**Российский экономический университет им. Г. В. Плеханова**

**ИМИСиЦЭ**

**Творческое задание  
 по дисциплине «Базы данных»  
Проектирование базы данных для автоматизации участия сотрудников и студентов вуза в научных конференциях и семинарах**

Выполнил:   
Воробьев Андрей Владимирович  
студент второго курса,  
направления прикладной математикой и информатики,  
Группа: 15.11Д-ПМПМИ03/21б (413)

Оглавление

[Введение 2](#_Toc128091456)

[Проектирование базы данных 2](#_Toc128091457)

[Описание предметной области 2](#_Toc128091458)

[Концептуальная модель предметной области 2](#_Toc128091459)

[Построение представлений (view) 10](#_Toc128091460)

[Реализация базы данных 11](#_Toc128091461)

[Заполнение данными 12](#_Toc128091462)

[SQL-запросы. Команда SELECT 16](#_Toc128091463)

[Заключение 19](#_Toc128091464)

[Список литературы 20](#_Toc128091465)

# Введение

Целью данной работы является проектирование базы данных для автоматизации участия сотрудников и студентов вуза в научных конференциях и семинарах. База данных позволит хранить и управлять информацией о студентах и преподавателях, группах студентов, конференциях и семинарах, их темах, городах и странах проведения, отзывах и авторах, а также контролировать оплату.

# Проектирование базы данных

## Описание предметной области

Участие в научных конференциях и семинарах является важной частью академической деятельности студентов и преподавателей. Для участия в конференциях и семинарах необходимо зарегистрироваться, выбрать тему и место проведения, оплатить посещение, принять участие в научной программе. После можно оставить отзыв.

Основными сущностями в базе данных являются студенты, которые могут быть организованы в группы, и преподаватели. Конференции и семинары могут быть проведены в различных городах и странах, иметь различные темы и авторов. Участники могут оставлять отзывы о конференциях и семинарах, которые также хранятся в базе данных.

Одной из главных задач базы данных является автоматизация процесса участия в научных конференциях и семинарах. Для этого база данных должна позволять регистрироваться на конференции и семинары, следить за своевременной оплатой, выбирать темы и места проведения, оставлять отзывы и управлять участниками.

## Концептуальная модель предметной области

Для начала создадим таблицу пользователей. В ней будет целых 9 столбцов. Первый – id пользователя. Затем будет указываться статус пользователя: студент или преподаватель. Далее три столбца: фамилия, имя, отчество, последней из которых – необязательный. Далее идут столбцы, содержащие email и пароль, которые будет вводить пользователь при входе в систему, тип данных varchar(100), т.к. нельзя использовать кириллицу при задании этих полей. Затем идут два внешних ключа: id группы и id кафедры студента или преподавателя.

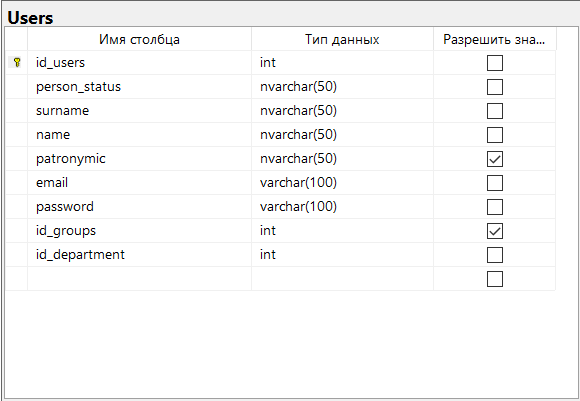


Рисунок . Таблица пользователей

Сразу закончим с сущностью пользователя (пока что, позже мы к ней вернемся) и опишем таблицы, которые соединены только с таблицей пользователей: кафедры и группы.

В таблице кафедр будет всего два столбца. Первый – id, и второй – само название кафедры.

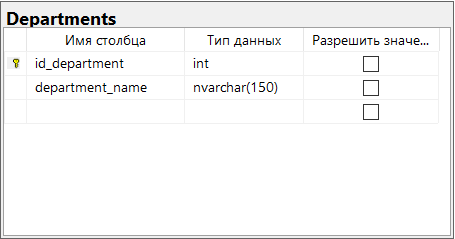


Рисунок . Таблица кафедр

В таблице группы будет также всего два столбца. Первый – id, и второй – название группы. Название группы будем считать строковой переменной, так как номера групп не во всех университетах являются просто int-переменными.

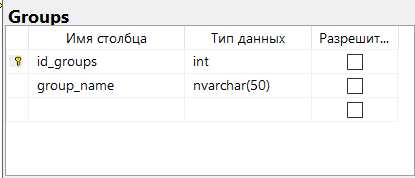


Рисунок . Таблица групп

Рассмотрим связи между уже созданными таблицами:

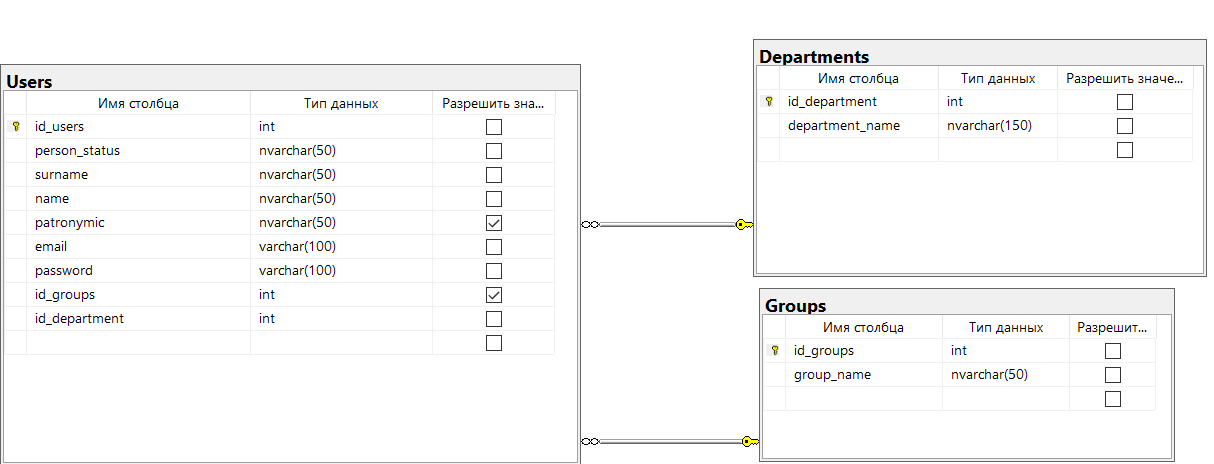


Рисунок . Связи с таблицей пользователей

У одной кафедры несколько сотрудников или студентов. В одной группе учатся несколько студентов.

Следующую сущностью, которую мы рассмотрим, будет сущность конференций и семинаров. Для удобства будем называть таблицу Conferences\_And\_Seminars сокращенно CAS. В таблице данной сущности будет 8 столбцов. Первый – номер id. Второй – название самой конференции. Так как название может быть достаточно большим, был выбран тип nvarchar(150). Затем идут столбцы времени начала и окончания мероприятия с типом, соответственно, datetime. В конце идут 4 внешних ключа: тема, на которую будет семинар/конференция, город, страна и автор, который будет проводить данное мероприятие.

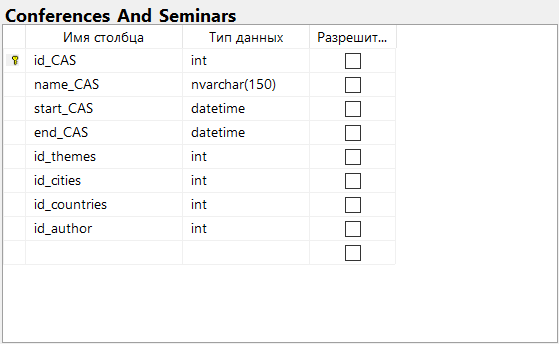


Рисунок . Таблица конференций и семинаров

Рассмотрим сущности, которые имеют внешние ключи в таблице Conferences\_And\_Seminars, сверху вниз.

Соответственно, начнем с сущности темы. Так как темы могут повторяться в разных конференциях и семинарах, а также для более удобного поиска конференций, было принято решение вынести темы в отдельную таблицу. Будем считать, что у одной конференции может быть только одна, самая главная тема. Поэтому связь с таблицей Conferences\_And\_Seminars будет один ко многим.

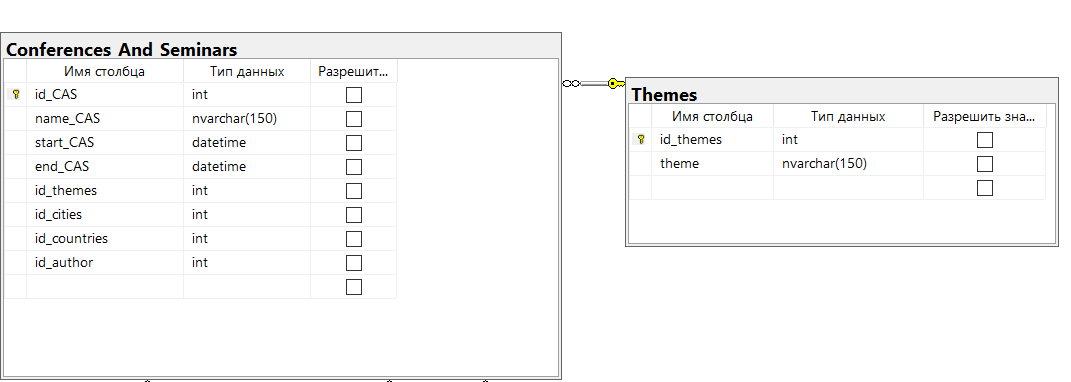


Рисунок . Добавили таблицу тем

Таблица Themes имеет всего 2 столбца: номер id и само название темы с типом nvarchar(150).

Далее добавляем сущность города. Связь будет 1 ко многим, так как в одном городе может проводиться несколько семинаров или конференций. В таблице будет 2 столбца: id и название города.

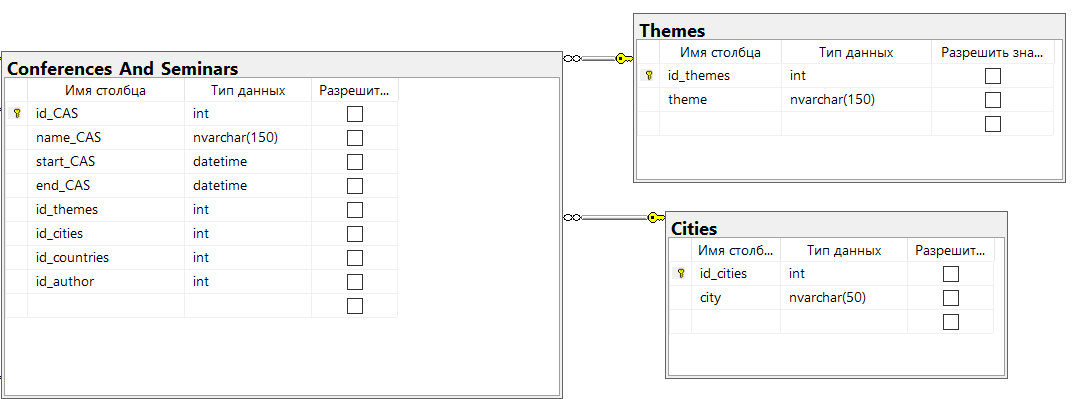


Рисунок . Добавили таблицу городов

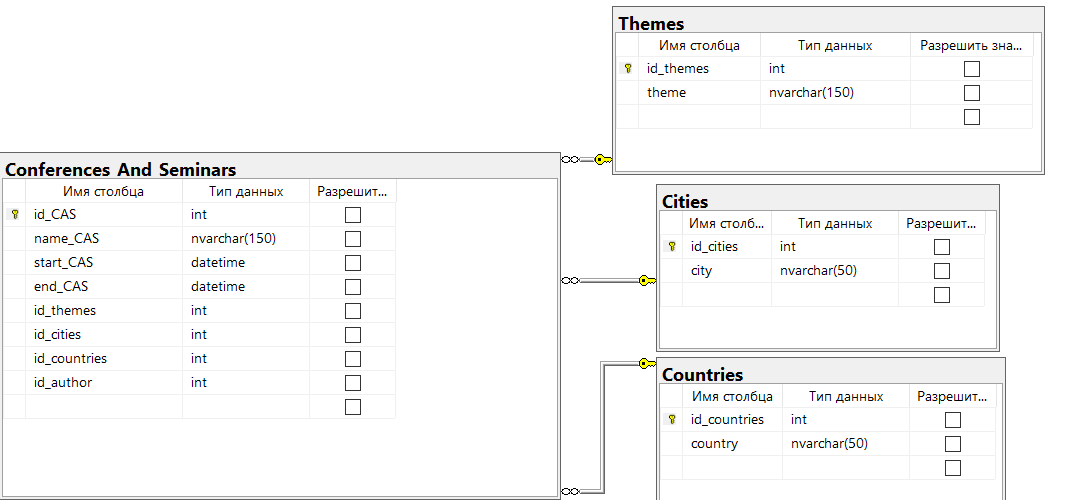
Далее аналогично добавляем таблицу стран. В таблице будет 2 столбца: id и название страны.

Рисунок . Добавили таблицу стран

Последняя сущность, которая имеет имеет связь только с таблицей Conferences\_And\_Seminars, является сущность автора. Автор это обычно ученый, который проводит семинары или научные конференции. Благодаря этой таблице будет удобно искать мероприятия, которые проводит конкретный спикер.

Так как не у всех авторов может иметься ученая степень, столбец degree был сделан необязательным.



Таким образом у нас есть две большие сущности: пользователь и конференции. Осталось их соединить и добавить еще две необходимые сущности: сведения о оплате для посещения конференций/семинаров и отзывы.

Начнем с создания сущности оплаты. В таблице будет 5 столбцов. Первый –id. Второй – стоимость, которую внёс участник конференции для оплаты конкретного мероприятия, тип переменной int: будем считать, что вносятся только целые суммы, без учета копеек. Третий – дата/время, в которые была произведена оплата с типом datetime соответственно. 4 и 5 – внешние ключи пользователя, производящего оплату и мероприятия, на которое пользователь планирует идти. Так как один пользователь может оплатить посещине нескольких конференций, а на одну конференцию может пойти несколько пользователей, то получается, что благодаря таблице Payment реализуется связь многие ко многим между таблицами Users и CAS. Или, по-другому говоря, между таблицами Users и Payment связь один ко многим, как и между таблицами CAS и Payment.

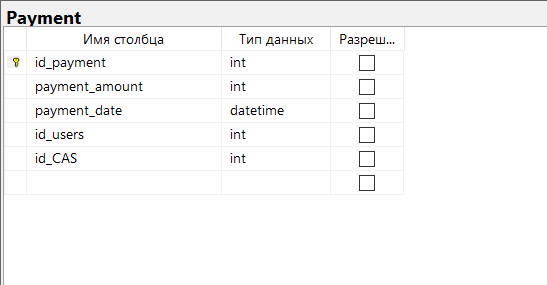


Рисунок . Таблица сведений об оплате

Теперь создадим сущность отзывов. Эта сущность будет иметь такие же связи, как и сущность оплаты, так как один пользователь может оставить несколько отзывов, а одному мероприятию могут оставить отзывы несколько пользователей. Т.е. между таблицами Users и Reviews связь один ко многим, как и между таблицами CAS и Reviews. В таблице будет 4 столбца: id, сам отзыв, а также два внешних ключа с таблицами пользователей и конференциями. В столбце отзывов выбран тип nvarchar(MAX), так как отзывы серьезных критиков могут быть достаточно больши. В будущем возможно уменьшение максимального размера, в зависимости от объемов отзывов.

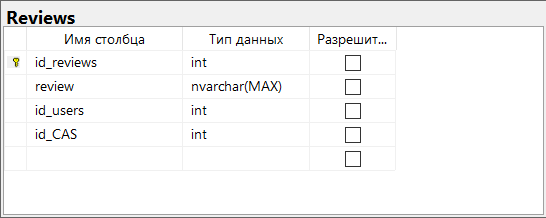


Рисунок . Таблица отзывов

Теперь, все что осталось – соединить таблицы Conferences\_And\_Seminars и Users связью многие ко многим, так как на одну конференцию или семинар могут пойти несколько студентов и учителей, а один студент/учитель может посетить несколько мероприятий. Для этого создадим дополнительную таблицу Users\_Conferences\_And\_Seminars. В ней будет 3 столбца: её id, внешние ключи таблиц Users и Conferences\_And\_Seminars.

Таким образом мы получаем конечную ER-диаграмму базы данных:

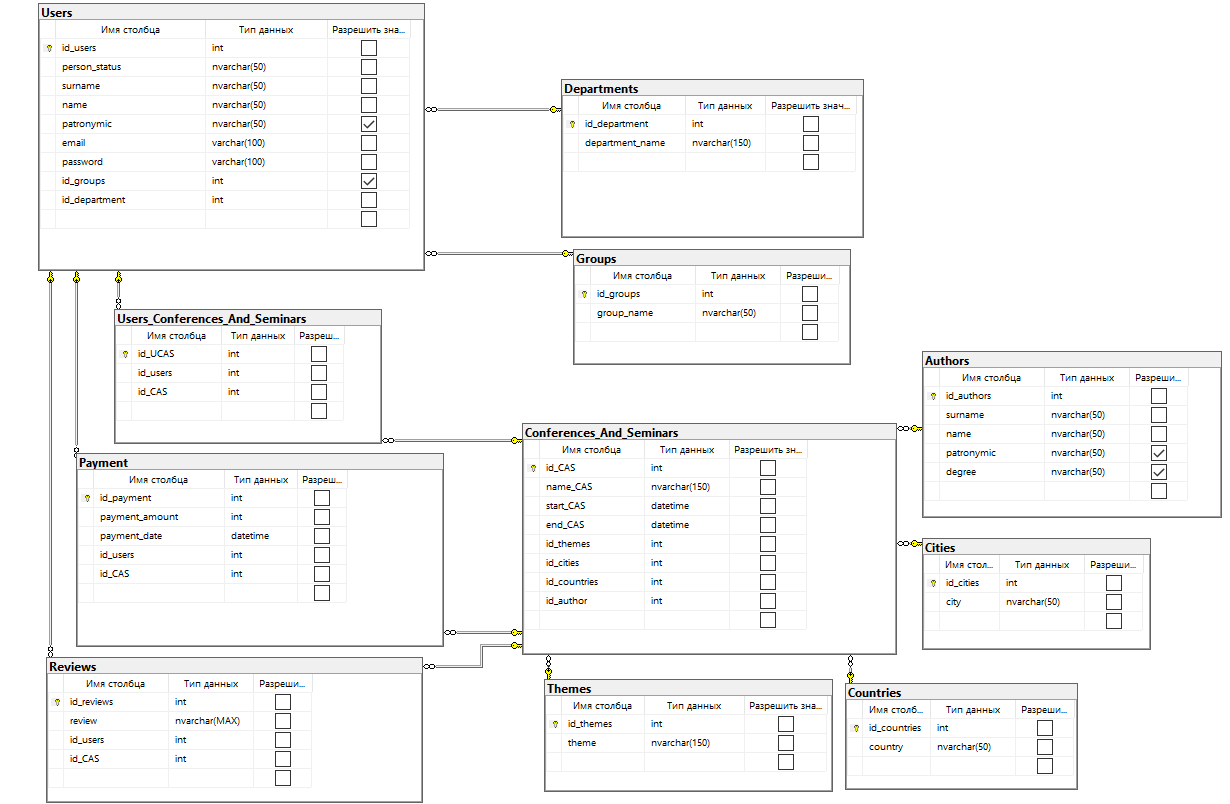


Рисунок . Финальная ER-диаграмма базы данных

Также стоит отметить, что во всех созданных таблицах на первичный ключ был выставлен автоинкремент (1, 1) для автоматического заполнения id-номера:

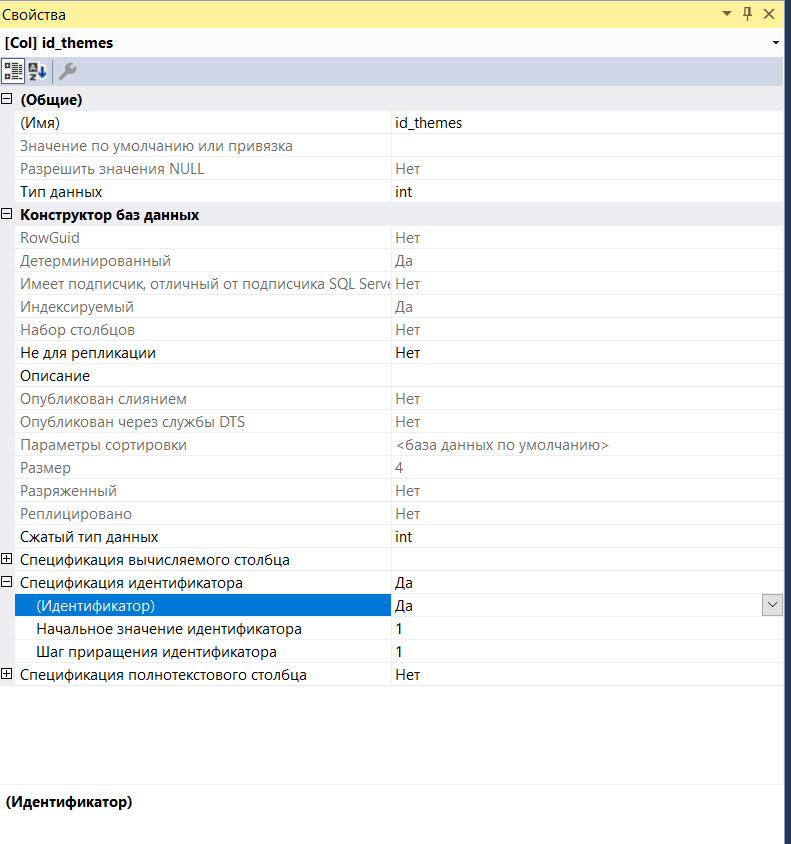


Рисунок . Примере выставления автоинкремента на таблицы Themes

## Построение представлений (view)

Рассмотрим два представления к уже созданной базе данных:

1. Вывод данных о всех студентах: студент, фамилия, имя, отчество, email, название кафедры, название группы.
2. Вывод всех конференций и семинаров: называние, время начала, время окончания, название темы, город, страна, фамилия автора.

Начнем с представления, выводящего информацию о студентах:

CREATE OR ALTER VIEW FIRST\_VIEW AS

SELECT person\_status, surname, name, patronymic,

email, group\_name, department\_name FROM Users

JOIN Groups ON Users.id\_groups = Groups.id\_groups

JOIN Departments ON Users.id\_department = Departments.id\_department

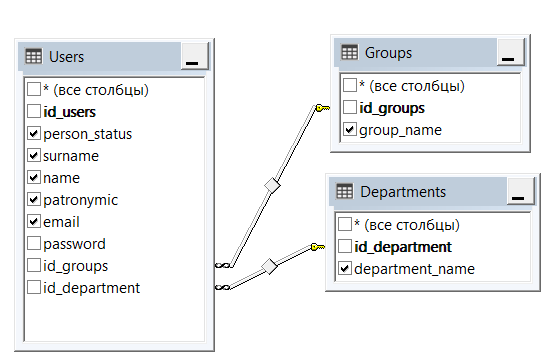


Рисунок . Схема представления информации о пользователях

Теперь создадим второе представление, содержащее информацию о конференциях и семинарах:

CREATE OR ALTER VIEW SECOND\_VIEW AS

SELECT name\_CAS, start\_CAS, end\_CAS, theme, country, city, surname FROM Conferences\_And\_Seminars

JOIN Themes ON Conferences\_And\_Seminars.id\_themes = Themes.id\_themes

JOIN Countries ON Conferences\_And\_Seminars.id\_countries = Countries.id\_countries

JOIN Cities ON Conferences\_And\_Seminars.id\_cities = Cities.id\_cities

JOIN Authors ON Conferences\_And\_Seminars.id\_author = Authors.id\_authors

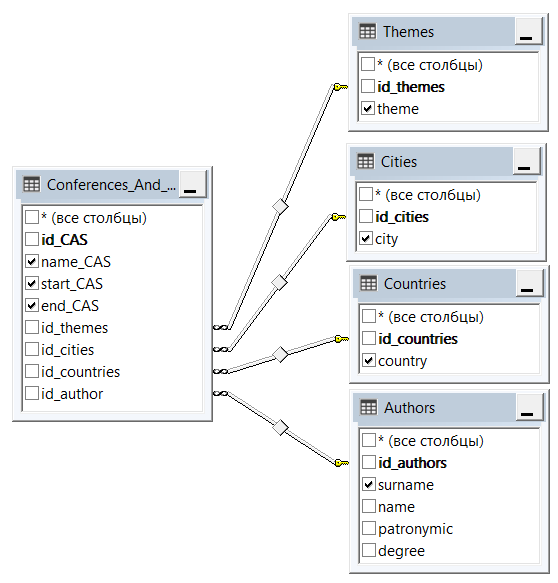


Рисунок . Схема представления информации о конференциях и семинарах

# Реализация базы данных

## Заполнение данными

Заполним все созданные таблицы с помощью SQL-запросов.

Таблица Themes:

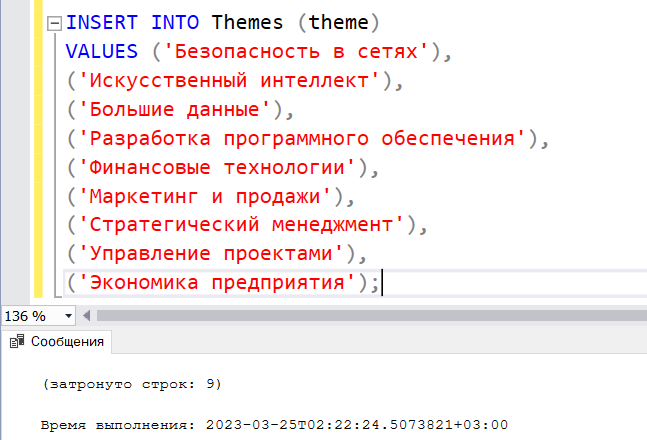


Таблица Authors:



Таблица Countries:

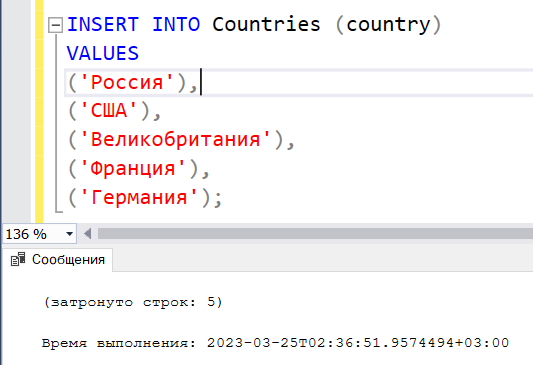


Таблица Cities:

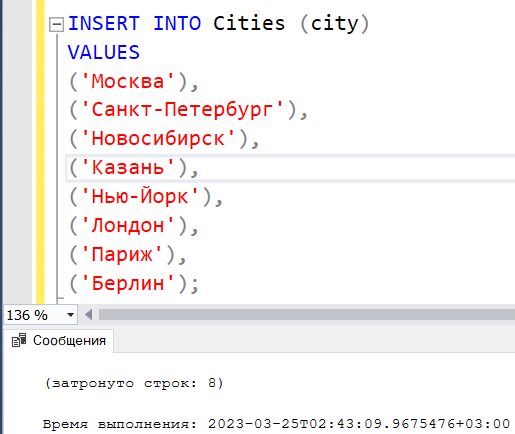


Таблица Conferences\_And\_Seminars:

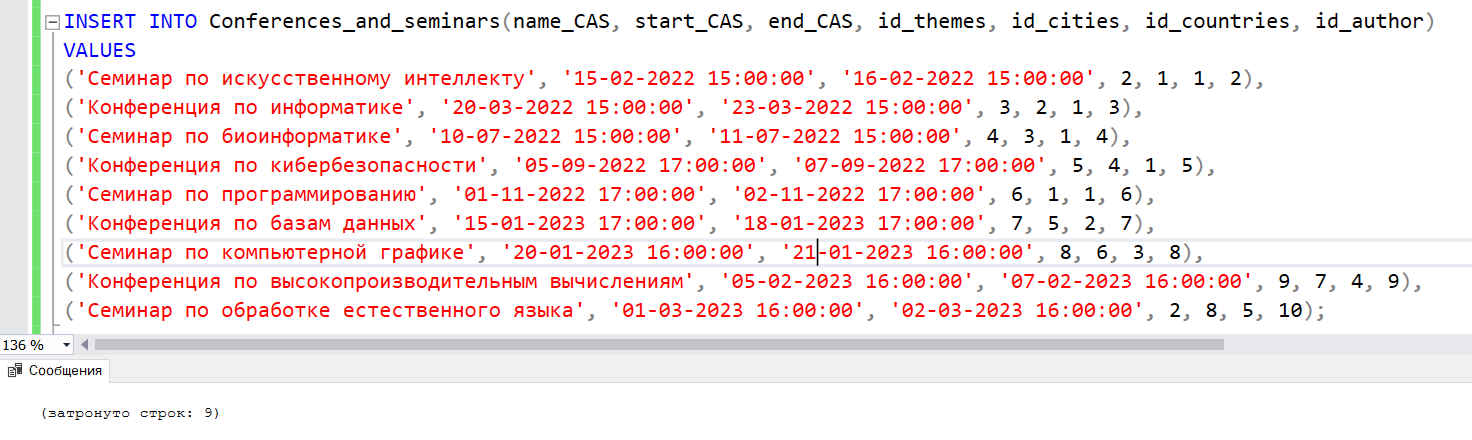


Таблица Groups:

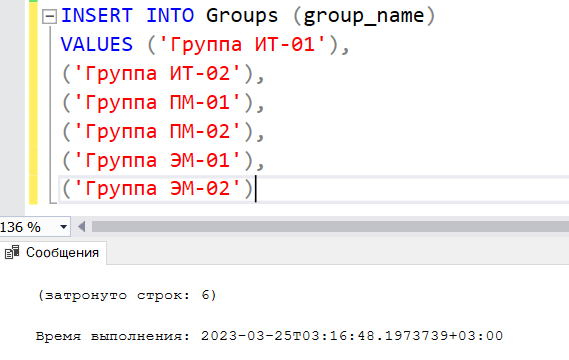


Таблица Departments:

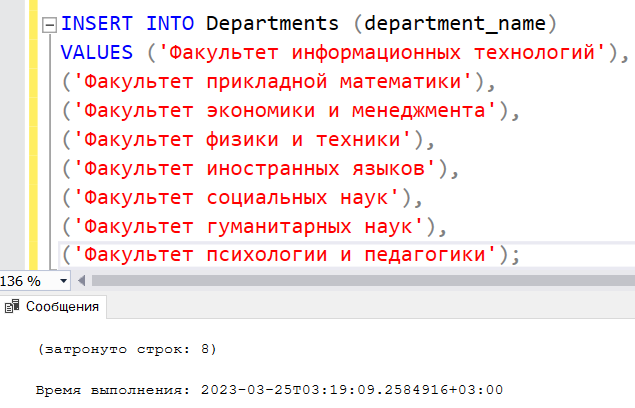


Таблица Users:

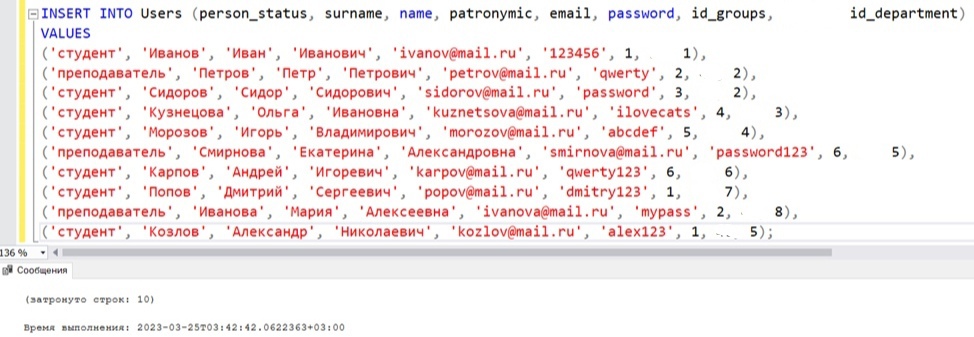


Таблица Payment:

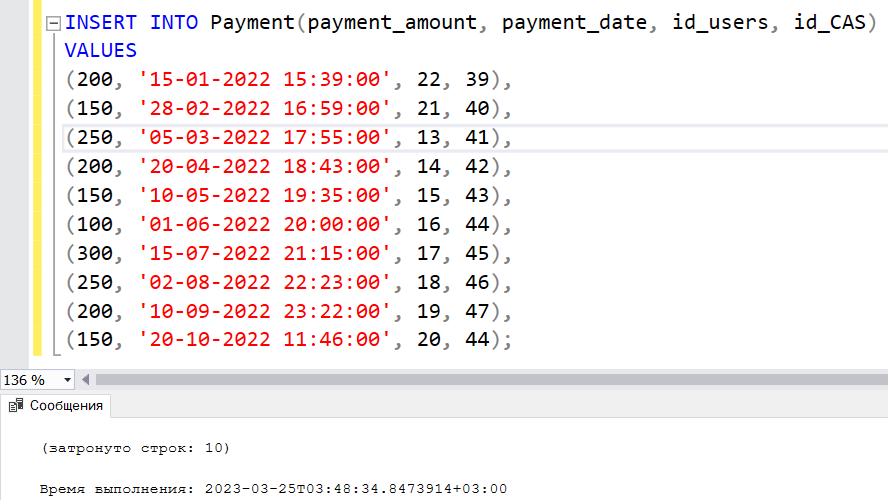


Таблица Reviews:

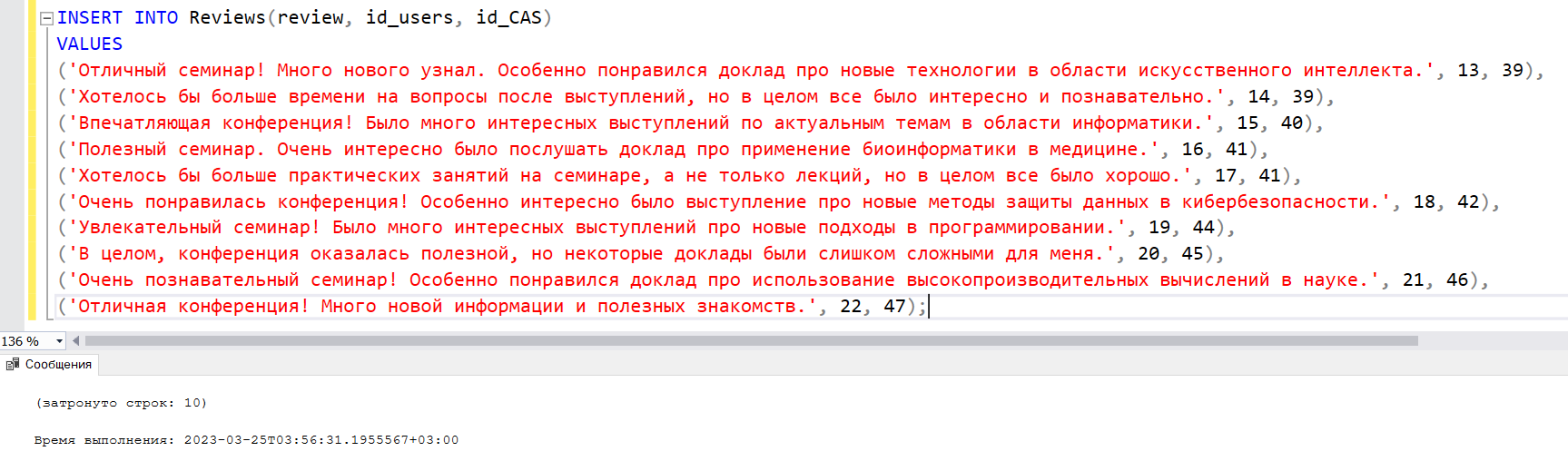
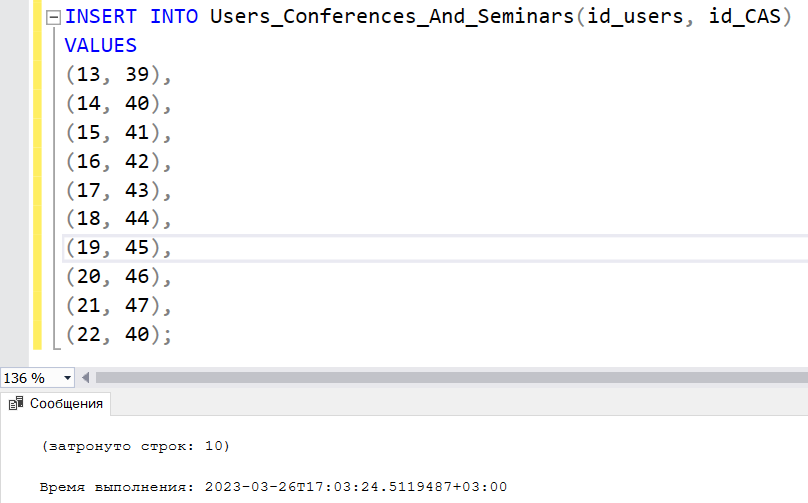


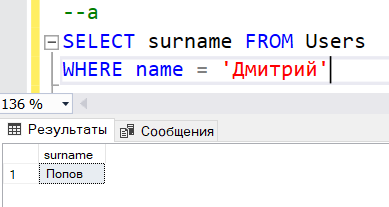
Таблица Users\_Conferences\_And\_Seminars:



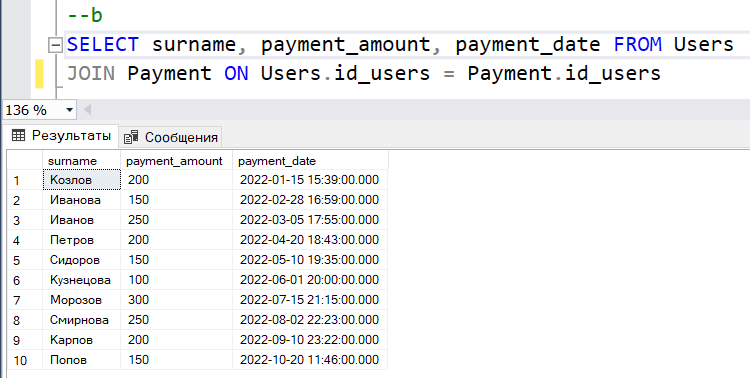
## SQL-запросы. Команда SELECT

Напишем несколько SELECT запросов к созданной базе данных.

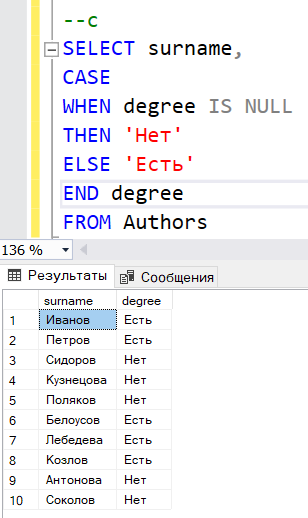
1. Первым запросом будет SELECT с использованием WHERE. Выведем фамилии пользователей с именем Дмитрий.



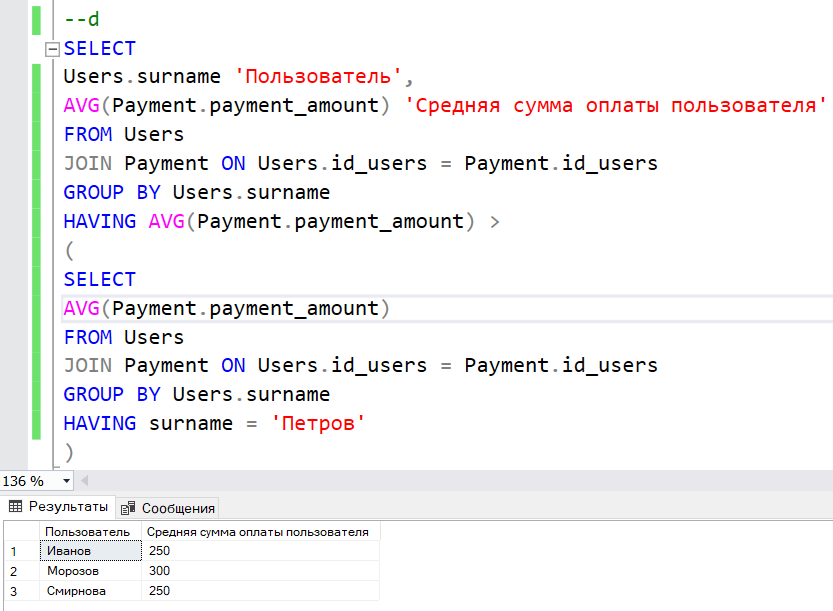
1. Напишем запрос с использованием INNER JOIN. Выведем фамилии пользователей, их сумму оплаты и дату оплаты.



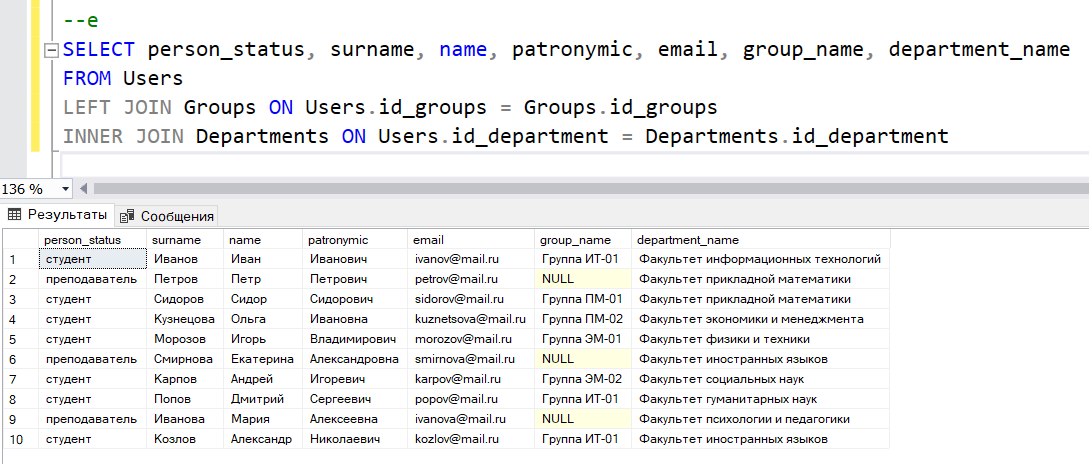
1. Напишем запрос с использованием CASE. Выведем фамилии авторов и “Есть”, если у них есть любая степень и “Нет”, если степени нет.



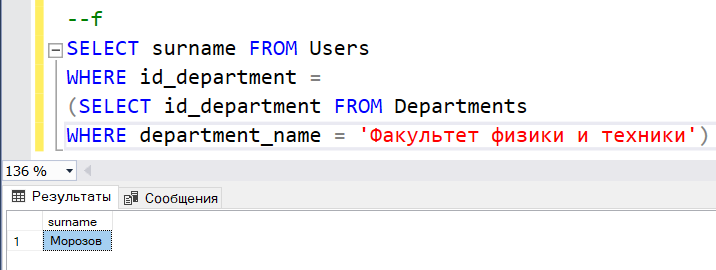
1. Составим запрос с использованием группировок, группировочных функций и условий на группы (HAVING). Выведем пользователей, средняя сумма потраченных рублей на конференции > средней суммы потраченных рублей на конференции пользователем с фамилией “Петров”



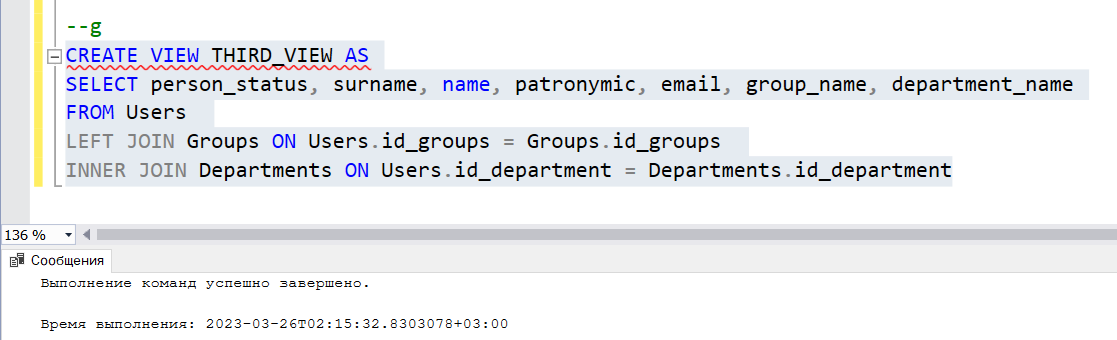
1. Составим запрос с использованием LEFT JOIN. Так как у преподавателей нет групп, в отличии от студентов, при объединении обычным INNER JOIN таблиц Users и Groups, мы получаем только студентов. Для вывода и студентов, и преподавателей, нужно использовать LEFT JOIN.



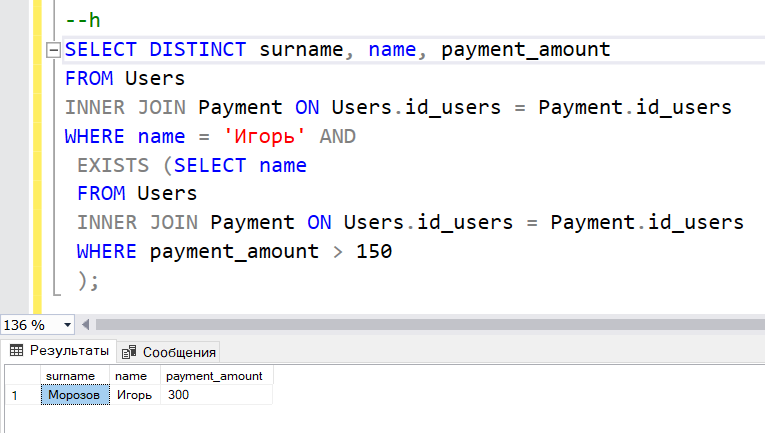
f. Составим запрос с использованием вложенного подзапроса SELECT. Выведем фамилии всех студентов и преподавателей с факультета “ Факультет физики и техники”.



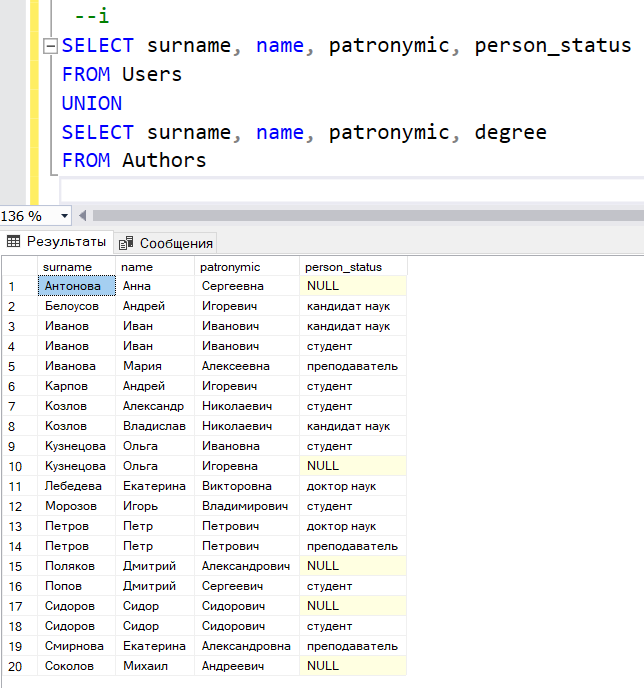
1. Создадим представление запроса из пункта “e”.



1. Составим запрос с использованием EXISTS. Выведем все оплаты конференций, суммы которых > 150 пользователей с именем Игорь.



1. Составим Запрос с использованием UNION. В первых трех столбцах выведем ФИО студентов, преподавателей и авторов, а в четвертом – значение студент/преподаватель для студентов и преподавателей, а для авторов - их ученые степени.



# Заключение

В результате проведенного исследования была разработана база данных, которая позволяет автоматизировать процесс участия сотрудников и студентов вуза в научных конференциях и семинарах. Такая база данных сделает удобную регистрацию на конференции и семинары, позволит управлять оплатой, выбирать темы и места проведения, оставлять отзывы и управлять участниками. Это сократит время и снизит затраты на организацию и участие в научных мероприятиях, а также улучшит качество научной работы за счет более удобной и эффективной системы управления научными конференциями и семинарами. Кроме того, такая база данных может быть полезна не только для университетов, но и для других организаций, занимающихся научной деятельностью.

# Список литературы

https://learn.microsoft.com/ru-ru/sql/sql-server/?view=sql-server-ver15