**МИНОБРНАУКИ РОССИИ**

Федеральное государственное бюджетное

образовательное учреждение высшего образования

**«Ухтинский государственный технический университет»**

**(УГТУ)**

Кафедра вычислительной техники, информационных систем и технологий

**КУРСОВОЙ ПРОЕКТ**

Дисциплина: «Управление данным»

Тема: «Учёт оборудования в учебном заведении»

Шифр 191407 Группа ИСТ-19о-Б Курс 3

Морданов Егор Владимирович

Проверил:  
 доцент кафедры ВТИСиТ П. В. Кожевникова

Ухта

2022

**МИНОБРНАУКИ РОССИИ**

Федеральное государственное бюджетное

образовательное учреждение высшего образования

**«Ухтинский государственный технический университет»**

**(УГТУ)**

Кафедра \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

УТВЕРЖДАЮ

**Зав. кафедрой**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, инициалы, фамилия

«\_\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 20\_\_\_ г.

**ПОЯСНИТЕЛЬНАЯ ЗАПИСКА**

к курсовому проекту (работе) по \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

наименование учебной дисциплины

на тему:\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Автор проекта (работы)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата, инициалы, фамилия

Направление подготовки\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

номер, наименование

Обозначение курсового проекта (работы)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_группа\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Руководитель проекта** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата инициалы, фамилия

Проект (работа) защищен(а) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

дата оценка

**Члены комиссии** \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата инициалы, фамилия

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

подпись, дата инициалы, фамилия

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

город, год защиты

**МИНОБРНАУКИ РОССИИ**

Федеральное государственное бюджетное

образовательное учреждение высшего образования

**«Ухтинский государственный технический университет»**

**(УГТУ)**

Кафедра \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**ЗАДАНИЕ НА КУРСОВОЙ ПРОЕКТ (РАБОТУ)**

Студент\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_код\_\_\_\_\_\_\_\_\_\_\_\_группа\_\_\_\_\_\_\_\_\_\_

Фамилия, инициалы

1.Тема\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

2. Срок предоставления проекта (работы) к защите

« » 20 г.

3.Исходные данные для проектирования (научного исследования)\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4. Содержание пояснительной записки курсового проекта (работы)

4.1.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.2.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.3.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.4.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.5.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

4.6.\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

5. Перечень графического материала \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

**Руководитель проекта (работы) \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Подпись, дата Фамилия, инициалы

**Задание принял к исполнению \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_**

Подпись, дата Фамилия, инициалы

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 5](#_Toc94603638)

[1 ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 7](#_Toc94603639)

[2 ФИЗИЧЕСКАЯ МОДЕЛЬ БАЗЫ ДАННЫХ 9](#_Toc94603640)

[3 ПРОЕКТ БАЗЫ ДАННЫХ 14](#_Toc94603641)

[3.1 Поддержание целостности 14](#_Toc94603642)

[3.2 Хранимые процедуры базы данных 15](#_Toc94603643)

[3.3 Представления и функция для получения отчётов.......... 17](#_Toc94603644)

[3.4 Обеспечение безопасности 18](#_Toc94603645)

[3.5 Создание тестовых примеров 20](#_Toc94603646)

[Заключение 23](#_Toc94603647)

[Список использованной литературы 24](#_Toc94603648)

[ПРИЛОЖЕНИЕ А 26](#_Toc94603649)

# ВВЕДЕНИЕ

Учет оборудования – ключевой источник получения актуальной и прозрачной информации об оборудовании предприятия и их состоянии.

Зачем нужно вести учет компьютерной техники?

1. Неожиданно выясняется, что часть оборудования переместили, часть списали, другая часть не известно куда делась. Бухгалтерия в ярости, руководитель ИТ в замешательстве, а эникейщики нарезают круги по фирме и впопыхах ищут, где же та или иная оргтехника. Так происходит потому, что бухгалтерии важен бухгалтерский учет, а в ИТ-структуре часто учет управленческий. Главной задачей управленческого учёта является ответ на вопрос, в каком состоянии находится организация, как необходимо распределить имеющиеся ресурсы, чтобы повысить эффективность деятельности. Т.е. выгодно, чтобы это оборудование стояло вот там? Отлично, переносим. Директор сказал, чтобы мы обеспечили рабочим местом нового сотрудника, а свободное оборудование числится в другой фирме холдинга? Вот это и есть управленческий учет, который часто не совпадает с бухгалтерским и тем более налоговым. Действительно, как решить вопрос с бухгалтерией? На самом деле это хороший вопрос. Очевидно, ИТ-отдел никто не пустит отражать в бухгалтерии все, что касается бух учета. Бухгалтерии вообще все равно где и как работает оборудование, им самое главное, чтобы в любой момент могли показать, где оно, оперативно подавались сведения о списании оргтехники, картриджей, клавиатур с мышками, а остальное им совершенно не важно.

2. Что мы будем делать, когда срочно нужно узнать, где стоит то или иное оборудование? По бухгалтерии ИБП закреплен за Ивановым, а где этот Иванов находится совершенно не ясно. Вроде бы наши ИТ-специалисты и переносили, и настраивали, но, когда и где - никто сразу и не вспомнит.

3. Нам поставили задачу докинуть на конкретном древнем компьютере память, чтобы работал побыстрее. Тут даже бухгалтерия не поможет, они могут вести учет комплектов, а уж о том, что "внутри" они и не знают. Чтобы понять, где можно достать лишней памяти, нам нужно методом перебора, вскрывать каждый компьютер или запускать на компьютере программу мониторинга и определять, что внутри и только после этого мы найдем то, что нам нужно. Что будет, если таких компьютеров 1000? Как быть?

Поэтому приходим к простому решению, что учет оборудования значительно упрощает работу как для сотрудников, так и для управляющих на предприятии. Целью курсовой работы является создание базы данных, удовлетворяющее требованиям для работы в системе.

Глава 1. Глава, содержащая описание предметной области. А также способы исследования ее.

Глава 2. Глава, содержащая описание использованной СУБД, а также логическую и физическую схему базы данных. Приводятся таблицы базы данных.

Глава 3.

Заключение. В заключение подводятся итоги проделанной работы с приведением выводов, сделанных в процессе разработки данного проекта.

1. ОПИСАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ
2. В данном процессе участвуют: преподаватели; системные администраторы ответственные за замену и починку, а также учет оборудования; сам учебный кабинет, в котором находится оборудование.
3. Реализация процесса.
   1. После заполнения заявления на работу системный администратор ответственный за учёт оборудования получает приказ о приёме на работу, нужных ему инструкций и данных для входа в программу учета оборудования.
   2. Он приходит на рабочее место, предоставляет приказ о приёме на работу, на основании которого ему сообщают данные для входа в программу и вводный инструктаж.
   3. Данные, которые сотрудник записывает в карточку учёта оборудования:
      1. Номер аудитории.
      2. Табельный номер оборудования.
      3. Тип оборудования.
      4. Наименования оборудования.
      5. Комплектующие оборудования (если есть).
      6. Состояние оборудования.
   4. После того как работник узнает данные, он должен их проверить на соответствие с данными.
   5. В случае поломки или выхода из строя какого-либо оборудования создается заявка на её починку/замену, само оборудования на время ремонтных работ не используется, студентам нужно использовать другое рядом находящиеся оборудование.
   6. Работник во время ремонтных работ обязан продиагностировать и предоставить, прежде чем чинить, стоимость починки оборудования с прилегающими ценами для замены комплектующих оборудования.
   7. Работник должен следить за жизненным циклом оборудования и определять, когда нужно провести профилактику, ремонт или списание в утиль.
4. В дальнейшем эти данные использует преподаватель и системный администратор.
5. ФИЗИЧЕСКАЯ МОДЕЛЬ БАЗЫ ДАННЫХ

Используемая СУБД:

* В качестве основного средства для разработки физической модели базы данных на тему «Подбора заведения общественного питания» была выбрана СУБД Microsoft SQL Server 2008. Аргументами в пользу этого программного обеспечения стали:
* Наличие опыта работы с этой СУБД в рамках практических занятий по дисциплине «СУБД»;
* Microsoft SQL Server 2019 является последней версией этой линейки продукта;
* Microsoft SQL Server 2019 проста в использовании и поддерживает все необходимые для создания проекта средства, такие как: хранимые процедуры и функции, представления, транзакции, триггеры и курсоры;
* Microsoft SQL Server 2019 предоставляет точный и гибкий контроль для обеспечения безопасности данных: разделение доступа к данным между разными пользователями через роли и схемы.

Перед построением физической модели базы данных обычно проводят концептуальное и логическое проектирование. В ходе анализа предметной области были выделены основные сущности, участвующие в процессе. На основе концептуальной модели была построена логическая модель данных (см. Рисунок 1), которая является начальным прототипом будущей базы данных. Кроме того, были сгенерированы отношения и проведена их нормализация.

Поскольку логическая модель не учитывает всех особенностей целевой СУБД, то следующим этапом проектирования является разработка физической модели базы данных (см. Рисунок 2), которая тоже в свою очередь подразделяется на несколько этапов. Первый – создание таблиц в целевой СУБД и поддержание согласованности данных с помощью ограничений. Такой метод поддержания целостности называется декларативным. Второй этап – анализ бизнес-правил, налагаемых на БД, и применение императивных средств для их поддержания (в случае, если декларативных средств СУБД для этого недостаточно). В качестве императивных средств могут выступать триггеры. Такой вид целостности называется процедурным.

Для решения поставленной задачи была составлена база данных, состоящая из семи таблиц: hardware (Таблица 1), hardware\_type (Таблица 2), class (Таблица 3), teacher (Таблица 4), persona (Таблица 5), sys\_admin (Таблица 6), post (Таблица 7).

Таблица hardware создана с целью хранения информации о оборудовании в кабинете. Таблица hardware\_type является справочной таблицей, хранящей данные о типах оборудования. Таблица class создана с целью хранить информацию в каком кабинете находится оборудование и какой преподаватель за ним закреплён. Таблица teacher создана с целью хранить информацию о преподавателях. Таблица sys\_admin создана с целью хранить информацию о системных администраторах. Таблица persona создана с целью хранить личную информацию преподавателя или системного администратора. Таблица post является справочной таблицей, хранящей в себе информацию о должностях.

Таблица 1 - Оборудование

|  |  |  |  |
| --- | --- | --- | --- |
| hardware | | | |
| **Имя** | **Тип** | **Пояснение** | **Ограничения** |
| hardware\_id | int | Идентификатор и серийный номер | PK |
| hardware\_type | int | Тип оборудования | FK |
| hardware\_status | varchar[50] | Статус оборудования |  |
| data\_check | date | Дата проверки |  |
| specification | text | Спецификация оборудования |  |
| class\_hardware | varchar[50] | Кабинет, в котором находится оборудование | FK |

Таблица 2- Справочник типов оборудования

|  |  |  |  |
| --- | --- | --- | --- |
| Hardware\_type | | | |
| **Имя** | **Тип** | **Пояснение** | **Ограничения** |
| Id\_type | Int | Идентификатор | PK |
| Hardware\_type | Varchar[50] | Тип оборудования |  |

Таблица 3 – Кабинет

|  |  |  |  |
| --- | --- | --- | --- |
| Class | | | |
| **Имя** | **Тип** | **Пояснение** | **Ограничения** |
| Class\_id | Varchar[50] | Идентификатор | PK |
| Teacher | Int | ID преподавателя | FK |

Таблица 4 – Сотрудник

|  |  |  |  |
| --- | --- | --- | --- |
| teacher | | | |
| **Имя** | **Тип** | **Пояснение** | **Ограничения** |
| Employee\_id | Int | Идентификатор | PK |
| employee\_ persona | Int | ID сотрудника | FK |
| employee\_type | Varchar[50] | Тип сотрудника |  |

Таблица 5 – Персона

|  |  |  |  |
| --- | --- | --- | --- |
| Persona | | | |
| **Имя** | **Тип** | **Пояснение** | **Ограничения** |
| Persona\_id | Int | Идентификатор | PK |
| First\_name | Varchar[50] | Имя сотрудника | NOT NULL |
| Second\_name | Varchar[50] | Фамилия сотрудника | NOT NULL |
| Middle\_name | Varchar[50] | Отчество сотрудника |  |
| Phone\_number | Varchar[20] | Номер телефона сотрудника |  |
| Post | Int | Должность сотрудника | FK |

Таблица 7 – Должность

|  |  |  |  |
| --- | --- | --- | --- |
| Post | | | |
| **Имя** | **Тип** | **Пояснение** | **Ограничения** |
| Post\_id | Int | Идентификатор | PK |
| Post\_name | Varchar[50] | Наименование должности | NOT NULL |

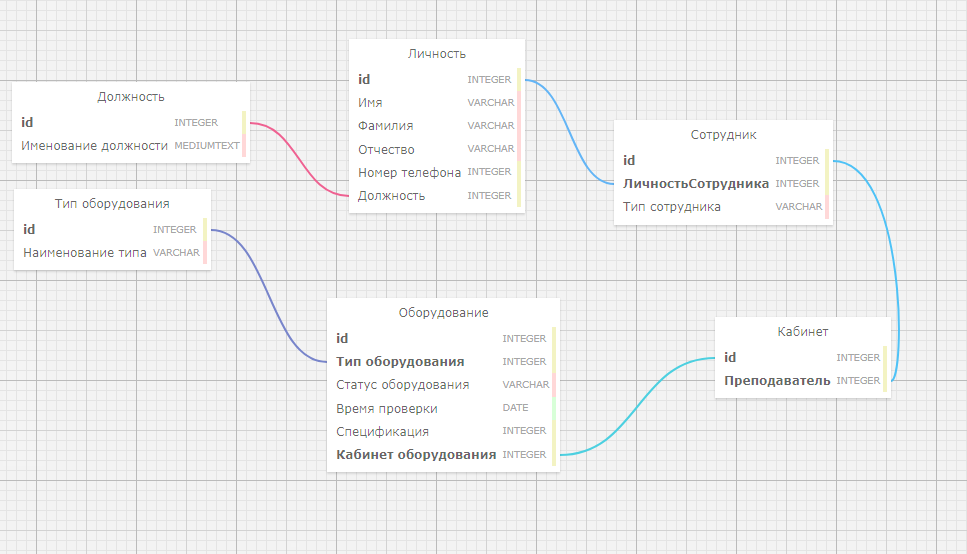


Рисунок 1 – Логическая модель базы данных

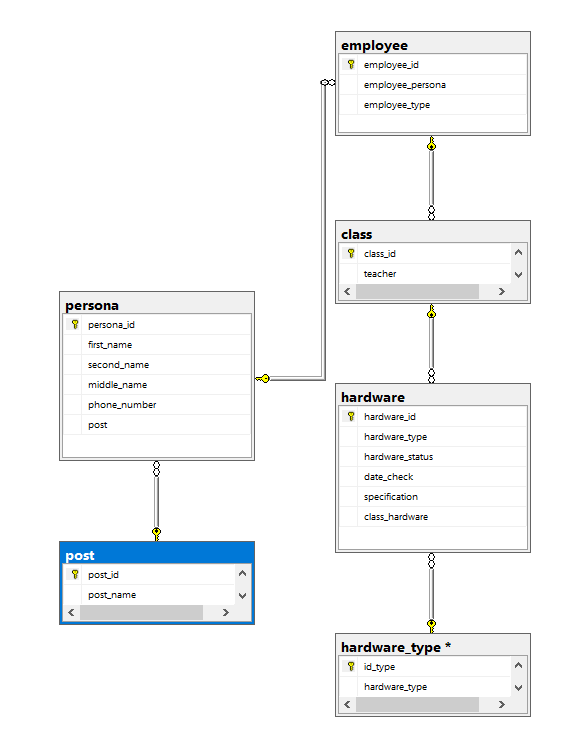


Рисунок 2 – Физическая модель бд

1. ПРОЕКТ БАЗЫ ДАННЫХ
2. Поддержание целостности

Процедурная целостность необходима для обеспечения корректности вводимых данных и соблюдения правил решаемой задачи (бизнес-логики). Существует несколько основных механизмов для поддержания такого вида целостности: триггеры, хранимые процедуры и функции. В данном курсовом проекте были использованы все, вышеперечисленные, механизмы.

Хранимые процедуры представляют собой набор команд, состоящий из одного или нескольких операторов SQL и сохраняемый в базе данных в откомпилированном виде, что обеспечивает быстрый доступ к данным. 33 В данном курсовом проекте хранимые процедуры были применены для ограничения доступа пользователей к таблицам базы данных, а также для разграничения прав на добавление/редактирование/удаление данных.

Поскольку в базе данных будет хранится серийные номера оборудования, то они являются уникальными и не могут повторятся, в случае повторения процедуры выполняют проверки на такие случаи и исправляет её, для добавления данных в базу данных.

Листинг 1 – Пример данной процедуры

|  |
| --- |
| CREATE PROCEDURE AddHardware @h\_id int, @htype\_id int,  @h\_status nvarchar(80), @h\_spec text,  @h\_class nvarchar(80) AS BEGIN DECLARE @h\_date date;  SET @h\_date = GETDATE();  IF EXISTS (SELECT hardware\_id FROM hardware WHERE hardware\_id = @h\_id) BEGIN  PRINT('Данный серийный номер уже был указан');  RETURN;  END;  IF NOT EXISTS(SELECT hardware\_type FROM hardware\_type WHERE hardware\_type.id\_type = @htype\_id) BEGIN  INSERT INTO hardware\_type (hardware\_type) VALUES ('unknown');  SELECT @htype\_id = IDENT\_CURRENT('hardware\_type');  END;  IF NOT EXISTS(SELECT class\_id FROM class WHERE class.class\_id = @h\_class)  INSERT INTO class VALUES(@h\_class, NULL);  INSERT INTO hardware VALUES(@h\_id, @htype\_id, @h\_status, GETDATE(), @h\_spec, @h\_class);  END; |

Тут произойдет проверка на существование в базе данных на такой же серийный номер и выведет сообщение «Данный серийный номер уже был указан».

1. Хранимые процедуры базы данных

Хранимые процедуры представляют собой набор команд, состоящий из одного или нескольких операторов SQL и сохраняемый в базе данных в откомпилированном виде, что обеспечивает быстрый доступ к данным. В данном курсовом проекте хранимые процедуры были применены для ограничения доступа пользователей к таблицам базы данных, а также для разграничения прав на добавление/редактирование/удаление данных.

Хранимые процедуры AddHardware, ChangeHardwareSpec, DeleteHardware, ShowHardwareReport, AddClassId, DeleteClassId, AddHardwareType

Таблица 8 Список процедур

|  |  |
| --- | --- |
| Процедура | Описание |
| AddHardware | Добавляет оборудование в базу данных. Пример:  EXEC AddHardware 1, 1, 'Исправен', 'Спецификация', '208'; |
| ChangeHardwareSpec | Редактирует спецификацию у указанного оборудования. Пример: EXEC ChangeHardwareSpec id, 'Спецификация'; |
| DeleteHardware | Удаляет оборудование по его серийному номеру. Пример:  EXEC DeleteHardware id; |
| ShowHardwareReport | Показывает всё оборудование находящиеся в базе данных.  Пример: EXEC ShowHardwareReport; |
| AddClassId | Добавляет кабинет в базу данных. Пример:  EXEC AddClassId '208'; |
| DeleteClassId | Удаляет кабинет из базы данных. Пример:  EXEC DeleteClassId '208'; |
| AddHardwareType | Добавляет новый тип оборудования в справочную таблицу.  Пример: EXEC AddHardwareType 'Монитор'; |

Триггеры представляют специальный тип хранимой процедуры, которая вызывается автоматически при выполнении определенного действия над таблицей или представлением, в частности, при добавлении, изменении или удалении данных, то есть при выполнении команд INSERT, UPDATE, DELETE.

Листинг 2 – Пример триггера в базе данных

|  |
| --- |
| CREATE TRIGGER DeleteUnusedHardwareTypes ON hardware AFTER DELETE AS BEGIN  IF NOT EXISTS(SELECT h.hardware\_id FROM hardware AS h, deleted AS d  WHERE h.hardware\_id = d.hardware\_id)  DELETE FROM hardware\_type WHERE id\_type IN (SELECT hardware\_type FROM deleted);  END; |

Данный триггер удаляет тип оборудования, если все оборудование использующий данный тип был удалён, тем самым убирает захламлённость типами оборудования в базе данных.

Листинг 3 – Процедуры, находящиеся в базе данных

|  |
| --- |
| CREATE PROCEDURE AddHardware @h\_id int, @htype\_id int,  @h\_status nvarchar(80), @h\_spec text,  @h\_class nvarchar(80) AS BEGIN  DECLARE @h\_date date;  SET @h\_date = GETDATE();  BEGIN TRANSACTION;  IF NOT EXISTS(SELECT hardware\_type FROM hardware\_type WHERE hardware\_type.id\_type = @htype\_id) BEGIN  INSERT INTO hardware\_type (hardware\_type) VALUES ('unknown');  SELECT @htype\_id = IDENT\_CURRENT('hardware\_type');  END;  IF NOT EXISTS(SELECT class\_id FROM class WHERE class.class\_id = @h\_class)  INSERT INTO class VALUES(@h\_class, NULL);  INSERT INTO hardware VALUES(@h\_id, @htype\_id, @h\_status, GETDATE(), @h\_spec, @h\_class);  COMMIT TRANSACTION;  END;  CREATE PROCEDURE ChangeHardwareSpec @h\_id int, @h\_spec text AS BEGIN  UPDATE hardware SET specification = @h\_spec WHERE hardware\_id = @h\_id;  END;  CREATE PROCEDURE DeleteHardware @h\_id int AS BEGIN  DELETE hardware WHERE hardware\_id = @h\_id;  END;  CREATE PROCEDURE ShowHardwareReport AS BEGIN  SELECT TOP (1000) \*  FROM [hardware].[dbo].[hardwareView]  END;  CREATE PROCEDURE AddClassID @classid nvarchar(50) AS BEGIN  IF NOT EXISTS(SELECT class\_id FROM class WHERE class.class\_id = @classid)  INSERT INTO class VALUES(@classid, NULL);  END; |

Продолжение листинга 3

|  |
| --- |
| CREATE PROCEDURE DeleteClassID @classid nvarchar(50) AS BEGIN  DELETE class WHERE class\_id = @classid;  END;  CREATE PROCEDURE AddHardwareType @h\_type nvarchar(50) AS BEGIN  INSERT INTO hardware\_type(hardware\_type) VALUES(@h\_type);  END; |

Процедура AddHardware является транзакцией.

Транзакция – неделимая с точки зрения воздействия на БД последовательность операторов манипулирования данными, приводящая к одному из двух возможных результатов: либо последовательность выполняется, если все операторы правильные, либо вся транзакция откатывается, если хотя бы один оператор не может быть успешно выполнен.

1. Представления и функция для получения отчётов

Представления - мощное средство СУБД, позволяющее скрыть реальную структуру БД от некоторых пользователей за счет определения представлений. В данном курсовом проекте представление использовалось для объединения данных из нескольких таблиц в статический отчет.

Пример представления:

Поскольку в базе данных кабинеты имеют справочную таблицу преподавателей, преподаватели отображаются в виде цифр, для этого используется представление ClassView, оно позволяет отобразить кабинеты, а также благодаря справочной таблицу отобразить данные преподавателя, который закреплён за кабинетом.

Листинг 4 – Представление кабинетов

|  |
| --- |
| CREATE VIEW [dbo].[ClassView]  AS  SELECT dbo.class.class\_id, dbo.persona.first\_name, dbo.persona.second\_name, dbo.persona.middle\_name, dbo.persona.phone\_number, dbo.post.post\_name  FROM dbo.post LEFT OUTER JOIN  dbo.persona ON dbo.post.post\_id = dbo.persona.post LEFT OUTER JOIN |

Продолжение листинга 4

|  |
| --- |
| dbo.teacher ON dbo.persona.persona\_id = dbo.teacher.persona\_teacher RIGHT OUTER JOIN  dbo.class ON dbo.teacher.teacher\_id = dbo.class.teacher |

Отчёт формируется за счёт представления оборудования, сотрудник может ввести команду и ему выведется всё оборудование находящиеся в кабинетах.

Листинг 5 – Отчёт оборудования

|  |
| --- |
| CREATE PROCEDURE ShowHardwareReport AS BEGIN  SELECT TOP (1000) \*  FROM [hardware].[dbo].[hardwareView]  END; |

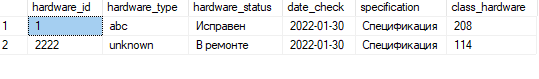


Рисунок 3 – Результат выполнения отчёта

1. Обеспечение безопасности

В ходе изучения предметной области в рамках курсового проекта по дисциплине «Управление данными» были определены основные пользователи для работы с базой данный:

* Администратор;
* Сотрудники;

Администратор заполняет всю информацию в базе данных, т.е. добавляет туда сотрудников, оборудования и т.д.

Сотрудники могут просматривать данные, находящиеся в базе данных, к примеру оборудование находящиеся в нужном им кабинете.

Таблица 9 – Права доступа пользователей.

|  |  |  |
| --- | --- | --- |
|  | Администратор | Сотрудник |
| Заполнение/редактирование информации в базе данных | + | - |
| Просмотр данных | + | + |

Для обеспечения безопасности базы данных было создано 2 пользователя.

Листинг 6 – Пользователи базы данных

|  |
| --- |
| CREATE USER Employee WITHOUT LOGIN;  CREATE USER Administrator WITHOUT LOGIN; |

Им были выданы следующие права (Таблица 3).

Таблица 10 – Матрица доступа пользователей.

|  |  |  |
| --- | --- | --- |
| Операции | Сотрудник | Администратор |
| AddHardware |  | RWED |
| ChangeHardwareSpec |  | RWED |
| DeleteHardware |  | RWED |
| ShowHardwareReport | R | RWED |
| AddClassID |  | RWED |
| DeleteClassID |  | RWED |
| AddHardwareType |  | RWED |

Права выдавались с помощью команды GRANT EXEC ON «Название команды» TO «Пользователь».

Таким образом, безопасность базы данных была обеспечена за счет создания необходимого числа учетных записей для различных ролей пользователей БД, ограничения их доступа к таблицам и предоставления им права выполнения хранимых процедур, отвечающих за необходимые им действия.

1. Создание тестовых примеров

Для проверки корректности физической модели и правильности реализованных процедур и функций в хранилище данных были введены тестовые данные для каждой из имеющихся сущностей. Процедура тестирования заключалась в том, что с помощью скрипта создавалась новая база данных. Далее запускался файл, содержащий таблицы БД и все декларативные ограничения, накладываемые на эти таблицы. После этого запускались файлы, в которых хранятся: хранимые процедуры вставки, обновления и удаления; хранимые процедуры и функции, необходимые для формирования отчетом; триггеры; индексы; матрица доступа. Далее запускался файл, содержащий тестовые данные для таблиц, затем вызывались функции формирования отчетов.

Добавление оборудования:

EXEC AddHardware 1, 1, 'Исправен', 'Спецификация', '208';

EXEC AddHardware 2222, 2, 'В ремонте', 'Спецификация', '114';

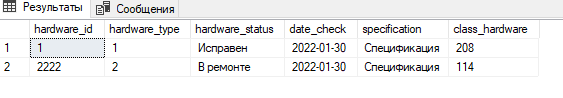


Рисунок 4 – Результат добавления оборудования

Добавление классов:

EXEC AddClassID 208;

EXEC AddClassID 114;

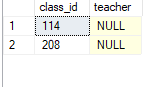


Рисунок 5 – Результат добавления кабинетов

Добавление типов оборудования:

EXEC AddHardwareType 'Монитор';

EXEC AddHardwareType 'Клавиатура';

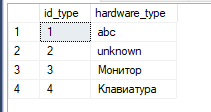


Рисунок 6 – Результат добавления типов оборудования

Добавление должностей

INSERT INTO post(post\_name) VALUES('Доцент'),('Администратор');

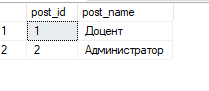


Рисунок 7 – Результат добавление кабинетов

Добавление сотрудников, для того чтобы их добавить нужно сначала добавить их персону, а потом уже закрепить их как сотрудников:

INSERT INTO persona(first\_name,second\_name,middle\_name,phone\_number,post)

VALUES('Егор','Морданов','Владимирович','89091274466','1'),

('Иван','Сычёв','Викторович','89091276644','2');

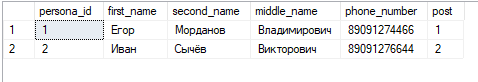


Рисунок 8 – Результат добавления персоны

INSERT INTO employee VALUES (1,1,'Преподаватель'),(2,2,'Системный администратор');

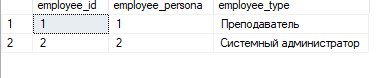


Рисунок 9 – Результат добавления сотрудника

Для более корректного отображения использовались различные представления такие как ClassView – для простого отображения кабинетов и сотрудников, которые за данными кабинетам закреплены.

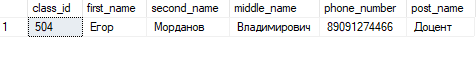


Рисунок 10 – Представление ClassView

Также использовалось представление hardwareView для создания отчёта об оборудовании в кабинетах.

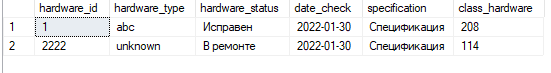


Рисунок 11 – Представление hardwareView

# Заключение

Целью курсовой работы было создание базы данных по теме «Учёт оборудования в учебном заведении».

В рамках курсового проекта был изучен бизнес-процесс по документации оборудования в учебном заведении, на основе критериев пользователя. Результатом моделирования стала логическая и физическая диаграммы БД, которые описывают целиком создаваемую базу данных.

На основе сформированных в курсовом проекте по «УД» запросов были реализованы хранимые процедуры и функции, которые предоставляют пользователям всех необходимые выходные формы. Также были добавлены процедуры для добавления, редактирования и удаления данных.

Для обеспечения безопасности и надежности хранилища данных были выделены основные пользователи системы, и ими были предоставлены только необходимые им права на работу с данными хранилища.

Таким образом, все поставленные задачи были решены, а цель достигнута.

# Список использованной литературы

1. Рочев К.В. Информационные технологии. Анализ и проектирование информационных систем [Текст]: учеб. пособие / К. В. Рочев, – Ухта: УГТУ, 2018. – 139 с.
2. Учебник по SQL [Электронный ресурс]. URL: <http://www.sql-tutorial.ru/>

(Дата обращения: 29.01.2022)

1. Создание логической модели базы данных [Электронный ресурс]. URL: <https://itteach.ru/bpwin/sozdanie-logicheskoy-modeli> (Дата обращения: 29.01.2022)

ПРИЛОЖЕНИЕ

# ПРИЛОЖЕНИЕ А

USE [master]

GO

/\*\*\*\*\*\* Object: Database [hardware] Script Date: 16.11.2021 12:19:04 \*\*\*\*\*\*/

CREATE DATABASE [hardware]

GO

USE [hardware]

GO

/\*\*\*\*\*\* Object: Table [dbo].[class] Script Date: 30.11.2021 10:47:28 \*\*\*\*\*\*/

CREATE TABLE [dbo].[class](

[class\_id] [varchar](50) NOT NULL,

[teacher] [int] NULL,

PRIMARY KEY CLUSTERED([class\_id])

)

GO

/\*\*\*\*\*\* Object: Table [dbo].[hardware] Script Date: 30.11.2021 10:47:28 \*\*\*\*\*\*/

CREATE TABLE [dbo].[hardware](

[hardware\_id] [int] NOT NULL,

[hardware\_type] [int] NULL,

[hardware\_status] [varchar](50) NULL,

[date\_check] [date] NULL,

[specification] [text] NULL,

[class\_hardware] [varchar](50) NULL,

PRIMARY KEY CLUSTERED([hardware\_id])

)

GO

/\*\*\*\*\*\* Object: Table [dbo].[hardware\_type] Script Date: 30.11.2021 10:47:28 \*\*\*\*\*\*/

CREATE TABLE [dbo].[hardware\_type](

[id\_type] [int] IDENTITY(1,1) NOT NULL,

[hardware\_type] [varchar](50) NULL,

PRIMARY KEY CLUSTERED([id\_type])

)

GO

/\*\*\*\*\*\* Object: Table [dbo].[persona] Script Date: 30.11.2021 10:47:28 \*\*\*\*\*\*/

CREATE TABLE [dbo].[persona](

[persona\_id] [int] IDENTITY(1,1) NOT NULL,

[first\_name] [varchar](50) NULL,

[second\_name] [varchar](50) NULL,

[middle\_name] [varchar](50) NULL,

[phone\_number] [varchar](50) NULL,

[post] [int] NULL,

PRIMARY KEY CLUSTERED([persona\_id])

)

GO

/\*\*\*\*\*\* Object: Table [dbo].[post] Script Date: 30.11.2021 10:47:28 \*\*\*\*\*\*/

CREATE TABLE [dbo].[post](

[post\_id] [int] IDENTITY(1,1) NOT NULL,

[post\_name] [varchar](50) NULL,

PRIMARY KEY CLUSTERED([post\_id])

)

GO

/\*\*\*\*\*\* Object: Table [dbo].[sys\_admin] Script Date: 30.11.2021 10:47:28 \*\*\*\*\*\*/

CREATE TABLE [dbo].[sys\_admin](

[sys\_id] [int] IDENTITY(1,1) NOT NULL,

[persona\_sys] [int] NULL,

PRIMARY KEY CLUSTERED([sys\_id])

)

GO

/\*\*\*\*\*\* Object: Table [dbo].[teacher] Script Date: 30.11.2021 10:47:28 \*\*\*\*\*\*/

CREATE TABLE [dbo].[teacher](

[teacher\_id] [int] IDENTITY(1,1) NOT NULL,

[persona\_teacher] [int] NULL,

PRIMARY KEY CLUSTERED([teacher\_id])

)

GO

/\*\*\*\*\*\* Object: View [dbo].[hardwareViev] Script Date: 30.11.2021 10:47:28 \*\*\*\*\*\*/

CREATE VIEW [dbo].[hardwareView]

AS

SELECT dbo.hardware.hardware\_id, dbo.hardware\_type.hardware\_type, dbo.hardware.hardware\_status, dbo.hardware.date\_check, dbo.hardware.specification, dbo.hardware.class\_hardware

FROM dbo.hardware INNER JOIN

dbo.hardware\_type ON dbo.hardware.hardware\_type = dbo.hardware\_type.id\_type INNER JOIN

dbo.class ON dbo.hardware.class\_hardware = dbo.class.class\_id

GO

/\*\*\*\*\*\* Object: View [dbo].[PersonaView] Script Date: 30.11.2021 10:47:28 \*\*\*\*\*\*/

CREATE VIEW [dbo].[PersonaView]

AS

SELECT dbo.persona.persona\_id, dbo.persona.first\_name, dbo.persona.second\_name, dbo.persona.middle\_name, dbo.persona.phone\_number, dbo.post.post\_name

FROM dbo.persona INNER JOIN

dbo.post ON dbo.persona.post = dbo.post.post\_id

GO

/\*\*\*\*\*\* Object: View [dbo].[TeacherView] Script Date: 30.11.2021 10:47:28 \*\*\*\*\*\*/

CREATE VIEW [dbo].[TeacherView]

AS

SELECT dbo.teacher.teacher\_id, dbo.persona.first\_name, dbo.persona.second\_name, dbo.persona.middle\_name, dbo.persona.phone\_number, dbo.post.post\_name

FROM dbo.persona INNER JOIN

dbo.post ON dbo.persona.post = dbo.post.post\_id INNER JOIN

dbo.teacher ON dbo.persona.persona\_id = dbo.teacher.persona\_teacher

GO

/\*\*\*\*\*\* Object: View [dbo].[ClassView] Script Date: 30.11.2021 10:47:28 \*\*\*\*\*\*/

CREATE VIEW [dbo].[ClassView]

AS

SELECT dbo.class.class\_id, dbo.persona.first\_name, dbo.persona.second\_name, dbo.persona.middle\_name, dbo.persona.phone\_number, dbo.post.post\_name

FROM dbo.post LEFT OUTER JOIN

dbo.persona ON dbo.post.post\_id = dbo.persona.post LEFT OUTER JOIN

dbo.teacher ON dbo.persona.persona\_id = dbo.teacher.persona\_teacher RIGHT OUTER JOIN

dbo.class ON dbo.teacher.teacher\_id = dbo.class.teacher

GO

ALTER TABLE [dbo].[class] WITH CHECK ADD CONSTRAINT [FK\_class\_teacher] FOREIGN KEY([teacher])

REFERENCES [dbo].[teacher] ([teacher\_id]) ON DELETE CASCADE

GO

ALTER TABLE [dbo].[class] CHECK CONSTRAINT [FK\_class\_teacher]

GO

ALTER TABLE [dbo].[hardware] WITH CHECK ADD CONSTRAINT [FK\_hardware\_class] FOREIGN KEY([class\_hardware])

REFERENCES [dbo].[class] ([class\_id]) ON DELETE CASCADE

GO

ALTER TABLE [dbo].[hardware] CHECK CONSTRAINT [FK\_hardware\_class]

GO

ALTER TABLE [dbo].[hardware] WITH CHECK ADD CONSTRAINT [FK\_hardware\_hardware\_type] FOREIGN KEY([hardware\_type])

REFERENCES [dbo].[hardware\_type] ([id\_type]) ON DELETE CASCADE

GO

ALTER TABLE [dbo].[hardware] CHECK CONSTRAINT [FK\_hardware\_hardware\_type]

GO

ALTER TABLE [dbo].[persona] WITH CHECK ADD CONSTRAINT [FK\_persona\_post] FOREIGN KEY([post])

REFERENCES [dbo].[post] ([post\_id]) ON DELETE CASCADE

GO

ALTER TABLE [dbo].[persona] CHECK CONSTRAINT [FK\_persona\_post]

GO

ALTER TABLE [dbo].[sys\_admin] WITH CHECK ADD CONSTRAINT [FK\_sys\_admin\_persona] FOREIGN KEY([persona\_sys])

REFERENCES [dbo].[persona] ([persona\_id]) ON DELETE CASCADE

GO

ALTER TABLE [dbo].[sys\_admin] CHECK CONSTRAINT [FK\_sys\_admin\_persona]

GO

ALTER TABLE [dbo].[teacher] WITH CHECK ADD CONSTRAINT [FK\_teacher\_persona] FOREIGN KEY([persona\_teacher])

REFERENCES [dbo].[persona] ([persona\_id]) ON DELETE CASCADE

GO

ALTER TABLE [dbo].[teacher] CHECK CONSTRAINT [FK\_teacher\_persona]

GO

Процедуры

CREATE PROCEDURE AddHardware @h\_id int, @htype\_id int,

@h\_status nvarchar(80), @h\_spec text,

@h\_class nvarchar(80) AS BEGIN

DECLARE @h\_date date;

SET @h\_date = GETDATE();

BEGIN TRANSACTION;

IF NOT EXISTS(SELECT hardware\_type FROM hardware\_type WHERE hardware\_type.id\_type = @htype\_id) BEGIN

INSERT INTO hardware\_type (hardware\_type) VALUES ('unknown');

SELECT @htype\_id = IDENT\_CURRENT('hardware\_type');

END;

IF NOT EXISTS(SELECT class\_id FROM class WHERE class.class\_id = @h\_class)

INSERT INTO class VALUES(@h\_class, NULL);

INSERT INTO hardware VALUES(@h\_id, @htype\_id, @h\_status, GETDATE(), @h\_spec, @h\_class);

COMMIT TRANSACTION;

END;

go

CREATE PROCEDURE DeleteHardware @h\_id int AS BEGIN

DELETE hardware WHERE hardware\_id = @h\_id;

END;

go

CREATE PROCEDURE ShowHardwareReport AS BEGIN

SELECT TOP (1000) \*

FROM [hardware].[dbo].[hardwareView]

END;

go

CREATE PROCEDURE AddClassID @classid nvarchar(50) AS BEGIN

IF NOT EXISTS(SELECT class\_id FROM class WHERE class.class\_id = @classid)

INSERT INTO class VALUES(@classid, NULL);

END;

go

CREATE PROCEDURE DeleteClassID @classid nvarchar(50) AS BEGIN

DELETE class WHERE class\_id = @classid;

END;

go

CREATE PROCEDURE AddHardwareType @h\_type nvarchar(50) AS BEGIN

INSERT INTO hardware\_type(hardware\_type) VALUES(@h\_type);

END;

go

CREATE TRIGGER DeleteUnusedHardwareType ON hardware AFTER DELETE AS BEGIN

IF NOT EXISTS(SELECT h.hardware\_id FROM hardware AS h, deleted AS d

WHERE h.hardware\_id = d.hardware\_id)

DELETE FROM hardware\_type WHERE id\_type IN (SELECT hardware\_type FROM deleted);

END;

GO

CREATE USER employee WITHOUT LOGIN;

GRANT EXECUTE ON ShowHardwareReport TO employee

GO

CREATE USER administrator WITHOUT LOGIN

GRANT EXECUTE ON AddHardware TO administrator

GO

CREATE USER administrator WITHOUT LOGIN

GRANT EXECUTE ON ChangeHardwareSpec TO administrator

GO

CREATE USER administrator WITHOUT LOGIN

GRANT EXECUTE ON DeleteHardware TO administrator

GO

CREATE USER administrator WITHOUT LOGIN

GRANT EXECUTE ON ShowHardwareReport TO administrator

GO

CREATE USER administrator WITHOUT LOGIN

GRANT EXECUTE ON AddClassID TO administrator

GO

CREATE USER administrator WITHOUT LOGIN

GRANT EXECUTE ON DeleteClassID TO administrator

GO

CREATE USER administrator WITHOUT LOGIN

GRANT EXECUTE ON AddHardwareType TO administrator