

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ  
Московский авиационный институт  
(национальный исследовательский университет)

Институт № 8  
Компьютерные науки и прикладная математика

Кафедра 806 «Вычислительная математика и программирование»

КУРСОВАЯ РАБОТА  
по дисциплине «Базы данных»

Тема:  
«Проектирование базы данных системы дистанционного обучения»

**Выполнил:** студент группы М8О-311Б-20

---

Комиссаров Антон Сергеевич

---

(Фамилия, имя, отчество)

**Преподаватель:**

Чумакова Екатерина Витальевна

---

(Фамилия, имя, отчество)

---

(подпись)

Оценка:

Дата:

Москва, 2022

# Содержание

<b>ВВЕДЕНИЕ</b>	<b>2</b>
<b>1 Анализ предметной области и постановка задачи</b>	<b>3</b>
1.1 Описание предметной области и назначение системы . . . . .	3
1.2 Цели разработки и способы достижения . . . . .	4
1.3 Ожидаемые результаты . . . . .	5
1.4 Информационный образ системы . . . . .	5
1.5 Входные и выходные потоки . . . . .	5
1.6 Функциональный образ системы . . . . .	6
1.7 Требования и ограничения на использование системы . . . . .	7
1.8 Взаимодействие с другими программами . . . . .	8
<b>2 Концептуальное и логическое проектирование базы данных</b>	<b>9</b>
2.1 Выделение информационных объектов и определение атрибутов объектов	9
2.2 Разработка логической структуры базы данных . . . . .	11
2.3 Построение диаграммы сущность-связь . . . . .	13
2.3.1 Раскрытие связей многие-ко-многим . . . . .	13
2.3.2 Поддержка ограничений целостности, списки доменов атрибутов	14
2.3.3 Нормализация объектной модели . . . . .	15
2.4 Выбор используемой СУБД . . . . .	15
<b>3 Физическое проектирование базы данных</b>	<b>16</b>
3.1 Генерация физической модели БД . . . . .	16
3.2 SQL-скрипт для создания базы данных и таблиц . . . . .	17
3.3 Генерация физической схемы БД . . . . .	21
3.4 Описание запросов и процедур к БД . . . . .	22
3.4.1 Создания основных объектов базы данных на языке SQL в син- таксисе выбранной СУБД . . . . .	22
3.4.2 Создания вспомогательных объектов базы данных . . . . .	24
<b>4 Разработка прикладной программы</b>	<b>29</b>
<b>ЗАКЛЮЧЕНИЕ</b>	<b>32</b>
<b>СПИСОК ЛИТЕРАТУРЫ</b>	<b>33</b>

# ВВЕДЕНИЕ

Современное общество всё чаще называют «информационным». Это связано с тем, что сегодня в любой среде человеческой деятельности одной из главных задач является организация хранения и обработки большого количества информации. В этом существенную помощь могут оказать компьютерные системы обработки данных. Основная цель подобных систем – повышение эффективности работы отдельной фирмы, предприятия или организации.

Многие существующие экономические, информационно-справочные, банковские программные комплексы используют для взаимодействия с информацией базы данных. База данных (БД) – это электронный архив, в котором специальным образом размещаются и форматируются данные. Для решения проблем обработки информации используются специальные инструментальные средства систем управления базами данных. Система управления базами данных (СУБД) – это программа, позволяющая сформировать базу данных, вносить в неё изменения и дополнения, производить поиск требуемых данных по запросам, обрабатывать хранящиеся данные, выводить данные на экран и на печать.

## 1 Анализ предметной области и постановка задачи

### 1.1 Описание предметной области и назначение системы

Разрабатываемая подсистема носит название "БД системы дистанционного обучения (СДО)" и предназначена для:

- автоматизации обработки информации о пользователях, курсах, темах и заданиях;
- обеспечения возможности оперативного поиска заданий и тестов, а также их результатов по выбранной теме;
- обеспечения возможности оперативного получения информации о существующих курсах и содержащихся в них темах.

База данных создаётся для информационного обслуживания сотрудников и учеников дистанционной системы обучения для ВУЗа. Данная система позволяет опера-

тивно создавать новые базовые элементы системы (курс, тема и т.д.). База данных должна содержать данные о пользователях (сотрудниках и учениках), курсах, темах, заданиях и тестах, а также результатах их выполнения.

В соответствии с предметной областью система строится с учётом следующих особенностей:

- каждый ученик (студент) учится в определённой группе, в каждой группе учатся несколько студентов;
- каждый пользователь может быть участником нескольких курсов;
- каждый студент может использовать несколько попыток для ответа на вопрос;
- каждая тема может содержать в себе занятия (материалы) для изучения и тестовые вопросы;
- для каждого вопроса есть банк содержимого вопросов.

## 1.2 Цели разработки и способы достижения

*Цели разработки:*

- обеспечить минимальные затраты времени для получения информации;
- упростить поиск нужной информации о тестах и их результатах;
- обеспечить оперативность изменения информации.

*Способы достижения:*

Указанные цели разработки могут быть достигнуты следующими способами:

- проанализировать документы, которые имеют отношение к решаемой задаче для выявления значимых объектов (сущностей) и атрибутов сущностей – данных, которые должны храниться в БД;
- описать модель данных в удобном для разработчиков способе (в виде ER–диаграммы);
- представить информацию в связанных таблицах;
- разработать запросы и процедуры для взаимодействия с темами, тестами и их результатами.

### 1.3 Ожидаемые результаты

В результате применения разрабатываемой системы ожидается:

- упрощение процессов изложения материала и его проверки преподавателями;
- быстрое и эффективное формирование необходимой информации (1 мин., вместо 10 при ручном поиске);
- выбор наиболее оптимальной информации для конкретного пользователя;
- увеличение количества обслуженных пользователей.

### 1.4 Информационный образ системы

Для создания ER-модели необходимо выделить сущности предметной области:

№ п/п	Наименование	Определение
1	Course	Курс – описание курса и его предназначение
2	Theme	Тема – название темы
3	Lesson	Занятие – учебный материал для изучения (текст, ссылка)
4	Question	Вопрос – название вопроса
5	Task	Конкретное задание – содержимое вопроса: текст вопроса и ответ на него
6	Result	Результат ответа на вопрос – информация об отвечающем и его вводе
7	Student	Студент – персональные данные об ученике
8	StudentGroup	Учебная группа студента – информация о группе: шифр, направление, факультет, курс
9	Worker	Сотрудник – персональные данные о работнике
10	WorkerRole	Должность сотрудника – описание должности работника

### 1.5 Входные и выходные потоки

*Входные:*

Данные о сотруднике:

Почта	Пароль	ФИО	Должность	Сфера деятельности

Данные о должности:

Наименование должности	Описание должности

Данные о студенте:

Почта	Пароль	ФИО	Учебная группа

Данные об учебной группе:

Наименование группы	Учебный курс	Факультет	Направление

*Выходные:*

- перечень курсов;
- перечень тем для каждого курса;
- информация о занятиях;
- информация о вопросах;
- информация о сотрудниках;
- информация об учениках;
- результаты ответов на вопросы.

## 1.6 Функциональный образ системы

Система предназначена для решения следующих задач:

1. Представить перечень курсов;
2. Представить перечень тем для каждого курса;

3. Выдать информацию о занятиях;
4. Выдать информацию о вопросах;
5. Выдать информацию о сотрудниках;
6. Выдать информацию об учениках;
7. Продемонстрировать результаты ответов на вопросы.

## 1.7 Требования и ограничения на использование системы

### *Пользователи системы:*

Данная система является вспомогательной системой и поэтому может использоваться широким кругом пользователей:

- ученик (студент);
- преподаватель;
- менеджер;
- администратор;
- другие сотрудники ВУЗа.

Также система может входить в состав других различных систем.

Администратор имеет полный доступ к системе, т.е. он может вносить изменения во все таблицы. Все остальные пользователи могут:

#### 1. Ученик:

- просмотр списка пользователей;
- просмотр перечня курсов;
- просмотр перечня тем каждого курса;
- просмотр учебных материалов;
- просмотр вопросов;
- просмотр и добавление результата ответа на вопрос.

## 2. Преподаватель:

- просмотр списка пользователей;
- просмотр перечня курсов;
- просмотр перечня тем каждого курса;
- просмотр учебных материалов;
- просмотр вопросов;
- добавление курсов;
- добавление, изменение и удаление тем;
- добавление, изменение и удаление занятий;
- добавление, изменение и удаление вопросов;
- просмотр результатов ответов на вопросы.

## 3. Менеджер:

- просмотр списка пользователей;
- просмотр перечня курсов;
- просмотр перечня тем каждого курса;
- просмотр учебных материалов;
- просмотр вопросов;
- просмотр и добавление пользователей;
- просмотр и добавление учебных групп.

## 1.8 Взаимодействие с другими программами

Разрабатываемая база данных является составной частью информационной системы **Distance Learning System (DLS)**, программное обеспечение которой предоставляет пользовательский интерфейс доступа к данным БД и их частичной модификации. Информационная система может взаимодействовать с другими системами обучения ВУЗа.



## 2 Концептуальное и логическое проектирование базы данных

### 2.1 Выделение информационных объектов и определение атрибутов объектов

Для каждой сущности из п. 1.4. определим первичный ключ (служащий для идентификации записей) отношения и перечень атрибутов, имеющих значение в предметной области задачи. Для каждого отношения указаны атрибуты с их внутренним названием, типом и длиной:

#### Course

Поле	Тип данных	Назначение	Примечание
CourseID	int	Идентификатор курса	Первичный ключ
CourseName	varchar(255)	Название	Обязательное поле
CourseSubject	varchar(100)	Учебный предмет	
CourseNumber	int	Номер учебного курса	

#### Theme

Поле	Тип данных	Назначение	Примечание
ThemeID	int	Идентификатор темы	Первичный ключ
ThemeName	varchar(100)	Название	Обязательное поле
CourseID	int	Идентификатор курса	Обязательное поле, внешний ключ
Pos	int	Позиция в курсе	Обязательное поле

#### Lesson

Поле	Тип данных	Назначение	Примечание
LessonID	bigint	Идентификатор занятия	Первичный ключ
LessonName	varchar(100)	Название	
RawText	text	Текст занятия	
URL	varchar(1024)	Ссылка на материал	
ThemeID	int	Идентификатор темы	Обязательное поле, внешний ключ
Pos	int	Позиция в теме	Обязательное поле

## Question

Поле	Тип данных	Назначение	Примечание
QuestionID	bigint	Идентификатор вопроса	Первичный ключ
QuestionName	varchar(100)	Название	
ThemeID	int	Идентификатор темы	Обязательное поле, внешний ключ
Pos	int	Позиция в теме	Обязательное поле

## Task

Поле	Тип данных	Назначение	Примечание
TaskID	bigint	Идентификатор задания	Первичный ключ
TaskText	text	Текст задания	Обязательное поле
Answer	tinytext	Ответ	Обязательное поле
QuestionID	bigint	Идентификатор вопроса	Обязательное поле, внешний ключ

## Result

Поле	Тип данных	Назначение	Примечание
TaskID	bigint	Идентификатор задания	Составной первичный ключ, внешний ключ
PersonID	bigint	Идентификатор ученика	Составной первичный ключ, внешний ключ
ResultDate	timestamp	Дата и время	Составной первичный ключ
Answer	tinytext	Введённый ответ	Обязательное поле
Truth	bool	Правильность	Обязательное поле

## StudentGroup

Поле	Тип данных	Назначение	Примечание
GroupID	int	Идентификатор группы	Первичный ключ
GroupName	varchar(15)	Шифр группы	Обязательное поле
TrainingCourse	int	Учебный курс	
Faculty	int	Факультет	
Direction	varchar(150)	Направление подготовки	

## Student

Поле	Тип данных	Назначение	Примечание
PersonID	bigint	Идентифкатор ученика	Первичный ключ
PersonName	varchar(100)	ФИО	Обязательное поле
Mail	varchar(100)	Почта	Обязательное поле
Pass	varchar(64)	Хэш пароля	Обязательное поле
RegDate	date	Дата регистрации	
LastLog	timestamp	Дата и время последнего входа	
GroupID	int	Идентификатор группы	Внешний ключ

## WorkerRole

Поле	Тип данных	Назначение	Примечание
RoleID	int	Идентифкатор должности	Первичный ключ
RoleName	varchar(50)	Название	Обязательное поле
RoleDescription	varchar(255)	Описание	

## WorkerRole

Поле	Тип данных	Назначение	Примечание
PersonID	bigint	Идентифкатор сотрудника	Первичный ключ
PersonName	varchar(100)	ФИО	Обязательное поле
Mail	varchar(100)	Почта	Обязательное поле
Pass	varchar(64)	Хэш пароля	Обязательное поле
RegDate	date	Дата регистрации	
LastLog	timestamp	Дата и время последнего входа	
RoleID	int	Идентификатор должности	Обязательное поле, внешний ключ
WorkDescript	varchar(255)	Сфера деятельности	

## 2.2 Разработка логической структуры базы данных

Для построения ER-модели на основе выделенных сущностей, необходимо определить типы связей между ними. MySQL Workbench поддерживает три типа связей (или отношений): один-ко-многим, один-к-одному, многие-ко-многим.

Связь типа один-ко-многим – межтабличное отношение, при котором любая запись в первой таблице может быть связана несколькими записями во второй, но в то же время любая запись второй таблицы связана только с одной записью в первой.

Например, связь между таблицами «StudentGroup» и «Student» в базе данных «СДО» – связь типа один-ко-многим, так как к учебной группе могут быть прикреплены несколько студентов, но студент может числиться только в одной группе.

Если при этом связующее поле в одной из таблиц является ключевым, то такая таблица называется главной (таблица «Курсы»). Вторая таблица, участвующая в связи, называется подчиненной (таблица «Темы»). При этом связующее поле подчинённой таблицы обычно называют внешним (или чужим) ключом.

Связь многие-ко-многим установлена между таблицами Student - Course, а также между таблицами Student - Worker. Все остальные связи в базе данных «СДО» - это связи типа один-ко-многим. В результате была получена ER-модель, приведённая на рисунке 1.

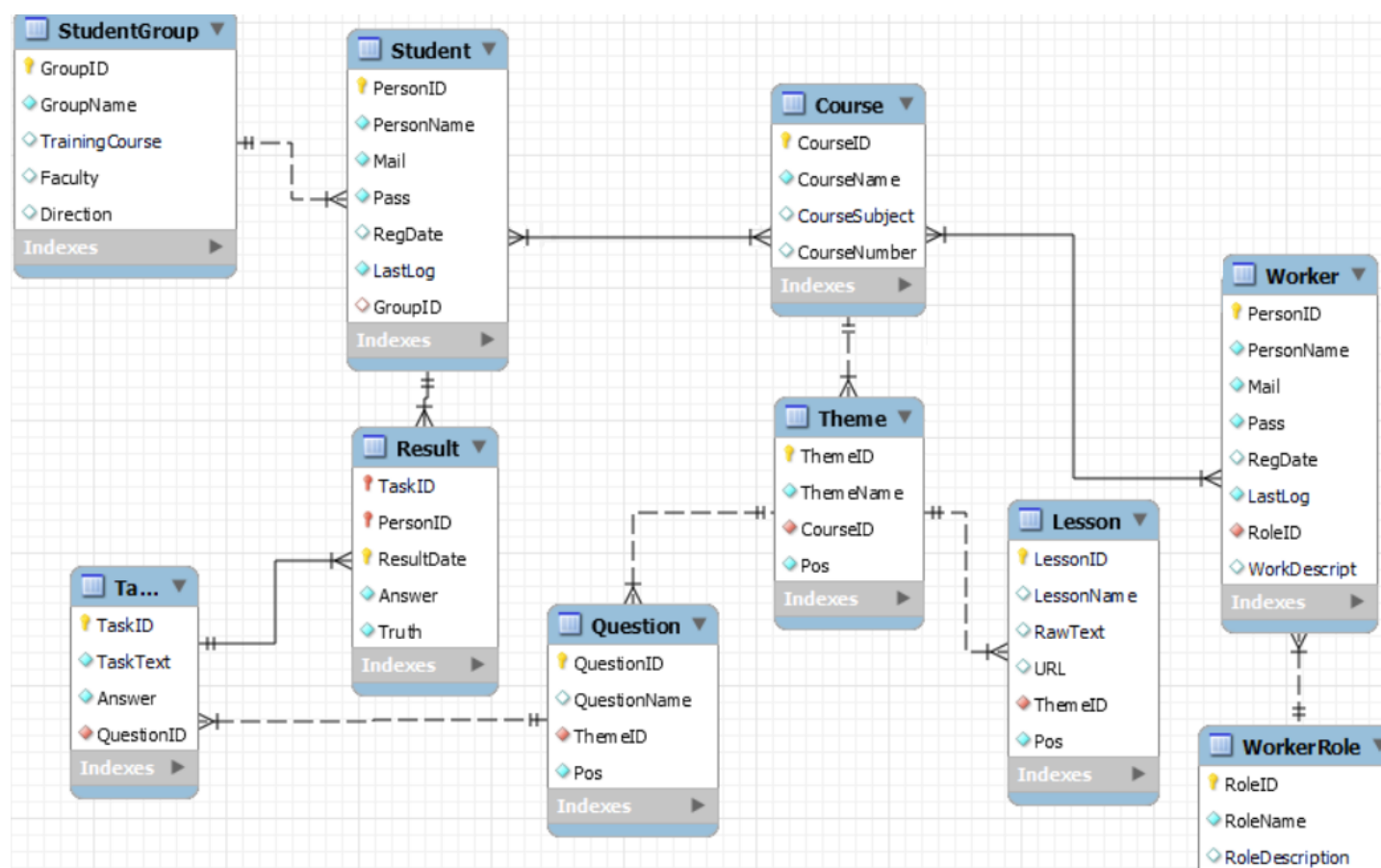


Рис. 1: Логическая структура БД

## 2.3 Построение диаграммы сущность-связь

### 2.3.1 Раскрытие связей многие-ко-многим

В полученной ER-модели имеется две связи многие-ко-многим, которые не поддерживаются ни одной из реляционных СУБД. Подобные связи преобразуются (раскрываются) через промежуточную таблицу. Таким образом, в модель будут добавлены две сущности, каждая из которых является подчинённой по отношению к таблицам, объединённым связью многие-ко-многим (т.е. связывается с этими таблицами связью многие-к-одному). При этом каждое из отношений будет иметь составной первичный ключ из двух внешних ключей главных таблиц.

Для связи Course и Student в виде таблицы:

#### CourseStudent

Поле	Тип данных	Назначение	Примечание
CourseID	int	Идентифкатор курса (FK)	Составной первичный ключ
PersonID	bigint	Идентифкатор ученика (FK)	Составной первичный ключ
LastLog	timestamp	Последний доступ к курсу	

Отношение, описывающее курсы, на которые записан каждый ученик.

Для связи Course и Worker в виде таблицы:

#### CourseWorker

Поле	Тип данных	Назначение	Примечание
CourseID	int	Идентифкатор курса (FK)	Составной первичный ключ
PersonID	bigint	Идентифкатор сотрудника (FK)	Составной первичный ключ
LastLog	timestamp	Последний доступ к курсу	

Отношение, описывающее курсы, к которым имеет непосредственный доступ каждый сотрудник.

В обоих случаях связь многие-ко-многим обусловлена тем, что у одного курса может быть несколько пользователей, а у одного курса может быть несколько курсов.

### 2.3.2 Поддержка ограничений целостности, списки доменов атрибутов

Для сохранения целостности данных определим следующие ограничения целостности:

- обязательные данные;
- ограничения для доменов атрибутов;
- целостность сущностей;
- ссылочная целостность.

При определении атрибутов сущностей были указаны как обязательные, так и необязательные для заполнения данные, т.е. эти атрибуты могут иметь пустые (NULL) и непустые (NOT NULL) значения соответственно.

В качестве дополнительных ограничений целостности зададим домены некоторым атрибутам. Домен представляет набор допустимых значений для одного или нескольких атрибутов. Определим следующие домены:

- атрибут *CourseNumber* – по сути представляет собой год обучения в ВУЗе и может изменяться от 1 до 6, поэтому его домен состоит из 6 целых значений;
- атрибут *Pos* – позиция элемента темы внутри неё, которая может принимать целые значения от 0 до  $n - 1$  включительно, где  $n$  – количество элементов в теме: домен состоит из  $n$  целых значений.

Данные ограничения устанавливаются при определении доменов атрибутов, присутствующих в модели данных.

Первичный ключ любой сущности не может содержать пустого значения, для этого в свойствах атрибута в MySQL Workbench зададим обязательность (NOT NULL). Внешний ключ связывает каждую строку дочернего отношения с той строкой родительского отношения, которая содержит это же значение соответствующего первичного ключа. Понятие ссылочной целостности означает, что если внешний ключ содержит некоторое значение, то оно обязательно должно присутствовать в первичном ключе одной из строк родительского отношения. Для обеспечения целостности данных в свойствах связей установим значение cascade на изменения delete и update для каскадного обновления связанных полей.

### 2.3.3 Нормализация объектной модели

Хранение информации в реляционной БД, предполагает такую структуру, в которой данные хранятся в оптимальном объеме. Для этого предусмотрен механизм нормализации. Проверим на соответствие как минимум 3-й нормальной форме полученных отношений диаграммы.

Все таблицы в данной системе обладают следующими свойствами:

1. Каждый элемент таблицы представляет 1 элемент данных (атомарность).
2. Все столбцы таблиц однородные.
3. Отсутствуют повторяющиеся атрибуты (соответствие 1НФ).
4. Отсутствуют атрибуты, зависящие только от части уникального идентификатора (2НФ).
5. Отсутствуют атрибуты, зависящие от атрибутов, не входящих в уникальный идентификатор (3НФ), т.е. все отношения находится в 3-й нормальной форме, т.к. находятся во 2НФ и каждый не ключевой атрибут нетранзитивно зависит от первичного ключа.

## 2.4 Выбор используемой СУБД

Анализ информационных задач показывает, что для реализации требуемых функций подходят почти все СУБД (MS Access, MySQL, MS SQL Server и др.). Все они поддерживают реляционную модель данных и предоставляют разнообразные обширные возможности для работы с данными.

В MySQL Workbench есть возможность сгенерировать физическую базу данных на основе разработанной логической модели.

Учитывая вышесказанное, для выполнения курсовой работы остановим свой выбор на СУБД MySQL. MySQL – это система управления реляционными базами данных, отличающаяся отличной масштабируемостью, гибкостью, простотой в использовании, а также скоростью работы. Область применения: хранение больших объемов данных, хранение высокоценных данных или данных, требующих соблюдения режима секретности.

## 3 Физическое проектирование базы данных

### 3.1 Генерация физической модели БД

В результате проведенной разработки получили ER-диаграмму физической модели БД, которая приведена на рисунке 2.

При создании физической базы данных:

- каждая простая сущность превращается в таблицу, имя сущности становится именем таблицы;
- каждый атрибут становится возможным столбцом с тем же именем; может выбираться более точный формат; столбцы, соответствующие необязательным атрибутам, могут содержать неопределенные значения;
- компоненты уникального идентификатора сущности превращаются в первичный ключ таблицы.

В итоге формируется sql-скрипт, содержащий запросы на создание базы данных и таблиц с определенными связями и ограничениями целостности.



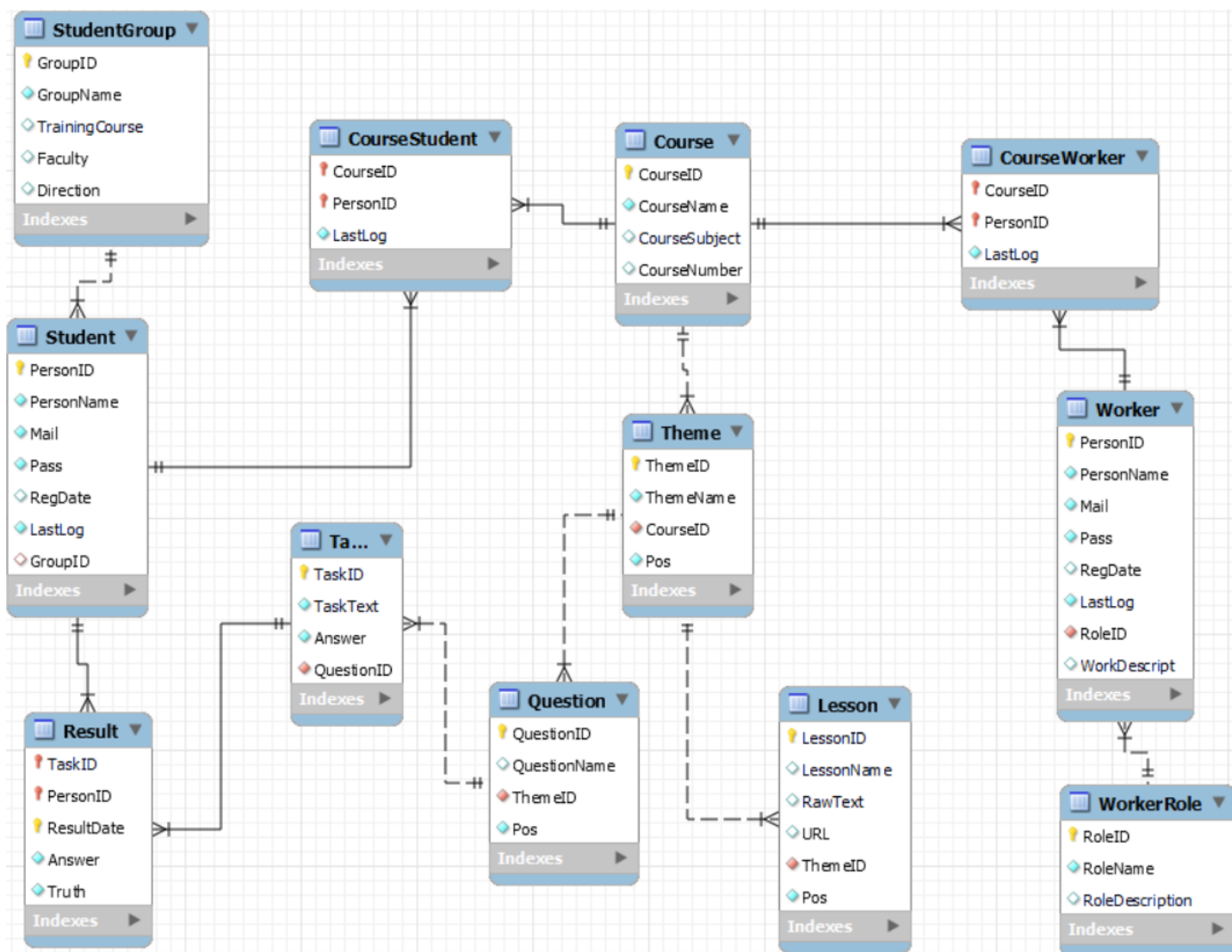


Рис. 2: Физическая модель БД

### 3.2 SQL-скрипт для создания базы данных и таблиц

```

1 SET @OLD_UNIQUE_CHECKS=@@UNIQUE_CHECKS, UNIQUE_CHECKS=0;
2 SET @OLD_FOREIGN_KEY_CHECKS=@@FOREIGN_KEY_CHECKS, FOREIGN_KEY_CHECKS=0;
3 SET @OLD_SQL_MODE=@@SQL_MODE,
  ↪ SQL_MODE='ONLY_FULL_GROUP_BY,STRICT_TRANS_TABLES,NO_ZERO_IN_DATE,NO_ZERO_DATE,ERROR_FOR_
4
5 CREATE SCHEMA IF NOT EXISTS `DLS` ;
6 USE `DLS` ;
7
8 CREATE TABLE IF NOT EXISTS `DLS`.`Course` (
9 `CourseID` INT NULL DEFAULT NULL AUTO_INCREMENT,
10 `CourseName` VARCHAR(255) NOT NULL,
  
```

```

11 `CourseSubject` VARCHAR(100) NULL DEFAULT NULL,
12 `CourseNumber` INT NULL DEFAULT NULL,
13 PRIMARY KEY (`CourseID`));
14
15 CREATE TABLE IF NOT EXISTS `DLS`.`StudentGroup` (
16 `GroupID` INT NULL DEFAULT NULL AUTO_INCREMENT,
17 `GroupName` VARCHAR(15) NOT NULL,
18 `TrainingCourse` INT NULL DEFAULT NULL,
19 `Faculty` INT NULL DEFAULT NULL,
20 `Direction` VARCHAR(150) NULL DEFAULT NULL,
21 PRIMARY KEY (`GroupID`));
22
23 CREATE TABLE IF NOT EXISTS `DLS`.`Student` (
24 `PersonID` BIGINT NULL DEFAULT NULL AUTO_INCREMENT,
25 `PersonName` VARCHAR(100) NOT NULL,
26 `Mail` VARCHAR(100) NOT NULL,
27 `Pass` VARCHAR(64) NOT NULL,
28 `RegDate` DATE NULL DEFAULT current_date,
29 `LastLog` TIMESTAMP NOT NULL DEFAULT current_timestamp,
30 `GroupID` INT NULL DEFAULT NULL,
31 PRIMARY KEY (`PersonID`),
32 INDEX (`GroupID` ASC) VISIBLE,
33 CONSTRAINT ``
34 FOREIGN KEY (`GroupID`)
35 REFERENCES `DLS`.`StudentGroup` (`GroupID`)
36 ON DELETE cascade
37 ON UPDATE cascade);
38
39 CREATE TABLE IF NOT EXISTS `DLS`.`WorkerRole` (
40 `RoleID` INT NOT NULL AUTO_INCREMENT,
41 `RoleName` VARCHAR(50) NOT NULL,
42 `RoleDescription` VARCHAR(255) NULL DEFAULT NULL,
43 PRIMARY KEY (`RoleID`));
44
45 CREATE TABLE IF NOT EXISTS `DLS`.`Worker` (
46 `PersonID` BIGINT NULL DEFAULT NULL AUTO_INCREMENT,
47 `PersonName` VARCHAR(100) NOT NULL,
48 `Mail` VARCHAR(100) NOT NULL,
49 `Pass` VARCHAR(64) NOT NULL,
50 `RegDate` DATE NULL DEFAULT current_date,

```

```

51 `LastLog` TIMESTAMP NOT NULL DEFAULT current_timestamp,
52 `RoleID` INT NOT NULL,
53 `WorkDescript` VARCHAR(255) NULL DEFAULT NULL,
54 PRIMARY KEY (`PersonID`),
55 INDEX (`RoleID` ASC) VISIBLE,
56 CONSTRAINT ``
57 FOREIGN KEY (`RoleID`)
58 REFERENCES `DLS`.`WorkerRole` (`RoleID`)
59 ON UPDATE cascade);
60
61 CREATE TABLE IF NOT EXISTS `DLS`.`CourseStudent` (
62 `CourseID` INT NOT NULL,
63 `PersonID` BIGINT NOT NULL,
64 `LastLog` TIMESTAMP NOT NULL DEFAULT current_timestamp,
65 PRIMARY KEY (`CourseID`, `PersonID`),
66 INDEX (`CourseID` ASC) VISIBLE,
67 INDEX (`PersonID` ASC) VISIBLE,
68 CONSTRAINT ``
69 FOREIGN KEY (`CourseID`)
70 REFERENCES `DLS`.`Course` (`CourseID`)
71 ON DELETE cascade,
72 CONSTRAINT ``
73 FOREIGN KEY (`PersonID`)
74 REFERENCES `DLS`.`Student` (`PersonID`)
75 ON DELETE cascade);
76
77 CREATE TABLE IF NOT EXISTS `DLS`.`CourseWorker` (
78 `CourseID` INT NOT NULL,
79 `PersonID` BIGINT NOT NULL,
80 `LastLog` TIMESTAMP NOT NULL DEFAULT current_timestamp,
81 PRIMARY KEY (`CourseID`, `PersonID`),
82 INDEX (`CourseID` ASC) VISIBLE,
83 INDEX (`PersonID` ASC) VISIBLE,
84 CONSTRAINT ``
85 FOREIGN KEY (`CourseID`)
86 REFERENCES `DLS`.`Course` (`CourseID`)
87 ON DELETE cascade,
88 CONSTRAINT ``
89 FOREIGN KEY (`PersonID`)
90 REFERENCES `DLS`.`Worker` (`PersonID`)

```

```

91  ON DELETE cascade);
92
93  CREATE TABLE IF NOT EXISTS `DLS`.`Theme` (
94  `ThemeID` INT NULL DEFAULT NULL AUTO_INCREMENT,
95  `ThemeName` VARCHAR(100) NOT NULL,
96  `CourseID` INT NOT NULL,
97  `Pos` INT NOT NULL,
98  PRIMARY KEY (`ThemeID`),
99  INDEX (`CourseID` ASC) VISIBLE,
100 CONSTRAINT ``
101 FOREIGN KEY (`CourseID`)
102 REFERENCES `DLS`.`Course` (`CourseID`)
103 ON DELETE cascade);
104
105 CREATE TABLE IF NOT EXISTS `DLS`.`Lesson` (
106 `LessonID` BIGINT NULL DEFAULT NULL AUTO_INCREMENT,
107 `LessonName` VARCHAR(100) NULL DEFAULT NULL,
108 `RawText` TEXT NULL DEFAULT NULL,
109 `URL` VARCHAR(1024) NULL DEFAULT NULL,
110 `ThemeID` INT NOT NULL,
111 `Pos` INT NOT NULL,
112 PRIMARY KEY (`LessonID`),
113 INDEX (`ThemeID` ASC) VISIBLE,
114 CONSTRAINT ``
115 FOREIGN KEY (`ThemeID`)
116 REFERENCES `DLS`.`Theme` (`ThemeID`)
117 ON DELETE cascade);
118
119 CREATE TABLE IF NOT EXISTS `DLS`.`Question` (
120 `QuestionID` BIGINT NULL DEFAULT NULL AUTO_INCREMENT,
121 `QuestionName` VARCHAR(100) NULL DEFAULT NULL,
122 `ThemeID` INT NOT NULL,
123 `Pos` INT NOT NULL,
124 PRIMARY KEY (`QuestionID`),
125 INDEX (`ThemeID` ASC) VISIBLE,
126 CONSTRAINT ``
127 FOREIGN KEY (`ThemeID`)
128 REFERENCES `DLS`.`Theme` (`ThemeID`)
129 ON DELETE cascade);
130

```

```

131 CREATE TABLE IF NOT EXISTS `DLS`.`Task` (
132   `TaskID` BIGINT NULL DEFAULT NULL AUTO_INCREMENT,
133   `TaskText` TEXT NOT NULL,
134   `Answer` TINYTEXT NOT NULL,
135   `QuestionID` BIGINT NOT NULL,
136   PRIMARY KEY (`TaskID`),
137   INDEX (`QuestionID` ASC) VISIBLE,
138   CONSTRAINT ``
139   FOREIGN KEY (`QuestionID`)
140   REFERENCES `DLS`.`Question` (`QuestionID`)
141   ON DELETE cascade);
142
143 CREATE TABLE IF NOT EXISTS `DLS`.`Result` (
144   `TaskID` BIGINT NOT NULL,
145   `PersonID` BIGINT NOT NULL,
146   `ResultDate` TIMESTAMP NOT NULL DEFAULT current_timestamp,
147   `Answer` TINYTEXT NOT NULL,
148   `Truth` TINYINT NOT NULL,
149   PRIMARY KEY (`TaskID`, `PersonID`, `ResultDate`),
150   INDEX (`TaskID` ASC) VISIBLE,
151   INDEX (`PersonID` ASC) VISIBLE,
152   CONSTRAINT ``
153   FOREIGN KEY (`TaskID`)
154   REFERENCES `DLS`.`Task` (`TaskID`)
155   ON DELETE cascade,
156   CONSTRAINT ``
157   FOREIGN KEY (`PersonID`)
158   REFERENCES `DLS`.`Student` (`PersonID`)
159   ON DELETE cascade);
160
161 SET SQL_MODE=@OLD_SQL_MODE;
162 SET FOREIGN_KEY_CHECKS=@OLD_FOREIGN_KEY_CHECKS;
163 SET UNIQUE_CHECKS=@OLD_UNIQUE_CHECKS;

```

### 3.3 Генерация физической схемы БД

Результатом выполнения sql-скрита является физическая база данных, схема данных которой представлена на рисунке 3.

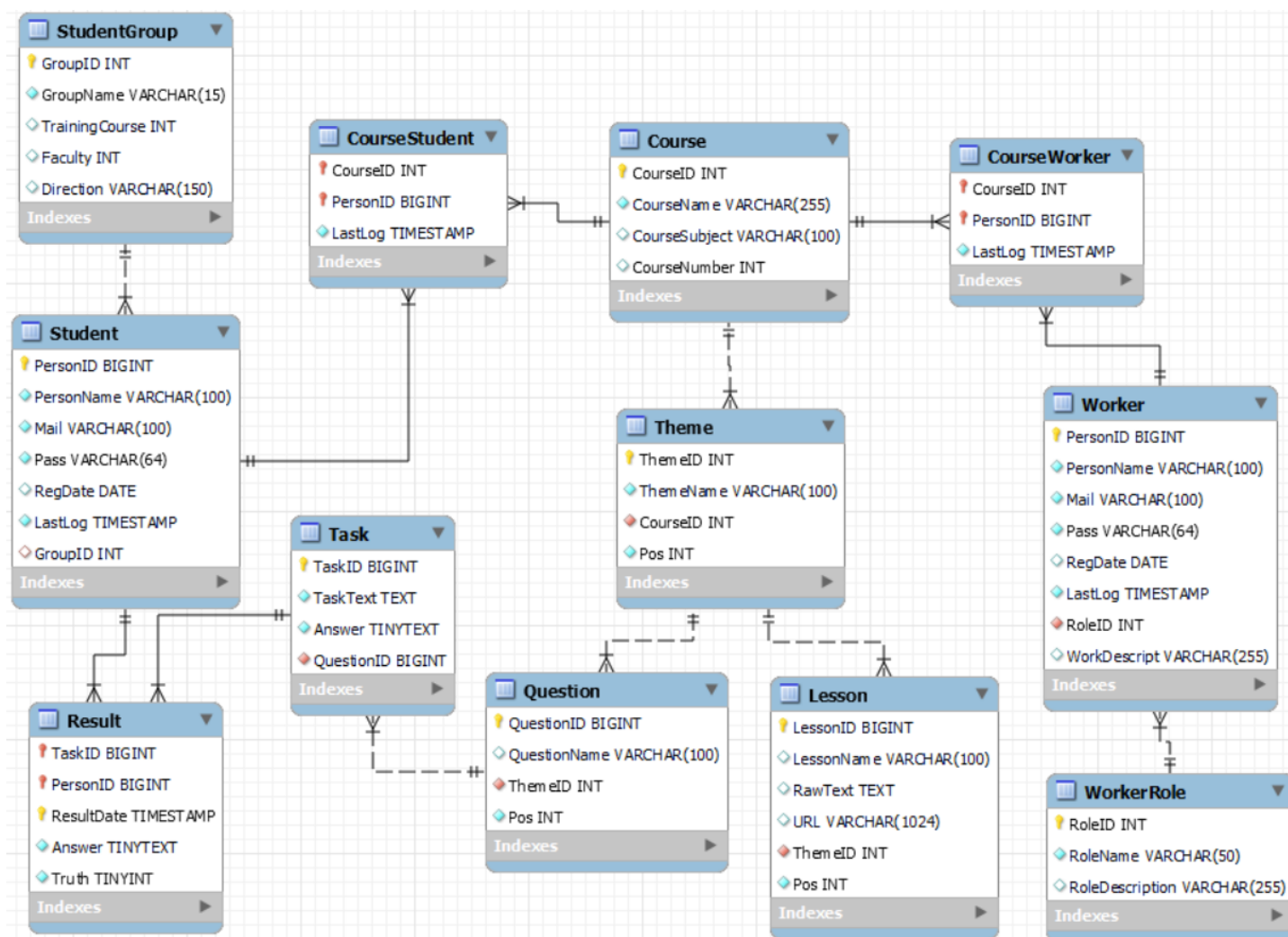


Рис. 3: Физическая схема БД

## 3.4 Описание запросов и процедур к БД

### 3.4.1 Создания основных объектов базы данных на языке SQL в синтаксисе выбранной СУБД

Для демонстрации приведем примеры реализации основных запросов, реализующих основные заявленные функции:

- добавление нового пользователя

```

1  insert Student(PersonName, Mail, Pass, GroupID)
2  values('Комиссаров Антон Сергеевич', 'tosha@mail.ru',
   ↪ '39045c3f9ea34ed30b0f1ee40a8b65b46a2a865e39b4d1eb7e032aaba11df951', 1);
3  insert Student(PersonName, Mail, Pass)
4  values('KAC', 'tosha3@mail.ru',
   ↪ '39045c3f9ea34ed30b0f1ee40a8b65b46a2a865e39b4d1eb7e032aaba11df951');
  
```

PersonID	PersonName	Mail	Pass	RegDate	LastLog	GroupID
1	Комиссаров Антон Серге...	tosha@mail.ru	3904...	2022-12-12	2022-12-13 20:25:00	1
2	КАС	tosha3@mai...	3904...	2022-12-12	2022-12-12 02:16:28	NULL

- процесс аутентификации

```

1  select PersonID, UserType from LoginTable where Mail = "tosha@mail.ru"
2  and Pass = "39045c3f9ea34ed30b0f1ee40a8b65b46a2a865e39b4d1eb7e032aaba11df951";

```

PersonID	UserType
1	0

- обновление даты и времени последнего доступа к системе

```

1  update Student set LastLog = current_timestamp() where PersonID = 1;

```

PersonID	PersonName	Mail	Pass	RegDate	LastLog	GroupID
1	Комиссаров Антон Серге...	tosha@mail.ru	3904...	2022-12-12	2022-12-14 09:38:59	1

- проверка подписки на конкретный курс

```

1  select exists(select * from CourseWorker where CourseID = 1 and PersonID = 1);

```

exists(select * from CourseWorker where CourseID = 1 and PersonID = 1)
0

- получение всех курсов, на которые подписан пользователь

```

1  select CourseWorker.CourseID, CourseName, CourseSubject, CourseNumber
2  from CoyrseWorker join Course on Course.CourseID = CourseWorker.CourseID
3  where PersonID = 1;

```

CourseID	CourseName	CourseSubject	CourseNumber
1	Базы данных	NULL	3

### 3.4.2 Создания вспомогательных объектов базы данных

Приведём примеры нескольких готовых триггеров, представлений и процедур:

- триггер на добавление нового пользователя с проверкой наличия добавляемой почты в базе

```

1  create trigger InsertStudentTrigger before insert on Student
2  for each row begin
3      if ((select count(*) from Student where new.mail =
4          mail) != 0) then
5          SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'SameMail';
6      end if;
7  end//

```

- процедура для получения роли администратора

```

1  create procedure MakePersonAdmin (in PersID bigint, in AdminRoleID int)
2  begin
3      declare PersonMail varchar(100);
4      set PersonMail = (select Mail from Student where PersID = PersonID);
5      insert Worker(Mail, Pass, PersonName, RegDate, RoleID)
6      select Mail, Pass, PersonName, RegDate, AdminRoleID from Student where
       ↪ Student.PersonID = PersID;
7      delete from Student where Student.PersonID = PersID;
8      select PersonID from Worker where PersonMail = Mail;
9  end //

```

- процедура для добавления новой темы

```

1  create procedure AddTheme (in CID int, in TName varchar(100))
2  begin

```



```

3      declare TPos int;
4      set TPos = (select count(*) from Theme where CourseID = CID);
5      insert Theme (ThemeName, CourseID, Pos) values (TName, CID, TPos);
6      select ThemeID from Theme where CourseID = CID and Pos = TPos;
7  end //

```

- процедура для удаления темы

```

1  create procedure DeleteTheme (in CID int, in TID int)
2  begin
3      declare ThemePos int;
4      set ThemePos = (select Pos from Theme where ThemeID = TID);
5      update Theme set Pos = Pos - 1 where CourseID = CID and Pos > ThemePos;
6      delete from Theme where ThemeID = TID;
7  end //

```

- процедура для изменения позиции темы в курсе

```

1  create procedure ChangeThemePos (in CID int, in TID int, in NewPos int)
2  begin
3      declare OldPos int;
4      set OldPos = (select Pos from Theme where ThemeID = TID);
5      if (OldPos > NewPos) then
6          update Theme set Pos = Pos + 1 where CourseID = CID and NewPos <= Pos and Pos
            ↪ < OldPos;
7      elseif (OldPos < NewPos) then
8          update Theme set Pos = Pos - 1 where CourseID = CID and Pos <= NewPos;
9      end if;
10     update Theme set Pos = NewPos where ThemeID = TID;
11 end //

```

- процедура для получения своих результатов

```

1  create procedure GetMyResults (in TID int, in PID int)
2  begin
3      select Pos, ResultDate, TaskText, Result.Answer, Truth from Result
4      join Task on Task.TaskID = Result.TaskID
5      join Question on Task.QuestionID = Question.QuestionID
6      where ThemeID = TID and PersonID = PID order by ResultDate;
7  end //

```

- процедура для получения задания из банка содержимого по заданному вопросу

```

1  create procedure GetTask (in QID int, in PID int)
2  begin
3      declare TID bigint;
4      if ((select count(*) from Task where QuestionID = QID) = 0) then
5          SIGNAL SQLSTATE '45000' SET MESSAGE_TEXT = 'NoTasks';
6      end if;
7      set TID = (select TaskID from Task where QuestionID = QID
8          and TaskID not in (select TaskID from Result where PersonID = PID) order by
9          ↪ Rand() limit 1);
10     if (TID is not null) then
11         select (select QuestionName from Question where QuestionID = QID), TaskText,
12             ↪ Answer, TaskID from Task where TaskID = TID;
13     end if;
14 end //

```

- представление для авторизации (объединение двух таблиц с аккаунтами)

```

1  create or replace view LoginTable
2  as select Mail, Pass, PersonId, 0 as UserType from Student
3  union select Mail, Pass, PersonId, RoleID from Worker;

```

Mail	Pass	PersonId	UserType
tosha@mail.ru	39045c3f9ea34ed30b0f1ee40a8b65b46a2a865...	1	0
tosha3@mail.ru	39045c3f9ea34ed30b0f1ee40a8b65b46a2a865...	2	0
tosha2@mail.ru	39045c3f9ea34ed30b0f1ee40a8b65b46a2a865...	1	3

- представление для просмотра всех пользователей системы с указанием даты и времени последнего входа, а также категории пользователя (учебной группы / должности)

```

1  create or replace view UsersTable
2  as select Mail, PersonName, LastLog, GroupName from Student
3  left join StudentGroup on StudentGroup.GroupID = Student.GroupID
4  union select Mail, PersonName, LastLog, RoleName from Worker
5  join WorkerRole on WorkerRole.RoleID = Worker.RoleID;

```

Mail	PersonName	LastLog	GroupName
tosha@mail.ru	Комиссаров Антон Сергеевич	2022-12-14 09:49:25	M8O-311Б-20
tosha3@mail.ru	KAC	2022-12-12 02:16:28	NULL
tosha2@mail.ru	KAC	2022-12-14 09:59:36	Администратор

- представление для просмотра пользователей по каждому курсу

```

1  create or replace view CourseUsersTable
2  as select CourseStudent.CourseID, Mail, PersonName, CourseStudent.LastLog,
   ↪ 'Ученик' as GroupName from CourseStudent
3  join Student on Student.PersonID = CourseStudent.PersonID
4  union select CourseID, Mail, PersonName, CourseWorker.LastLog, RoleName from
   ↪ CourseWorker
5  join Worker on Worker.PersonID = CourseWorker.PersonID
6  join WorkerRole on WorkerRole.RoleID = Worker.RoleID;
```

CourseID	Mail	PersonName	LastLog	GroupName
1	tosha@mail.ru	Комиссаров Антон Сергеевич	2022-12-14 09:42:54	Ученик
1	tosha2@mail.ru	KAC	2022-12-14 09:59:33	Администратор

Опишем для всех групп пользователей права доступа к каждой таблице. Права доступа должны быть распределены так, чтобы для каждого объекта БД был хотя бы один пользователь, который имеет право добавлять и удалять данные из объекта. Права приведены в табл. 1. Используются следующие сокращения:

- s – чтение данных (SELECT);
- i – добавление данных (INSERT);
- u – модификация данных (UPDATE);
- d – удаление данных (DELETE).

Таблица 1: Права доступа к таблицам для групп пользователей

Таблицы	Группы пользователей (роли)			
	Ученик	Преподаватель	Менеджер	Администратор
Course	S	S	S	SIUD
Theme	S	SIUD	S	SIUD
Lesson	S	SIUD	S	SIUD
Question	S	SIUD	S	SIUD
Task	S	SIUD	S	SIUD
Result	SI	S	S	SIUD
Student	S	S	SIUD	SIUD
StudentGroup	S	S	SIUD	SIUD
Worker	S	S	S	SIUD
WorkerRole	S	S	S	SIUD
CourseStudent	S	S	SIUD	SIUD
CourseWorker	S	S	S	SIUD

Права назначает администратор БД (или администратор безопасности, если система сложная и администраторов несколько).

Права доступа пользователей предоставляются с помощью команды GRANT. Рассмотрим для примера права ученика (student\_user). Права доступа к отношениям Course и Result могут быть описаны следующим образом:

**GRANT SELECT ON Course TO student\_user;**

**GRANT SELECT, INSERT ON Result TO student\_user;**

Права доступа менеджера (manager\_user) к представлению LoginTable могут быть описаны следующим образом:

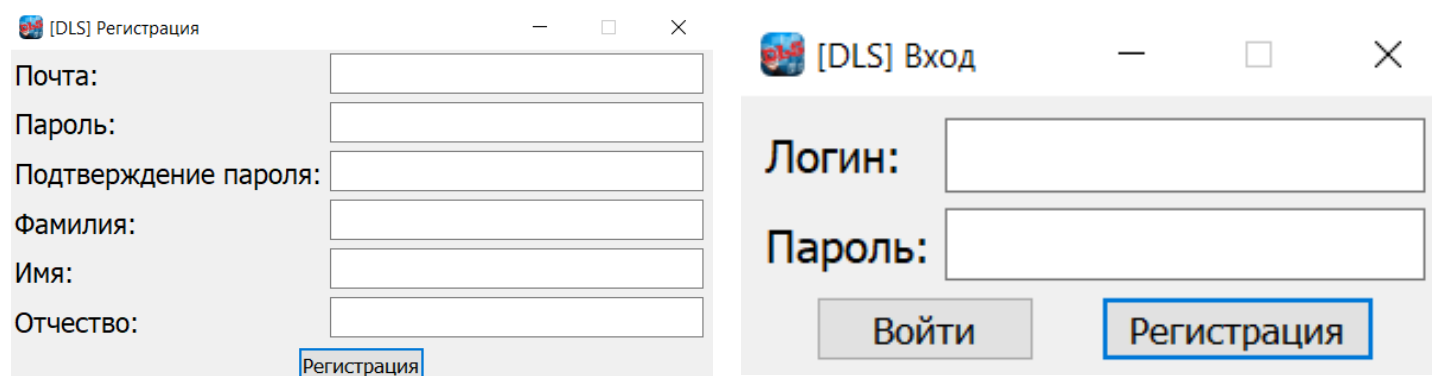
**GRANT SELECT ON LoginTable TO manager\_user;**

## 4 Разработка прикладной программы

**Distance Learning System [DLS]** – прикладная программа для демонстрации возможностей базы данных системы дистанционного обучения и взаимодействия с ней.

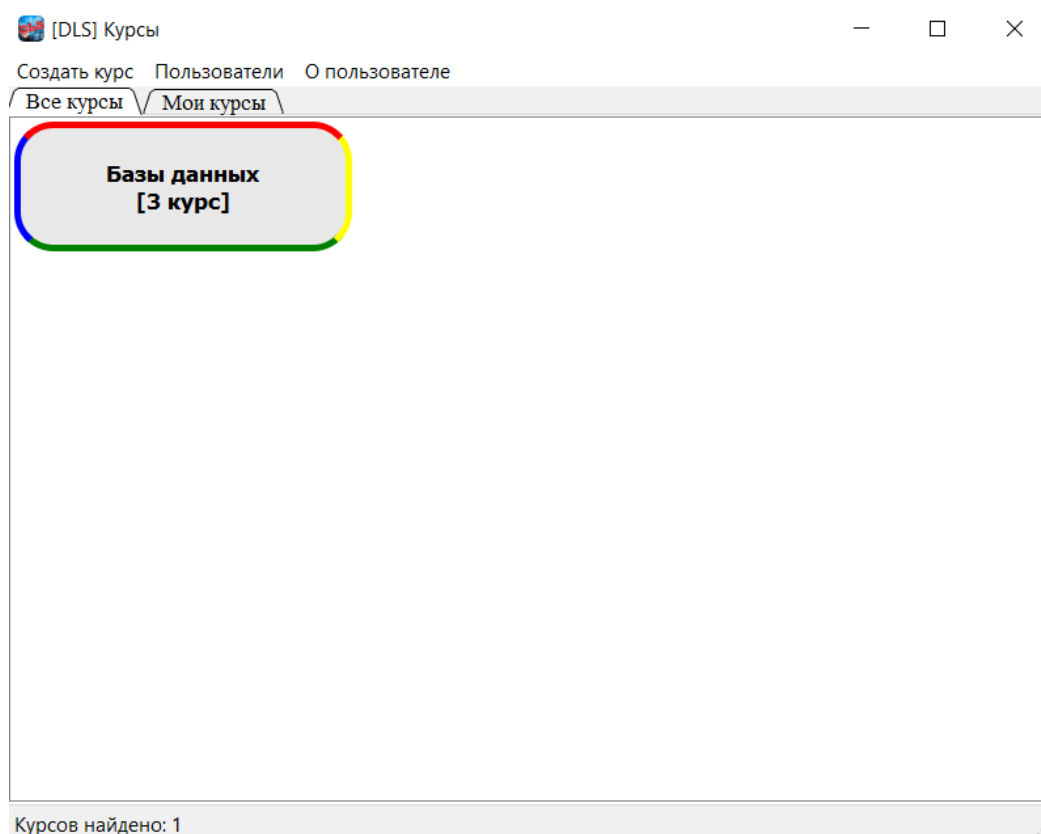
Программа предоставляет следующие возможности:

- Регистрация и авторизация пользователей.




The image shows two side-by-side screenshots of the DLS application interface. The left window is titled "[DLS] Регистрация" (Registration) and contains input fields for "Почта:" (Email), "Пароль:" (Password), "Подтверждение пароля:" (Confirm password), "Фамилия:" (Surname), "Имя:" (Name), and "Отчество:" (Patronymic). A "Регистрация" button is at the bottom right. The right window is titled "[DLS] Вход" (Login) and contains input fields for "Логин:" (Login) and "Пароль:" (Password). It has "Войти" (Login) and "Регистрация" buttons at the bottom.


- Просмотр всех курсов и тех, на которые подписан пользователь.



The image shows a screenshot of the "[DLS] Курсы" (Courses) window. At the top, there are tabs: "Создать курс" (Create course), "Пользователи" (Users), and "О пользователе" (About user). Below these are sub-tabs: "Все курсы" (All courses) and "Мои курсы" (My courses). The main content area displays a rounded rectangle with the text "Базы данных [3 курс]" (Database [3 course]). At the bottom, a status bar indicates "Курсов найдено: 1" (Courses found: 1).


- Просмотр информации о конкретном пользователе, а также списка всех пользователей и подписчиков каждого курса.

 [DLS] О пользователе
 ✕




ФИО: KAC  
 Почта: tosha2@mail.ru  
 Дата регистрации: 2022-12-16  
 Должность: Администратор  
 Деятельность: Полный контроль над DLS и её БД  
 Обязанности: Не указано

OK

 [DLS] Просмотр пользователей
 

tosha3@mail.ru: KAC.	Последний вход: 2022-12-16 21:13:29.
tosha2@mail.ru: KAC.	Последний вход: 2022-12-16 21:13:48. Администратор
tosha@mail.ru: Комиссаров Антон Сергеевич.	Последний вход: 2022-12-16 21:13:29. M8O-3115-20

- Просмотр, добавление, изменение и удаление тем, а также просмотр их содержания.

 [DLS] Курс "Базы данных"
 

—
□
✕

Создать тему
 Изменить название темы
 Изменить позицию темы
 Удалить тему
 Пользователи
 Результаты

Тема "Триггеры"

Занятия

[Занятие №1] Определение триггеров

[Занятие №2] Картинка с примером

Вопросы

[Вопрос №1] Параметры

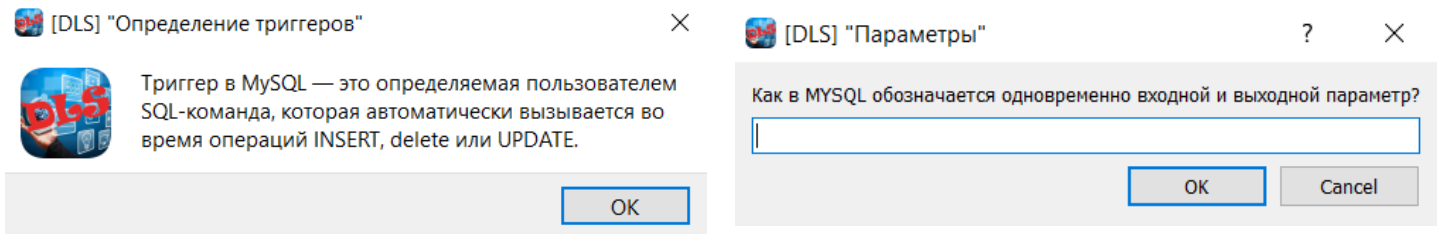
[Вопрос №2] Переменные

[Вопрос №3] Вывод

Пунктов найдено: 5

30

- Просмотр учебных материалов, а также возможность ответа на вопрос.



- Просмотр своих результатов, отсортированных по дате и времени ответа на вопрос.

[DLS] Результаты		
[2022-12-16 21:31:09]	Вопрос №1: Как в MYSQL обозначается одновременно входной и выходной параметр?	Результат: 1 балл. Ваш ответ: "inout"
[2022-12-16 21:31:16]	Вопрос №1: Есть ли в MYSQL параметры по умолчанию в процедурах? 1) Да 2) Нет	Результат: 1 балл. Ваш ответ: "2"
[2022-12-16 21:31:20]	Вопрос №1: Сколько в MYSQL видов параметров в процедурах?	Результат: 1 балл. Ваш ответ: "4"

# ЗАКЛЮЧЕНИЕ

Таким образом, в работе было показано, как создать базу данных «Дистанционная система обучения» для повышения эффективности работы преподавателей и процесса получения знаний в целом.

Также были реализованы возможности:

- добавления новых курсов;
- добавления, изменения, удаления тем;
- добавления новых пользователей;
- просмотра пользователей системы по каждому курсу и системе в целом;
- создания и просмотра результатов выполнения тестовых вопросов.

И функциональные возможности:

1. Представлять перечень курсов;
2. Представлять перечень тем для каждого курса;
3. Выдавать информацию о занятиях;
4. Выдавать информацию о вопросах;
5. Выдавать информацию о сотрудниках;
6. Выдавать информацию об учениках;
7. Представлять результаты ответов на вопросы.



# СПИСОК ЛИТЕРАТУРЫ

1. Баженова И.Ю. Основы проектирования приложений баз данных (2-е изд.) – М.: НОУ «Интуит», 2016 – 237 с.
2. <http://window.edu.ru/resource/437/79437/files/Учебное%20пособие.pdf>
3. <https://metanit.com/sql/mysql/>
4. <https://dev.mysql.com/doc/workbench/en/wb-eer-diagram-editor.html>