

СОДЕРЖАНИЕ

[ВВЕДЕНИЕ 3](#_Toc42052501)

[1 Актуальные проблемы баз данных 5](#_Toc42052502)

[1.1 Классификация баз данных 5](#_Toc42052503)

[1.2 Рынок программного обеспечения СУБД 8](#_Toc42052504)

[1.3 Выбор СУБД 13](#_Toc42052505)

[2 Проектирование базы данных документооборота предприятия 15](#_Toc42052506)

[2.1 Постановка задачи 15](#_Toc42052507)

[2.2 Построение инфологической модели базы данных 16](#_Toc42052508)

[2.3 Преобразование ER–диаграммы в схему базы данных 18](#_Toc42052509)

[2.4 Составление реляционных отношений 19](#_Toc42052510)

[2.5 Нормализация полученных отношений (до 3НФ) 20](#_Toc42052511)

[3 Разработка базы данных 22](#_Toc42052512)

[3.1 Физическое проектирование БД 22](#_Toc42052513)

[3.2 Реализация пользовательских запросов 25](#_Toc42052514)

[3.3 Интерфейс пользователя 33](#_Toc42052515)

[ЗАКЛЮЧЕНИЕ 40](#_Toc42052516)

[СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ 41](#_Toc42052517)

[ПРИЛОЖЕНИЯ 43](#_Toc42052518)

# ВВЕДЕНИЕ

Современные компании и организации функционируют в условиях большого объема постоянно изменяющейся информации, которую необходимо оперативно анализировать и принимать правильные решения.

С экономической точки зрения информация, представленная в системе, получаемая и накапливаемая в процессе деятельности станет в управлении как фактор увеличения объемов производства и повышения его эффективности. При этом информация как средство управления выступает своеобразным коррелятором организованности системы и служит основой процесса принятия управленческих решения, а также оптимизации последних с учетом принимаемых критериев эффективности.

Бурно развивается вычислительная техника и информационные технологии. Трудно найти сейчас компанию, не занимающуюся развитием информационных технологий. Современные руководители фирм полностью отдают себе отчет в том, что в настоящее время успешность и прибыльность компании полностью зависят в том числе, и от уровня развития IT-технологий, скорости и качества обработки информации, обоснованности и взвешенности принимаемых решений.

Обработка больших объемов информации в современных компаниях невозможна без использования информационных систем, которые стали самостоятельным научно-техническим направлением.

Целью курсовой работы является рассмотрение и освоение технологии ведения учета поручений руководителя и их исполнения с использованием автоматизированной системы.

Для достижения поставленной цели потребовалось решить следующие задачи:

* изучить программные средства проектирования базы данных и ее реализации;
* провести объектно-ориентированный анализ выбранной предметной области и построить ее модель;
* реализовать базу данных в среде Access.
* В качестве инструментальной среды проектирования используются AllFusion Process Modeler и Access. Для описания создания базы данных используется язык UML.

Предметом курсовой работы выступает документооборот предприятия.

Объектом исследования выступает информационная среда MS Access, где будет реализована система.

Основные теоретические и методологические вопросы формирования и реализации автоматизированных систем управления нашли отражение в работах таких авторов, как Ю.П. Анисимов, Ю.П. Анискин, М.И. Бухалков, В.Н. Васильев, О.С. Виханский, Е.Г. Гинзбург, В.Н. Гончаров, С.Ю. Глазьев, А.П. Градов, В.Ф. Ершов, А.К. Казанцев, В.Д. Калачанов, А.И. Канащенков, Г.Б. Клейнер, Г.А. Краюхин, Э.С. Минаев, Н.К. Моисеева.

# 1 Актуальные проблемы баз данных

## 1.1 Классификация баз данных

В 1965 г. на конференции CODASYL (Conference on Data System Languages) создана рабочая группа, которая должна была определить спецификации среды, допускавшие бы разработку баз данных и управление данными. В 1971 году был опубликован отчет о результатах работы этой группы, определявший три компонента:

* сетевая схема (организации базы данных в целом);
* подсхема (часть базы данных, как она видится пользователям и приложениям);
* язык управлении данными, включавший язык описания данных (ЯОД, или DDL – Data definition Language) и язык манипулирования данными (ЯМД или DML – Data Manipulation Language).

Системы, созданный на основе результатов CODASYL относятся к СУБД первого поколения и используют сетевые и иерархические модели данных.

В 1970 г. Э. Кодд опубликовал статью о реляционной модели данных, что ознаменовало новый этап в развитии баз данных и послужило мощным толчок к развитию реляционных СУБД. Коммерческие СУБД, использующие реляционную модель данных, появились в конце 1970-х – начале 1980-х гг. Особо следует отметить СУБД System R (IBM, 1976 г.), в которой был использован язык SQL. СУБД, использующие реляционную модель данных, – это СУБД второго поколении.

В 1976 г. П. Чен представил модель «сущность – связь», определив тем самым технологию проектировании баз данных и дав толчок новому этапу в их развитии. В результате чего появились расширенная реляционная модель данных и семантические модели данных.

С ростом сложности приложений стали использовать объектно-ориентированные модели данных, в результате чего появились объектно-ориентированные и объектно-реляционные СУБД, которые определяются как СУБД третьего поколения [16, с. 11].

Существует множество подходов к классификации баз данных в зависимости от признака, по которому осуществляется классификация (рис. 1).



Рисунок 1 – Классификация баз данных [8, с. 18]

При рассмотрении вычислений принято различать централизованные базы данных, расположенные на «универсальных ЭВМ» (mainframe) и БД архитектуры «клиент-сервер». Централизованные БД доминировали в 70-80-х годах, архитектура «клиент-сервер» является основной в настоящее время.

При централизованном подходе вычисления проводятся па «большой» универсальной вычислительной машине, к которой подключены терминалы. Электронные картотеки (или наборы данных) организованы в виде множества пар «Главная – Подчиненная». Запись главной картотеки содержит (физические) указатели па одну или несколько записей подчиненной картотеки. Общая организация БД на логическом уровне может быть представлена в виде иерархии (дерева), где узлу (вершины дерева) представляют картотеки, имеющие одинаковую структуру записей. Такие БД называются иерархическими - классическим представителем БД этого класса является система IMS фирмы IBM.

В 80-х годах начали использовать БД, логическая структура данных которых представляется в виде сети – внутри структуры данных разрешаются циклы. Такая модель данных называется сетевой. И в сетевой, и в иерархической модели данных работа с данными состояла в «перемещении фокуса вычисления» по «Графу картотек». Поэтому эти две модели данных иногда называют навигационными моделями, чтобы отличать их от реляционной модели.

Для определения данных и разработки прикладных программ в навигационных БД использовался (и продолжает использоваться) алгоритмический язык COBOL. Чтобы оценить масштаб современного применения навигационных БД, использующихся в «старых» системах (или в унаследованных системах – Legacy system), достаточно сказать, что число программистов, пишущих на языке COBOL, в настоящее время превышает два миллиона человек во всем мире.

В архитектуре «клиент-сервер» используются распределенные вычисления – на стороне пользователей (или «клиентов», т.е. на стороне прикладных программ) вычисления проводятся на ПЭВМ, которые имеют доступ к данным на сервере. Сервер – это логический процесс, который обеспечивает обслуживание запросов других процессов на предоставление доступа к общему ресурсу (например, к принтеру, к электронной почте и т.п.). Сервер не посылает данных клиенту, пока от него не поступит запрос. Сервер отвечает за управление синхронизацией обслуживания клиентов и за связи сервера с другими процессами. Таким образом, сервер – это не узел локальной вычислительной сети (ЛВС), а логический процесс, отвечающий за обработку запросов к ВД.

Клиент – это процесс, посылающий серверу запрос на обслуживание. Клиент может начать взаимодействие е сервером, а сервер – нет. Как правило, клиент – это прикладная программа, выполняющаяся на рабочем месте (например, ПЭВМ) пользователя. Она обеспечивает автоматизацию бизнес-задач пользователя, например, ввод данных о заказе клиента. Клиент и сервер, как правило, выполняющаяся на различных узлах сети, хотя это не обязательно – они могут физически выполняться на одной и той же машине. Технология «клиент-сервер» обеспечивает распределенную обработку данных используются различные вычислители, связанные друг с другом сетью.

На физическом уровне СУБД может хранить данные на многих ЭВМ, – узлах сети. В этом случае говорят о распределенной базе данных. Клиенты могут не знать о таком распределении и рассматривать данные (на логическом уровне) как «хранимые в одном месте», т.е. на одной машине. Взаимодействие клиента с сервером осуществляется с использованием языка, ориентированного на работу с БД. Для реляционных БД таким языком является SQL. После обработки запроса сервер БД возвращает клиенту только данные, удовлетворяющие запросу.

## 1.2 Рынок программного обеспечения СУБД

Как уже отмечалось ранее для управления базами данных используются систему управления базами данных, которые также могут быть классифицированы по различным признакам (рис. 2).



Рисунок 2 – Классификация СУБД

CУБД представляет собой оболочку, с помощью которой при организации структуры таблиц и заполнения их данными получается та или иная база данных. В связи с этим полезно поговорить о системе программно-технических, организационных и «человеческих» составляющих (рис. 3). Программные средства включают систему управления, обеспечивающую ввод-вывод, обработку и хранение информации, создание, модификацию и тестирование БД, трансляторы.



Рисунок 3 – Состав СУБД

Базовыми внутренними языками программирования являются языки четвертого поколения. В качестве базовых языков могут использоваться С, C++, Pascal, Object Pascal. Язык C++ позволяет строить программы на языке Visual Basic с широким спектром возможностей, более близком и понятном даже пользователю-непрофессионалу, и на непроцедурном (декларативном) языке структурированных запросов SQL. Следует отметить, что исторически для системы управления базой данных сложились три языка:

1. Язык описания данных (ЯОД), называемый также языком описания схем, – для построения структуры («шапки») таблиц БД;
2. Язык манипулирования данными (ЯМД) – для заполнения БД данными и операций обновления (запись, удаление, модификация);
3. Язык запросов – язык поиска наборов величин в файле в соответствии с заданной совокупностью критериев поиска и выдачи затребованных данных без изменения содержимого файлов и БД (язык преобразования критериев в систему команд).

В настоящее время функции всех трех языков выполняет язык SQL, относящийся к классу языков, базирующихся на исчислении кортежей (кортеж чаще всего является единицей информации), языки СУБД FoxPro, Visual Basic for Application (СУБД Access) и т.д.

Вместе с тем сохранились и языки запросов, например, язык запросов по примеру Query By Example (QBE) класса исчисления доменов. Отметим, что эти языки в качестве «информационной единицы» БД используют отдельную запись. С помощью языков БД создаются приложения, базы данных и интерфейс пользователя, включающий экранные формы, меню, отчеты. При создании БД на базе СУБД FoxPro эти элементы (объекты) фиксируются в отдельных файлах, которые, в свою очередь, сосредоточиваются в одном файле, называемом проектом. После отработки БД проект преобразуется в приложение. В СУБД Access все созданные объекты размещаются в одном файле.

Более точно, к числу функций СУБД принято относить следующие:

1. Непосредственное управление данными во внешней памяти. Эта функция включает обеспечение необходимых структур внешней памяти как для хранения данных, непосредственно входящих в БД, так и для служебных целей, например, для ускорения доступа к данным в некоторых случаях (обычно для этого используются индексы). В некоторых реализациях СУБД активно используются возможности существующих файловых систем, в других работа производится вплоть до уровня устройств внешней памяти. Но подчеркнем, что в развитых СУБД пользователи в любом случае не обязаны знать, использует ли СУБД файловую систему, и если использует, то как организованы файлы. В частности, СУБД поддерживает собственную систему именования объектов БД.

2. Управление буферами оперативной памяти. СУБД обычно работают с БД значительного размера; по крайней мере, этот размер обычно существенно больше доступного объема оперативной памяти. Понятно, что если при обращении к любому элементу данных будет производиться обмен с внешней памятью, то вся система будет работать со скоростью устройства внешней памяти. Практически единственным способом реального увеличения этой скорости является буферизация данных в оперативной памяти. При этом, даже если операционная система производит общесистемную буферизацию (как в случае ОС UNIX), этого недостаточно для целей СУБД, которая располагает гораздо большей информацией о полезности буферизации той или иной части БД. Поэтому в развитых СУБД поддерживается собственный набор буферов оперативной памяти с собственной дисциплиной замены буферов.

3. Управление транзакциями.

Транзакция – это последовательность операций над БД, рассматриваемых СУБД как единое целое.

Либо транзакция успешно выполняется, и СУБД фиксирует изменения БД, произведенные этой транзакцией, во внешней памяти, либо ни одно из этих изменений никак не отражается на состоянии БД.

То свойство, что каждая транзакция начинается при целостном состоянии БД и оставляет это состояние целостным после своего завершения, делает очень удобным использование понятия транзакции как единицы активности пользователя по отношению к БД. При соответствующем управлении параллельно выполняющимися транзакциями со стороны СУБД каждый из пользователей может в принципе ощущать себя единственным пользователем СУБД (на самом деле, это несколько идеализированное представление, поскольку в некоторых случаях пользователи многопользовательских СУБД могут ощутить присутствие своих коллег).

4. Журнализация.

Одним из основных требований к СУБД является надежность хранения данных во внешней памяти. Под надежностью хранения понимается то, что СУБД должна быть в состоянии восстановить последнее согласованное состояние БД после любого аппаратного или программного сбоя. Обычно рассматриваются два возможных вида аппаратных сбоев: так называемые мягкие сбои, которые можно трактовать как внезапную остановку работы компьютера (например, аварийное выключение питания), и жесткие сбои, характеризуемые потерей информации на носителях внешней памяти.

Журнал – это особая часть БД, недоступная пользователям СУБД и поддерживаемая с особой тщательностью (иногда поддерживаются две копии журнала, располагаемые на разных физических дисках), в которую поступают записи обо всех изменениях основной части БД. В разных СУБД изменения БД журнализуются на разных уровнях: иногда запись в журнале соответствует некоторой логической операции изменения БД (например, операции удаления строки из таблицы реляционной БД), иногда – минимальной внутренней операции модификации страницы внешней памяти; в некоторых системах одновременно используются оба подхода.

Во всех случаях придерживаются стратегии «упреждающей» записи в журнал (так называемого протокола Write Ahead Log – WAL). Грубо говоря, эта стратегия заключается в том, что запись об изменении любого объекта БД должна попасть во внешнюю память журнала раньше, чем измененный объект попадет во внешнюю память основной части БД. Известно, что если в СУБД корректно соблюдается протокол WAL, то с помощью журнала можно решить все проблемы восстановления БД после любого сбоя.

Самая простая ситуация восстановления – индивидуальный откат транзакции. Строго говоря, для этого не требуется общесистемный журнал изменений БД. Достаточно для каждой транзакции поддерживать локальный журнал операций модификации БД, выполненных в этой транзакции, и производить откат транзакции, путем выполнения обратных операций, следуя от конца локального журнала. В некоторых СУБД так и делают, но в большинстве систем локальные журналы не поддерживают, а индивидуальный откат транзакции выполняют по общесистемному журналу, для чего все записи от одной транзакции связывают обратным списком (от конца к началу).

5. Поддержка языков БД

Для работы с базами данных используются специальные языки, в целом называемые языками баз данных. В ранних СУБД поддерживалось несколько специализированных по своим функциям языков.

SDL служил главным образом для определения логической структуры БД, т.е. той структуры БД, какой она представляется пользователям. DML содержал набор операторов манипулирования данными, т.е. операторов, позволяющих заносить данные в БД, удалять, модифицировать или выбирать существующие данные [8, с. 22-28].

## 1.3 Выбор СУБД

Исследования выявили несколько возможных средств, которые могут быть использованы при создании разрабатываемой АИС. В их число вошли такие среды разработки, как Microsoft Excel и Microsoft Access из пакета Microsoft Office, 1С, MS SQL. В крупных проектах, как правило, используются такие среды, как 1С и MS SQL, рассчитанные на большие компании, способные работать с несколькими сотнями клиентов одновременно, обрабатывая большие потоки информации. Но эти системы, вместе со своими большими достоинствами, имеют небольшой недостаток – лицензия даже на одного клиента имеет высокую стоимость.

Количество обрабатываемой информации разрабатываемой системы не столь велико, чтобы использовать мощные среды разработки. Оплачиваемый продукт не будет окупаться, так как слишком дорог для поставленной задачи. Кроме того, для обслуживания 1С и MS SQL необходим специальный персонал. Сложность обучения пользования данными продуктами персонала, также имеет значение. Достаточно большое количество обучающей литературы требует дополнительного времени работников для ее изучения. И поэтому выбор пал на пакет Microsoft Office.

Пакет Microsoft Office включает в себя два продукта, которые могут быть использованы для создания базы данных. Сравнив их между собой, увидим, что Excel, несмотря на свою низкую стоимость и простоту использования, имеет ряд недостатков в сравнении с Access. Например, отсутствие защиты ключом, сложность использования более чем одним человеком, а также, неудобство работы с данными в текстовом формате, так как изначально программа была нацелена на табличную обработку цифровой информации.

Плюсы Access: удобный интерфейс как для программиста, так и для простого пользователя ПК; создание экранных форм; большой выбор функциональных возможностей, которые предоставляют Мастера (Wizard); хранение информации в едином файле, что позволяет не просматривать каталоги в поисках необходимого файла; достаточно большое количество литературы для программиста (некоторые среды не имеют документации на русском языке); высокая надежность сохранности информации при пользовании небольшим количеством клиентов; OLE (Object Linking and Embedding) при помощи которого можно связывать компоненты MS в Access.

Широкий диапазон возможностей, который предоставляет Access и который наиболее подходит под заданные требования, заставляет делать выбор в его пользу.

# 2 Проектирование базы данных документооборота предприятия

## 2.1 Постановка задачи

В курсовой работе в качестве предметной области рассматривается ведение оперативного учета поручений руководителя и их исполнения. Данный учет осуществляется секретариатом, который ведет документооборот только с использованием средств MS Word и непосредственной передачей документов соответствующим сотрудникам и подразделениям предприятия. Такая система увеличивает время на передачу поручения, а также расход на ведение внутреннего документооборота.

Предметной областью данного курсовой работы является документооборот. Бизнес-процесс выглядит следующим образом:

1. Руководитель выдает поручение сотруднику или подразделению.
2. Поручение фиксируется в журнале. Здесь отображаются дата, руководитель, содержание поручения, категория (важность), срок исполнения, кому поручено (исполнитель или подразделение).
3. После исполнения поручения создается документ, в котором отображается дата, вид документа, по какому поручению

Для автоматизации данного процесса нужно разработать аналитический отчет об исполнительской дисциплине за заданный период времени, сгруппированный по исполнителям: количество выданных поручений, количество исполненных поручений, процент исполненных в срок, количество неисполненных поручений.

Определим границы информационной поддержки пользователей:

1) Функциональные возможности:

* ведение БД (запись, чтение, модификация, удаление в архив);
* обеспечение логической непротиворечивости БД;
* реализация наиболее часто встречающихся запросов в готовом виде;
* предоставление возможности сформировать произвольный запрос на языке манипулирования данными.

2) Готовые запросы:

* получение списка текущих клиентов;
* определение эффективности;
* определение неэффективности;

## 2.2 Построение инфологической модели базы данных

Цель инфологического проектирования – обеспечить наиболее естественный для человека способ сбора и представления информации, хранимой в проектируемой базе данных. Из-за этого модель данных нужно строить аналогично естественному языку (естественный язык, однако, нельзя использовать в чистом виде из-за проблемы с компьютерной обработкой и неоднозначности понятий естественного языка).

Первым этапом инфологического проектирования является выявление сущностей и зависимостей между ними.

В соответствии с поставленной задачей можно выделить следующие сущности:

1. Сущность «Подразделения» содержит информацию об обособленных подразделениях и отделах предприятия. В каждом подразделении работают несколько сотрудников.
2. Сущность «Должности» включает список должностей сотрудников.
3. Сущность «Сотрудники» содержит информацию о сотрудниках предприятия. Каждый сотрудник числится в определенном подразделении.
4. Сущность «Важность» содержит статусы важности того или иного поручения или указа.
5. Сущность «Поручения» содержит список документов с указанием даты и лица, подписавшего документ.
6. Сущность «Подразделение получатель». Подразделение» содержит список подразделений-получателей поручения, если поручение издается не для конкретного лица, а для всего отдела, а также дату исполнения.
7. Сущность «Сотрудник получатель» содержит список сотрудников-получателей поручения, если поручение издается для конкретного лица, а также дату исполнения.

Построим ER–диаграмму (рис. 4).



Рисунок 4 – ER–диаграмма базы данных

## 2.3 Преобразование ER–диаграммы в схему базы данных

База данных создается на основании схемы базы данных. Для преобразования ER–диаграммы в схему БД приведем уточненную ER-диаграмму, содержащая атрибуты сущностей (рис. 5).



Рисунок 5 – Уточненная ER-диаграмма

Преобразование ER-диаграммы в схему БД выполняется путем сопоставления каждой сущности и каждой связи, имеющей атрибуты, отношения (таблицы БД).

## 2.4 Составление реляционных отношений

Каждое реляционное отношение соответствует одной сущности (объекту ПО) и в него вносятся все атрибуты сущности. Для каждого отношения необходимо определить первичный ключ и внешние ключи (если они есть). В том случае, если базовое отношение не имеет потенциальных ключей, вводится суррогатный первичный ключ, который не несет смысловой нагрузки и служит только для идентификации записей (например, счетчик).

Потенциальными ключами отношений «Поручения» является атрибут «Номер», а остальных – «Код».

Отношения приведены в табл. 1-3.

Таблица 1 – Схема отношения «Подразделения»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Тип | Ключ |
| Код | Счетчик | Да |
| Наименование | Короткий текст |  |

Таблица 2 – Схема отношения «Должности»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Тип | Ключ |
| Код | Счетчик | Да |
| Наименование | Короткий текст |  |

Таблица 3 – Схема отношения «Сотрудники»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Тип | Ключ |
| Код | Счетчик | Да |
| Фамилия | Короткий текст |  |
| Имя | Короткий текст |  |
| Отчество | Короткий текст |  |
| Дата рождения | Дата и время |  |
| Адрес | Короткий текст |  |
| Телефон | Короткий текст |  |
| Подразделение | Числовой |  |
| Должность | Числовой |  |

Таблица 4 – Схема отношения «Важность»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Тип | Ключ |
| Код | Счетчик | Да |
| Наименование | Короткий текст |  |

Таблица 5 – Схема отношения «Поручения»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Тип | Ключ |
| Номер | Счетчик | Да |
| Дата | Дата и время |  |
| Лицо, подписавшее документ | Числовой |  |
| Содержание | Длинный текст |  |
| Срок исполнения | Дата и время |  |
| Степень важности | Числовой |  |

Таблица 6 – Схема отношения «Подразделение получатель»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Тип | Ключ |
| Код | Счетчик | Да |
| Поручение | Числовой |  |
| Подразделение | Числовой |  |
| Дата исполнения | Дата и время |  |

Таблица 7 – Схема отношения «Сотрудник получатель»

|  |  |  |
| --- | --- | --- |
| Содержание поля | Тип | Ключ |
| Код | Счетчик | Да |
| Поручение | Числовой |  |
| Сотрудник | Числовой |  |
| Дата исполнения | Дата и время |  |

## 2.5 Нормализация полученных отношений (до 3НФ)

1НФ. Для приведения таблиц к 1НФ требуется составить прямоугольные таблицы (один атрибут – один столбец) и разбить сложные атрибуты на простые, а многозначные атрибуты вынести в отдельные отношения.

Поскольку сложных полей нет, переходим ко второй форме нормализации.

2НФ. В нашем случае составные первичные ключи отсутствуют.

3НФ. Во всех отношениях все атрибуты зависят от первичных ключей, следовательно, отношения соответствуют третьей нормальной форме.

4НФ. Отношения данного примера не нарушают 4НФ, т.к. не содержат нетривиальных многозначных зависимостей.

Схема базы данных (рис. 5) и схемы отношений базы данных (табл. 1-7) не изменились. Получаем логическую схему информационной системы (рис. 6).



Рисунок 6 – Логическая модель данных информационной системы

# 3 Разработка базы данных

## 3.1 Физическое проектирование БД

Создадим ранее описанные таблицы:

1. Сущность «Подразделения»:

CREATE TABLE Подразделения (

Код COUNTER CONSTRAIN,

Наименование VARCHAR(225) NOT NULL,

PRIMARY KEY (Код)

);

2. Сущность «Должности»:

CREATE TABLE Должности (

Код COUNTER CONSTRAIN,

Наименование VARCHAR(225) NOT NULL,

PRIMARY KEY (Код)

);

3. Сущность «Сотрудники»:

CREATE TABLE Сотрудники(

Код COUNTER CONSTRAIN,

Фамилия VARCHAR(225) NOT NULL,

Имя VARCHAR(225) NOT NULL,

Отчество VARCHAR(225) NOT NULL,

Дата рождения DATE NOT NULL,

Адрес VARCHAR(225) NOT NULL,

Телефон VARCHAR(225) NOT NULL,

Подразделение INTEGER NOT NULL,

Должность INTEGER NOT NULL,

PRIMARY KEY (Код),

FOREIGN KEY ([Подразделение])

REFERENCES Подразделения(Код)

ON DELETE RESTRICT

ON UPDATE RESTRICT,

FOREIGN KEY ([Должность])

REFERENCES Должность(Код)

ON DELETE RESTRICT

ON UPDATE RESTRICT

);

4. Сущность «Важность»:

CREATE TABLE Важность (

Код COUNTER CONSTRAIN,

Наименование VARCHAR(225) NOT NULL,

PRIMARY KEY (Код)

);

5. Сущность «Поручения»:

CREATE TABLE Поручения(

Номер COUNTER CONSTRAIN,

Лицо, подписавшее документ INTEGER,

Содержание VARCHAR(MAX),

Дата DATE NOT NULL,

Срок исполнения DATE,

Степень важности INTEGER NOT NULL,

PRIMARY KEY (Номер),

FOREIGN KEY ([Сотрудники])

REFERENCES Сотрудники(Код)

ON DELETE RESTRICT

ON UPDATE RESTRICT,

FOREIGN KEY ([Важность])

REFERENCES Важность (Код)

ON DELETE RESTRICT

ON UPDATE RESTRICT

);

6. Сущность «Подразделение получатель»:

CREATE TABLE [Подразделение получатель](

Код COUNTER CONSTRAIN,

Поручение INTEGER,

Подразделение INTEGER,

Дата исполнения DATE NOT NULL,

PRIMARY KEY (Код),

FOREIGN KEY ([Поручение])

REFERENCES Поручения(Код)

ON DELETE RESTRICT

ON UPDATE RESTRICT,

FOREIGN KEY ([Подразделение])

REFERENCES Подразделения(Код)

ON DELETE RESTRICT

ON UPDATE RESTRICT

);

7. Сущность «Сотрудник получатель»:

CREATE TABLE [Сотрудник получатель](

Код COUNTER CONSTRAIN,

Поручение INTEGER,

Сотрудник INTEGER,

Дата исполнения DATE NOT NULL,

PRIMARY KEY (Код),

FOREIGN KEY ([Поручение])

REFERENCES Поручения(Код)

ON DELETE RESTRICT

ON UPDATE RESTRICT,

FOREIGN KEY ([Сотрудник])

REFERENCES Сотрудники(Код)

ON DELETE RESTRICT

ON UPDATE RESTRICT

);

В результате реализации в среде СУБД MS Access получается схема данных (рис. 7).

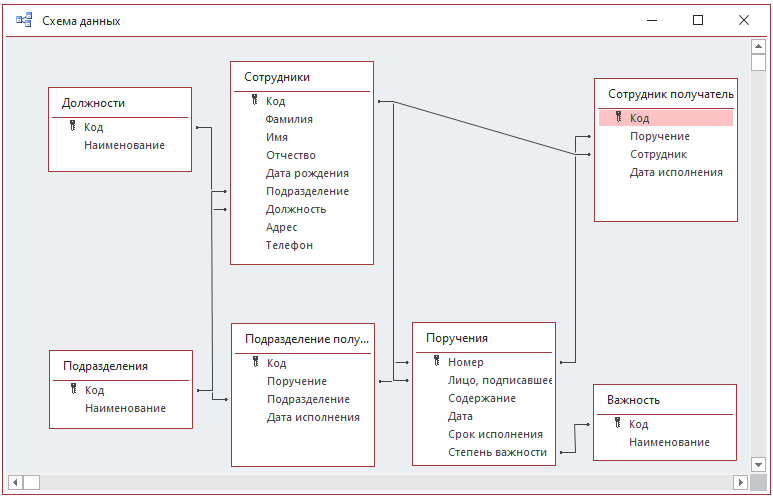


Рисунок 7 – Схема данных в СУБД MS Access

## 3.2 Реализация пользовательских запросов

1. Определим долю поручений по степени важности:

SELECT Поручения.[Степень важности], ROUND(Count(Поручения.Номер) / (SELECT Count(\*) FROM [Поручения]) \* 100,2) AS [Доля, %]

FROM Поручения

GROUP BY Поручения.[Степень важности];

Результат запроса приведен ниже (рис. 8).

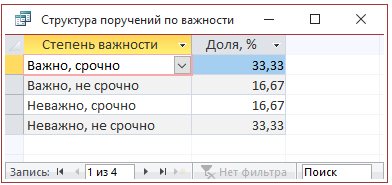


Рисунок 8 – Результат запроса на получение структуры поручений

2. Определим количество поручений для каждого подразделения и их долю:

SELECT T1.Подразделение, Count(T1.Количество) AS [Число поручений] , ROUND(Count(T1.Количество) / (SELECT Count(\*) FROM [Поручения]) \* 100,2) AS [Доля, %] FROM

(SELECT Подразделения.Наименование AS Подразделение, Count(Поручения.Номер) AS Количество

FROM Подразделения INNER JOIN (Поручения INNER JOIN [Подразделение получатель] ON Поручения.Номер = [Подразделение получатель].Поручение) ON Подразделения.Код = [Подразделение получатель].Подразделение

GROUP BY Подразделения.Наименование

UNION ALL

SELECT Подразделения.Наименование AS Подразделение, Count(Поручения.Номер) AS Количество

FROM Подразделения INNER JOIN (Сотрудники INNER JOIN (Поручения INNER JOIN [Сотрудник получатель] ON Поручения.Номер = [Сотрудник получатель].Поручение) ON Сотрудники.Код = [Сотрудник получатель].Сотрудник) ON Подразделения.Код = Сотрудники.Подразделение

GROUP BY Подразделения.Наименование) AS T1

GROUP BY T1.Подразделение

Результат запроса приведен ниже (рис. 9).

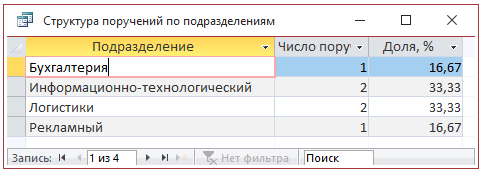


Рисунок 9 – Результат запроса на получение структуры поручений

3. Получение списка всех поручений:

SELECT Номер, Дата, Важность.Наименование AS [Степень важности], [Срок исполнения, в днях], Статус, [Срок задержки], Исполнитель, Сотрудники.Фамилия & ' ' & LEFT(Сотрудники.Имя,1) & '.' & LEFT(Сотрудники.Отчество,1) & '.' AS [Лицо, подписавшее документ]

FROM ((SELECT Поручения.Номер, Поручения.Дата, [Поручения].[Степень важности], DateDiff('d',[Дата],[Срок исполнения]) AS [Срок исполнения, в днях], IIf(IsNull([Подразделение получатель]![Дата исполнения]),'Выполняется','Выполнен') AS [Статус], IIf(DateDiff('d',[Поручения]![Срок исполнения],[Подразделение получатель]![Дата исполнения])>0,DateDiff('d',[Поручения]![Срок исполнения],[Подразделение получатель]![Дата исполнения]),0) AS [Срок задержки], Подразделения.Наименование & ' отдел' AS Исполнитель, Поручения.[Лицо, подписавшее документ]

FROM Подразделения INNER JOIN (Поручения INNER JOIN [Подразделение получатель] ON Поручения.Номер = [Подразделение получатель].Поручение) ON Подразделения.Код = [Подразделение получатель].Подразделение

UNION ALL

SELECT Поручения.Номер, Поручения.Дата, [Поручения].[Степень важности], DateDiff('d',[Дата],[Срок исполнения]) AS [Срок исполнения, в днях], IIf(IsNull([Сотрудник получатель]![Дата исполнения]),'Выполняется','Выполнен') AS [Статус],IIf(DateDiff('d',[Поручения]![Срок исполнения],[Сотрудник получатель]![Дата исполнения])>0,DateDiff('d',[Поручения]![Срок исполнения],[Сотрудник получатель]![Дата исполнения]),0) AS [Срок задержки], Сотрудники.Фамилия & ' ' & LEFT(Сотрудники.Имя,1) & '.' & LEFT(Сотрудники.Отчество,1) & '.' AS Исполнитель, Поручения.[Лицо, подписавшее документ]

FROM Сотрудники INNER JOIN (Поручения

INNER JOIN [Сотрудник получатель] ON Поручения.Номер = [Сотрудник получатель].Поручение) ON (Сотрудники.Код = [Сотрудник получатель].Сотрудник)) AS T1 INNER JOIN Сотрудники ON T1.[Лицо, подписавшее документ] = Сотрудники.Код) INNER JOIN Важность ON Важность.Код = T1.[Степень важности]

ORDER BY T1.Номер;

Результат запроса приведен ниже (рис. 10).

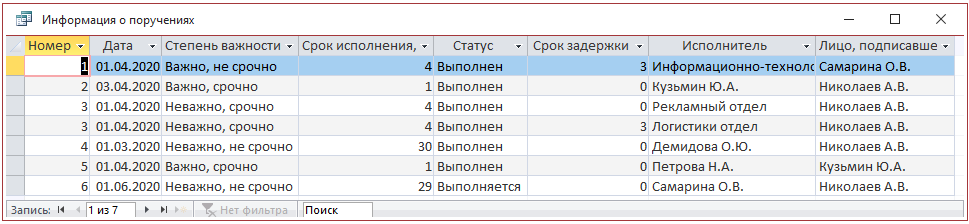


Рисунок 10 – Результат запроса на получение списка всех поручений

4. На основании полученного запроса выберем те поручения, которые еще выполняются:

SELECT [Информация о поручениях].Номер, [Информация о поручениях].Дата, [Информация о поручениях].[Степень важности], [Информация о поручениях].[Срок исполнения, в днях], [Информация о поручениях].[Срок задержки], [Информация о поручениях].Исполнитель, [Информация о поручениях].[Лицо, подписавшее документ]

FROM [Информация о поручениях]

WHERE [Информация о поручениях].Статус="Выполняется";

Результат запроса приведен ниже (рис. 11).

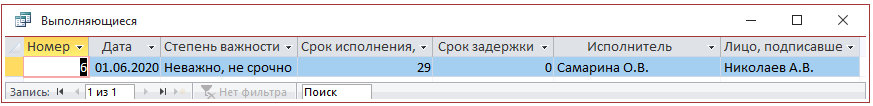


Рисунок 11 – Результат запроса на получение выполняющихся поручений

5. На основании полученного запроса выберем также поручения, которые были выполнены с задержкой:

SELECT [Информация о поручениях].Номер, [Информация о поручениях].Дата, [Информация о поручениях].[Степень важности], [Информация о поручениях].[Срок исполнения, в днях], [Информация о поручениях].[Срок задержки], [Информация о поручениях].Исполнитель, [Информация о поручениях].[Лицо, подписавшее документ]

FROM [Информация о поручениях]

WHERE [Информация о поручениях].[Срок задержки]>0;

Результат запроса приведен ниже (рис. 12).

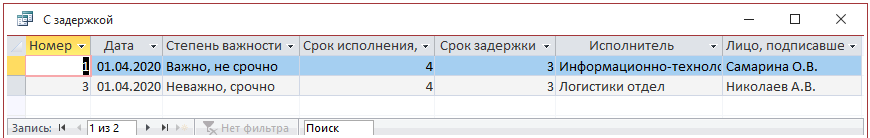


Рисунок 12 – Результат запроса на получение поручений, выполненных с задержкой

6. На основании полученных запросов определим структуру выполненных и выполняемых поручений:

SELECT "Выполняющиеся с задержкой" AS Статус,Count(Выполняющиеся.[Срок задержки]) AS [Число поручений], ROUND(Count(Выполняющиеся.[Срок задержки]) / (SELECT Count(\*) FROM [Информация о поручениях]) \* 100,2) AS [Доля, %]

FROM Выполняющиеся

WHERE Выполняющиеся.[Срок задержки]>0

UNION ALL

SELECT "Выполняющиеся без задержки" AS Статус,Count(Выполняющиеся.[Срок задержки]) AS [Число поручений], ROUND(Count(Выполняющиеся.[Срок задержки])/ (SELECT Count(\*) FROM [Информация о поручениях]) \* 100,2) AS [Доля, %]

FROM Выполняющиеся

WHERE Выполняющиеся.[Срок задержки]=0

UNION ALL

SELECT "Выполненные с задержкой" AS Статус,Count(\*) AS [Число поручений], ROUND(Count(\*)/ (SELECT Count(\*) FROM [Информация о поручениях]) \* 100,2) AS [Доля, %]

FROM [C задержкой]

UNION ALL

SELECT "Выполненные без задержки" AS Статус,Count([Информация о поручениях].[Срок задержки]) AS [Число поручений], ROUND(Count([Информация о поручениях].[Срок задержки])/ (SELECT Count(\*) FROM [Информация о поручениях]) \* 100,2) AS [Доля, %]

FROM [Информация о поручениях]

WHERE [Информация о поручениях].[Срок задержки]=0

Результат запроса приведен ниже (рис. 13).

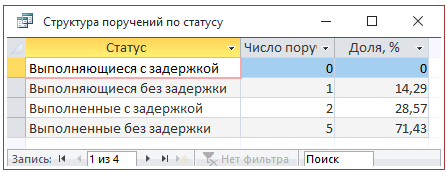


Рисунок 13 – Результат запроса на получение структуры поручений по статусу

7. Определим поручения, выданные определенным подписантом:

SELECT [Информация о поручениях].Номер, [Информация о поручениях].Дата, [Информация о поручениях].[Степень важности], [Информация о поручениях].[Срок исполнения, в днях], [Информация о поручениях].Статус, [Информация о поручениях].[Срок задержки], [Информация о поручениях].Исполнитель, [Информация о поручениях].[Лицо, подписавшее документ]

FROM [Информация о поручениях]

WHERE [Информация о поручениях].[Лицо, подписавшее документ] Like "\*" & [Введите часть фамилии] & "\*";

Результат запроса для поиска по фильтру «нико» представлен ниже (рис. 14).

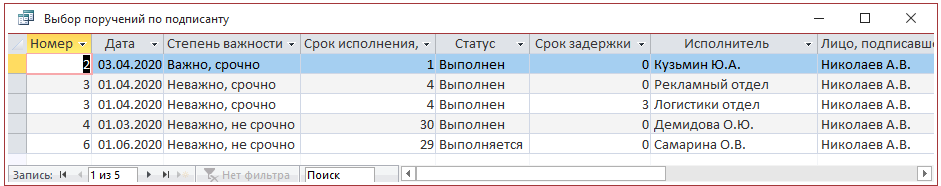


Рисунок 14 – Результат запроса на получение поручений заданного подписанта

8. Определим поручения, выданные определенному исполнителю:

SELECT [Информация о поручениях].Номер, [Информация о поручениях].Дата, [Информация о поручениях].[Степень важности], [Информация о поручениях].[Срок исполнения, в днях], [Информация о поручениях].Статус, [Информация о поручениях].[Срок задержки], [Информация о поручениях].Исполнитель, [Информация о поручениях].[Лицо, подписавшее документ]

FROM [Информация о поручениях]

WHERE [Информация о поручениях].Исполнитель Like "\*" & [Введите часть фамилии или названия отдела исполнителя] & "\*";

Результат запроса для поиска по фильтру «ам» представлен ниже (рис. 15).

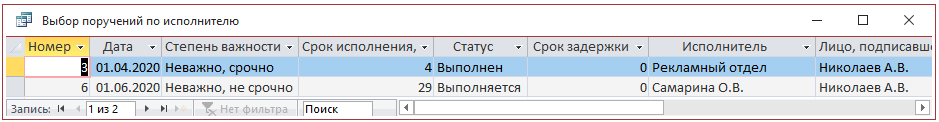


Рисунок 15 – Результат запроса на получение поручений заданному исполнителю

9. Определим поручения, выданные за определенный период:

SELECT [Информация о поручениях].Номер, [Информация о поручениях].Дата, [Информация о поручениях].[Степень важности], [Информация о поручениях].[Срок исполнения, в днях], [Информация о поручениях].Статус, [Информация о поручениях].[Срок задержки], [Информация о поручениях].Исполнитель, [Информация о поручениях].[Лицо, подписавшее документ]

FROM [Информация о поручениях]

WHERE [Информация о поручениях].Дата Between CDate([Введите начальную дату]) And CDate([Введите конечную дату]);

Результат запроса за период с 01.01.2020 по 30.03.2020 представлен ниже (рис. 16).

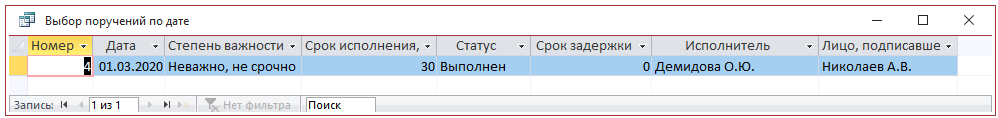


Рисунок 16 – Результат запроса на получения поручений за период

10. Определим поручения, подписанные определенным подписантом или выданные определенным исполнителем за определенный период:

SELECT [Информация о поручениях].Номер, [Информация о поручениях].Дата, [Информация о поручениях].[Степень важности], [Информация о поручениях].[Срок исполнения, в днях], [Информация о поручениях].Статус, [Информация о поручениях].[Срок задержки], [Информация о поручениях].Исполнитель, [Информация о поручениях].[Лицо, подписавшее документ]

FROM [Информация о поручениях]

WHERE ((([Информация о поручениях].Дата) Between CDate([Введите начальную дату]) And CDate([Введите конечную дату])) AND (([Информация о поручениях].Исполнитель) Like "\*" & [Введите часть фамилии или названия отдела исполнителя] & "\*") OR (([Информация о поручениях].[Лицо, подписавшее документ]) Like "\*" & [Введите часть фамилии] & "\*"));

Результат запроса для поиска исполнителей и подписантов по фильтру «ам» за период с 01.01.2020 по 01.05.2020 представлен ниже (рис. 17).

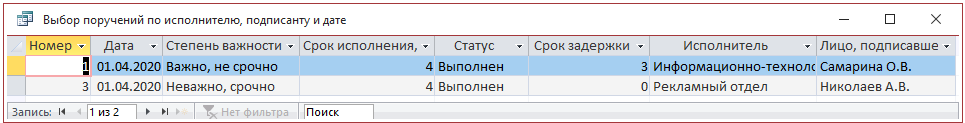


Рисунок 17 – Результат запроса на получения поручений за период

## 3.3 Интерфейс пользователя

Описать формы для ввода информации, главную кнопочную форму, форму для формирования отчетов, формы для поиска информации, привести скриншоты окон программы. Указать задачи БД, которые можно решать, используя данные интерфейсы.

Для всех таблиц и отчетов создадим формы на основе автоформы в столбец, используя «Мастер форм».

Для запуска БД требуется наличие MS Access. Программы запускается двойным кликом по значку с базой. В результате открывается окно программы (рис. 18).

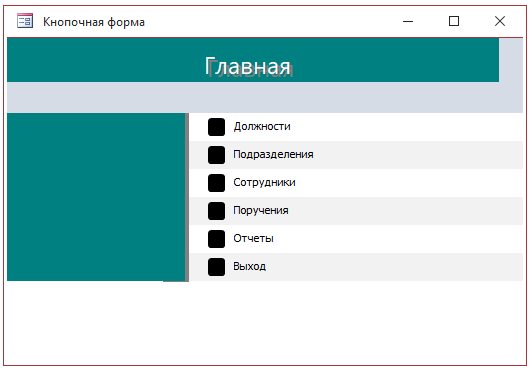


Рисунок 18 – Главное окно

Команды меню позволяют осуществлять быстрый переход между формами и получать отчеты (меню «Отчеты»).

Так при клике на кнопку «Должности» открывается форма добавления должностей (рис. 19).

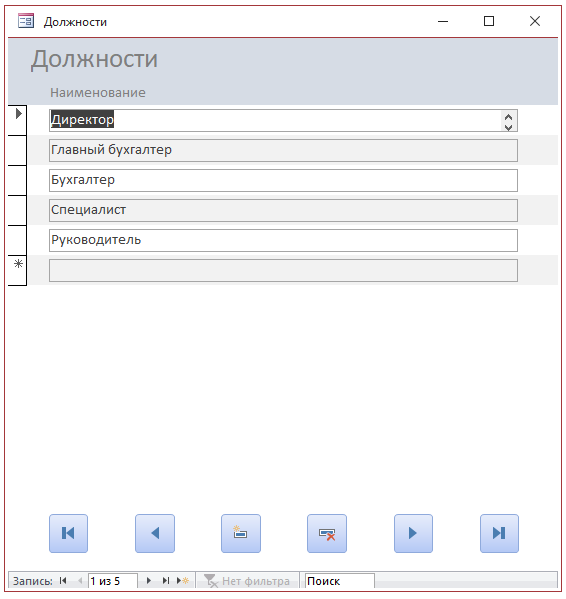


Рисунок 19 – Окно «Должности»

Аналогичный вид имеет форма «Подразделения».

При клике на кнопку «Сотрудники», откроется окно изменения и просмотра списка сотрудников (рис. 20).

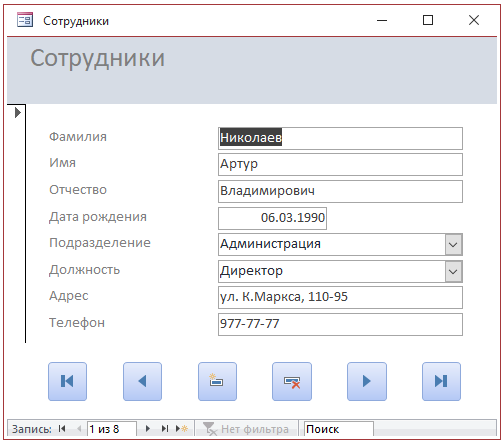


Рисунок 20 – Окно «Сотрудники»

Также можно просмотреть и изменить список поручений (рис. 21).

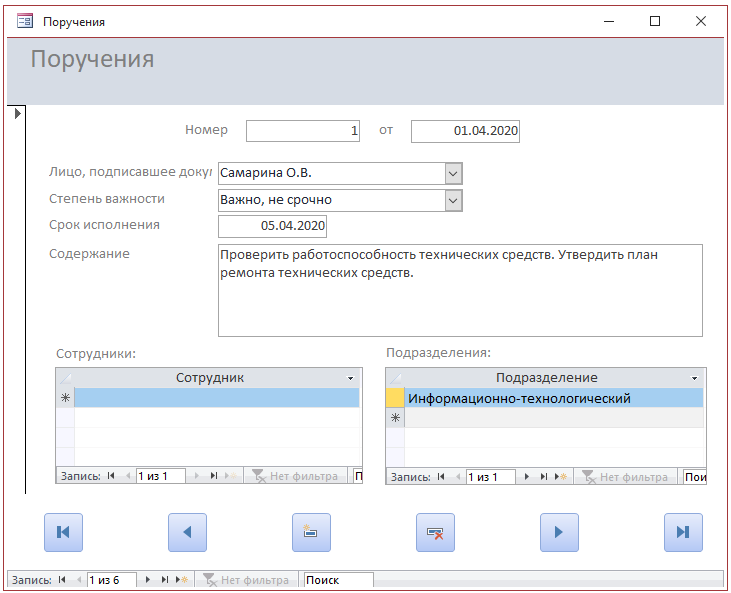


Рисунок 21 – Окно «Поручения»

Для каждого поручения можно выбрать несколько подразделений и сотрудников. Кнопки внизу форму позволяют быстро осуществлять навигацию по форме, а также добавлять и удалять данные.

В меню «Отчеты» представляется собой две группы отчетов (рис. 22), каждая из которых состоит из списка отчетов, доступных для пользователя (рис. 23-24).

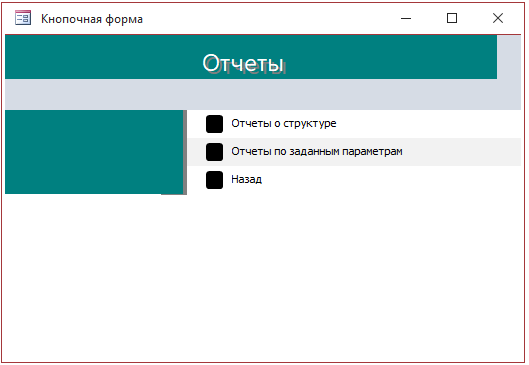


Рисунок 22 – Меню «Отчеты»

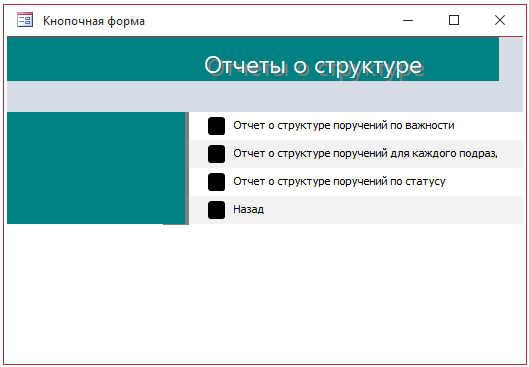


Рисунок 23 – Меню «Отчеты о структуре»

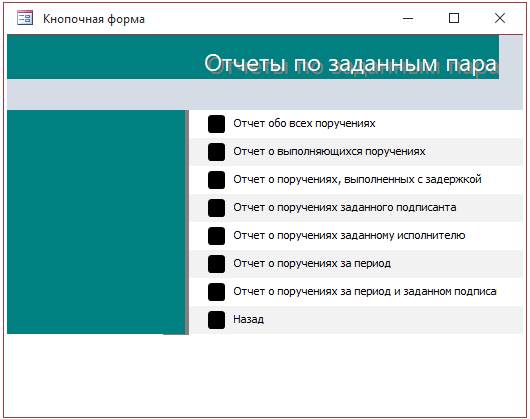


Рисунок 24 – Меню «Отчеты по заданным параметрам»

Отчеты, приведенные в данных формах представлены в приложении А.

Таким образом, созданная база позволяет достаточно просто и быстро осуществлять учет и ведение внутреннего документооборота предприятия, что снижает нагрузку на секретариат, позволяет сэкономить время на передачу информации и средства на копирование и передачу приказов.

# ЗАКЛЮЧЕНИЕ

В современном мире нам приходится сталкиваться с огромным количеством информации, которую необходимо запоминать или где-то хранить. Так как человеческий мозг не может справиться с такой задачей на помощь приходят компьютеры, где самая разнообразная информация может храниться в разных форматах.

Так как на сегодняшний день практически все работающие базы данных соответствуют реляционной модели, наиболее целесообразным является создание именно данного типа базы данных.

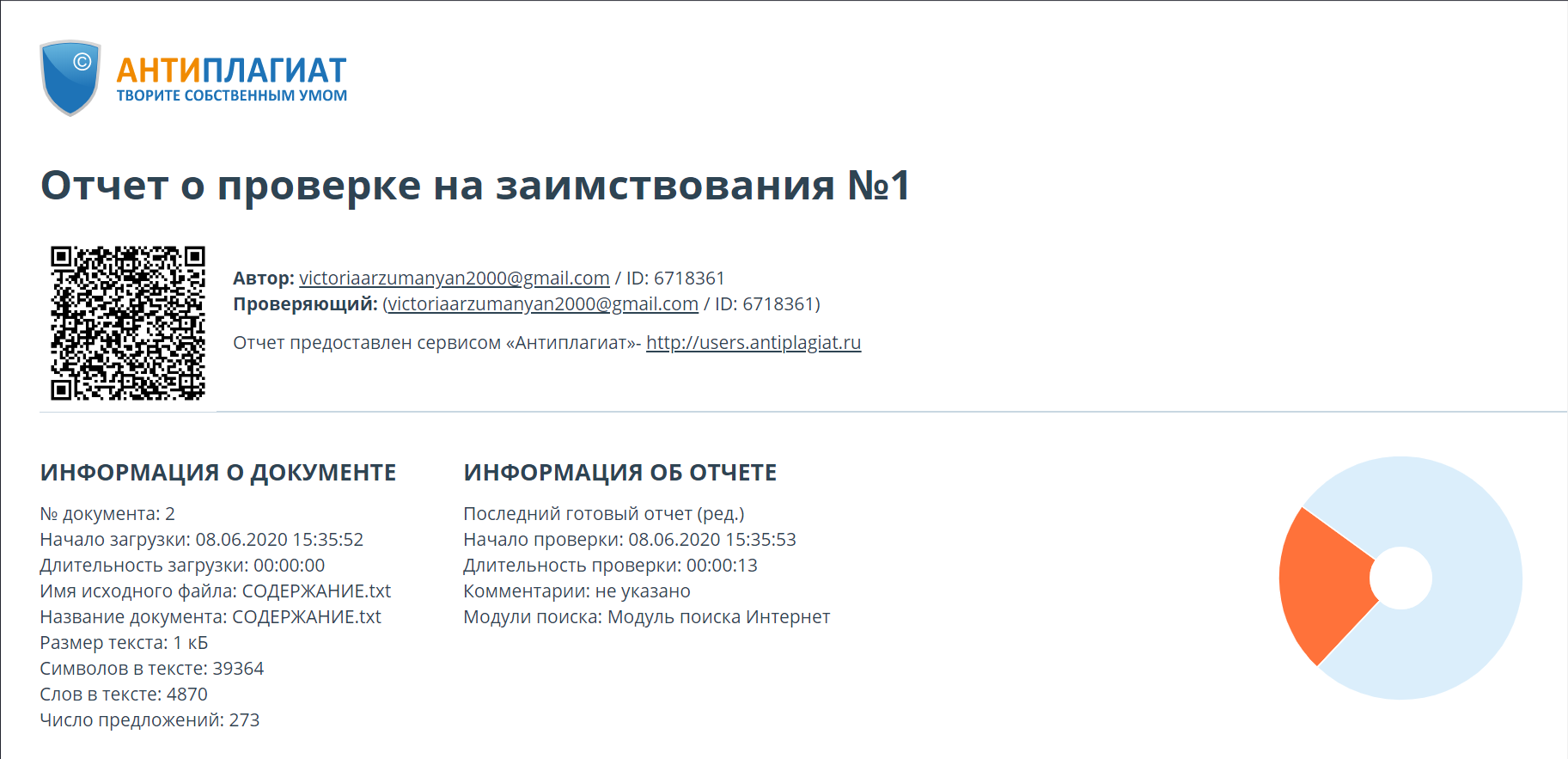
В процессе выполнения курсовой работы была разработана локальная база данных для решения задачи учета внутреннего документооборота. База данных обладает минимальной избыточностью, все таблицы находятся в третьей нормальной форме, отсутствуют аномалии добавления, исключения, модификации.

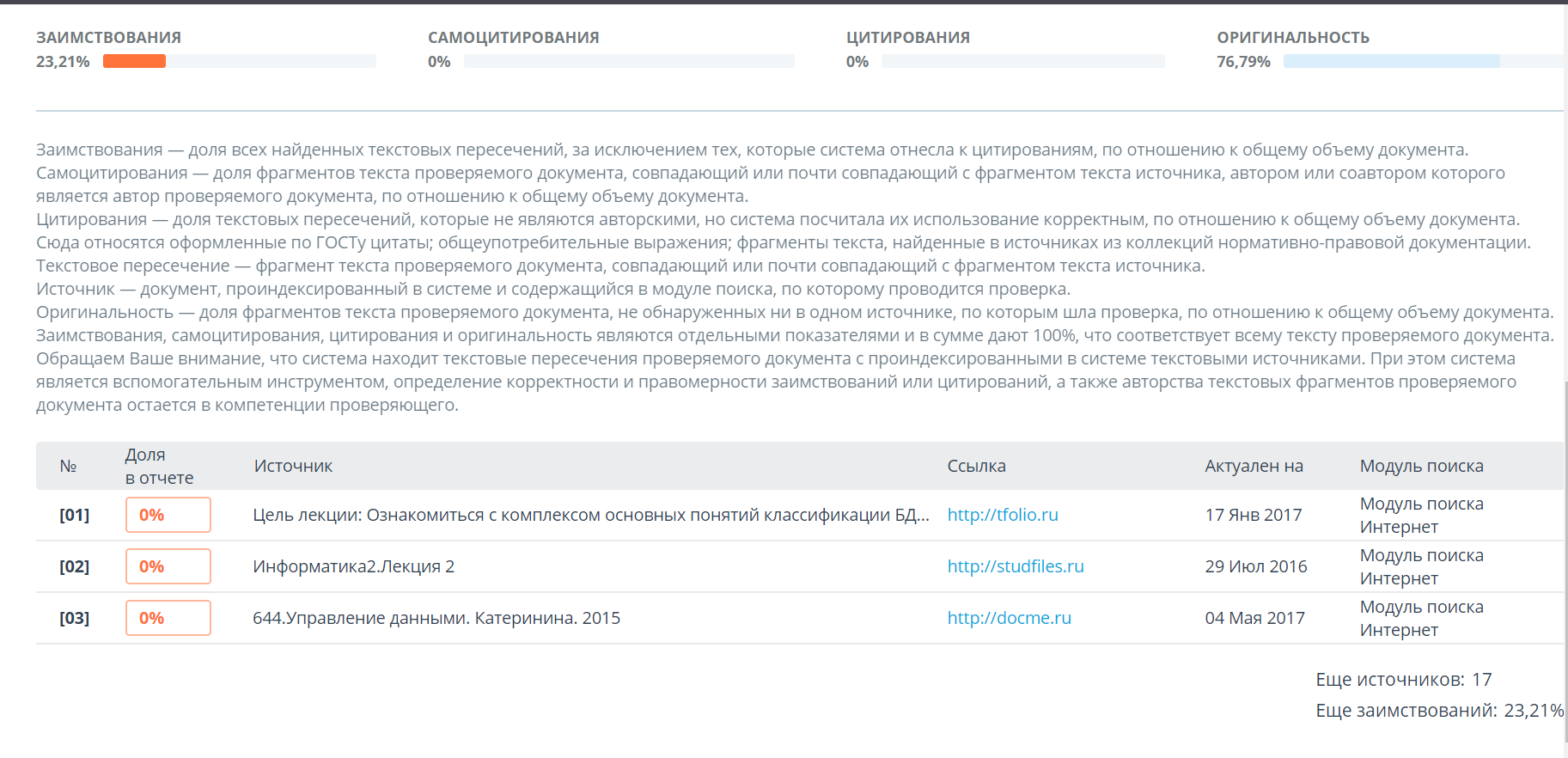
В работе была разработана даталогическая модель базы данных средствами Microsoft Access, включающая 7 таблиц и связи между ними. Также были разработаны 10 запросов для отображения данных из таблиц.

Таким образом, разработанная база данных может применять для учета и автоматизации внутреннего документооборота на предприятиях.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ И ЛИТЕРАТУРЫ

1. Гурвиц Г.А. Microsoft Access 2010. Разработка приложений на реальном примере. – СПб.: БХВ-Петербург, 2010. – 484 с.
2. Илюшечкин, В. М. Основы использования и проектирования баз данных: учебник для СПО. – М.: Юрайт, 2017. – 213 с
3. Кузин А.В., Левонисова С.В. Базы данных: учебное пособие для вузов. – М.: Академия, 2016. – 315 с.
4. Кумсков М.И. Базы данных и процессы их создания. – М.: МГУ, 2014. – 136 с.
5. Маркин А.В. Программирование на SQL в 2 ч. Часть 1: учебник и практикум для бакалавриата и магистратуры / А. В. Маркин. – М.: Юрайт, 2017. – 362 с.
6. Маркин, А. В. Программирование на SQL в 2 ч. Часть 2: учебник и практикум для бакалавриата и магистратуры / А. В. Маркин. – М: Юрайт, 2017. – 292 с.
7. Ролланд Ф.Д. Основные концепции баз данных.: Пер с англ. – М.: Вильямс, 2013 – 256 с.
8. Сирант О.В., Коваленко Т.А. Работа с базами данных. – М.: ИНТУИТ, 2016. –150 с.
9. Советов Б.Я. Базы данных: теория и практика: Учебник для бакалавров / Б.Я. Советов, В.В. Цехановский, В.Д. Чертовской. – М.: Юрайт, 2017. – 463 c.
10. Сычев А. Н. ЭВМ и периферийные устройства: учеб. пособие / А. Н. Сычев. – Томск: ФДО, ТУСУР, 2016. – 113 с.
11. Таненбаум Э. Компьютерные сети. – СПб.: Питер, 2017 – 992 с.
12. Фаронов В.В. Delphi. Программирование на языке высокого уровня: учебник для вузов. – СПб.: Питер, 2003. – 639 с.
13. Фуфаев Э.В. Базы данных: Учебное пособие для студентов учреждений среднего профессионального образования / Э.В. Фуфаев, Д.Э. Фуфаев. – М.: ИЦ Академия, 2012. – 320 c.
14. Хомоненко А.Д., Цыганков В.М., Мальцев М.Г. Базы Данных. – СПб.: КОРОНА принт, 2012 – 672 с.
15. Цехановский В.В., Чертовской В.Д. Управление данными: учебник для вузов – СПб.: Лань, 2015. - 432 с.
16. Шустова Л.И., Тараканов О.В. Базы данных: Учебник. – М.: ИНФРА-М, 2016. – 304 с.
17. Большой энциклопедический словарь / Ред. А.М. Прохоров. – М.: Большая Российская энциклопедия, 2015. – 1456 с.
18. Грей Дж. Управление данными: Прошлое, Настоящее и Будущее. [Электронный ресурс]. – Режим доступа: citforum.ru/database/classics/gray/
19. Поддержка Microsoft Office. [Электронный ресурс]. – Режим доступа: <https://support.office.com/ru-ru/>





# ПРИЛОЖЕНИЯ

Приложение А. Отчеты

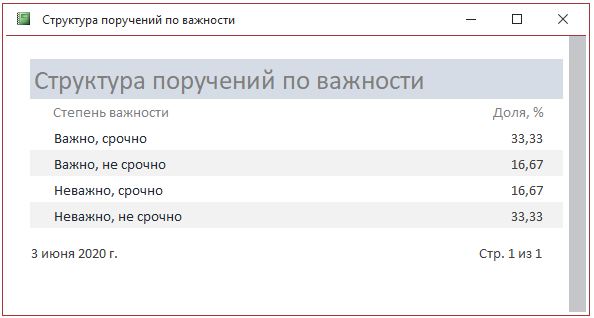


Рисунок А.1 – Отчет о структуре поручений по важности

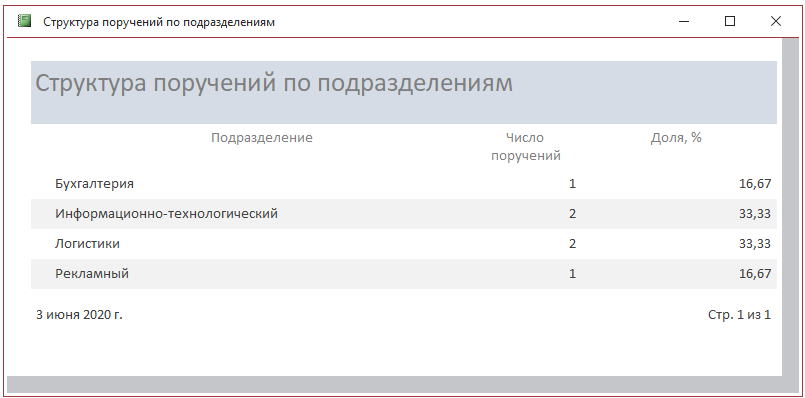


Рисунок А.2 – Отчет о структуре поручений для каждого подразделения

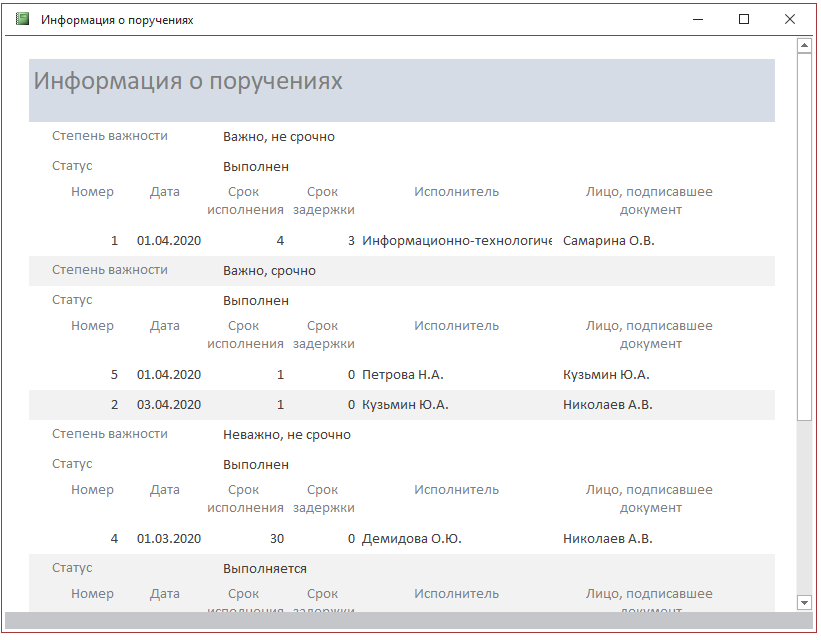


Рисунок А.3 – Отчет обо всех поручениях

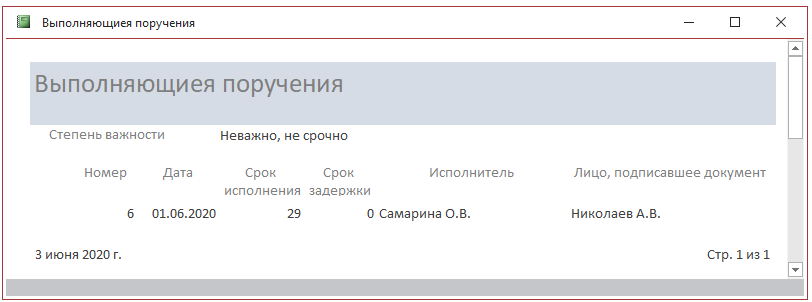


Рисунок А.4 – Отчет о выполняющихся поручениях

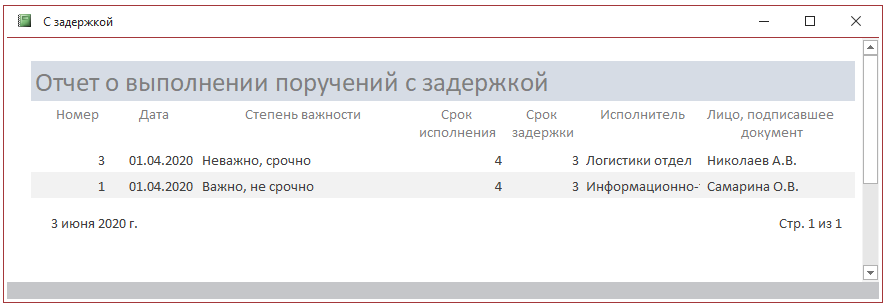


Рисунок А.5 – Отчет о поручениях, выполненных с задержкой

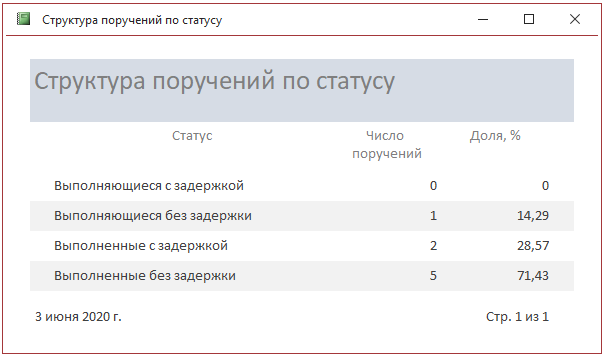


Рисунок А.6 – Отчет о структуре поручений по статусу

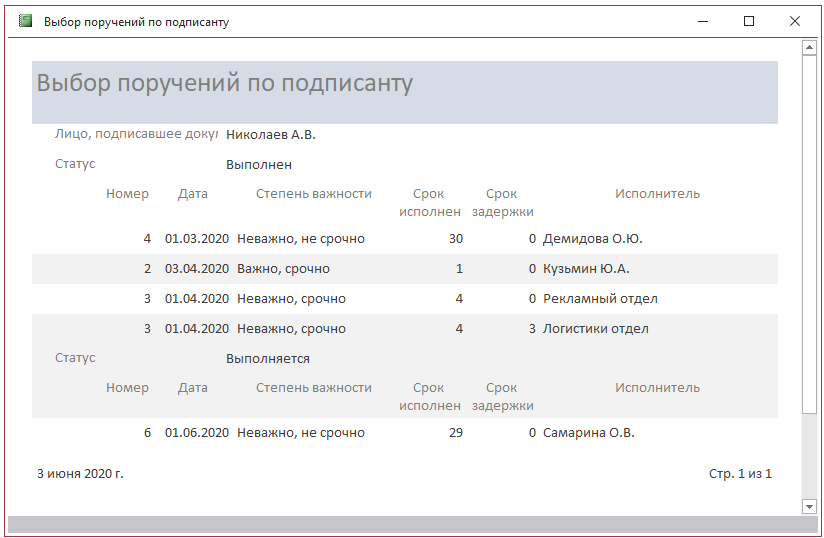


Рисунок А.7 – Отчет о поручениях заданного подписанта

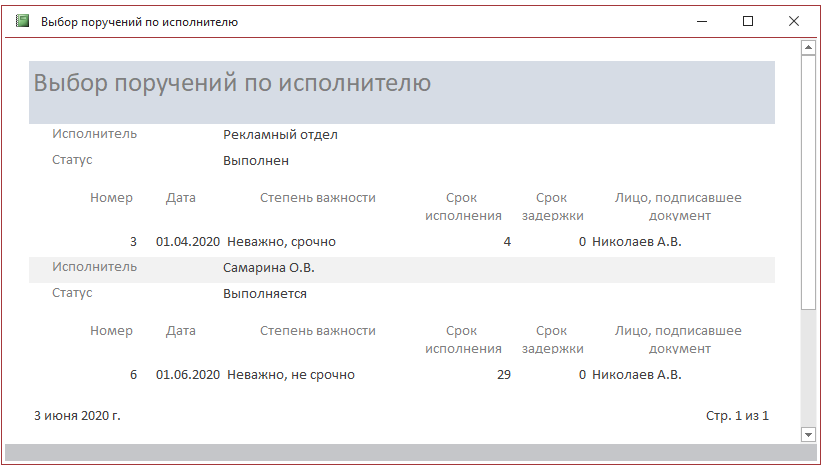


Рисунок А.8 – Отчет о поручениях заданному исполнителю

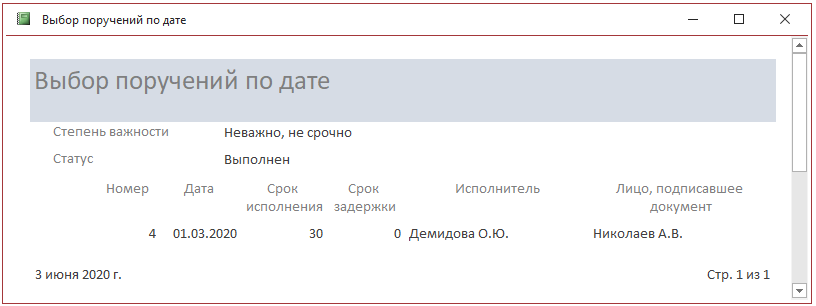


Рисунок А.9 – Отчет о поручениях за период

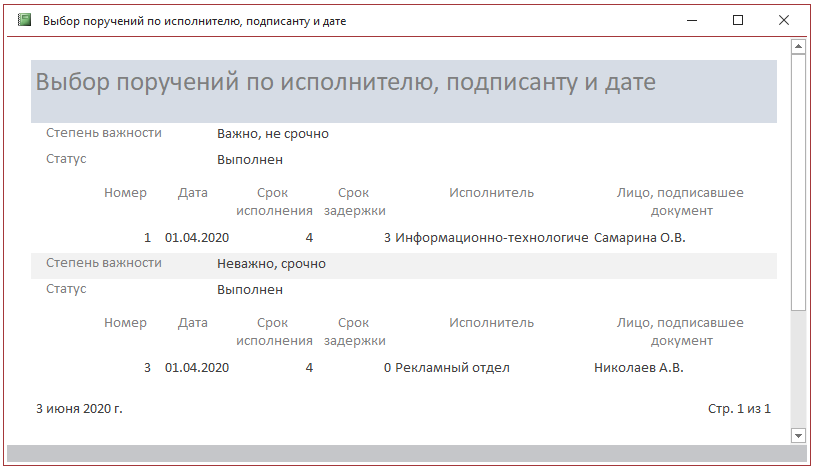


Рисунок А.10 – Отчет о поручениях за период и заданном подписанту и исполнителю