Министерство образования и науки Российской Федерации

Санкт-Петербургский политехнический университет Петра Великого

Институт компьютерных наук и технологий

**Системы Управления базами данных**

**Курсовая работа: Графический Интерфейс Базы данных PostgresSql**

**отчет: курсора**

|  |  |
| --- | --- |
| Выполнили  студент гр.в3530904/70321 | А. М. Ильинский |
| Преподаватель В.В. Амосов | И.В.Прокофьев |

Санкт-Петербург

2019

Приложение для редактирование журнала и справочника таксопарка. Данное приложение было сделано средствами QT:QTcreator, языком программирования - с++. Расскажем, как был написан код для данного приложения.

Приложение состоит из четырех разделов файлов: это \*.pro файл, заголовочные файлы \*.h , основные файлы \*.cpp и файлы дизайна \*ui.

Для редактора кода файла нам понадобится только первые три раздела. В файле \*.ui строится дизайн из виджетов (кнопок, окон для заполнения, окон выбора и т.д.)

Мы пишем файл с двумя модулями – это sql и модуля печати базы данных в pdf (printsupport) , вписать эти модули нужно в файл \*.pro.

.

Далее зайдем в \*h файлы пропишем переменные QSqlDatabase db;(для создание переменной базы данных) и QSqlRelationalTableModel\* model; для построение таблиц

Само приложение состоит из двух курсоров , курсоры ранее созданной базой даных в postgres sql. (pg admin4) Которы представлен ниже:

Таблица авто

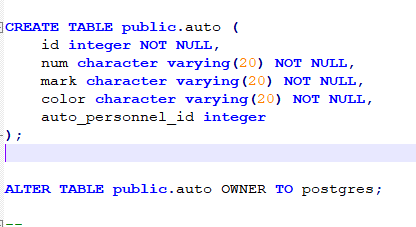


Таблица персонал

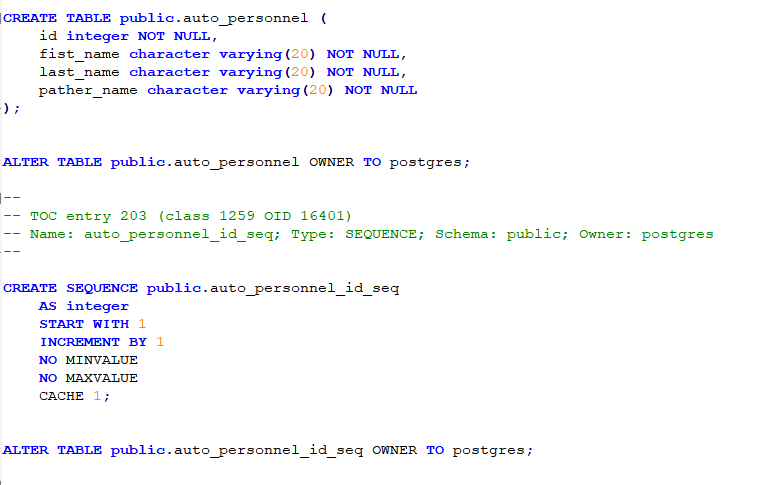


Таблица журнал

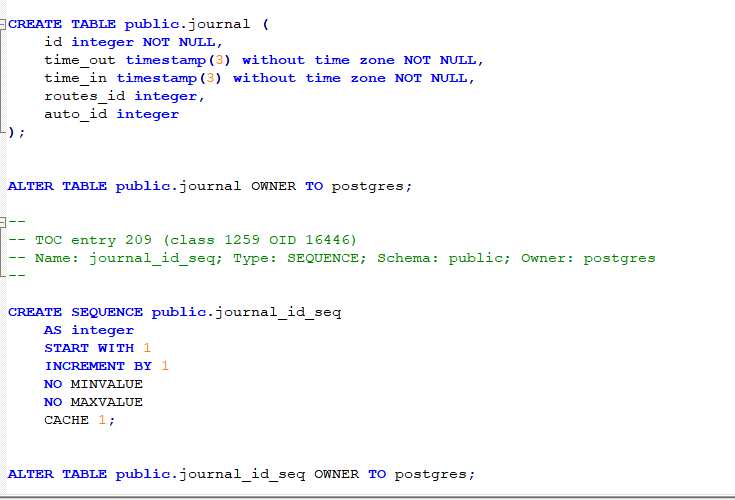
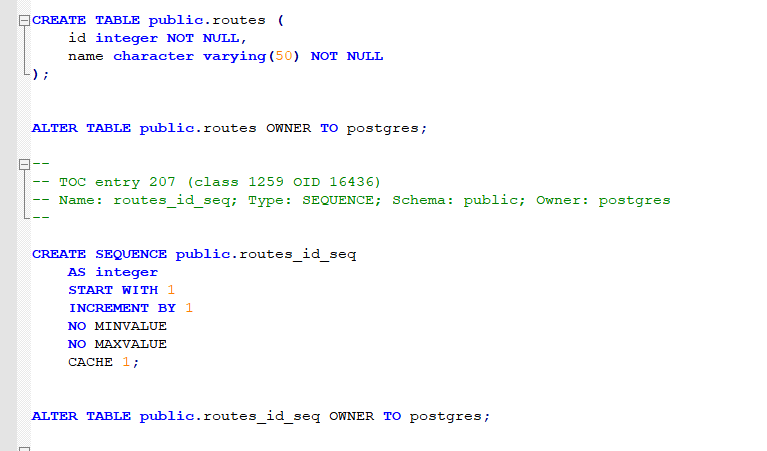
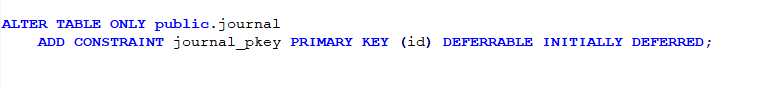


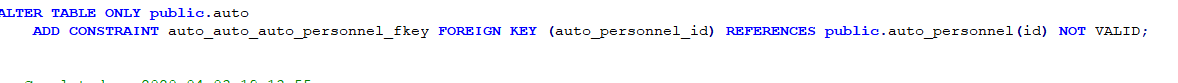
Таблица маршрутов



Пример присвоения ключа id



Пример присвоения связанного ключа



Дальше идет создание второго курсора, кода в самом QT

Создадим нашу базу

void MainWindow::**on\_avtor\_2\_clicked**()//при нажатии на кнопку происходит действие

{

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

db.setHostName("127.0.0.1");// серввер

db.setDatabaseName("postgres");// база

db.setUserName(""+ui->comboBox->currentText());логин выбирает пользователь из предложенных

db.setPassword(""+ui->parol->text());//пароль пользователь должен вести сам

if (!db.open()){

QMessageBox::critical(this, "ошибка", "введите правильно пользователь и пароль");

}

else {

hide();//закрыть окно

qDebug() << "Success!";

wjournal\* window = new wjournal(this);

window->show();// в случае успешного ввда откроется окно с самой главной таблицы журнал

}

}

Следующий курсор будет построение выбора одной из таблицы (все остальные таблицы были построены аналогично

void wjournal::**on\_vstavkadata\_clicked**()

{

jmodel = new QSqlRelationalTableModel ();

jmodel ->setJoinMode(QSqlRelationalTableModel::LeftJoin);// включить пкстые ячейки

jmodel -> *setEditStrategy*(QSqlTableModel::OnManualSubmit);

jmodel -> *setTable*("journal");// select \* from journal

jmodel -> *setRelation*(jmodel->fieldIndex("routes\_id"),

QSqlRelation("routes","id", "name"));

jmodel -> *setRelation*(jmodel->fieldIndex("auto\_id"),

QSqlRelation("auto","id", "num"));// связь между таблиц и взятие столбцов наименование улиц и номеров машин

jmodel -> *select*();

ui->journal->*setModel*(jmodel);

ui -> journal-> setItemDelegate(new QSqlRelationalDelegate(ui->journal));

}//при нажатии на кнопку вводит select \* from journal, так же нам нужна здесm сязь наименований улиц и номеров машин

void wjournal::**on\_journal\_clicked**(const QModelIndex &index)

{

int temp\_nom = ui->journal->model()->*data*(ui->journal->model()->*index*(index.row(),0)).toInt();

ui->jid->setText(QString::number(temp\_nom));

QString temp\_nom2 = ui->journal->model()->*data*(ui->journal->model()->*index*(index.row(),1)).toString();

ui->time\_out->setText(temp\_nom2);

QString temp\_nom3 = ui->journal->model()->*data*(ui->journal->model()->*index*(index.row(),2)).toString();

ui->time\_in->setText(temp\_nom3);

QString temp\_nom4 = ui->journal->model()->*data*(ui->journal->model()->*index*(index.row(),3)).toString();

ui->routes\_id->setText(temp\_nom4);

QString temp\_nom5 = ui->journal->model()->*data*(ui->journal->model()->*index*(index.row(),4)).toString();

ui->auto\_id->setText(temp\_nom5); // при нажатие на таблицу происходит выборка из базы данных и отображение их в соотвeствующих окнах

}

void wjournal::**on\_addrowjournal\_clicked**() // добаление строки

{

qDebug() << "inserting row" << jmodel -> insertRow(jmodel ->*rowCount*());

}

void wjournal::**on\_resettingcelljournal\_clicked**() // сброс ячейки

{

jmodel->revertAll();

}

void wjournal::**on\_resettingrowjournal\_clicked**()// сброс строки

{

jmodel ->*select*();

}

void wjournal::**on\_potvergeniejournal\_clicked**()// подтверждение действие добавление или удаления

{

jmodel -> submitAll();

}

void wjournal::**on\_deletejournal\_clicked**() .// удаление строки

{

int selectedRow = ui->journal->currentIndex().row();

if (selectedRow >=0){

qDebug() << "deleting row:" << jmodel->removeRow(selectedRow);

}

else{

QMessageBox::critical(this, "ошибка", "строки не помечены");

qDebug() << "no row selected";

}

}

void wjournal::**on\_printer\_triggered**() //печать таблицы

{

QString str;

str.append("<html><head></head><body><center>"+QString("отчет таблица журнал маршрутов"));

str.append("<table border=1><tr>");

str.append("<td>"+QString("номер")+"</td>");

str.append("<td>"+QString("отпрака")+"</td>");

str.append("<td>"+QString("прибытие")+"</td>");

str.append("<td>"+QString("улица")+"</td>");

str.append("<td>"+QString("номер машины")+"</td></tr>");

QSqlQuery\* query = new QSqlQuery();

query->exec("select j.id, j.time\_out, j.time\_in, r.name, a.num from journal j join routes r on r.id = j.routes\_id join auto a on a.id =j.auto\_id order by id asc"); // зададим запрос для распечатывание таблицы, привязав наименование улиц и наименование номеров машин, также отсортируем это все по id

while(query->next())

{

str.append("<tr><td>");

str.append(query->value(0).toString());

str.append("</td><td>");

str.append(query->value(1).toString());

str.append("</td><td>");

str.append(query->value(2).toString());

str.append("</td><td>");

str.append(query->value(3).toString());

str.append("</td><td>");

str.append(query->value(4).toString());

str.append("</td></tr>");

}

str.append("</table></center></body></html>");

QPrinter printer;

printer.setOrientation(QPrinter::Portrait);

printer.setOutputFormat(QPrinter::PdfFormat);

printer.setPaperSize(QPrinter::A4);// построение отчета в pdf

QString path = QFileDialog::getSaveFileName(NULL,"соханить в pdf", "отчет журнал маршрутов", "PDF(\*.pdf)");

if(path.isEmpty())return;//его сохраннение

printer.setOutputFileName(path);

QTextDocument doc;

doc.setHtml(str);

doc.print(&printer);

}

void wjournal::**on\_personal\_triggered**()// пример для перехода в другую таблицу

{

hide();

wautopersonal\* window = new wautopersonal(this);

window->show();// создаем и показываем новое окно

}

Вывод

На выходе мы получаем графическое приложение состоящие из четырех таблиц. Это приложение мы можем редактировать и просто просматривать данные, а также по окончании, работ распечатать данные из таблицы в формате pdf.