\ Санкт-Петербургское государственное бюджетное

профессиональное образовательное учреждение

«Ижорский колледж»

КУРСОВАЯ РАБОТА

ТЕМА: «Онлайн аукцион»

по ПМ.11 МДК.11.01 Технология разработки и защиты баз данных

Выполнил

обучающийся группы 232c

Алхасов Абдулнетиф Азикович

Проверил

преподаватель спец. дисциплин

специальности 09.02.07

Информационные системы и программирование

Венедиктов Д.В.

Санкт-Петербург, 2025ЗАДАНИЕ ДЛЯ ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ

1. Изучить теоретические основы технологии разработки баз данных, включая этапы проектирования, нормализации, реализации и тестирования.

2. Рассмотреть методы и средства защиты баз данных от несанкционированного доступа, атак и вредоносных программ.

3. Провести анализ существующих систем управления базами данных (СУБД) с точки зрения их функциональности, безопасности и удобства использования.

4. Разработать проект базы данных для конкретной предметной области, учитывая требования к безопасности и конфиденциальности данных.

5. Реализовать разработанный проект в выбранной СУБД.

6. Протестировать разработанную базу данных на предмет соответствия требованиям безопасности и надёжности.

7. Оценить эффективность разработанной системы защиты базы данных и предложить рекомендации по её улучшению.

8. Оформить результаты работы в виде курсовой работы, включающей введение, основную часть, заключение, список использованных источников и приложения.

В основной части курсовой работы необходимо подробно рассмотреть следующие вопросы:

* основные этапы разработки базы данных;
* методы обеспечения целостности и безопасности данных;
* механизмы аутентификации и авторизации пользователей;
* шифрование данных и защита от вредоносного ПО;
* особенности реализации проекта базы данных в выбранной СУБД;
* результаты тестирования и оценка эффективности системы защиты.

Для выполнения задания рекомендуется использовать современные научные источники, а также практический опыт разработки и защиты баз данных.

ПЛАН-ГРАФИК ВЫПОЛНЕНИЯ КУРСОВОЙ РАБОТЫ

|  |  |  |
| --- | --- | --- |
| Дата | Содержание работ | Отметка о выполнении |
| 10.09 | Выбор темы курсовой работы |  |
| 17.09 | Знакомство с методическими рекомендациями по выполнению курсовой работы |  |
| 24.09 | Знакомство с планом-графиком выполнения курсовой работы |  |
| 08.10 | Работа с нормативно-правовыми документами, учебной литературой |  |
| 22.10 | Анализ предметной области. Постановка задачи |  |
| 29.10 | Составление введения к курсовой работе |  |
| 05.11 | Описание предметной области и функции решаемых задач |  |
| 12.11 | Выбор средств для выполнения курсовой работы. Выбор среды разработки |  |
| 19.11 | Концептуально-логическое проектирование. Составление ER-диаграммы |  |
| 10.12 | Создание и заполнение базы данных |  |
| 24.12 | Представления в базе данных |  |
| 21.01 | Процедуры в базе данных |  |
| 04.02 | Создание ролей в базе данных |  |
| 18.02 | Аутентификация пользователей в базе данных |  |
| 25.02 | Импорт и экспорт базы данных |  |
| 04.03 | Разработка стратегии резервного копирования базы данных |  |
| 11.03 | Тестирование базы данных |  |
| 18.03 | Оптимизация базы данных |  |
| 25.03 | Составление заключения к курсовой работе |  |
| 01.04 | Составление списка источников информации, используемых при выполнении курсовой работы |  |
| 08.04 | Подготовка курсовой работы к защите |  |

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 7](#_Toc180594792)

[ГЛАВА 1. ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ БАЗЫ ДАННЫХ 8](#_Toc180594793)

[1.1. Анализ предметной области 8](#_Toc180594794)

[1.2. Анализ технологий для разработки базы данных 8](#_Toc180594795)

[1.3. Постановка задачи курсовой работы 10](#_Toc180594796)

[1.3.1. Цели курсовой работы 10](#_Toc180594797)

[1.3.2. Задачи курсовой работы 11](#_Toc180594798)

[1.4.1. Функциональные требования к системе 11](#_Toc180594799)

[1.4.2. Нефункциональные требования к системе 11](#_Toc180594800)

[1.5. Выбор программных средств для разработки базы данных 11](#_Toc180594801)

[1.6. Выбор среды для разработки базы данных 14](#_Toc180594802)

[ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ 17](#_Toc180594803)

[2.1. Концептуально-логическое моделирование 17](#_Toc180594804)

[2.1.1 Концептуальное моделирование 17](#_Toc180594805)

[2.1.2. Логическое моделирование 18](#_Toc180594806)

[2.2. Описание информационных объектов базы данных 20](#_Toc180594807)

[2.3.1. Создание базы данных 25](#_Toc180594808)

[2.3.2. Заполнение базы данных 25](#_Toc180594809)

[2.4. Представления в базе данных 27](#_Toc180594810)

[2.5. Процедуры в базе данных 30](#_Toc180594811)

[2.6. Создание ролей в базе данных 30](#_Toc180594812)

[2.7. Аутентификация пользователей в базе данных 30](#_Toc180594813)

[2.8. Импорт и экспорт базы данных 30](#_Toc180594814)

[2.9. Разработка стратегии резервного копирования базы данных 30](#_Toc180594815)

[2.10.1. Тестирование базы данных 31](#_Toc180594816)

[2.10.2. Оптимизация базы данных 31](#_Toc180594817)

[ЗАКЛЮЧЕНИЕ 32](#_Toc180594818)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 33](#_Toc180594819)

# ВВЕДЕНИЕ

В современном цифровом мире данные представляют собой один из ключевых ресурсов, влияющих на стратегические решения и операционные процессы организаций. Базы данных стали основным инструментом для хранения, обработки и управления информацией, обеспечивая эффективное взаимодействие с данными. Разработка баз данных – это сложный процесс, требующий комплексного подхода, включающего анализ требований, проектирование, реализацию и поддержку.

С учетом растущего объема информации и усложнения бизнес-процессов, значимость качественного проектирования баз данных трудно переоценить. Правильная структура базы данных обеспечивает не только ее функциональность и производительность, но и легкость в управлении данными, что в свою очередь влияет на оперативность принятия решений.

Однако с увеличением объема данных и многообразием технологий, связанных с их хранением, возникает необходимость уделять особое внимание защите данных. Утечки информации и кибератаки могут привести к значительным финансовым потерям и подрыву доверия со стороны клиентов. Поэтому технологии защиты баз данных становятся критически важными для бизнеса всех размеров.

Эта работа сосредоточится на ключевых аспектах разработки и защиты баз данных, включая методы проектирования и элементы безопасности, которые должны быть внедрены для обеспечения надежности и защиты данных в любой организации. Изучение этих тем поможет определить лучшие практики, которые должны применяться для достижения устойчивости и эффективности работы с информацией в условиях динамично меняющейся технологической среды.

# ГЛАВА 1. ФУНКЦИОНАЛЬНЫЕ ВОЗМОЖНОСТИ БАЗЫ ДАННЫХ

## 1.1. Анализ предметной области

Технология разработки и защиты баз данных охватывает широкий спектр процессов, методов и инструментов, необходимых для создания, управления и обеспечения безопасности баз данных. Это область, которая активно развивается в ответ на растущие требования к хранению и защите информации, что делает её актуальной для различных отраслей, таких как финансы, здравоохранение, образование и многие другие.

## 1.2. Анализ технологий для разработки базы данных

Разработка баз данных включает в себя использование различных технологий и инструментов, которые помогают осуществлять проектирование, реализацию, управление и оптимизацию баз данных. Рассмотрим основные технологии и подходы, которые широко применяются в этой сфере.

**Системы управления базами данных (СУБД)**

СУБД являются основными инструментами для создания и управления базами данных. Они могут быть разделены на несколько категорий:

* реляционные СУБД (RDBMS): MySQL, PostgreSQL, Oracle, Microsoft SQL Server;
* нереляционные СУБД (NoSQL): MongoDB, Cassandra, Redis;
* гибридные СУБД: Microsoft Azure Cosmos DB.

**Языки программирования**

* SQL (Structured Query Language): основной язык для работы с реляционными базами данных. Позволяет создавать, изменять, запрашивать и управлять данными;
* Python и JavaScript: часто используются для взаимодействия с базами данных в веб-приложениях. Библиотеки, такие как SQLAlchemy или Sequelize, упрощают работу с базами данных;
* Java и C#: используются в крупных корпоративных приложениях, особенно с реляционными базами данных.

**Инструменты для проектирования баз данных**

* ER-моделирование: Инструменты, такие как Lucidchart, Draw.io или ER/Studio, позволяют создавать диаграммы сущностей и взаимосвязей (ER-диаграммы), которые помогают визуализировать структуру базы данных;
* системы управления версиями: Git и другие системы позволяют контролировать изменения в схемах баз данных и процессах миграции.

**Инструменты для обеспечения безопасности баз данных**

* шифрование: использование технологий шифрования данных (например, AES) для защиты информации как «в покое», так и «в движении»;
* аутентификация и авторизация: применение OAuth, OpenID Connect и других протоколов для управления доступом к базе данных;
* мониторинг и аудит: инструменты, такие как Splunk, могут использоваться для отслеживания запросов и анализа событий безопасности.

**Методы оптимизации производительности**

* индексация: создание индексов для ускорения выполнения запросов, особенно в реляционных базах данных.
* кэширование: использование кэшей, таких как Redis или Memcached, для ускорения доступа к часто запрашиваемым данным.
* балансировка нагрузки: распределение запросов между несколькими серверами баз данных для повышения производительности и отказоустойчивости.

Выбор технологий для разработки базы данных зависит от специфики проекта, объема данных, ожидаемых нагрузок и требований к безопасности. Комплексный подход к выбору инструментов и методов способствует созданию эффективных и надежных систем управления данными, которые могут адаптироваться к изменяющимся условиям и требованиям.

## 1.3. Постановка задачи курсовой работы

С ростом объемов данных, а также увеличением угроз безопасности информации, эффективная разработка и защита баз данных становятся крайне важными задачами для организаций. Безопасность данных имеет критическое значение для решения вопросов конфиденциальности, целостности и доступности информации.

Успешное выполнение работы будет способствовать более глубокому пониманию актуальных проблем и новых подходов в области безопасного управления данными.

### 1.3.1. Цели курсовой работы

Основная цель заключается в создании эффективной и безопасной базы данных, способной обеспечить надежный доступ к информации и защиту от потери или несанкционированного доступа.

### 1.3.2. Задачи курсовой работы

* проектирование архитектуры баз данных;
* разработка схемы данных и моделей;
* обеспечение безопасности данных через шифрование, аутентификацию и авторизацию;
* реализация механизмов резервного копирования и восстановления;
* мониторинг и аудит доступа к данным.

## 1.4.1. Функциональные требования к системе

* возможность создания, обновления и удаления данных;
* поддержка сложных запросов и отчетов;
* обеспечение многопользовательского доступа.

## 1.4.2. Нефункциональные требования к системе

* высокая производительность и надежность;
* защита от несанкционированного доступа;
* удобный интерфейс для пользователей.

## 1.5. Выбор программных средств для разработки базы данных

Выбор программных средств для разработки баз данных является ключевым этапом, который влияет на функциональность, производительность и безопасность системы.

При выполнении данной курсовой работы я планирую использовать следующие программные средства:

* система управления базами данных – MySQL;
* языки программирования – Java, SQL;
* средства проектирования базы данных – *ER-моделирование:* Lucidchart, *ER-моделирование:* Liquibase;

инструменты для обеспечения безопасности данных – *Шифрование:*Vormetric,

*Аудит и мониторинг:* Splunk;

средства тестирования и оптимизации приложений – *Тестирование:* JMeter,

*Оптимизация:* EXPLAIN.

Выбор программных средств для разработки базы данных зависит от специфических требований проекта, таких как объем данных, тип доступа, требования к безопасности и производительности. Важно учитывать будущую масштабируемость и поддерживаемость решений, чтобы создать надежную и эффективную систему.

## 1.6. Выбор среды для разработки базы данных

Выбор подходящей среды для разработки базы данных является критически важным этапом проекта, который может оказать значительное влияние на эффективность работы команды и качество конечного продукта.

При выполнении данной курсовой работы я планирую использовать следующие среды и инструменты:

* IDE и редактор кода – pgAdmin;
* система контейнеризации – Docker;
* система управления версиями – git;

фреймворк и библиотека – Hibernate.

Выбор среды для разработки базы данных требует комплексного подхода с учетом специфики проекта, требований к производительности, безопасности и интеграции. Правильный выбор инструментария и технологии существенно влияет на успех проекта, поэтому следует тщательно проанализировать свои потребности и доступные решения.

# ГЛАВА 2. ПРОЕКТИРОВАНИЕ БАЗЫ ДАННЫХ

## 2.1. Концептуально-логическое моделирование

Концептуально-логическое моделирование – это важный этап в проектировании базы данных, который позволяет структурировать данные и определить их взаимосвязи. Этот процесс разделяется на два ключевых этапа: концептуальное моделирование и логическое моделирование.

### 2.1.1 Концептуальное моделирование

Целью концептуального моделирования является определение основных сущностей и их связей в предметной области, не вдаваясь в детали реализации.

**Идентификация сущностей**

Сущности: Товар, Аукционер, Пользователь, Ставки;

**Определение атрибутов**

Атрибуты: Товар: ProductID, AuctionerID, Название, Описание, Начальная ставка.

Пользователь: UserID, Имя, Фамилия, **Электронная почта, Пароль,** Дата регистрации**.**

Аукционер: AuctionerID, Баланс, Рейтинг.

Ставки: BidID, UserID, ProductID, Размер ставки, Время ставок.

**Установление связей**

Один пользователь может делать ставки на множество товаров.

Один аукционер может выставлять много товаров.

Каждый товар принадлежит только одному аукционеру.

**Создание ER-диаграммы**

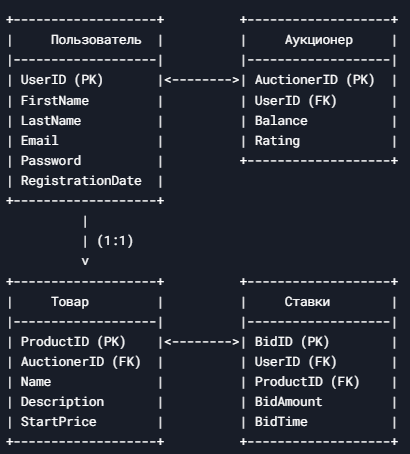


Рисунок 1 – ER-диаграмма разрабатываемой базы данных

### 2.1.2. Логическое моделирование

Целью логического моделирования является перевод концептуальной модели в логическую, с уточнением структуры базы данных и обязательств для хранения данных.

**Определение первичных ключе**

Сущность “Пользователь” перечный ключ: UserID.

Сущность “ Товар” перечный ключ: ProductID

Сущность “ Аукционер” перечный ключ: AuctionerID

Сущность “ Ставки” перечный ключ: BidID

**Формализация связей**

Многие-ко-многим между «Пользователь» и «Товар». Один на множество между «Пользователь» и «Аукционер». Один на множество между «Товар» и «Ставкой». Один на множество между «Товар» и «Аукционер» Многие-ко-многим между «Пользователь» и «Ставкой».

**Определение типов данных**

Товар: «ProductID» - числа, «AuctionerID» - числа, «Название» - символьный, «Описание» - символьный, «Начальная ставка» - числа. Пользователь: «UserID» - числа , «Имя» - символьный, «Фамилия» - символьный, «Электронная почта» - символьный, «Пароль» - символьный, «Дата регистрации» - числа. Аукционер: «AuctionerID» - числа, «Баланс» - числа, «Рейтинг» - числа. Ставки: «BidID» - числа, «UserID» - числа, «ProductID» - числа, «Размер ставки» - числа, «Время» - числа.

**Создание логической схемы базы данных**

#### 1. Таблица **Пользователь (User)**

* **UserID** (INT, PRIMARY KEY) – уникальный идентификатор пользователя.
* **Имя** (VARCHAR(50)) – имя пользователя.
* **Фамилия** (VARCHAR(50)) – фамилия пользователя.
* **Электронная почта** (VARCHAR(100)) – электронная почта пользователя.
* **Пароль** (VARCHAR(255)) – пароль пользователя.
* **Дата регистрации** (DATE) – дата регистрации пользователя.

#### 2. Таблица **Аукционер (Auctioner)**

* **AuctionerID** (INT, PRIMARY KEY) – уникальный идентификатор аукционера.
* **Баланс** (DECIMAL(10, 2)) – баланс аукционера.
* **Рейтинг** (INT) – рейтинг аукционера.

#### 3. Таблица **Товар (Product)**

* **ProductID** (INT, PRIMARY KEY) – уникальный идентификатор товара.
* **AuctionerID** (INT, FOREIGN KEY REFERENCES Auctioner(AuctionerID)) – идентификатор аукционера, который выставил товар.
* **Название** (VARCHAR(100)) – название товара.
* **Описание** (TEXT) – описание товара.
* **Начальная ставка** (DECIMAL(10, 2)) – начальная ставка на товар.

#### 4. Таблица **Ставки (Bids)**

* **BidID** (INT, PRIMARY KEY) – уникальный идентификатор ставки.
* **UserID** (INT, FOREIGN KEY REFERENCES User(UserID)) – идентификатор пользователя, сделавшего ставку.
* **ProductID** (INT, FOREIGN KEY REFERENCES Product(ProductID)) – идентификатор товара, на который сделана ставка.
* **Размер ставки** (DECIMAL(10, 2)) – размер ставки.
* **Время ставки** (DATETIME) – время, когда была сделана ставка.

Концептуально-логическое моделирование является важным этапом разработки базы данных, который позволяет углубиться в детализацию структуры данных и обеспечить целостность и согласованность информации. Правильное моделирование позволяет избежать проблем на следующих этапах разработки и эксплуатации базы данных.

## 2.2. Описание информационных объектов базы данных

Информационные объекты базы данных представляют собой основные строительные блоки, на которых базируется структура любой базы данных. Они включают в себя различные сущности и их атрибуты, которые позволяют организовать, хранить и управлять данными.

Таблица 1 – информационные объекты разрабатываемой базы данных

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Объект | Атрибуты | Связи | Ограничения | Тип данных |
| Пользователь | UserID | 1:N с "Ставки" | UserID – PRIMARY KEY | INT |
| Имя | VARCHAR(25) |
| Фамилия | VARCHAR(25) |
| Электронная почта | VARCHAR(50) |
| Пароль | VARCHAR(25) |
| Дата регистрации | INT |
| Товар | ProductID | 1:N с "Ставки" | ProductID – PRIMARY KEY | INT |
| Название | VARCHAR(50) |
| Описание | N:1 с "Аукционер" | Начальная\_ставка > 0 | VARCHAR(50) |
| Начальная\_ставка | VARCHAR(50) |
| AuctionerID | DECIMAL(10,2) |

Описание информационных объектов базы данных позволяет структурировать и организовать информацию, обеспечивая эффективное хранение и доступ к данным. Правильное определение сущностей, их атрибутов и связей является ключом к успешному проектированию базы данных и ее функциональной полноте.

## 2.3.1. Создание базы данных

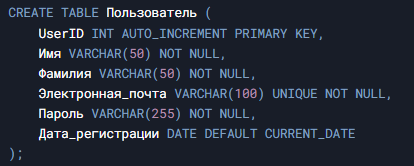


Рисунок 2 – созданная база данных

## 2.3.2. Заполнение базы данных

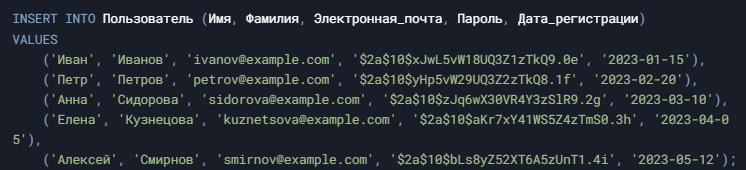


Рисунок 3 – добавление объекта в базу данных

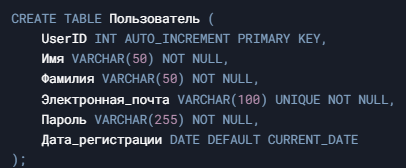


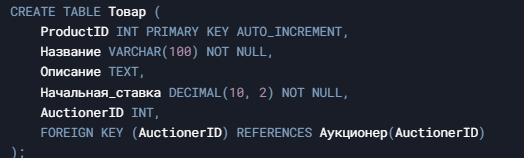
Рисунок 4 – добавление атрибутов объекту в базе данных



Рисунок 5 – установление связей между объектами в базе данных



Рисунок 6 – выбор типа данных для объекта в базе данных



## 2.4. Представления в базе данных

Представление в базе данных – это виртуальная таблица, которая формируется на основе результата выполнения SQL-запроса. Она не содержит данных, а лишь определяет, как данные будут отображаться пользователям.

Функции представлений:

* упрощение сложных запросов – позволяют объединять данные из нескольких таблиц и показывать только нужные поля, что делает работу с данными более удобной;
* безопасность – можно ограничить доступ пользователей к определённым столбцам или строкам баз данных, предоставляя доступ только к представлениям;
* логическая независимость – изменения в базах данных не влияют на внешний интерфейс, так как представления могут скрывать изменения структуры таблиц;
* кодовая реиспользуемость – позволяют сократить код, избавляя от необходимости повторно писать одни и те же запросы.

Представление создается с помощью команды CREATE VIEW.

|  |
| --- |
| CREATE TABLE Product (  ProductID INT PRIMARY KEY AUTO\_INCREMENT,  Название VARCHAR(100) NOT NULL,  Описание TEXT,  Начальная\_ставка DECIMAL(10, 2) NOT NULL,  AuctionerID INT NOT NULL,  FOREIGN KEY (AuctionerID) REFERENCES Auctioner(AuctionerID)  ); |

После создания представления его можно использовать так же, как и таблицу:

|  |
| --- |
| SELECT \* FROM ProductView  WHERE Начальная\_ставка > 1000; |

Представления могут быть обновляемыми или не обновляемыми. Обновляемые представления позволяют выполнять операции INSERT, UPDATE и DELETE, при этом изменения будут применяться к базовой таблице.

|  |
| --- |
| UPDATE Product  SET Начальная\_ставка = 1600.00  WHERE ProductID = 1; |

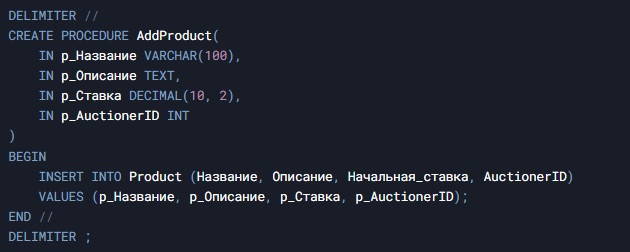
Представления можно изменять с помощью команды CREATE OR REPLACE VIEW или удалять с помощью DROP VIEW

|  |
| --- |
| CREATE OR REPLACE VIEW ProductView AS  SELECT Название, Описание, Начальная\_ставка  FROM Product  WHERE Начальная\_ставка > 1000; |

|  |
| --- |
| DELETE FROM ProductView  WHERE Начальная\_ставка < 1000; |

Представления в базах данных являются мощным инструментом для работы с данными. Они обеспечивают большую гибкость, безопасность и легкость в использовании, способствуя более эффективной организации и обработке информации. Понимание работы с представлениями – важный аспект для разработчиков и администраторов баз данных.

## 2.5. Процедуры в базе данных



## 2.6. Создание ролей в базе данных



## 2.7. Аутентификация пользователей в базе данных



## 2.8. Импорт и экспорт базы данных

**Экспорт базы данных** позволяет создать резервную копию данных или переместить их на другой сервер или в другую среду разработки.

**Чтобы экспортировать базу данных, нужно выполнить следующие шаги**:

1. **Иметь доступ к серверу на базе Linux**, на котором работает MySQL или MariaDB.
2. **Знать название базы данных** и иметь доступ (логин и пароль) к аккаунту, который имеет, по крайней мере, доступ read only (только для чтения).
3. **Ввести команду**:

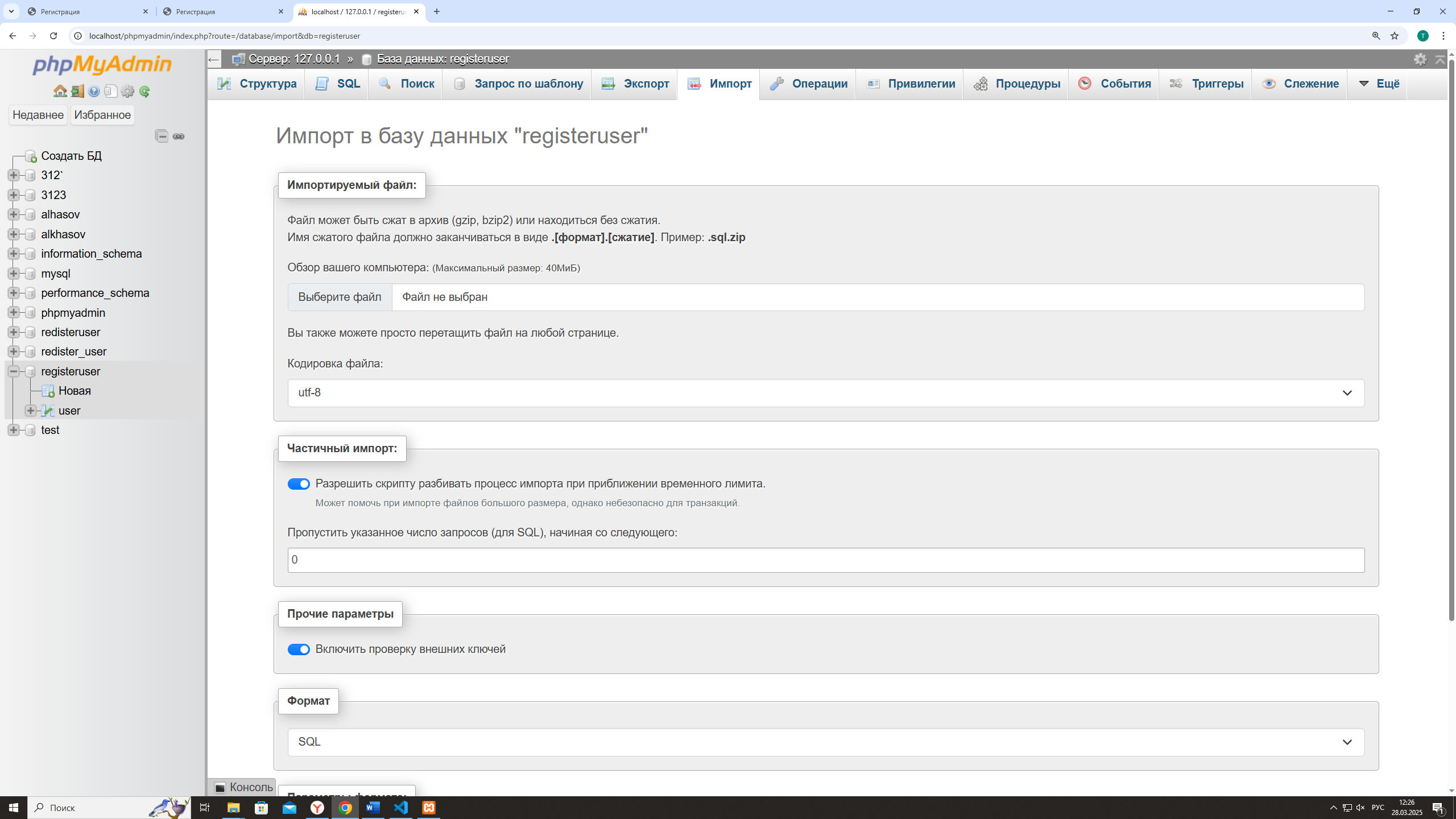
mysqldump -u имя\_пользователя -p название\_БД > data-dump.sql



**Импорт базы данных** — это процесс загрузки данных из предварительно созданного файла резервной копии или другого источника данных в базу данных.

**способ — через phpMyAdmin**:

1. В списке слева выбрать нужную базу данных.
2. Перейти во вкладку «Импорт».
3. Выбрать файл для импорта на компьютере и проверить настройки. Скорее всего, они подойдут для импортирования файла, но при желании их можно изменить.
4. Нажать кнопку «Вперёд» — и будет выполнен импорт файла.

****

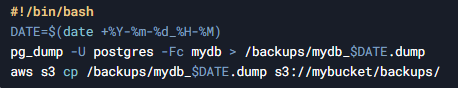
## 2.9. Разработка стратегии резервного копирования базы данных

## **Цели резервного копирования**

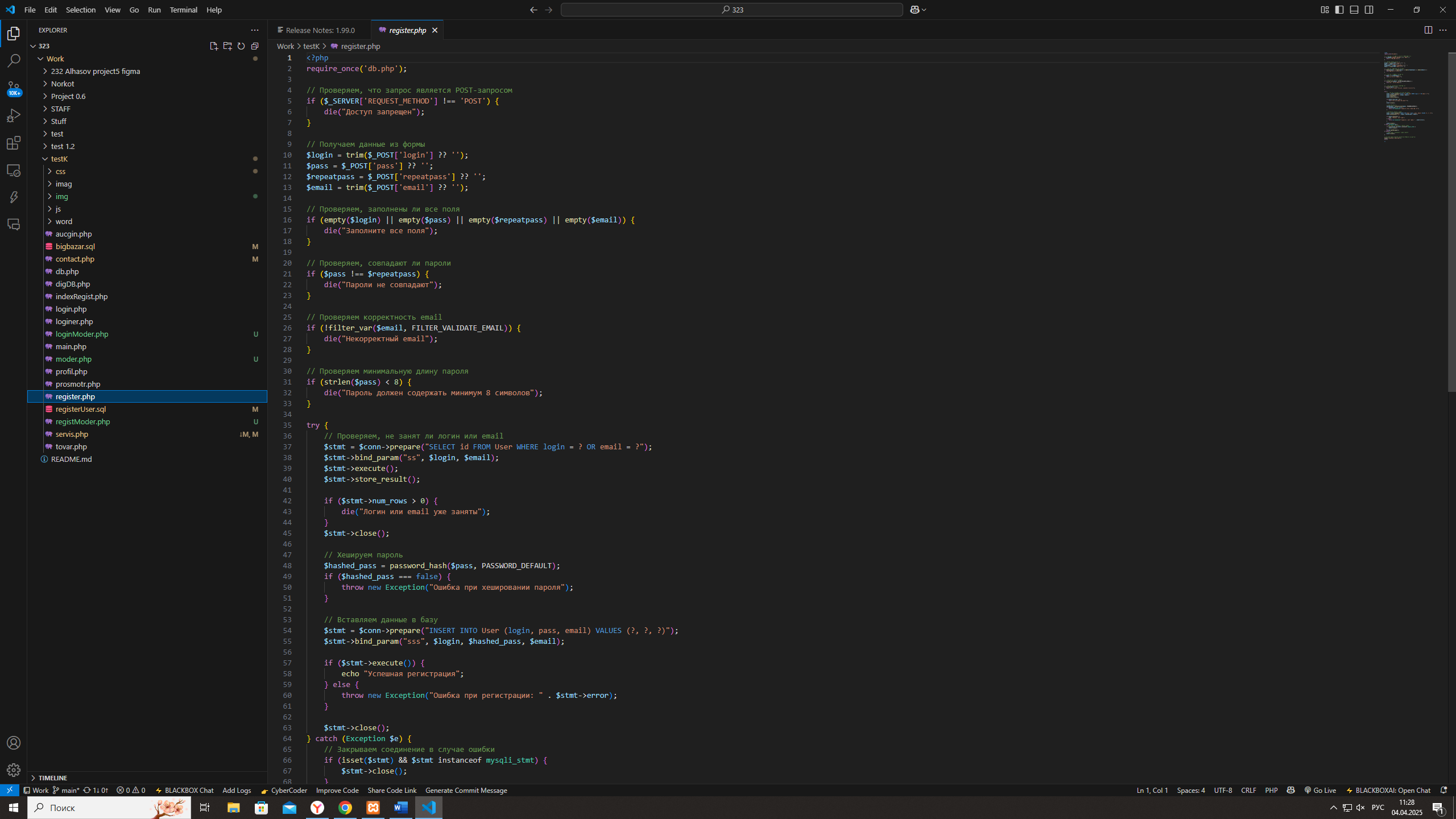
* **Минимизация потерь данных** (RPO — Recovery Point Objective).
* **Быстрое восстановление** (RTO — Recovery Time Objective).
* **Защита от**:
  + Аппаратных сбоев.
  + Ошибок администрирования (например, DROP TABLE).
  + Вредоносных атак (ransomware, SQL-инъекции).
  + Стихийных бедствий (если данные хранятся в одном дата-центре).

## **Частота и срок хранения**

* **Частота**: зависит от динамики данных (например, финансовая система — ежечасные бэкапы, блог — раз в день).
* **Срок хранения**:
  + 7 последних дней — ежедневные бэкапы.
  + 4 последние недели — еженедельные.
  + 12 месяцев — ежемесячные.
* **Правило 3-2-1**:
  + **3** копии данных.
  + **2** разных типа носителей (SSD + лента/облако).
  + **1** копия в другом месте.



## 2.10.1. Тестирование базы данных

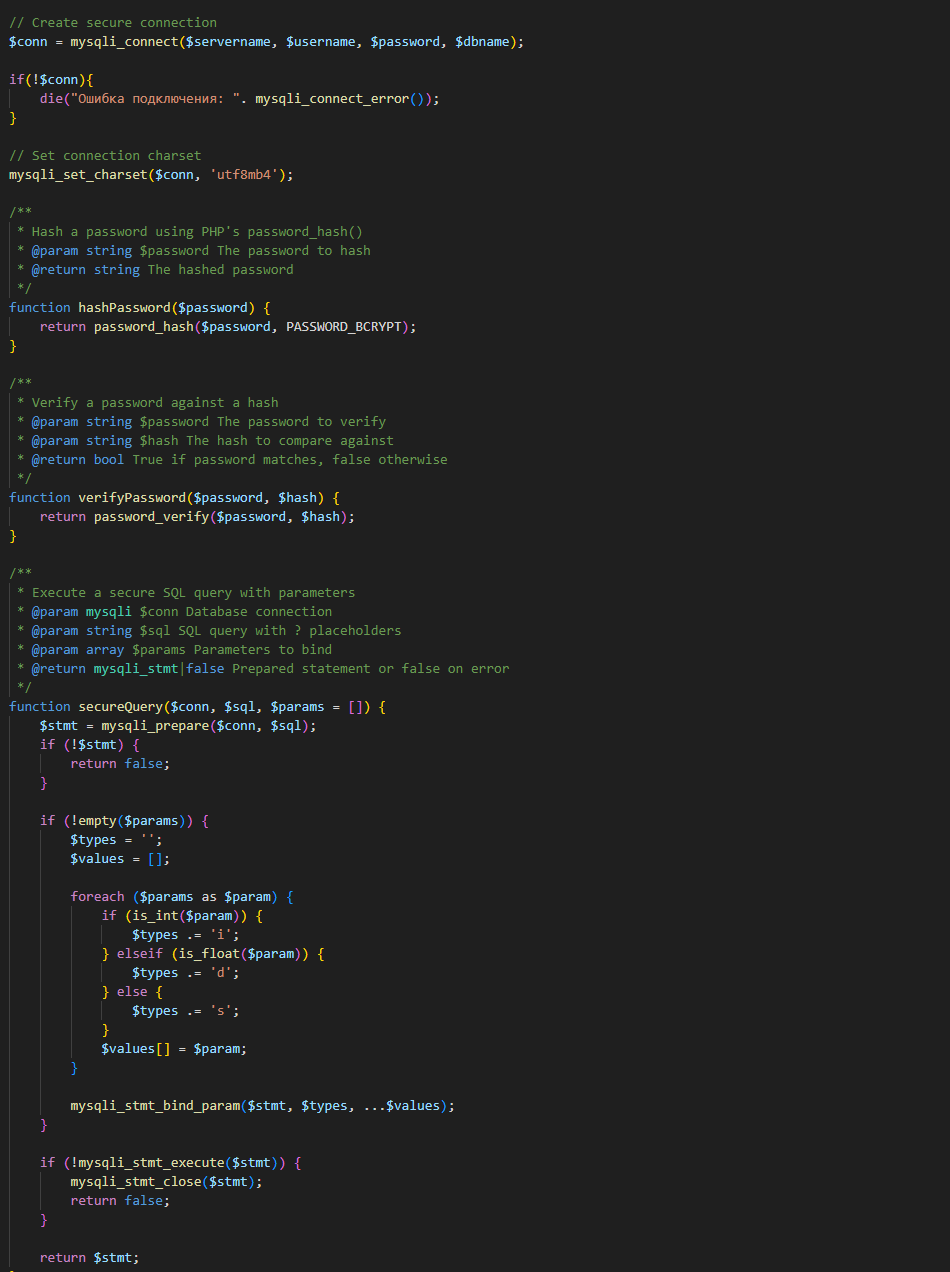
* **Регистрация и аутентификация пользователей**
  + Корректность валидации данных (email, пароль).
  + Восстановление пароля. 
  + 

#### **Нагрузочное тестирование**

* Проверка работы системы при большом количестве пользователей.
* Анализ времени отклика при размещении ставок.
* Тестирование конкурентных ставок .

#### **Тестирование безопасности**

* Защита от SQL-инъекций.
* Проверка авторизации .
* Защита от DDoS-атак .



## 2.10.2. Оптимизация базы данных

Оптимизация базы данных (БД) — это процесс улучшения её производительности, уменьшения времени отклика и эффективного использования ресурсов. Оптимизация необходима для обеспечения быстрой работы приложений, особенно при больших объемах данных и высокой нагрузке.

**Основные методы оптимизации**

#### **1. Оптимизация структуры базы данных**

* **Нормализация** – устранение избыточности данных за счет разбиения таблиц и установления связей между ними.
* **Денормализация** – в некоторых случаях (например, для сложных запросов) допустимо дублирование данных для ускорения выборки.
* **Выбор оптимальных типов данных** – использование подходящих типов полей (INT, VARCHAR, TEXT и т. д.) для экономии памяти и ускорения обработки.

#### **2. Оптимизация запросов**

* **Использование индексов** – создание индексов для часто запрашиваемых столбцов ускоряет поиск, но замедляет вставку и обновление.
* **Оптимизация SQL-запросов** – избегание сложных подзапросов, использование JOIN вместо вложенных SELECT, ограничение выборки (LIMIT).
* **Анализ выполнения запросов** – использование EXPLAIN для выявления узких мест в запросах.

#### **3. Кэширование**

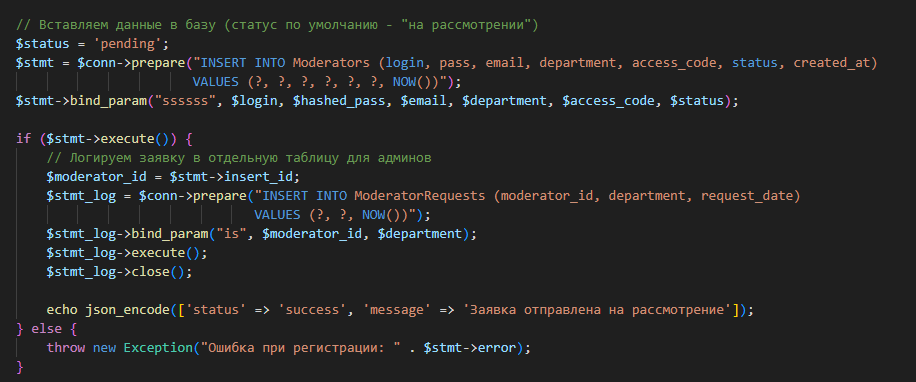
* **Кэширование запросов** – сохранение результатов часто выполняемых запросов в памяти (Redis, Memcached).
* **Кэширование на уровне приложения** – хранение готовых данных для уменьшения нагрузки на БД.

#### **4. Масштабирование**

* **Вертикальное масштабирование** – увеличение мощности сервера (CPU, RAM, SSD).
* **Горизонтальное масштабирование** – распределение нагрузки между несколькими серверами (шардинг, репликация).

#### **5. Обслуживание базы данных**

* **Регулярное обновление статистики** – для правильного планирования запросов.
* **Очистка и дефрагментация** – удаление неиспользуемых данных, перестроение индексов.
* **Резервное копирование** – предотвращение потери данных.



# ЗАКЛЮЧЕНИЕ

В ходе изучения технологии разработки и защиты баз данных было выявлено, что успешная работа с данными в современных условиях требует комплексного подхода, включающего как аспекты проектирования, так и обеспечения безопасности. Принимая во внимание стремительное развитие информационных технологий, применение современных методов и инструментов становится неизбежным. Это позволяет не только оптимизировать процессы хранения и обработки данных, но и защитить их от возможных угроз.

Особое внимание следует уделить вопросам шифрования данных, аутентификации пользователей и резервного копирования. Эффективная защита баз данных требует активного мониторинга и регулярного обновления систем безопасности.

Таким образом, подходы к разработке и защите баз данных должны быть гибкими и адаптивными, чтобы соответствовать быстро меняющимся требованиям и угрозам. В будущем важно продолжать исследовать и внедрять новые технологии, что позволит обеспечить надежность и безопасность информации на всех уровнях.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

1. Волик, М.В. Разработка базы данных в Access / М.В. Волик. – Издательство «Прометей», 2021. – 88 с.
2. Даккет, Дж. PHP и MySQL. Серверная веб-разработка / Дж. Даккет. – Издательство «Эксмо», 2023. – 688 с.
3. Дьяков, И.А. Базы данных. Язык SQL : учебное пособие / И.А. Дьяков. – Тамбов : Тамбовский государственный технический университет (ТГТУ), 2022. – 82 с.
4. Ла Рокка, М. Продвинутые и структуры данных. / М. ла Рокка. – Издательство «Питер», 2024. – 848 с.
5. Никсон, Р. Создаём динамические веб-сайты с помощью PHP, MySQL, JavaScript, CSS и HTML5. 6-е изд. / Р. Никсон. – Издательство «Питер»: Санкт-Петербург, 2023. – 832 с.
6. https://www.figma.com/ (дата обращения: 15.03.2025).
7. https://www.github.com/ (дата обращения: 16.03.2025).

ПРИЛОЖЕНИЯ

