Краевое государственное бюджетное профессиональное образовательное учреждение «Алтайский архитектурно-строительный колледж»

**09.02.03 «Программирование в компьютерных системах»**

ОТЧЁТ

по учебной практике

период прохождения практики «16» июня 2020г. – «20» июня 2020г.

|  |
| --- |
|  |
| **УП.02.02 Разработка, администрирование и защита баз данных** |
| **ПМ.02 Разработка, администрирование и защита баз данных** |
|  |

|  |  |  |
| --- | --- | --- |
| Выполнил студент группы ПСК-71 |  | / Кузнецова М.Е./ |
| Руководитель практики от учебного заведения |  | / Беляева М.А./ |

Оценка\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Барнаул 2020

**СОДЕРЖАНИЕ**

[ВВЕДЕНИЕ 3](#_Toc43498451)

[1. Правила безопасности при работе с компьютером 4](#_Toc43498452)

[2. Система управления базами данных 4](#_Toc43498453)

[2.1. Структура и функции системы управления базами данных 4](#_Toc43498454)

[2.2. Классификация СУБД по способу доступа к базе данных 7](#_Toc43498455)

[2.3. Язык SQL в системах управления базами данных 8](#_Toc43498456)

[2.4. СУБД Microsoft SQL Server 10](#_Toc43498457)

[3. Объектно-ориентированный язык C# 13](#_Toc43498458)

[3.1. История возникновения 13](#_Toc43498459)

[3.2. Общее описание языка 14](#_Toc43498460)

[3.3. Особенности языка С# 16](#_Toc43498461)

[3.4. Сфера применения C# 17](#_Toc43498462)

[4. Разработка проекта 18](#_Toc43498463)

[4.1. Порядок создания базы данных 18](#_Toc43498464)

[ЗАКЛЮЧЕНИЕ 21](#_Toc43498465)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ 23](#_Toc43498466)

# ВВЕДЕНИЕ

Учебная практика – это элемент образовательного процесса, основная цель которого интегрировать теоретические знания и практические умения. Достигается посредством выполнения ряда более частных задач: закрепления знаний в ходе их непосредственного применения; выработки.

В ходе прохождения учебной практики, разрабатывалась база данных. Результатом освоения учебной практики является овладение видом профессиональной деятельности.

|  |  |
| --- | --- |
| **Код** | **Наименование результата обучения** |
| ПК 2.1 | Разрабатывать объекты баз данных. |
| ПК 2.2 | Реализовывать базу данных в конкретной системе управления базами данных (далее – СУБД). |
| ПК 2.3 | Решать вопросы администрирования баз данных. |
| ПК 2.4 | Реализовывать методы и технологии защиты информации в базах данных. |

Цель учебной практики: получение навыков при работе с базами данных, с использованием среды Microsoft SQL Server Management Studio и языком программирования С# в интегрированной среде разработки Microsoft Visual Studio.

Задачи:

1. Приобретение практических навыков организации взаимодействия приложения с базой данных.
2. Отработка технологии связывания таблиц базы данных.
3. Закрепление навыков организации взаимодействия форм.
4. Разработка модели базы данных.
5. Отработка технологии дополнения и коррекции записей в таблице.

# Правила безопасности при работе с компьютером

Компьютер – это сложное устройство, работающее под напряжением, поэтому на него тоже распространяются техника безопасности к электрическим устройствам.

* Применимо к компьютеру соблюдайте основные правила:

Не работать с компьютером при наличии внешних повреждений корпуса или изоляции силовых кабелей. В этом случае требуется замена кабелей или обращение в сервисный центр.

* Не класть на корпус [системного блока](https://www.pc-school.ru/sistemnyj-blok/) и не хранить на нем разные предметы, особенно тяжелые, т.к. в этом случае может возникнуть вибрация, которая может вызвать нарушения работы компьютера.
* Не рекомендуется включать компьютер в розетки без заземления. Розетки и вилки должны быть цельными, без повреждений.
* Не включать компьютер в помещении с высокой влажностью.
* Не оставлять работающий ПК без присмотра длительное время.
* Провода и силовые кабеля компьютера должны быть расположены так, чтобы исключить возможность наступить на них или поставить что-то тяжелое.
* Нельзя работать с компьютером при открытом корпусе системного блока.

# Система управления базами данных

# Структура и функции системы управления базами данных

Система управления базами данных (СУБД) — специализированная программа или комплекс программ, предназначенная для организации и ведения базы данных. Она обеспечивает хранение данных и взаимодействие пользователя с БД, позволяя пользователям производить поиск, сортировку и выборку информации в базе данных, а некоторым пользователям – добавлять, удалять и изменять записи в БД.

В зависимости от того, с какой базой данных работает СУБД, она может быть:

* иерархической,
* сетевой,
* реляционной,
* объектно-реляционной,
* объектно-ориентированной.

СУБД обычно содержит следующие компоненты:

* Ядро, которое отвечает за управление данными во внешней и оперативной памяти.
* Процессор языка базы данных, обеспечивающий оптимизацию запросов на извлечение и изменение данных, и создание, как правило, машинно-независимого исполняемого внутреннего кода.
* Подсистему поддержки времени исполнения, которая интерпретирует программы манипуляции данными, создающие пользовательский интерфейс с СУБД.
* Сервисные программы (внешние утилиты), обеспечивающие ряд дополнительных возможностей по обслуживанию информационной системы.

В связи с этим можно сформулировать основную функцию СУБД:

Определение данных - СУБД должна предоставлять средства определения данных в виде исходной формы (схемы данных) и преобразования этих определений в соответствующую объектную форму. То есть СУБД преобразовывает данные в форму, необходимую для хранения их в базе данных.

Манипулирование данными - СУБД должна быть способна обрабатывать запросы пользователя на выборку, изменение или удаление данных, уже существующих в базе, или на добавление в нее новых данных. То есть, СУБД обеспечивает интерфейс между пользователями и базами данных.

Управление хранением данных и доступом к ним - СУБД осуществляет программную поддержку хранения данных в запоминающем устройстве ЭВМ и управляет всеми действиями, производимыми с данными.

Защита и поддержка целостности данных - СУБД должна контролировать пользовательские запросы и определять, кому доступны операции изменения данных, а кому доступны только операции получения данных. Также она следит за целостностью данных, хранящихся в БД. СУБД осуществляет журнализацию изменений, резервное копирование и восстановление базы данных после сбоев.

Таким образом, основная функция системы управления базами данных – осуществление интерфейса пользователя и базы данных. Большинство современных крупных банков данных рассчитаны на работу нескольких пользователей, поэтому СУБД осуществляет разделение времени между пользователями при одновременном их доступе к базе данных, а также разделение полномочий между разными типами пользователей. Например, бухгалтер на предприятии может только получать информацию из банка данных, а главный экономист может вносить изменения в банк данных.

СУБД выполняет эти функции с помощью определенного информационно-логического языка, или языка запросов. В большинстве СУБД для этого используется язык SQL.

СУБД реляционного типа освобождает пользователя от необходимости знать форматы хранения данных, методы доступа и методы управления памятью. Изменение физической структуры базы данных не влияет на работоспособность прикладных программ, работающих с нею.

По степени универсальности различают два класса СУБД:

* системы общего назначения;
* специализированные системы.

СУБД общего назначения не ориентированы на какую-либо предметную область или на информационные потребности какой-либо группы пользователей. Каждая система такого рода реализуется как программный продукт, способный функционировать на некоторой модели компьютеров в определенной операционной системе и поставляется многим пользователям как коммерческое изделие. Такие СУБД обладают средствами настройки на работу с конкретной базой данных. Специализированные СУБД создаются в редких случаях при невозможности или нецелесообразности использования СУБД общего назначения.

# Классификация СУБД по способу доступа к базе данных

По способу доступа к БД системы управления ими подразделяются на три типа:

* Файл-серверные - в файл-серверных СУБД файлы данных располагаются централизованно на файл-сервере. Ядро СУБД располагается на каждом клиентском компьютере. Доступ к данным осуществляется через локальную сеть. Синхронизация чтений и обновлений осуществляется посредством файловых блокировок. Преимуществом этой архитектуры является низкая нагрузка на ЦП сервера, а недостатком — высокая загрузка локальной сети.
* Клиент-серверные - такие СУБД состоят из клиентской части (которая входит в состав прикладной программы) и сервера (см. Клиент-сервер). Клиент-серверные СУБД, в отличие от файл-серверных, обеспечивают разграничение доступа между пользователями и мало загружают сеть и клиентские машины. Сервер является внешней по отношению к клиенту программой, и по надобности его можно заменить другим. Недостаток клиент-серверных СУБД в самом факте существования сервера (что плохо для локальных программ — в них удобнее встраиваемые СУБД) и больших вычислительных ресурсах, потребляемых сервером.

Клиент-серверные СУБД предоставляют больше возможностей для профессиональной работы с данными, поэтому они чаще всего используются в крупных предприятиях и организациях. Они больше всего подходят к крупным информационным системам с одним или несколькими серверами, обладающими большой производительностью. Даже в случае большого количества пользователей, работающих с ними, они не очень сильно загружают сеть.

* Встраиваемые - библиотека, которая позволяет унифицированным образом хранить большие объёмы данных на локальной машине. Доступ к данным может происходить через SQL либо через особые функции СУБД. Встраиваемые СУБД быстрее обычных клиент-серверных и не требуют установки сервера, поэтому востребованы в локальном ПО, которое имеет дело с большими объёмами данных (например, геоинформационные системы).

Таким образом, для использования в крупных организациях, в том числе на промышленных предприятиях, больше подходят клиент-серверные СУБД. Мы рассмотрим особенности таких распространенных СУБД, как Oracle и MS SQL Server.

# Язык SQL в системах управления базами данных

SQL (англ. Structured Query Language — язык структурированных запросов) — универсальный компьютерный язык, применяемый для создания, модификации и управления данными в реляционных базах данных. SQL в его чистом (базовом) виде является информационно-логическим языком, а не языком программирования. Вместе с тем стандарт языка спецификацией SQL/PSM предусматривает возможность его процедурных расширений, с учётом которых язык уже вполне может рассматриваться в качестве языка программирования.

Язык SQL делится на четыре части:

* Операторы определения данных.
* Операторы манипуляции данными.
* Операторы определения доступа к данным.
* Операторы управления транзакциями.

Язык SQL имеет следующие особенности:

Преимущества:

Независимость от конкретной СУБД – большинство текстов SQL-запросов, содержащие DDL и DML, могут быть достаточно легко перенесены из одной СУБД в другую.

Наличие стандартов - наличие стандартов и набора тестов для выявления совместимости и соответствия конкретной реализации SQL общепринятому стандарту только способствует «стабилизации» языка.

Декларативность - с помощью SQL программист описывает только то, какие данные нужно извлечь или модифицировать. То, каким образом это сделать, решает СУБД непосредственно при обработке SQL-запроса.

Недостатки:

Несоответствие реляционной модели данных. В частности они указывают на следующие проблемы SQL:

* Повторяющиеся строки.
* Неопределённые значения (nulls).
* Явное указание порядка колонок слева направо.
* Колонки без имени и дублирующиеся имена колонок.
* Отсутствие поддержки свойства «=».
* Использование указателей.
* Высокая избыточность.

Сложность - хотя SQL и задумывался, как средство работы конечного пользователя, в конце концов он стал настолько сложным, что превратился в инструмент программиста.

Отступления от стандартов - несмотря на наличие международного стандарта ANSI SQL-92, многие компании, занимающиеся разработкой СУБД, вносят изменения в язык SQL, применяемый в разрабатываемой СУБД, тем самым отступая от стандарта.

Несмотря на недостатки, язык SQL вполне пригоден для использования в реляционных СУБД. Обычно СУБД позволяет создавать и редактировать запросы на языке SQL с помощью графической оболочки программы, освобождая пользователя от знания языка SQL. Пользователь может создавать запросы на выборку данных по определенному условию, на проведение расчёта по данным и т. д. Также большинство СУБД позволяют представлять данные в форме отчета, т.е. в форме, пригодной для вывода на печать. Вводить данные в БД пользователи могут с помощью форм. Форма представляет собой определенным образом организованный интерфейс для ввода данных.

Также доступ к данным, хранящимся в БД, может быть организован с помощью страниц доступа к данным. Эти страницы могут храниться на сервере корпоративной сети (Intranet), если сервер имеет поддержку страниц доступа к данным. Доступ к этой странице в данном случае осуществляется с помощью протокола http.

# СУБД Microsoft SQL Server

Одной из самых распространенных СУБД, используемых в крупных организациях, является Microsoft SQL server, СУБД от компании Microsoft. Она использует реляционную модель данных. Microsoft SQL server является клиент-серверной СУБД, что делает ее хорошей для использования в крупных организациях, имеющих мощный сервер. Платформа данных MS SQL Server включает следующие инструменты для работы с базой данных:

* Поддержка структурированных и неструктурированных (XML) данных.
* Replication Services - репликация данных для распределённых и мобильных приложений обработки данных, высокая доступность систем, масштабируемый параллелизм со вторичными хранилищами данных для отчётных решений предприятия и интеграция с разнородными системами, включая существующие базы данных Oracle.
* Notification Services - развитые возможности уведомлений для разработки и внедрения масштабируемых приложений, способных доставлять персонализированные, своевременные обновления информации множеству соединённых и мобильных устройств.
* Integration Services - возможности извлечения, преобразования и загрузки для хранилищ данных и интеграции данных в масштабе предприятия.
* Analysis Services - аналитическая обработка в реальном времени (OLAP) для быстрого, сложного анализа больших и смешанных наборов данных, использующая многомерное хранение.
* Reporting Services - исчерпывающее решение для создания, управления и доставки как традиционных бумажных отчётов, так и интерактивных, основанных на технологии WWW отчётов.
* Инструменты управления - SQL Server включает средства управления для развитого управления и настройки баз данных, также как и тесную интеграцию с такими инструментами, как Microsoft Operations Manager (MOM) и Microsoft Systems Management Server (SMS).
* Инструменты разработки - SQL Server предлагает интегрированные инструменты разработки для ядра базы данных, извлечения, трансформации и загрузки данных, извлечения информации, OLAP и отчётности, которые тесно интегрированы с Microsoft Visual Studio® для предоставления сквозных возможностей разработки приложений. Каждая главная подсистема SQL Server поставляется со своей собственной объектной моделью и набором API для расширения системы данных в любом направлении, которое уникально для вашего бизнеса.

Особенностью данной СУБД являются следующие средства:

SQL Server Management Studio - упрощает управление путём предоставления единой интегрированной консоли управления для мониторинга и управления реляционной базой данных SQL Server, Integration Services, Analysis Services, Reporting Services, Notification Services и SQL Mobile на большом числе распределённых серверов и баз данных.

Администраторы баз данных могут выполнять несколько задач одновременно, включая следующие: создание и выполнение запроса, просмотр серверных объектов, управление объектом, отслеживание активности системы и просмотр оперативной справки. S

QL Server Management Studio содержит среду разработки для создания, редактирования и управления сценариев и хранимых процедур, используя Transact-SQL, многомерные выражения (MDX), XMLA и SQL Server Mobile Edition. Management Studio легко интегрируется с системой контроля версий. Management Studio также содержит инструменты для планирования задач.

Объекты Управления SQL являются новым набором объектов для программирования, которые предоставляет полную функциональность для управления базой данных SQL Server. Фактически, Management Studio построена на Объектах Управления SQL.

SMO реализована в виде сборки Microsoft .NET Framework. SMO можно использовать для автоматизации распространённых задач администрирования SQL Server, таких как программное получение конфигурационных настроек, создание баз данных, выполнение сценариев Transact-SQL, создание заданий Агента SQL Server и планирование резервного копирования. Объектная модель SMO является более безопасной, надёжной и масштабируемой заменой Распределённым Объектам Управления (DMO), которые входят в состав предыдущих версий SQL Server.

Выделенное административное соединение SQL Server привносит выделенное административное соединение для доступа к серверу даже если он не отвечает или недоступен по иной причине. Это позволяет вам выполнять диагностические функции или операторы Transact-SQL для разрешения проблем на сервере. Административное соединение активируется членами фиксированной серверной роли sysadmin и доступно только через утилиту командной строки SQLCMD либо локально, либо с удалённой машины.

Поддержка Web служб В SQL Server можно разрабатывать Web службы XML в уровне базы данных, используя SQL Server в качестве слушателя HTTP. Это даёт новый способ доступа к данным приложениям, которые сосредоточены вокруг Web служб.

В SQL Server можно использовать HTTP для прямого доступа к SQL Server, без применения слушателя промежуточного уровня, такого как Microsoft Internet Information Services (IIS). SQL Server предоставляет интерфейс Web службы для выполнения SQL выражений и вызова функций и процедур. Результаты запросов возвращаются в XML формате и здесь можно воспользоваться преимуществами инфраструктуры Web служб Visual Studio.

Важно отметить, что MS SQL Server рассчитан на работу именно на платформе Windows. Это ограничивает ее использование в различных сферах деятельности. Кроме того, компания Microsoft ориентируется на снижение стоимости СУБД, что приводит к уменьшению эффективности средств обеспечения надежности и безопасности. Это является важным фактором при выборе СУБД для организации.

# Объектно-ориентированный язык C#

# История возникновения

Язык C# появился на свет в июне 2000 г. в результате кропотливой работы большой группы разработчиков компании Microsoft, возглавляемой Андерсом Хейлсбергом (Anders Hejlsberg). Появление языка C# и инициативы .NET отнюдь не случайно пришлось на начало лета 2000 г.

Именно к этому моменту компания Microsoft подготовила промышленные версии новых компонентных технологий и решений в области обмена сообщениями и данными, а также создания Internet-приложений (COM+, ASP+, ADO+, SOAP, Biztalk Framework).

Несомненно, лучшим способом продвижения этих новинок является создание инструментария для разработчиков с их полноценной поддержкой. В этом и заключается одна из главных задач нового языка C#.

Кроме того, Microsoft не могла больше расширять все те же инструменты и языки разработки, делая их все более и более сложными.

Наступил момент, когда необходимо начать с чистого листа для того, чтобы создать простой, но имеющий сложную структуру набор языков, сред и средств разработки, которые позволят разработчику легко создавать современные программные продукты.

С# и .NET являются той самой отправной точкой. Если говорить упрощенно, то .NET представляет собой новую платформу, новый API (англ. Application Programming Interface - интерфейс прикладного программирования) для программирования в Windows, а С# - новый язык, созданный с нуля, для работы с этой платформой, а также для извлечения всех выгод из прогресса сред разработки и достижений объектно-ориентированного программирования в течение последних 20 лет.

# Общее описание языка

Учитывая все подобные пожелания, Microsoft разработала новый язык - C#. В него входит много полезных особенностей - простота, объектная ориентированность, типовая защищенность, "сборка мусора", поддержка совместимости версий и многое другое.

Данные возможности позволяют быстро и легко разрабатывать приложения, особенно COM+ приложения и Web сервисы. При создании C#, его авторы учитывали достижения многих других языков программирования: C++, C, Java, SmallTalk, Delphi, Visual Basic и т.д.

Надо заметить, что по причине того, что C# разрабатывался с чистого листа, у его авторов была возможность (которой они явно воспользовались), оставить в прошлом все неудобные и неприятные особенности (существующие, как правило, для обратной совместимости), любого из предшествующих ему языков.

В результате получился действительно простой, удобный и современный язык, по мощности не уступающий С++, но существенно повышающий продуктивность разработок.

C#, как в несомненно современном языке, также существуют характерные особенности для обхода ошибок, там все переменные автоматически инициализируются средой и обладают типовой защищенностью, что позволяет избежать неопределенных ситуаций в случае, если программист забудет инициализировать переменную в объекте или попытается произвести недопустимое преобразование типов.

Также в C# были предприняты меры для исключения ошибок при обновлении программного обеспечения. Изменение кода, в такой ситуации, может непредсказуемо изменить суть самой программы. Чтобы помочь разработчикам бороться с этой проблемой C# включает в себя поддержку совместимости версий (vesioning).

В частности, в отличии от C++ и Java, если метод класса был изменен, это должно быть специально оговорено. Это позволяет обойти ошибки в коде и обеспечить гибкую совместимость версий. Также новой особенностью является native поддержка интерфейсов и наследования интерфейсов. Данные возможности позволяют разрабатывать сложные системы и развивать их со временем.

В C# была унифицирована система типов, теперь вы можете рассматривать каждый тип как объект. Несмотря на то, используете вы класс, структуру, массив или встроенный тип, вы можете обращаться к нему как к объекту. Объекты собраны в пространства имен (namespaces), которые позволяют программно обращаться к чему-либо.

Это значит, что вместо списка включаемых файлов заголовков в своей программе вы должны написать какие пространства имен, для доступа к объектам и классам внутри них, вы хотите использовать.

В C# выражение using позволяет вам не писать каждый раз название пространства имен, когда вы используете класс из него. Теперь в C# все упрощено - теперь символ Unicode называется просто char (а не wchar\_t, как в С++) и 64-битное целое теперь - long (а не int64). Также в C# нет знаковых и беззнаковых символьных типов. В C#, также как и в Visual Basic после каждого выражения case в блоке switch подразумевается break. И более не будет происходить странных вещей если вы забыли поставить этот break.

Однако если вы действительно хотите чтобы после одного выражения case программа перешла к следующему вы можете переписать свою программу с использованием, например, оператора goto.. В C# нет указателей. В действительности нетривиальность указателей соответствовала их полезности.

В соответствии с этим в C# присутствуют Delegates - как прямой аналог указателя на функцию, но их отличает типовая защищенность, безопасность и полное соответствие концепциям объектно-ориентированного программирования. Так в C#, помимо просто выражения using, предоставляется еще одна очень удобная возможность - использование дополнительного имени (alias) пространства имен или класса. Современность C# проявляется и в новых шагах к облегчению процесса отладки программы.

# Особенности языка С#

* Полный и хорошо определенный набор основных типов.
* Встроенная поддержка автоматической генерации XML-документации. Автоматическое освобождение динамически распределенной памяти.
* Возможность отметки классов и методов атрибутами, определяемыми пользователем, (это может быть полезно при документировании и способно воздействовать на процесс компиляции - например, можно пометить методы, которые должны компилироваться только в отладочном режиме).
* Полный доступ к библиотеке базовых классов .NET, а также легкий доступ к Windows API (если это действительно необходимо).
* Указатели и прямой доступ к памяти, если они необходимы (однако язык разработан таким образом, что практически во всех случаях можно обойтись и без этого). Поддержка свойств и событий в стиле Visual Basic.
* Простое изменение ключей компиляции. Позволяет получать исполняемые файлы или библиотеки компонентов .NET, которые могут быть вызваны другим кодом так же, как элементы управления ActiveX (компоненты СОМ).
* Возможность использования C# для написания динамических web-страниц ASP.NET. Одной из областей, для которых не предназначен этот язык, являются критичные по времени и высокопроизводительные программы, когда имеет значение, занимать на исполнение цикла 1000 или 1050 машинных циклов, и освобождать ресурсы требуется немедленно. C++ остается в этой области наилучшим из языков высокого уровня. В C# отсутствуют некоторые ключевые моменты, необходимые для создания высокопроизводительных приложений, в частности подставляемые функции и деструкторы, выполнение которых гарантируется в определенных точках кода.

# Сфера применения C#

C# — один из языков программирования, который может использоваться для создания приложений. Некоторые наиболее часто встречающиеся типы приложений:

* Приложения Windows - это приложения вроде Microsoft Office, имеющие знакомый "Windows-подобный" вид и представление. Создавать такие приложения достаточно просто с помощью модуля .NET Framework, который называется Windows Forms и представляет собой библиотеку управляющих элементов (кнопок, панелей инструментов, меню и т. п.); эта библиотека может использоваться для создания пользовательского интерфейса (user interface, Ul) Windows.
* Web-приложения - эти приложения представляют собой web-страницы, которые могут просматриваться любым web-браузером. В состав .NET Framework входит мощная система динамического создания содержимого web-страниц, позволяющая идентифицировать пользователя, обеспечивать безопасность и пр. Эта система называется Active Server Pages.NET (ASP.NET — активные серверные страницы .NET); для создания приложений ASP.NET можно применять Web Forms языка С#.
* Web-службы - это новый замечательный способ создания гибких распределенных приложений. С помощью web-служб можно обмениваться по Интернету практически любыми данными с использованием единого простого синтаксиса независимо оттого, какой язык программирования применялся при создании web-службы и на какой системы она размещена.
* Приложениям всех перечисленных типов может потребоваться доступ к базам данных, что осуществляется с помощью раздела .NET Framework, называемого Active Data Objects.NET (ADO.NET — активные объекты с данными .NET).
* Также можно использовать и многие другие ресурсы, например, инструменты для создания сетевых компонентов, графического вывода, выполнения сложных математических вычислений и т.д.

# Разработка проекта

# Порядок создания базы данных

* 1. Создание таблиц в базе данных, с помощью Microsoft SQL Server Management Studio (Рис.1-2).

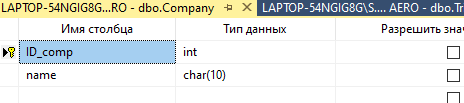


Рисунок 1 – Таблица Company

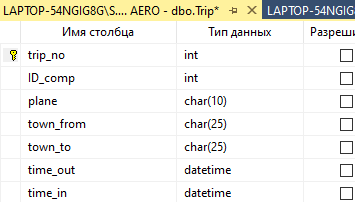


Рисунок 2 – Таблица Trip

* 1. Конструирование диаграммы и связей между таблицами (Рис.3).

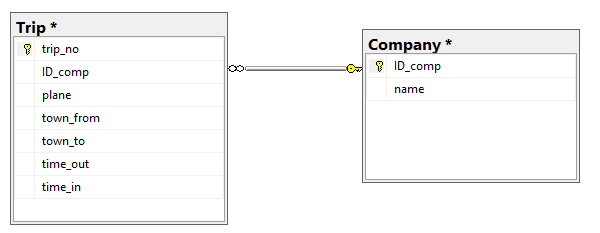


Рисунок 3 – Диаграмма

* 1. Создание проекта WPF и подготовка его в соответствии с паттерном MVC (Рис.4).

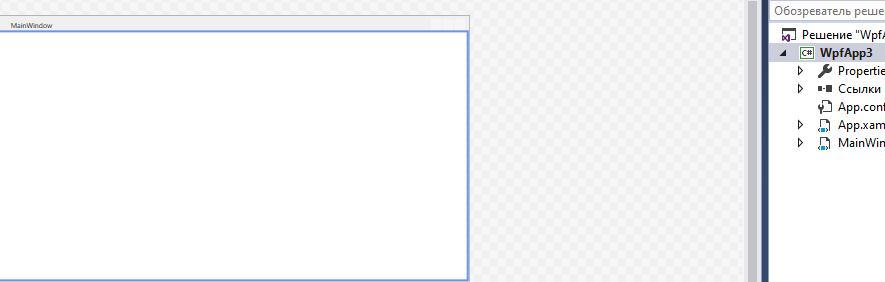


Рисунок 4 – Проект WPF

* 1. Добавление библиотеки EF (Рис.5-6).

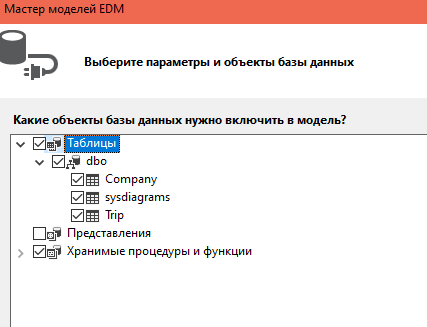


Рисунок 5 – Модель

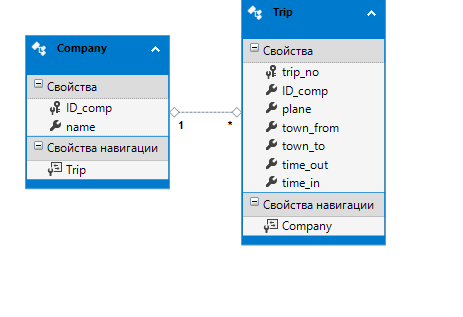


Рисунок 6 – Таблицы

* 1. Подготовка EF для того, что можно было добавлять элементы, как отображения, так и редактирования (Рис.7-8).

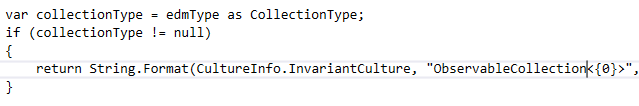


Рисунок 7 - Замена вхождения "ICollection" на "ObservableCollection"

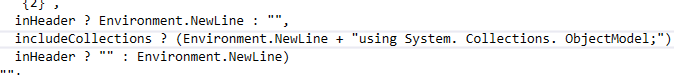


Рисунок 8 - Замена вхождения "System.Collections.Generic"   
на "System.Collections.ObjectModel"

* 1. Удаление стартового окна и замена его на страницу, указание в файле App.xaml новой страницы как стартовой (Рис.9-10).

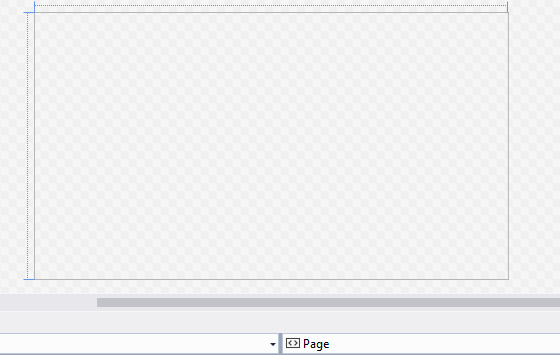


Рисунок 9 – Страница

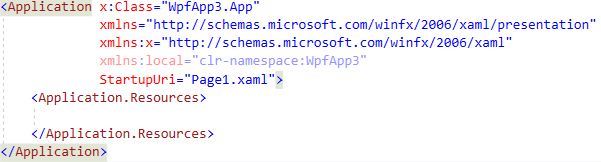


Рисунок 10 – Указание стартовой страницы

* 1. Создание разметки, состоящей из шапки (header), подвала (footer) и тела (body) (Рис.11-16).

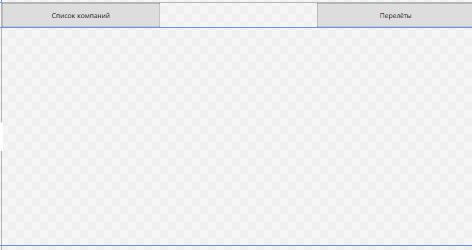


Рисунок 11 – Разметка страницы «HomePage»

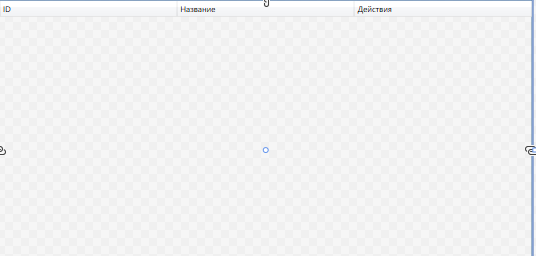


Рисунок 12 – Разметка страницы «CompanyPage»

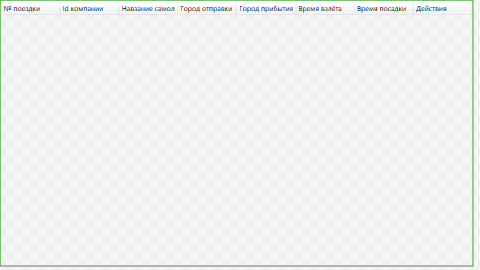


Рисунок 13 - Разметка страницы «TripPage»

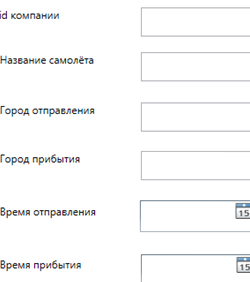


Рисунок 14 – Добавление рейсов



Рисунок 15 – Готовая страница «Рейсы»

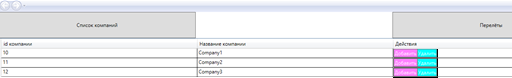


Рисунок 16 - Готовая страница «Компаний»

# ЗАКЛЮЧЕНИЕ

В ходе прохождения учебной практики, была реализована поставленная цель и разработана база данных. Изучены основы языка SQL.

Выполнены поставленные задачи, в которые входит следующее:

* 1. Приобретение практических навыков организации взаимодействия приложения с базой данных.
  2. Отработка технологии связывания таблиц базы данных.
  3. Закрепление навыков организации взаимодействия форм.
  4. Разработка модели базы данных.
  5. Отработка технологии дополнения и коррекции записей в таблице.

Из этого следует, что язык SQL ориентирован на операции с данными, представленными в виде логически взаимосвязанных совокупностей таблиц-отношений. Важнейшая особенность его структур – ориентация на конечный результат обработки данных, а не на процедуру этой обработки. Язык SQL сам определяет, где находятся данные, индексы и даже какие наиболее эффективные последовательности операций следует использовать для получения результата, а потому указывать эти детали в запросе к базе данных не требуется.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ

* 1. Дейт. К. Дж. Введение в системы баз данных – Introduction to Database Systems. — 8-е изд. — М.: «Вильямс», 2006.
  2. Елманова Н., Федоров А. Oracle и Microsoft SQL Server: прошлое, настоящее и будущее.
  3. Информатика: Базовый курс/Под ред. С.В.Симоновича. – СПБ.:Питер,2002. 400 с.
  4. Фридланд А.Я. Информатика и компьютерные технологии. А.Я. Фридланд, Л.С. Ханамирова.- М.: Астрель. 2003.204 с.
  5. Роберт Э. Уолтерс, Майкл Коулс SQL Server 2008: ускоренный курс для профессионалов = Accelerated SQL Server 2008. — М.: «Вильямс», 2008. — С. 768. — ISBN 978-5-8459-1481-1.
  6. Роберт Виейра Программирование баз данных Microsoft SQL Server 2005. Базовый курс = Beginning Microsoft SQL Server 2005 Programming. — М.: «Диалектика», 2007. — С. 832. — ISBN 0-7645-8433-2.
  7. Майк Гандерлой, Джозеф Джорден, Дейвид Чанц Освоение Microsoft SQL Server 2005 = Mastering Microsoft SQL Server 2005. — М.: «Диалектика», 2007. — С. 1104. — ISBN 0-7821-4380-6.
  8. Microsoft® SQL Server™ 2005.Реализация и обслуживание. Учебный курс Microsoft (Э.кзамен 70-431). — М.: «Питер», 2007. — С. 767. — ISBN 978-5-91180-3
  9. Петкович Д. Microsoft SQL Server 2008. Руководство для начинающих - www.bhv.ru/books/book.php?id=186172. — C.: БХВ-Петербург, 2009. — С. 752. — ISBN 978-5-9775-0149-1.