

Оглавление

Расширенное задание3

Введение. 6

1. Проектирование БД10
2. Ссылочная целостность.. 14
3. Протокол настройки СУБД и создания БД.. 15
4. Протокол работы программы-клиента17

5. Заключение32

6. Список источников 33

7. SQL-скрипт..34

8. Исходный код клиента...44

**Расширенное задание**

|  |  |  |  |
| --- | --- | --- | --- |
| 21 | Организация учебного процесса на кафедре вуза | Ведение БД учебной нагрузки кафедры, учебных и рабочих планов (какая кафедра какие дисциплины читает каким специальностям) (без учета преподавателей) | 1 |

|  |  |  |  |
| --- | --- | --- | --- |
| № пункта задания | Краткое наименование (суть критерия) | Дополнительное требование | Балл |
| 1 | Соответствие схемы БД схеме задания (п.1 задания на КР) |  |  |
| 2 | Индексирование БД (физическая и логическая организация БД) |  |  |
| 3 | Реализация и объяснение работы запросов (п.3.) в SQL-скриптах | CASE |  |
| VIEW |  |
| Подзапрос в SELECT |  |
| Подзапрос в FROM |  |
| Подзапрос в WHERE |  |
| Коррелированный подзапрос |  |
| Агрегативные функции, группировка записей |  |
| Предикаты ANY(SOME) или ALL |  |
| 4 | Объяснение работы триггеров |  |  |
| 5 | Использование хранимых процедур |  |  |
| 6 | Использование в запросах пользовательских хранимых функций |  |  |
| 7 | Реализация и объяснение работы курсора |  |  |
| 8 | Права, роли, разграничение доступа, разрешение и запрет |  |  |
| 9 | Демонстрация работы клиента в виде отдельного приложения |  |  |
| 10 | Ввод, модификация, удаление данных на стороне клиента |  |  |
| 11 | Демонстрация работы запросов на стороне клиента (п.2 задания на КР) |  |  |
| 12 | Оформление КР | Пункты 1-11 в части оформления курсовой работы |  |



**ВВЕДЕНИЕ**

База данных представляет собой упорядоченный набор структурированной информации или данных, которые обычно хранятся в электронном виде в компьютерной системе. База данных обычно управляется [системой управления базами данных (СУБД)](https://www.oracle.com/cis/database/what-is-database/#WhatIsDBMS). Данные вместе с СУБД, а также приложения, которые с ними связаны, называются системой баз данных, или, для краткости, просто базой данных.

Базы данных бывают централизованными, которые хранятся на одном компьютере, и распределенными, которые хранятся на нескольких компьютерах некоторой сети.

Логическую структуру хранимых в базе данных называют моделью представления данных. К основным моделям представления данных относятся следующие: иерархическая, сетевая, реляционная, постреляционная, многомерная и объектно-ориентированная.

Система управления базами данных (СУБД) – это комплекс языковых и программных средств, предназначенных для создания, ведения и совместного использования БД многими пользователями. Обычно СУБД различают по используемой модели данных. Так, СУБД, основанные на использовании реляционной модели данных, называют реляционными СУБД.

Приложение представляет собой программу или комплекс программ, обеспечивающих автоматизацию обработки информации для прикладной задачи. Приложения, разработанные в среде СУБД, часто называют приложениями СУБД, а приложения, разработанные вне СУБД, – внешними приложениями.

Достоинством организации информационной системы по архитектуре клиент-сервер является сочетание централизованного хранения, обслуживания и коллективного доступа к общей корпоративной информации с индивидуальной работой пользователей над персональной информацией.

Важнейшим достоинством применения БД в информационных системах является обеспечение независимости данных от прикладных программ. Это дает возможность пользователям не заниматься проблемами представления данных на физическом уровне: размещения данных в памяти, методов доступа к ним и т. д.

SQL (Structured Query Language) – Декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

Запрос SQL - это запрос, создаваемый при помощи инструкций SQL.

Обеспечение целостности БД составляет необходимое условие функционирования БД.

Целостность БД – свойство БД, означающее, что база данных содержит полную и непротиворечивую информацию, необходимую и достаточную для корректного функционирования приложений

Нормальная форма — свойство отношения в реляционной модели данных, характеризующее его с точки зрения избыточности, потенциально приводящей к логически ошибочным результатам выборки или изменения данных. Нормальная форма определяется как совокупность требований, которым должно удовлетворять отношение.

Процесс преобразования отношений базы данных к виду, отвечающему нормальным формам, называется нормализацией. Нормализация — это процесс организации данных в базе данных, включающий создание таблиц и установление отношений между ними в соответствии с правилами, которые обеспечивают защиту данных и делают базу данных более гибкой, устраняя избыточность и несогласованные зависимости.

Первая нормальная форма

Переменная отношения находится в первой нормальной форме тогда и только тогда, когда в любом допустимом значении отношения каждый его кортеж содержит только одно значение для каждого из атрибутов.

Вторая нормальная форма

Переменная отношения находится во второй нормальной форме тогда и только тогда, когда она находится в первой нормальной форме и каждый неключевой атрибут неприводимо (функционально полно) зависит от ее потенциального ключа.

Третья нормальная форма

Переменная отношения находится в третьей нормальной форме тогда и только тогда, когда она находится во второй нормальной форме, и отсутствуют транзитивные функциональные зависимости неключевых атрибутов от ключевых.

PostgreSQL — свободная объектно-реляционная система управления базами данных (СУБД). PostgreSQL базируется на языке SQL

Реляционная база данных -это набор данных с предопределенными связями между ними. Эти данные организованны в виде набора таблиц, состоящих из столбцов и строк. В таблицах хранится информация об объектах, представленных в базе данных. В каждом столбце таблицы хранится определенный тип данных, в каждой ячейке – значение атрибута.

Существует три вида связей между таблицами. Тип создаваемых связей зависит от того, как определяются связанные столбцы.

Связь "один ко многим" являются наиболее распространенным типом связи. В такого рода связях строка в таблице А может иметь много строк в таблице B. Но строка в таблице B может иметь только одну строку в таблице А. Например, таблицы "Издатели" и "Названия" имеют связь "один ко многим". То есть, каждый издатель выпускает много названий. Но каждое название принадлежит только одному издателю.

Связь "один ко многим" создается, если только один из связанных столбцов является основным ключом или имеет уникальное ограничение.

В связи "один к одному" строка в таблице А может иметь не более одной совпадающей строки в таблице B, и наоборот. Связь "один к одному" создается, если оба связанных столбца являются первичными ключами или имеют уникальные ограничения.

Этот тип отношений не распространен, потому что большая часть информации, которая связана таким образом, будет в одной таблице.

В связи "многие ко многим" строка в таблице А может иметь много совпадающих строк в таблице B, и наоборот. Вы создаете такую связь, определяя третью таблицу, которая называется промежуточной таблицей. Первичный ключ промежуточной таблицы состоит из внешних ключей как таблицы А, так и таблицы B.

Проектирование БД

Процесс проектирования включает следующие этапы:

* Определение назначения базы данных

База данных kurs предназначена для организации учебной деятельности на кафедре.

* Разделение данных по таблицам

Первая таблица содержит информацию о кафедрах, вторая о дисциплинах, третья о специальностях, четвертая о дисциплинах, читаемых на конкретных специальностях.

* Задание первичных ключей
  + - Таблица кафедры: первичный ключ – код кафедры
    - Таблица дисциплины: первичный ключ – код дисциплины
    - Таблица специальностей: первичный ключ – код специальности
    - Таблица Дисциплин на специальности: первичный ключ – код дисциплины
* Настройка связей между таблицами

Связь между таблицей кафедры и дисциплины – один ко многим

Связь между таблицей дисциплины и дисциплины на специальности – один ко многим

Связь между таблицей специальности и дисциплины на специальности – один ко многим

* Применение правил нормализации

База данных kurs находится в 1 НФ, потому что отношении нет одинаковых кортежей, все значения атрибутов атомарны.

База данных находится во второй нормальной форме (2НФ) потому что она находится в 1НФ и нет неключевых атрибутов, зависящих от части сложного ключа.

База данных находиться в третьей нормальной форме, она приведена ко второй нормальной форме и каждый не ключевой столбец независим друг от друг.

Инфологическая модель

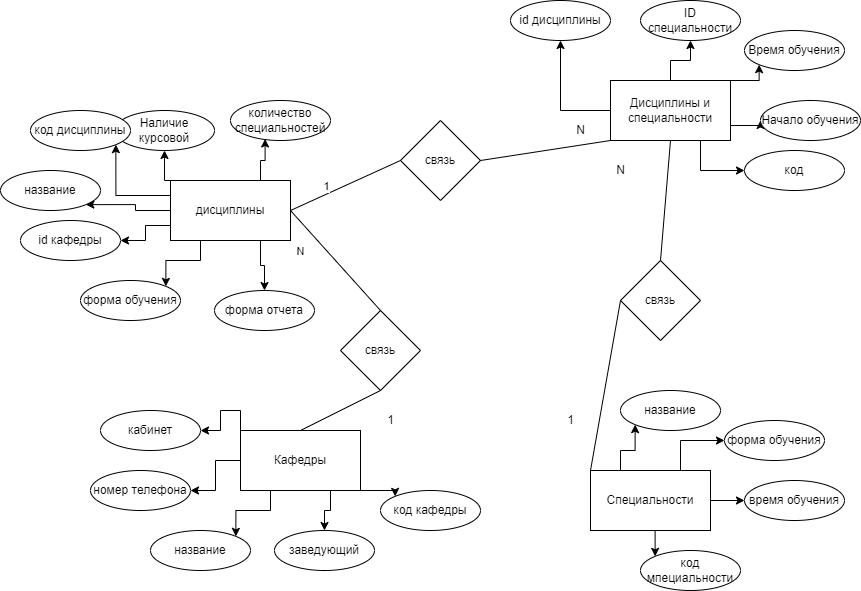


Рисунок 1. – Инфологическая модель.

Даталогическая модель

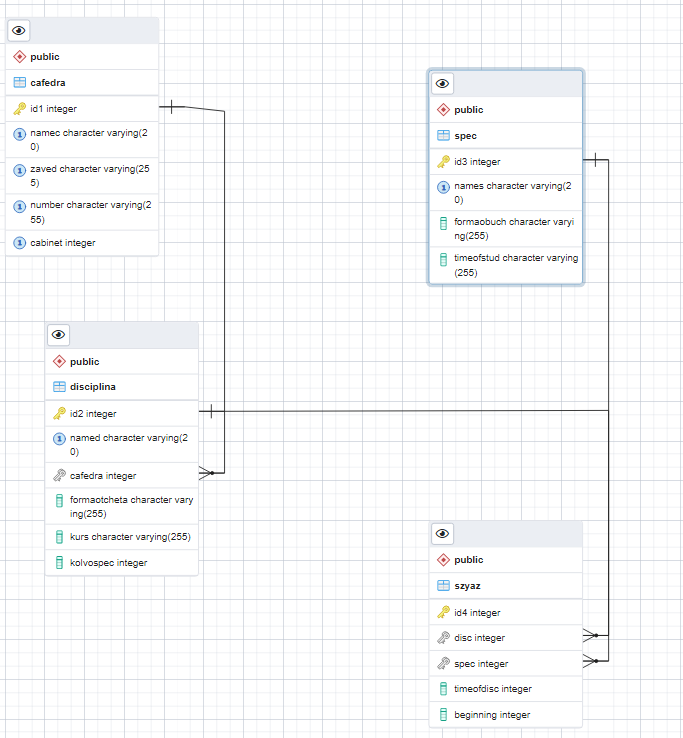


Рисунок 2. – Даталогическая модель.

В данном случае в таблице:

1. Cafedra – таблица, содержит информацию о кафедрах
   1. Namec – название
   2. Zaved – заведующий
   3. Number – номер телефона
   4. Cabinet – номер кабинета
2. Disciplina – таблица, содержащая информацию о дисциплинах и институте
   1. Named – название
   2. Cafedra – id кафедры
   3. Formaobuch – форма обучения
   4. Formaotcheta – форма сдачи экзамена
   5. Kurs – есть ли курсовая работа
   6. Kolvospec – количство специальностей, ка которых изучается данная дисциплина
3. Szyaz – таблица, содержащая информацию о дисциплинах, читаемых на конкретных специальностях
   1. Disc – id дисциплины
   2. Spec – id специальности
   3. Timeofdisc – время обучения
   4. Beginning – начало обучения
4. Spec – таблица, содержащая информацию о специальностях
   1. Names – название
   2. Formaobuch – форма обучения
   3. Timeofstud – время обучения на специальности

Разграничение прав доступа

В базе данных kurs существует несколько пользователей с разными правами.

Пользователь usersel, роль которого – guests, имеет возможность только выборки данных из таблиц.

Пользователь user1, роль которого – admin, не имеет ограничений, являясь superuser.

Таким образом, при использовании клиент-серверного приложения будут выполнятся только те действия, которые доступны данному пользователю.

Ссылочная целостность данных.

В базе данных kurs ссылочная целостность обуспечена комбинацией первичных и внешних ключей.

* В таблице cafedra: id1 – PK
* В таблице disciplina: id2 – PK, cafedra – FK, который ограничен значениями id1 таблицы cafedra
* В таблице spec: id4 – PK
* В таблице szyaz: id3 – PK, disc – FK, который ограничен значениями id2 таблицы disciplina, spec – FK, ограниченный значениями id4 таблицы spec.

Для повышения эффективности доступа к данным используются индексы.

В таблице cafedra создан индекс i2 с методом доступа btree на столбец namec.

Также он устанавливает ограничение UNIQUE.

B-дерево, которое лежит в основе индекса - это самобалансирующееся дерево, которое поддерживает отсортированные данные и позволяет выполнять поиск, вставку, удаление и последовательный доступ за логарифмическое время.

Планировщик запросов PostgreSQL рассмотрит возможность использования индекса B-дерева всякий раз, когда столбцы индекса участвуют в сравнении.

Также создан хэш индекс i3 для таблицы disciplina на столбец named.

Индексы хеширования могут обрабатывать только простое сравнение равенства (=). Это означает, что всякий раз, когда индексированный столбец участвует в сравнении с использованием оператора равенства (=), планировщик запросов рассмотрит возможность использования хеш-индекса.

Протокол настройки СУБД и создания БД

1. Создание таблиц

* cafedra
* disciplina
* szyaz
* sepc

1. Индексы:

* btree индекс i2 таблицы cafedra, стлобец namec
* hash индекс индекс i3 таблицы disciplina, столбец named

1. Многотабличное представление viewn, которое осуществляет выборку из таблицы disciplina и cafedra
2. Хранимые процедуры:

* addcaf – добавление информации в таблицу cafedra
* delcaf – удаление из таблицы cafedra
* updcaf – редактирование информации таблицы cafedra
* deldisc – удаление информации из таблицы disciplina
* insdisc – добавление информации в таблицу disciplina
* upddisc – редактирование информации в таблице disciplina
* insspec – добавление информации в таблицу spec
* delspec – удаление информации из таблицы spec
* updspec – редактирование информации в таблице spec
* inssz - добавление информации в таблицу szyaz
* delsz – удаление информации из таблицы szyaz
* updsz – редактирование информации в таблице szyaz
* updprk2 – меняет namec из cafedra на новое значение с помощью курсора
* delall – функция, которая содержит транзакцию для удаления всех данных из таблиц, код для подтверждения изменений – 123.

1. Функции:

* function t1, t2, t3 – триггерные функции для обновления информации в kolvospec в таблице disciplina
* getsr – скалярная функция, возвращающая среднее время обучения на данной дисциплине по разным специальностям
* selcafedra – функция, возвращающая таблицу, где содержится информация об одной кафедре с определенным именем, задаваемым пользователем

Протокол работы программы-клиента

Авторизация

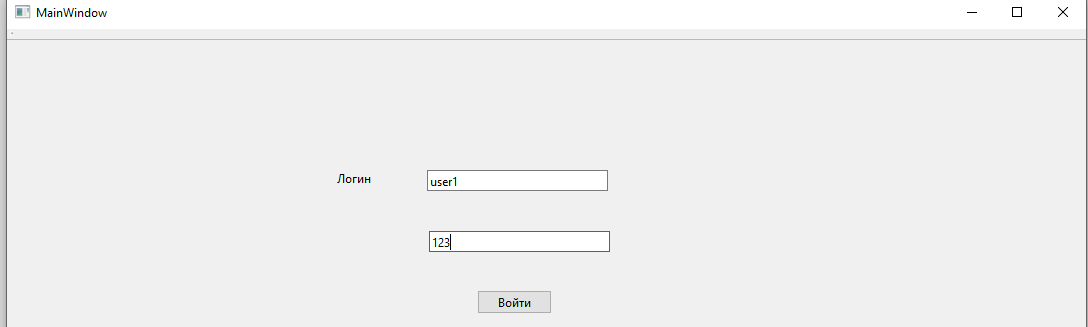


Рисунок 3. - Авторизация

Переход на главное окно, где можно вывести необходимую информацию, а также выполнить фнукиции selcafedra, getsr, delall. Вывод информации и- таблиц spec, cafedra, szyaz, disciplina показан на рисунках 4-7.

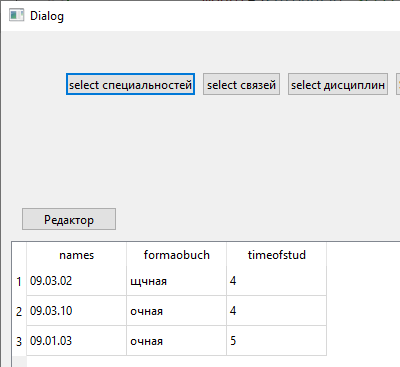


Рисунок 4. – select spec

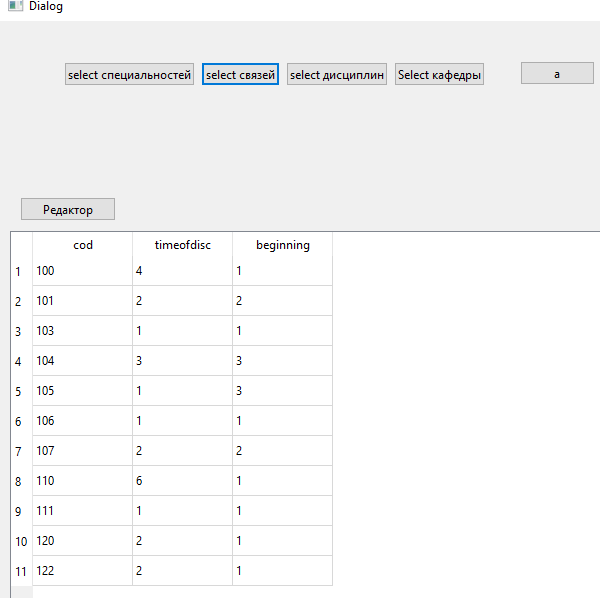


Рисунок 5. – select szyaz

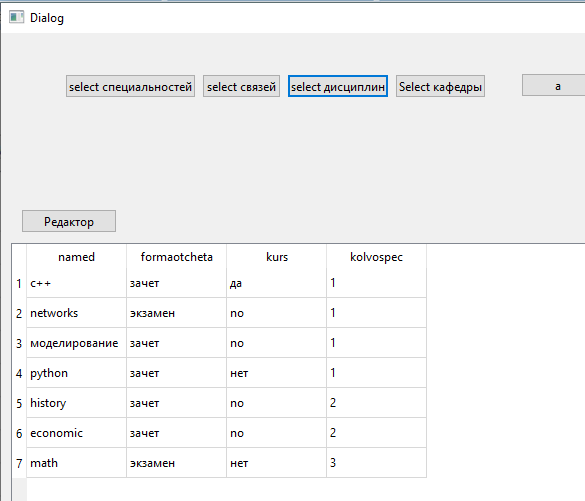


Рисунок 6. – disciplina

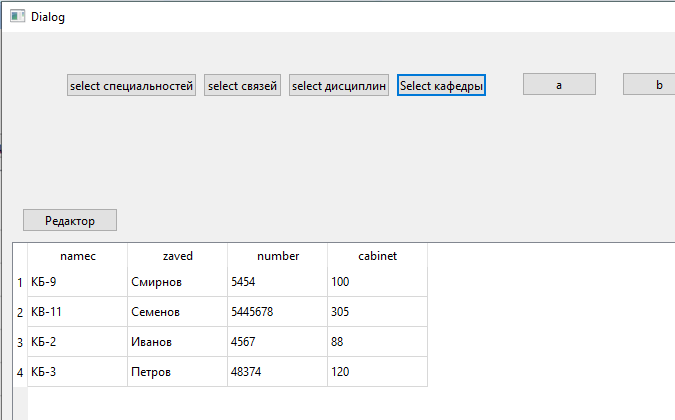


Рисунок 7. – select cafedra

При нажатии на кнопку a происходит вывод многотабличного запроса с case выражением,а именно вывод таблицы szyaz с информацией о специальности из таблицы spec

Запрос: "SELECT szyaz.cod, szyaz.timeofdisc, szyaz.beginning, spec.names, disciplina.named FROM szyaz RIGHT JOIN spec ON CASE spec.id3 WHEN szyaz.spec THEN 1 ELSE 0 END = 1 RIGHT JOIN disciplina on case disciplina.id2 when szyaz.disc THEN 1 ELSE 0 END = 1");

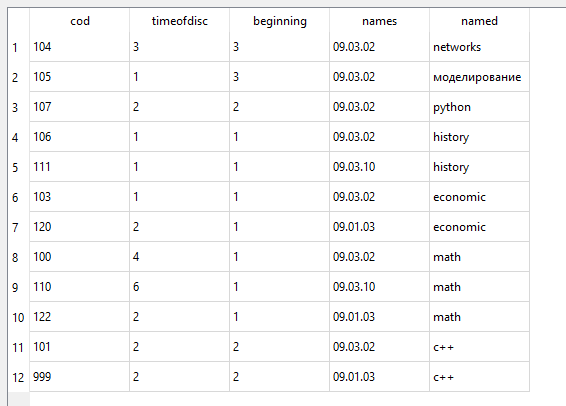


Рисунок 8.- a

Многотабличный view, вывод при нажатии на кнопку b.

Запрос " SELECT named, namec FROM viewn"

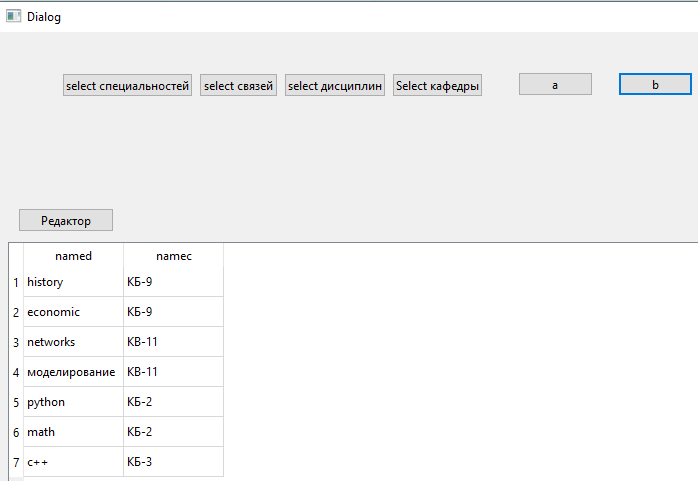


Рисунок 9. – b

Запросы содержащие подзапрос в разделах SELECT,FROM,WHERE выполняются по нажатию на кнопки c1,c2,c3.

Выборка из таблицы disciplina, а также zaved из cafedra

Запрос: SELECT named, (select zaved from cafedra where cafedra.id1 = disciplina.cafedra) as zaved from disciplina"

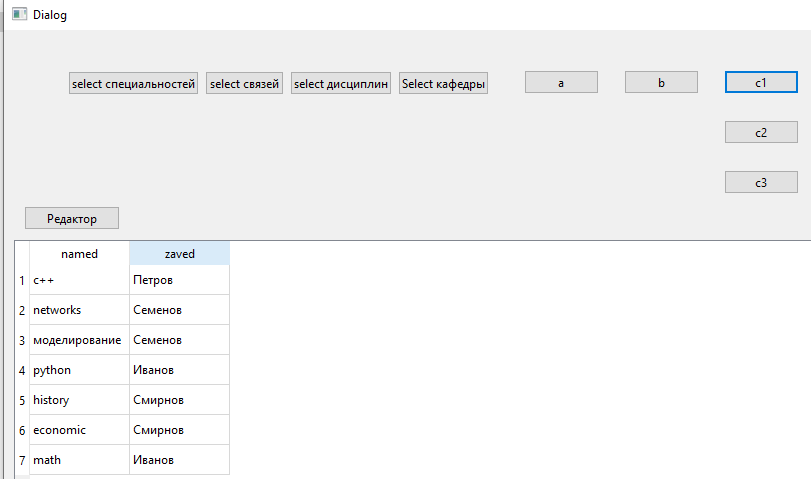


Рисунок 10. – c1

Запрос: SELECT named, formaotcheta FROM (SELECT \* FROM disciplina WHERE formaotcheta = 'экзамен') as a"

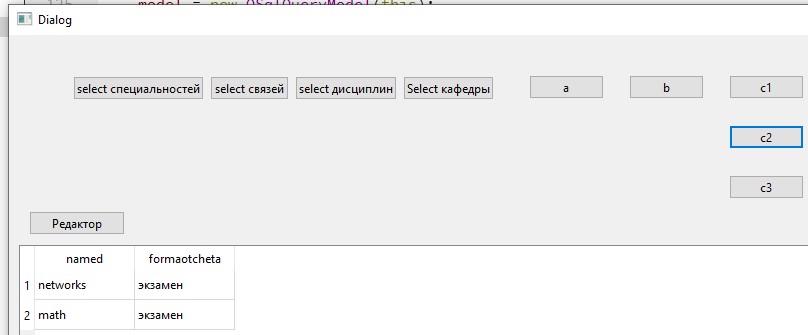


Рисунок 11.- c2

Запрос: SELECT named FROM disciplina WHERE id2 in (SELECT disc FROM Szyaz where beginning = 1)"

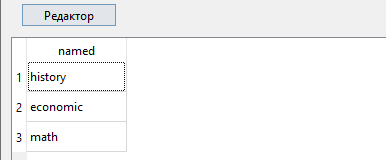


Рисунок 12.-c3

Коррелированные подзапросы

Запрос " SELECT a.namec, a.zaved, a.number from cafedra a where 'да' in "

"( select b.Kurs from disciplina b where b.cafedra = a.id1) "

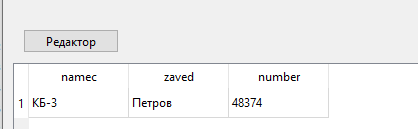


Рисунок 13. – d1

Запрос: SELECT disc, spec, (select cafedra from disciplina d where d.id2 = s.disc) as cafedra, "

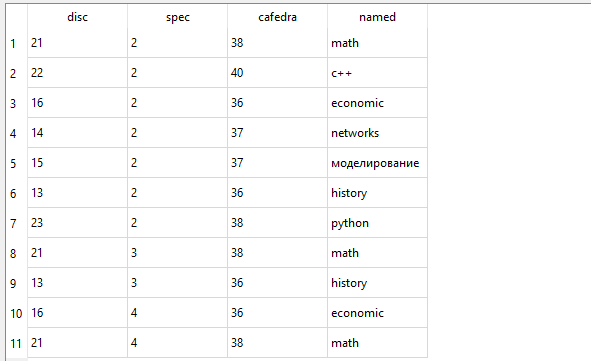
"(select named from disciplina d where d.id2 = s.disc)

Рисунок 14.- d2

Запрос.: "SELECT named , cafedra, (select zaved from cafedra where cafedra.id1 = disciplina.cafedra) as zaved from disciplina "

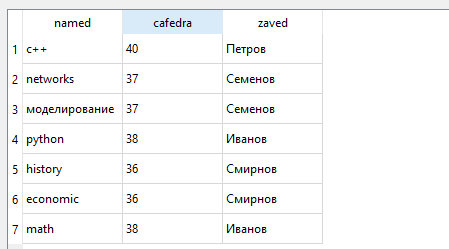


Рисунок 15.- d3

Многотабличный запрос с группировкой, агрегатной функцией и параметром в having, вызывается после нажатия кнопки e

Запрос: "SELECT named, formaotcheta, spec from disciplina JOIN Szyaz ON Szyaz.disc = disciplina.id2 GROUP BY named, formaotcheta, spec having MIN(timeofdisc) >2"

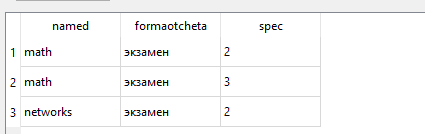


Рисунок 16.- e

Запросы с any, all

Запрос: "SELECT cod, timeofdisc from Szyaz where timeofdisc >= all (select timeofdisc from Szyaz)"

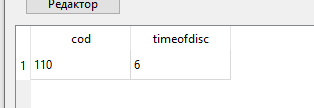


Рисунок 17. – f1

Запрос: "SELECT named, cod timeofdisc from (select \*from disciplina join Szyaz on disciplina.id2 = Szyaz.disc) as a where timeofdisc > some (select timeofdisc from Szyaz where beginning>=2)"

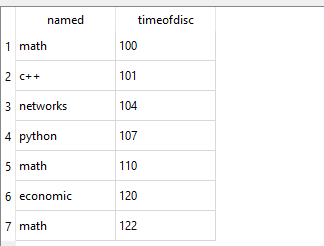


Рисунок 18.- f2

Запрос: "select named from disciplina where id2 = any (select disc from Szyaz)"

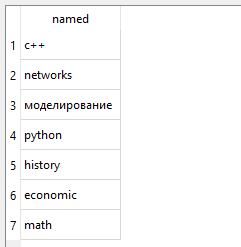


Рисунок 19. – f3

При нажатии на кнопку sk\_func происходит вызов функции getsr, которая возвращает среднее время обучения дисциплине на специальности, введенной пользователем(Рисунок 20).

db.open();

model = new QSqlQueryModel(this);

QSqlQuery **query**( db );

QString **l** = QString("select getsr('%1')").arg(ui->lineEdit->text());

model->setQuery(l);

Скрипт 1. – Обработка нажатия sk\_func

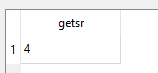


Рисунок 20.- sk\_func при вводе math

При нажатии на кнопку table\_func происходит вызов функции selcaf, которая возвращает таблицу с информацией только о той кафедре, имя которой ввел пользователь.

db.open();

model = new QSqlQueryModel(this);

QSqlQuery **query**( db );

QString **l** = QString("select namec, zaved, number, cabinet from public.selcafedra('%1')").arg(ui->lineEdit\_3->text());

model->setQuery(l);

ui->tableView->*setModel*(*model*);

Скрипт 2. – Обработка нажатия table\_func

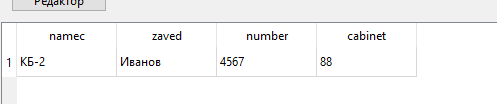


Рисунок 21. – table\_func при вводе значения КБ-2

db.open();

model = new QSqlQueryModel(this);

QSqlQuery **query**( db );

QString **l** = QString(" call public.updprk2('%1', '%2')").arg(ui->lineEdit\_2->text(),ui->lineEdit\_4->text() );

model->setQuery(l);

ui->tableView->*setModel*(*model*);

Скрипт 3. – Обработка нажатия кнопки curs

При вводе значений КБ-5 и KB-11 и нажатии на кнопку curs, имя кафедры меняется на КБ-5.

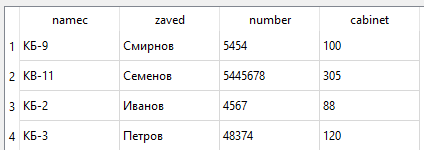


Рисунок 22. – cafedra до и после curs

При нажатии на кнопку «редактор» открывается следующее окно:

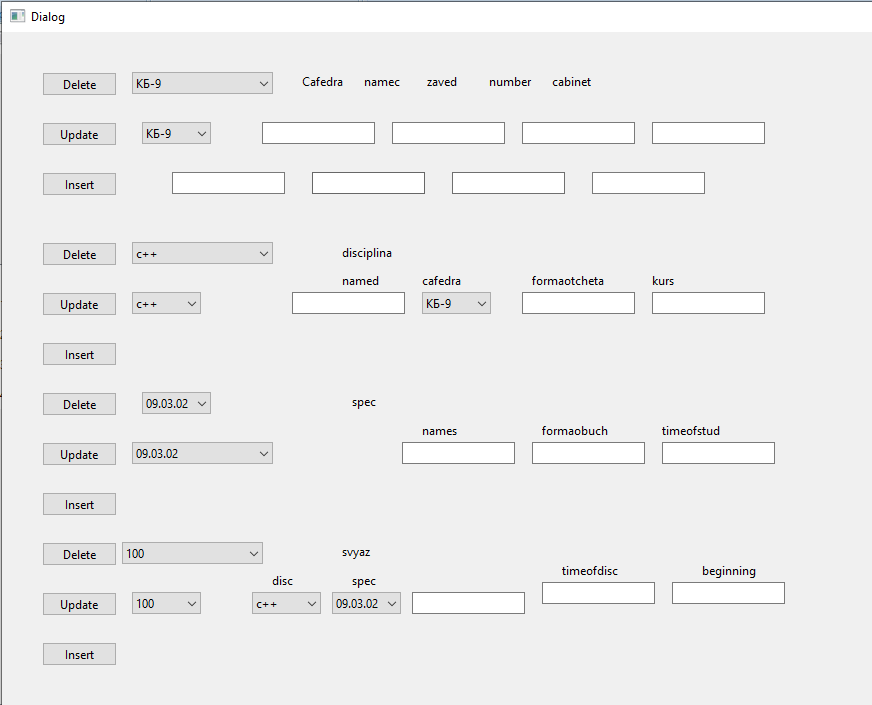


Рисунок 23.-редактор

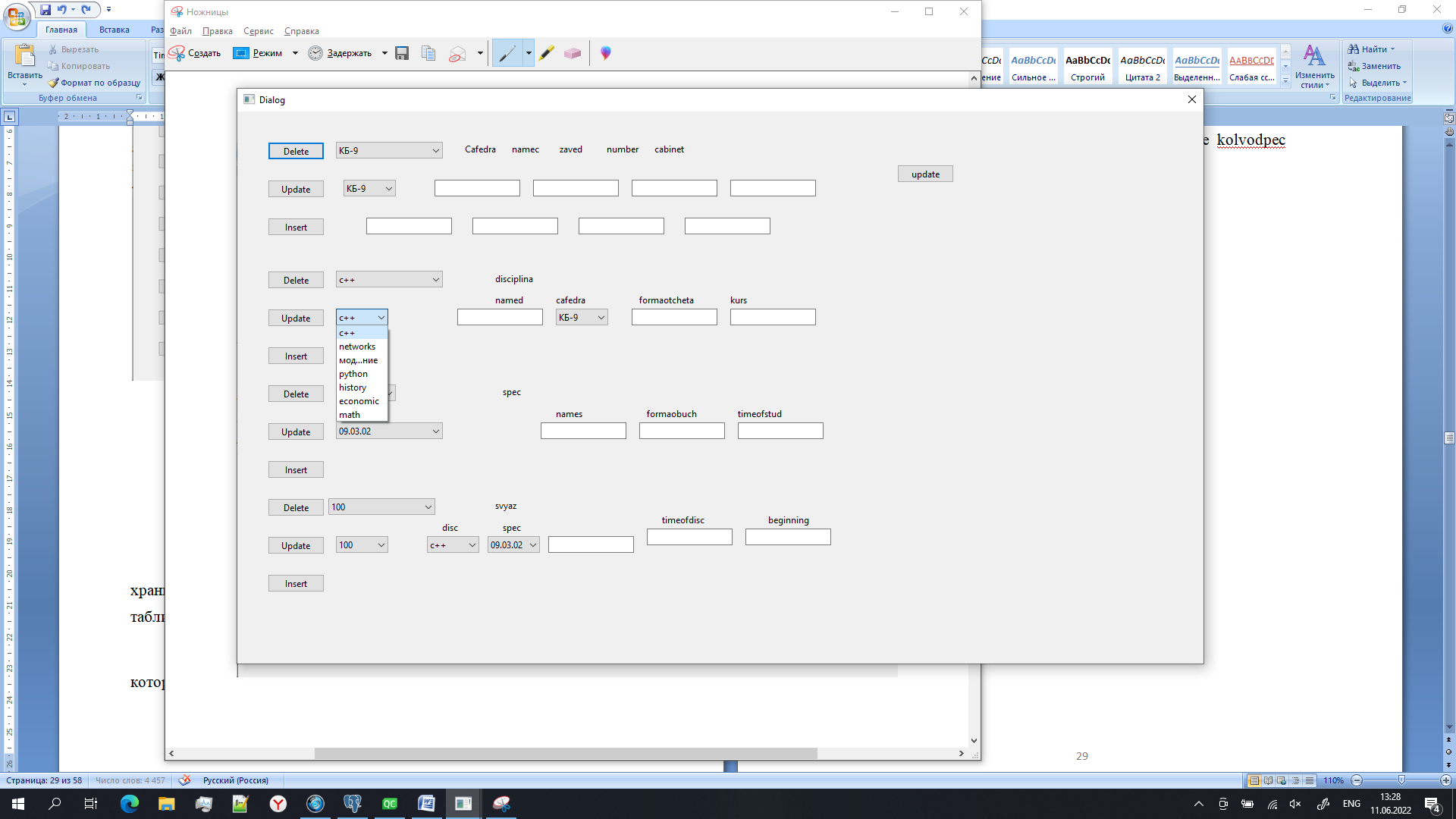


Рисунок 24. – Выбор значений столбцов

При вводе данных и нажатии на кнопку, выполняется соответствующая хранимая процедура для удаления, обновления или добавления информации в таблицу.

При изменении информации в таблице Szyaz срабатывает триггер, которые изменяет значение kolvospec в таблице disciplina

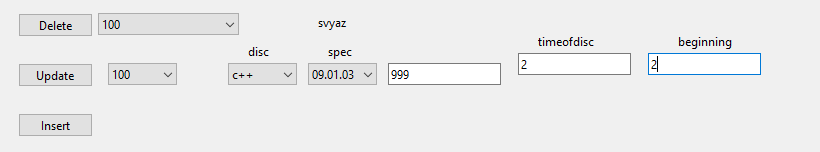


Рисунок 25. – Добавление информации в szyaz

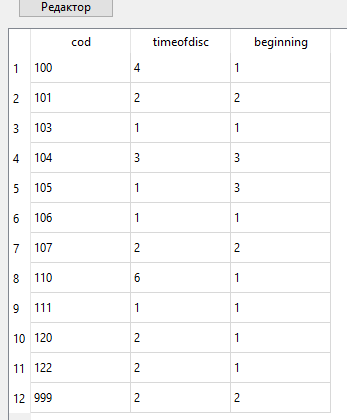


Рисунок 26. – Информация добавилась в szyaz

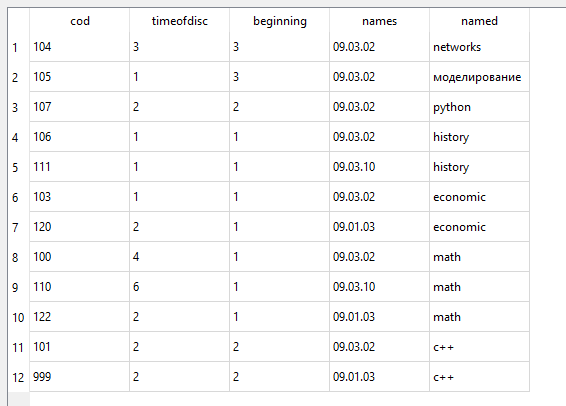


Рисунок 27. – Вывод запроса a



Рисунок 28. – Работа триггера и изменение kolvodpec

Заключение

В ходе выполнения курсовой работы было разработано клиент-серверное приложение для работы с базой данных. В приложении реализованы запросы согласно заданию, возможность добавления, удаления, обновления данных в БД. Создана БД с процедурами, индексами, триггерами, транзакциями, функциями, ролями и пользователями с различными привелегиями.

Список источников.

1. Докуметация Postgresql. [Электронный ресурс] https://postgrespro.ru/docs/postgresql (дата обращения 10.05.2022)
2. PostgreSql – Типы индесов. [Электронный ресурс] URL: https://progler.ru/blog/postgresql-tipy-indeksov
3. Основные сведенья о создании баз данных. [Электронный ресурс] URL: https://support.microsoft.com/ru-ru/office/основные-сведения-о-создании-баз-данных-eb2159cf-1e30-401a-8084-bd4f9c9ca1f5 (дата обращения 10.05.2022)
4. Что такое база данных. [Электронный ресурс] URL https://www.oracle.com/cis/database/what-is-database/ (дата обращения 10.05.2022)
5. QT Documentation. [Электронный ресурс] URL: https https://doc.qt.io/
6. Коррелирующие подзапросы [Электронный ресурс] Дата обновления – 20.07.2017 URL <https://metanit.com/sql/sqlserver/6.1.php> (дата обращения 10.05.2022)
7. Вложенные запросы. [Электронный ресурс] https://docs.microsoft.com/ru-ru/SQL/relational-databases/performance/subqueries?view=sql-server-2016 (дата обращения 10.05.2022)

Приложение A – SQL скрипт

CREATE TABLE cafedra(

id1 serial primary key,

namec VARCHAR(20) UNIQUE,

zaved VARCHAR(255) UNIQUE,

number VARCHAR(255) UNIQUE,

cabinet INT UNIQUE

);

create TABLE disciplina(

id2 serial primary key,

named VARCHAR(20) UNIQUE,

cafedra INT REFERENCES cafedra(id1),

formaotcheta VARCHAR(255),

Kurs VARCHAR(255),

kolvospec int

);

CREATE TABLE spec(

id3 serial primary key,

names VARCHAR(20) UNIQUE,

formaobuch VARCHAR(255),

timeofstud VARCHAR(255)

);

create TABLE Szyaz(

id4 serial PRIMARY KEY,

disc INT REFERENCES disciplina(id2),

spec INT REFERENCES spec(id3),

cod int UNIQUE,

timeofdisc int,

beginning int

);

ALTER TABLE Szyaz ADD CONSTRAINT a UNIQUE(disc, spec)

create VIEW viewn AS

SELECT \* FROM disciplina

RIGHT JOIN cafedra ON CASE cafedra.id1

WHEN disciplina.cafedra

THEN 1

ELSE 0

END = 1

update viewn set formaotcheta = 'очная',Kurs = 'no' WHERE named = 'math'

CREATE FUNCTION t1()

RETURNS TRIGGER

AS $$

BEGIN

UPDATE disciplina SET kolvospec = kolvospec+1 where id2 = NEW.disc;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

create trigger tr1 after insert

on szyaz

for each ROW

EXECUTE PROCEDURE t1()

------

CREATE FUNCTION t2()

RETURNS TRIGGER

AS $$

BEGIN

UPDATE disciplina SET kolvospec = kolvospec-1 where id2 = OLD.disc;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

create trigger tr2 after DELETE

on szyaz

for each ROW

EXECUTE PROCEDURE t2()

----------------

create FUNCTION t3()

RETURNS TRIGGER

AS $$

BEGIN

UPDATE disciplina SET kolvospec=kolvospec-1 where id2 = OLD.disc ;

UPDATE disciplina SET kolvospec=kolvospec+1 where id2 = NEW.disc;

RETURN NEW;

END;

$$ LANGUAGE plpgsql;

create trigger tr3 after UPDATE

on szyaz

for each ROW

EXECUTE PROCEDURE t3()

CREATE PROCEDURE addcaf(

v\_namec VARCHAR(20),

v\_zaved VARCHAR(20),

v\_number INT,

v\_cabinet INT)

LANGUAGE SQL

AS $$

INSERT INTO cafedra(namec, zaved, number, cabinet)

VALUES (v\_namec, v\_zaved, v\_number, v\_cabinet);

$$;

create PROCEDURE delcaf(

v\_id1 VARCHAR(20))

LANGUAGE SQL

AS $$

DELETE FROM Szyaz WHERE disc in (select id2 from disciplina where cafedra = (select id1 from cafedra where namec = v\_id1));

--DELETE FROM spec WHERE id3 in (select spec FROM Szyaz WHERE disc in (select id2 from disciplina where cafedra = @id1))

DELETE FROM disciplina WHERE cafedra = (select id1 from cafedra where namec = v\_id1);

DELETE FROM cafedra WHERE namec = v\_id1;

$$;

CREATE PROCEDURE updcaf(

v\_id1 VARCHAR(20),

v\_namec VARCHAR(20),

v\_zaved VARCHAR(20),

v\_number INT,

v\_cabinet INT)

LANGUAGE SQL

as $$

update cafedra

set namec = v\_namec, zaved = v\_zaved, number = v\_number, cabinet = v\_cabinet where v\_id1 = namec

$$;

create PROCEDURE insdisc(

vnamed VARCHAR(20),

vcafedra VARCHAR(20),

vformaotcheta VARCHAR(20),

vKurs VARCHAR(20))

LANGUAGE SQL

as $$

insert into disciplina(named, cafedra, formaotcheta,Kurs, kolvospec)

VAlues(vnamed, (select id1 from cafedra where namec = vcafedra),vformaotcheta, vKurs, 0);

$$;

CREATE PROCEDURE upddisc(

vid2 VARCHAR(20),

vnamed VARCHAR(20),

vcafedra INT,

vformaotcheta VARCHAR(20),

vKurs VARCHAR(20))

LANGUAGE SQL

as $$

update disciplina

set named = vnamed, cafedra = vcafedra, formaotcheta=vformaotcheta,Kurs = vKurs where vid2 = named

$$;

create PROCEDURE deldisc(

vid2 VARCHAR(20))

LANGUAGE SQL

AS $$

DELETE FROM Szyaz WHERE disc = (select id2 from disciplina where named = vid2);

DELETE FROM disciplina WHERE named = vid2;

$$;

create PROCEDURE delsz(

vid4 INT)

LANGUAGE SQL

AS $$

DELETE FROM Szyaz WHERE cod = vid4;

$$;

CREATE PROCEDURE updsz

(

vid4 int,

vdisc int,

vspec INT,

vtimeofdisc int,

vbeginning int)

LANGUAGE SQL

as $$

update Szyaz

set disc = vdisc, spec = vspec, timeofdisc=vtimeofdisc,beginning = vbeginning where cod = vid4

$$;

CREATE PROCEDURE inssz

(

vdisc varchar(20),

vspec varchar(20),

vcod int,

vtimeofdisc int,

vbeginning int)

LANGUAGE SQL

as $$

insert into Szyaz(disc, spec, cod, timeofdisc,beginning) VAlues((select id2 from disciplina where named = vdisc),(select id3 from spec where names = vspec),vcod, vtimeofdisc,vbeginning);

$$;

CREATE PROCEDURE insspec(

vnames VARCHAR(20),

vformaobuch VARCHAR(20),

vtimeofstud int)

LANGUAGE SQL

as $$

insert into spec(names, formaobuch, timeofstud) VAlues(vnames, vformaobuch, vtimeofstud);

$$;

create PROCEDURE delspec(

vid3 VARCHAR(20))

LANGUAGE SQL

AS $$

DELETE FROM Szyaz WHERE spec = (select id3 from spec where names = vid3);

DELETE FROM spec WHERE names = vid3;

$$;

CREATE PROCEDURE updspec(

vid3 VARCHAR(20),

vnames VARCHAR(20),

vformaobuch VARCHAR(20),

vtimeofstud int)

LANGUAGE SQL

as $$

update spec

set names = vnames, formaobuch = vformaobuch, timeofstud = vtimeofstud where vid3 = names;

$$;

create function getsr(vdisc VARCHAR(20))

RETURNS numeric(10,2)

as $$

declare vsr numeric(10,2);

begin

vsr = (SELECT AVG(timeofdisc) FROM Szyaz WHERE disc = (select id2 from disciplina where named = vdisc));

RETURN vsr;

end;

$$ LANGUAGE plpgsql;

create function selcafedra(vname VARCHAR(20))

returns table(namec varchar,zaved varchar,number varchar ,cabinet int)

as $$

select namec,zaved,number,cabinet from cafedra where namec = vname ;

$$ LANGUAGE sql;

select \* from public.selcafedra('КБ-2')

create PROCEDURE updprk2(

vupdname varchar(20),

vid VARCHAR(20))

LANGUAGE PLPGSQL

as $$

DECLARE curs CURSOR FOR SELECT namec from cafedra;

Did1 VARCHAR(20);

Damec VARCHAR(20);

begin

open curs ;--for update of namec

LOOP

FETCH curs into Did1;

exit when not found;

IF (Did1 = vid)

THEN

update cafedra set namec = vupdname where current of curs;

END IF;

END LOOP;

close curs;

end;

$$;

CREATE ROLE admin WITH

NOLOGIN

SUPERUSER

CREATEDB

CREATEROLE

INHERIT

NOREPLICATION

CONNECTION LIMIT -1

PASSWORD '123';

CREATE ROLE guests WITH

NOLOGIN

NOSUPERUSER

NOCREATEDB

NOCREATEROLE

INHERIT

NOREPLICATION

CONNECTION LIMIT -1;

create user user1;

GRANT admin TO user1;

create user usersel;

GRANT guests TO usersel;

ALTER ROLE user1

SUPERUSER

CREATEDB

CREATEROLE;

ALTER ROLE usersel

PASSWORD '123';

ALTER ROLE user1

PASSWORD '123';

grant select on spec to guests;

grant select on disciplina to guests;

grant select on Szyaz to guests;

grant select on cafedra to guests;

create procedure delall(i boolean )

LANGUAGE plpgsql

as $$

declare i2 boolean;

begin

i2=i;

--delete from disciplina

delete from szyaz;

if i2 = true then

commit;

else

rollback;

end if;

end;

$$ ;

commit

rollback

CREATE UNIQUE INDEX i2

ON public.cafedra USING btree

(namec)

;

CREATE INDEX i3

ON public.disciplina USING hash

(named)

;

Приложение B – Исходный код клинта

Cafedra.h

#ifndef CAFEDRA\_H

#define CAFEDRA\_H

#include<QSqlTableModel>

#include <QDialog>

#include "redcaf.h"

namespace Ui {

class cafedra;

}

class cafedra : public QDialog

{

Q\_OBJECT

public:

explicit cafedra(QWidget \*parent = nullptr);

~cafedra();

private slots:

void on\_pushButton\_clicked();

void on\_pushButton\_7\_clicked();

void on\_pushButton\_6\_clicked();

void on\_pushButton\_5\_clicked();

void on\_a\_butt\_clicked();

void on\_b\_butt\_clicked();

void on\_pushButton\_c1\_clicked();

void on\_pushButton\_c2\_clicked();

void on\_pushButton\_c3\_clicked();

void on\_pushButton\_d1\_clicked();

void on\_pushButton\_d2\_clicked();

void on\_pushButton\_d3\_clicked();

void on\_pushButton\_e\_clicked();

void on\_pushButton\_f1\_clicked();

void on\_pushButton\_f2\_clicked();

void on\_pushButton\_f3\_clicked();

void on\_pushButton\_8\_clicked();

void on\_pushButton\_c3\_2\_clicked();

void on\_pushButton\_c3\_3\_clicked();

void on\_pushButton\_2\_clicked();

void on\_pushButton\_3\_clicked();

private:

redcaf \*fc;

Ui::cafedra \*ui;

QSqlDatabase db;

QSqlQuery \*qry;

QSqlQueryModel \*model;

public slots:

void slot(QSqlDatabase db);

signals:

void signal2(QSqlDatabase);

};

#endif // CAFEDRA\_H

Mainwindow.h

#ifndef MAINWINDOW\_H

#define MAINWINDOW\_H

#include<QSqlTableModel>

#include <QDialog>

#include <QMainWindow>

#include"cafedra.h"

namespace Ui {

class MainWindow;

}

class MainWindow : public QMainWindow

{

Q\_OBJECT

public:

explicit MainWindow(QWidget \*parent = 0);

~MainWindow();

private slots:

void on\_pushButton\_clicked();

private:

Ui::MainWindow \*ui;

QSqlDatabase db;

cafedra \*form;

signals:

void signal(QSqlDatabase);

};

#endif // MAINWINDOW\_H

Redcaf.h

#ifndef REDCAF\_H

#define REDCAF\_H

#include <QSqlDatabase>

#include <QSqlQuery>

#include "qsqlquerymodel.h"

#include <QDialog>

namespace Ui {

class redcaf;

}

class redcaf : public QDialog

{

Q\_OBJECT

public:

explicit redcaf(QWidget \*parent = nullptr);

~redcaf();

private slots:

void on\_pushButton\_2\_clicked();

void on\_pushButton\_3\_clicked();

void on\_pushButton\_4\_clicked();

void on\_pushButton\_6\_clicked();

void on\_pushButton\_14\_clicked();

void on\_pushButton\_5\_clicked();

void on\_pushButton\_7\_clicked();

void on\_pushButton\_9\_clicked();

void on\_pushButton\_8\_clicked();

void on\_pushButton\_10\_clicked();

void on\_pushButton\_12\_clicked();

void on\_pushButton\_11\_clicked();

void on\_pushButton\_13\_clicked();

private:

QSqlDatabase db;

QSqlQuery qr;

QSqlQueryModel \*model;

Ui::redcaf \*ui;

public slots:

void slot2(QSqlDatabase db);

};

#endif // REDCAF\_H

Mainwindow.cpp

#include "mainwindow.h"

#include "ui\_mainwindow.h"

#include <QDebug>

#include<QMessageBox>

#include <QSqlDatabase>

#include <QSqlQuery>

#include <QSqlRecord>

#include <QSqlError>

#include <QtGui>

#include<QSqlTableModel>

#include <QDate>

MainWindow::MainWindow(QWidget \*parent) :

QMainWindow(parent),

ui(new Ui::MainWindow)

{

ui->setupUi(this);

form = new cafedra;

connect(this, &MainWindow::signal, form, &cafedra::slot);

}

MainWindow::~MainWindow()

{

delete ui;

}

void MainWindow::on\_pushButton\_clicked()

{

QString login = ui->lineEdit->text();

QString pass = ui->lineEdit\_2->text();

QSqlDatabase db = QSqlDatabase::addDatabase( "QPSQL" );

db.setDatabaseName( "kurs" );

db.setHostName("localhost");

db.setPort(5433);

db.setUserName(login);

db.setPassword(pass);

if(db.open()){

//QSqlQuery query( db );

this->close();

form->show();

emit signal(db);

}

}

Cafedra.cpp

#include "cafedra.h"

#include "ui\_cafedra.h"

#include<QSqlTableModel>

#include <QDebug>

#include<QMessageBox>

#include <QSqlDatabase>

#include <QSqlQuery>

#include <QSqlRecord>

#include <QSqlError>

cafedra::**cafedra**(QWidget \***parent**) :

QDialog(*parent*),

ui(new Ui::cafedra)

{

ui->setupUi(this);

fc = new redcaf;

connect(this, &cafedra::signal2, fc, &redcaf::slot2);

}

cafedra::~**cafedra**()

{

delete ui;

}

void cafedra::**on\_pushButton\_clicked**()

{

db.open();

//QSqlQuery query( db );

model = new QSqlQueryModel(this);

model->setQuery("SELECT namec, zaved, number, cabinet FROM cafedra");

ui->tableView->*setModel*(*model*);

}

void cafedra::**slot**(QSqlDatabase **db**)

{

this->db = db;

}

void cafedra::**on\_pushButton\_7\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery("SELECT names, formaobuch,timeofstud FROM spec");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_6\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery("SELECT named, formaotcheta, kurs,kolvospec FROM disciplina");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_5\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery("SELECT cod, timeofdisc, beginning FROM Szyaz");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_a\_butt\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery( "SELECT szyaz.cod, szyaz.timeofdisc, szyaz.beginning, spec.names, disciplina.named FROM szyaz RIGHT JOIN spec ON CASE spec.id3 WHEN szyaz.spec THEN 1 ELSE 0 END = 1 RIGHT JOIN disciplina on case disciplina.id2 when szyaz.disc THEN 1 ELSE 0 END = 1");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_b\_butt\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery(" SELECT named, namec FROM viewn");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_c1\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery(" SELECT named, (select zaved from cafedra where cafedra.id1 = disciplina.cafedra) as zaved from disciplina");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_c2\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery(" SELECT named, formaotcheta FROM (SELECT \* FROM disciplina WHERE formaotcheta = 'экзамен') as a");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_c3\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery(" SELECT named FROM disciplina WHERE id2 in (SELECT disc FROM Szyaz where beginning = 1)");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_d1\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery(" SELECT a.namec, a.zaved, a.number from cafedra a where 'да' in "

"( select b.Kurs from disciplina b where b.cafedra = a.id1) ");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_d2\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery("SELECT disc, spec, (select cafedra from disciplina d where d.id2 = s.disc) as cafedra, "

"(select named from disciplina d where d.id2 = s.disc)"

" from Szyaz s");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_d3\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery("SELECT named , cafedra, (select zaved from cafedra where cafedra.id1 = disciplina.cafedra) as zaved from disciplina ");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_e\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery("SELECT named, formaotcheta, spec from disciplina JOIN Szyaz ON Szyaz.disc = disciplina.id2 GROUP BY named, formaotcheta, spec having MIN(timeofdisc) >2");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_f1\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery("SELECT cod, timeofdisc from Szyaz where timeofdisc >= all (select timeofdisc from Szyaz)");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_f2\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery("SELECT named, cod timeofdisc from (select \*from disciplina join Szyaz on disciplina.id2 = Szyaz.disc) as a where timeofdisc > some (select timeofdisc from Szyaz where beginning>=2)");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_f3\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

model->setQuery("select named from disciplina where id2 = any (select disc from Szyaz)");

ui->tableView->*setModel*(*model*);

}

void cafedra::**on\_pushButton\_8\_clicked**()

{

// this->close();

fc->show();

emit signal2(db);

}

void cafedra::**on\_pushButton\_c3\_2\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

QSqlQuery **query**( db );

QString **l** = QString("select getsr('%1')").arg(ui->lineEdit->text());

model->setQuery(l);

ui->tableView->*setModel*(*model*);

ui->lineEdit->clear();

}

void cafedra::**on\_pushButton\_c3\_3\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

QSqlQuery **query**( db );

QString **l** = QString("select namec, zaved, number, cabinet from public.selcafedra('%1')").arg(ui->lineEdit\_3->text());

model->setQuery(l);

ui->tableView->*setModel*(*model*);

ui->lineEdit->clear();

}

void cafedra::**on\_pushButton\_2\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

QSqlQuery **query**( db );

QString **l** = QString(" call public.updprk2('%1', '%2')").arg(ui->lineEdit\_2->text(),ui->lineEdit\_4->text() );

model->setQuery(l);

ui->tableView->*setModel*(*model*);

ui->lineEdit\_2->clear();

ui->lineEdit\_4->clear();

}

void cafedra::**on\_pushButton\_3\_clicked**()

{

db.open();

model = new QSqlQueryModel(this);

QSqlQuery **query**( db );

if (ui->lineEdit\_5->text() == "123"){

QString **d** = QString(" call delall(true)");

query.exec(d);

}

else{

QString **d** = QString(" call delall(false)");

query.exec(d);

}

ui->lineEdit\_5->clear();

}

Redcaf.cpp

#include "redcaf.h"

#include "qsqlquerymodel.h"

#include "ui\_redcaf.h"

#include <QDebug>

#include <QComboBox>

#include <QVariant>

redcaf::redcaf(QWidget \*parent) :

QDialog(parent),

ui(new Ui::redcaf)

{

ui->setupUi(this);

//redcaf::on\_pushButton\_14\_clicked();

}

redcaf::~redcaf()

{

delete ui;

}

void redcaf::slot2(QSqlDatabase db)

{

this->db = db;

}

void redcaf::on\_pushButton\_2\_clicked()

{

QSqlQuery query( db );

db.open();

QString l = QString("call delcaf('%1')").arg(ui->comboBox->currentText());

query.exec(l);

redcaf::on\_pushButton\_14\_clicked();

}

void redcaf::on\_pushButton\_3\_clicked()

{

QSqlQuery query( db );

db.open();

QString l = QString("call updcaf('%1', '%2','%3', '%4','%5')").arg(ui->comboBox\_2->currentText(), ui->lineEdit\_u1->text(), ui->lineEdit\_u2->text(), ui->lineEdit\_u3->text(),ui->lineEdit\_u4->text());

query.exec(l);

redcaf::on\_pushButton\_14\_clicked();

}

void redcaf::on\_pushButton\_4\_clicked()

{

QString l = QString("call addcaf('%1', '%2','%3', '%4')").arg(ui->lineEdit\_i1->text(), ui->lineEdit\_i2->text(), ui->lineEdit\_i3->text(),ui->lineEdit\_i4->text());

QSqlQuery query( db );

db.open();

query.exec(l);

redcaf::on\_pushButton\_14\_clicked();

ui->lineEdit\_i1->clear();

ui->lineEdit\_i2->clear();

ui->lineEdit\_i3->clear();

ui->lineEdit\_i4->clear();

redcaf::on\_pushButton\_14\_clicked();

//db.close();

}

void redcaf::on\_pushButton\_6\_clicked()

{

QSqlQuery query( db );

db.open();

QString l = QString("call deldisc('%1')").arg(ui->comboBox\_4->currentText());

query.exec(l);

redcaf::on\_pushButton\_14\_clicked();

}

void redcaf::on\_pushButton\_14\_clicked()

{

QSqlQuery query( db );

db.open();

model = new QSqlQueryModel(this);

model->setQuery("SELECT namec FROM cafedra");

ui->comboBox->setModel(model);

model = new QSqlQueryModel(this);

model->setQuery("SELECT namec FROM cafedra");

ui->comboBox\_2->setModel(model);

model = new QSqlQueryModel(this);

model->setQuery("SELECT named FROM disciplina");

ui->comboBox\_3->setModel(model);

model = new QSqlQueryModel(this);

model->setQuery("SELECT named FROM disciplina");

ui->comboBox\_4->setModel(model);

model = new QSqlQueryModel(this);

model->setQuery("SELECT names FROM spec");

ui->comboBox\_9->setModel(model);

model = new QSqlQueryModel(this);

model->setQuery("SELECT names FROM spec");

ui->comboBox\_8->setModel(model);

model = new QSqlQueryModel(this);

model->setQuery("SELECT cod FROM szyaz");

ui->comboBox\_11->setModel(model);

model = new QSqlQueryModel(this);

model->setQuery("SELECT cod FROM szyaz");

ui->comboBox\_12->setModel(model);

model = new QSqlQueryModel(this);

model->setQuery("SELECT namec FROM cafedra");

ui->comboBox\_d2->setModel(model);

model = new QSqlQueryModel(this);

model->setQuery("SELECT named FROM disciplina");

ui->comboBox\_sz1->setModel(model);

model = new QSqlQueryModel(this);

model->setQuery("SELECT names FROM spec");

ui->comboBox\_sz2->setModel(model);

}

void redcaf::on\_pushButton\_5\_clicked()

{

QSqlQuery query( db );

db.open();

QString l = QString("call upddisc('%1', '%2','%3', '%4','%5')").arg(ui->comboBox\_3->currentText(), ui->lineEdit\_d1->text(), ui->comboBox\_d2->currentText(), ui->lineEdit\_d3->text(),ui->lineEdit\_d4->text());

query.exec(l);

ui->lineEdit\_d1->clear();

ui->lineEdit\_d3->clear();

ui->lineEdit\_d4->clear();

redcaf::on\_pushButton\_14\_clicked();

}

void redcaf::on\_pushButton\_7\_clicked()

{

QString l = QString("call insdisc('%1', '%2','%3', '%4')").arg(ui->lineEdit\_d1->text(), ui->comboBox\_d2->currentText(), ui->lineEdit\_d3->text(),ui->lineEdit\_d4->text());

QSqlQuery query( db );

db.open();

query.exec(l);

ui->lineEdit\_d1->clear();

ui->lineEdit\_d3->clear();

ui->lineEdit\_d4->clear();

redcaf::on\_pushButton\_14\_clicked();

}

void redcaf::on\_pushButton\_9\_clicked()

{

QSqlQuery query( db );

db.open();

QString l = QString("call delspec('%1')").arg(ui->comboBox\_9->currentText());

query.exec(l);

redcaf::on\_pushButton\_14\_clicked();

}

void redcaf::on\_pushButton\_8\_clicked()

{

QSqlQuery query( db );

db.open();

QString l = QString("call updspec('%1', '%2','%3', '%4')").arg(ui->comboBox\_8->currentText(), ui->lineEdit\_s1->text(), ui->lineEdit\_s2->text(), ui->lineEdit\_s3->text());

query.exec(l);

ui->lineEdit\_s1->clear();

ui->lineEdit\_s2->clear();

ui->lineEdit\_s3->clear();

redcaf::on\_pushButton\_14\_clicked();

}

void redcaf::on\_pushButton\_10\_clicked()

{

QString l = QString("call insspec('%1', '%2','%3')").arg(ui->lineEdit\_s1->text(), ui->lineEdit\_s2->text(), ui->lineEdit\_s3->text());

QSqlQuery query( db );

db.open();

query.exec(l);

ui->lineEdit\_s1->clear();

ui->lineEdit\_s2->clear();

ui->lineEdit\_s3->clear();

redcaf::on\_pushButton\_14\_clicked();

}

void redcaf::on\_pushButton\_12\_clicked()

{

QSqlQuery query( db );

db.open();

QString l = QString("call delsz('%1')").arg(ui->comboBox\_12->currentText());

query.exec(l);

redcaf::on\_pushButton\_14\_clicked();

}

void redcaf::on\_pushButton\_11\_clicked()

{

QSqlQuery query( db );

db.open();

QString l = QString("call updsz('%1', '%2','%3', '%4','%5')").arg(ui->comboBox\_11->currentText(), ui->comboBox\_sz1->currentText(), ui->comboBox\_sz2->currentText(), ui->lineEdit\_sz3->text(), ui->lineEdit\_sz4->text());

qDebug()<< l;

query.exec(l);

ui->lineEdit\_sz3->clear();

ui->lineEdit\_sz4->clear();

redcaf::on\_pushButton\_14\_clicked();

}

void redcaf::on\_pushButton\_13\_clicked()

{

QString l = QString("call inssz('%1', '%2','%3','%4','%5')").arg( ui->comboBox\_sz1->currentText(), ui->comboBox\_sz2->currentText(), ui->lineEdit\_sz4\_2->text(), ui->lineEdit\_sz3->text(), ui->lineEdit\_sz4->text());

QSqlQuery query( db );

db.open();

qDebug()<< l;

query.exec(l);

ui->lineEdit\_sz3->clear();

ui->lineEdit\_sz4->clear();

ui->lineEdit\_sz4\_2->clear();

redcaf::on\_pushButton\_14\_clicked();

}