|  |
| --- |
| МИНИСТЕРСТВО ТРАНСПОРТА РОССИЙСКОЙ ФЕДЕРАЦИИ  ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  «РОССИЙСКИЙ УНИВЕРСИТЕТ ТРАНСПОРТА» (РУТ (МИИТ))  ИНСТИТУТ ТРАНСПОРТНОЙ ТЕХНИКИ И СИСТЕМ УПРАВЛЕНИЯ   \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  Кафедра «Управление и защита информации» |
|  |
| **КУРСОВОЙ ПРОЕКТ**  **по дисциплине «Основы построения защищенных баз данных»** |
|  |
|  |
|  |
| Выполнил: ст. гр. ТКИ – 441  Поткин Д.А. |
| Проверил: к.т.н., доц.  Васильева М.А. |
|  |
|  |
|  |
| Москва 2024 |

**Оглавление**

[Введение 3](#_Toc167896418)

[Проектирование базы данных 4](#_Toc167896419)

[Создание базы данных 9](#_Toc167896420)

[Создание таблиц 9](#_Toc167896421)

[Создание ограничений на пустые значения 10](#_Toc167896422)

[Добавление первичных ключей 11](#_Toc167896423)

[Добавление внешних ключей 12](#_Toc167896424)

[Создание ограничений на уникальное значение 13](#_Toc167896425)

[Заполнение базы данных 14](#_Toc167896426)

[Формирование запросов 15](#_Toc167896427)

[Создание функций 20](#_Toc167896428)

[Создание процедур 27](#_Toc167896429)

[Вывод 31](#_Toc167896430)

[Список литературы 32](#_Toc167896431)

Введение

Целью курсового проекта является изучение методов и закрепление знаний в проектировании реляционных баз данных (РБД) в системе управления базами данных (СУБД) PostgreSQL.

В данном курсовом проекте ставится задача разработать РБД в СУБД PostgreSQL по теме «ВУЗ», необходимо разработать скрипты на создание и заполнение РБД.

База данных создаётся для информационного обслуживания работников деканата. В ВУЗе учатся студенты, которые разбиты на группы. По каждой специальности обучается несколько групп. Каждая кафедра ведет занятия по нескольким дисциплинам. Занятия для студентов проводят преподаватели с разных кафедр.

Готовые запросы:

* Выдавать информацию о студенте по № зачетной книжки, по ФИО.
* Выдавать список предметов, читаемых данной кафедрой.
* Выдавать список преподавателей, проводящих занятие в данной группе.
* Выдавать список групп, обучающихся на данной специальности.

Проектирование базы данных

Выделим базовые сущности базы данных «ВУЗ»:

* Студенты,
* Группы,
* Специальности,
* Кафедры,
* Дисциплины,
* Преподаватели;

Были созданы следующие сущности:

1. Таблица "Students" - содержит информацию о студентах
2. Таблица "Groups" – содержит информацию о группах
3. Таблица "Specialties" - содержит информацию о специальностях
4. Таблица "Departments" - содержит информацию о кафедрах
5. Таблица "Disciplines" - содержит информацию о дисциплинах
6. Таблица "Teachers" - содержит информацию про преподавателей

На рисунке 1 представлена ER-диаграмма БД.

|  |
| --- |
|  |
| 1. – ER-диаграмма базы данных «ВУЗ» |

На рисунке 2 представлена уточненная ER-диаграмма БД.

|  |
| --- |
|  |
| 1. – ER-диаграмма базы данных «ВУЗ» |

Представленная ER-диаграмма соответствует нормальной форме Бойса-Кодда.

1. - Схема отношения таблицы Students

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор | ID | int4 | Первичный ключ, обязательное поле |
| Имя студента | Name | varchar(50) | Обязательное поле |
| Номер зачетной книжки | NumberGradebook | int4 | Обязательное поле |
| Идентификатор группы | GroupID | int4 | Внешний ключ, обязательное поле |

1. - Схема отношения таблицы Groups

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор | ID | int4 | Первичный ключ, обязательное поле |
| Номер группы | GroupNumber | int4 | Обязательное поле |
| Идентификатор специальности | SpecialityID | int4 | Внешний ключ, обязательное поле |

1. - Схема отношения таблицы Specialties

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор | ID | int4 | Первичный ключ, обязательное поле |
| Название специальности | Title | varchar(100) | Обязательное поле |

1. - Схема отношения таблицы Disciplines

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор | ID | int4 | Первичный ключ, обязательное поле |
| Название дисциплины | Title | varchar(100) | Обязательное поле |
| Идентификатор кафедры | DepartmentID | int4 | Внешний ключ, обязательное поле |

1. - Схема отношения таблицы Teachers

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор | ID | int4 | Первичный ключ, обязательное поле |
| Имя преподавателя | Name | varchar(50) | Обязательное поле |

1. - Схема отношения таблицы Departments

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор | ID | int4 | Внешний ключ, обязательное поле |
| Название кафедры | Title | int4 | Внешний ключ, обязательное поле |

1. - Схема отношения таблицы Groups\_Teachers

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор группы | GroupID | int4 | Внешний ключ, обязательное поле |
| Идентификатор преподавателя | TeacherID | int4 | Внешний ключ, обязательное поле |

1. - Схема отношения таблицы Disciplines\_Teachers

|  |  |  |  |
| --- | --- | --- | --- |
| **Содержание поля** | **Имя поля** | **Тип** | **Примечания** |
| Идентификатор дисциплины | DisciplineID | int4 | Внешний ключ, обязательное поле |
| Идентификатор преподавателя | TeacherID | int4 | Внешний ключ, обязательное поле |

Создание базы данных

БД проектируется в среде DBeaver. Напишем скрипты для создания базы данных и таблиц.

Создание базы данных «ВУЗ»:

**create** **database** **"University"**;

Создание таблиц

Создание таблицы Students:

**create** **table** **"Students"** (

**"ID"** **int**,

**"Name"** **varchar**(50),

**"NumberGradebook"** **int**,

**"GroupID"** **int**

);

Создание таблицы Groups:

**create** **table** **"Groups"** (

**"ID"** **int**,

**"GroupNumber"** **int**,

**"SpecialityID"** **int**

);

Создание таблицы Specialties:

**create** **table** **"Specialties"** (

**"ID"** **int**,

**"Title"** **varchar**(100)

);

Создание таблицы Departments:

**create** **table** **"Departments"** (

**"ID"** **int**,

**"Title"** **varchar**(100)

);

Создание таблицы Disciplines:

**create** **table** **"Disciplines"** (

**"ID"** **int**,

**"Title"** **varchar**(100),

**"DepartamentID"** **int**

);

Создание таблицы Teachers:

**create** **table** **"Teachers"** (

**"ID"** **int**,

**"Name"** **varchar**(50)

);

Создание таблицы Groups\_Teachers:

**create** **table** **"Groups\_Teachers"** (

**"GroupID"** **int**,

**"TeacherID"** **int**

);

Создание таблицы Disciplines\_Teachers:

**create** **table** **"Disciplines\_Teachers"** (

**"DisciplineID"** **int**,

**"TeacherID"** **int**

);

Создание ограничений на пустые значения

В каждой из таблиц добавим ограничения на пустые значения для столбцов, которые не могут являться пустыми.

**alter** **table** **"Departments"**

**alter** **column** **"ID"** **set** **not** **null**,

**alter** **column** **"Title"** **set** **not** **null**;

**alter** **table** **"Disciplines"**

**alter** **column** **"ID"** **set** **not** **null**,

**alter** **column** **"Title"** **set** **not** **null**,

**alter** **column** **"DepartamentID"** **set** **not** **null**;

**alter** **table** **"Disciplines\_Teachers"**

**alter** **column** **"DisciplineID"** **set** **not** **null**,

**alter** **column** **"TeacherID"** **set** **not** **null**;

**alter** **table** **"Groups"**

**alter** **column** **"ID"** **set** **not** **null**,

**alter** **column** **"GroupNumber"** **set** **not** **null**,

**alter** **column** **"SpecialityID"** **set** **not** **null**;

**alter** **table** **"Groups\_Teachers"**

**alter** **column** **"GroupID"** **set** **not** **null**,

**alter** **column** **"TeacherID"** **set** **not** **null**;

**alter** **table** **"Specialties"**

**alter** **column** **"ID"** **set** **not** **null**,

**alter** **column** **"Title"** **set** **not** **null**;

**alter** **table** **"Students"**

**alter** **column** **"ID"** **set** **not** **null**,

**alter** **column** **"Name"** **set** **not** **null**,

**alter** **column** **"NumberGradebook"** **set** **not** **null**,

**alter** **column** **"GroupID"** **set** **not** **null**;

**alter** **table** **"Teachers"**

**alter** **column** **"ID"** **set** **not** **null**,

**alter** **column** **"Name"** **set** **not** **null**;

Добавление первичных ключей

Установим на поле с идентификатором «ID» ограничение первичного ключа.

**alter** **table** **"Departments"**

**add** **constraint** **"Pk\_Departments"** **primary** **key** (**"ID"**);

**alter** **table** **"Disciplines"**

**add** **constraint** **"Pk\_Disciplines"** **primary** **key** (**"ID"**);

**alter** **table** **"Disciplines\_Teachers"**

**add** **constraint** **"Pk\_Disciplines\_Teachers"** **primary** **key** (**"DisciplineID"**,**"TeacherID"**);

**alter** **table** **"Groups"**

**add** **constraint** **"Pk\_Groups"** **primary** **key** (**"ID"**);

**alter** **table** **"Groups\_Teachers"**

**add** **constraint** **"Pk\_Groups\_Teachers"** **primary** **key** (**"GroupID"**,**"TeacherID"**);

**alter** **table** **"Specialties"**

**add** **constraint** **"Pk\_Specialties"** **primary** **key** (**"ID"**);

**alter** **table** **"Students"**

**add** **constraint** **"Pk\_Students"** **primary** **key** (**"ID"**);

**alter** **table** **"Teachers"**

**add** **constraint** **"Pk\_Teachers"** **primary** **key** (**"ID"**);

Добавление внешних ключей

Сформируем связи между таблицами с помощью добавления внешних ключей.

**alter** **table** **"Disciplines"**

**add** **constraint** **"Fk\_Departments\_Disciplines"** **foreign** **key** (**"DepartamentID"**) **references** **"Departments"** (**"ID"**);

**alter** **table** **"Disciplines\_Teachers"**

**add** **constraint** **"Fk\_Disciplines\_Disciplines\_Teachers"** **foreign** **key** (**"DisciplineID"**) **references** **"Disciplines"** (**"ID"**);

**alter** **table** **"Disciplines\_Teachers"**

**add** **constraint** **"Fk\_Teachers\_Disciplines\_Teachers"** **foreign** **key** (**"TeacherID"**) **references** **"Teachers"** (**"ID"**);

**alter** **table** **"Groups"**

**add** **constraint** **"Fk\_Specialties\_Groups"** **foreign** **key** (**"SpecialityID"**) **references** **"Specialties"** (**"ID"**);

**alter** **table** **"Groups\_Teachers"**

**add** **constraint** **"Fk\_Groups\_Groups\_Teachers"** **foreign** **key** (**"GroupID"**) **references** **"Groups"** (**"ID"**);

**alter** **table** **"Groups\_Teachers"**

**add** **constraint** **"Fk\_Teachers\_Groups\_Teachers"** **foreign** **key** (**"TeacherID"**) **references** **"Teachers"** (**"ID"**);

**alter** **table** **"Students"**

**add** **constraint** **"Fk\_Groups\_Students"** **foreign** **key** (**"GroupID"**) **references** **"Groups"** (**"ID"**);

Создание ограничений на уникальное значение

Добавим ограничение уникальности для полей.

**alter** **table** **"Departments"**

**add** **constraint** **"U\_Departments\_Title"**

**unique** (**"Title"**);

**alter** **table** **"Departments"**

**add** **constraint** **"U\_Departments\_ID"**

**unique** (**"ID"**);

**alter** **table** **"Disciplines"**

**add** **constraint** **"U\_Disciplines\_ID"**

**unique** (**"ID"**);

**alter** **table** **"Disciplines"**

**add** **constraint** **"U\_Disciplines\_Title"**

**unique** (**"Title"**);

**alter** **table** **"Groups"**

**add** **constraint** **"U\_Groups\_ID"**

**unique** (**"ID"**);

**alter** **table** **"Groups"**

**add** **constraint** **"U\_Groups\_GroupNumber"**

**unique** (**"GroupNumber"**);

**alter** **table** **"Specialties"**

**add** **constraint** **"U\_Specialties\_ID"**

**unique** (**"ID"**);

**alter** **table** **"Specialties"**

**add** **constraint** **"U\_Specialties\_Title"**

**unique** (**"Title"**);

**alter** **table** **"Students"**

**add** **constraint** **"U\_Students\_ID"**

**unique** (**"ID"**);

**alter** **table** **"Students"**

**add** **constraint** **"U\_NumberGradebook\_ID"**

**unique** (**"NumberGradebook"**);

**alter** **table** **"Teachers"**

**add** **constraint** **"U\_Teachers\_ID"**

**unique** (**"ID"**);

Заполнение базы данных

Создадим скрипты для заполнения базы данных «ВУЗ» с целью дальнейшего формирования запросов.

**insert** **into** **"Departments"** (**"ID"**, **"Title"**) **values**

(1, **'Факультет информационных технологий'**),

(2, **'Факультет экономики'**),

(3, **'Медицинский факультет'**);

**insert** **into** **"Disciplines"** (**"ID"**, **"Title"**, **"DepartamentID"**) **values**

(1, **'Программирование'**, 1),

(2, **'Макроэкономика'**, 2),

(3, **'Анатомия'**, 3);

**insert** **into** **"Disciplines\_Teachers"** (**"DisciplineID"**, **"TeacherID"**) **values**

(1, 1),

(1, 3),

(2, 2),

(2, 1),

(3, 3);

**insert** **into** **"Groups"** (**"ID"**, **"GroupNumber"**, **"SpecialityID"**) **values**

(1, 101, 1),

(2, 201, 2),

(3, 301, 3);

**insert** **into** **"Groups\_Teachers"** (**"GroupID"**, **"TeacherID"**) **values**

(1, 1),

(1, 2),

(2, 2),

(3, 3),

(3, 2);

**insert** **into** **"Specialties"** (**"ID"**, **"Title"**) **values**

(1, **'Информационные технологии'**),

(2, **'Экономика'**),

(3, **'Медицина'**);

**insert** **into** **"Students"** (**"ID"**, **"Name"**, **"NumberGradebook"**, **"GroupID"**) **values**

(1, **'Иванов Иван'**, 12345, 1),

(2, **'Петров Петр'**, 54321, 2),

(3, **'Сидоров Алексей'**, 98765, 3);

**insert** **into** **"Teachers"** (**"ID"**, **"Name"**) **values**

(1, **'Смирнова Ольга'**),

(2, **'Козлов Иван'**),

(3, **'Петрова Мария'**);

Формирование запросов

Сформируем запросы для проверки корректности заполнения базы данных после ее создания.

1. Выдавать информацию о студенте по № зачетной книжки.

**select** *s*."Name", *s*."NumberGradebook", "Groups"."GroupNumber", "Specialties"."Title" **AS** *Speciality*

**from** "Students" *s*

**join** "Groups" **on** *s*."GroupID" = "Groups"."ID"

**join** "Specialties" **on** "Groups"."SpecialityID" = "Specialties"."ID"

**where** *s*."NumberGradebook" = 98765;

|  |
| --- |
|  |
| 1. – Вывод информации о студенте по номеру зачетной книжки |

1. Выдавать информацию о студенте по имени.

**select** *s*."Name", *s*."NumberGradebook", "Groups"."GroupNumber", "Specialties"."Title" **AS** *Speciality*

**from** "Students" *s*

**join** "Groups" **on** *s*."GroupID" = "Groups"."ID"

**join** "Specialties" **on** "Groups"."SpecialityID" = "Specialties"."ID"

**where** *s*."Name" = **'Иванов Иван'**;

|  |
| --- |
|  |
| 1. – Вывод информации о студенте по имени |

1. Выдавать список предметов, читаемых данной кафедрой.

**select** "Disciplines"."Title"

**from** "Disciplines"

**join** "Departments" **on** "Disciplines"."DepartamentID" = "Departments"."ID"

**where** "Departments"."Title" = **'Факультет экономики'**;

|  |
| --- |
|  |
| 1. – Вывод списка предметов, читаемых данной кафедрой |

1. Выдавать список преподавателей, проводящих занятие в данной группе.

**select** "Teachers"."Name"

**from** "Teachers"

**join** "Disciplines\_Teachers" **on** "Teachers"."ID" = "Disciplines\_Teachers"."TeacherID"

**join** "Disciplines" **on** "Disciplines\_Teachers"."DisciplineID" = "Disciplines"."ID"

**join** "Groups\_Teachers" **on** "Teachers"."ID" = "Groups\_Teachers"."TeacherID"

**join** "Groups" **on** "Groups\_Teachers"."GroupID" = "Groups"."ID"

**where** "Disciplines"."Title" = **'Макроэкономика'** **and** "Groups"."GroupNumber" = 101;

|  |
| --- |
|  |
| 1. – Вывод списка преподавателей, проводящих занятие в данной группе |

1. Выдавать список групп, обучающихся на данной специальности.

**select** "Groups"."GroupNumber"

**from** "Groups"

**join** "Specialties" **on** "Groups"."SpecialityID" = "Specialties"."ID"

**where** "Specialties"."Title" = **'Экономика'**;

|  |
| --- |
|  |
| 1. – Вывод списка групп, обучающихся на данной специальности |

Создание функций

1. Выдавать информацию о студенте по № зачетной книжки.

**CREATE** **OR** **REPLACE** **FUNCTION** get\_student\_info(Number\_Gradebook **INT**)

**RETURNS** **TABLE** (

student\_id **int**,

student\_name **varchar**(50),

gradebook\_number **int**,

group\_number **int**,

specialty\_title **varchar**(100)

)

**AS** **$$**

**BEGIN**

**RETURN** QUERY

**SELECT**

s.**"ID"** **AS** student\_id,

s.**"Name"** **AS** student\_name,

s.**"NumberGradebook"** **AS** gradebook\_number,

g.**"GroupNumber"** **AS** group\_number,

sp.**"Title"** **AS** specialty\_title

**FROM**

**"Students"** s

**INNER** **JOIN** **"Groups"** g **ON** s.**"GroupID"** = g.**"ID"**

**INNER** **JOIN** **"Specialties"** sp **ON** g.**"SpecialityID"** = sp.**"ID"**

**WHERE**

s.**"NumberGradebook"** = Number\_Gradebook;

**END**;

**$$** **LANGUAGE** plpgsql;

|  |
| --- |
|  |
| 1. – Результат вызова функции |

1. Выдавать информацию о студенте по имени.

**CREATE** **OR** **REPLACE** **FUNCTION** get\_student\_info(p\_student\_name **varchar**(50))

**RETURNS** **TABLE** (

student\_id **int**,

student\_name **varchar**(50),

gradebook\_number **int**,

group\_number **int**,

specialty\_title **varchar**(100)

)

**AS** **$$**

**BEGIN**

**RETURN** QUERY

**SELECT**

s.**"ID"** **AS** student\_id,

s.**"Name"** **AS** student\_name,

s.**"NumberGradebook"** **AS** gradebook\_number,

g.**"GroupNumber"** **AS** group\_number,

sp.**"Title"** **AS** specialty\_title

**FROM**

**"Students"** s

**INNER** **JOIN** **"Groups"** g **ON** s.**"GroupID"** = g.**"ID"**

**INNER** **JOIN** **"Specialties"** sp **ON** g.**"SpecialityID"** = sp.**"ID"**

**WHERE** s.**"Name"** = p\_student\_name;

**END**;

**$$** **LANGUAGE** plpgsql;

|  |
| --- |
|  |
| 1. – Результат вызова функции |

1. Выдавать список предметов, читаемых данной кафедрой.

**CREATE** **OR** **REPLACE** **FUNCTION** get\_department\_disciplines(p\_department\_title **varchar**(100))

**RETURNS** **TABLE** (

discipline\_id **int**,

discipline\_title **varchar**(100)

)

**AS** **$$**

**BEGIN**

**RETURN** QUERY

**SELECT**

d.**"ID"** **AS** discipline\_id,

d.**"Title"** **AS** discipline\_title

**FROM**

**"Disciplines"** d

**INNER** **JOIN** **"Departments"** dep **ON** d.**"DepartamentID"** = dep.**"ID"**

**WHERE**

dep.**"Title"** = p\_department\_title;

**END**;

**$$** **LANGUAGE** plpgsql;

|  |
| --- |
|  |
| 1. – Результат вызова функции |

1. Выдавать список преподавателей, проводящих занятие в данной группе.

**CREATE** **OR** **REPLACE** **FUNCTION** get\_teachers\_in\_group(p\_group\_id **int**)

**RETURNS** **TABLE** (

teacher\_id **int**,

teacher\_name **varchar**(50)

)

**AS** **$$**

**BEGIN**

**RETURN** QUERY

**SELECT**

t.**"ID"** **AS** teacher\_id,

t.**"Name"** **AS** teacher\_name

**FROM**

**"Teachers"** t

**INNER** **JOIN** **"Groups\_Teachers"** gt **ON** t.**"ID"** = gt.**"TeacherID"**

**WHERE** gt.**"GroupID"** = p\_group\_id;

**END**;

**$$** **LANGUAGE** plpgsql;

|  |
| --- |
|  |
| 1. – Результат вызова функции |

1. Выдавать список групп, обучающихся на данной специальности.

**CREATE** **OR** **REPLACE** **FUNCTION** get\_specialty\_groups(p\_specialty\_title **varchar**(100))

**RETURNS** **TABLE** (

group\_id **int**,

group\_number **int**,

specialty\_title **varchar**(100)

)

**AS** **$$**

**BEGIN**

**RETURN** QUERY

**SELECT**

g.**"ID"** **AS** group\_id,

g.**"GroupNumber"** **AS** group\_number,

s.**"Title"** **AS** specialty\_title

**FROM**

**"Groups"** g

**INNER** **JOIN** **"Specialties"** s **ON** g.**"SpecialityID"** = s.**"ID"**

**WHERE**

s.**"Title"** = p\_specialty\_title;

**END**;

**$$** **LANGUAGE** plpgsql;

|  |
| --- |
|  |
| 1. – Результат вызова функции |

Создание процедур

1. Процедура добавления записи в таблицу "Specialties".

**CREATE** **OR** **REPLACE** **PROCEDURE** add\_specialty(p\_id **int**, p\_title **varchar**(100))

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**INSERT** **INTO** **"Specialties"** (**"ID"**, **"Title"**)

**VALUES** (p\_id, p\_title);

**END**;

**$$**;

**CALL** add\_specialty(4, **'Английский язык'**);

|  |
| --- |
|  |
| 1. – Результат вызова процедуры |

1. Процедура добавления записи в таблицу "Groups".

**CREATE** **OR** **REPLACE** **PROCEDURE** add\_group(p\_id **int**, p\_group\_number **int**, p\_speciality\_id **int**)

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**INSERT** **INTO** **"Groups"** (**"ID"**, **"GroupNumber"**, **"SpecialityID"**)

**VALUES** (p\_id, p\_group\_number, p\_speciality\_id);

**END**;

**$$**;

**CALL** add\_group(4, 401, 4);

|  |
| --- |
|  |
| 1. – Результат вызова процедуры |

1. Процедура добавления записи в таблицу "Students".

**CREATE** **OR** **REPLACE** **PROCEDURE** add\_student(p\_id **int**, p\_name **varchar**(50), p\_number\_gradebook **int**, p\_group\_id **int**)

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**INSERT** **INTO** **"Students"** (**"ID"**, **"Name"**, **"NumberGradebook"**, **"GroupID"**)

**VALUES** (p\_id, p\_name, p\_number\_gradebook, p\_group\_id);

**END**;

**$$**;

**CALL** add\_student(4, **'Козлов Сергей'**, 10255, 4);

|  |
| --- |
|  |
| 1. – Результат вызова процедуры |

1. Процедура добавления записи в таблицу "Departments".

**CREATE** **OR** **REPLACE** **PROCEDURE** add\_department(p\_id **int**, p\_title **varchar**(100))

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**INSERT** **INTO** **"Departments"** (**"ID"**, **"Title"**)

**VALUES** (p\_id, p\_title);

**END**;

**$$**;

**CALL** add\_department(4, **'Факультет иностранных языков'**);

|  |
| --- |
|  |
| 1. – Результат вызова процедуры |

1. Процедура добавления записи в таблицу "Disciplines".

**CREATE** **OR** **REPLACE** **PROCEDURE** add\_discipline(p\_id **int**, p\_title **varchar**(100), p\_department\_id **int**)

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**INSERT** **INTO** **"Disciplines"** (**"ID"**, **"Title"**, **"DepartamentID"**)

**VALUES** (p\_id, p\_title, p\_department\_id);

**END**;

**$$**;

**CALL** add\_discipline(4, **'Французский язык'**, 4);

|  |
| --- |
|  |
| 1. – Результат вызова процедуры |

1. Процедура добавления записи в таблицу "Teachers".

**CREATE** **OR** **REPLACE** **PROCEDURE** add\_teacher(p\_id **int**, p\_name **varchar**(50))

**LANGUAGE** plpgsql

**AS** **$$**

**BEGIN**

**INSERT** **INTO** **"Teachers"** (**"ID"**, **"Name"**)

**VALUES** (p\_id, p\_name);

**END**;

**$$**;

**CALL** add\_teacher(4, **'Захарова Татьяна'**);

|  |
| --- |
|  |
| 1. – Результат вызова процедуры |

Вывод

В результате создания курсового проекта была спроектирована, создана и заполнена база данных для информационного обслуживания работников деканата. Все тестовые запросы к базе данных «ВУЗ» выполняются корректно.

Были изучены операторы SQL на примере диалекта СУБД Postgres Pro, необходимые для фильтрации набора данных; операторы, необходимые для работы с данными, находящимися в разных таблицах, получен навык создания запросов на соединение данных из множества таблиц. Изучены операторы языка определения данных DDL и операторы SQL диалекта PostgreSQL, необходимые для вставки, удаления и модификации данных.

Список литературы

1. Балакина Е. П., Васильева М. А., Филипченко К. М. Информационное обеспечение систем управления. Методические указания к курсовому проектированию. Учебно-методическое пособие. Издание второе, исправленное и дополненное, 2023. 102 –с.
2. Васильева М. А., Степанов Д. Е. Операторы языка определения данных. Учебно– методическое пособие. – М.: РУТ (МИИТ). 2024. – 33с
3. Васильева М. А., Балакина Е. П., Васильева М. В. Соединение данных из множества таблиц. Рекомендации по выполнению работы и перечень типовых заданий. Учебно–методическое пособие. Издание второе, исправленное. – М.: РУТ (МИИТ).2024. – 72с.
4. Васильева М. А, Филипченко К. М., Пугачёв В. А. Фильтрация набора данных. Рекомендации по выполнению работы и перечень типовых заданий: Учебно-методическое пособие. Издание третье, исправленное и дополненное. Москва–2024, 133 с.
5. Васильева М. А., Кузнецов К. К. Операторы языка управления данными. Учебно– методическое пособие. – М.: РУТ (МИИТ).2024. – 36с.