Министерство образования Республики Беларусь

Учреждение образования

БЕЛОРУССКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ

ИНФОРМАТИКИ И РАДИОЭЛЕКТРОНИКИ

Факультет компьютерного проектирования

Кафедра проектирования информационно-компьютерных систем

Отчет

по лабораторной работе №4

на тему:

**Разработка информационной модели предметной области. Модели представления программного средства и их описание. Постановка задачи**

Проверил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ФИО преподавателя

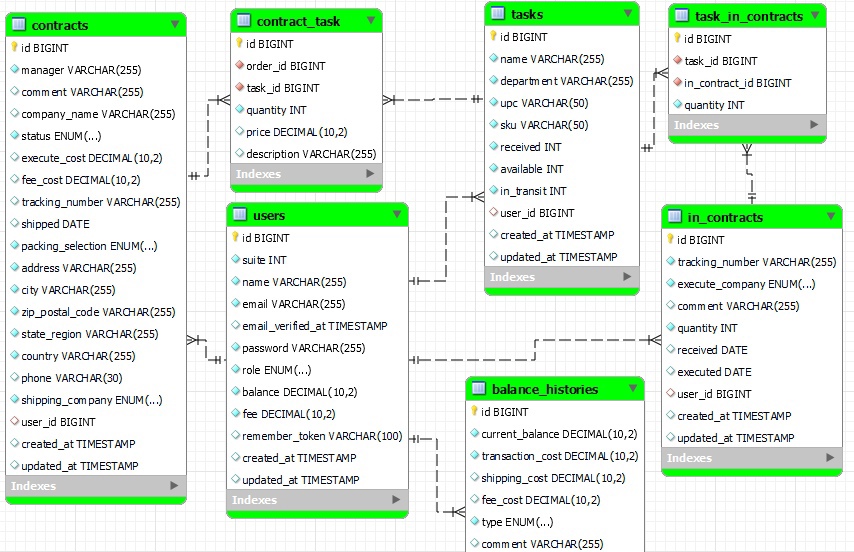
(подпись)

Выполнил: \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ ФИО студента

(подпись)

Минск, 2021

Разработаем информационную модель предметной области:



Разработка таблиц была выполнена с помощью программной среды MySQL Workbench.

Рассмотрим данные таблицы базы данных текущего проекта.

Таблица «истории баланса» содержит данные об истории баланса пользователей. Поля таблицы истории баланса:

1 id – первичный ключ таблицы идентифицирующий запись. Это поле заполняется автоматически значениями типа BIGINT.

2 current\_balance (текущий баланс) – в поле хранится информация о текущем балансе пользователя. Это поле заполняется значениями типа DECIMAL (10, 2).

3 transaction\_cost (сумма транзакции) – в поле хранится информация о сумме транзакции. Это поле заполняется значениями типа DECIMAL (10, 2).

4 shipping\_cost (стоимость доставки) – в поле хранится информация о стоимости доставки товаров. Это поле заполняется значениями типа DECIMAL (10, 2).

5 fee\_cost (стоимость комиссии) – в поле хранится информация о стоимости комиссии за выполнение операций. Это поле заполняется значениями типа DECIMAL (10, 2).

6 type (тип) – поле, указывающее тип баланса пользователя. Данное поле заполняется значениями типа ENUM (…).

7 comment (комментарий) – поле заполняется комментарием к операции над балансом. Данное поле заполняется значениями типа VARCHAR (255).

Таблица «пользователи» содержит данные об учетной записи пользователя системы. Поля таблицы пользователи:

1 id – первичный ключ таблицы идентифицирующий запись. Это поле заполняется автоматически значениями типа BIGINT.

2 suite (уникальный номер) – поле, указывающее уникальный номер пользователя. Данное поле заполняется значениями типа INT.

3 name (имя пользователя) – в поле занесены данные фамилии, имени и отчества пользователя. Это поле заполняется значениями типа VARCHAR (255).

4 email (электронная почта). В поле занесены данные электронной почты пользователя, которые служат для авторизации. Это поле заполняется значениями типа VARCHAR (255). Это поле уникальное, так как нельзя записать в таблицу пользователей с одинаковыми электронными почтами.

5 password (пароль) – хранится пароль пользователя. Поле заполняется значениями типа VARCHAR (255).

6 role (роль) – поле, указывающее роль пользователя (user или admin). Данное поле заполняется значениями типа ENUM (…).

7 balance (баланс) – в поле хранится информация о балансе данного пользователя. Это поле заполняется значениями типа DECIMAL (10, 2).

8 fee (ставка комиссии) – в поле хранится информация о ставке комиссии за операции. Это поле заполняется значениями типа DECIMAL (10, 2).

Таблица «в контракте» содержит данные о контрактах пользователя. Поля таблицы в контракте:

1 id – первичный ключ таблицы идентифицирующий запись. Это поле заполняется автоматически значениями типа BIGINT.

2 tracking\_number (код отслеживания) – в поле занесены данные о коде отслеживания. Это поле заполняется значениями типа VARCHAR (255).

3 execute\_company (компания-исполнитель) – поле, указывающее название компании исполнителя контракта. Данное поле заполняется значениями типа ENUM (…).

4 comment (комментарий) – поле заполняется комментарием к доставке товаров пользователю. Данное поле заполняется значениями типа VARCHAR (255).

5 quantity (количество) – поле, указывающее количество товара для доставки. Данное поле заполняется значениями типа INT.

6 received (полученный товар) – поле заполняется информацией о дате получения товара. Данное поле заполняется значениями типа DATE.

7 executed (исполненный контракт) – поле заполняется информацией о дате исполнения контракта. Данное поле заполняется значениями типа DATE.

Таблица задачи в контракте содержит информацию о том, какой контракт был заключен и на какое количество товаров. Данная таблица связывает две таблицы: в контракте и задачи. Поля таблицы задачи в контракте:

1 id – первичный ключ таблицы идентифицирующий запись. Это поле заполняется автоматически значениями типа BIGINT.

2 quantity (количество) – поле, указывающее количество товара в контракте. Данное поле заполняется значениями типа INT.

Таблица «задачи» содержит данные о задачах. Поля таблицы задачи:

1 id – первичный ключ таблицы идентифицирующий запись. Это поле заполняется автоматически значениями типа BIGINT.

2 name (название) – поле заполняется названием задачи, которая прописывается в контракте. Данное поле заполняется значениями типа VARCHAR (255).

3 department (отдел) – поле заполняется названием отдела, которому поручено выполнение поставленной задачи. Данное поле заполняется значениями типа VARCHAR (255).

4 upc (универсальный товарный код) – поле заполняется универсальным товарным кодом. Данное поле заполняется значениями типа VARCHAR (50).

5 sku (единица учета запасов) – поле заполняется единицей учета запасов. Данное поле заполняется значениями типа VARCHAR (50).

6 received (зарезервировано) – поле, указывающее на количество зарезервированного товара. Данное поле заполняется значениями типа INT.

7 available (доступно) – поле, указывающее на количество доступного товара. Данное поле заполняется значениями типа INT.

8 in\_transit (в пути) – поле, указывающее на количество товара, находящегося в пути. Данное поле заполняется значениями типа INT.

Таблица «задачи контракта» содержит информацию о том, к какому контракту относится какая задача. Данная таблица связывает две таблицы: контракты и задачи. Поля таблицы задачи контракта:

1 id – первичный ключ таблицы идентифицирующий запись. Это поле заполняется автоматически значениями типа BIGINT.

2 quantity (количество) – поле, указывающее на количество задач в контракте. Данное поле заполняется значениями типа INT.

3 price (цена) – в поле хранится информация о сумме, прописанной в контракте. Это поле заполняется значениями типа DECIMAL (10, 2).

4 description (описание) – поле заполняется описанием к каждой задаче контракта. Данное поле заполняется значениями типа VARCHAR (255).

Таблица «контракты» содержит данные о контрактах пользователей. Поля таблицы контракты:

1 id – первичный ключ таблицы идентифицирующий запись. Это поле заполняется автоматически значениями типа BIGINT.

2 manager (менеджер) – поле заполняется именем менеджера, который будет заключать контракт. Данное поле заполняется значениями типа VARCHAR (255).

3 comment (комментарий) – поле заполняется комментарием к контракту. Данное поле заполняется значениями типа VARCHAR (255).

4 company\_name (комания) – поле заполняется именем компании. Данное поле заполняется значениями типа VARCHAR (255).

5 status (статус) – поле, указывающее какой статус у контракта на данный момент. Данное поле заполняется значениями типа ENUM (…).

6 execute\_cost (стоимость исполнения) – в поле хранится информация о том, сколько будет стоить исполнение контракта. Это поле заполняется значениями типа DECIMAL (10, 2).

7 fee\_cost (стоимость комиссии) – в поле хранится информация о стоимости комиссии за выполнение операции над контрактом. Это поле заполняется значениями типа DECIMAL (10, 2).

8 tracking\_number (код отслеживания) – в поле занесены данные о коде отслеживания. Это поле заполняется значениями типа VARCHAR (255).

9 shipped (доставлено) – поле заполняется информацией о дате доставления контракта. Данное поле заполняется значениями типа DATE.

10 packing\_selection (тип упаковки) – поле, указывающее в какую упаковку завернуть контракт. Данное поле заполняется значениями типа ENUM (…).

11 address (адрес) – в поле занесены данные об адресе, где был заключен контракт. Это поле заполняется значениями типа VARCHAR (255).

12 city (город) – в поле занесены данные о городе, где был заключен контракт. Это поле заполняется значениями типа VARCHAR (255).

13 zip\_postal\_code (почтовый индекс) – в поле занесены данные о почтовом индексе. Это поле заполняется значениями типа VARCHAR (255).

14 state\_region (область) – в поле занесены данные об области, где был заключен контракт. Это поле заполняется значениями типа VARCHAR (255).

15 country (страна) – в поле занесены данные о стране, где был заключен контракт. Это поле заполняется значениями типа VARCHAR (255).

16 phone (номер телефона) – в поле занесены данные о номере телефона. Это поле заполняется значениями типа VARCHAR (30).

17 shipping\_company (транспортная компания) – поле, указывающее название транспортной компании, которая доставляет контракты. Данное поле заполняется значениями типа ENUM (…).

Перейдем к рассмотрению моделей представления программного средства и их описанию.

В данной лабораторной работе будут построены такие диаграммы как диаграмма состояний и диаграмма последовательности.

Диаграмма состояний предназначена для отображения состояний объектов системы, имеющих сложную модель поведения. Она показывает пространство состояний системы или ее элементов, события, которые влекут переход из одного состояния в другое, действия, которые происходят при изменении состояния. Объекты меняют своё состояние в ответ на происходящие события и стечением времени. Диаграмма состояний представляет состояния объекта и переходы между ними, а также начальное и конечное состояние объекта.

Диаграмма состояний данной системы:

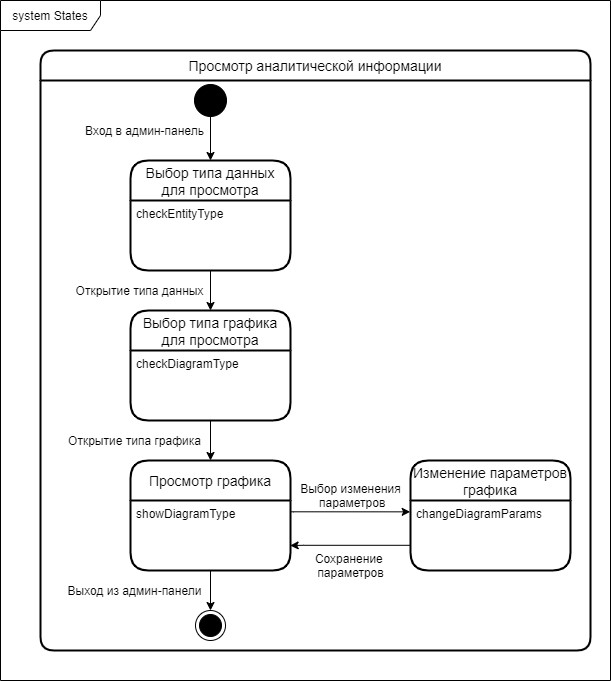


Диаграмма состояний описывает один конкретный набор требований к состояниям системе. В данном случае описывает требования к состояниям в рамках просмотра графической аналитической информации о сущностях системы.

Алгоритм изменения состояний системы, проходящих в рамках основного состояния «Просмотр аналитической информации»:

1 «Выбор типа данных для просмотра». Начинается с сразу после начального состояния через переход «Вход в админ-панель». Означает, что в данный момент пользователь открыл панель управления сайта и выбирает, для какой сущности (договора, заказы и так далее) смотреть аналитическую информацию.

2 «Выбор типа графика для просмотра». Начинается после предыдущего состояния через переход «Открытие типа данных». Означает, что пользователь выбрал и открыл нужную сущность для аналитики и выбирает какой тип графика смотреть.

3 «Просмотр информации». Начинается после предыдущего события с перехода «Открытие типа графика». Выполняется выбор и открытие нужного графика пользователем системы, пользователь просматривает выбранный график. После данного состояния, если пользователь выбирает закрытие системы, то через переход «Выход из админ-панели» происходит переход к конечному состоянию.

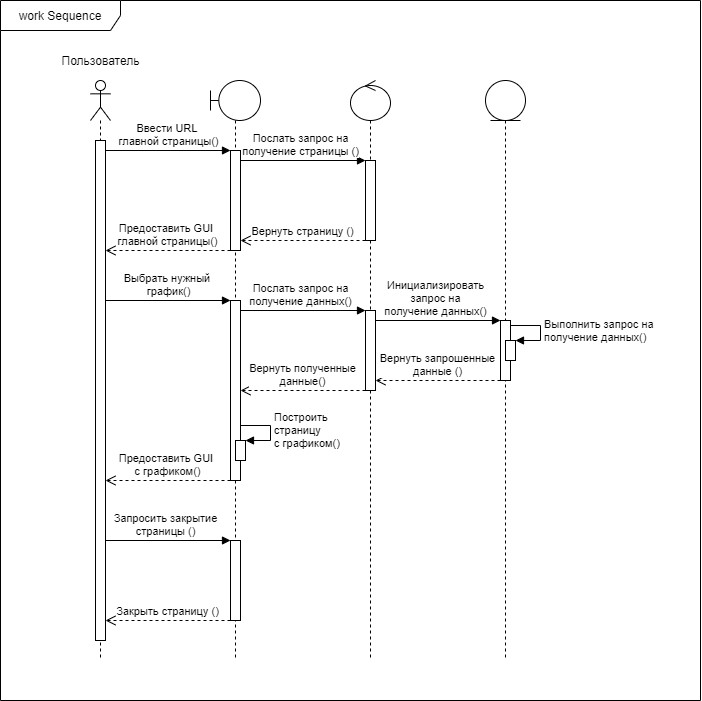
4 «Изменение параметров графика». Если пользователь в предыдущем состоянии выбрал изменение параметров графика (переход «Выбор изменения параметров»), происходит переход к данному состоянию. В рамках состояния пользователь выбирает различные параметры графика. Переходит обратно к предыдущему состоянию через переход «Сохранение параметров».

Через диаграмму состояния отчётливо видны требования к системе с точки зрения пользователя системы для просмотра графической информации. Как итог, реализация диаграмма состояний в системе описывает требования.

Диаграмма последовательности – это диаграмма, где для некоторого набора объектов на единой временной оси показан жизненный цикл какого-либо определённого объекта.

В рамках курсового проекта диаграмма последовательности отражает алгоритм просмотра графика с аналитической информацией администратором в системе.

Диаграмма последовательности:



Алгоритм просмотра аналитической информации в системе имеет следующий вид:

1. пользователь вводит адрес главной страницы;
2. клиентское приложение по адресу посылает запрос на сервер;
3. сервер возвращает страницу клиентскому приложению, адрес которой был введён;
4. клиентское приложение строит страницу и отображает её;
5. пользователь выбирает нужный график;
6. клиентское приложение посылает запрос о получении данных для графика;
7. сервер запрашивает данные из базы данных из сервера MySQL;
8. сервер MySQL выбирает данные;
9. данные возвращаются серверу приложения;
10. сервер приложения возвращает данные клиентскому приложению;
11. клиентское приложение строит график по полученным данным;
12. построенный график отображается для пользователя системы;
13. пользователь системы после просмотра графика нажимает кнопку закрытия страницы;
14. клиентское приложение закрывает страницу.

Теперь перейдем к постановке задачи.

Основной задачей является создание программного средства автоматизации договорной деятельности торгового предприятия.

Что касается подзадач курсового проекта, то выделяют следующие:

– создать веб-приложение, которое будет хранить информацию о договорах торгового предприятия;

– создать максимально легкий, понятный и удобный пользовательский интерфейс;

– реализовать авторизацию пользователей для предоставления им разных прав;

– организовать надежный обмен данных между клиентом и сервером;

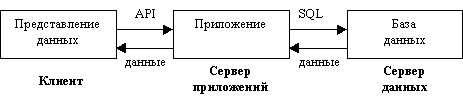
– обеспечить хранение данных средствами производительной и надежной СУБД.

Для реализации поставленных задач нужно определиться с какими инструментами будет производиться их выполнение. Наиболее эффективную работу с централизованной БД обеспечивает архитектура клиент/сервер. В отличие от системы удаленной обработки, в которой имеется только один компьютер, клиент/серверная система состоит из множества компьютеров, объединенных в сеть. Компьютеры, называемые клиентами, занимаются обработкой прикладных программ. Компьютеры, называемые серверами, занимаются обработкой БД.

Тип компьютеров, используемых в качестве клиентов может быть разным, это могут быть большие ЭВМ или микрокомпьютеры. Однако, как правило, функции клиентов выполняют почти всегда ПК. В роли сервера может выступать компьютер любого типа, но по экономическим причинам функции сервера чаще всего также выполняют ПК, но имеющие более высокую производительность.

На сервере сети размещается БД и устанавливается мощная серверная СУБД – сервер баз данных. Сервер БД – это программный компонент, обеспечивающий хранение больших объемов информации, ее обработку и представление ее пользователям в сетевом режиме.

На компьютере-клиенте приложение-клиент формирует запрос к БД. Серверная СУБД обеспечивает интерпретацию запроса, его выполнение, формирование результата запроса и пересылку его по сети на клиентский компьютер. Клиентское приложение интерпретирует его необходимым образом и представляет пользователю. Клиентское приложение может также посылать запрос на обновление БД и серверная СУБД вносит необходимые изменения в БД. Схема архитектуры клиент/сервер:



В архитектуре клиент/сервер функции клиентского приложения и серверной СУБД разделены.

Функции клиентского приложения разбиваются на следующие группы:

1 Ввод-вывод данных (презентационная логика) – это часть кода клиентского приложения, которая определяет, что пользователь видит на экране, когда работает с приложением.

2 Бизнес-логика – это часть кода клиентского приложения, которая определяет алгоритм решения конкретных задач приложения.

3 Обработка данных внутри приложения (логика базы данных) – это часть кода клиентского приложения, которая связывает данные сервера с приложением. Для этой связи используется процедурный язык запросов SQL, с помощью которого осуществляется выборка и модификация данных в серверных СУБД.

Сервер баз данных в общем случае осуществляет целый комплекс действий по управлению данными. Основными среди них являются следующие:

– выполнение пользовательских запросов на выбор и модификацию данных и метаданных, получаемых от клиентских приложений, функционирующих на ПК локальной сети;

– хранение и резервное копирование данных;

– поддержка ссылочной целостности данных согласно определенным в БД правилам;

– обеспечение авторизованного доступа к данным на основе проверки прав и привилегий пользователя.

Все серверные СУБД имеют клиентскую часть, которая обращается к БД посредством СУБД. Между клиентским приложением и СУБД не существует прямой связи и дополнительно встраиваются программные модули, позволяющие клиентскому приложению получать доступ к БД, создаваемым с помощью разных СУБД. Такие модули называются механизмами доступа к данным.

Следующей стадией разработки программного средства будет проектирование. На этом этапе для упрощения визуализации процесса проектирования используются так называемые нотации – схематическое выражение характеристик разрабатываемого программного средства.

После того, как программное средство будет спроектировано, произойдет переход к следующей стадии жизненного цикла – непосредственно разработке. На этом этапе будет разрабатываться программное средство в соответствии с ранее определенными требованиями.

После того, как этап разработки программного средства будет завершен, необходимо будет протестировать полученное программное средство. На этом этапе необходимо будет найти дефекты в программном средстве и сравнить описанное в требованиях поведение системы с реальным.

При обнаружении дефекта, необходимо будет его исправить, после чего повторить тестирование и на этот раз убедиться, что проблема исправлена и само исправление не стало причиной появления новых дефектов в системе.

После того, как тестирование программного средства пройдет успешно, необходимо будет пройти этап документации. Он будет включать в себя составление руководства пользователя, а также презентацию программного средства.

Выполнение всех описанных выше стадий жизненного цикла программного средства является обязательным при проектировании, разработке, тестировании и запуске программного средства.