**МИНОБРНАУКИ РОССИИ**

**федеральное государственное бюджетное образовательное учреждение высшего образования**

**«САНКТ-ПЕТЕРБУРГСКИЙ ГОСУДАРСТВЕННЫЙ ЭКОНОМИЧЕСКИЙ**



**УНИВЕРСИТЕТ»**

Факультет информатики и прикладной математики

Кафедра прикладной математики и экономико-математических методов

**КУРСОВАЯ РАБОТА** по дисциплине:

# «Системы управления базами данных»

Тема: «Разработка базы данных для школы»

Направление: 01.03.02 Прикладная математика и информатика

Направленность: Прикладная математика и информатика в экономике и управлении

Обучающийся: Попова Софья Ивановна

Группа: ПМ-1901 Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Проверил: Иванова Виктория Валерьевна

Должность: доцент

Оценка: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ Дата: \_\_\_\_\_\_\_\_\_

Подпись: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Санкт-Петербург

2021

СОДЕРЖАНИЕ

[«Системы управления базами данных» 1](#_Toc91619047)

[1. ХАРАКТЕРИСТИКА ПРЕДМЕТНОЙ ОБЛАСТИ 3](#_Toc91619048)

[1.1. Описание предметной области 3](#_Toc91619049)

[1.2. Определение проблемы, для которой будет создаваться продукт 3](#_Toc91619050)

[2. РАЗРАБОТКА ТЕХНИЧЕСКОГО ПРЕДЛОЖЕНИЯ 4](#_Toc91619051)

[2.1. Формулировка видения и назначения 4](#_Toc91619052)

[2.2. Бизнес-требования 4](#_Toc91619053)

[2.3. Концепция продукта 5](#_Toc91619054)

[2.4. Анализ требований пользователя с помощью User Stories 6](#_Toc91619055)

[3. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ 7](#_Toc91619056)

[3.1. Анализ информационных потоков 7](#_Toc91619057)

[3.2. Определение информационных объектов 7](#_Toc91619058)

[3.3. Нормализация 8](#_Toc91619059)

[3.4. Информационно-логическая модель 10](#_Toc91619060)

[4. Реализация в программе Mysql Workbench 11](#_Toc91619061)

[4.1 Создание базы данных 11](#_Toc91619062)

[4.2. Triggers 13](#_Toc91619063)

[4.3 Views 14](#_Toc91619064)

[4.4 Stored Procedures 16](#_Toc91619065)

[5. Заключение 25](#_Toc91619066)

# 1. ХАРАКТЕРИСТИКА ПРЕДМЕТНОЙ ОБЛАСТИ

## 1.1. Описание предметной области

Университет – это сложная структура со множеством объектов. В ней присутствует множество связанных между собой элементов, таких как преподаватели, которые могут как просто вести предметы, сами предметы разные для разных направлений и семестров. Студенты, которые учатся, сдают экзамены и получают стипендию. Также в базе данных содержится вся справочная информация о них. И предметы, у которых есть названия, количество часов, семестр и преподаватели. которые их ведут.

Конечно, можно добавить ещё множество других элементов, например, кружки, библиотека, текущие оценки, посещаемость, но я рассматриваю базовое функционирование университета.

## 1.2. Определение проблемы, для которой будет создаваться продукт

Университет должен функционировать без сбоев, поэтому очень важна четкая структура и понимание, кто будет вести предметы, как учатся студенты, как начисляется стипендия, как составляется расписание сессии.

# 2. РАЗРАБОТКА ТЕХНИЧЕСКОГО ПРЕДЛОЖЕНИЯ

## 2.1. Формулировка видения и назначения

Рассматриваемому университету необходима автоматизация учебного процесса, чтобы можно было отслеживать успеваемость студентов, преподавательский состав и сделать работу университета более эффективной.

Построенная мной база данных поможет быстро вывести список любой группы или информацию по определенному студенту и много других функций.

## 2.2. Бизнес-требования

Целью проекта является создание удобной и понятной базы данных для обеспечения функционирования любого университета.

Такая система сильно упростит внутренние процессы и поможет университету работать более оптимально.

## 2.3. Концепция продукта

Продукт разрабатывается для преподавателей/студентов/работников вуза. Рассмотрим требования к разрабатываемой системе:

Таблица 1 – Требования пользователей к разрабатываемой системе

|  |  |
| --- | --- |
| For | *Преподаватель/студент/деканат* |
| Who | *Хочет получить информацию про университет* |
| The (product name) | *Приложение Университет* |
| That | *Является средой планирования и организации рабочего процесса, которое позволяет согласовать график работы в зависимости от нагрузки в группе и личной занятости преподавателя, информировать об академических задолженностях, а также об отсутствии таковых, управлять сведениями о выполненной работе и студентах.*  *Приложение позволяет формировать оптимальный график работы. Система способствует планированию личного рабочего пространства преподавателя, мотивирует к повышению качества работы.* |
| Unlike | *Существующей организации процесса, при который преподаватель берет слишком много нагрузки. Или у группы очень сложная программа.* |
| Our product | *Оптимизирует работу преподавателя, даст ему комфортную рабочую среду, сократит расходы государства (на…), повысит лояльность работников* |

Таким образом, данная система оптимизирует работу, обеспечивая здоровую рабочую атмосферу.

## 2.4. Анализ требований пользователя с помощью User Stories

|  |  |
| --- | --- |
| Кто | Сотрудник деканата |
| Что | Отслеживание должников |
| Зачем | Чтобы обнулить стипендию, предупредить студента о последствиях накопления долгов в учебе |

|  |  |
| --- | --- |
| Кто | Сотрудник деканата |
| Что | Отслеживание иногородних студентов |
| Зачем | Чтобы предоставлять общежитие и возмещать затраты на дорогу во время карантина |

|  |  |
| --- | --- |
| Кто | Сотрудник деканата |
| Что | Отчисление студентов |
| Зачем | Чтобы студенты могли поступить в другой вуз и не числились просто так. |

|  |  |
| --- | --- |
| Кто | Сотрудник деканата |
| Что | Начисление стипендии |
| Зачем | Чтобы мотивировать студентов получать хорошие оценки и наградить за успехи в учебе |

|  |  |
| --- | --- |
| Кто | Студент |
| Что | Просмотр оценок |
| Зачем | Чтобы быть в курсе учебного процесса и своевременно принимать меры |

|  |  |
| --- | --- |
| Кто | Студент |
| Что | Возможность найти контакты преподавателя |
| Зачем | Чтобы решать сложности в учебе напрямую |

|  |  |
| --- | --- |
| Кто | Студент |
| Что | Просмотр расписания сессии |
| Зачем | Чтобы своевременно начать подготовку и спланировать свое время |

|  |  |
| --- | --- |
| Кто | Преподаватель |
| Что | Просмотр оценок студентов |
| Зачем | Чтобы быть в курсе учебного процесса и следить, насколько хорошо усваивают материал студенты. |

|  |  |
| --- | --- |
| Кто | Преподаватель |
| Что | Просмотр расписания сессии |
| Зачем | Чтобы удобно составлять свое расписание |

# 3. ПРОЕКТИРОВАНИЕ ИНФОРМАЦИОННОГО ОБЕСПЕЧЕНИЯ

## 3.1. Анализ информационных потоков

В создаваемой базе данных на вход подается информация о студентах, содержащее фио, дату рождения, город проживания и курс обучения.

В других таблицах хранится информация о преподавателях, оценках, предметах и другая справочная информация.

## 3.2. Определение информационных объектов

Для создаваемой базы данных необходима информация о преподавателях: ID, ФИО. Тогда сущность учителя будет выглядеть:

Таблица 2 – Сущность учитель

|  |
| --- |
| Teacher |
| ID учителя |
| Фамилия учителя |
| Имя учителя |
| Отчество учителя |
| Email |
| Нагрузка |

Для создаваемой базы данных необходима информация о студентах: ID , ФИО, дата рождения, курс и группа . Тогда сущность студента будет выглядеть: (Таблица 3).

Таблица 3 – Сущность Студент

|  |
| --- |
| Student |
| ID студента |
| Фамилия |
| Имя |
| Ивановна |
| Курс |
| Дата рождения |
| Группа |
| Стипендия |

В университете также есть предметы, они содержат: ID акции, ID, название предмета, часы, которые отводятся на его чтение и семестр, в котором он читается.

Таблица 4 – Сущность предмет

|  |
| --- |
| SubjectOfStudy |
| ID предмета |
| Название предмета |
| Часы |
| Семестр |

## 3.3. Нормализация

Нормализация — процесс уменьшения избыточности информации в таблицах реляционной БД, то есть построение оптимальной структуры таблиц и связей.

Для получения информации об успеваемости и датах проведения экзаменов.

Таблица 5 – Сущность экзамены

|  |
| --- |
| MarkOfExam |
| ID экзамена |
| ID студента |
| ID предмета |
| Оценка |
| Дата экзамена |

Также в базе данных есть информация о преподавании: какой преподаватель ведет каждый предмет.

Таблица 6 – Сущность преподавание

|  |
| --- |
| Teaching |
| ID учителя |
| ID предмета |
| Группа |

Также в базе данных есть информация о группах: номер и название.

|  |
| --- |
| studentgroup |
| ID группы |
| название |

## 3.4. Информационно-логическая модель

Информационно-логическая модель отображает данные предметной области в виде совокупности информационных объектов и связей между ними. Построим ее по полученным нормализованным сущностям.



Рисунок 1 – Информационно-логическая модель

Таким образом, построенная информационно-логическая модель отражает сущности базы данных в третьей нормальной форме и связи между ними.

# 4. Реализация в программе Mysql Workbench

## 4.1 Создание базы данных

В первую очередь я создаю свою базу данных – university.

Далее создаю с ней таблицы, которые описаны выше. Код создания таблиц:

CREATE TABLE Teacher

(

TeacherID INT AUTO\_INCREMENT,

Lastname VARCHAR(20),

Firstname VARCHAR(20),

Patronymic VARCHAR(20),

Email VARCHAR(20),

TeacherHour int,

PRIMARY KEY(TeacherID));

CREATE TABLE Student

( StudentID INT AUTO\_INCREMENT,

Lastname VARCHAR(20),

Firstname VARCHAR(20),

Patronymic VARCHAR(20),

Kurs INT,

DataOfBirth DATE,

City VARCHAR(200),

GroupStudy VARCHAR(20),

PRIMARY KEY(StudentID));

CREATE TABLE SubjectOfStudy

( SubjectID INT AUTO\_INCREMENT,

SubjectName VARCHAR(20),

SubjHour int,

Semester int,

PRIMARY KEY(SubjectID));

CREATE TABLE MarkOfExam

( ExamID int ,

StudentID int,

FOREIGN KEY (StudentID) REFERENCES Student (StudentID),

SubjectID int,

FOREIGN KEY (SubjectID) REFERENCES SubjectOfStudy(SubjectID),

Mark int ,

ExamData int ,

primary key(ExamID, StudentID,SubjectID));

CREATE TABLE Teaching

(

TeacherID int,

FOREIGN KEY (TeacherID) REFERENCES Teacher (TeacherID),

SubjectID int,

GroupStudent int,

FOREIGN KEY (SubjectID) REFERENCES SubjectOfStudy(SubjectID),

primary key( TeacherID,SubjectID));

Второй шаг – заполнение таблиц данными, чтобы можно было проверять корректность запросов и процедур. Данные для заполнения я брала в генераторе случайных данных людей или придумывала сама.

Третий шаг – реализация процедур/вью/триггеров, которые обеспечат правильное и удобное использование моей базы данных.

## 4.2. Triggers

Триггер — это хранимая процедура, которая не вызывается непосредственно, а исполняется при наступлении определенного события (вставка, удаление, обновление строки ).

1. Mark

Это триггер для таблицы markodexam, в которой хранятся оценки за экзамены. Он помогает не допускать ошибок в выставлении оценок по 5-ти балльной системе, то есть задает ограничение : 0 < оценка < 6.

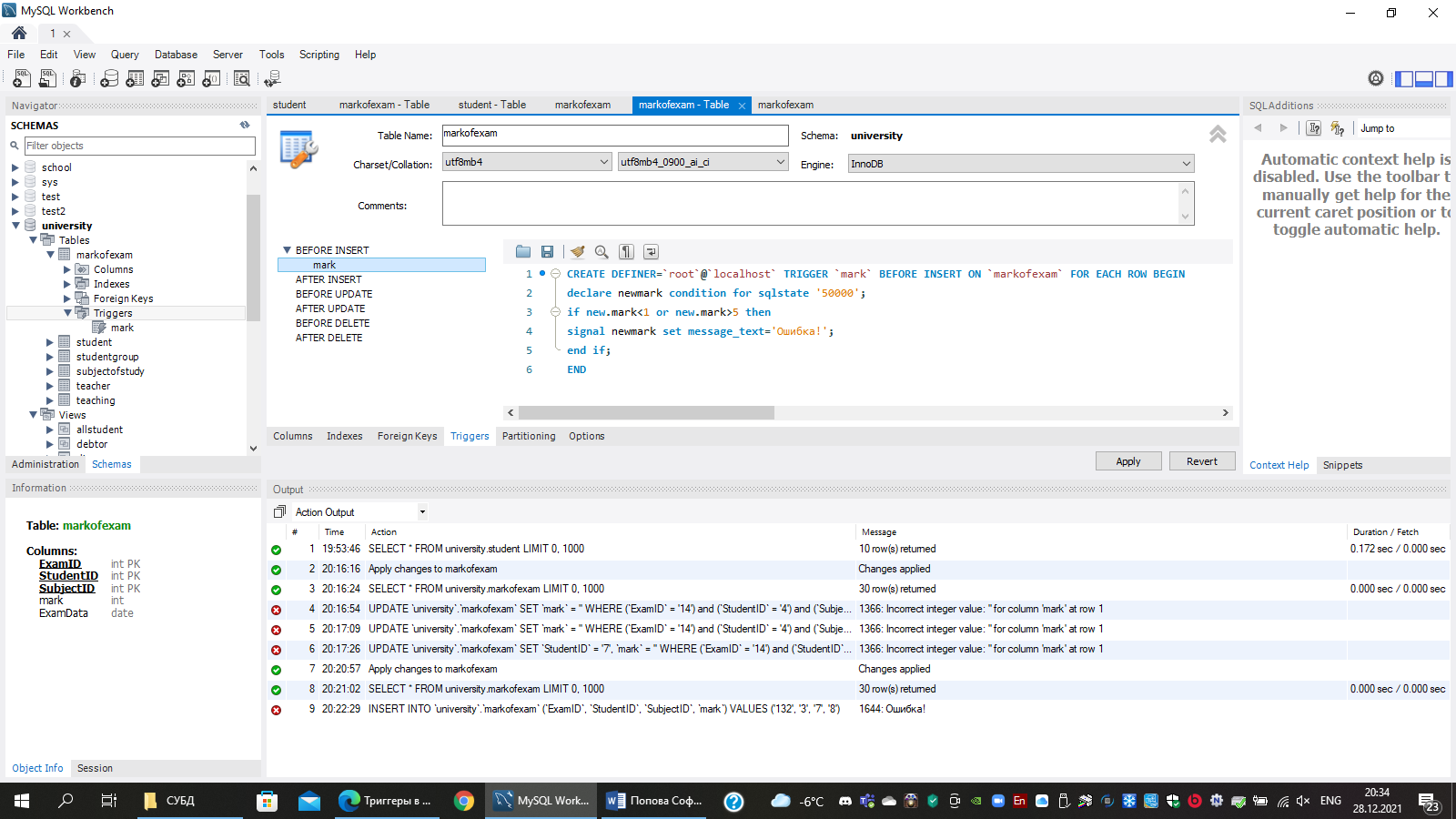


Рисунок 2 – Триггер mark

1. Student\_BEFORE\_INSERT

Это триггер, относящийся к таблице student, в которой хранится информация о студентах. Он отвечает за начисление каждому поступившему студенту стандартной стипендии в размере 2100.

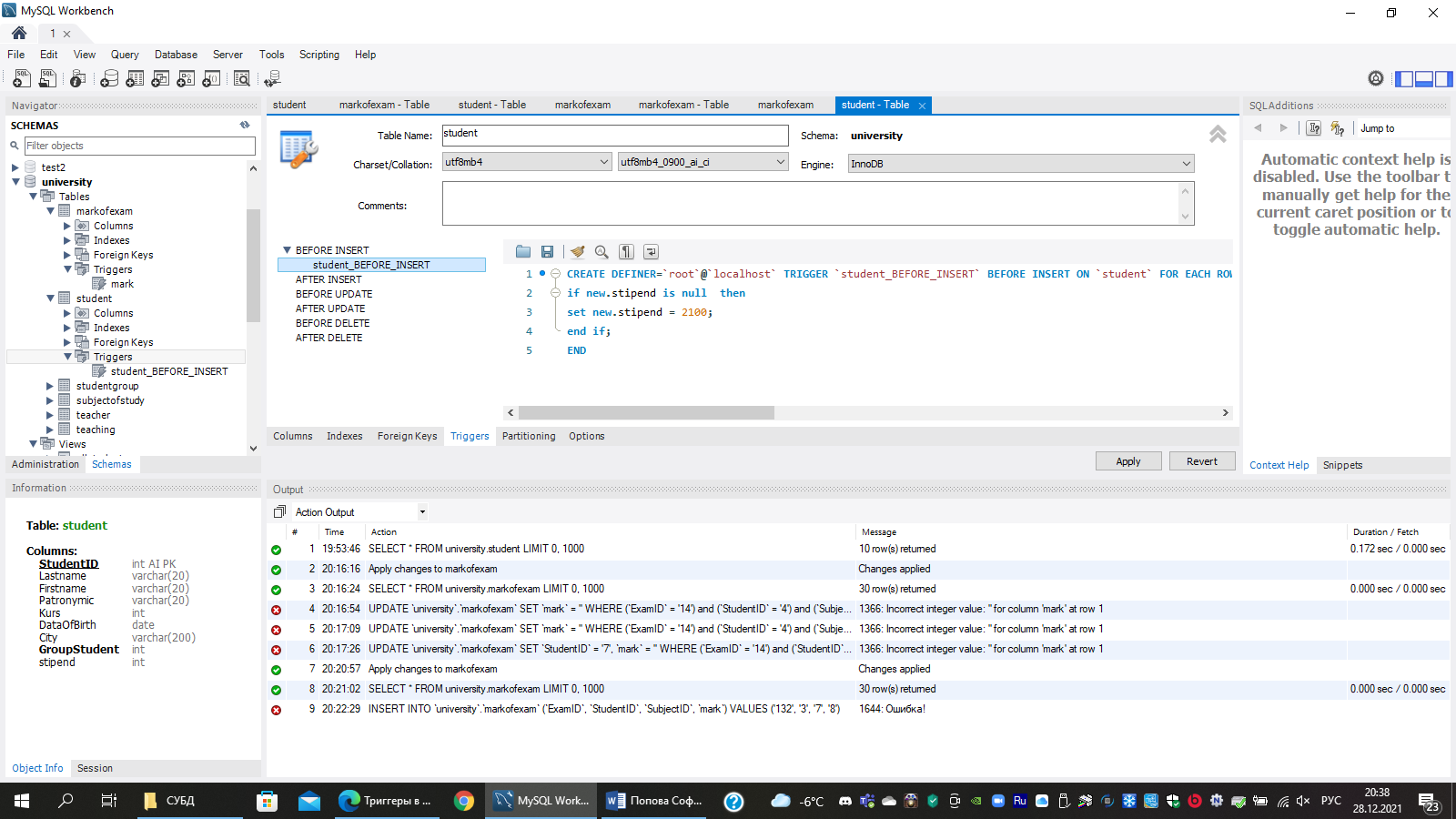


Рисунок 3 – Триггер student

## 4.3 Views

Представление (VIEW) - обьект данных, который не содержит никаких данных его владельца. Это - тип таблицы, чье содержание выбирается из других таблиц с помощью выполнения запроса. Поскольку значения в этих таблицах меняются, то автоматически, их значения могут быть показаны представлением.

1. AllStudents

Показывает всех студентов нашего вуза и всю информацию о них:

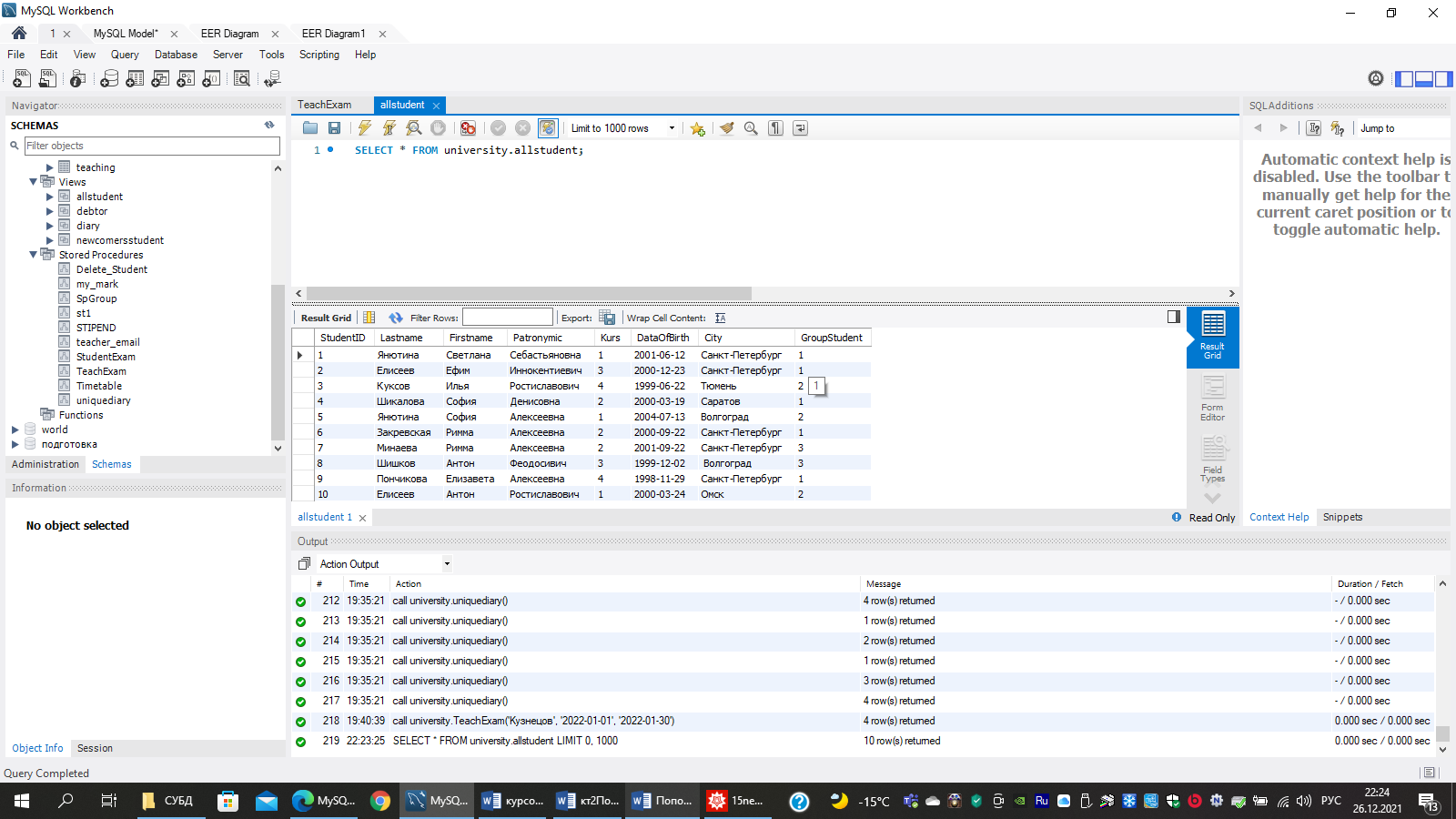


Рисунок 4 – VIEW AllStudents

1. Debtor

Показывает всех студентов с долгами в учебе:

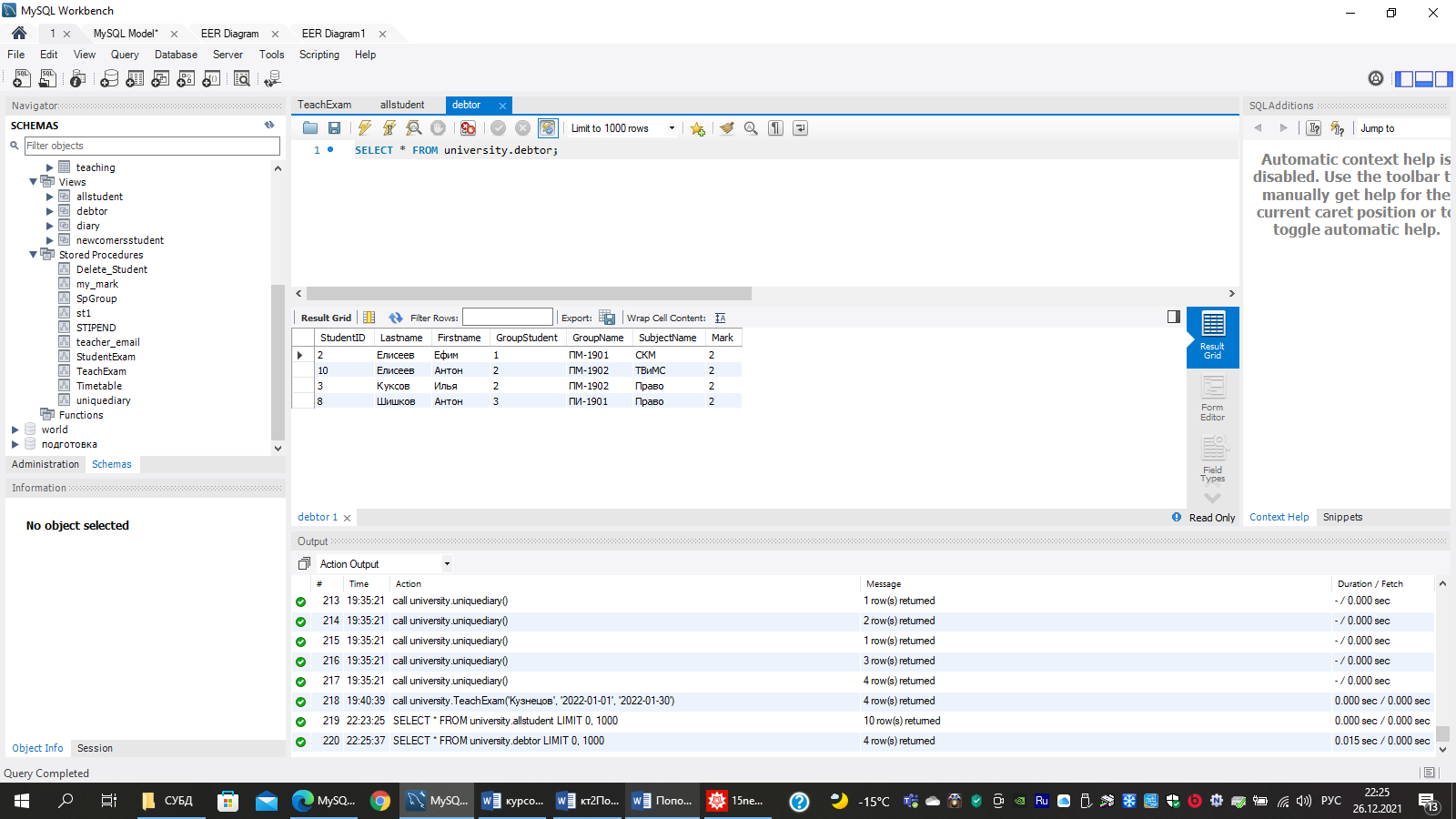


Рисунок 5 – VIEW debtor

1. Diary

Показывает всех студентов и все оценки за все экзамены:

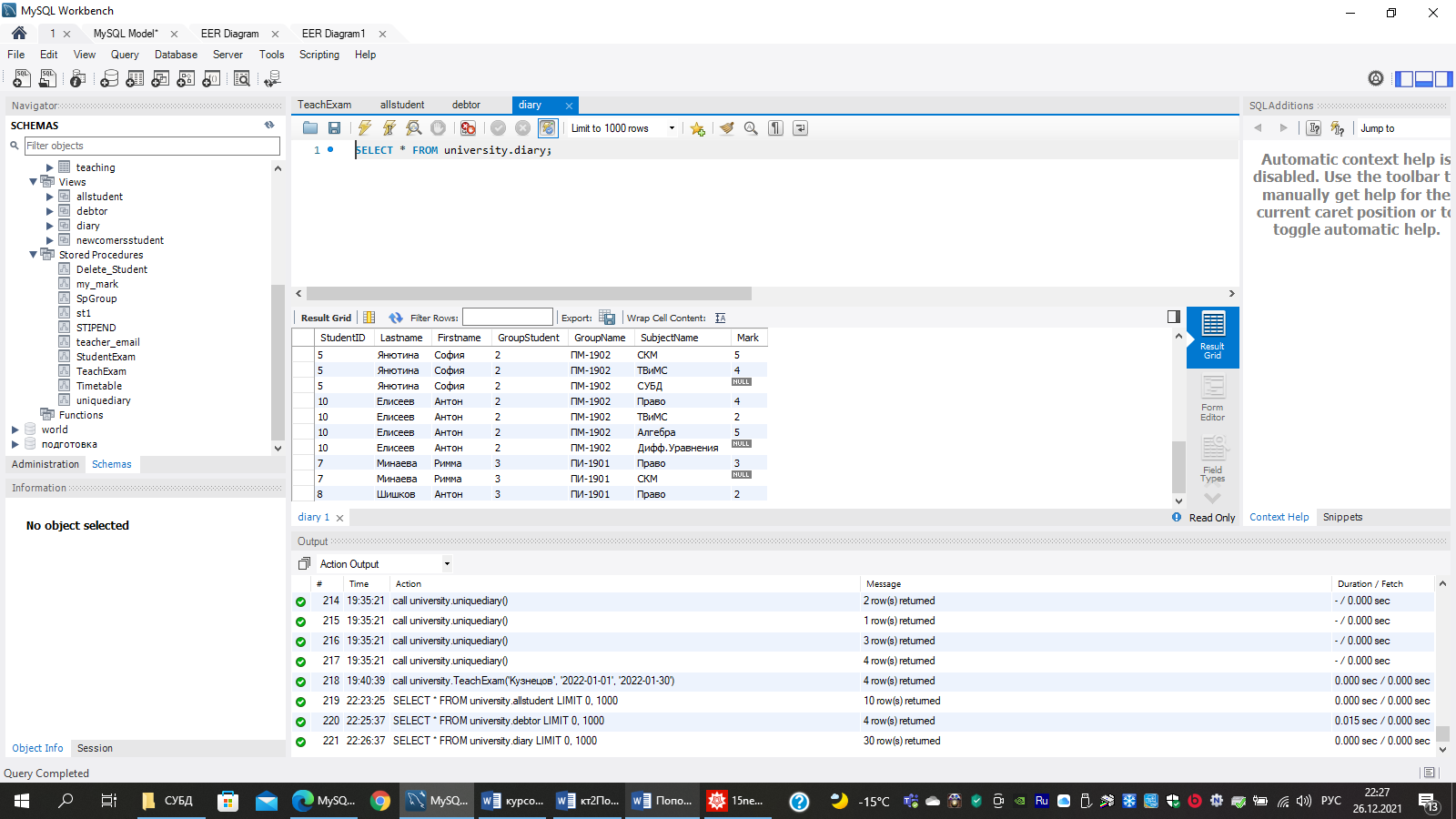


Рисунок 6 – VIEW Diary

1. Newcommersstudent

Показывает список иногородних студентов:

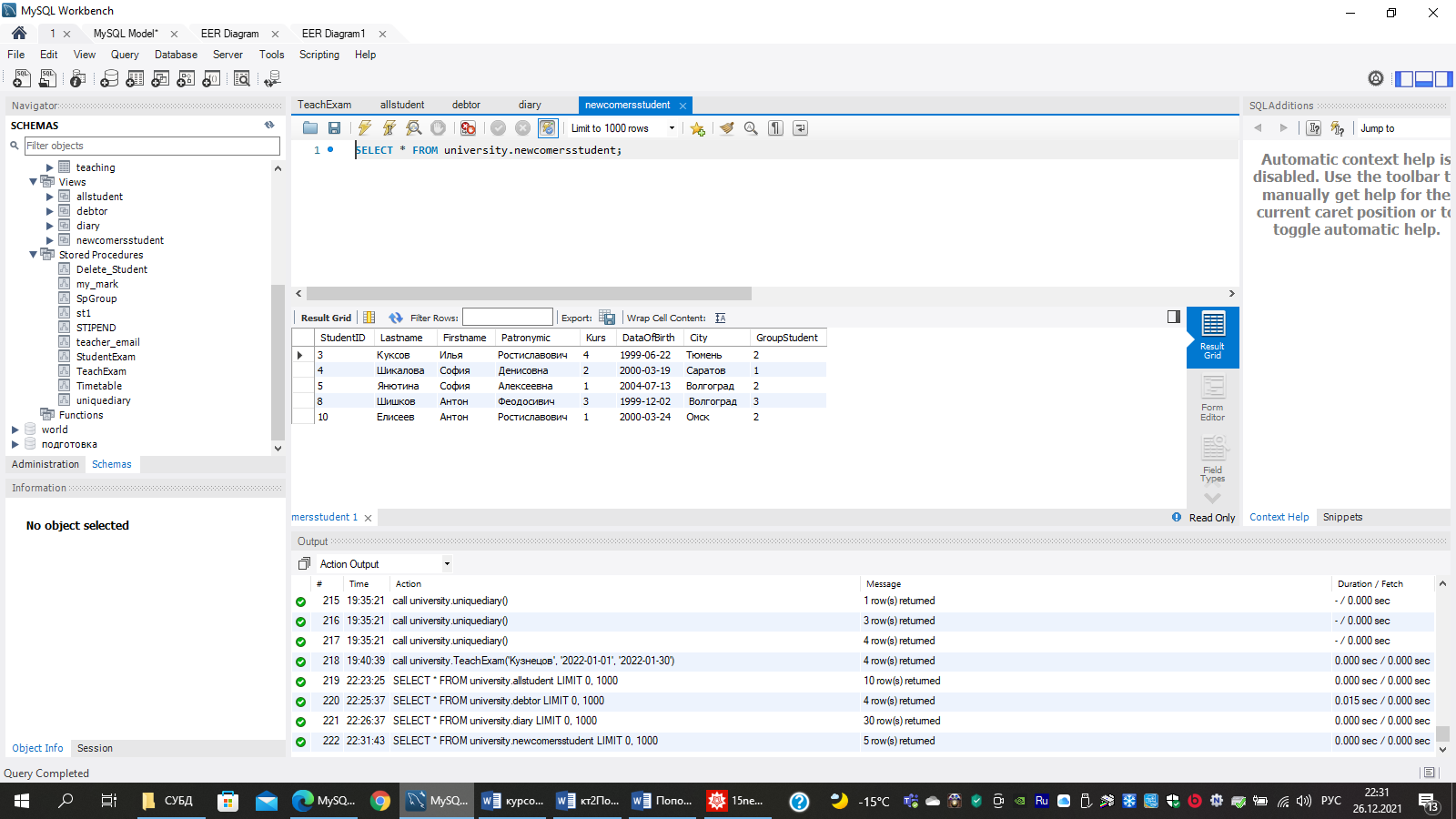


Рисунок 7 – VIEW Newcommersstudent

## Stored Procedures

Хранимая процедура (Stored Procedure) — это набор команд, хранимый на сервере и выполняемый как единое целое. Хранимые процедуры являются механизмом, с помощью которого можно создавать подпрограммы, работающие на сервере и управляемые его процессами.

1. Delete Student

Процедура удаляет студента, когда нужно быстро его отчислить.

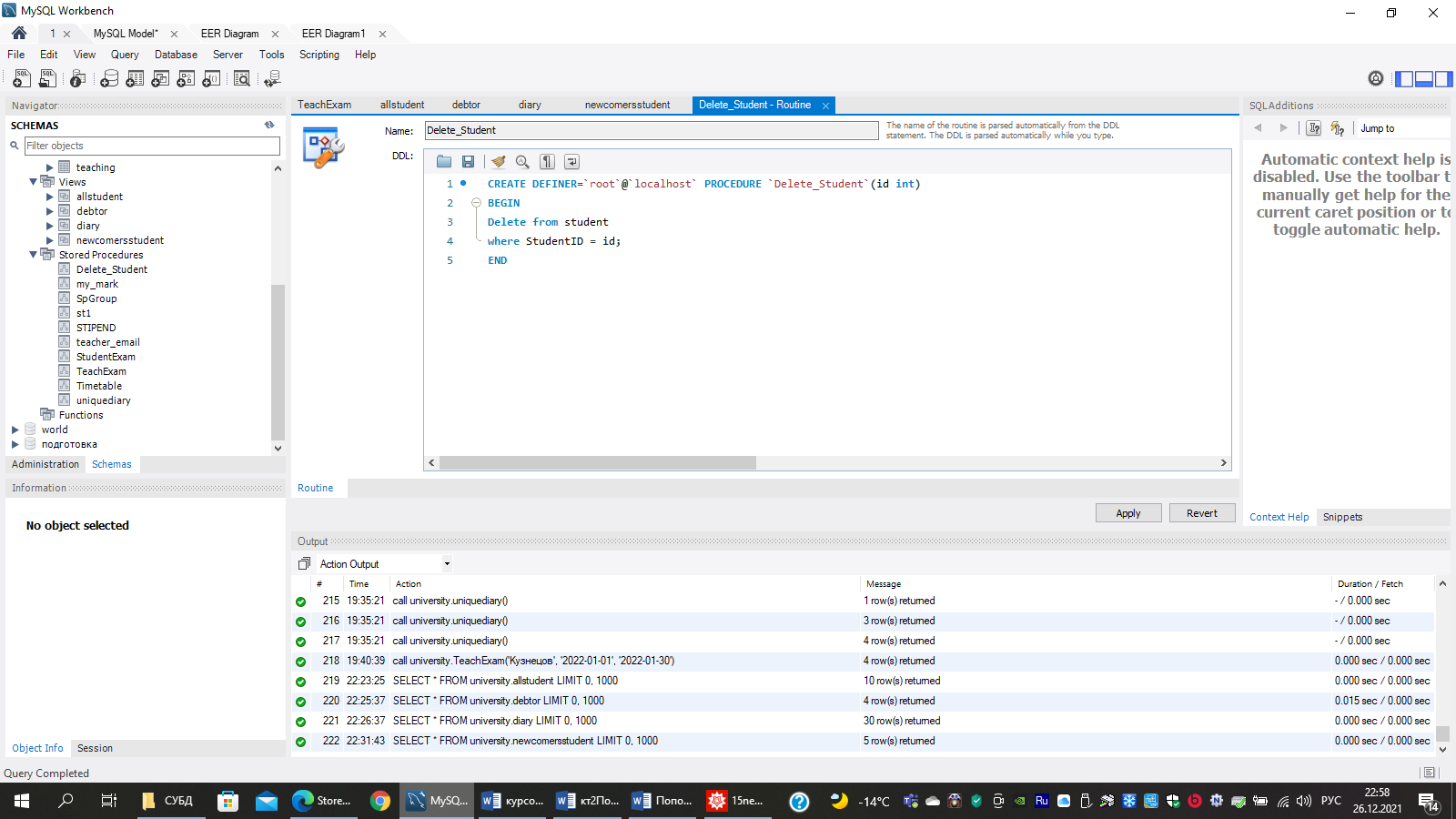


Рисунок 8 – Procedure Delete Student

1. My mark

Процедура показывает оценки по отдельно выбранному студенту:

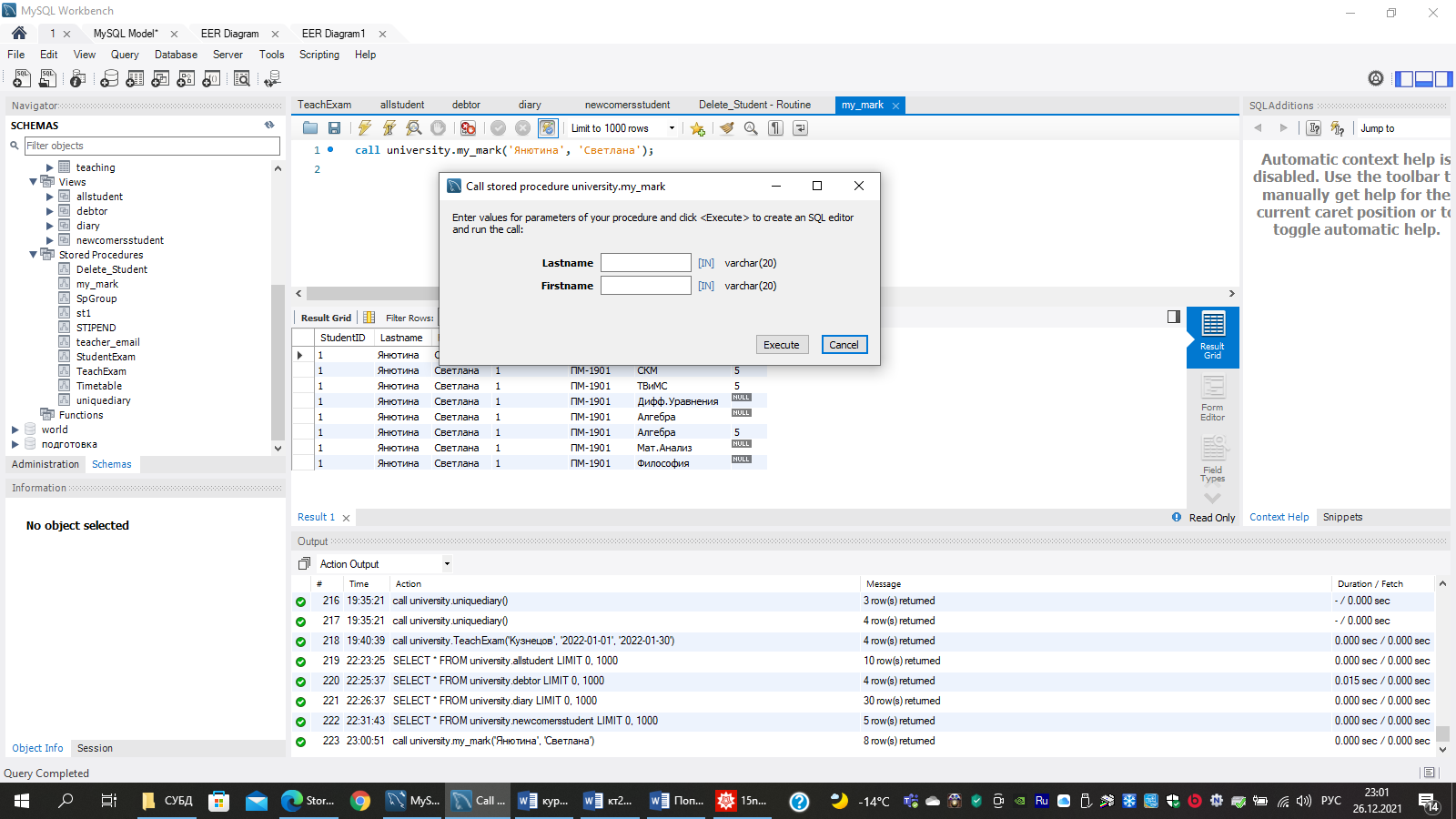


Рисунок 9 – Procedure My mark

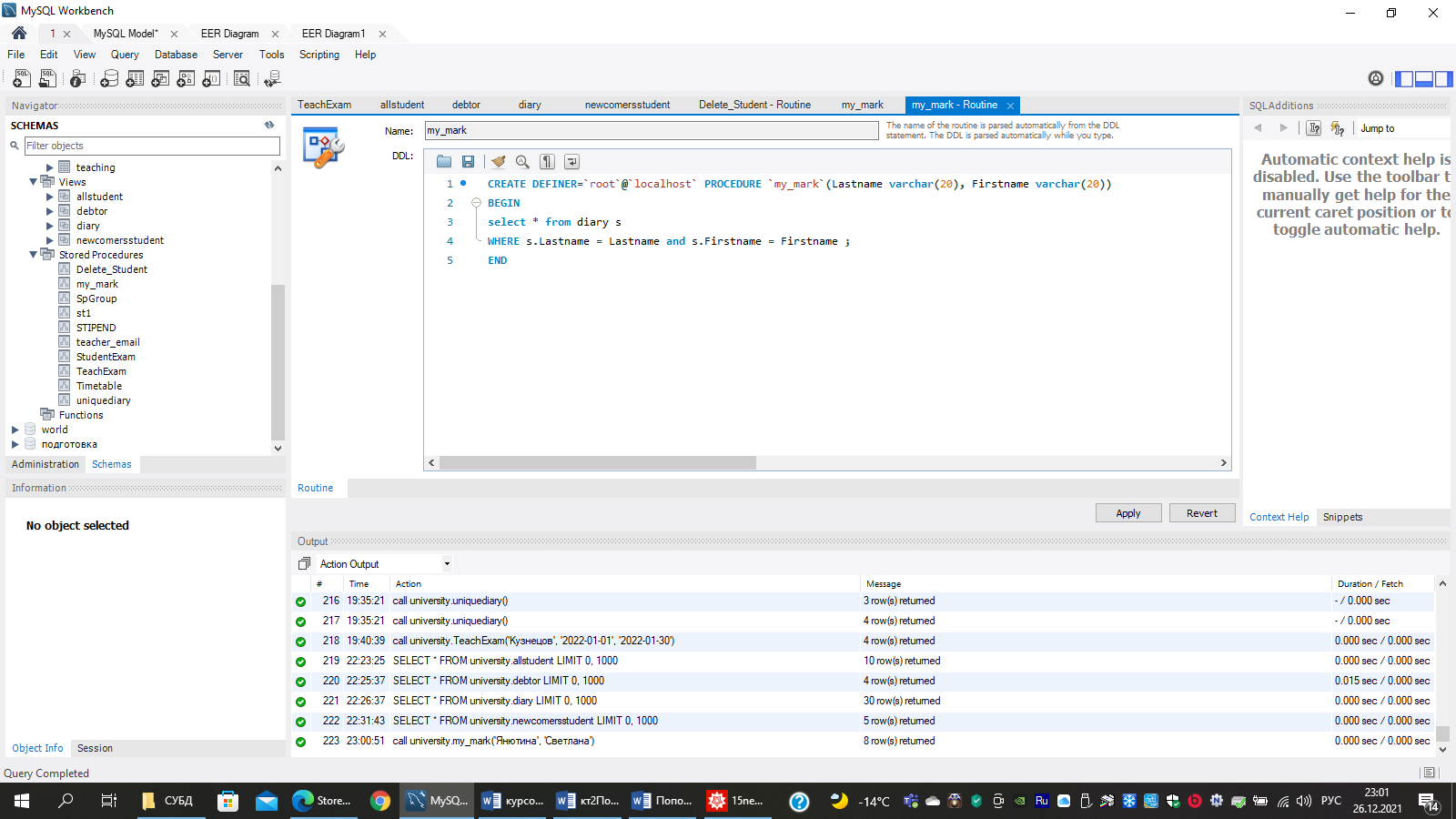


Рисунок 10 – Procedure My mark(Результат)

1. SpGroup

Процедура показывает список студентов конкретной группы:

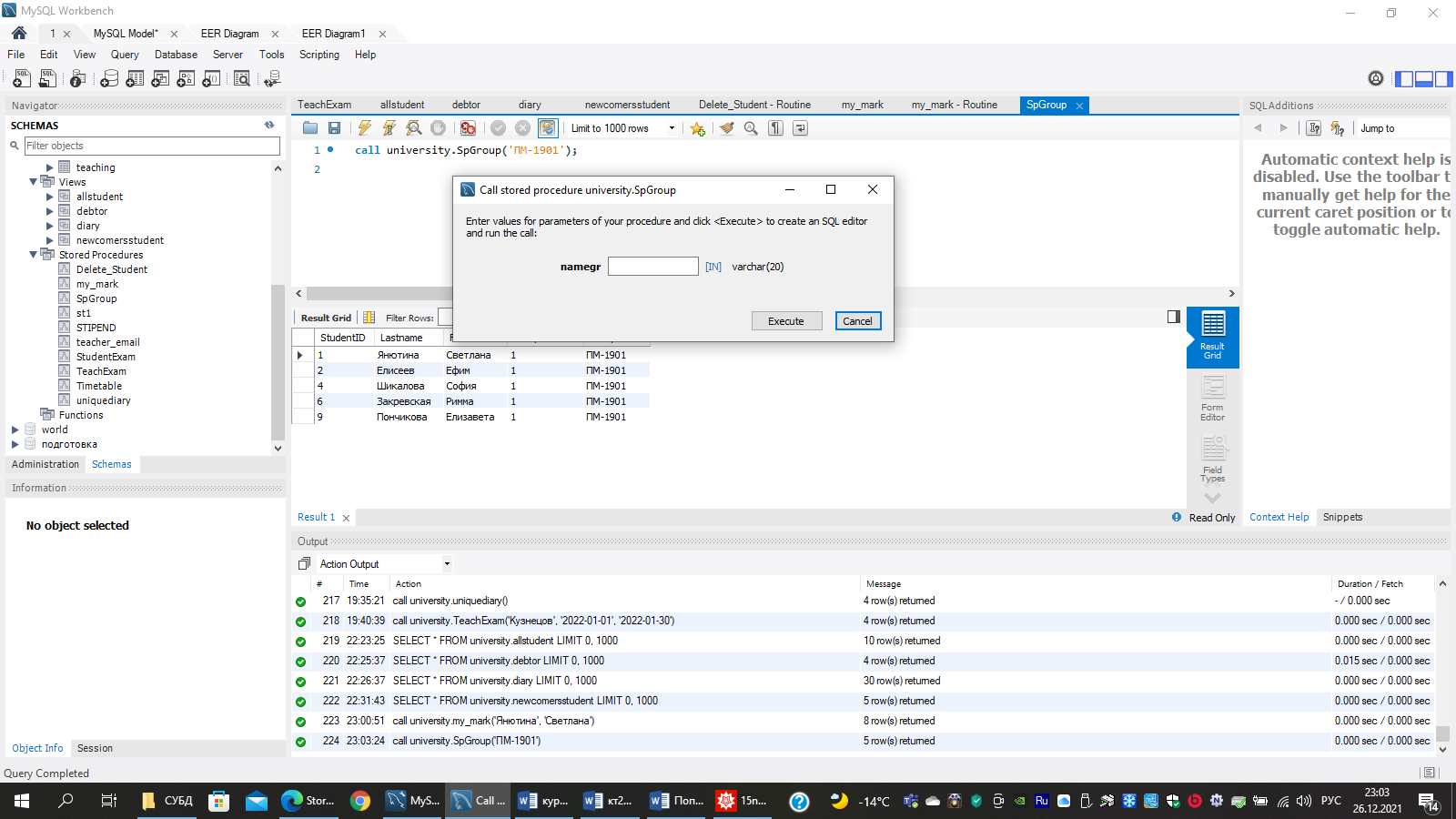


Рисунок 11 – Procedure SpGroup

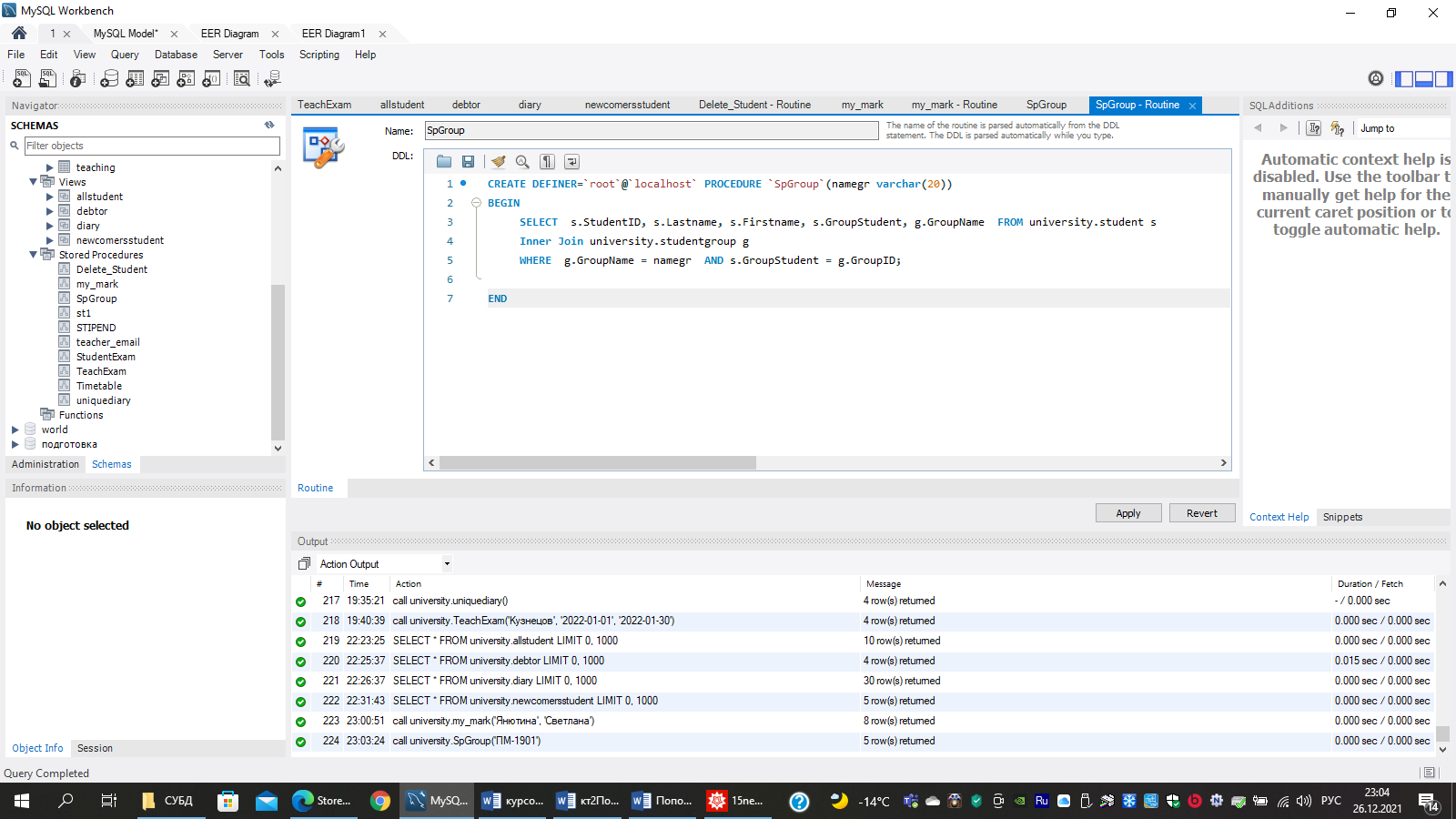


Рисунок 12 – Procedure SpGroup(Результат)

1. Stipend

Процедура для назначения стипендии по результатам семестра:

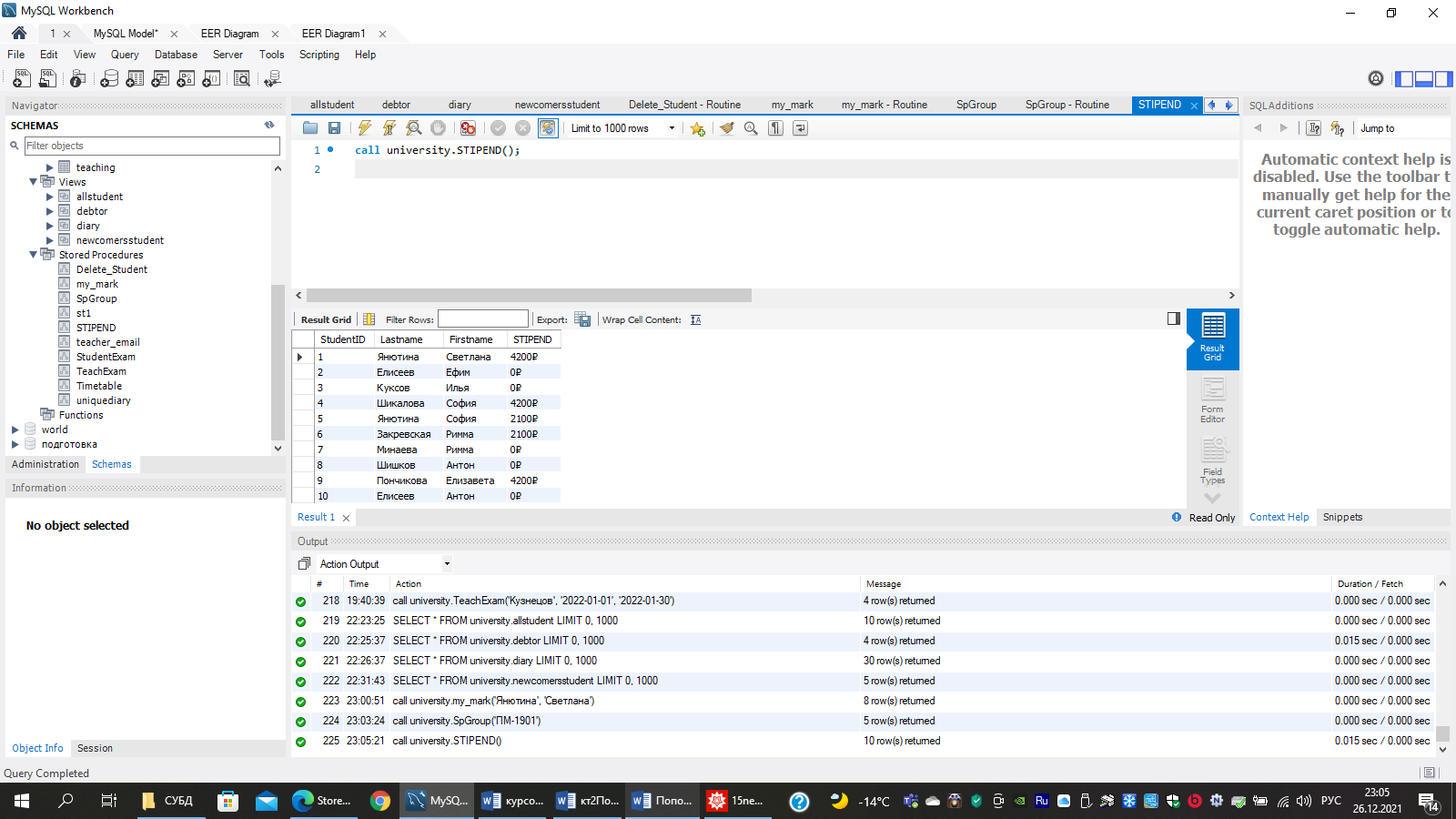


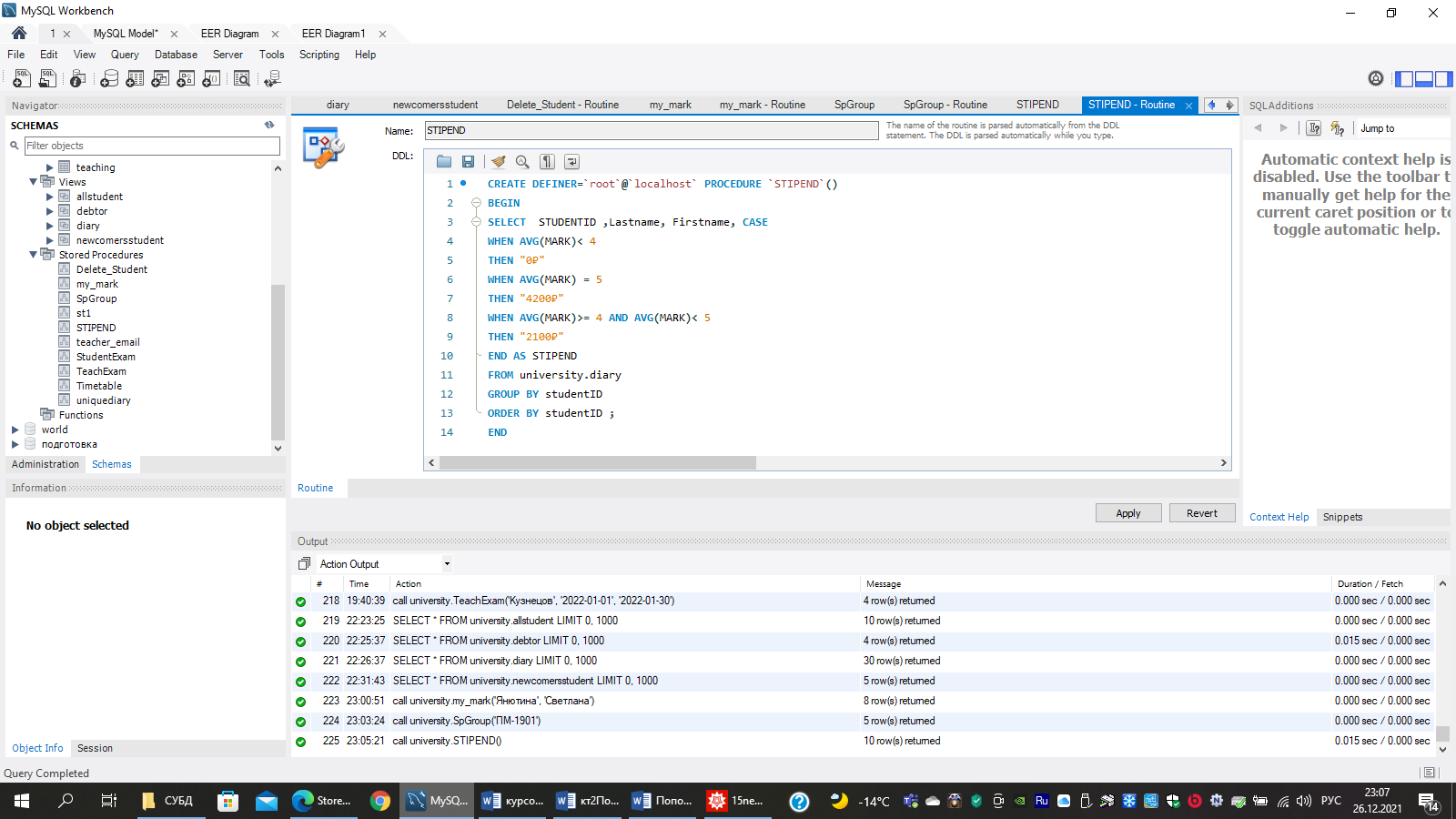
Рисунок 13 – Procedure Stipend 

Рисунок 13 – Procedure Stipend(Результат)

1. Teacher email

Процедура находит адрес электронной почты преподавателя по его имени и фамилии:

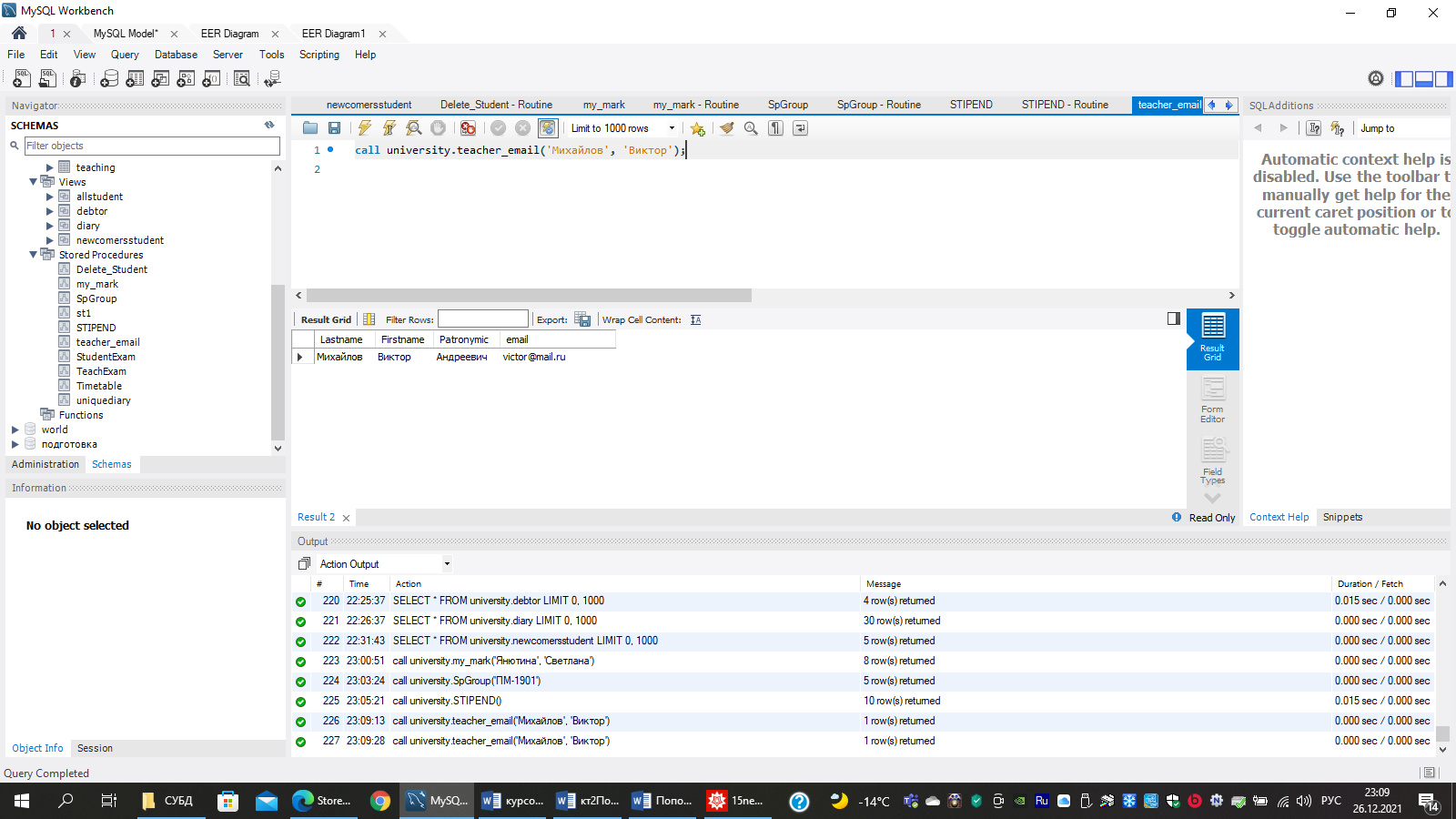


Рисунок 14 – Procedure Teacher email

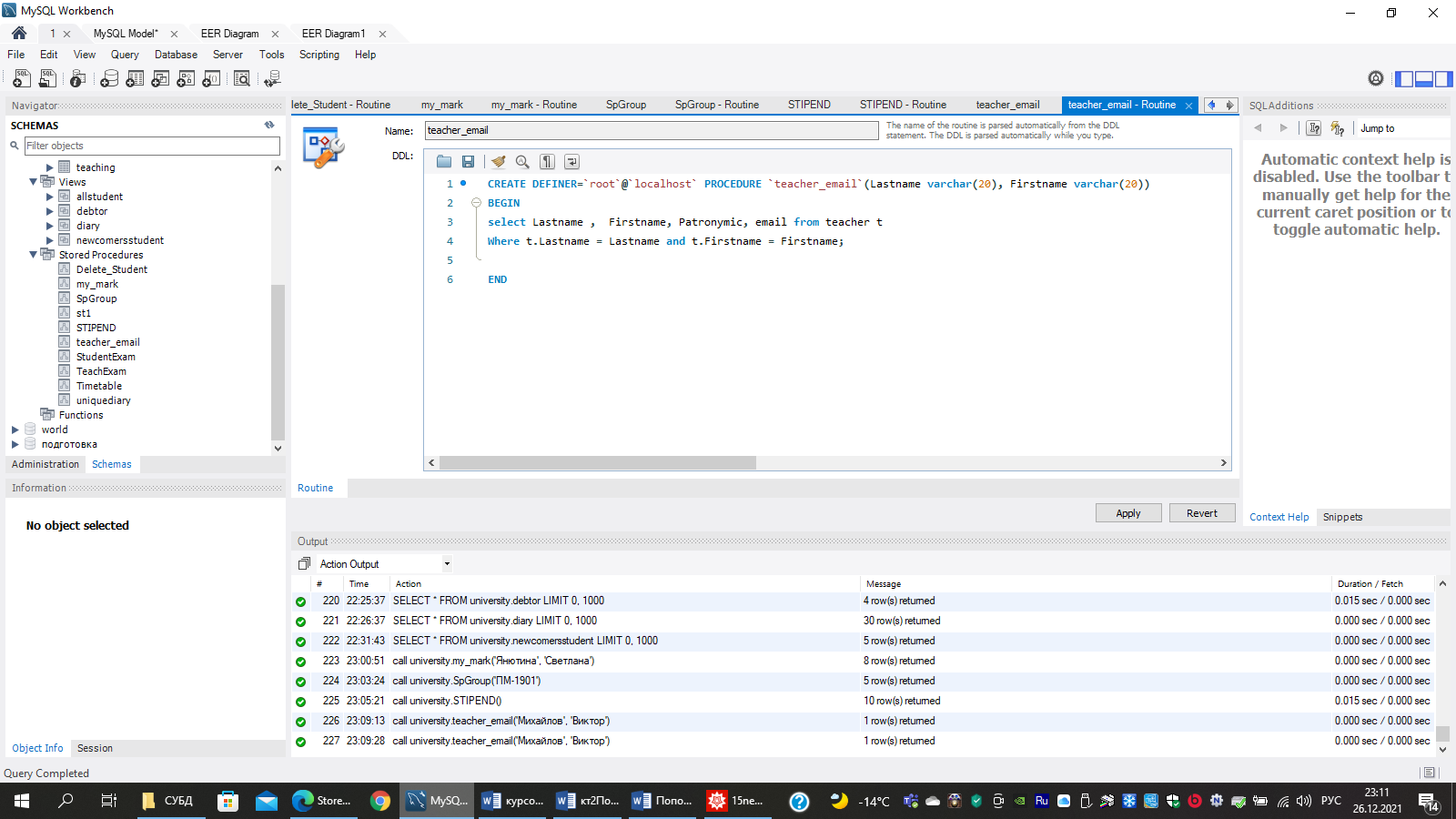


Рисунок 15 – Procedure Teacher email (Результат)

1. Student Exam

Процедура показывает, кто из преподавателей принимает экзамены у определенного студента.

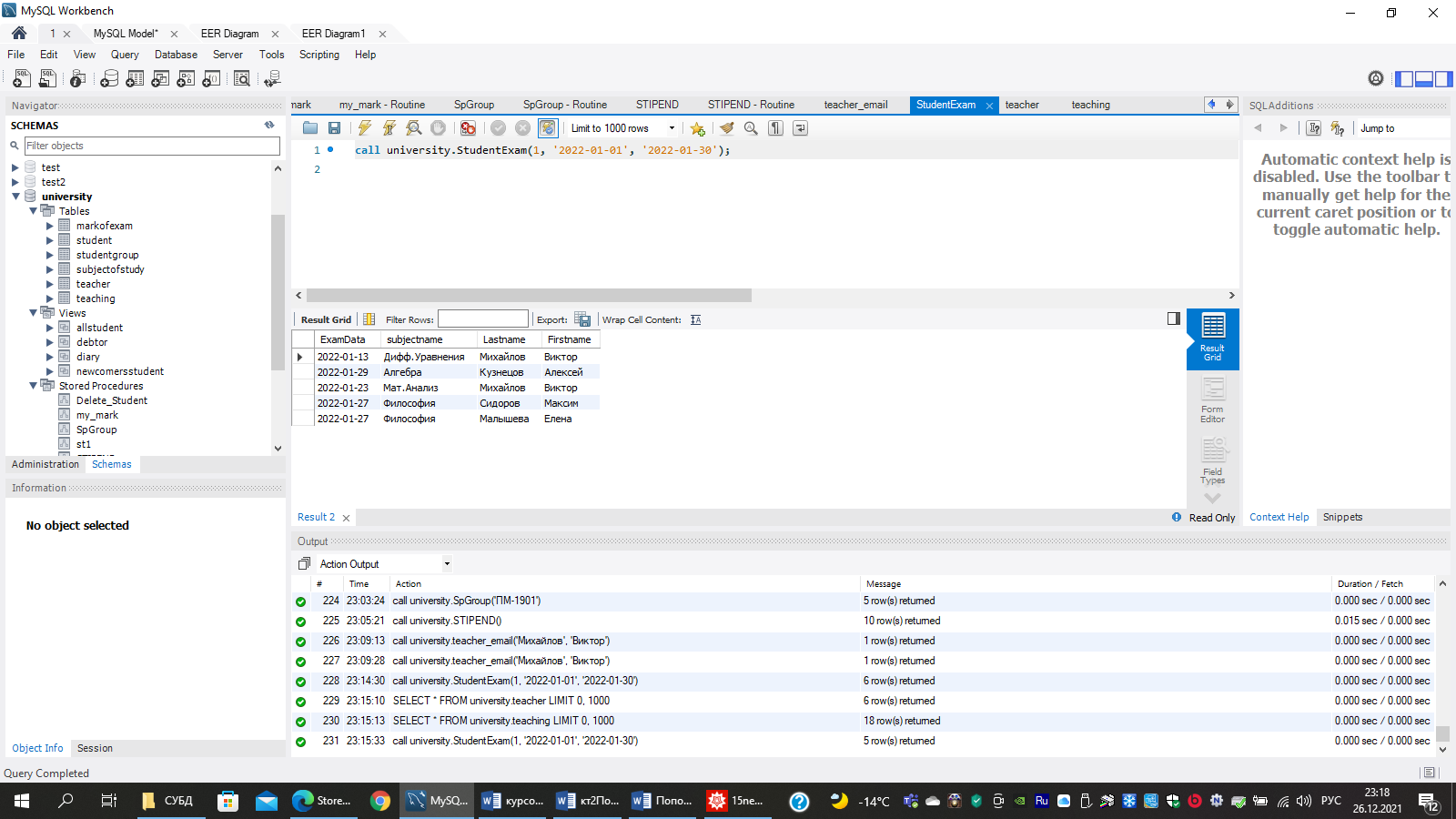


Рисунок 16 – Procedure Student Exam

В данном случае у первой студентки экзамен по философии принимают сразу 2 преподавателя.

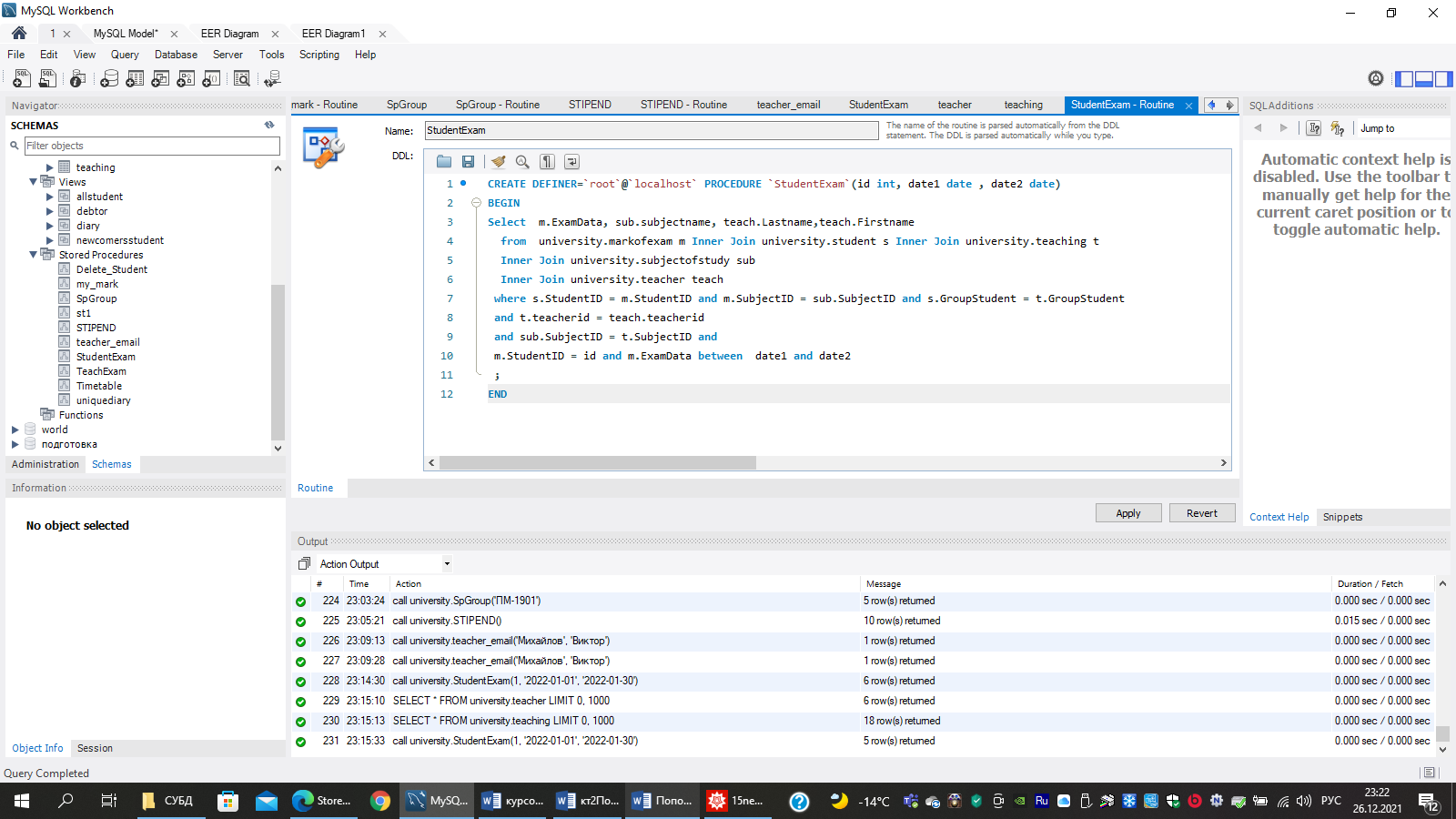


Рисунок 17 – Procedure Student Exam (Результат)

1. Teacher Exam

Процедура показывает расписание сессии для преподавателя, вводятся с клавиатуры – фамилия и даты сессии:

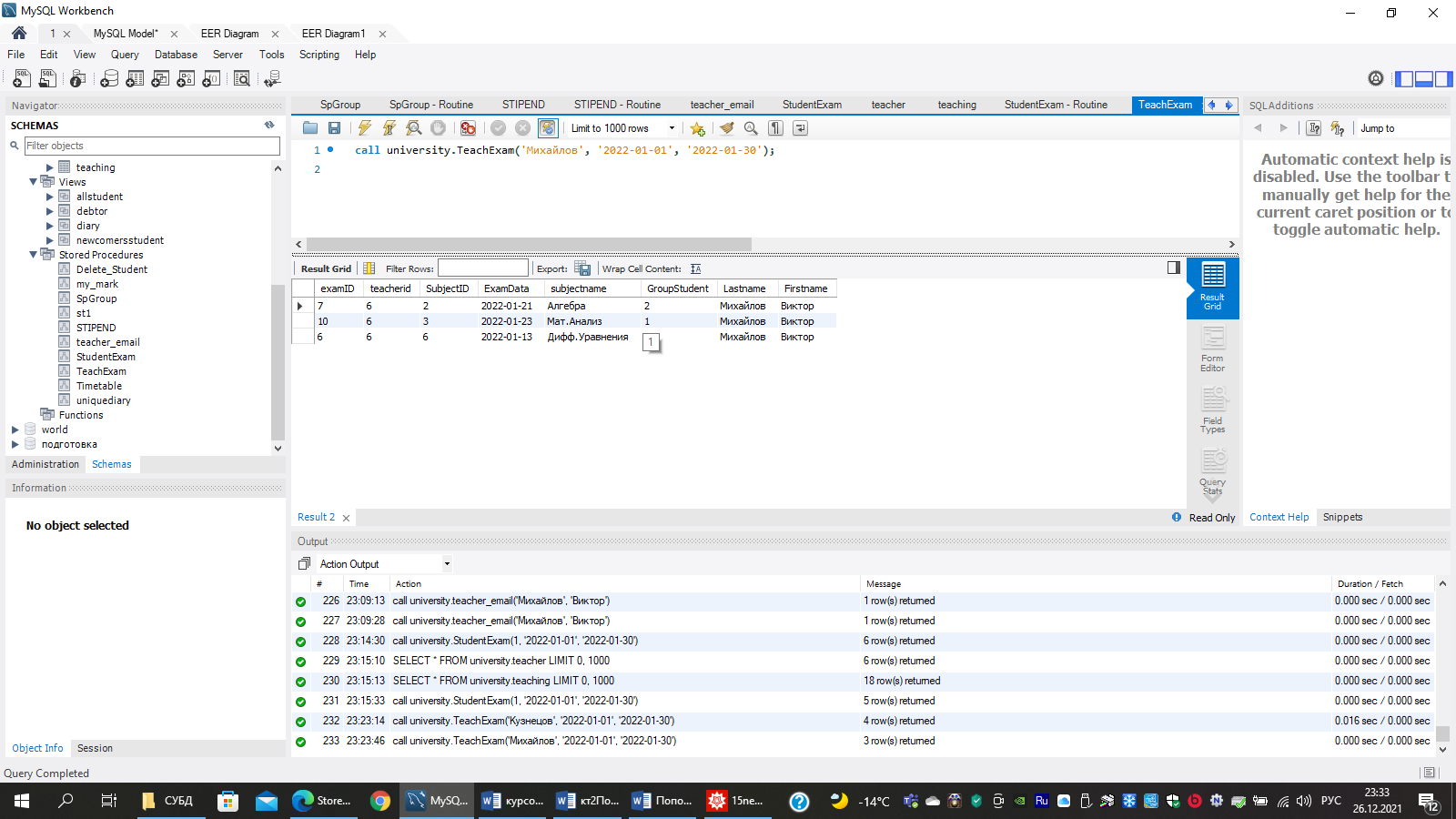


Рисунок 18 – Procedure Teacher Exam

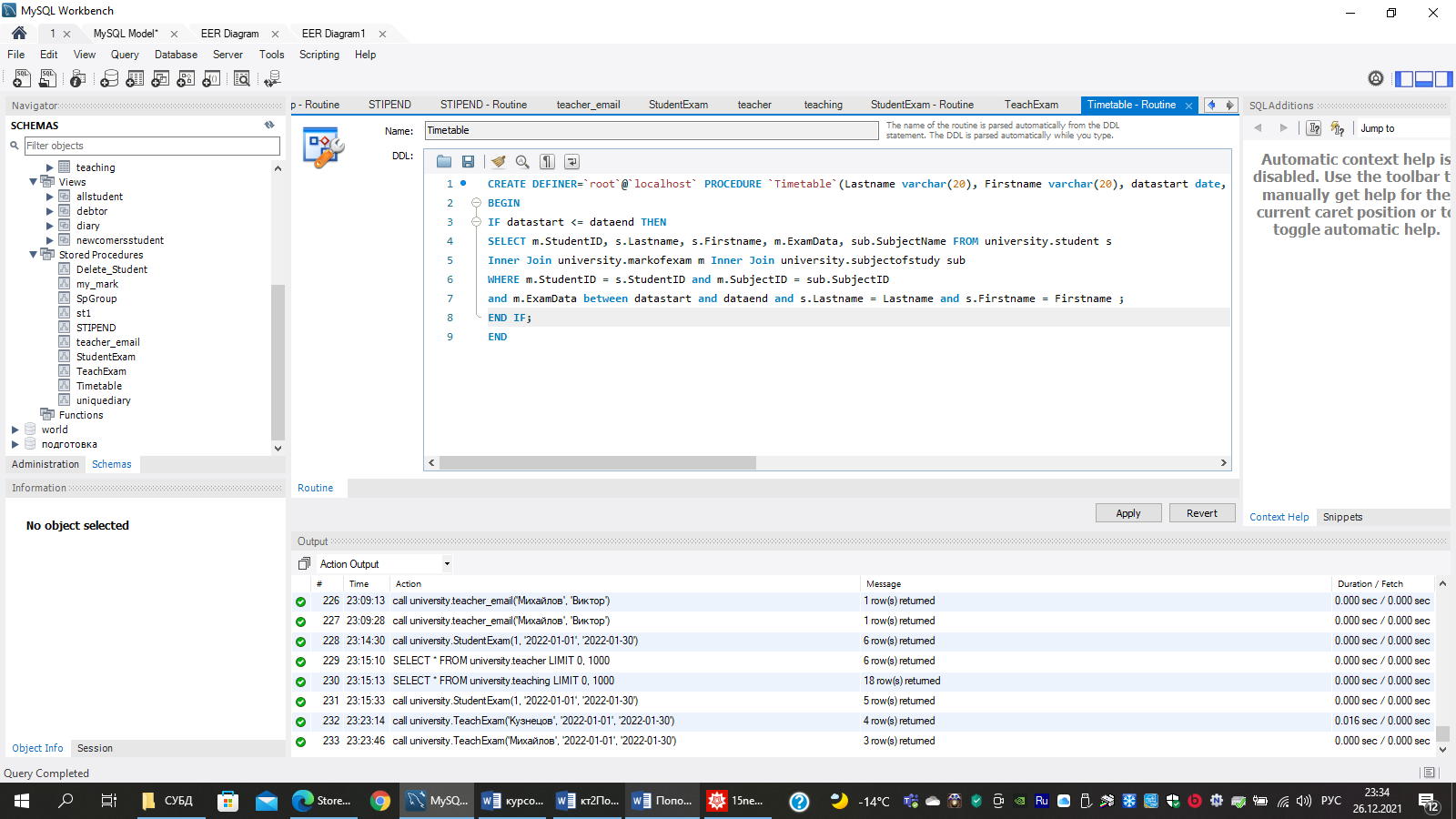


Рисунок 19 – Procedure Teacher Exam

1. Timetable

Процедура, похожая на процедуру Student Exam, их отличие в том, что здесь студент вводит не свой id, если он его забыл и на выходе получает таблицу с расписанием экзаменов в указанные им даты, но без принимающих преподавателей.

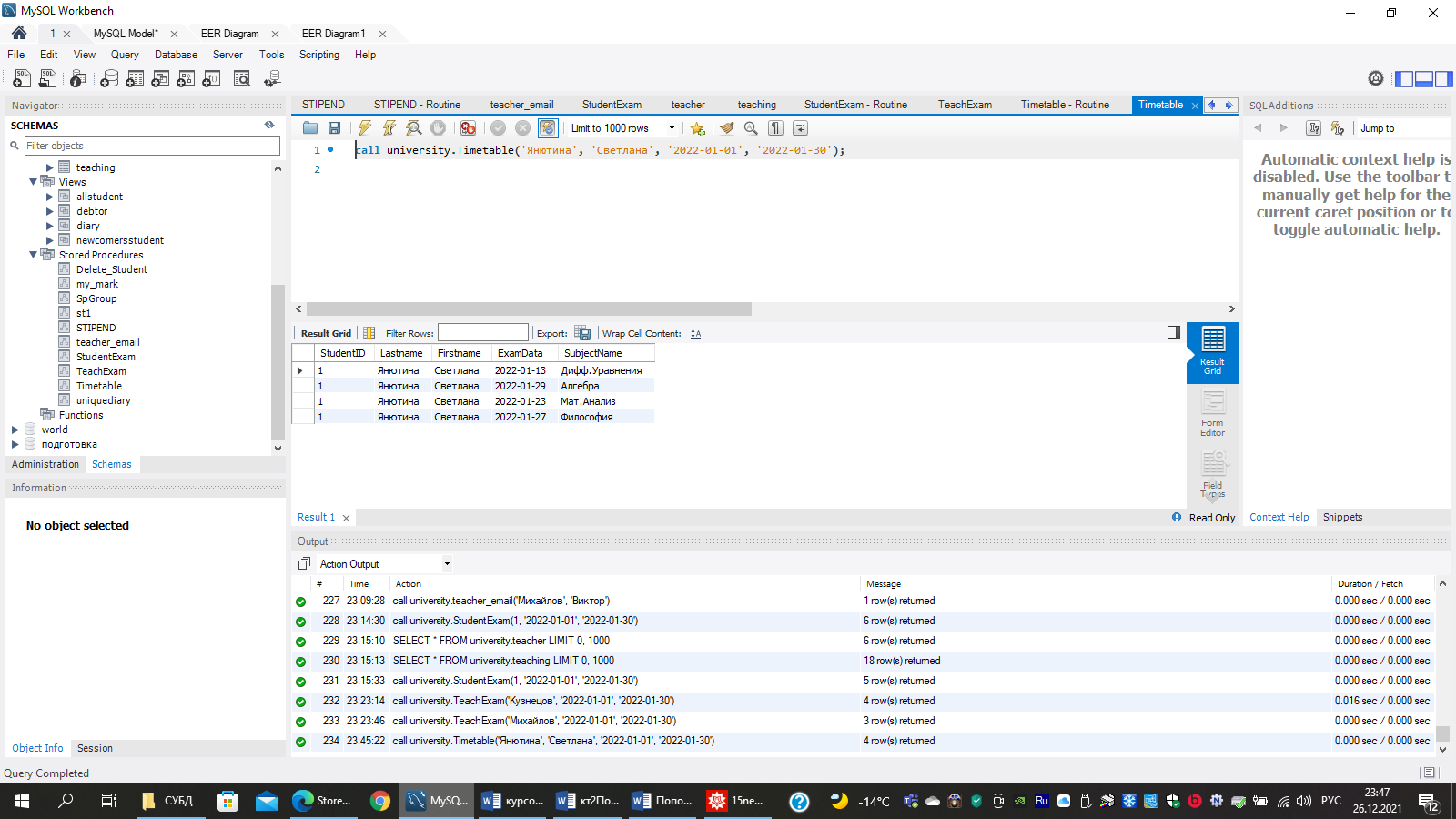


Рисунок 20 – Procedure Timetable

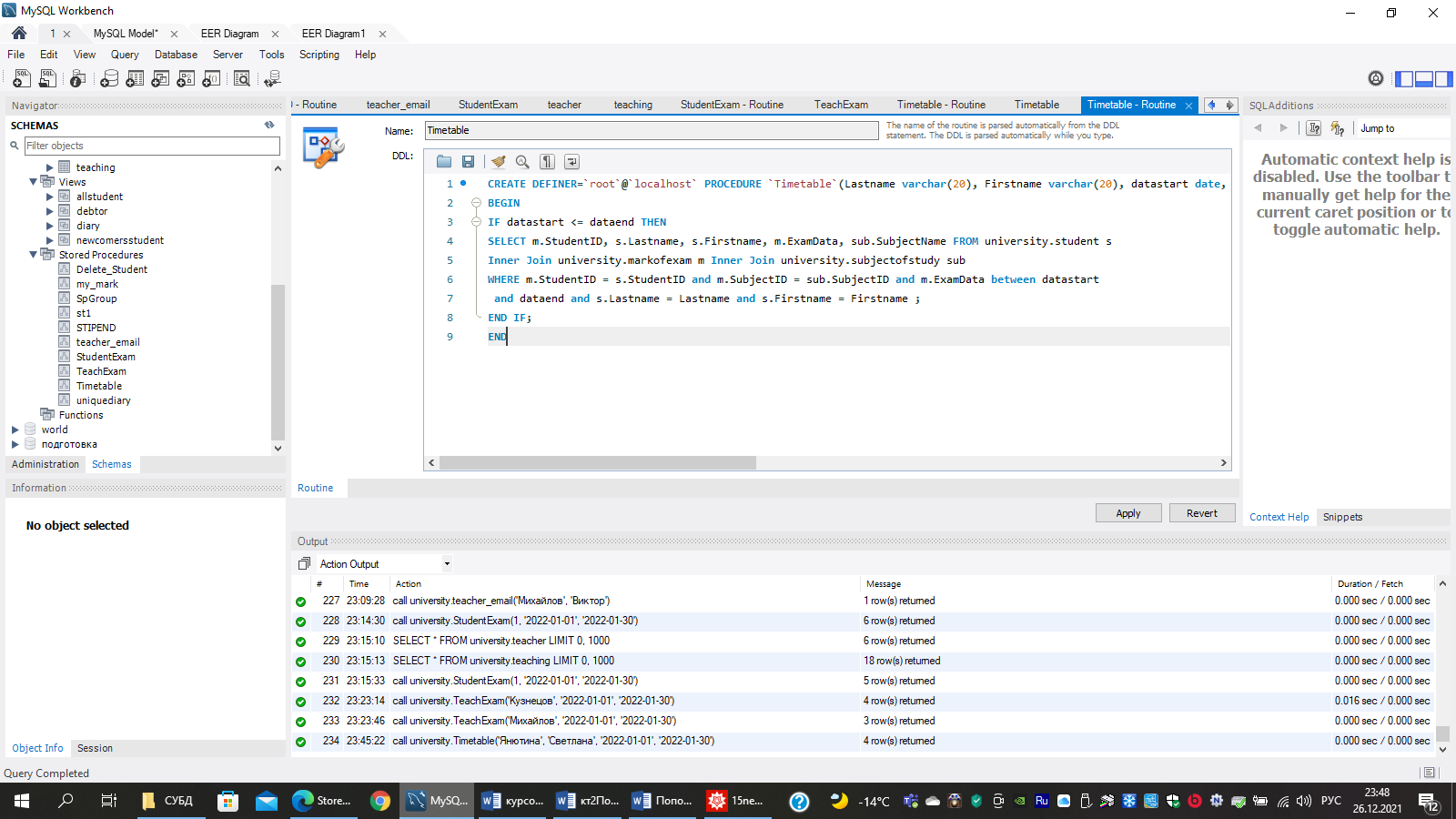


Рисунок 21 – Procedure Timetable

1. Unique diary

Процедура выводящая дневник по всем студентам, но каждый студент находится в отдельной вкладке, что очень удобно для просмотра, получается практически брс:

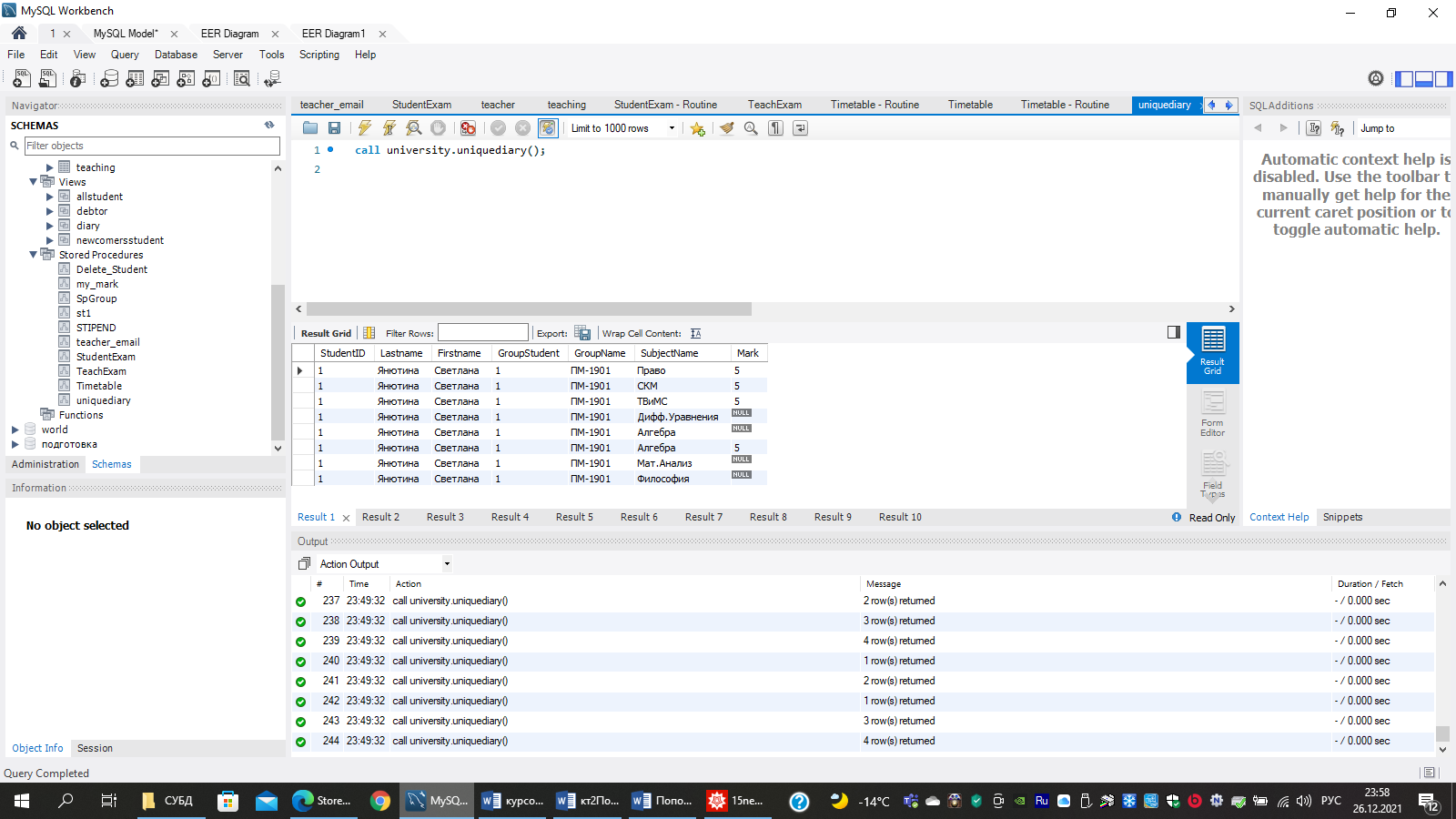


Рисунок 22 – Procedure Unique diary

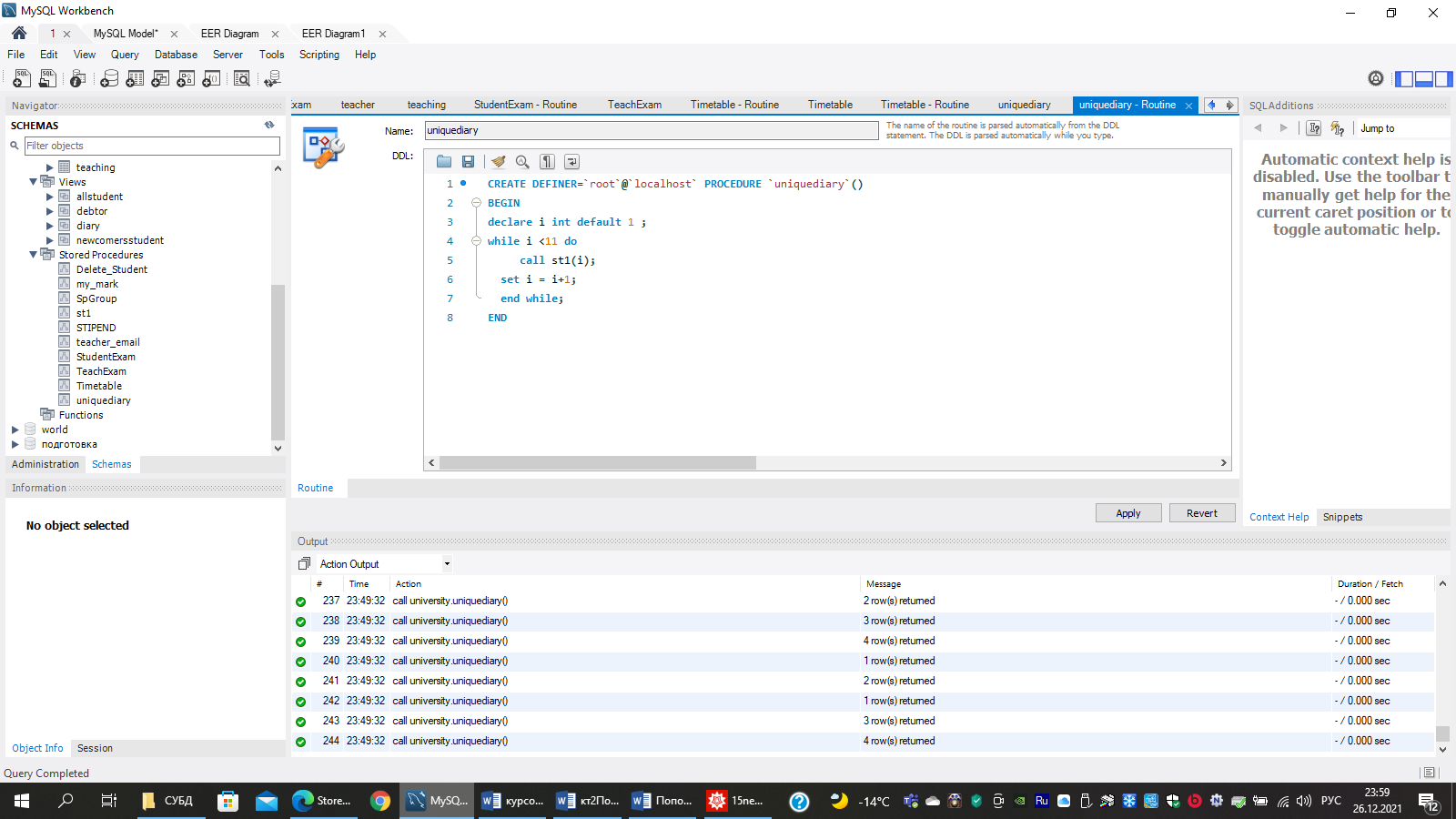


Рисунок 23 – Procedure Unique diary

Для этой процедуры была создана промежуточная процедура:

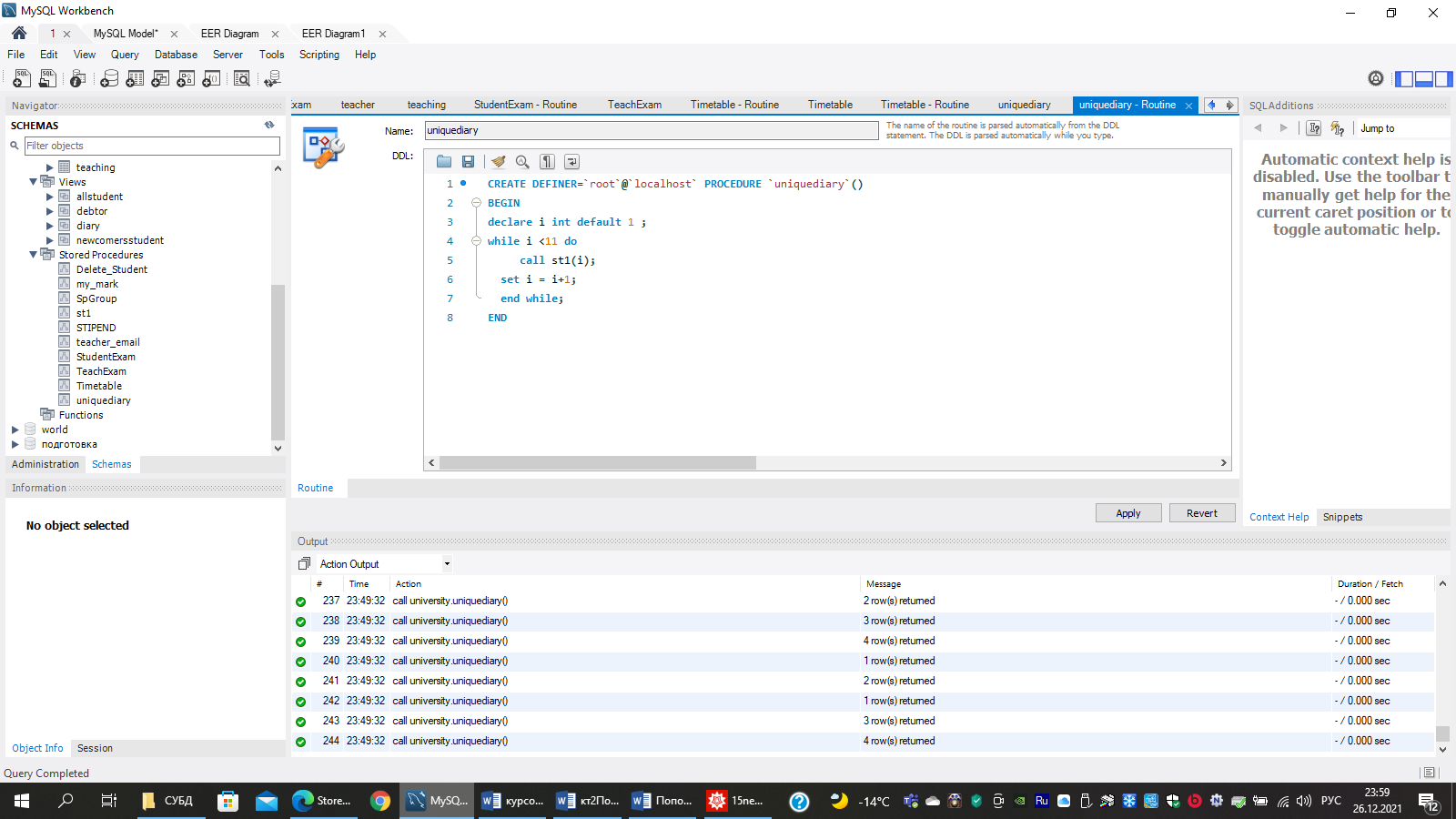


Рисунок 24 – Procedure для создания Unique diary

1. Update\_Data\_Base

Процедура предназначена для обновления базы данных - удаление оценок за прошедший период, после обновления они должны будут храниться в архиве, а в нашем приложении появятся новые.

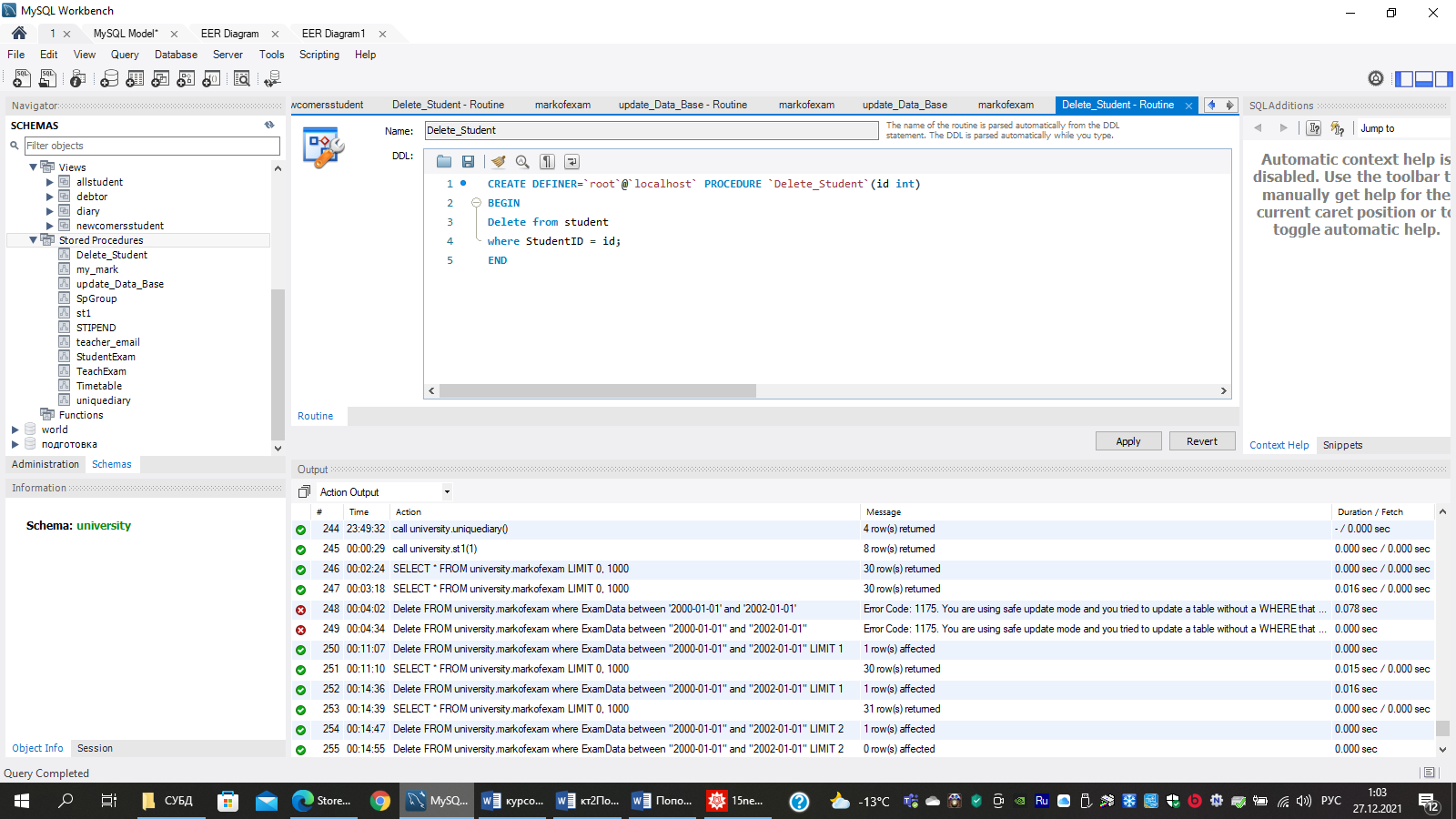


Рисунок 25 – Procedure для обновления базы данных

1. HowMuchDay

Процедура считает сколько дней вам останется на подготовку к экзамену, если Вы начнете готовится в определенный день.

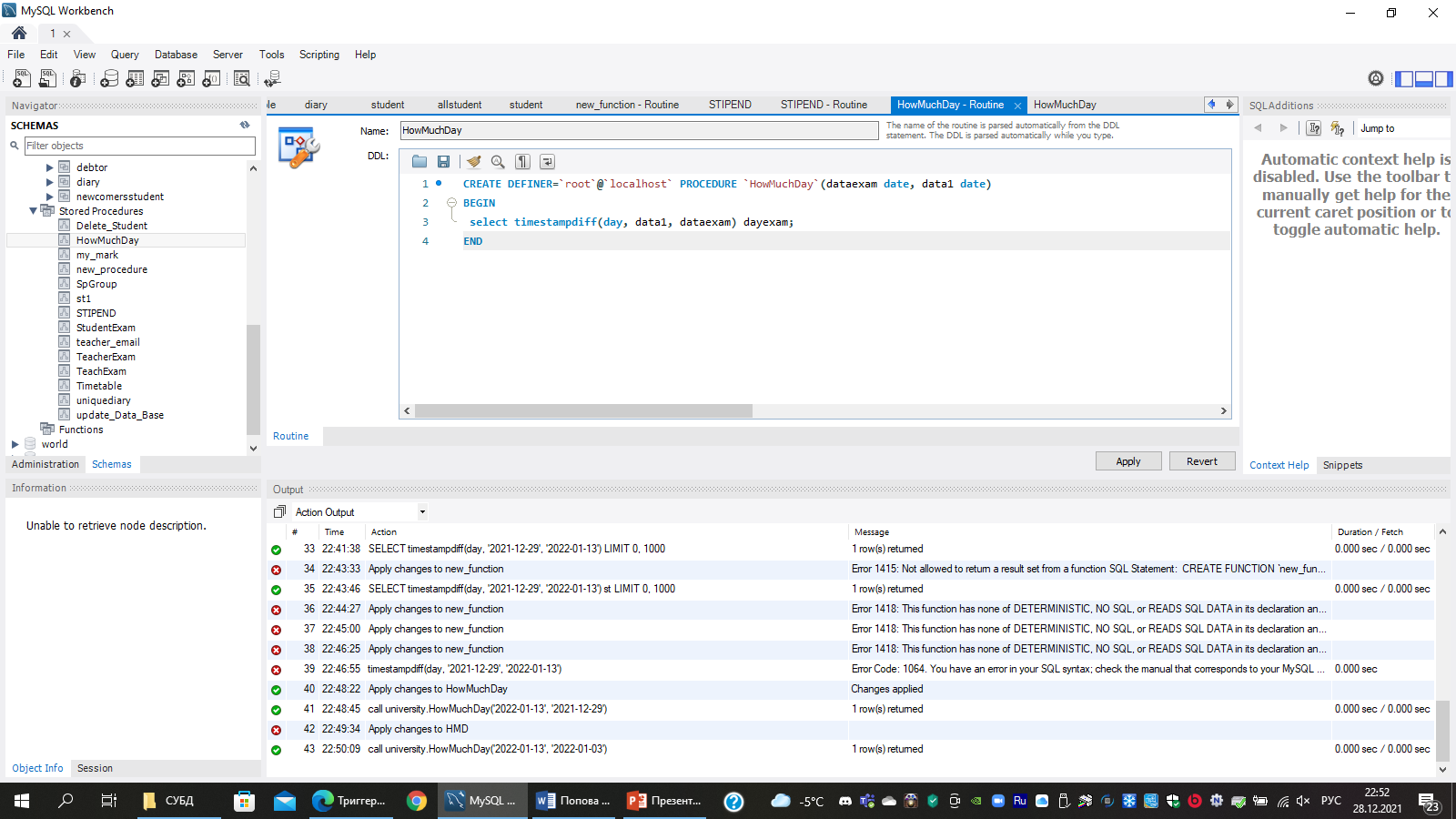


Рисунок 25 – Procedure HowMuchDay

# Заключение

Таким образом, я реализовала базу данных для университета, которая включает в себя информацию про студентов, преподавателей, предметы, группы, экзамены и многое другое.

База данных university обеспечивает упорядоченное хранение всех данных и позволяет пользователю быстро и легко получать желаемые сведения, а также она делает работу университета значительно прозрачней и эффективней.