASE mydb TO analysts;
   GRANT USAGE ON SCHEMA public TO analysts;
   ```
   - Назначьте права на работу с таблицами (например, только чтение):
   ```
   GRANT SELECT ON ALL TABLES IN SCHEMA public TO analysts;
   ```
   - Для новых таблиц в схеме настройте права по умолчанию:
   ```
   ALTER DEFAULT PRIVILEGES IN SCHEMA public GRANT SELECT ON TABLES TO analysts;
   ```
   - Проверьте назначенные права:
   ```
   \dp public.*
   ```
4. Создание пользователей с расширенными правами
   - Создайте роль для разработчиков:
   ```
   CREATE ROLE developers NOLOGIN;
   CREATE USER dev_user WITH PASSWORD 'devpass';
   GRANT developers TO dev_user;
   ```
   - Назначьте группе разработчиков полный доступ к схеме:
   ```
   GRANT ALL PRIVILEGES ON ALL TABLES IN SCHEMA public TO developers;
   GRANT ALL PRIVILEGES ON SCHEMA public TO developers;
   ```
   - Проверьте права для этой группы.
5. Отзыв прав доступа (REVOKE)
   - Отзовите у группы аналитиков право на чтение из определённой таблицы:
   ```
   REVOKE SELECT ON sensitive_table FROM analysts;
   ```
   - Проверьте, что пользователь из этой группы больше не может выполнить запрос к этой таблице.
6. Проверка работы прав доступа
   - Подключитесь к базе данных под разными пользователями (analyst1, dev_user).
   - Выполните тестовые SQL-запросы:
   ```
   SELECT * FROM <имя_таблицы>;
   INSERT INTO <имя_таблицы> ...;
   ```
   - Зафиксируйте, какие операции доступны каждому пользователю, а какие вызывают ошибку доступа.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.3. Управление доступом к базам данных

Лабораторное занятие № 42
Управление правами доступа в Microsoft SQL Server с использованием SQL Server Management Studio (SSMS)

Цель: освоить навыки управления пользователями, ролями и правами доступа в Microsoft SQL Server с помощью графического интерфейса SQL Server Management Studio (SSMS), а также научиться применять принцип наименьших привилегий для обеспечения безопасности баз данных.

Выполнив работу, вы будете уметь:

У 2.2.1 Осуществлять основные функции по администрированию баз данных;

У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных

У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:

SQL Server, SQL Server Management Studio

Задание:

1. Изучить архитектуру безопасности SQL Server: серверные и базы данных роли, уровни прав доступа.
2. Создать новых пользователей и роли на уровне сервера и базы данных.
3. Назначить пользователям и ролям права доступа к объектам базы данных (таблицам, представлениям, схемам).
4. Отработать операции по назначению (GRANT) и отзыву (DENY, REVOKE) прав.
5. Проверить работу прав доступа, выполнив тестовые SQL-запросы от имени разных пользователей.
6. Оформить результаты работы в виде отчёта.

Порядок выполнения работы:

1. Подготовка и теоретическая часть
   - Подключитесь к экземпляру SQL Server через SSMS с правами администратора (например, под учётной записью sa или члена роли sysadmin).
   - Ознакомьтесь с основными понятиями:
     - Логины (Logins) — учётные записи на уровне сервера.
     - Пользователи (Users) — сопоставление логинов с конкретной базой данных.
     - Роли (Roles) — серверные и базы данных (для группировки пользователей).
     - Привилегии (Permissions) — права на выполнение операций.
   - Изучите основные разделы в обозревателе объектов (Object Explorer): «Безопасность» → «Имена для входа» (Logins), «Пользователи» (Users), «Роли» (Roles).
2. Создание логинов и пользователей
   - Создайте новый логин на уровне сервера:
     - В обозревателе объектов щёлкните правой кнопкой мыши по «Имена для входа» → «Создать имя для входа».
     - Укажите имя, выберите тип проверки подлинности (например, «Проверка подлинности SQL Server»), задайте пароль.
   - Создайте пользователя в базе данных для этого логина:

- Раскройте нужную базу данных → «Безопасность» → «Пользователи».
- Щёлкните правой кнопкой мыши → «Создать пользователя».

- В поле «Имя пользователя» укажите имя, в поле «Имя для входа» выберите созданный ранее логин.
- Повторите действия для нескольких пользователей.

3. Создание ролей и назначение прав
   - Создайте роль на уровне базы данных:
     - В разделе «Роли базы данных» → «Роли базы данных по определению» → щёлкните правой кнопкой мыши → «Создать роль базы данных».
     - Укажите имя роли и, при необходимости, владельца.
   - Добавьте пользователей в созданную роль:
     - Откройте свойства роли → вкладка «Члены» → «Добавить» → выберите пользователей.
   - Назначьте роли права доступа к объектам:
     - Откройте свойства роли → вкладка «Защищаемые объекты».
     - Нажмите «Поиск», выберите нужную схему или таблицу.
     - В разделе «Состояние разрешений» установите флажки для необходимых действий (например, SELECT, INSERT, UPDATE, DELETE, EXECUTE).
   - Примените изменения.
4. Назначение и отзыв прав отдельным пользователям
   - Назначьте отдельному пользователю право на выполнение определённого запроса:
     - В разделе «Пользователи» откройте свойства пользователя → вкладка «Защищаемые объекты».
     - Выберите объект и установите нужные права.
   - Отзовите право у пользователя или роли:
     - В том же окне снимите флажок с права или выберите «Запретить» (DENY) для проверки работы ограничений.
5. Проверка работы прав доступа
   - Выполните подключение к базе данных от имени созданного пользователя (в SSMS: «Файл» → «Новое соединение с запросом к базе данных...»).
   - Попробуйте выполнить различные SQL-запросы:
     - SELECT * FROM <имя_таблицы>;
     - INSERT INTO <имя_таблицы> ...;
     - UPDATE <имя_таблицы> ...;
   - Зафиксируйте, какие операции доступны каждому пользователю, а какие вызывают ошибку доступа (например: «Запрет доступа», «В разрешении отказано»).

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.3. Управление доступом к базам данных

Лабораторное занятие № 43
Настройка аутентификации и шифрования соединения в MySQL

Цель: освоить навыки настройки аутентификации пользователей и шифрования соединений в MySQL для обеспечения безопасного доступа к базам данных, а также научиться проверять и тестировать защищённые соединения.

Выполнив работу, вы будете уметь:
У 2.2.1 Осуществлять основные функции по администрированию баз данных;
У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
MySQL

Задание:
1. Изучить механизмы аутентификации и шифрования в MySQL.
2. Настроить использование SSL/TLS для шифрования клиент-серверного соединения.
3. Создать пользователей с различными методами аутентификации.
4. Проверить, что соединения с сервером MySQL возможны только с использованием шифрования.
5. Оформить результаты работы в виде отчёта.

Порядок выполнения работы:
1. Подготовка и теоретическая часть
   - Подключитесь к серверу MySQL с правами администратора (например, под пользователем root).
   - Ознакомьтесь с основными понятиями:
     - методы аутентификации (mysql_native_password, caching_sha2_password);
     - роль SSL/TLS в защите данных;
     - файлы сертификатов и ключей: ca, server-cert, server-key.
   - Убедитесь, что в конфигурации MySQL (файл *my.cnf* или *my.ini*) есть или можно добавить параметры для работы с SSL:

```
[mysqld]
ssl-ca=/path/to/ca.pem
ssl-cert=/path/to/server-cert.pem
ssl-key=/path/to/server-key.pem
require_secure_transport=ON
```

   - Перезапустите сервер MySQL для применения изменений.
2. Генерация SSL-сертификатов (если нет готовых)
   - Сгенерируйте самоподписанные сертификаты с помощью OpenSSL (или используйте готовые):

```
openssl req -new -x509 -days 365 -nodes -out ca.pem -keyout ca-key.pem
openssl req -newkey rsa:2048 -days 365 -nodes -keyout server-key.pem -out server-req.pem
openssl rsa -in server-key.pem -out server-key.pem
openssl x509 -req -in server-req.pem -days 365 -CA ca.pem -CAkey ca-key.pem -set_serial 01 -out server-cert.pem
```

   - Разместите файлы сертификатов в директории, указанной в конфигурации MySQL.
3. Проверка поддержки SSL в MySQL
   - Выполните SQL-запрос для проверки статуса SSL:

```
SHOW VARIABLES LIKE '%ssl%';
```

   - Убедитесь, что переменные *have_ssl* и *have_openssl* равны *YES*.
   - Подключитесь к серверу и выполните:

```
SHOW STATUS LIKE 'Ssl_cipher';
```
   - Если значение не пустое, соединение зашифровано.
4. Создание пользователей с требованием SSL
   - Создайте пользователя, которому разрешено подключаться только по SSL:
```
CREATE USER 'ssl_user'@'%' IDENTIFIED BY 'password' REQUIRE SSL;
```
   - Назначьте ему необходимые права:
```
GRANT SELECT, INSERT ON mydb.* TO 'ssl_user'@'%';
```
   - Создайте пользователя без требования SSL для сравнения:
```
CREATE USER 'plain_user'@'%' IDENTIFIED BY 'password';
GRANT SELECT ON mydb.* TO 'plain_user'@'%';
```
5. Тестирование соединений
   - Попробуйте подключиться к серверу от имени *ssl_user* без использования SSL (например, через *mysql* без параметров *--ssl-mode=REQUIRED*). Соединение должно быть отклонено.
   - Подключитесь с явным требованием SSL:
```
mysql --user=ssl_user --password --host=localhost --ssl-mode=REQUIRED
```
   - Убедитесь, что соединение установлено успешно.
   - Проверьте, что пользователь *plain_user* может подключиться без SSL.
   - Если в конфигурации включено *require_secure_transport=ON*, убедитесь, что все соединения без SSL блокируются.
6. Настройка аутентификации (плагинов)
   - Посмотрите, какой плагин аутентификации используется по умолчанию:
```
SHOW VARIABLES LIKE 'default_authentication_plugin';
```
   - Создайте пользователя с явно заданным плагином:
```
CREATE USER 'native_user'@'%' IDENTIFIED WITH mysql_native_password BY
'password';
CREATE USER 'sha256_user'@'%' IDENTIFIED WITH caching_sha2_password BY
'password';
```
   - Проверьте, что пользователи могут подключаться, и сравните поведение разных плагинов.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.3. Управление доступом к базам данных

Лабораторное занятие № 44
Аудит действий пользователей в базе данных с помощью встроенных инструментов PostgreSQL

Цель: освоить навыки настройки и использования встроенных инструментов аудита в PostgreSQL для отслеживания действий пользователей, анализа событий безопасности и формирования отчётов о работе с базой данных.

Выполнив работу, вы будете уметь:
У 2.2.1 Осуществлять основные функции по администрированию баз данных;

У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
PostgreSQL, pgAdmin, DBeaver

Задание:

1. Изучить возможности аудита в PostgreSQL (pgAudit, стандартные средства логирования).
2. Включить и настроить модуль pgAudit для сбора детальных логов действий пользователей.
3. Создать тестовых пользователей и выполнить от их имени различные SQL-операции.
4. Проанализировать журналы аудита, найти записи о выполненных действиях.
5. Оформить результаты работы в виде отчёта.

Порядок выполнения работы:

1. Подготовка и теоретическая часть
   - Подключитесь к серверу PostgreSQL с правами супер-пользователя.
   - Ознакомьтесь с механизмами аудита:
     - стандартные средства логирования (logging_collector, log_statement);
     - расширение pgAudit для расширенного аудита.
   - Убедитесь, что в вашей версии PostgreSQL поддерживается pgAudit (обычно для версий 10 и выше).
2. Установка и настройка pgAudit
   - Установите расширение pgAudit.
   - Добавьте shared_preload_libraries = 'pgaudit' в конфигурационный файл *postgresql.conf*.
   - Перезапустите сервер PostgreSQL для загрузки модуля.
   - В базе данных выполните:
   ```
   CREATE EXTENSION pgaudit;
   ```
   - Настройте параметры логирования в *postgresql.conf*:
     - *logging_collector=on*;
     - *log_destination='stderr'* (или 'csvlog' для удобства анализа);
     - *log_directory='pg_log'*;
     - *log_filename='postgresql-%Y-%m-%d_%H%M%S.log'*.
   - Перезапустите сервер.
3. Настройка политик аудита
   - Создайте тестовых пользователей:
   ```
   CREATE USER auditor WITH PASSWORD 'audpass';
   CREATE USER testuser WITH PASSWORD 'testpass';
   ```
   - Назначьте права:
   ```
   GRANT CONNECT ON DATABASE mydb TO auditor, testuser;
   GRANT USAGE ON SCHEMA public TO auditor, testuser;
   GRANT SELECT, INSERT ON ALL TABLES IN SCHEMA public TO testuser;
   ```
   - Настройте аудит для базы данных или отдельных объектов. Например, чтобы логировать все действия:
   ```
   ALTER DATABASE mydb SET pgaudit.log = 'all';
   ```
   - Или для конкретных таблиц:
   ```
   ALTER TABLE public.employees SET (pgaudit.log = 'all');
   ```
4. Проведение тестовых операций
   - Подключитесь к базе данных под именем *testuser* и выполните различные действия:
     - SELECT * FROM public.employees LIMIT 5;

- INSERT INTO public.employees ...;
- UPDATE public.employees ...;
- DELETE FROM public.employees ...;

- Попробуйте выполнить действие, на которое у пользователя нет прав (например, DROP TABLE), чтобы увидеть запись о нарушении.

5. Анализ журналов аудита
   - Найдите файлы журналов в каталоге *pg_log* (или указанном в *log_directory*).
   - Откройте последний файл журнала и найдите записи, связанные с действиями *testuser*.
   - Проанализируйте структуру записей аудита: время, имя пользователя, база данных, тип операции, объект, текст запроса.
   - Для удобства анализа можно использовать фильтры по имени пользователя или типу события.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.4. Резервное копирование баз данных

Лабораторное занятие № 45
Создание резервной копии базы данных MySQL с использованием утилиты mysqldump

Цель: освоить навыки резервного копирования базы данных MySQL с помощью утилиты командной строки mysqldump для обеспечения сохранности и восстановления данных.

Выполнив работу, вы будете уметь:
У 2.2.1 Осуществлять основные функции по администрированию баз данных;
У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных
У 2.3.1 Дать независимую оценку уровня безопасности
У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
MySQL

Задание:
1. Изучить назначение и возможности утилиты mysqldump.
2. Создать резервную копию указанной базы данных.
3. Проверить целостность созданного файла резервной копии.
4. Восстановить базу данных из созданной резервной копии.
5. Оформить отчёт о проделанной работе.

Порядок выполнения работы:

1. Подготовка
   - Убедитесь, что на сервере установлен MySQL и доступна утилита mysqldump.
   - Получите доступ к командной строке сервера или используйте терминал.
   - Определите имя пользователя, пароль и название базы данных, для которой будет создаваться резервная копия.
2. Создание резервной копии

Откройте терминал и выполните команду:

```
mysqldump -u [имя_пользователя] -p [имя_базы_данных] > backup.sql
```

Введите пароль пользователя по запросу.

3. Проверка резервной копии
   - Убедитесь, что файл backup.sql создан и не пуст.
   - Просмотрите первые строки файла командой:

```
head backup.sql
```

4. Восстановление базы данных из резервной копии
   - Создайте новую пустую базу данных (или очистите существующую):

```
mysqladmin -u [имя_пользователя] -p create [имя_новой_базы_данных]
```

   - Восстановите данные из файла:

```
mysql -u [имя_пользователя] -p [имя_новой_базы_данных] < backup.sql
```

5. Проверка восстановления
   - Подключитесь к новой базе данных и убедитесь, что таблицы и данные восстановлены корректно.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.4. Резервное копирование баз данных

Лабораторное занятие № 46
Резервное копирование базы данных PostgreSQL с помощью pg_dump и pg_dumpall

Цель: освоить навыки резервного копирования баз данных и всей системы PostgreSQL с использованием утилит pg_dump и pg_dumpall для обеспечения сохранности данных и возможности их восстановления.

Выполнив работу, вы будете уметь:
У 2.2.1 Осуществлять основные функции по администрированию баз данных;
У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных
У 2.3.1 Дать независимую оценку уровня безопасности
У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
PostgreSQL

Задание:

1. Изучить назначение и возможности утилит pg_dump и pg_dumpall.
2. Создать резервную копию отдельной базы данных с помощью pg_dump.
3. Создать резервную копию всех баз данных и ролей с помощью pg_dumpall.
4. Проверить целостность созданных файлов резервных копий.
5. Оформить отчёт о проделанной работе.

Порядок выполнения работы:

1. Подготовка
   - Убедитесь, что на сервере установлен PostgreSQL и доступны утилиты pg_dump и pg_dumpall.
   - Получите доступ к командной строке сервера или используйте терминал.
   - Определите имя пользователя, пароль и название базы данных, для которой будет создаваться резервная копия.
2. Создание резервной копии отдельной базы данных (pg_dump)

   Откройте терминал и выполните команду:

   ```
   pg_dump -U [имя_пользователя]
   -F c -f backup_db.dump [имя_базы_данных]
   ```

   Введите пароль пользователя по запросу.

   `-F c` — формат custom (рекомендуется для гибкости восстановления).

   `-f` — имя выходного файла.
3. Проверка резервных копий
   - Убедитесь, что файлы backup_db.dump и backup_all.sql созданы и не пусты.
   - Просмотрите первые строки файлов командами:

   ```
   head backup_db.dump
   head backup_all.sql
   ```
4. Восстановление базы данных из резервной копии (pg_restore)
   - Создайте новую пустую базу данных:

   ```
   createdb -U [имя_пользователя] [имя_новой_базы_данных]
   ```
   - Восстановите данные из файла:

   ```
   pg_restore -U [имя_пользователя] -d [имя_новой_базы_данных]
   backup_db.dump
   ```
5. Восстановление всех баз данных (psql)
   - Восстановите все базы и роли:

   ```
   psql -U [имя_пользователя] -f backup_all.sql postgres
   ```
6. Проверка восстановления
   - Подключитесь к восстановленным базам данных и убедитесь, что таблицы и данные восстановлены корректно.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.4. Резервное копирование баз данных

Лабораторное занятие № 47
Настройка и выполнение резервного копирования в Microsoft SQL Server с использованием SSMS

Цель: освоить навыки создания и настройки заданий резервного копирования баз данных в Microsoft SQL Server с помощью SQL Server Management Studio (SSMS) для обеспечения сохранности и возможности восстановления данных.

Выполнив работу, вы будете уметь:

У 2.2.1 Осуществлять основные функции по администрированию баз данных;

У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных

У 2.3.1 Дать независимую оценку уровня безопасности

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;

У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:

Microsoft SQL Server

Задание:

1. Изучить интерфейс SSMS и инструменты для резервного копирования.
2. Создать резервную копию пользовательской базы данных с помощью мастера планов обслуживания.
3. Настроить расписание автоматического выполнения резервного копирования.
4. Проверить наличие и целостность созданных файлов резервных копий.
5. Оформить отчёт о проделанной работе.

Порядок выполнения работы:

1. Подготовка
   - Убедитесь, что на компьютере установлен Microsoft SQL Server и SQL Server Management Studio (SSMS).
   - Запустите SSMS и подключитесь к нужному экземпляру сервера.
   - Определите имя базы данных, для которой будет выполняться резервное копирование.
2. Создание плана обслуживания для резервного копирования
   - В обозревателе объектов (Object Explorer) раскройте узел Management → Maintenance Plans.
   - Щёлкните правой кнопкой мыши по Maintenance Plans и выберите Maintenance Plan Wizard.
   - Введите имя плана (например, BackupPlan) и описание.
   - На шаге выбора задач (Select Maintenance Tasks) отметьте Back Up Database (Full).
   - На шаге выбора баз данных (Select Database(s)) укажите нужную базу данных.
   - Настройте параметры резервного копирования:
     - Выберите тип резервной копии (рекомендуется Full).
     - Укажите путь к папке для хранения файлов резервных копий.
     - При необходимости включите проверку целостности (Verify backup integrity).

- Завершите работу мастера, при желании настройте расписание выполнения (можно сделать это позже).

3. Настройка расписания
   - После создания плана щёлкните по нему правой кнопкой мыши и выберите Modify.
   - В нижней части окна настройте расписание (Schedule):
     - Укажите частоту (например, ежедневно, еженедельно).
     - Задайте время запуска.
   - Сохраните изменения.
4. Запуск и проверка выполнения
   - Для немедленного запуска щёлкните по плану правой кнопкой мыши и выберите Execute.
   - Дождитесь завершения задачи.
   - Проверьте журнал выполнения (History) на наличие ошибок.
   - Перейдите в указанную папку и убедитесь, что файл резервной копии создан и имеет ненулевой размер.
5. Восстановление из резервной копии (опционально)
   - Щёлкните правой кнопкой мыши по базе данных → Tasks → Restore → Database.
   - Выберите источник восстановления (From device), укажите файл резервной копии.
   - Настройте параметры восстановления и выполните восстановление.
   - Проверьте, что данные восстановлены корректно.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.4. Резервное копирование баз данных

Лабораторное занятие № 48
Автоматизация резервного копирования базы данных MongoDB с использованием скриптов

Цель: освоить навыки автоматизации процесса резервного копирования базы данных MongoDB с помощью встроенных утилит и скриптов для регулярного и надёжного сохранения данных.

Выполнив работу, вы будете уметь:
У 2.2.1 Осуществлять основные функции по администрированию баз данных;
У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных
У 2.3.1 Дать независимую оценку уровня безопасности
У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
MongoDB

Задание:

1. Изучить утилиты mongodump и mongorestore для работы с резервными копиями MongoDB.
2. Написать скрипт для создания резервной копии указанной базы данных.
3. Настроить автоматическое выполнение скрипта по расписанию (например, с помощью cron в Linux или Task Scheduler в Windows).
4. Проверить работоспособность скрипта и наличие созданных резервных копий.
5. Оформить отчёт о проделанной работе.

Порядок выполнения работы:

1. Подготовка
   - Убедитесь, что на сервере установлен MongoDB и доступны утилиты командной строки (mongodump, mongorestore).
   - Получите доступ к терминалу (Linux/macOS) или командной строке (Windows).
   - Определите имя пользователя, пароль (если используется аутентификация) и название базы данных для резервного копирования.
   - Создайте папку для хранения резервных копий, например /backups/mongodb.
2. Создание скрипта для резервного копирования
   - Откройте текстовый редактор и создайте новый файл, например, backup_mongo.sh (для Linux/macOS) или backup_mongo.bat (для Windows).

   Пример для Linux/macOS (backup_mongo.sh):

```
#!/bin/bash

DATE=$(date +%Y-%m-%d)
BACKUP_DIR="/backups/mongodb/$DATE"
MONGO_DB="mydatabase"
MONGO_USER="user"
MONGO_PASS="password"
MONGO_HOST="localhost"

mkdir -p $BACKUP_DIR
mongodump --host $MONGO_HOST -u $MONGO_USER -p $MONGO_PASS --db
$MONGO_DB --out $BACKUP_DIR
echo "Резервная копия создана: $BACKUP_DIR"
```

   - Сохраните файл и сделайте его исполняемым:chmod +x backup_mongo.sh

   Пример для Windows (backup_mongo.bat):

```
@echo off
set DATE=%DATE:~10,4%-%DATE:~4,2%-%DATE:~7,2%
set BACKUP_DIR=C:\backups\mongodb\%DATE%
set MONGO_DB=mydatabase
set MONGO_USER=user
set MONGO_PASS=password
set MONGO_HOST=localhost

mkdir %BACKUP_DIR%
mongodump --host %MONGO_HOST% -u %MONGO_USER% -p %MONGO_PASS% --
db %MONGO_DB% --out %BACKUP_DIR%
echo Резервная копия создана: %BACKUP_DIR%
```

3. Тестирование скрипта
   - Запустите скрипт вручную из терминала/командной строки:

- Linux/macOS: ./backup_mongo.sh
- Windows: backup_mongo.bat

– Убедитесь, что в папке /backups/mongodb/2026-04-23 (или аналогичной) появились файлы резервной копии.

4. Автоматизация выполнения скрипта

Для Linux/macOS (cron):

– Откройте редактор crontab:

```
crontab -e
```

– Добавьте строку для ежедневного запуска в 02:00:

```
0 2 * * * /путь/к/backup_mongo.sh
```

– Сохраните и закройте редактор.

Для Windows (Task Scheduler):

– Откройте «Планировщик заданий».
– Создайте новую задачу, укажите имя и описание.
– На вкладке «Триггеры» настройте расписание (например, ежедневно).
– На вкладке «Действия» добавьте новое действие:
  – Программа или сценарий: путь к cmd.exe
  – Аргументы: /c "C:\путь\к\backup_mongo.bat"
– Сохраните задачу.

5. Проверка автоматизации
   – Дождитесь времени выполнения задачи или запустите её вручную через планировщик.
   – Проверьте, что новая папка с резервной копией создана автоматически.
6. Восстановление из резервной копии (опционально)
   – Для проверки восстановления используйте команду:

```
mongorestore --db mydatabase /backups/mongodb/2026-04-23/mydatabase
```

   – Убедитесь, что данные успешно восстановлены в тестовой базе.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.5. Восстановление баз данных

Лабораторное занятие № 49
Восстановление данных из резервной копии MySQL с проверкой целостности данных

Цель: освоить практические навыки восстановления базы данных MySQL из резервной копии, а также научиться проверять целостность и корректность восстановленных данных.

Выполнив работу, вы будете уметь:
У 2.2.1 Осуществлять основные функции по администрированию баз данных;
У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных

У 2.3.1 Дать независимую оценку уровня безопасности

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;

У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
MySQL

Задание:

1. Изучить процесс восстановления базы данных из резервной копии, созданной утилитой mysqldump.
2. Восстановить базу данных из файла .sql в новую (тестовую) базу.
3. Выполнить проверку целостности и полноты восстановленных данных.
4. Оформить отчёт о проделанной работе.

Порядок выполнения работы:

1. Подготовка
   - Убедитесь, что на сервере установлен MySQL и доступна утилита командной строки mysql.
   - Получите доступ к терминалу или командной строке сервера.
   - Убедитесь, что у вас есть файл резервной копии (например, backup.sql) и известны параметры доступа к серверу MySQL (имя пользователя, пароль, имя хоста).
2. Создание тестовой базы данных
   - Подключитесь к серверу MySQL:

   ```
   mysql -u [имя_пользователя] -p
   ```

   - Введите пароль.
   - Создайте новую пустую базу данных для восстановления:

   ```
   CREATE DATABASE db_restore_test;
   ```

   - Выйдите из консоли MySQL:

   ```
   EXIT;
   ```

3. Восстановление базы данных из резервной копии
   - Выполните команду восстановления:

   ```
   mysql -u [имя_пользователя] -p db_restore_test < backup.sql
   ```

   - Введите пароль пользователя по запросу.
   - Дождитесь завершения процесса. В случае успеха в терминале не появится сообщений об ошибках.
4. Проверка целостности и полноты данных
   - Подключитесь к восстановленной базе:

   ```
   mysql -u [имя_пользователя] -p db_restore_test
   ```

   - Проверьте список таблиц:

   ```
   SHOW TABLES;
   ```

   - Сравните с перечнем таблиц в исходной базе.
   - Проверьте количество строк в ключевых таблицах:

   ```
   SELECT COUNT(*) FROM имя_таблицы;
   ```

   - Сравните с аналогичными запросами к исходной базе.
   - Выполните выборочные запросы для проверки содержимого:

   ```
   SELECT * FROM имя_таблицы LIMIT 10;
   ```

   - Проверьте наличие индексов и ограничений:

   ```
   SHOW INDEX FROM имя_таблицы;
   SHOW CREATE TABLE имя_таблицы;
   ```

5. Анализ результатов

- Сравните результаты проверки с исходными данными.
- Убедитесь, что структура таблиц, данные, индексы и ограничения восстановлены корректно.
- Зафиксируйте обнаруженные расхождения (если они есть).

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.5. Восстановление баз данных

Лабораторное занятие № 50
Восстановление базы данных PostgreSQL на новый сервер с сохранением всех параметров

Цель: освоить процесс переноса и восстановления базы данных PostgreSQL на новый сервер с сохранением всех настроек, ролей, схем, данных и прав доступа.

Выполнив работу, вы будете уметь:
У 2.2.1 Осуществлять основные функции по администрированию баз данных;
У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных
У 2.3.1 Дать независимую оценку уровня безопасности
У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
PostgreSQL

Задание:
1. Изучить процесс резервного копирования и восстановления PostgreSQL с помощью утилит pg_dumpall и pg_restore.
2. Выполнить резервное копирование всех баз данных и глобальных объектов (ролей, табличных пространств) со старого сервера.
3. Развернуть PostgreSQL на новом сервере.
4. Восстановить все параметры, роли и базы данных на новом сервере.
5. Проверить целостность и корректность восстановленных данных и прав доступа.
6. Оформить отчёт о проделанной работе.

Порядок выполнения работы:
1. Подготовка
   - Убедитесь, что у вас есть доступ к старому и новому серверам.
   - На обоих серверах должен быть установлен PostgreSQL (желательно одной версии).
   - Определите параметры подключения к старому серверу (имя пользователя, пароль, имя хоста, порты).

- Создайте на новом сервере каталоги для данных и резервных копий.

2. Резервное копирование со старого сервера
   - Резервное копирование глобальных объектов (роли, табличные пространства):
     - Выполните команду:

```
pg_dumpall -g -U [имя_пользователя] -f globals.sql
```

     - Введите пароль по запросу.
   - Резервное копирование всех баз данных:
     - Выполните команду:

```
pg_dumpall -U [имя_пользователя] -f all_databases.sql
```

     - Введите пароль по запросу.
   - (Опционально) Резервное копирование отдельных баз в пользовательском формате:
     - Для ускорения восстановления и гибкости используйте:

```
pg_dump -U [имя_пользователя] -F c -f db1.dump [имя_базы]
```

     - Перенесите файлы globals.sql, all_databases.sql (или отдельные дампы) на новый сервер.
3. Подготовка нового сервера
   - Установите PostgreSQL (если не установлен).
   - Инициализируйте кластер базы данных:

```
 initdb -D /путь/к/каталогу/данных
```

   - Запустите службу PostgreSQL:

```
pg_ctl -D /путь/к/каталогу/данных start
```

   - Убедитесь, что сервер запущен и доступен.
4. Восстановление на новом сервере
   - Восстановление глобальных объектов:
     - Подключитесь к базе postgres:

```
psql -U [имя_пользователя] -d postgres
```

     - Выполните:

```
\i globals.sql
```

   - Восстановление всех баз данных:
     - Если использовался файл all_databases.sql:

```
psql -U [имя_пользователя] -f all_databases.sql postgres
```

     - Если использовались отдельные дампы:

```
pg_restore -U [имя_пользователя] -d [имя_базы] db1.dump
```

   - Проверьте, что все базы и роли созданы:

```
\l (список баз), \du (список ролей).
```

5. Проверка целостности и параметров
   - Подключитесь к восстановленным базам данных.
   - Проверьте структуру таблиц:

```
\d [имя_таблицы]
```

   - Проверьте права доступа:

```
\dp [имя_таблицы]
```

   - Выполните тестовые запросы для проверки данных.
   - Сравните параметры конфигурации (postgresql.conf, pg_hba.conf) на старом и новом серверах.
6. Анализ результатов
   - Убедитесь, что все базы, роли, права доступа и данные перенесены корректно.
   - Зафиксируйте обнаруженные расхождения (если они есть).

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.5. Восстановление баз данных

Лабораторное занятие № 51
Восстановление базы данных Microsoft SQL Server из полной резервной копии с использованием SSMS

Цель: освоить практические навыки восстановления базы данных Microsoft SQL Server из полной резервной копии (.bak) с помощью графического интерфейса SQL Server Management Studio (SSMS).

Выполнив работу, вы будете уметь:

У 2.2.1 Осуществлять основные функции по администрированию баз данных;

У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных

У 2.3.1 Дать независимую оценку уровня безопасности

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;

У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:

Microsoft SQL Server

Задание:

1. Изучить процесс восстановления баз данных в SSMS.
2. Создать новую (пустую) базу данных для восстановления.
3. Выполнить восстановление из существующего файла полной резервной копии.
4. Проверить целостность и корректность восстановленных данных.
5. Оформить отчёт о проделанной работе.

Порядок выполнения работы:

1. Подготовка
   - Убедитесь, что на компьютере установлены Microsoft SQL Server и SQL Server Management Studio (SSMS).
   - Запустите SSMS и подключитесь к целевому экземпляру сервера.
   - Подготовьте файл полной резервной копии (.bak) базы данных. Если его нет, создайте его, выполнив резервное копирование любой тестовой базы.
   - Определите, в какую базу данных будет производиться восстановление (можно создать новую или использовать существующую).
2. Создание новой базы данных
   - В обозревателе объектов (Object Explorer) щёлкните правой кнопкой мыши по узлу Databases.
   - Выберите New Database....
   - Введите имя базы данных (например, DB_Restore_Test).
   - Нажмите OK.
3. Восстановление базы данных из резервной копии
   - Щёлкните правой кнопкой мыши по узлу Databases и выберите Restore Database....
   - В открывшемся окне:

- На вкладке General в поле To database выберите или введите имя базы данных, в которую будет выполнено восстановление (например, DB_Restore_Test).
- В разделе Source for restore выберите Device.
- Нажмите кнопку с тремя точками (...) справа от поля.
- В окне Select backup devices нажмите Add.
- Укажите путь к файлу резервной копии (.bak) и нажмите OK.
- Нажмите OK в окне выбора устройств.

- Перейдите на вкладку Files:
  - Здесь можно изменить пути к файлам данных (.mdf) и журналов транзакций (.ldf) восстановленной базы, если это необходимо (например, если на диске C: мало места).
- Перейдите на вкладку Options:
  - Рекомендуется установить флажок Overwrite the existing database (WITH REPLACE), чтобы перезаписать существующую базу данных.
  - При необходимости установите флажок Take tail-log backup before restore, если восстанавливаете последнюю копию.
- Нажмите OK для запуска процесса восстановления.
- Дождитесь завершения операции. Внизу окна SSMS появится сообщение об успешном восстановлении.

4. Проверка целостности и корректности данных
   - После завершения восстановления обновите узел Databases в обозревателе объектов.
   - Убедитесь, что база данных появилась в списке и её статус — (Online).
   - Разверните базу данных → Tables. Убедитесь, что структура таблиц восстановлена.
   - Выполните несколько тестовых запросов для проверки данных:
     - Раскройте базу данных → New Query.
     - Выполните запросы:

```
SELECT TOP 10 * FROM [ИмяТаблицы];
SELECT COUNT(*) FROM [ИмяТаблицы];
```

   - Сравните результаты с данными в исходной базе (если есть возможность).
5. Анализ результатов
   - Убедитесь, что база данных восстановлена успешно, все таблицы и данные на месте.
   - Зафиксируйте время выполнения операции и объём восстановленных данных.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.5. Восстановление баз данных

Лабораторное занятие № 52
Восстановление базы данных MongoDB из резервного архива

Цель: освоить практические навыки восстановления базы данных MongoDB из резервной копии, созданной с помощью утилиты mongodump, и научиться проверять целостность восстановленных данных.

Выполнив работу, вы будете уметь:

У 2.2.1 Осуществлять основные функции по администрированию баз данных;

У 2.2.2 Настраивать политики безопасности при работе с сервером баз данных

У 2.3.1 Дать независимую оценку уровня безопасности

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;

У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:

MongoDB

Задание:

1. Изучить процесс восстановления базы данных MongoDB с помощью утилиты mongorestore.
2. Восстановить базу данных из ранее созданного архива в новую (тестовую) базу.
3. Проверить целостность и корректность восстановленных данных.
4. Оформить отчёт о проделанной работе.

Порядок выполнения работы:

1. Подготовка
   - Убедитесь, что на сервере установлен MongoDB и доступны утилиты командной строки (mongorestore).
   - Получите доступ к терминалу (Linux/macOS) или командной строке (Windows).
   - Убедитесь, что у вас есть папка с резервной копией (например, /backups/mongodb/2026-04-23), содержащая поддиректорию с именем базы данных.
   - Определите имя пользователя, пароль (если используется аутентификация) и название базы данных для восстановления.
2. Создание тестовой базы данных

Восстановление можно выполнить как в существующую, так и в новую базу данных. Для лабораторной работы рекомендуется создать новую.

   - Подключитесь к серверу MongoDB:mongo
   - Создайте новую пустую базу данных (на самом деле она будет создана при первом добавлении данных):

```
use db_restore_test
```

   - Выйдите из консоли:

```
exit
```

3. Восстановление базы данных из резервного архива
   - Откройте терминал/командную строку.
   - Выполните команду восстановления:

Если аутентификация не требуется:

```
mongorestore --db db_restore_test --drop /backups/mongodb/2026-04-23/dbname
```

Если требуется аутентификация:

```
mongorestore --db db_restore_test --drop -u [имя_пользователя] -p [пароль] --authenticationDatabase admin /backups/mongodb/2026-04-23/dbname
```

Пояснения к ключам:

   - --db db_restore_test — имя базы данных, в которую будут восстановлены данные.

- --drop — важный ключ, который перед восстановлением удаляет все существующие коллекции в целевой базе. Это гарантирует, что структура и данные будут точно соответствовать архиву.
- --authenticationDatabase admin — указывает базу данных, в которой хранятся учётные данные пользователя (обычно admin).
- /backups/mongodb/2026-04-23/dbname — путь к папке, содержащей данные одной конкретной базы данных из архива.

4. Проверка целостности и корректности данных
   - Подключитесь к восстановленной базе данных:

```
mongo db_restore_test
```

   - Проверьте список коллекций (аналог таблиц в SQL):
   - show collections
   - Сравните полученный список со списком коллекций в исходной базе.
   - Проверьте количество документов в ключевых коллекциях:

```
db.collection_name.find().count()
```

   - Выполните выборочные запросы для проверки содержимого документов:

```
db.collection_name.find().limit(5).pretty()
```

   - Убедитесь, что индексы были восстановлены:

```
db.collection_name.getIndexes()
```

   - Выйдите из консоли:

```
exit
```

5. Анализ результатов

Сравните результаты проверки (количество коллекций, документов, структуру индексов) с данными исходной базы на момент создания резервной копии. Убедитесь, что данные восстановлены полностью и корректно.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.6. Мониторинг событий, возникающих в процессе работы баз данных

Лабораторное занятие № 53
Настройка и использование утилиты MySQL Performance Schema для мониторинга работы базы данных

Цель: освоить навыки включения, настройки и использования системной схемы Performance Schema в MySQL для сбора и анализа метрик производительности сервера баз данных, а также выявления «узких мест» в работе запросов.

Выполнив работу, вы будете уметь:
У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;
У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;

У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

У 2.3.1 Дать независимую оценку уровня безопасности

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;

У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
MySQL

Задание:

1. Изучить назначение и архитектуру Performance Schema.
2. Проверить текущее состояние и версию Performance Schema на сервере.
3. Включить и настроить сбор данных о событиях ожидания и использовании потоков.
4. Сгенерировать тестовую нагрузку на базу данных.
5. Выполнить запросы к таблицам Performance Schema для анализа производительности.
6. Оформить отчёт с выводами о работе сервера.

Порядок выполнения работы:

1. Подготовка и проверка состояния
   - Подключитесь к серверу MySQL с правами администратора:

```
mysql -u root -p
```

   - Проверьте, включена ли Performance Schema и её версию:

```
SHOW VARIABLES LIKE 'performance_schema';
SELECT VERSION();
```

   - Убедитесь, что сервер поддерживает эту функцию.
2. Включение и настройка Performance Schema
   - Если функция отключена, её можно включить через конфигурационный файл my.cnf (или my.ini):
     - Добавьте или измените строку:

```
performance_schema = ON
```

     - Перезапустите службу MySQL.
   - (Для лабораторной работы, если нет доступа к конфигу, убедитесь, что она включена).
   - Проверьте, какие инструменты (consumers) и источники (instruments) включены:

```
SELECT * FROM performance_schema.setup_consumers;
SELECT * FROM performance_schema.setup_instruments WHERE ENABLED = 'YES';
```

   - Включите сбор данных о событиях ожидания для SQL-операторов (если отключено):

```
UPDATE performance_schema.setup_consumers SET ENABLED = 'YES' WHERE NAME = 'events_statements_current';
UPDATE performance_schema.setup_instruments SET ENABLED = 'YES', TIMED = 'YES' WHERE NAME = 'statement/sql/select';
```

3. Генерация тестовой нагрузки

Для получения данных для анализа необходимо выполнить несколько запросов. Создайте тестовую таблицу и наполните её данными:

```
CREATE DATABASE IF NOT EXISTS test_perf;
USE test_perf;
CREATE TABLE orders (id INT PRIMARY KEY, customer_name VARCHAR(100), amount DECIMAL(10,2));
-- Вставьте 1000 строк (это может занять время)
INSERT INTO orders (id, customer_name, amount) SELECT seq, CONCAT('User_', seq), RAND()*1000 FROM seq_1_to_1000;
```

Выполните несколько типичных запросов для создания нагрузки:

```
SELECT COUNT(*) FROM orders WHERE amount > 500;
SELECT customer_name, SUM(amount) FROM orders GROUP BY customer_name HAVING SUM(amount) > 1000 LIMIT 10;
```

4. Анализ данных Performance Schema

После выполнения запросов проанализируйте собранные данные.

– Анализ самых долгих запросов:

```
SELECT
    event_name AS statement,
    timer_wait/1000000000000 AS duration_seconds,
    sql_text
FROM performance_schema.events_statements_history_long
ORDER BY timer_wait DESC
LIMIT 5;
```

– Анализ использования потоков (сессий):

```
SELECT
    PROCESSLIST_ID,
    THREAD_ID,
    PROCESSLIST_USER,
    PROCESSLIST_HOST,
    PROCESSLIST_DB,
    PROCESSLIST_COMMAND,
    PROCESSLIST_TIME
FROM performance_schema.threads;
```

– Анализ событий ожидания (I/O, блокировки):

```
SELECT
    EVENT_NAME,
    COUNT_STAR AS total_occurrences,
    SUM_TIMER_WAIT/1000000000000 AS total_wait_seconds
FROM
performance_schema.events_waits_summary_global_by_event_name
ORDER BY total_wait_seconds DESC
LIMIT 10;
```

Форма представления результата:

Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.6. Мониторинг событий, возникающих в процессе работы баз данных

Лабораторное занятие № 54
Использование утилиты pg_stat_activity в PostgreSQL для отслеживания активных соединений и запросов

Цель: изучить возможности утилиты pg_stat_activity в PostgreSQL для мониторинга активных соединений и выполняющихся запросов, а также научиться анализировать нагрузку на сервер баз данных.

Выполнив работу, вы будете уметь:

У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;

У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;

У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

У 2.3.1 Дать независимую оценку уровня безопасности

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;

У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:

PostgreSQL

Задание:

1. Подключиться к базе данных PostgreSQL.
2. Изучить структуру и назначение представления pg_stat_activity.
3. Получить список всех активных соединений к базе данных.
4. Отобразить выполняющиеся в данный момент SQL-запросы.
5. Проанализировать нагрузку на сервер, выявив длительные и блокирующие запросы.
6. Научиться завершать нежелательные или зависшие соединения.

Порядок выполнения работы:

1. Откройте терминал или командную строку и подключитесь к вашей базе данных PostgreSQL с помощью команды:
   ```
   psql -U <имя_пользователя> -d <имя_базы_данных>
   ```
2. Выполните запрос для просмотра всех активных соединений:
   ```
   SELECT * FROM pg_stat_activity;
   ```
   Изучите возвращаемые поля: pid, usename, datname, client_addr, state, query и др.
3. Для отображения только активных (выполняющихся) запросов используйте:
   ```
   SELECT pid, usename, datname, client_addr, query
   FROM pg_stat_activity
   WHERE state = 'active';
   ```
4. Найдите длительные запросы, отсортировав по времени выполнения:
   ```
   SELECT pid, usename, datname, query, now() - query_start AS duration
   FROM pg_stat_activity
   WHERE state = 'active'
   ORDER BY duration DESC;
   ```
5. Если необходимо завершить нежелательное соединение, используйте команду:
   ```
   SELECT pg_terminate_backend(<pid>);
   ```
   где <pid> — идентификатор процесса, который нужно завершить.
6. Сделайте выводы по результатам работы: сколько активных соединений, какие запросы выполняются дольше всего, были ли обнаружены блокирующие процессы.

Форма представления результата:

Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.6. Мониторинг событий, возникающих в процессе работы баз данных

Лабораторное занятие № 55
Анализ блокировок и ожиданий в Microsoft SQL Server с помощью DMVs (Dynamic Management Views)

Цель: изучить механизмы блокировок и ожиданий в Microsoft SQL Server, а также научиться использовать динамические административные представления (Dynamic Management Views, DMVs) для анализа и диагностики проблем, связанных с конкуренцией за ресурсы.

Выполнив работу, вы будете уметь:

У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;

У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;

У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

У 2.3.1 Дать независимую оценку уровня безопасности

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;

У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:

Microsoft SQL Server

Задание:

1. Ознакомиться с основными DMV, предназначенными для анализа блокировок и ожиданий.
2. Научиться определять текущие блокировки в системе.
3. Выявить процессы, которые находятся в состоянии ожидания ресурсов.
4. Проанализировать причины блокировок и определить, какие объекты и запросы участвуют в конфликте.
5. Научиться завершать нежелательные или зависшие сессии.

Порядок выполнения работы:

1. Подключитесь к вашему экземпляру Microsoft SQL Server с помощью SQL Server Management Studio (SSMS) или другого инструмента для работы с T-SQL.
2. Анализ текущих блокировок

Выполните запрос для просмотра активных блокировок:

```
SELECT
    request_session_id AS session_id,
    resource_type,
    resource_database_id,
    resource_description,
```

```
    request_mode,
    request_status
FROM sys.dm_tran_locks;
```

Изучите типы ресурсов, режимы блокировок и статусы запросов.

3. Определение сессий, ожидающих ресурсы

Выполните запрос для поиска сессий, находящихся в состоянии ожидания:

```
SELECT
    session_id,
    wait_type,
    wait_duration_ms,
    blocking_session_id,
    resource_description
FROM sys.dm_os_waiting_tasks
WHERE session_id > 50; -- системные сессии обычно < 50
```

Обратите внимание на тип ожидания, длительность и идентификатор блокирующей сессии.

4. Анализ блокирующих и блокируемых запросов

Для более детального анализа выполните:

```
SELECT
    wt.session_id,
    wt.blocking_session_id,
    wt.wait_duration_ms,
    wt.wait_type,
    wt.resource_description,
    t.text AS sql_text,
    s.login_name,
    s.host_name
FROM sys.dm_os_waiting_tasks wt
JOIN sys.dm_exec_sessions s ON wt.session_id = s.session_id
LEFT JOIN sys.dm_exec_requests r ON wt.session_id = r.session_id
CROSS APPLY sys.dm_exec_sql_text(r.sql_handle) t
WHERE s.session_id > 50;
```

Это позволит увидеть, какие именно запросы вызывают блокировки.

5. Завершение нежелательной сессии

Если необходимо завершить сессию, используйте команду:

```
KILL <session_id>;
```

где <session_id> — идентификатор сессии, которую нужно прервать.

6. Сделайте выводы по результатам работы: какие типы блокировок встречаются чаще всего, какие запросы вызывают наибольшие ожидания, как быстро удаётся разрешать конфликты.

Форма представления результата:

Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.6. Мониторинг событий, возникающих в процессе работы баз данных

Лабораторное занятие № 56
Установка и настройка Prometheus для сбора метрик производительности базы данных MySQL

Цель: освоить процесс установки и базовой настройки системы мониторинга Prometheus для сбора и визуализации метрик производительности базы данных MySQL с помощью экспортёра mysqld_exporter.

Выполнив работу, вы будете уметь:

У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;

У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;

У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

У 2.3.1 Дать независимую оценку уровня безопасности

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;

У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
Microsoft SQL Server

Задание:

1. Установить и настроить Prometheus.
2. Развернуть и подключить экспортёр mysqld_exporter для сбора метрик из MySQL.
3. Настроить Prometheus для регулярного сбора метрик с экспортёра.
4. Проверить доступность метрик и убедиться, что данные о производительности MySQL собираются корректно.
5. Проанализировать основные метрики производительности базы данных через веб-интерфейс Prometheus.

Порядок выполнения работы:

1. Установка Prometheus
   - Скачайте последнюю версию Prometheus для вашей операционной системы с официального сайта.
   - Распакуйте архив и перейдите в каталог Prometheus.
   - Убедитесь, что сервис Prometheus запускается и доступен по адресу http://localhost:9090.
2. Установка и настройка mysqld_exporter
   - Скачайте mysqld_exporter с официальной страницы проекта на GitHub.
   - Запустите экспортёр, указав параметры подключения к вашей базе данных MySQL:

```
./mysqld_exporter --config.my-cnf=/path/to/my.cnf
```

В файле my.cnf должны быть указаны логин и пароль пользователя с правами на просмотр метрик.

   - Проверьте, что экспортёр запущен и метрики доступны по адресу `http://localhost:9104/metrics`.
3. Настройка Prometheus для сбора метрик
   - Откройте файл конфигурации prometheus.yml.
   - Добавьте новый job для сбора метрик с экспортёра:

```
scrape_configs:
  - job_name: 'mysql'
```

```
    static_configs:
      - targets: ['localhost:9104']
```

- Перезапустите Prometheus, чтобы применить изменения.

4. Проверка сбора метрик
   - Откройте веб-интерфейс Prometheus (http://localhost:9090).
   - Перейдите в раздел Status → Targets, убедитесь, что цель localhost:9104 имеет статус UP.
   - Выполните тестовый запрос к метрикам, например:

```
up{job="mysql"}
```

   Убедитесь, что значение равно 1.
5. Анализ метрик MySQL
   - Изучите основные метрики, такие как:
     - mysql_global_status_threads_connected — количество активных соединений.
     - mysql_global_status_queries — общее число запросов.
     - mysql_global_status_innodb_row_lock_waits — ожидание блокировок строк.
   - Сформируйте графики или таблицы для анализа нагрузки на базу данных.
6. Сделайте выводы по результатам работы: какие метрики наиболее информативны, как часто обновляются данные, какие проблемы производительности можно выявить с помощью Prometheus.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.6. Мониторинг событий, возникающих в процессе работы баз данных

Лабораторное занятие № 57
Использование MongoDB Profiler для отслеживания производительности запросов

Цель: изучить возможности встроенного профилировщика MongoDB для анализа производительности запросов, научиться выявлять медленные операции и оптимизировать работу базы данных.

Выполнив работу, вы будете уметь:
У 2.1.1 Производить идентификацию проблем, связанных с нормальным функционированием базы данных;
У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;
У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;
У 2.3.1 Дать независимую оценку уровня безопасности
У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации;
У 2.4.1 Производить формирование требований к обработке данных и их извлечению.

Материальное обеспечение:
MongoDB

Задание:

1. Изучить назначение и уровни работы профилировщика MongoDB.
2. Включить профилировщик и настроить его для сбора информации о медленных запросах.
3. Выполнить тестовые запросы к базе данных.
4. Проанализировать собранные профилировщиком данные.
5. Научиться интерпретировать результаты профилирования и предлагать способы оптимизации

Порядок выполнения работы:

1. Подключение к MongoDB
   - Откройте терминал или командную строку.
   - Подключитесь к вашему экземпляру MongoDB с помощью оболочки mongo или mongosh:

   ```
   mongo <имя_базы_данных>
   ```
2. Изучение текущего состояния профилировщика
   - Выполните команду для просмотра текущего уровня профилирования:

   ```
   db.getProfilingStatus()
   ```
   - Ознакомьтесь с возможными уровнями:
     - 0 — профилирование отключено.
     - 1 — профилируются только медленные запросы (по умолчанию > 100 мс).
     - 2 — профилируются все операции.
3. Включение и настройка профилировщика
   - Включите профилирование всех запросов (уровень 2) для сбора максимального объёма данных:

   ```
   db.setProfilingLevel(2)
   ```
   - (Опционально) Измените порог медленных запросов, например, до 10 мс:

   ```
   db.setProfilingLevel(1, { slowms: 10 })
   ```
4. Выполнение тестовых запросов
   - Выполните несколько различных запросов к вашей базе данных: поиск, обновление, агрегация, сортировка по большим коллекциям.
   - Используйте как быстрые, так и заведомо медленные запросы (например, без индексов).
5. Анализ данных профилировщика
   - Просмотрите собранные данные профилирования:

   ```
   db.system.profile.find().pretty()
   ```
   - Изучите ключевые поля в документах профиля:
     - op — тип операции (query, insert, update, command).
     - ns — пространство имён (база.коллекция).
     - query — параметры запроса.
     - nreturned — количество возвращённых документов.
     - millis — время выполнения запроса в миллисекундах.
     - planSummary — краткое описание плана выполнения запроса (например, COLLSCAN, IXSCAN).
6. Интерпретация результатов и оптимизация
   - Найдите самые медленные запросы.
   - Проанализируйте план выполнения (planSummary). Если используется COLLSCAN (полный обход коллекции), рассмотрите возможность создания индекса.

- Предложите способы оптимизации: создание индексов, реструктуризация запроса, изменение схемы данных.

7. Отключение профилировщика
   - После завершения анализа отключите профилирование или верните его на минимальный уровень:

   ```
   db.setProfilingLevel(0)
   ```
8. Сделайте выводы по результатам работы: какие запросы оказались наиболее ресурсоёмкими, как влияет наличие индексов на производительность, какие рекомендации по оптимизации вы можете дать на основе анализа профиля.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.7. Протоколирование событий, возникающих в процессе работы баз данных

Лабораторное занятие № 58
Настройка и анализ журнала ошибок (error log) в MySQL

Цель: научиться настраивать журнал ошибок (error log) в MySQL, анализировать его содержимое для диагностики проблем, выявлять критические и предупреждающие события, а также оптимизировать параметры логирования для повышения надёжности и безопасности работы сервера.

Выполнив работу, вы будете уметь:
У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;
У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;
У 2.3.1 Дать независимую оценку уровня безопасности
У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации.

Материальное обеспечение:
MySQL

Задание:
1. Изучить назначение и структуру журнала ошибок MySQL.
2. Найти и просмотреть текущий журнал ошибок на сервере.
3. Изменить параметры конфигурации для управления журналом ошибок (путь, ротация, уровень детализации).
4. Сгенерировать тестовые ошибки и предупреждения.
5. Проанализировать записи в журнале, определить причины возникших проблем.
6. Сделать выводы о важности регулярного анализа журнала ошибок для администрирования MySQL.

Порядок выполнения работы:

1. Изучение текущего журнала ошибок
   - Подключитесь к серверу, на котором установлен MySQL.
   - Определите путь к журналу ошибок. Для этого выполните команду в оболочке MySQL:

     ```
     SHOW VARIABLES LIKE 'log_error';
     ```
   - Откройте файл журнала ошибок с помощью любого текстового редактора или команды tail/cat в терминале.
   - Ознакомьтесь с форматом записей: дата и время, уровень ошибки (Error, Warning, Note), идентификатор процесса, текст сообщения.
2. Генерация тестовых событий
   - Выполните несколько действий, которые гарантированно вызовут записи в журнале:
     - Попытайтесь подключиться с неверным паролем.
     - Создайте таблицу с уже существующим именем.
     - Запустите запрос к несуществующей таблице.
     - Попробуйте создать пользователя с некорректными правами.
   - После каждого действия проверяйте появление новых записей в журнале ошибок.
3. Настройка параметров журнала ошибок
   - Откройте конфигурационный файл MySQL (обычно my.cnf или my.ini).
   - Найдите или добавьте секцию [mysqld].
   - Настройте следующие параметры:
     - log_error=/path/to/your/custom_error.log — укажите собственный путь к файлу журнала.
     - log_warnings=2 — включить расширенное логирование предупреждений.
     - log_error_verbosity=3 — максимальный уровень детализации сообщений.
   - Перезапустите службу MySQL, чтобы изменения вступили в силу.
4. Анализ записей журнала
   - Повторно выполните действия, вызывающие ошибки и предупреждения.
   - Откройте настроенный журнал ошибок и проанализируйте новые записи.
   - Определите:
     - Время и причину каждой ошибки.
     - Тип события (критическая ошибка, предупреждение, информационное сообщение).
     - Идентификатор процесса или пользователя, вызвавшего проблему.
5. Формирование выводов
   - Опишите, как журнал ошибок помогает выявлять проблемы с безопасностью (неудачные попытки входа), производительностью (медленные запросы, блокировки) и стабильностью (аварийные завершения).
   - Сделайте вывод о необходимости регулярного мониторинга и анализа журнала ошибок для своевременного реагирования на инциденты.
6. Завершение работы
   - Верните параметры конфигурации к исходным значениям (если это требуется).
   - Остановите тестовые сессии и очистите созданные для эксперимента объекты.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.7. Протоколирование событий, возникающих в процессе работы баз данных

Лабораторное занятие № 59
Конфигурация и просмотр логов событий в PostgreSQL с использованием параметра logging_collector

Цель: освоить навыки настройки и анализа логов событий в PostgreSQL с использованием параметра logging_collector, научиться контролировать и интерпретировать записи о событиях в системе для диагностики и обеспечения безопасности.

Выполнив работу, вы будете уметь:

У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;

У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

У 2.3.1 Дать независимую оценку уровня безопасности

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации.

Материальное обеспечение:

PostgreSQL

Задание:

1. Изучить назначение и принцип работы параметра logging_collector в PostgreSQL.
2. Настроить параметры логирования в конфигурационном файле postgresql.conf для сбора логов событий.
3. Запустить или перезапустить сервер PostgreSQL для применения изменений.
4. Сгенерировать события, которые будут отражены в логах (например, ошибки, успешные подключения, выполнение запросов).
5. Найти и проанализировать созданные лог-файлы, определить, какие события были зафиксированы.
6. Сделать выводы о возможностях мониторинга и диагностики с помощью логов событий.

Порядок выполнения работы:

1. Открыть конфигурационный файл postgresql.conf (обычно находится в каталоге данных PostgreSQL).
2. Найти и установить следующие параметры:
   - logging_collector = on — включить сборщик логов.
   - log_destination = 'stderr' или 'csvlog' — выбрать формат и способ хранения логов.
   - log_directory = 'pg_log' — указать каталог для хранения логов.
   - log_filename = 'postgresql-%Y-%m-%d_%H%M%S.log' — задать шаблон имени файла.

- При необходимости настроить другие параметры (например, log_statement, log_min_duration_statement).

3. Сохранить изменения и перезапустить сервер PostgreSQL.
4. Выполнить ряд действий в базе данных, которые должны быть отражены в логах (например, создать таблицу, вызвать ошибку, подключиться к базе).
5. Перейти в каталог логов и просмотреть созданные файлы.
6. Проанализировать содержимое логов, найти записи о выполненных действиях.
7. Оформить отчёт с описанием выполненных шагов, примерами записей из логов и выводами о возможностях использования логирования для администрирования и диагностики.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.
Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.
Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.7. Протоколирование событий, возникающих в процессе работы баз данных

Лабораторное занятие № 60
Протоколирование событий доступа к данным в Microsoft SQL Server и анализ логов

Цель: освоить методы протоколирования событий доступа к данным в Microsoft SQL Server, научиться настраивать аудит, анализировать и интерпретировать журналы аудита для обеспечения безопасности и контроля доступа к информации.

Выполнив работу, вы будете уметь:
У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;
У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;
У 2.3.1 Дать независимую оценку уровня безопасности
У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации.

Материальное обеспечение:
Microsoft SQL Server

Задание:
1. Изучить возможности аудита доступа к данным в Microsoft SQL Server.
2. Настроить аудит (SQL Server Audit) для протоколирования событий доступа к определённой таблице или базе данных.
3. Сгенерировать события доступа (например, SELECT, INSERT, UPDATE, DELETE) к данным.
4. Проанализировать созданные журналы аудита с помощью встроенных средств SQL Server.

5. Сделать выводы о возможностях мониторинга и контроля доступа к данным на основе логов.

Порядок выполнения работы:

1. Создание объекта аудита
   - В среде SQL Server Management Studio (SSMS) создать новый объект аудита (Audit), указав имя и место хранения логов (например, в файл или в журнал приложений Windows).
2. Настройка спецификации аудита
   - Создать спецификацию аудита базы данных (Database Audit Specification), указав, какие события необходимо протоколировать (например, доступ к определённой таблице, действия пользователей).
3. Включение аудита
   - Включить созданный аудит и спецификацию аудита.
4. Генерация событий
   - Выполнить ряд операций с данными (например, выполнить SELECT, INSERT, UPDATE, DELETE к выбранной таблице), чтобы в журнале появились соответствующие записи.
5. Анализ логов
   - Открыть журналы аудита с помощью SSMS или просмотреть файлы логов.
   - Найти и проанализировать записи о выполненных действиях, определить, кто, когда и какие операции выполнял.
6. Оформление отчёта
   - Описать выполненные шаги, привести примеры записей из журнала аудита, сделать выводы о возможностях аудита для обеспечения безопасности данных.

Форма представления результата:

Отчет по выполненной лабораторной работе.

Критерии оценки:

Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.

Оценка «неудовлетворительно» ставится, если задание не выполнено.

Тема 2.7. Протоколирование событий, возникающих в процессе работы баз данных

Лабораторное занятие № 61
Включение и настройка логирования операций в MongoDB с использованием параметра profilingLevel

Цель: освоить методы включения и настройки логирования операций в MongoDB с помощью параметра profilingLevel, научиться анализировать профилировочные логи для оптимизации производительности и диагностики работы базы данных.

Выполнив работу, вы будете уметь:

У 2.1.2 Принимать решения по локализации проблем, связанных с нормальным функционированием базы данных;

У 2.1.3 Документировать внештатные ситуации связанные с нормальным функционированием базы данных;

У 2.3.1 Дать независимую оценку уровня безопасности

У.2.3.3 Разрабатывать перечень рекомендаций по дальнейшей эксплуатации БД с максимальной защитой хранящейся информации.

Материальное обеспечение:
MongoDB

Задание:

1. Изучить назначение и возможные значения параметра profilingLevel в MongoDB.
2. Включить и настроить профилирование операций на сервере или для отдельной базы данных.
3. Сгенерировать различные типы запросов к базе данных, чтобы они были отражены в профилировочных логах.
4. Проанализировать содержимое системной коллекции system.profile и сделать выводы о производительности запросов.
5. Оформить отчёт с описанием выполненных шагов и примерами профилировочных записей.

Порядок выполнения работы:

1. Изучение теории
   - Ознакомиться с назначением профилирования в MongoDB и возможными значениями параметра profilingLevel (0 — выключено, 1 — логируются медленные операции, 2 — логируются все операции).
2. Включение профилирования
   - Подключиться к MongoDB через командную строку или оболочку mongo.
   - Включить профилирование для всего сервера или для конкретной базы данных с помощью команды:

   ```
   db.setProfilingLevel(1, { slowms: 100 })
   ```

   (где 1 — уровень профилирования, 100 — порог медленности в миллисекундах).
3. Генерация событий
   - Выполнить различные операции с данными: простые и сложные запросы, вставку, обновление, удаление документов.
4. Анализ логов
   - Просмотреть содержимое коллекции system.profile:

   ```
   db.system.profile.find().pretty()
   ```

   - Проанализировать структуру записей, выделить медленные запросы, оценить их влияние на производительность.
5. Отключение профилирования
   - После завершения анализа вернуть параметр profilingLevel в значение 0:

   ```
   db.setProfilingLevel(0)
   ```
6. Оформление отчёта
   - Описать выполненные действия, привести примеры записей из коллекции system.profile, сделать выводы о работе механизма профилирования и его значимости для администрирования и оптимизации MongoDB.

Форма представления результата:
Отчет по выполненной лабораторной работе.

Критерии оценки:
Оценка «отлично» ставится, если задание выполнено верно.

Оценка «хорошо» ставится, если ход выполнения задания верный, но была допущена одна или две ошибки, приведшие к неправильному результату.

Оценка «удовлетворительно» ставится, если приведено неполное выполнение задания.
Оценка «неудовлетворительно» ставится, если задание не выполнено.