Содержание

Введение…………………………………………………………………………...5

1.Техническое задание……………………………………………………………6

1.1 Анализ предметной области……………………………………………….…6

1.1.1 Общие сведения………………………………………………………….….6

1.1.2 Документы, на основании которых создается система…………………...6

1.1.3 Назначение и цели создания системы……………………………………...6

1.1.4 Требования к системе……………………………………………………….6

1.2 Постановка задачи…………………………………………………………….8

2. Технический проект информационной системы……………………………..9

2.1 Функциональная модель……………………………………………………...9

2.1.1 Контекстная диаграмма и диаграммы детализации процессов…………..9

2.1.2 Диаграмма верхнего уровня……………………………………………….15

2.2 Информационная модель……………………………………………………16

2.2.1 Идентификация сущностей и связей. ER-диаграмма логического уровня. …………………………………………………………………………………….16

2.2.2 ER-диаграмма физического уровня. Ограничения ссылочной целостности………………………………………………………… ……………………..19

2.3 Верификация спроектированной логической модели…………………….20

3. Реализация системы…………………………………………………………..25

3.1 SQL-определения регламентированных запросов………………………...25

3.2 Описание клиентского приложения………………………………………..34

4. Результаты тестирования информационной системы……………………...47

Заключение……………………………………………………………………….48

Список использованных литературных источников…………………………..49

Приложения……………………………………………………………………...50

Приложение 1…………………………………………………………………….50

Приложение 2…………………………………………………………………….51

Приложение 3…………………………………………………………………….52

Введение

Основой для учета, контроля и планирования служат всевозможные картотеки, регистрационные журналы, списки и т.д. Они постепенно накапливаются и обновляются. При большом объеме информации поиск и обобщение необходимых сведений, осуществляемых вручную, представляют собой довольно трудоемкий процесс.

Сейчас же, в век компьютерных технологий и больших прогрессов в этой сфере, на смену картотекам пришли очень удобные, надёжные, быстродействующие компьютерные системы. Для решения задач с использованием компьютерных систем используется специальное программное обеспечение, часто объединенное в крупные информационно-справочные системы.

Первоначально для накопления и хранения информации на ЭВМ применялись локальные массивы (или файлы), при этом для каждой из решаемых функциональных задач создавались собственные файлы исходной и результатной информации. Это приводило к значительному дублированию данных, усложняло их обновление, затрудняло решение взаимосвязанных проблемных задач.

Постепенно с развитием программного обеспечения ЭВМ появились идеи создания управляющих систем, которые позволяли бы накапливать, хранить и обновлять взаимосвязанные данные по целому комплексу решаемых задач, например при автоматизации бухгалтерского учета, на предприятии. Эти идеи нашли свое воплощение в системах управления базами данных (СУБД). СУБД взаимодействуют не с локальными, а взаимосвязанными по информации массивами, называемыми базами данных. С появлением персональных компьютеров СУБД становятся наиболее популярным средством обработки табличной информации.

Одной из таких информационно-справочных систем является данная система учета учащихся. Она позволяет автоматизировать многие процессы, связанные с менеджментом задач. Эта программа должна облегчить работу сотрудников и решить большинство проблем, вызванных недостатками бумажных носителей.

1.Техническое задание

## 1.1 Анализ предметной области.

### 1.1.1 Общие сведения

Объект автоматизации – любая средняя школа.

### 1.1.2 Документы, на основании которых создается система

* Информация об учащихся;
* Информация о преподавателях;

### 1.1.3 Назначение и цели создания системы

1.1.3.1 Назначение системы

Проектирование многопользовательской системы управления задачами.

Систему предполагается использовать в средних школах для упрощения взаимодействия и ускорения рабочих процессов.

Система позволит облегчить взаимодействие сотрудников, так как всё взаимодействие сотрудников сведётся к управлению задачами в проектируемой системе.

1.1.3.2 Цели создания системы

Систему предполагается создать для улучшения качества работы организации, в результате упрощения взаимодействия и повышения скорости работы сотрудников.

Критерий оценки достижений целей системы – снижения затрат по времени на взаимодействие и коммуникацию в организации. Так как система позволит увеличить скорость обмена информацией, то снизятся затраты по времени на коммуникацию.

### 1.1.4 Требования к системе

1.1.4.1 Краткие сведения

Школа - учебное заведение, государственное или частное, дающее общее образование, то есть совокупность знаний о природе, обществе и познании, и соответствующие умения и навыки, необходимые человеку в жизни независимо от профессии.

Время в школе разделяется на следующие этапы: внеурочная деятельность, перемена, урок. Урок и перемена, как правило, сменяют друг друга несколько раз. Завершается же учебный день обычно внеурочной деятельностью.

Урок - занятие школьников по предметам.

Перемена - отдых между уроками.

Внеурочная деятельность – это такая система дополнительного образования, при которой ученик располагается в школе после уроков (и ему/ей обеспечивается питание, а также дается возможность для подготовки своих уроков).

В школах существует такие понятия как отметка и оценка. Отметки в школах выставляются по десятибалльной системе (и имеют значение от 1 до 10). Отметки 1 и 2 являются неудовлетворительной оценкой. Отметки 3,4,5 является удовлетворительной, но недостаточно высокой. Отметки 6,7,8 – хорошая оценка, показывающая хорошие знания. Отметки 9,10 - является отличной оценкой, лучшей возможной.

1.1.4.2 Требования к системе в целом

Система должна удовлетворять следующим требованиям:

* Надежности;
* Безопасности;
* Требования к защите информации от несанкционированного доступа. Должна быть предусмотрена защита от несанкционированного доступа к данным, ввода данных, их удаления;
* Информация, хранящаяся в системе, должна быть защищена от аварийных ситуаций, влияния внешних воздействий;
* Квалификация персонала. Персонал должен быть обучен правилам работы с системой, при этом не обязательно наличие специального технического образования.

1.1.4.3 Требования к функциям, выполняемые системой

Функции, выполняемые подсистемами объекта автоматизации:

* Предоставление полной информации об учащихся. Включая дату рождения и ФИО;
* Предоставление полной информации о преподавателях. Включая номер телефона;
* Предоставление полной информации о всех полученных оценках;
* Предоставление полной информации об опекунах каждого учащегося;
* Предоставление полной информации о средней успеваемости по каждому предмету;
* Предоставление полной информации о средней успеваемости по каждому классу;
* Предоставление полной информации о составе каждого класса;

1.2 Постановка задачи

При работе с бумажной документацией неизбежны огромные затраты времени, ошибки в ведении документации, а также неэффективность использования человеческих ресурсов. Поэтому внедрение автоматизированной системы данных является актуальным и выгодным направлением.

Создание базы данных позволит автоматизировать процесс работы организации и упростит задачи менеджмента и контроля задач.

2. Технический проект информационной системы

2.1 Функциональная модель

2.1.1 Контекстная диаграмма и диаграммы детализации процессов

Для разработки функциональной модели информационной системы для выбранной предметной области используем методологию BPMN2.0 и CASE-средства концептуального проектирования функциональных моделей информационных систем Sparx Systems Enterprise Architect. В контекстные диаграммы входит описание цели моделирования, области (описания того, что будет рассматриваться в качестве компонента системы, а что в качестве внешнего воздействия) и точки зрения (позиции, с которой будет строиться модель).

Вначале разрабатывают модель верхнего уровня процессов, отображающую подпроцессы в ИС (рисунок 2.1), затем проводят декомпозицию подпроцессов – строят модели, отображающие детали подпроцессов.

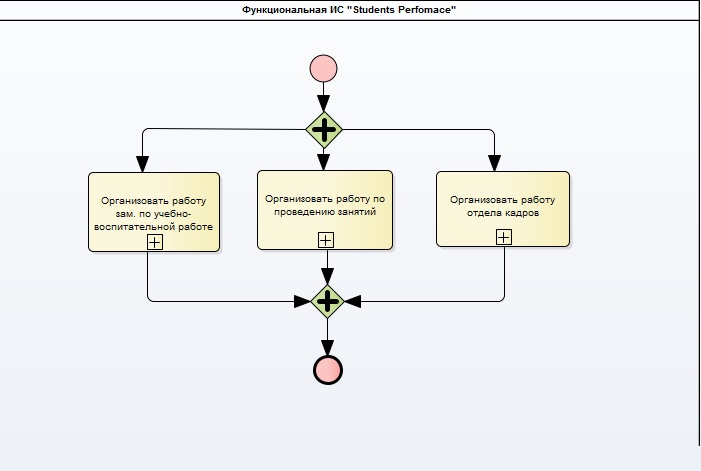


Рисунок 2.1 - Диаграмма подпроцессов функциональной модели (первый уровень)

Процесс начинается с определенного (начального) события, состоит из всех действий, которые следует выполнить для достижения цели, и завершается достижением цели или запуском других процессов. Процесс следует понимать, как сквозной набор работ (например, «от поступления заявки до оплаты»), пересекающий всю систему, чтобы выполнить поставленную цель и доставить ценность потребителю.

Подпроцессы других уровней изображены на рисунках (рисунок 2.2-2.7).

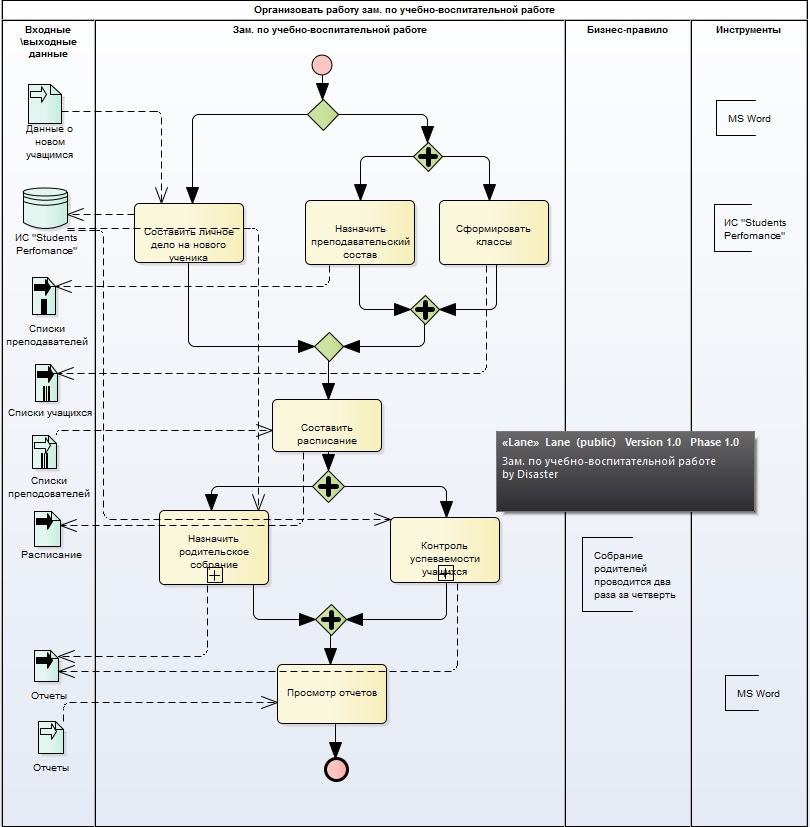


Рисунок 2.2 - Диаграмма декомпозиции подпроцесса «Организовать работу зам. по учебно-воспитательной работе» (второй уровень)

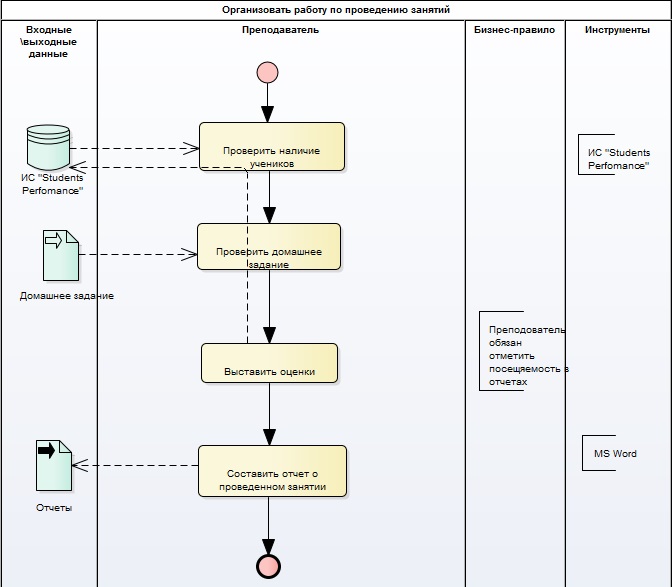


Рисунок 2.3 - Диаграмма декомпозиции подпроцесса «Организовать работу по проведению занятий» (второй уровень)

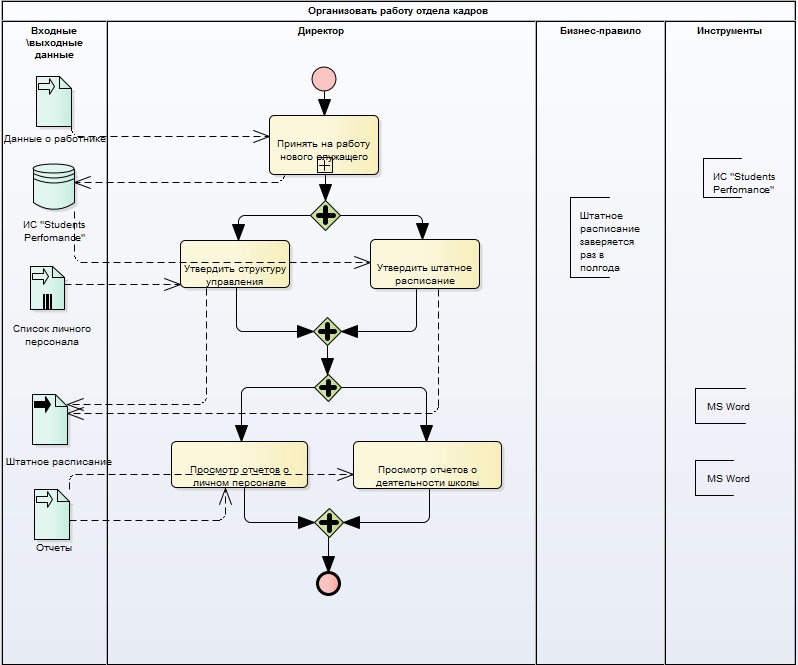


Рисунок 2.4 - Диаграмма декомпозиции подпроцесса «Организовать работу отдела кадров» (второй уровень)

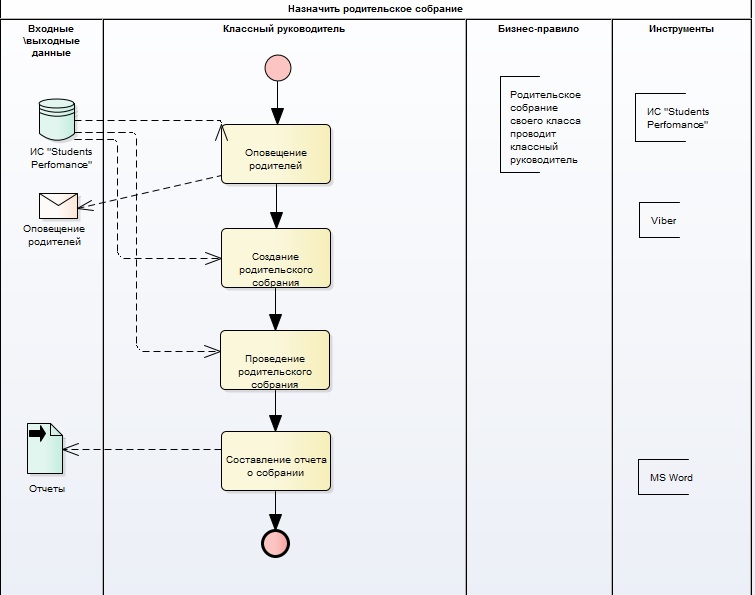


Рисунок 2.5 - Диаграмма декомпозиции подпроцесса «Назначить родительское собрание» (третий уровень)

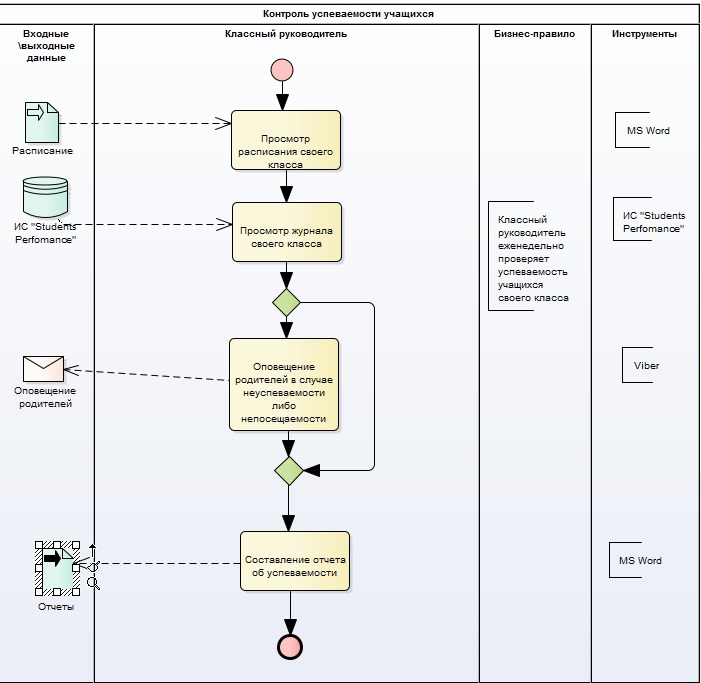


Рисунок 2.6 - Диаграмма декомпозиции подпроцесса «Контроль успеваемости учащихся» (третий уровень)

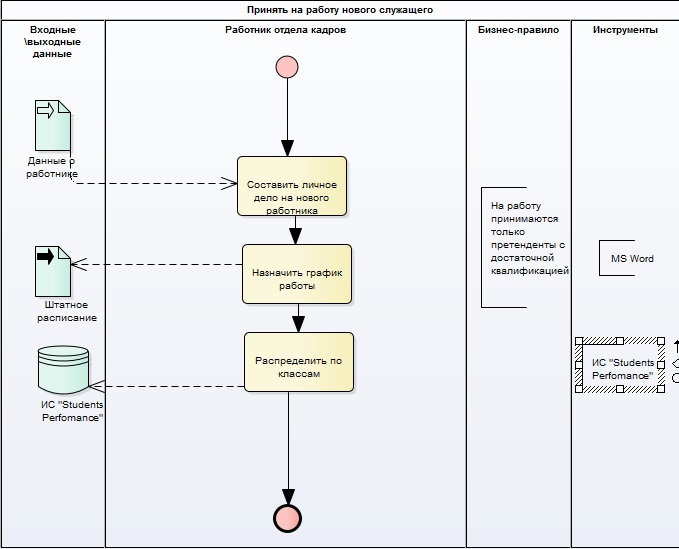


Рисунок 2.7 - Диаграмма декомпозиции подпроцесса «Принять на работу нового служащего» (третий уровень)

2.1.2 Диаграмма верхнего уровня

Для разработки информационной модели использовалось приложение Sparx Systems Enterprise Architect. Диаграммой верхнего уровня представлена в Приложении 1 (рисунок 2.1).

2.2 Информационная модель

2.2.1 Идентификация сущностей и связей. ER-диаграмма логического уровня

При помощи Sparx Systems Enterprise Architect также можно создать диаграмму логического уровня. Логический уровень – это абстрактный взгляд на данные. Объекты модели, представляемые на нем, называются сущностями и атрибутами. Логическая модель данных является универсальной, т.к. не зависит от конкретной СУБД.

Для отображения информационной модели рассматриваемого процесса на логической модели используются следующие сущности:

* Students - для хранения данных об учащихся: номер, имя, отчество, фамилия, дата рождения, адрес, номер класса;
* Guardians – для хранения данных об опекунах: номер, ФИО, номер телефона;
* Teachers – для хранения данных об учителях: номер, имя, отчество, фамилия, телефон, номер предмета;
* HeadTeachers – для хранения данных о завуч: номер, имя, отчество, фамилия, телефон;
* HeadMasters – для хранения данных о директоре: номер, имя, отчество, фамилия, телефон;
* Users – для хранения данных о всех пользователях: номер, логин, пароль, номер роли;
* Roles – для хранения данных о допустимых в системе ролях: номер, название роли;
* Classes – для хранения данных о классах: номер, название класса, номер классного руководителя;
* Subjects – для хранения данных о предметах: номер, название предмета;

На логическом уровне проектирования в моделируемой базе данных присутствуют следующие типы связей между описанными сущностями:

* неиденцифицирующие связи;
* иденцифицирующие связи;
* связи многие-ко-многим;

Связь между сущностями Users и Students обязательная, идентифицирующая. Связь 1:1, т.к. для одного учащегося существует только одна запись;

Связь между сущностями Users и Teachers обязательная, идентифицирующая. Связь 1:1, т.к. для одного учителя существует только одна запись;

Связь между сущностями Users и HeadTeachers обязательная, идентифицирующая. Связь 1:1, т.к. для одного завуч существует только одна запись;

Связь между сущностями Users и HeadMasters обязательная, идентифицирующая. Связь 1:1, т.к. для одного директора существует только одна запись;

Связь между сущностями Roles и Users обязательная, неидентифицирующая. Связь 1:M, т.к. одна роль может быть у многих пользователей;

Связь между сущностями Classes и Students обязательная, неидентифицирующая. Связь 1:M, т.к. в одном классе могут учится много учащихся;

Связь между сущностями Teachers и Classes обязательная, неидентифицирующая. Связь 1:М, т.к. один предователь может быть во многих классах;

Связь между сущностями Subjects и Teachers обязательная, неидентифицирующая. Связь 1:М, т.к. один предмет могут преподавать многие преподаватели;

Связь между сущностями Students и Subjects M:M, т.к. многие учащиеся могут присутствовать на многих предметах;

Связь между сущностями Guardians и Students M:M, т.к. у многих учащихся может бы несколько опекунов;

Для разрешения связей М:М были введены дополнительные сущности: Marks и Guardians\_Students;

ER-диаграмма логического уровня представлена в Приложении 2 (рисунок 2.8)

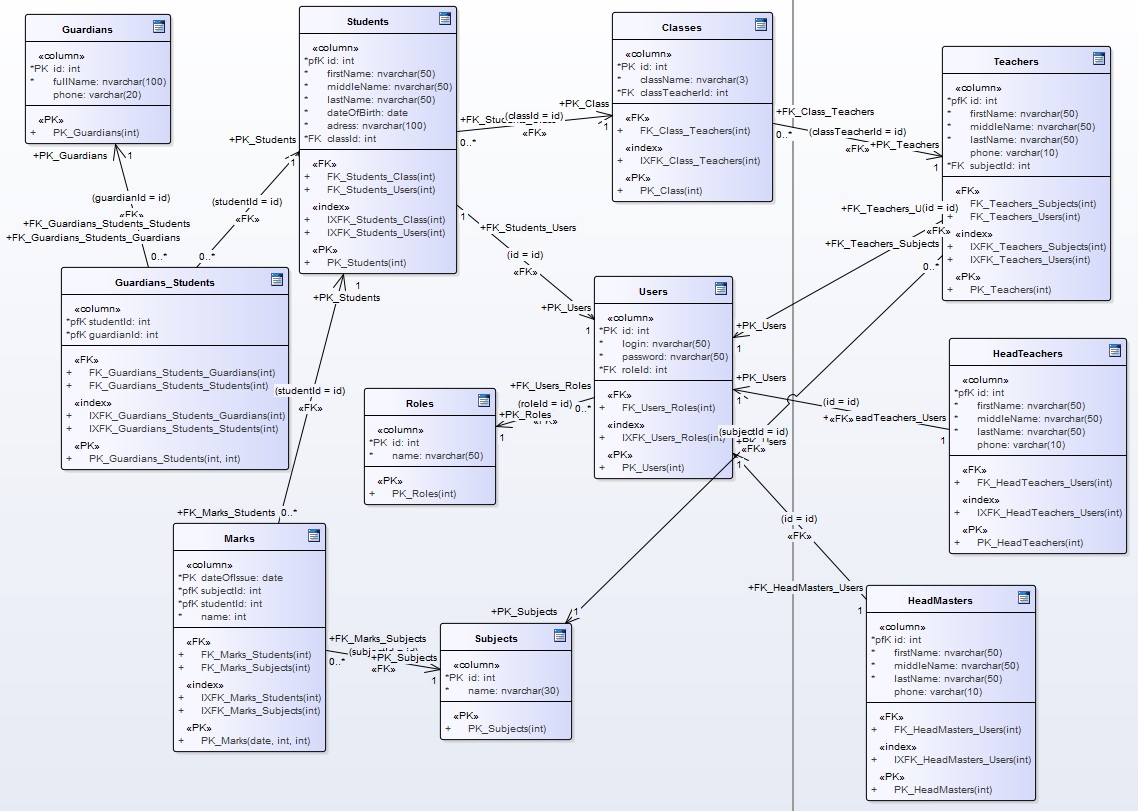


Рисунок 2.8 – ER-диаграмма логического уровня

2.2.2 ER-диаграмма физического уровня. Ограничение ссылочной ценности

Физическая модель данных зависит от конкретной СУБД. В ней содержится информация обо всех объектах БД. Одной и той же логической модели может соответствовать несколько разных физических.

Ссылочная целостность реализована с учетом следующих ограничений предметной области:

* При удалении пользователя в таблице Users, удаляется и учащийся с таким же номером из таблицы Students: каскадное удаление;
* При удалении пользователя в таблице Users, удаляется и учащийся с таким же номером из таблицы Teachers: каскадное удаление;
* При удалении пользователя в таблице Users, удаляется и учащийся с таким же номером из таблицы HeadTeachers: каскадное удаление;
* При удалении пользователя в таблице Users, удаляется и учащийся с таким же номером из таблицы HeadMasters: каскадное удаление;
* В таблице Classes название класса должно быть уникальным (UNIQUE);
* В таблице Users логин должен быть уникальным (UNIQUE);
* В таблице Roles название роли должно быть уникальным (UNIQUE);
* В таблице Subjects название предмета должно быть уникальным (UNIQUE);

ER-диаграмма физического уровня представлена в Приложении 3 (рисунок 2.9)

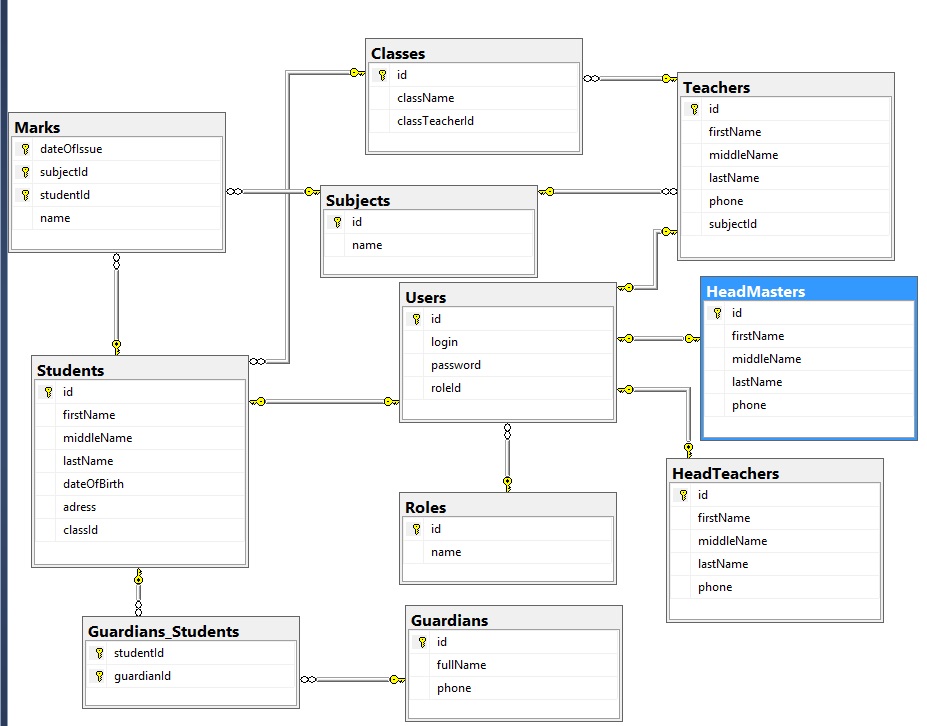


Рисунок 2.9 – ER-диаграмма физического уровня

2.3 Верификация спроектированной логической модели

После разработки информационной модели ее следует связать с функциональной моделью. Такая связь гарантирует завершенность анализа, обеспечивает наличие источников данных (сущностей) для всех процессов, подпроцессов и действий.

На данном этапе производится сопоставление действий в модели BPMN с объектами модели данных (сущностями и атрибутами). При этом нужно учитывать, что действие в функциональной модели процессов может быть связано с несколькими атрибутами различных сущностей. Выполненное сопоставление функциональной и информационной моделей оформляется отчетом о верификации моделей (таблица 5.1), который позволяет удостоверить, что в функциональной модели отсутствуют действия, не нашедшие свое отражение

в базе данных, а в информационной модели нет сущностей и атрибутов, не связанных ни с какими действиями.

Таблица 1 – Отчет о верификации модели

|  |  |  |
| --- | --- | --- |
| **Имя действия** | **Имя сущности** | **Имя атрибута** |
| Составить личное дело на нового ученика | Students | id |
| firstName |
| middleName |
| lastName |
| dateOfBirth |
| adress |
| classId |
| Users | id |
| login |
| password |
| roleId |
| Roles | id |
| name |
| Назначить родительское собрание | Guardians | id |
| fullName |
| phone |
| Назначить преподавательский cостав | Teachers | id |
| firstName |
| middleName |
| lastName |
| phone |
| subjectId |
| Сформировать классы | Classes | id |
| name |
| classTeacherId |
| Составить расписание | Classes | id |
| name |
| classTeacherId |
| Teachers | id |
| firstName |
| middleName |
| lastName |
| phone |
| subjectId |

Продолжение таблицы 1 – Отчет верификации модели

|  |  |  |
| --- | --- | --- |
| **Имя действия** | **Имя сущности** | **Имя атрибута** |
| Составить расписание | Subjects | id |
| name |
| Просмотр отчетов | Students | id |
| firstName |
| middleName |
| lastName |
| dateOfBirth |
| adress |
| Проверить наличие учеников | Students | id |
| firstName |
| middleName |
| lastName |
| Выставить оценки | Students | id |
| firstName |
| middleName |
| lastName |
| Subjects | id |
| name |
| Составить отчет о проведенном занятии | Students | id |
| firstName |
| middleName |
| lastName |
| Просмотр расписания своего класса | Classes | id |
| name |
| classTeacherId |
| Subjects | id |
| name |
| Просмотр журнала своего класса | Classes | id |
| name |
| classTeacherId |
| Subjects | id |
| name |
| Оповещение родителей | Guardians | id |
| fullName |
| phone |

Продолжение таблицы 1 – Отчет верификации модели

|  |  |  |
| --- | --- | --- |
| **Имя действия** | **Имя сущности** | **Имя атрибута** |
| Составление отчета об успеваемости | Students | id |
| firstName |
| middleName |
| lastName |
| dateOfBirth |
| Subjects | id |
| name |
| Составить личное дело на нового ученика | HeadTeachers | id |
| firstName |
| lastName |
| middleName |
| Утвердить структуру управления | HeadTeachers | id |
| firstName |
| lastName |
| middleName |
| phone |
| Утвердить штатное расписание | Teachers | id |
| firstName |
| middleName |
| lastName |
| Classes | id |
| name |
| classTeacherId |
| Subjects | id |
| name |
| Просмотр отчетов о личном персонале | HeadMasters | id |
| firstName |
| lastName |
| middleName |
| Teachers | id |
| firstName |
| middleName |
| lastName |

Продолжение таблицы 1 – Отчет верификации модели

|  |  |  |
| --- | --- | --- |
| **Имя действия** | **Имя сущности** | **Имя атрибута** |
| Просмотр отчетов о личном персонале | Students | id |
| firstName |
| middleName |
| lastName |
| Просмотр отчетов о деятельности школы | HeadMasters | id |
| firstName |
| lastName |
| middleName |
| Teachers | id |
| firstName |
| middleName |
| lastName |
| Students | id |
| firstName |
| middleName |
| lastName |
| Составить личное дело на нового работника | Teachers | id |
| firstName |
| middleName |
| lastName |
| phone |
| Назначить график работы | Teachers | id |
| firstName |
| middleName |
| lastName |
| Classes | id |
| name |
| classTeacherId |
| Распределить по классам | Teachers | id |
| firstName |
| middleName |
| lastName |
| Classes | id |
| name |
| classTeacherId |

3. Реализация системы

3.1. SQL-определения регламентированных запросов

Физически представление реализовано в виде SQL-запроса, на основе которого производится выборка из одной или нескольких таблиц или представлений.

/\*Средний балл по классам\*/

ALTER PROCEDURE [dbo].[sp\_AVG\_AverangMarkClass]

AS

BEGIN

SET NOCOUNT ON;

SELECT

A.className 'Класс',

AVG(A.name) 'Средний балл'

FROM (SELECT

Classes.className,

Marks.name

FROM (Marks INNER JOIN Subjects ON Marks.subjectId = Subjects.id)

INNER JOIN (Students INNER JOIN Classes ON Students.classId = Classes.id)

ON Students.id = Marks.studentId) AS A

GROUP BY A.className

END

/\*Средний балл по предмету\*/

ALTER PROCEDURE [dbo].[spAVG\_AverangMarkSubject]

AS

BEGIN

SET NOCOUNT ON;

SELECT

B.name 'Предмет',

C 'Средний балл'

FROM (SELECT Marks.subjectId, AVG(Marks.name) AS C FROM Marks

GROUP BY subjectId) AS A INNER JOIN (SELECT DISTINCT Subjects.name, Subjects.id

FROM Subjects INNER JOIN Marks ON Subjects.id = Marks.subjectId) AS B ON A.subjectId = B.id

END

/\*Вывод всех родители\*/

ALTER PROCEDURE [dbo].[spGuardians\_GetAllGuardians]

AS

BEGIN

SET NOCOUNT ON;

SELECT \*

FROM Guardians

ORDER BY Guardians.fullName

END

/\*Добавление опекуна\*/

ALTER PROCEDURE [dbo].[spGuardians\_InsertGuardian]

@studentId int,

@guardianFullName nvarchar(100),

@phone varchar(20)

AS

BEGIN

SET NOCOUNT ON;

DECLARE @guardianId int

INSERT INTO Guardians

VALUES (@guardianFullName, @phone)

SET @guardianId = SCOPE\_IDENTITY()

INSERT INTO Guardians\_Students

VALUES (@studentId, @guardianId)

END

/\*Поиск опекунов для выбранного учащегося\*/

ALTER PROCEDURE [dbo].[spGuardiansStudent\_GetGuardians]

@studentId int

AS

BEGIN

SET NOCOUNT ON;

SELECT Guardians.FullName 'ФИО', Guardians.Phone 'Телефон', Students.Adress 'Адрес'

FROM Students INNER JOIN (Guardians INNER JOIN Guardians\_Students ON Guardians.[Id] = Guardians\_Students.[GuardianId])

ON Students.[Id] = Guardians\_Students.[StudentId]

WHERE Students.id = @studentId

END

/\*Удаление оценки\*/

ALTER PROCEDURE [dbo].[spMarks\_DeleteMark]

@dateOfIssue date,

@studentId int,

@subjectId int

AS

BEGIN

SET NOCOUNT ON;

DELETE FROM Marks

WHERE Marks.dateOfIssue = @dateOfIssue AND Marks.studentId = @studentId AND Marks.subjectId = @subjectId

END

/\*Добавление оценки\*/

ALTER PROCEDURE [dbo].[spMarks\_InsertMark]

@dateOfIssue date,

@subjectId int,

@studentId int,

@markName int

AS

BEGIN

SET NOCOUNT ON;

INSERT INTO Marks

VALUES (@dateOfIssue, @subjectId, @studentId, @markName)

END

/\*Удаление учащегося\*/

ALTER PROCEDURE [dbo].[spStudents\_DeleteStudent]

@studentId int

AS

BEGIN

SET NOCOUNT ON;

DELETE Users

WHERE Users.id = @studentId

END

/\*Список всех учащихся\*/

ALTER PROCEDURE [dbo].[spStudents\_GetAllStudents]

AS

BEGIN

SET NOCOUNT ON;

SELECT Students.id,

Students.lastName 'Фамилия',

Students.firstName 'Имя',

Students.middleName 'Отчество',

Students.dateOfBirth 'Дата рождения',

Students.adress 'Адрес',

Classes.className 'Класс'

FROM Students INNER JOIN Classes ON Students.classId = Classes.id

END

/\*Поиск учащегося по номеру\*/

ALTER PROCEDURE [dbo].[spStudents\_GetStudent]

@studentId int

AS

BEGIN

SET NOCOUNT ON;

SELECT

Students.lastName 'Фамилия',

Students.firstName 'Имя',

Classes.className 'Класс'

FROM Students INNER JOIN Classes ON Students.classId = Classes.id

WHERE Students.id = @studentId

END

/\*Добавление учащегося\*/

ALTER PROCEDURE [dbo].[spStudents\_InsertStudent]

@firstName nvarchar(50),

@middleName nvarchar(50),

@lastName nvarchar(50),

@dateOfBirth date,

@adress nvarchar (100),

@classId int,

@password nvarchar(50)

AS

BEGIN

SET NOCOUNT ON;

DECLARE @id int

SET @id = IDENT\_CURRENT('Users') + 1

INSERT INTO Users

VALUES(CONVERT(nvarchar, @id), @password, 4)

SET @id = IDENT\_CURRENT('Users')

INSERT INTO Students

VALUES (@id, @firstName, @middleName, @lastName, @dateOfBirth, @adress, @classId)

END

/\*Изменение данных учащегося\*/

ALTER PROCEDURE [dbo].[spStudents\_UpdateStudent]

@studentId int,

@firstName nvarchar(50),

@middleName nvarchar(50),

@lastName nvarchar(50),

@adress nvarchar(100),

@dateOfBirth date,

@classId int

AS

BEGIN

SET NOCOUNT ON;

UPDATE Students

SET

firstName = @firstName,

middleName = @middleName,

lastName = @lastName,

adress = @adress,

dateOfBirth = @dateOfBirth,

classId = @classId

WHERE Students.id = @studentId

END

/\*Поиск учащегося по номеру класса\*/

ALTER PROCEDURE [dbo].[spStudentsClasses\_GetStudentsOfClass]

@classId int

AS

BEGIN

SET NOCOUNT ON;

SELECT Students.id,

Students.lastName 'Фамилия',

Students.firstName 'Имя'

FROM Students INNER JOIN Classes ON Students.classId = Classes.id

WHERE Classes.id = @classId

END

/\*Вывод оценок по каждому предмету для учащегося\*/

ALTER PROCEDURE [dbo].[spStudentsMarks\_GetAllMarksOfStudent]

@studentId int

AS

BEGIN

SET NOCOUNT ON;

SELECT

Subjects.name 'Предмет',

Marks.name 'Оценка',

Marks.dateOfIssue 'Дата выставления'

FROM Students INNER JOIN (Marks INNER JOIN Subjects ON Marks.subjectId = Subjects.id) ON Students.id = Marks.studentId

WHERE Students.id = @studentId

ORDER BY Subjects.name

END

/\*Вывод оценок учащегося по всем предметам\*/

ALTER PROCEDURE [dbo].[spStudentsMarks\_GetMarksForAllSubjects]

@classId int

AS

BEGIN

SET NOCOUNT ON;

SELECT

Students.id,

Students.lastName 'Фамилия',

Students.firstName 'Имя',

Marks.name 'Оценка',

Marks.dateOfIssue 'Дата выставления',

Subjects.name 'Предмет'

FROM (Marks INNER JOIN Subjects ON Marks.subjectId = Subjects.id)

INNER JOIN (Students INNER JOIN Classes ON Students.classId = Classes.id)

ON Students.id = Marks.studentId

WHERE Classes.id = @classId

ORDER BY Students.lastName, Students.firstName

END

/\*Вывод оценок учащегося по номеру предмета и классу\*/

ALTER PROCEDURE [dbo].[spStudentsMarks\_GetMarksForSubject]

@subjectId int,

@classId int

AS

BEGIN

SET NOCOUNT ON;

SELECT

Students.id,

Students.lastName 'Фамилия',

Students.firstName 'Имя',

Marks.name 'Оценка',

Marks.dateOfIssue 'Дата выставления',

Subjects.id 'SubjectId'

FROM (Marks INNER JOIN Subjects ON Marks.subjectId = Subjects.id)

INNER JOIN (Students INNER JOIN Classes ON Students.classId = Classes.id)

ON Students.id = Marks.studentId

WHERE Subjects.id = @subjectId AND Classes.id = @classId

ORDER BY Students.lastName, Students.firstName

END

/\*Удаление преподавателя\*/

ALTER PROCEDURE [dbo].[spTeacher\_DeleteTeacher]

@teacherId int

AS

BEGIN

SET NOCOUNT ON;

DELETE Users

WHERE Users.id = @teacherId

END

/\*Добавление нового преподавателя\*/

ALTER PROCEDURE [dbo].[spTeacher\_InsertTeacher]

@firstName nvarchar(50),

@middleName nvarchar(50),

@lastName nvarchar(50),

@phone varchar(10),

@subjectId int,

@password nvarchar(50),

@roleId int

AS

BEGIN

SET NOCOUNT ON;

DECLARE @id int

SET @id = IDENT\_CURRENT('Users') + 1

INSERT INTO Users

VALUES(CONVERT(nvarchar, @id), @password, @roleId)

SET @id = IDENT\_CURRENT('Users')

INSERT INTO Teachers

VALUES(@id, @firstName, @middleName, @lastName, @phone, @subjectId)

END

/\*Изменение данных преподавателя\*/

ALTER PROCEDURE [dbo].[spTeacher\_UpdateTeacher]

@teacherId int,

@firstName nvarchar(50),

@middleName nvarchar(50),

@lastName nvarchar(50),

@phone varchar(10),

@roleId int,

@subjectId int

AS

BEGIN

SET NOCOUNT ON;

UPDATE Teachers

SET

firstName = @firstName,

middleName = @middleName,

lastName = @lastName,

phone = @phone,

subjectId = @subjectId

WHERE Teachers.id = @teacherId

UPDATE Users

SET roleId = @roleId

WHERE Users.id = @teacherId

END

/\*Вывод данных всех преподавателей\*/

ALTER PROCEDURE [dbo].[spTeachers\_GetAllTeachers]

AS

BEGIN

SET NOCOUNT ON;

SELECT Teachers.id,

Teachers.lastName 'Фамилия',

Teachers.firstName 'Имя',

Teachers.middleName 'Отчество',

Subjects.name 'Предмет',

Teachers.phone 'Телефон'

FROM Teachers INNER JOIN Subjects ON Teachers.subjectId = Subjects.id

END

/\*Поиск предмета по номеру преподавателя\*/

ALTER PROCEDURE [dbo].[spTeachersSubjects\_GetSubjectForTeacher]

@teacherId int

AS

BEGIN

SET NOCOUNT ON;

SELECT \*

FROM Teachers INNER JOIN Subjects ON Teachers.subjectId = Subjects.id

WHERE Teachers.id = @teacherId

END

/\*Поиск пользователей по логину и паролю\*/

ALTER PROCEDURE [dbo].[spUsers\_GetLoginAndPassword]

@login nvarchar(50),

@password nvarchar(50)

AS

BEGIN

SET NOCOUNT ON;

SELECT \*

FROM Users

WHERE Users.login = @login and Users.password = @password

END

3.2. Описание клиентского приложения

При разработке клиентского приложения были созданы следующие формы.

При запуске приложения появляется форма «Вход в систему» (рисунок 3.1). На форме присутствует кнопка «Войти», нажатие на которую открывает другие формы в зависимости от роли пользователя. Существует 5 основных форм: «Персонал», «Завуч», «Журнал классного руководителя», «Журнал преподавателя», «Оценки учащегося».

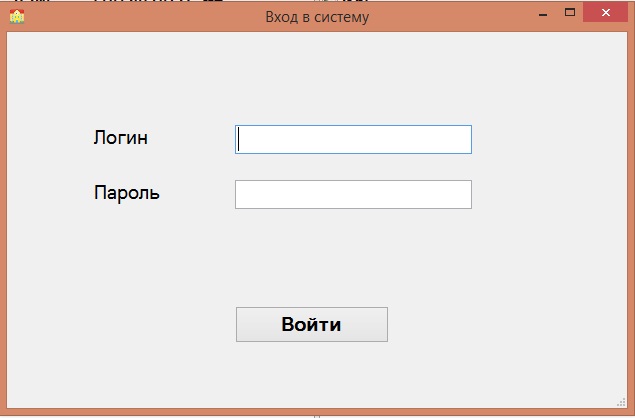


Рисунок 3.1 – Форма «Вход в систему»

После проверки правильности логина и пароля осуществляется вход в систему.

Рассмотрим вход для пользователя «Директор» (открытие формы «Персонал» (рисунок 3.2)). На этой форме имеется список всех преподавателей, а также кнопки: «Добавить», «Изменить», «Удалить», а также кол-во имеющегося персонала.

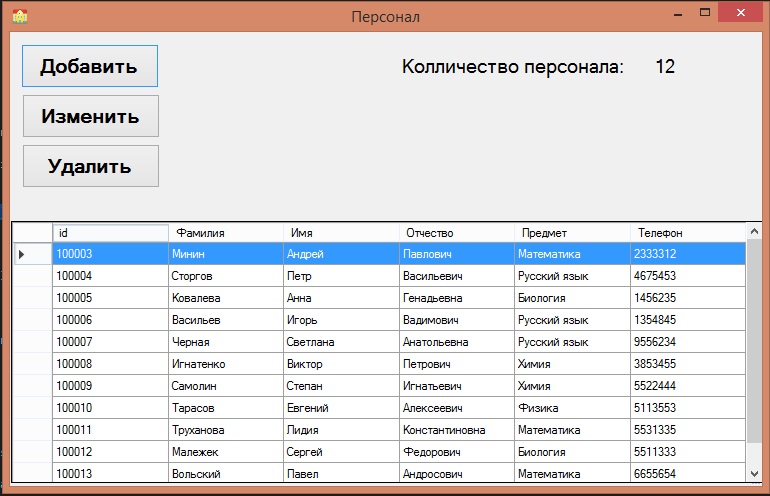


Рисунок 3.2 – Форма «Персонал»

При нажатии кнопки «Добавить» вызывается форма «Добавить преподавателя» (рисунок 3.3). На этой форме имеются поля для ввода данных преподавателя, а также возможность добавления нового предмета. Выбор предмета и должность можно выбрать из выпадающего меню. При нажатии на ссылку «Добавить предмет» вызывается форма «Добавить новый предмет» (рисунок 3.4).

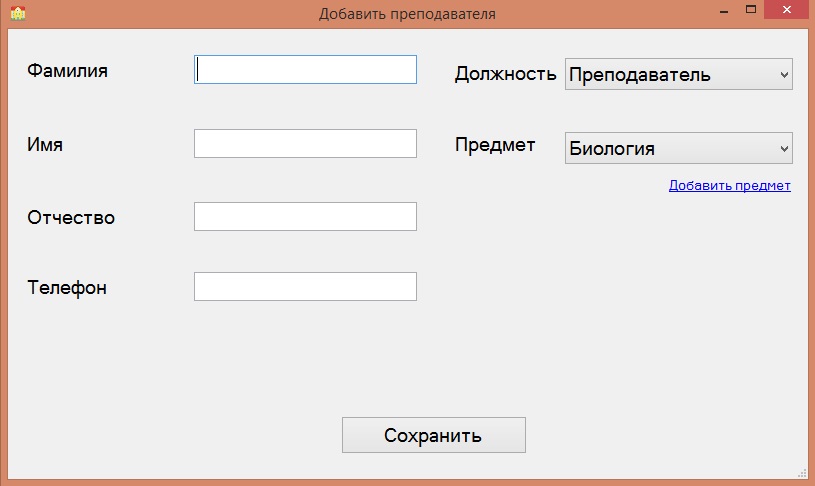


Рисунок 3.3 – Форма «Добавить преподавателя»

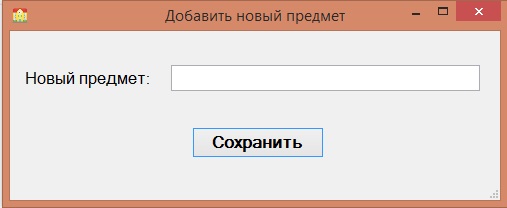


Рисунок 3.4 – Форма «Добавить новый предмет»

После корректного ввода названия нового предмета и нажатия кнопки «Сохранить» на форме «Добавить новый предмет» (рисунок 3.4), новый предмет добавляется в список существующих предметов базы данных.

После корректного ввода данных нового преподавателя и нажатия кнопки «Сохранить» на форме «Добавить преподавателя» (рисунок 3.3), данные добавляются в список существующих преподавателей, а также в список пользователей.

При нажатии кнопки «Изменить» на форме «Персонал» (рисунок 3.2) вызывается форма «Изменить данные преподавателя» (рисунок 3.5). На этой форме поля заполняются автоматически в зависимости от выбранного в списке преподавателя. После изменения данных и нажатия кнопки «Сохранить», измененные данные также изменять и в базе данных.

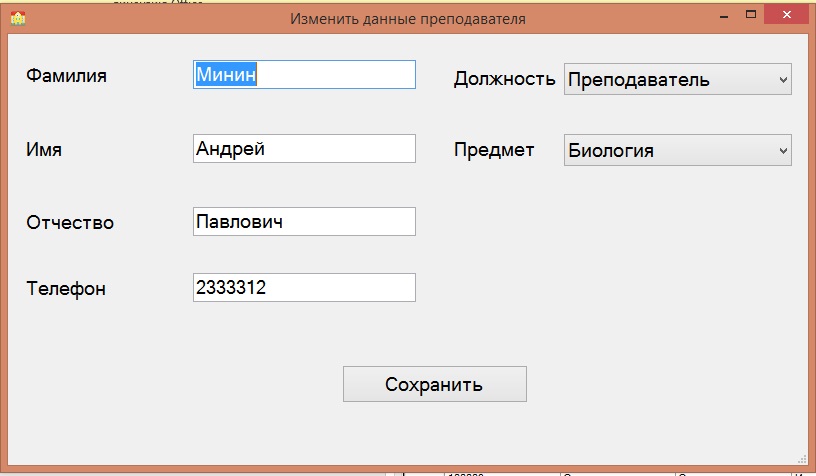


Рисунок 3.5 – Форма «Изменить данные преподавателя»

При нажатии кнопки «Удалить» на форме «Персонал» (рисунок 3.2) удаляется выбранный из списка преподаватель. Также он удаляется из базы данных.

Теперь рассмотрим вход в систему пользователя «Завуч». После подтверждения правильности логина и пароля открывается форма «Завуч» (рисунок 3.6). На этой форме имеются кнопки: «Работа с учащимися» и «Отчеты». При нажатии кнопки «Работа с учащимися» вызывается форма «Учащиеся» (рисунок 3.7).

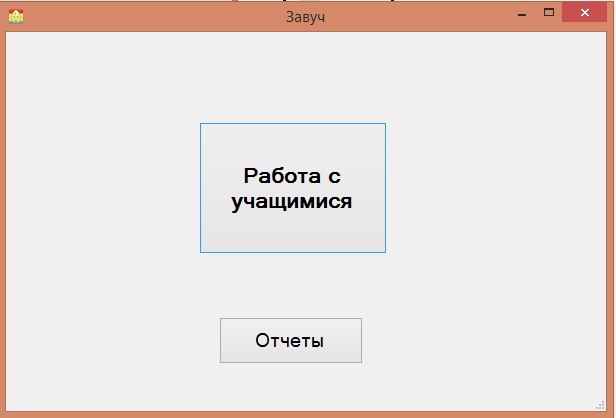


Рисунок 3.6 – Форма «Завуч»

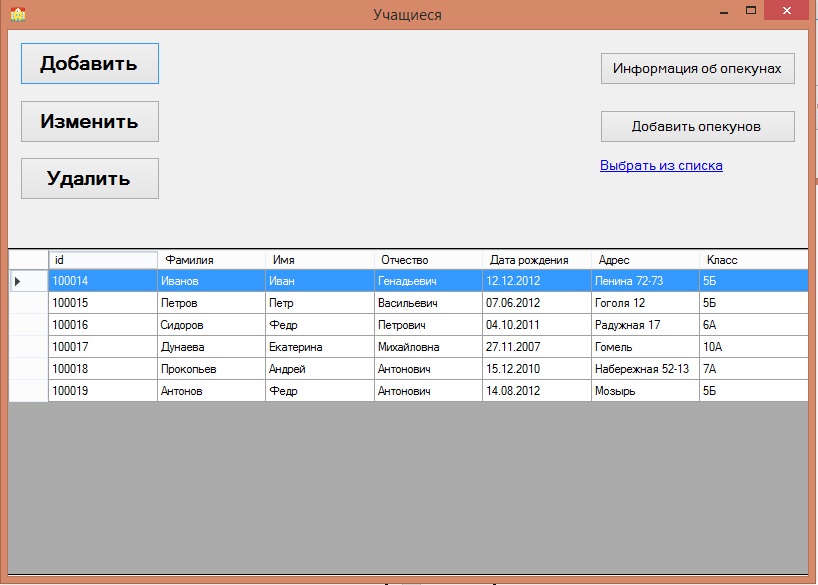


Рисунок 3.7 – Форма «Учащиеся»

На форме «Учащиеся» (рисунок 3.7) имеются список всех учащихся, кнопки «Добавить», «Изменить», «Удалить», «Информация об опекунах», «Добавить опекунов», а также ссылка для выбора опекунов из существующего списка. При нажатии на кнопку «Добавить» вызывается форма «Добавить учащегося» (рисунок 3.8). На этой форме имеются поля для заполнения данных о новом учащемся, а также выбор класса из выпадающего меню. После корректного ввода данных и нажатия кнопки «Сохранить» данные добавляются базу данных и список существующих учащихся.

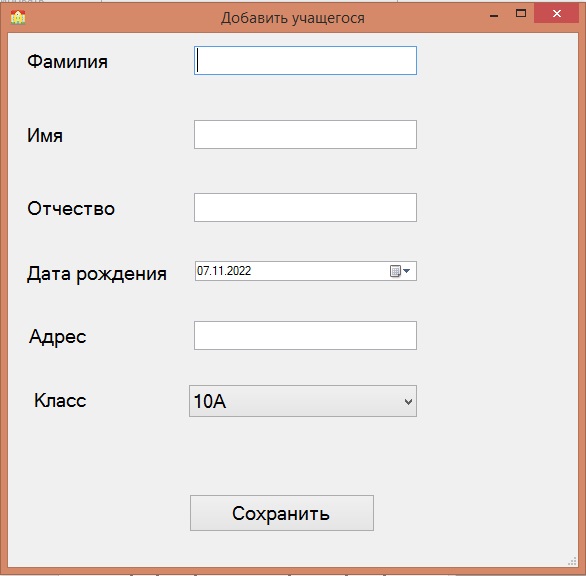


Рисунок 3.8 – Форма «Добавить учащегося»

При нажатии кнопки «Изменить» вызывается форма «Изменить данные учащегося» (рисунок 3.9). На этой форме заполняются поля автоматически в зависимости от выбранного учащегося. После изменения данных и нажатия кнопки «Сохранить» измененные данные изменяются и базе данных, а также в списке учащихся.

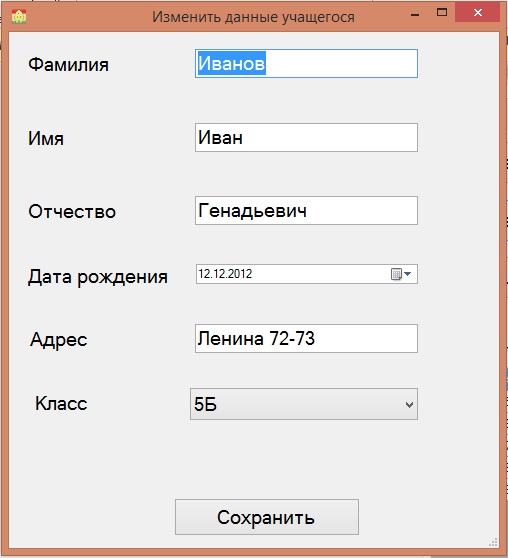


Рисунок 3.9 – Форма «Изменить данные учащегося»

При нажатии кнопки «Удалить» учащийся удаляется из базы данных.

При нажатии кнопки «Информация об опекунах» на форме «Учащиеся» вызывается форма «Информация об опекунах» (рисунок 3.10), в которой представлен список опекунов выбранного учащегося. При нажатии кнопки «Добавить опекунов» вызывается форма «Добавить опекунов» (рисунок 3.11), в которой после ввода корректных данных и нажатия кнопки «Сохранить», новый опекун добавляется для выбранного учащегося, а также вносится в список существующих опекунов и базу данных. Также опекунов для выбранного учащего из существующего списки при нажатии на ссылку «Выбрать из списка». Список существующих опекунов представлен на рисунке 3.12 (кнопка «Добавить опекуна» добавляет опекуна для выбранного учащегося).

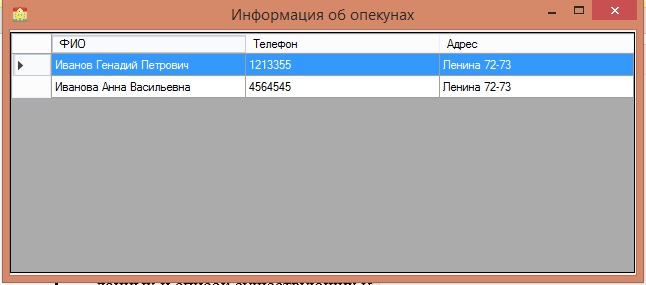


Рисунок 3.10 – Форма «Информация об опекунах»

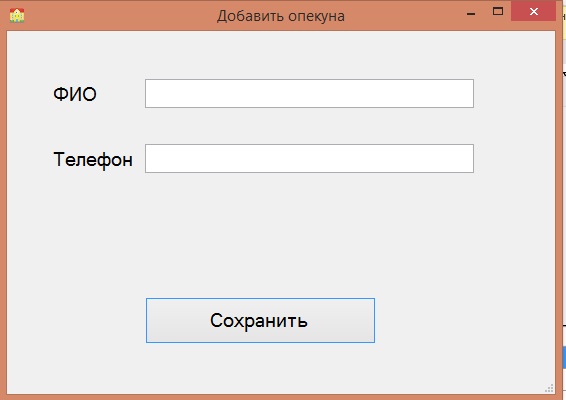


Рисунок 3.11 – Форма «Добавить опекуна»

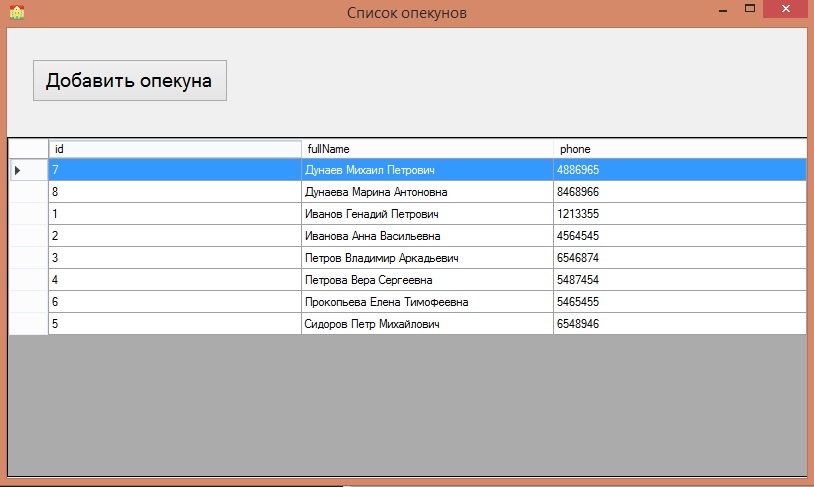


Рисунок 3.12 – Форма «Список опекунов»

Теперь вернемся к форме «Завуч» (рисунок 3.6). Кнопка «Отчеты» при нажатии вызывает форму «Отчеты» (рисунок 3.13-3.14), на которой имеются информация по кол-ву учащихся и среднему баллу по школе, а также кнопки «Средний балл по предметам» и «Средний балл по классам», при нажатии на которые можно получить отчеты по среднему баллу по каждому предмету или по среднему баллу по каждому классу соответственно.

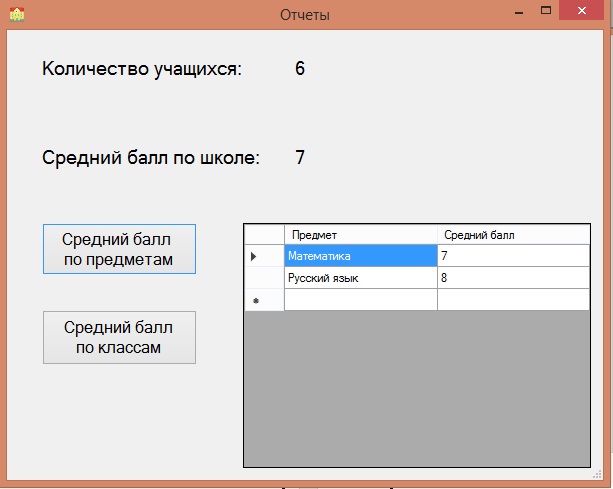


Рисунок 3.13 – Форма «Отчеты» со средним баллом по предметам

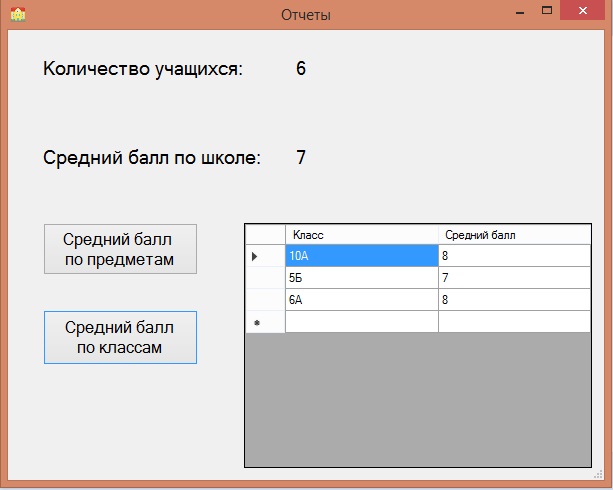


Рисунок 3.14 – Форма «Отчеты» со средним баллом по классам

Теперь рассмотрим вход в систему пользователя «Классный руководитель». После проверки правильности ввода логина и пароля открывается форма «Журнал классного руководителя» (рисунок 3.15-3.16). На этой форме присутствует таб-меню с двумя вкладками: «Урок» и «Журнал». На первой имеется список учащихся с оценками по данному предмету, который преподают преподаватель, кнопки «Добавить», «Сохранить», «Удалить», выпадающий список с существующими классами, кнопка «Обновить» (для обновления данных в списке при смене класса), а также ФИО преподавателя и название предмета.

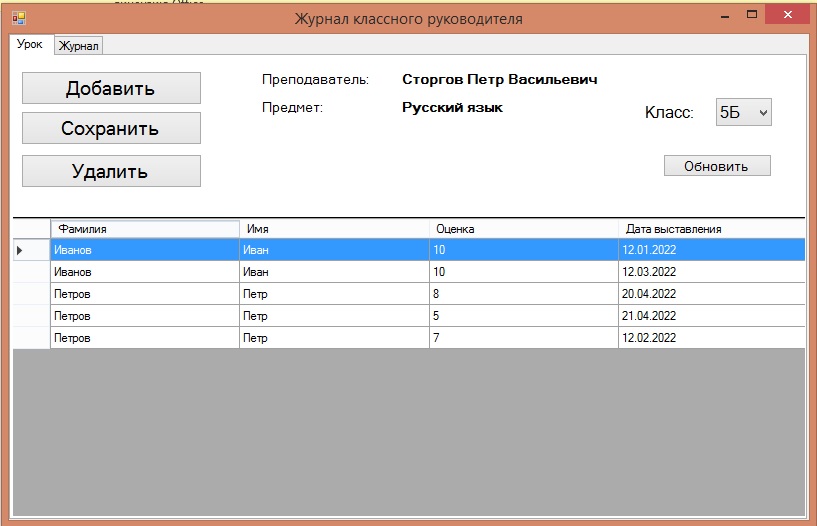


Рисунок 3.15 – Форма «Журнал классного руководителя» с вкладкой «Урок»

На второй вкладке – список учащихся с оценками по каждому предмету для класса, находящиеся под руководством данного классного руководителя. Также имеется выпадающий список с выбором класса для тех случаев, когда классный руководитель имеет под своей опекой несколько классов. Кнопка «Обновить» выполняет те же функции, что и на вкладке «Урок».

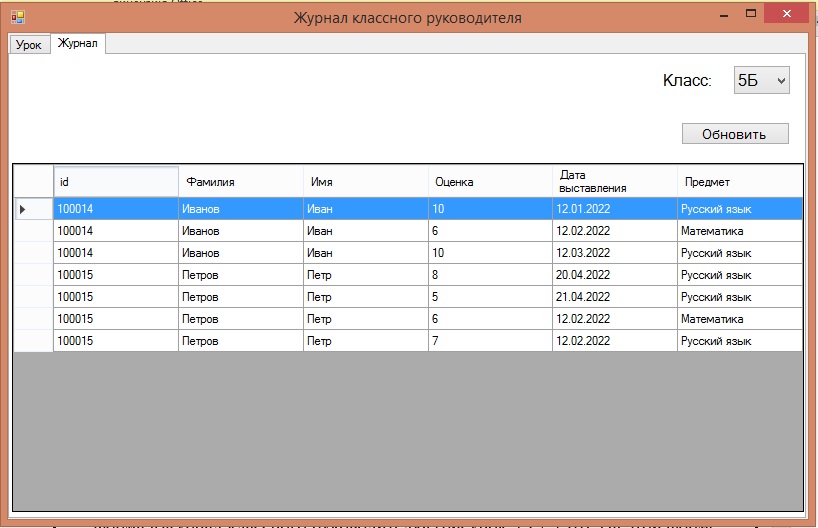


Рисунок 3.16 – Форма «Журнал классного руководителя» с вкладкой «Журнал»

Вернемся к вкладке «Урок». При нажатии кнопки «Добавить» вызывается форма «Добавить учащегося в журнал оценок» (рисунок 3.17). На этой форме присутствует список учащихся данного класса. При нажатии кнопки «Добавить учащегося» выбранный учащийся добавляется в журнал для выставления оценки. Оценка и дата выставления вводится преподавателем в списке добавленному учащемуся.

При нажатии кнопки «Удалить» строка с оценкой удаляется из списка. Также существует возможность изменения оценок и даты выставления.

При нажатии кнопки «Сохранить» сохраняются измененные данные списка оценок, данные заносятся в базу данных.

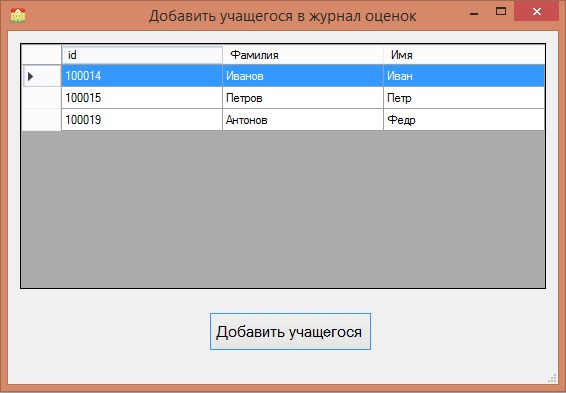


Рисунок 3.17 – Форма «Добавить учащегося в журнал оценок»

При входе в систему пользователя «Преподаватель» открывается форма «Журнал преподавателя» (рисунок 3.18), на которой присутствует таб-меню с единственной вкладкой «Урок». Внешний вид и функционал этой вкладки идентичен внешнему виду и функционалу вкладке «Урок» на форме «Журнал классного руководителя» (рисунок 3.15).

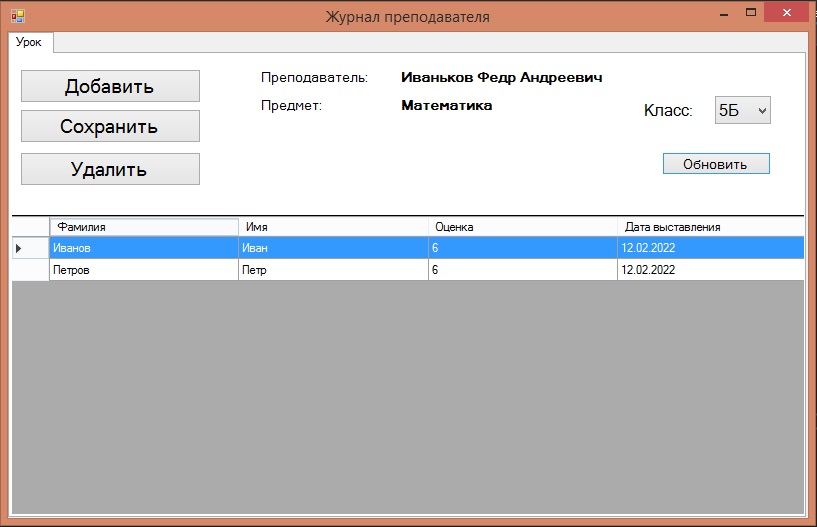


Рисунок 3.18 – Форма «Журнал преподавателя»

При входе в систему пользователя «Учащийся» открывается форма «Оценки учащегося» (рисунок 3.19). На этой форме имеется список оценок данного учащегося по каждому предмету, ФИО и класс.

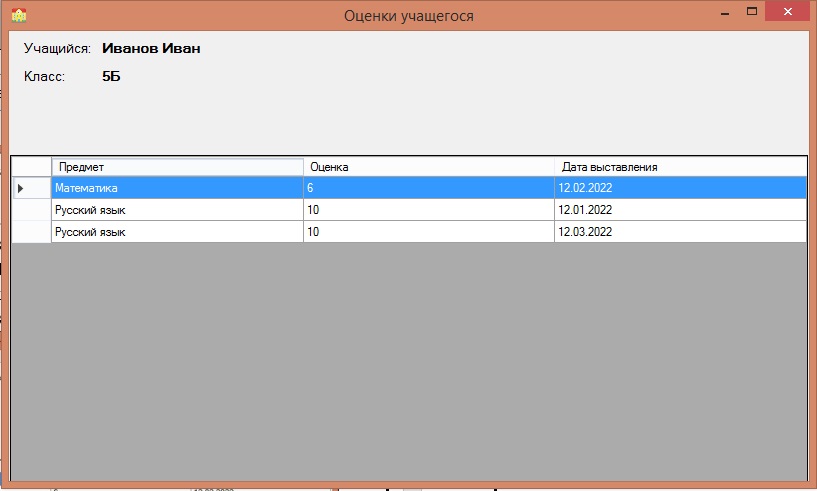


Рисунок 3.19 – Форма «Оценки учащегося»

4. Результаты тестирования информационной системы

Для нормальной работы системы необходимо 256 Мбайт оперативной памяти, Windows NT/2000/XP и выше, MSSQL Server 2012 и выше.

Результатом тестирования может послужить безошибочное добавление, изменение, удаление, а также просмотр информации, связанной с работой системы успеваемости учащихся.

В данном курсовом проекте разработана база данных с клиентским приложением, которая хранится на SQL сервере. Формы составлены на основе запросов и таблиц, и используются для добавления и изменения, просмотра и удаления информации в базе данных.

# Заключение

Прогресс, достигнутый за последние несколько лет во всех аспектах вычислительной техники, включая теорию, технологию и приложения, привели к значительному расширению области применения компьютеров и росту числа их пользователей. Существенной частью современного общества являются разнообразные системы доступа и хранения информации, которые являются неотъемлемой составляющей современного научно-технического прогресса.

Существует много веских причин перевода существующей информации на компьютерную основу, т.к. более быстрая обработка данных и централизация их хранения с использованием клиент/серверных технологий позволяют сберечь значительные средства, а главное и время для получения необходимой информации, а также упрощает доступ и ведение.

Современные СУБД - многопользовательские системы управления базой данных, которые специализируется на управлении массивом информации одним или множеством одновременно работающих пользователей.

Во время выполнения курсовой работы была разработана автоматизированная информационная система учета успеваемости учащихся «Students Perfomance».

При создании ИС использовались Visual Studio 2019, средство моделирования баз данных Sparx Systems Enterprise Architect, средство моделирования бизнес-процессов Sparx Systems Enterprise Architect, сервер баз данных Microsoft SQL SERVER 2012, язык баз данных T-SQL.

При разработке программного обеспечения использовались хранимые процедуры и SQL-запросы.

Ограничение целостности и безопасности базы данных было обеспечено за счет использование хранимых процедур, ссылочной целостности.

Все пункты курсового задания были выполнены, разработанная система выполняет поставленную перед ней задачу.

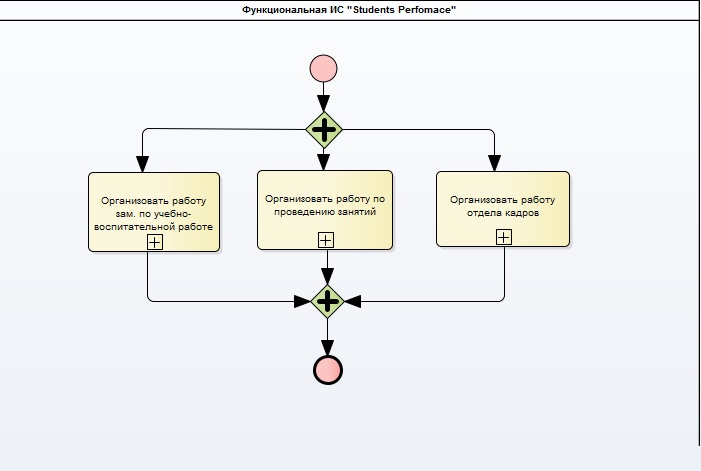
Список использованных литературных источников

1. Microsoft SQL Server 7 для профессионалов. – СПб.: Питер, 2000. – 896 с.
2. Алан Бьюли, Энди Оппель. Изучаем SQL. – М.:Символ, 2007. – 312 с.
3. Жилинский А. Самоучитель Microsoft SQL Server 2005. – СПб:BHV, 2004. – 224 с.
4. [Кишори Бхамидипати](http://oz.by/books/more106510.html?id_search=127378#writer). SQL. Справочник программиста.–М.:Эком, 2003. – 304 с.
5. Конспект лекций «Базы данных»
6. Роберт Виейра. Программирование баз данных Microsoft SQL Server 2005. Базовый курс. – М.:Вильямс, 2003. – 848 с.
7. Уилтон П., Колби Дж. SQL для начинающих. – М.: Вильямс, 2006. – 496 с.
8. Фленов М. Transact SQL в подлиннике. – СПб:BHV, 2006. – 576 с.
9. Практическое руководство по SQL. – М.: Вильямс, 2002. – 352 с.

Приложения

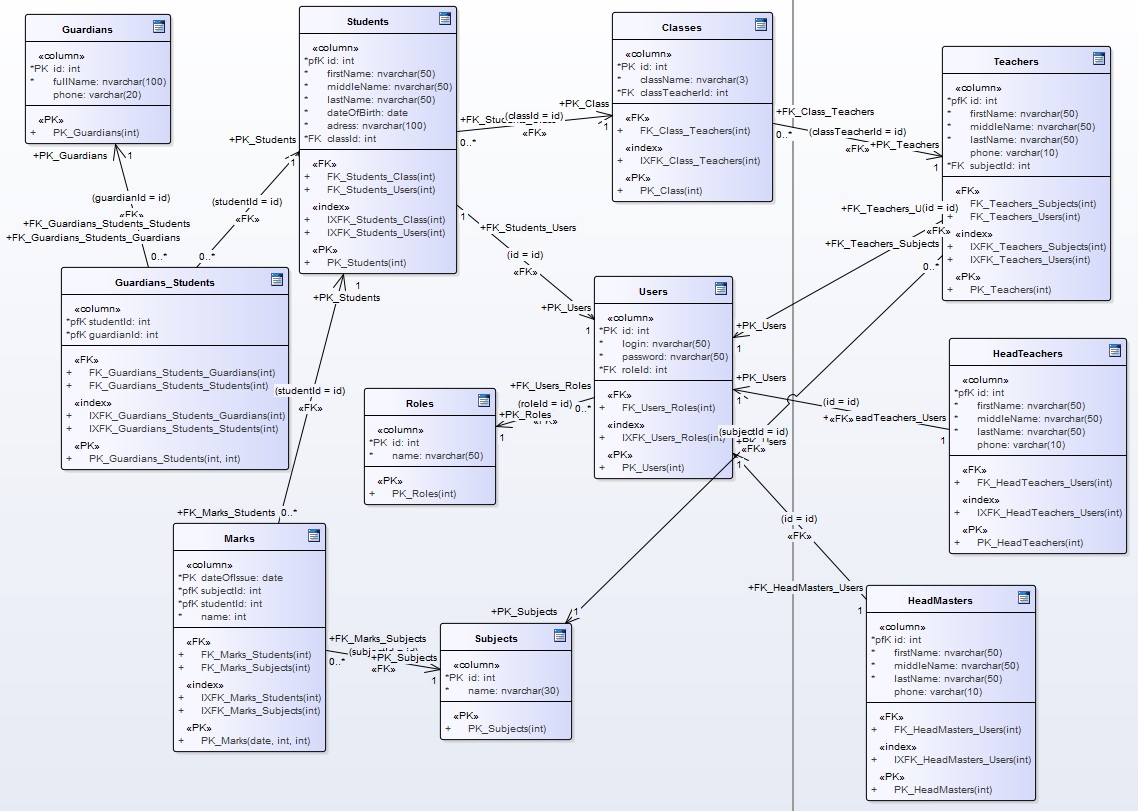
Приложение 1

Диаграмма верхнего уровня функциональной модели



Приложение 2

ER-диаграмма логического уровня



Приложение 3

ER-диаграмма физического уровня

