МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное бюджетное образовательное учреждение высшего образования

«Тихоокеанский государственный университет»

Кафедра «Программное обеспечение вычислительной техники и автоматизированных систем»

Создание электронного аналога документа

Лабораторная работа №3

по дисциплине «Базы данных»

Выполнил студент Пшеничный Д. О.

Факультет, группа ФКФН, ПО(аб)-81

Руководитель работы  Федосеев А.А.

Хабаровск – 2021г.

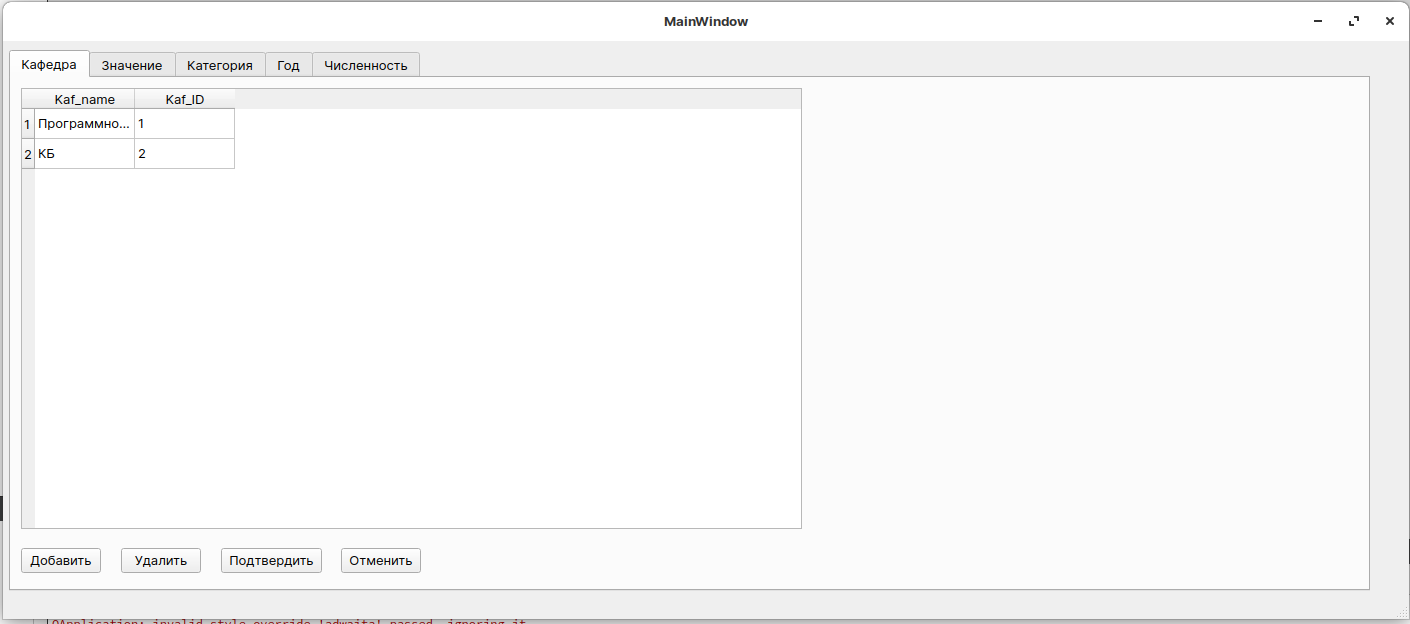
**Постановка задачи**

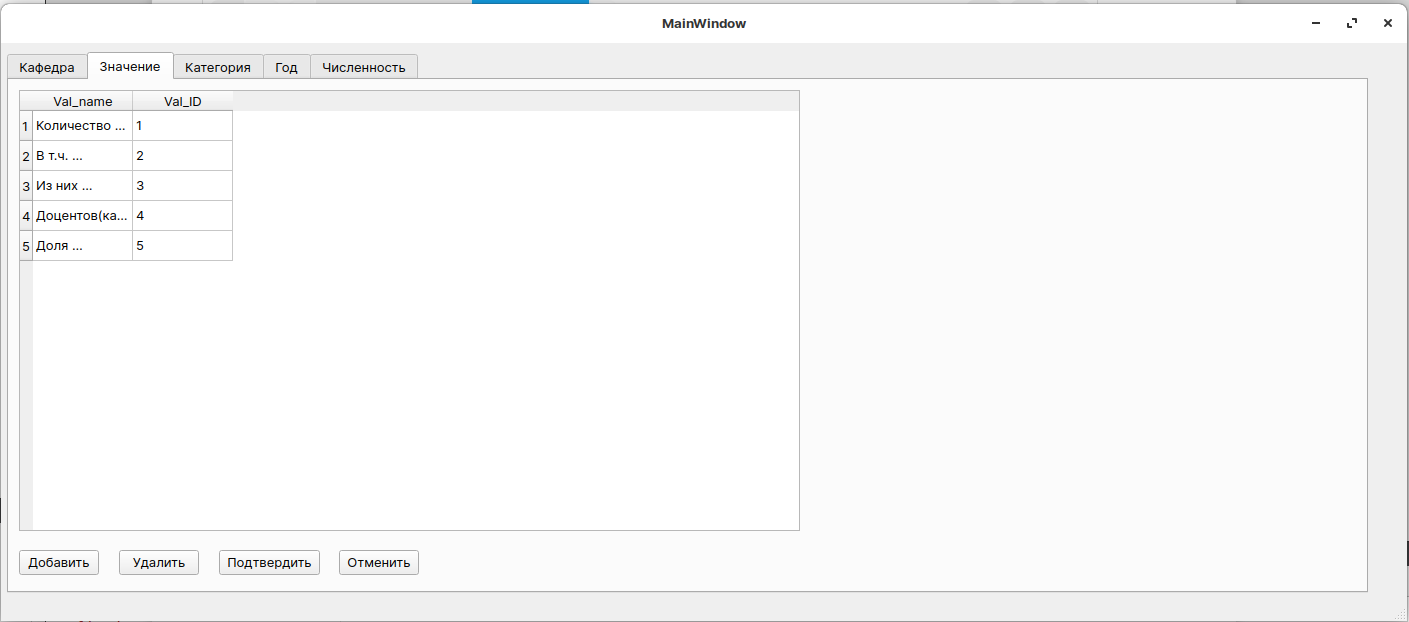
В процессе выполнения третьей лабораторной работы написать программу с использованием библиотеки Qt и СУБД SQLite, обеспечивающей работу с электронным аналогом документа. Под электронным аналогом понимается экранная форма для редактирования данных, максимально адаптированная под пользователя и похожая на выходной документ. В силу технических ограничений электронный аналог реализуется при помощи формы вида “master-detail” (см. ниже). Интерфейс программы должен быть согласован с преподавателем. Программа должна обеспечивать**:**

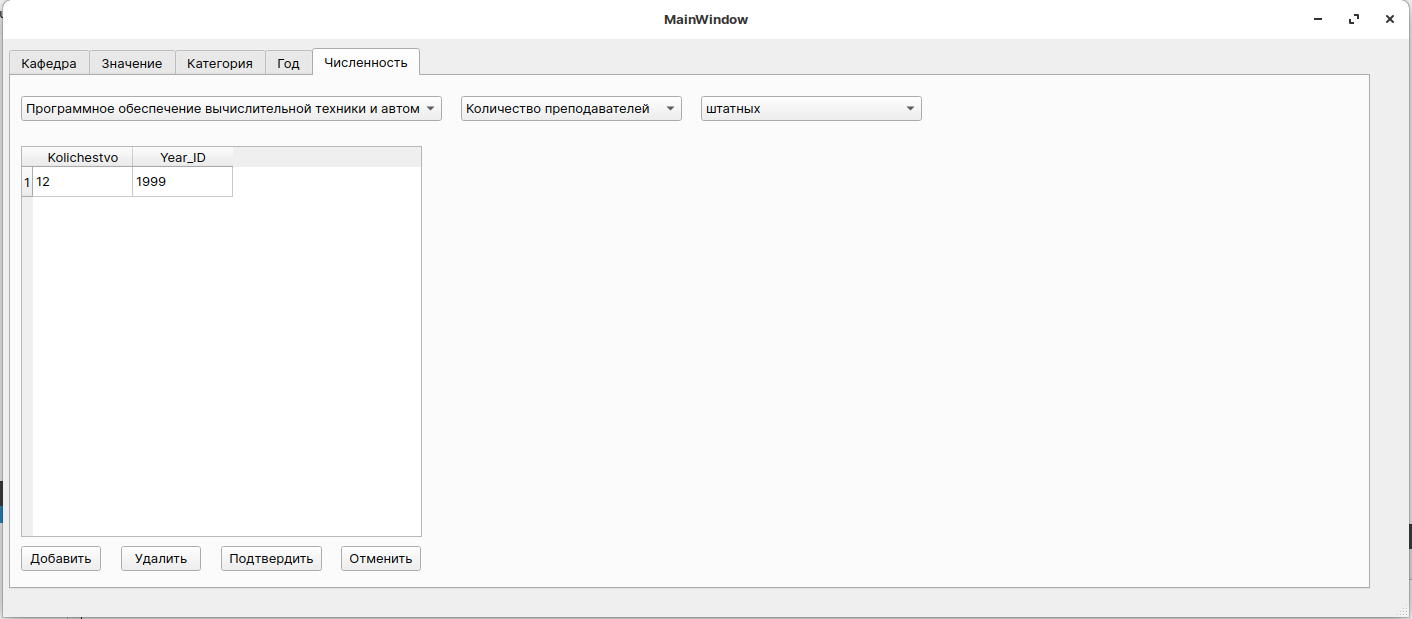
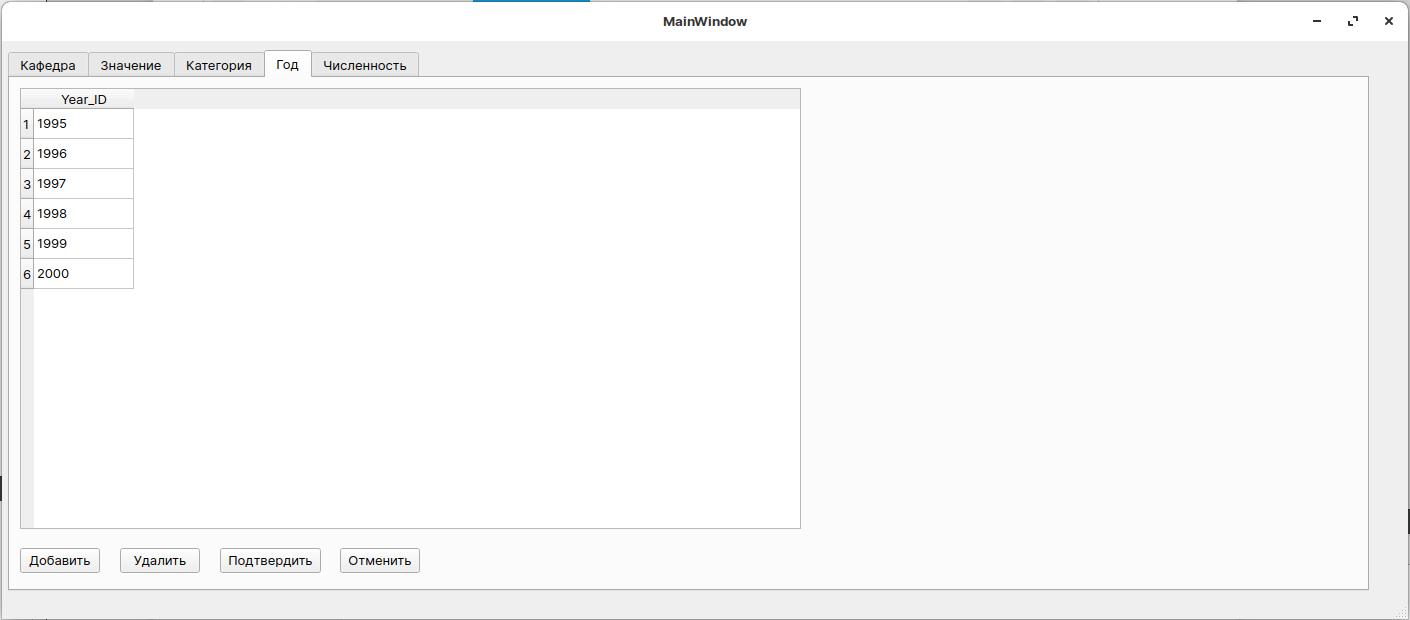
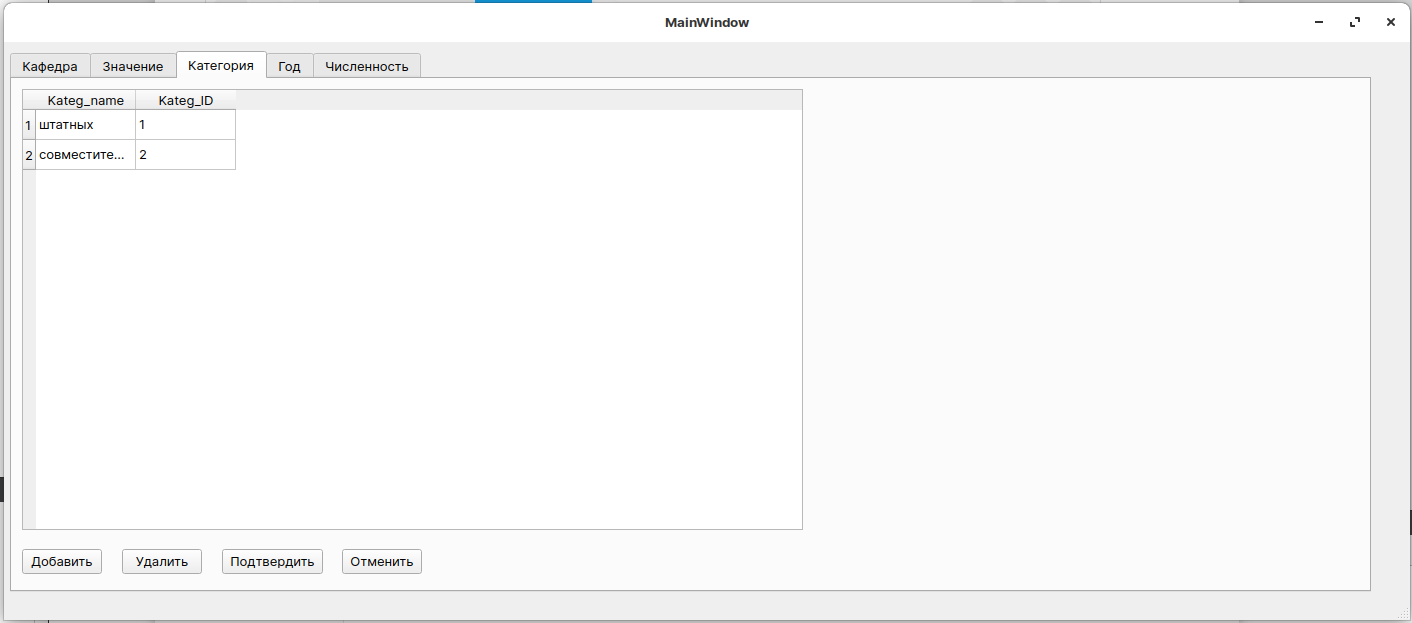
* работу с электронным аналогом документа. Форма должна быть максимально адаптирована под пользователя;
* работу со справочными таблицами. При редактировании справочных таблиц искусственные кодификаторы должны быть скрыты от пользователя;
* наличие формы вида “master-detail”;
* обеспечить контроль ограничения целостности по существованию с обеспечением возможности каскадного удаления с выдачей предупреждающего сообщения пользователю.

По результатам работы необходимо продемонстрировать возможность корректного ведения локальной базы данных с использованием созданной в работе системы экранных форм.

**Скриншоты программы**







**Код программы**

int MainWindow::**initDB**(){

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

QString filename ="/home/orangenal/Документы/labs/5\_semestr/DB's/lab2/Lab2/base3-02.db";

db.setDatabaseName(filename);

if (QFileInfo::exists(filename)) {

db.open(); db.exec("pragma foreign\_keys=on"); } else { QMessageBox::critical(this, "Error", "Can not open database"); return 1; }

ui->KafedraView->setWindowTitle("Kafedra"); ui->KafedraView->show(); Kafedra = new QSqlTableModel(0, db); Kafedra->*setTable*("Kafedra"); Kafedra->*select*(); ui->KafedraView->*setModel*(*Kafedra*); Kafedra->*setEditStrategy*(QSqlTableModel::OnManualSubmit);

ui->ValueView->setWindowTitle("Value"); ui->ValueView->show(); Value = new QSqlTableModel(0, db); Value->*setTable*("Value"); Value->*select*(); ui->ValueView->*setModel*(*Value*); Value->*setEditStrategy*(QSqlTableModel::OnManualSubmit);

ui->KategoriaView->setWindowTitle("Kategoria"); ui->KategoriaView->show(); Kategoria = new QSqlTableModel(0, db); Kategoria->*setTable*("Kategoria"); Kategoria->*select*(); ui->KategoriaView->*setModel*(*Kategoria*); Kategoria->*setEditStrategy*(QSqlTableModel::OnManualSubmit);

ui->YearView->setWindowTitle("Year"); ui->YearView->show(); \_Year = new QSqlTableModel(0, db); \_Year->*setTable*("\_Year"); \_Year->*select*(); ui->YearView->*setModel*(*\_Year*); \_Year->*setEditStrategy*(QSqlTableModel::OnManualSubmit);

ui->ChislennostView->setWindowTitle("Chislennost"); ui->ChislennostView->show(); Chislennost = new QSqlTableModel(0, db); Chislennost->*setTable*("Chislennost"); Chislennost->*select*(); ui->ChislennostView->*setModel*(*Chislennost*); Chislennost->*setEditStrategy*(QSqlTableModel::OnManualSubmit);

Kafedra\_SUB = new QSqlQueryModel(0); ui->KafedraRefView->*setModel*(*Kafedra\_SUB*);

Value\_SUB = new QSqlQueryModel(0); ui->ValueRefView->*setModel*(*Value\_SUB*);

Kategoria\_SUB = new QSqlQueryModel(0); ui->KategoriaRefView->*setModel*(*Kategoria\_SUB*);

Year\_SUB = new QSqlQueryModel(0); ui->YearRefView->*setModel*(*Year\_SUB*);

UpdateDB();

return 0;}

void MainWindow::**UpdateDB**(){

Kafedra\_SUB->setQuery("SELECT \* FROM Kafedra"); Value\_SUB->setQuery("SELECT \* FROM Value"); Kategoria\_SUB->setQuery("SELECT \* FROM Kategoria"); Year\_SUB->setQuery("SELECT \* FROM \_Year");

}