**lМИНИСТЕРСТВО ЦИФРОВОГО РАЗВИТИЯ, СВЯЗИ И   
МАССОВЫХ КОММУНИКАЦИЙ РОССИЙСКОЙ ФЕДЕРАЦИИ**

**Федеральное государственное бюджетное   
образовательное учреждение высшего образования**

**«Санкт-Петербургский Государственный Университет  
Телекоммуникаций им. проф. М.А. Бонч-Бруевича»**

**Факультет информационных технологий и программной инженерии**

**Кафедра систем обработки данных**

**КУРСОВОЙ ПРОЕКТ**

по дисциплине «Управление данными»

на тему «Информационная система Вуза»:

|  |  |
| --- | --- |
|  | Выполнил:  Студент группы ИСТ-312  Кандиков Максим Вадимович |
|  | (подпись) |
|  | Принял:  Старший преподаватель кафедры БИС  к. с.-х. н. Медведев Сергей Алексеевич |
|  | (подпись) |
|  | Оценка: **\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_** |

Санкт-Петербург, 2024

**Содержание**

[ВВЕДЕНИЕ 3](#_Toc185557789)

[ГЛАВА 1. ИНФОЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 5](#_Toc185557790)

[1.1. Таблица атомарных объектов 5](#_Toc185557791)

[1.2. Составление списка сущностей 6](#_Toc185557792)

[1.3. Диаграмма сущностей 7](#_Toc185557793)

[ГЛАВА 2. ДАТАЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ 8](#_Toc185557794)

[2.1 Составление списка таблиц. 8](#_Toc185557795)

[2.2. Нормализация таблиц до НФБК 11](#_Toc185557796)

[2.3. Итоговый список таблиц 16](#_Toc185557797)

[2.4. Запросы в нотации реляционной алгебры 17](#_Toc185557798)

[ГЛАВА 3. РЕАЛИЗАЦИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ В СУБД POSTGRES 19](#_Toc185557799)

[3.1. Скрипт создания базы данных 19](#_Toc185557800)

[3.2. Диаграмма базы данных 27](#_Toc185557801)

[ЗАКЛЮЧЕНИЕ 28](#_Toc185557802)

# **ВВЕДЕНИЕ**

В современном образовательном процессе управление данными становится неотъемлемой частью эффективного функционирования учебных заведений. Автоматизация задач, связанных с учетом студентов, преподавателей, дисциплин и академических достижений, позволяет значительно повысить точность, прозрачность и скорость работы системы. Именно эти аспекты стали основой для создания данной информационной системы.

Курсовой проект посвящен разработке базы данных, которая позволяет систематизировать данные о студентах, группах, специальностях, дисциплинах и преподавателях. Предметная область охватывает ключевые аспекты организации учебного процесса в университете. Например, при поступлении в вуз студент указывает свой номер паспорта, номер телефона и почту, на основании чего ему выдается студенческий билет. Каждый студент попадает в определенную группу, которая формируется по выбранной специальности. Группа не только структурирует учебный процесс, но и упрощает управление студентами, позволяя отслеживать год их набора.

У каждой группы есть название, а также связь с определенной специальностью. Это позволяет автоматически назначать студентам соответствующие дисциплины. Например, группа "ИСТ-312" объединяет студентов специальности "Информационные системы и технологии", для которых предусмотрены дисциплины "Управление данными", "Алгоритмы и структуры данных" и "Теория вероятностей и математическая статистика". Каждая дисциплина содержит данные о количестве часов, форме сдачи и описании, что упрощает управление учебным процессом.

Преподаватели, привязанные к кафедрам, ведут дисциплины, соответствующие их профессиональной специализации. Кафедра контролирует образовательные программы, распределяет курсы и адаптирует их под потребности студентов. У преподавателей фиксируются такие параметры, как ФИО, должность, контактные данные и стаж работы. Например, преподаватель кафедры "Информатики и компьютерного дизайна" может вести курс "Дизайн графических систем" для студентов разных групп.

Особое внимание уделяется учету статуса студентов и их финансовой мотивации. Успешно обучающиеся студенты могут получать стипендии, которые включают информацию о размере, типе (академическая или социальная) и сроках выплат. Например, академическая стипендия является важным инструментом поддержки студентов очной формы обучения.

Цель данного курсового проекта — разработать информационную систему, которая обеспечивает удобное управление учебными данными, их целостность и надежность. Мой курсовой проект нацелен на создание надежного инструмента, который упрощает учет учебных процессов и обеспечивает удобство для всех участников образовательного процесса.

# **ГЛАВА 1. ИНФОЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ**

Инфологическое моделирование будет проводиться с использованием восходящего метода. При использовании этого метода в начале составляется таблица атомарных объектов. Затем на основании этой таблицы выделяются сущности, между ними устанавливаются связи. Сущности и связи отображаются на диаграмме.

# **1.1. Таблица атомарных объектов**

Для приведённого во Введении описания предметной области таблица атомарных объектов будет выглядеть следующим образом:

Таблица 1.

|  |  |
| --- | --- |
| Атомарный объект | Зависимость |
| Номер паспорта студента | - |
| Фамилия студента | Номер паспорта студента |
| Имя студента | Номер паспорта студента |
| Отчество студента | Номер паспорта студента |
| Специальность студента | - |
| Почта студента | Номер паспорта студента |
| Статус (обучается, отчислен) | Номер паспорта студента |
| Телефон студента | Номер паспорта студента |
| Номер паспорта преподавателя | - |
| Фамилия преподавателя | Номер паспорта преподавателя |
| Имя преподавателя | Номер паспорта преподавателя |
| Отчество преподавателя | Номер паспорта преподавателя |
| Телефон преподавателя | Номер паспорта преподавателя |
| Почта преподавателя | Номер паспорта преподавателя |
| Должность | Номер паспорта преподавателя |
| Стаж работы | Номер паспорта преподавателя |
| Тип стипендии (академическая, социальная) | Номер паспорта студента |
| Размер | Тип стипендии |
| Дата начала выплат | Тип стипендии |
| Дата окончания выплат | Тип стипендии |
| Название группы | Номер паспорта студента |
| Год набора группы | Название группы |
| Название дисциплины | - |
| Количество часов | Название дисциплины |
| Форма сдачи | Название дисциплины |
| Описание дисциплины | Название дисциплины |
| Название специальности | Название группы |
| Код специальности | Название специальности |
| Продолжительность обучения | Название специальности |
| Название кафедры | - |

# **1.2. Составление списка сущностей**

На основании таблицы атомарных объектов можно выделить следующие сущности с атрибутами:

* Студенты: хранят информацию о студентах, включая номер паспорта, ФИО, контактные данные и номер студенческого билета.
* Группы: содержат данные о группах, включая их название, год набора и связь со специальностью.
* Специальности: предоставляют информацию о коде, названии и продолжительности обучения.
* Дисциплины: включают название, описание, форму сдачи и количество часов.
* Преподаватели: включают ФИО, должность, контактные данные и связь с кафедрой.
* Кафедры: содержат данные о названии и количестве преподавателей.
* Стипендии: включают тип, размер, даты начала и окончания выплат.

# **1.3. Диаграмма сущностей**

Ниже представлена диаграмма сущностей в нотации Чена.

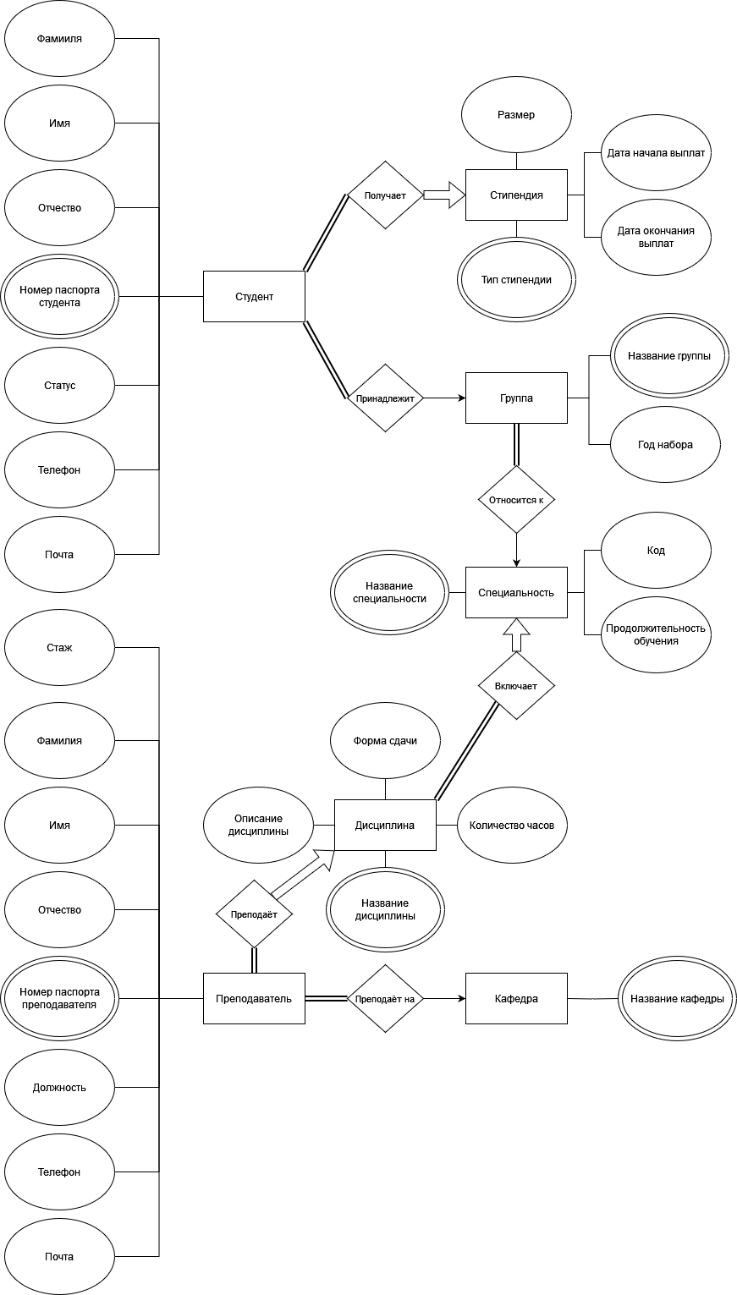


Рисунок 1. Диаграмма сущностей в нотации Чена

# **ГЛАВА 2. ДАТАЛОГИЧЕСКОЕ МОДЕЛИРОВАНИЕ ПРЕДМЕТНОЙ ОБЛАСТИ**

Даталогическое моделирование будет проводиться с использованием реляционной модели данных. В качестве исходного материала берутся результаты инфологического моделирования.

# **2.1 Составление списка таблиц.**

На основе построенной на этапе инфологического моделирования онтологии можно выделить следующие таблицы:

Таблица 1: Студенты (Students)Students(StudentID, PassportNumber, LastName, FirstName, MiddleName, Status, Phone, Email, GroupID)

* StudentID — первичный ключ.
* PassportNumber — альтернативный ключ, так как каждый студент имеет уникальный номер паспорта.
* GroupID — внешний ключ, ссылающийся на таблицу "Группы"

Таблица 2: Группы (Groups)Groups(GroupID, GroupName, EnrollmentYear, SpecialtyID)

* GroupID — первичный ключ.
* GroupName — альтернативный ключ, так как каждая группа имеет уникальное название.
* SpecialtyID — внешний ключ, ссылающийся на таблицу "Специальности".

Таблица 3: Специальности (Specialties)  
Specialties(SpecialtyID, SpecialtyName, Code, DurationYears)

* SpecialtyID — первичный ключ.
* SpecialtyName — альтернативный ключ, так как каждая специальность имеет уникальное название.

Таблица 4: Дисциплины (Disciplines)  
Discpilines(DisciplineID, DisciplineName, Description, ExamType, Hours)

* DisciplineID — первичный ключ.
* DisciplineName — альтернативный ключ, так как название дисциплины уникально.

Таблица 5: Преподаватели (Teachers)  
Teachers(TeacherID, PassportNumber, LastName, FirstName, MiddleName, WorkExperience, Position, Phone, Email, DepartmentID)

* TeacherID — первичный ключ.
* PassportNumber — альтернативный ключ, так как каждый преподаватель имеет уникальный номер паспорта.
* DepartmentID — внешний ключ, ссылающийся на таблицу "Кафедры".

Таблица 6: Кафедры (Departments)  
Departments(DepartmentID, DepartmentName)

* DepartmentID — первичный ключ.
* DepartmentName — альтернативный ключ, так как у каждой кафедры уникальное название.

Таблица 7: Стипендии (Scholarships)  
Scholarships(ScholarshipID, Type, Amount)

* ScholarshipID — первичный ключ.
* Type — альтернативный ключ, так как тип стипендии уникален.

Таблица 8: Преподаватели и дисциплины (Teacher\_Discipline)  
Teacher\_Discipline(TeacherID, DisciplineID*)*

* Составной первичный ключ *—* TeacherID + DisciplineID.
* TeacherID — внешний ключ, ссылающийся на таблицу "Преподаватели".
* DisciplineID — внешний ключ, ссылающийся на таблицу "Дисциплины".

Таблица 9: Дисциплины и специальности (Discipline\_Specialty)  
Discipline\_Specialty(DisciplineID, SpecialtyID)

* Составной первичный ключ — DisciplineID + SpecialtyID.
* DisciplineID — внешний ключ, ссылающийся на таблицу "Дисциплины”.
* SpecialtyID — внешний ключ, ссылающийся на таблицу "Специальности".

Таблица 10: Студенты и стипендии (Student\_Scholarship)  
Student\_Scholarship(StudentID, ScholarshipID, StartDate, EndDate)

* Составной первичный ключ — StudentID + ScholarshipID.
* StudentID — внешний ключ, ссылающийся на таблицу "Студенты".
* ScholarshipID — внешний ключ, ссылающийся на таблицу "Стипендии".

# **2.2. Нормализация таблиц до НФБК**

В данном разделе я провёл анализ всех таблиц на соответствие нормальным формам: 1НФ, 2НФ, 3НФ и НФБК и выполнил по каждой таблице проверку.

**Таблица: Студенты (Students)**Схема таблицы: Students(StudentID, PassportNumber, LastName, FirstName, MiddleName, Status, Phone, Email, GroupID).

Потенциальные ключи: PassportNumber, Phone, Email

* 1НФ: Все атрибуты атомарны. StudentID, PassportNumber, Status, Phone, и Email — хранят числовые или строковые значения. ФИО (LastName, FirstName, MiddleName) разделены на отдельные поля. GroupID содержит ссылку на группу.

Таблица соответствует 1НФ.

* 2НФ:

Таблица соответствует 2НФ, так как ключ не составной.

* 3НФ:

Может ли быть для одного GroupID несколько LastName? Да.

Может ли быть для одного GroupID несколько FirstName? Да.

Может ли быть для одного GroupID несколько MiddleName? Да.

Может ли быть для одного LastName несколько FirstName? Да.

Может ли быть для одного LastName несколько MiddleName? Да.

Может ли быть для одного LastName несколько GroupID? Да.

Таблица соответствует 3НФ.

* НФБК:

Таблица соответствует НФБК, так как все детерминанты являются суперключами, нет зависимостей от неключевых атрибутов к ключевым, и все функциональные зависимости корректны.

**Таблица: Группы (Groups)**Схема таблицы: Groups(GroupID, GroupName, EnrollmentYear, SpecialtyID).

Потенциальные ключи: GroupName

* 1НФ: Все атрибуты атомарны. GroupID — числовой идентификатор. GroupName — строка, используется целиком. EnrollmentYear и SpecialtyID содержат числовые значения.

Таблица соответствует 1НФ.

* 2НФ:

Таблица соответствует 2НФ, так как ключ не составной.

* 3НФ:

Может ли быть для одного EnrollmentYear несколько SpecialityID? Да.

Может ли быть для одного SpecialityID несколько EnrollmentYear? Да.

Таблица соответствует 3НФ.

* НФБК:

Таблица соответствует НФБК, потому что ключ не составной, а из-за этого НФБК нарушено быть не может.

**Таблица: Специальности (Specialties)**Схема таблицы: Specialties(SpecialtyID, SpecialtyName, Code, DurationYears).

Потенциальные ключи: SpecialityName

* 1НФ: Все атрибуты атомарны. SpecialtyID, SpecialtyName, Code, DurationYears содержат числовые или строковые значения. Таблица соответствует 1НФ.
* 2НФ:

Таблица соответствует 2НФ, так как ключ не составной.

* 3НФ:

Может ли быть для одного Code несколько DurationYears? Да.

Может ли быть для одного DurationYears несколько Code? Да.

Таблица соответствует 3НФ.

* НФБК:

Таблица соответствует НФБК, потому что ключ не составной, а из-за этого НФБК нарушено быть не может.

**Таблица: Дисциплины (Disciplines)**  
Схема таблицы: Disciplines(DisciplineID, DisciplineName, Description, ExamType, Hours).

Потенциальный ключ: DisciplineName, Description

* 1НФ: Все атрибуты атомарны. DisciplineID, DisciplineName, Description, ExamType, Hours содержат числовые или строковые значения. Таблица соответствует 1НФ.
* 2НФ:

Таблица соответствует 2НФ, так как ключ не составной.

* 3НФ:

Может ли быть для одного ExamType несколько Hours? Да.

Может ли быть для одного Hours несколько ExamType? Да.

Таблица соответствует 3НФ.

* НФБК:

Таблица соответствует НФБК, потому что ключ не составной, а из-за этого НФБК нарушено быть не может.

**Таблица: Преподаватели (Teachers)**  
Схема таблицы: Teachers(TeacherID, PassportNumber, LastName, FirstName, MiddleName, WorkExperience, Position, Phone, Email, DepartmentID).

Потенциальный ключ: PassportNumber, Phone, Email.

* 1НФ: Все атрибуты атомарны. TeacherID, PassportNumber, WorkExperience, Position, Phone, и Email содержат числовые или строковые значения. Таблица соответствует 1НФ.
* 2НФ:

Таблица соответствует 2НФ, так как ключ не составной.

* 3НФ:

Может ли быть для одного LastName несколько FirstName? Да.

Может ли быть для одного LastName несколько MiddleName? Да.

Может ли быть для одного LastName несколько DepartmentID? Да.

Может ли быть для одного LastName несколько Position? Да.

Может ли быть для одного LastName несколько WorkExpreience? Да.

Может ли быть для одного WorkExperience несколько Position? Да.

Может ли быть для одного Position несколько WorkExperience? Да.

Может ли быть для одного DepartmentID несколько Position? Да.

Может ли быть для одного Position несколько DepartmentID? Да.

Таблица соответствует 3НФ.

* НФБК:

Таблица соответствует НФБК, потому что ключ не составной, а из-за этого НФБК нарушено быть не может.

**Таблица: Кафедры (Departments)**  
Схема таблицы: Departments(DepartmentID, DepartmentName).

Потенциальный ключ: DepartmentName

* 1НФ: Все атрибуты атомарны. DepartmentID и DepartmentName содержат числовые или строковые значения. Таблица соответствует 1НФ.
* 2НФ:

Таблица соответствует 2НФ, так как ключ не составной.

* 3НФ:

Таблица соответствует 3НФ, потому что в ней нет неключевых атрибутов.

* НФБК:

Таблица соответствует НФБК, потому что ключ не составной, а из-за этого НФБК нарушено быть не может.

**Таблица: Стипендии (Scholarships)**  
Схема таблицы: Scholarships(ScholarshipID, Type, Amount)

Потенциальный ключ: Type

* 1НФ: Все атрибуты атомарны. ScholarshipID, Type, Amount, StartDate, и EndDate содержат числовые или строковые значения. Таблица соответствует 1НФ.
* 2НФ:

Таблица соответствует 2НФ, так как ключ не составной.

* 3НФ:

Таблица соответствует 3НФ, так как есть всего один неключевой атрибут.

* НФБК:

Таблица соответствует НФБК, потому что ключ не составной, а из-за этого НФБК нарушено быть не может.

**Таблицы связей многие ко многим:** (Teacher\_Discipline, Discipline\_Specialty, Student\_Scholarship) содержат только составные первичные ключи. Атрибуты зависят исключительно от этих ключей. Все таблицы соответствуют НФБК.

Все таблицы успешно проверены на соответствие нормальным формам, включая НФБК. Я не выявил нарушений нормализации.

# **2.3. Итоговый список таблиц**

В результате нормализации были получены следующие таблицы:

1. **Students**(StudentID, PassportNumber, LastName, FirstName, MiddleName, Status, Phone, Email, GroupID), причём атрибут PassportNumber является альтернативным ключом, а GroupID является внешним ключом, ссылающимся на таблицу Groups.
2. **Groups**(GroupID, GroupName, EnrollmentYear, SpecialtyID), причём атрибут GroupName является альтернативным ключом, а SpecialtyID является внешним ключом, ссылающимся на таблицу Specialties.
3. **Specialties**(SpecialtyID, SpecialtyName, Code, DurationYears), причём атрибут SpecialtyName является альтернативным ключом.
4. **Disciplines**(DisciplineID, DisciplineName, Description, ExamType, Hours), причём атрибут DisciplineName является альтернативным ключом.
5. **Teachers**(TeacherID, PassportNumber, LastName, FirstName, MiddleName, WorkExperience, Position, Phone, Email, DepartmentID), причём атрибут PassportNumber является альтернативным ключом, а DepartmentID является внешним ключом, ссылающимся на таблицу Departments.
6. **Departments**(DepartmentID, DepartmentName), причём атрибут DepartmentName является альтернативным ключом.
7. **Scholarships**(ScholarshipID, Type, Amount, StartDate, EndDate), причём атрибут Type является альтернативным ключом.
8. **Teacher\_Discipline**(TeacherID, DisciplineID), причём атрибуты TeacherID + DisciplineID составляют составной первичный ключ. TeacherID является внешним ключом, ссылающимся на таблицу Teachers, а DisciplineID является внешним ключом, ссылающимся на таблицу Disciplines.
9. **Discipline\_Specialty**(DisciplineID, SpecialtyID), причём атрибуты DisciplineID + SpecialtyID составляют составной первичный ключ. DisciplineID является внешним ключом, ссылающимся на таблицу Disciplines, а SpecialtyID является внешним ключом, ссылающимся на таблицу Specialties.
10. **Student\_Scholarship**(StudentID, ScholarshipID), причём атрибуты StudentID + ScholarshipID составляют составной первичный ключ. StudentID является внешним ключом, ссылающимся на таблицу Students, а ScholarshipID является внешним ключом, ссылающимся на таблицу Scholarships.

**2.4. Запросы в нотации реляционной алгебры**

Для реализации функциональных требований, обусловленных выбранной предметной областью, требуется сделать следующие запросы.

1. Получить список студентов, обучающихся в заданной группе:

πStudentID,LastName,FirstName(σGroupID=G1(Students))

1. Получить преподавателей, работающих на заданной кафедре:

πTeacherID,LastName,FirstName(σDepartmentID=D1(Teachers))

1. Получить список студентов, получающих заданный тип стипендии:

πStudentID,LastName,FirstName​(σType=′academic′​(Student⋈Student\_Scholarship⋈Scholarship))

1. Получить список дисциплин для заданной специальности:

πDisciplineName​(Discipline⋈Discipline\_Specialty⋈Specialty)

1. Получить количество студентов в каждой группе:

πGroupID,COUNT(StudentID)(Student)

# **ГЛАВА 3. РЕАЛИЗАЦИЯ ИНФОРМАЦИОННОЙ СИСТЕМЫ В СУБД POSTGRES**

Для реализации информационной системы была выбрана СУБД Postgres и среда разработки PgAdmin 4.

# **3.1. Скрипт создания базы данных**

-- Таблица Departments

CREATE TABLE IF NOT EXISTS Departments (

DepartmentID SERIAL PRIMARY KEY,

DepartmentName VARCHAR(100) NOT NULL UNIQUE

);

-- Таблица Specialties

CREATE TABLE IF NOT EXISTS Specialties (

SpecialtyID SERIAL PRIMARY KEY,

SpecialtyName VARCHAR(100) NOT NULL UNIQUE,

Code VARCHAR(10) NOT NULL,

DurationYears INT NOT NULL

);

-- Таблица Groups

CREATE TABLE IF NOT EXISTS Groups (

GroupID SERIAL PRIMARY KEY,

GroupName VARCHAR(50) NOT NULL UNIQUE,

EnrollmentYear INT NOT NULL,

SpecialtyID INT NOT NULL,

CONSTRAINT FK\_Group\_Specialty FOREIGN KEY (SpecialtyID) REFERENCES Specialties(SpecialtyID) ON UPDATE CASCADE

);

-- Таблица Students

CREATE TABLE IF NOT EXISTS Students (

StudentID SERIAL PRIMARY KEY,

PassportNumber VARCHAR(20) NOT NULL UNIQUE,

LastName VARCHAR(50) NOT NULL,

FirstName VARCHAR(50) NOT NULL,

MiddleName VARCHAR(50),

Status VARCHAR(20) NOT NULL,

Phone VARCHAR(15) NOT NULL UNIQUE,

Email VARCHAR(100) NOT NULL UNIQUE,

GroupID INT NOT NULL,

CONSTRAINT FK\_Student\_Group FOREIGN KEY (GroupID) REFERENCES Groups(GroupID) ON UPDATE CASCADE

);

-- Таблица Teachers

CREATE TABLE IF NOT EXISTS Teachers (

TeacherID SERIAL PRIMARY KEY,

PassportNumber VARCHAR(20) NOT NULL UNIQUE,

LastName VARCHAR(50) NOT NULL,

FirstName VARCHAR(50) NOT NULL,

MiddleName VARCHAR(50),

WorkExperience INT NOT NULL,

Position VARCHAR(50) NOT NULL,

Phone VARCHAR(15) NOT NULL UNIQUE,

Email VARCHAR(100) NOT NULL UNIQUE,

DepartmentID INT NOT NULL,

CONSTRAINT FK\_Teacher\_Department FOREIGN KEY (DepartmentID) REFERENCES Departments(DepartmentID) ON UPDATE CASCADE

);

-- Таблица Scholarships

CREATE TABLE IF NOT EXISTS Scholarships (

ScholarshipID SERIAL PRIMARY KEY,

Type VARCHAR(50) NOT NULL UNIQUE,

Amount DECIMAL(10, 2) NOT NULL

);

-- Таблица Disciplines

CREATE TABLE IF NOT EXISTS Disciplines (

DisciplineID SERIAL PRIMARY KEY,

DisciplineName VARCHAR(100) NOT NULL UNIQUE,

Description TEXT NOT NULL UNIQUE,

ExamType VARCHAR(50) NOT NULL,

Hours INT NOT NULL

);

-- Связь многие ко многим: Teachers\_Disciplines

CREATE TABLE IF NOT EXISTS Teachers\_Disciplines (

TeacherID INT NOT NULL,

DisciplineID INT NOT NULL,

PRIMARY KEY (TeacherID, DisciplineID),

CONSTRAINT FK\_TeacherDiscipline\_Teacher FOREIGN KEY (TeacherID) REFERENCES Teachers(TeacherID) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT FK\_TeacherDiscipline\_Discipline FOREIGN KEY (DisciplineID) REFERENCES Disciplines(DisciplineID) ON UPDATE CASCADE

);

-- Связь многие ко многим: Disciplines\_Specialties

CREATE TABLE IF NOT EXISTS Disciplines\_Specialties (

DisciplineID INT NOT NULL,

SpecialtyID INT NOT NULL,

PRIMARY KEY (DisciplineID, SpecialtyID),

CONSTRAINT FK\_DisciplineSpecialty\_Discipline FOREIGN KEY (DisciplineID) REFERENCES Disciplines(DisciplineID) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT FK\_DisciplineSpecialty\_Specialty FOREIGN KEY (SpecialtyID) REFERENCES Specialties(SpecialtyID) ON DELETE CASCADE ON UPDATE CASCADE

);

-- Связь многие ко многим: Students\_Scholarships

CREATE TABLE IF NOT EXISTS Students\_Scholarships (

StudentID INT NOT NULL,

ScholarshipID INT NOT NULL,

StartDate DATE NOT NULL,

EndDate DATE NOT NULL,

PRIMARY KEY (StudentID, ScholarshipID),

CONSTRAINT FK\_StudentScholarship\_Student FOREIGN KEY (StudentID) REFERENCES Students(StudentID) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT FK\_StudentScholarship\_Scholarship FOREIGN KEY (ScholarshipID) REFERENCES Scholarships(ScholarshipID) ON DELETE CASCADE ON UPDATE CASCADE,

CONSTRAINT CHK\_ScholarshipDates CHECK (EndDate >= StartDate)

);

-- Получить список студентов, обучающихся в заданной группе, табличная функция

CREATE FUNCTION GetStudentsByGroup(\_GroupID INT) RETURNS TABLE (

StudentID INT,

LastName VARCHAR,

FirstName VARCHAR

) AS $$

BEGIN

RETURN QUERY (

SELECT StudentID, LastName, FirstName

FROM Students

WHERE GroupID = \_GroupID

);

END;

$$ LANGUAGE plpgsql;

-- Получить преподавателей, работающих на заданной кафедре, табличная функция

CREATE FUNCTION GetTeachersByDepartment(\_DepartmentID INT) RETURNS TABLE (

TeacherID INT,

LastName VARCHAR,

FirstName VARCHAR

) AS $$

BEGIN

RETURN QUERY (

SELECT TeacherID, LastName, FirstName

FROM Teachers

WHERE DepartmentID = \_DepartmentID

);

END;

$$ LANGUAGE plpgsql;

-- Представление: студенты с их группами и специальностями

CREATE VIEW StudentsOverview AS

SELECT

s.StudentID,

s.LastName,

s.FirstName,

s.MiddleName,

s.Status,

s.Email,

g.GroupName,

sp.SpecialtyName

FROM Students s

INNER JOIN Groups g ON s.GroupID = g.GroupID

INNER JOIN Specialties sp ON g.SpecialtyID = sp.SpecialtyID;

-- Скалярная функция для подсчёта студентов в группе

CREATE FUNCTION GetStudentCountInGroup(\_GroupID INT) RETURNS INT AS $$

BEGIN

RETURN (

SELECT COUNT(\*)

FROM Students

WHERE GroupID = \_GroupID

);

END;

$$ LANGUAGE plpgsql;

-- Табличная функция для получения студентов по типу стипендии

CREATE FUNCTION GetStudentsByScholarshipType(\_ScholarshipType VARCHAR) RETURNS TABLE (

StudentID INT,

LastName VARCHAR,

FirstName VARCHAR

) AS $$

BEGIN

RETURN QUERY (

SELECT s.StudentID, s.LastName, s.FirstName

FROM Students s

JOIN Students\_Scholarships ss ON s.StudentID = ss.StudentID

JOIN Scholarships sc ON ss.ScholarshipID = sc.ScholarshipID

WHERE sc.Type = \_ScholarshipType

);

END;

$$ LANGUAGE plpgsql;

-- Получить список дисциплин для заданной специальности, представление

CREATE VIEW DisciplinesForSpecialty AS

SELECT d.DisciplineName

FROM Disciplines d

JOIN Disciplines\_Specialties ds ON d.DisciplineID = ds.DisciplineID

JOIN Specialties sp ON ds.SpecialtyID = sp.SpecialtyID;

-- Получить количество студентов в каждой группе

CREATE VIEW StudentCountByGroup AS

SELECT GroupID, COUNT(StudentID) AS StudentCount

FROM Students

GROUP BY GroupID;

-- Хранимая процедура для добавления нового студента

CREATE PROCEDURE AddNewStudent(

\_PassportNumber VARCHAR,

\_LastName VARCHAR,

\_FirstName VARCHAR,

\_MiddleName VARCHAR,

\_Status VARCHAR,

\_Phone VARCHAR,

\_Email VARCHAR,

\_GroupID INT

)

LANGUAGE plpgsql AS $$

BEGIN

INSERT INTO Students (

PassportNumber, LastName, FirstName, MiddleName, Status, Phone, Email, GroupID

) VALUES (

\_PassportNumber, \_LastName, \_FirstName, \_MiddleName, \_Status, \_Phone, \_Email, \_GroupID

);

END;

$$;

# **3.2. Диаграмма базы данных**

Разработанная база данных представлена на следующей диаграмме в нотации Мартина:

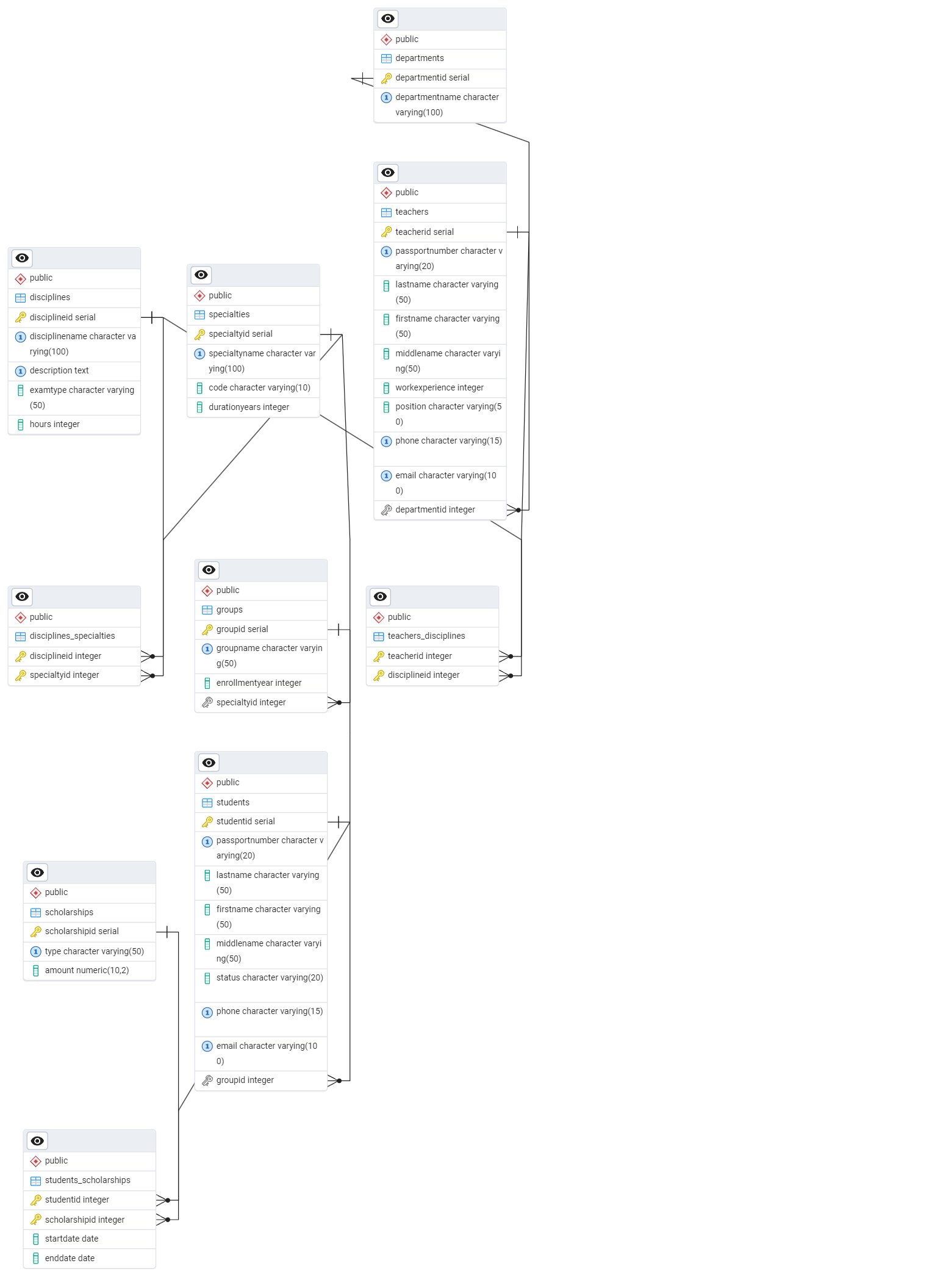


Рисунок 2. Диаграмма базы данных в нотации Мартина.

# **ЗАКЛЮЧЕНИЕ**

В ходе работы над курсовым проектом было проведено инфологическое моделирование предметной области. В результате этого моделирования была описана онтология из 7 сущностей, представленная на диаграмме (Рисунок 1). На основе этой онтологии была спроектирована реляционная база данных из 10 таблиц. Эта база данных была реализована на базе СУБД Postgres с помощью среды разработки PgAdmin 4. Все задачи по разработке информационной системы информационной системы вуза успешно выполнены. Был получен опыт инфологического и даталогического моделирования, опыт работы со средой разработки БД и СУБД Postgres.