ОГЛАВЛЕНИЕ

[Введение 2](#_Toc72421495)

[1 Логическая структура базы данных 3](#_Toc72421496)

[2 Нормализация отношений 3](#_Toc72421497)

[3. Физическая структура базы данных 5](#_Toc72421498)

[3.3 Создание таблиц в базе данных 5](#_Toc72421499)

[3.4 Индексы 8](#_Toc72421500)

[4 Создание представлений (готовых запросов) 8](#_Toc72421501)

[Заключение 10](#_Toc72421502)

[Список использованных источников 11](#_Toc72421503)

# ВВЕДЕНИЕ

# 1. Теоретическая часть – проектирование базы данных

## 1.1 Построение логической структуры базы данных;

## 1.2 Нормализация отношений

(для этого необходимо ввести ограничения, определить функциональные зависимости и ключи;

## 1.3 Приведение каждого отношения к третьей нормальной форме

o конечный и промежуточные этапы представить в виде ER-диаграммах в одной из следующих нотаций: Чена, UML или нотации IDEF1X)

# 2 Практическая часть – реализация базы данных в выбранной СУБД

помимо запросов на создание таблиц, индексов, представлений необходимо добавить и обосновать использование триггеров и функций в представлениях продемонстрировать внешнее и внутреннее соединение (JOIN) и сформулировать выбранные запросы в терминах реляционной алгебры, а также продемонстрировать правильность их обработки на примерах

# 3

# ВВЕДЕНИЕ

Тема курсовой работы: Информационная система «Фирма 12»

В качестве предметной области проектирования БД выбрана торговая фирма, которая осуществляет выполнение заказов, поступающих от клиентов.

Цель курсового проектирования – применить на практике знания, полученные в процессе изучения дисциплины "Базы данных", получить теоретические знания и практические навыки реализации баз данных. Осуществить анализ предметной области. Освоить концептуальное проектирование и научиться определять сущности и атрибуты БД. Научиться разрабатывать инфологическую модель БД в виде ER-диаграмм. Получить теоретические знания и практические навыки при физическом проектировании баз данных (БД). Научиться создавать даталогическую модель БД.

В качестве исходных данных предоставлены объекты и атрибуты описывающие торговую фирму, клиентов, поставщиков и т.д.

# 1 ИНФОЛОГИЧЕСКАЯ И ДАТАЛОГИЧЕСКАЯ СТРУКТУРА БАЗЫ ДАННЫХ, НОРМАЛИЗАЦИЯ ОТНОШЕНИЙ

Задание предлагает следующие объекты и атрибуты для создания базы данных торговой фирмы:

1. Фирма (название, юр. адрес, телефон, №счета в банке наименование товара, цена товара)
2. Склад (№склада, адрес склада, №заказа, наименование товара, единицы измерения, кол-во, цена, отметки об оплате)
3. Кадры (Название фирмы, ФИО, должность, образование, оклад, дата принятия на работу)
4. Поставщик (№поставщика, ФИО, юр. Адрес, №счета в банке, наименование товара, цена товара)
5. Транспорт (№склада, №автомобиля, марка автомобиля, грузоподъемность, год выпуска, ФИО владельца, дата перевозки)
6. Товар (Наименование, единицы измерения, сорт, цена, изготовитель, адрес изготовителя)
7. Клиент (№ клиента, ФИО клиента, юр. адрес клиента, №счета в банке, № заказа, наименование товара, цена товара)
8. Накладная (№ накладной, наименование товара, единицы измерения, количество, цена, сумма)
9. Счет (Наименование банка, №счета, ФИО, состояние счета, дата, наличие кредита, сумма кредита)
10. Заказ (№ заказа, дата заказа, ФИО, №счета, наименование товара, количество, цена товара).

В исходных данных видно множественное дублирование одних и тех же полей, таких как № счета, наименование товара, единица измерения и пр. Для исключения дублирования полей, облегчения управления записями в ходе нормализации необходимо выделить сущности так, чтобы если все его атрибуты были атомарны, то есть если ни один из его атрибутов нельзя было разделить на более простые атрибуты.

Обратим внимание на то, что клиент, поставщик и фирма имеют счет в банке. Выделим счет отдельной сущностью со следующими атрибутами:

IDSCHET уникальный идентификатор счета INT

BANK наименование банка VARCHAR

NUMSCHET номер счета в банке VARCHAR

STATUS состояние счета INT

DATA дата DATE

KREDIT наличие кредита INT

SUMMA сумма кредита FLOAT

Таким образом, сущности клиента, поставщика и фирмы будут ссылаться на счет по его уникальному идентификатору.

Таблица описывающая реквизиты фирмы(FIRMA):

IDFIRMA Идентификатор INT

NAZVAN Название VARCHAR

URADR Юридический адрес VARCHAR

PHONE Телефон VARCHAR

IDSCHET Идентификатор счета INT

Таблица клиентов фирмы(CLIENT):

IDKLIENT Идентификатор INT

FIO ФИО VARCHAR

URADR Юридический адрес VARCHAR

IDSCHET Счет в банке INT

Таблица описывающая поставщиков(POSTAV):

IDPOSTAV Идентификатор INT

FIO ФИО VARCHAR

URADR Юридический адрес VARCHAR

IDSCHET Счет INT

Таблица описывающая склады(SKLAD):

NSKLAD Идентификатор ` INT

ADRESS Адрес склада VARCHAR

NAIMEN Наименование VARCHAR

В товарах нам необходимо указать место хранения товара (склад) и его поставщика. Для исключения дублирования полей поставщиков и складов, организуем ссылку на склад и поставщика по их идентификатору.

Таблица описывающая товары(TOVAR):

IDTOVAR Идентификатор INT

NAIMEN\_TOVARA Наименование` VARCHAR

ED\_IZM Ед. измерения VARCHAR

SORT Сорт товара INT

CENA Цена FLOAT

IDPOSTAV Поставщик INT

SKLAD Склад INT

KOLVO Количество INT

В таблице заказов заказанный товар и клиент его заказавший так же будут ссылаться на соответствующие таблицы.

Таблица описывающая заказы(ZAKAZ):

IDZAKAZ Идентификатор INT

DATA Дата DATE

ID\_KLIENT Клиент INT

ID\_TOVAR Товар INT

KOL\_VO\_ZAK\_TOV Количество INT

CENA\_TOVARA Цена товара FLOAT

SUM\_ZAKAZA Сумма заказа FLOAT

Таблица накладных ссылается на таблицы товаров и клиентов.

Таблица описывающая накладные отпуска товаров(NAKLADNAYA):

IDNAKLADNAYA Идентификатор INT

IDTOVAR Товар INT

KOLVO Количество INT

SUMMA Сумма FLOAT

IDCLIENT Клиент INT

В таблице кадров для выплаты зарплаты на счет можно организовать ссылку на банковский счет работника.

Таблица описывающая кадры(KADRY):

IDKADRY Идентификатор INT

FIO ФИО VARCHAR

DOLGZNOST Должность VARCHAR

OBRAZOVANIE Образование VARCHAR

ZARPLATA Зарплата FLOAT

DATA\_PRINYATIA Дата приема DATE

IDSCHET Счет зарплатный INT

Транспорт приписан к конкретному складу, и ссылается на него по идентификатору.

Таблица описывающая транспорт(TRANSPORT):

ID\_TRANSPORT Идентификатор INT

NOM\_AUTO Номер авто VARCHAR

MARKA\_AUTO Марка авто VARCHAR

TONNAG Тоннаж FLOAT

GOD\_VIPUSKA Год выпуска INT

VLADELEC Владелец VARCHAR

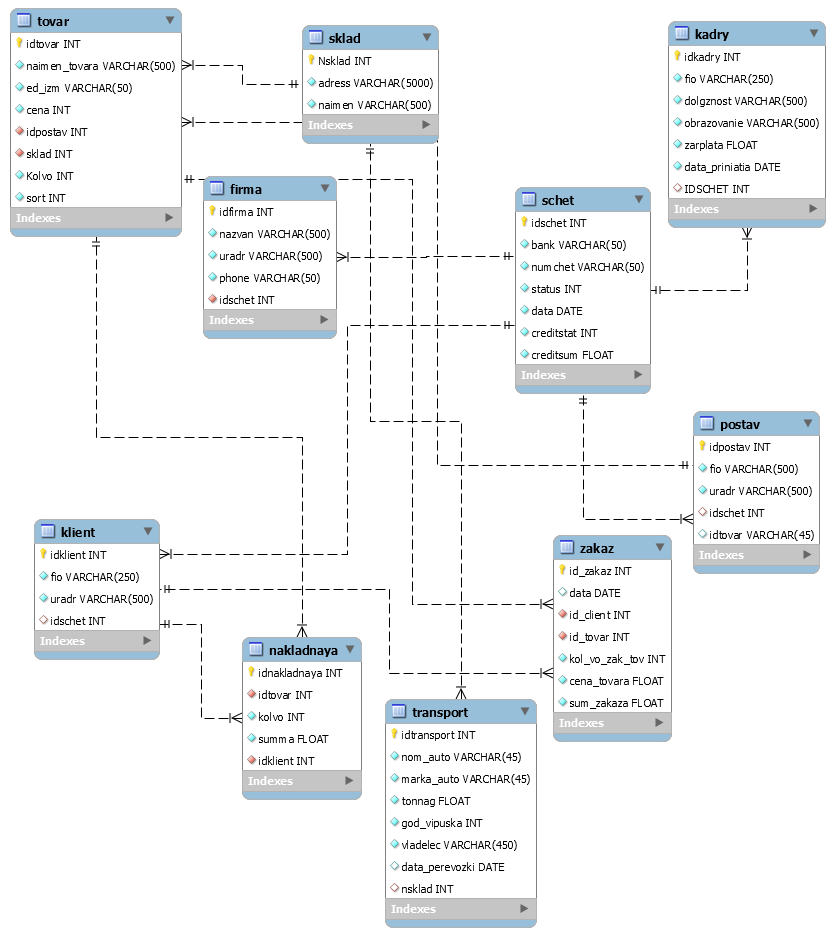
DATA\_PEREVOZKI Дата перевозки DATE

NSKLAD Склад INT

Для обеспечения целостности базы данных создадим внешние ключи. В этом случае мы наложим ограничения целостности базы с тем, чтобы исключить возможность появления несогласованных данных.

Для Сlient, Postav, Firma укажем внешний ключ IDSCHET ссылающийся на таблицу SCHET. Таким образом, мы свяжем клиентов, поставщиков и фирму с соответствующими им счетами.

Составим ER-диаграмму базы данных.



# 2. ФИЗИЧЕСКАЯ СТРУКТУРА БАЗЫ ДАННЫХ

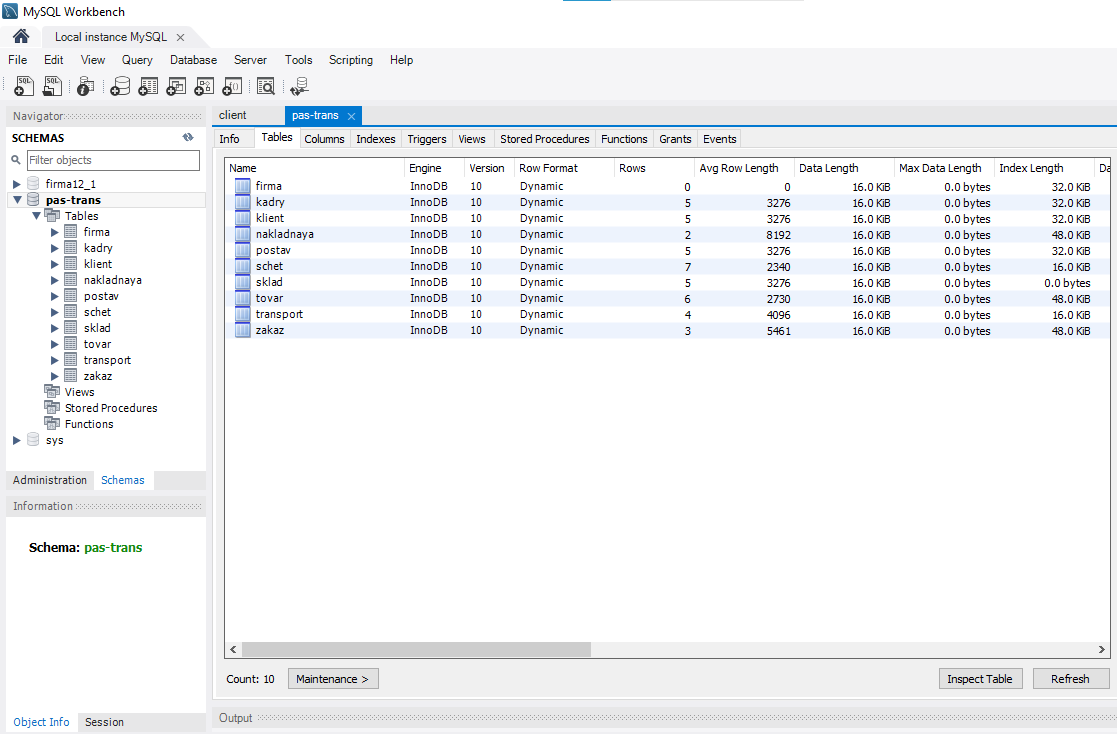
## 2.1 Создание таблиц в базе данных

Для создания таблиц используется DDL (Data Definition Language), для таблиц помимо указания имени таблицы необходимо указать так же имена полей, их тип и длину, ограничения целостности. Индексные и ключевые поля можно указать непосредственно в момент создания, либо добавить позже, при необходимости. Укажем для каждой таблицы так же краткий комментарий описывающий её содержимое.

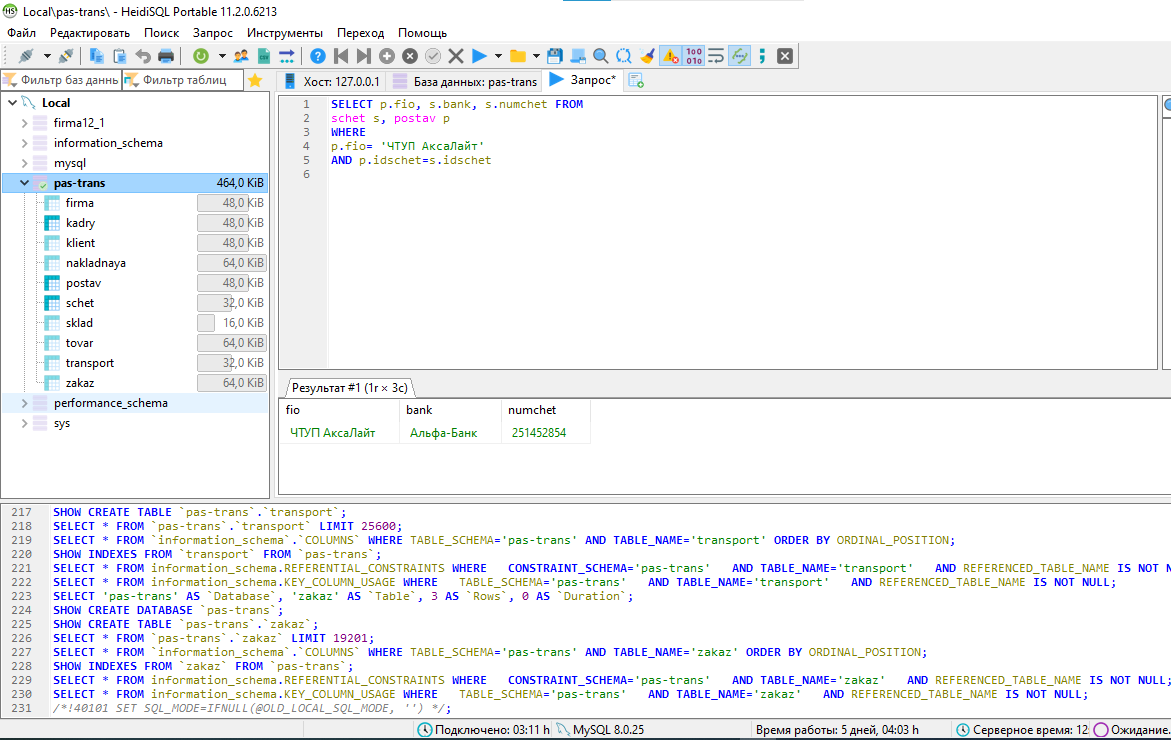
Для создания базы данных и таблиц в ней можно использовать графические инструменты поставляемые с выбраной базой данных, или командно-строковую оболочку, позволяющие создавать объекты базы и манипулировать ими путем команд.

Пример графической оболочки для создания и управления базой данных и её объектами.

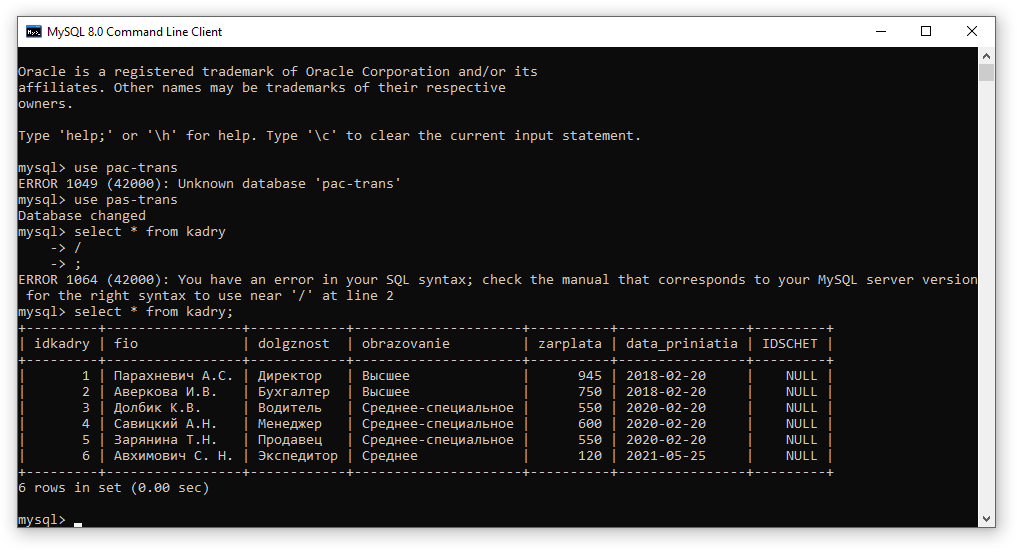
MySQLWorkbench



Графическая оболочка HeidiSQL



Командно-строковая версия оболочки



Создание таблиц с использованием DDL

CREATE TABLE IF NOT EXISTS `firma` (

`idfirma` int NOT NULL AUTO\_INCREMENT,

`nazvan` varchar(500) NOT NULL DEFAULT '',

`uradr` varchar(500) NOT NULL DEFAULT '',

`phone` varchar(50) NOT NULL DEFAULT '',

`idschet` int NOT NULL,

PRIMARY KEY (`idfirma`),

UNIQUE KEY `idfirma` (`idfirma`),

KEY `FKschetf` (`idschet`),

CONSTRAINT `FKschetf` FOREIGN KEY (`idschet`) REFERENCES `schet` (`idschet`)

) ENGINE=InnoDB AUTO\_INCREMENT=2 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci COMMENT='Описание фирмы';

CREATE TABLE IF NOT EXISTS `kadry` (

`idkadry` int NOT NULL AUTO\_INCREMENT,

`fio` varchar(250) NOT NULL,

`dolgznost` varchar(500) NOT NULL,

`obrazovanie` varchar(500) NOT NULL,

`zarplata` float NOT NULL,

`data\_priniatia` date NOT NULL,

`IDSCHET` int DEFAULT NULL,

PRIMARY KEY (`idkadry`),

UNIQUE KEY `idkadr` (`idkadry`),

KEY `FK\_kadry\_schet` (`IDSCHET`),

CONSTRAINT `FK\_kadry\_schet` FOREIGN KEY (`IDSCHET`) REFERENCES `schet` (`idschet`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci COMMENT='Сотрудники фирмы';

CREATE TABLE IF NOT EXISTS `klient` (

`idklient` int NOT NULL AUTO\_INCREMENT,

`fio` varchar(250) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL DEFAULT 'Не указано',

`uradr` varchar(500) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL DEFAULT 'Не указано',

`idschet` int DEFAULT NULL,

PRIMARY KEY (`idklient`),

UNIQUE KEY `idclient` (`idklient`),

KEY `FKschet` (`idschet`),

CONSTRAINT `FKschet` FOREIGN KEY (`idschet`) REFERENCES `schet` (`idschet`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

CREATE TABLE IF NOT EXISTS `nakladnaya` (

`idnakladnaya` int NOT NULL AUTO\_INCREMENT,

`idtovar` int NOT NULL,

`kolvo` int NOT NULL,

`summa` float NOT NULL,

`idklient` int NOT NULL,

PRIMARY KEY (`idnakladnaya`),

UNIQUE KEY `idnaklad` (`idnakladnaya`),

KEY `FKtovarn` (`idtovar`),

KEY `FK\_nakladnaya\_klient` (`idklient`),

CONSTRAINT `FK\_nakladnaya\_klient` FOREIGN KEY (`idklient`) REFERENCES `klient` (`idklient`),

CONSTRAINT `FKtovarn` FOREIGN KEY (`idtovar`) REFERENCES `tovar` (`idtovar`)

) ENGINE=InnoDB AUTO\_INCREMENT=3 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci COMMENT='Накладные на товар';

CREATE TABLE IF NOT EXISTS `postav` (

`idpostav` int NOT NULL AUTO\_INCREMENT,

`fio` varchar(500) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL DEFAULT 'Не указано',

`uradr` varchar(500) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL DEFAULT 'Не указано',

`idschet` int DEFAULT NULL,

`idtovar` varchar(45) DEFAULT NULL,

PRIMARY KEY (`idpostav`),

UNIQUE KEY `idpostav` (`idpostav`),

KEY `FKschetp` (`idschet`),

CONSTRAINT `FKschetp` FOREIGN KEY (`idschet`) REFERENCES `schet` (`idschet`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci COMMENT='Поставщики';

CREATE TABLE IF NOT EXISTS `schet` (

`idschet` int NOT NULL AUTO\_INCREMENT,

`bank` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL DEFAULT 'Не указано',

`numchet` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL,

`status` int NOT NULL DEFAULT '0',

`data` date NOT NULL,

`creditstat` int NOT NULL DEFAULT '0',

`creditsum` float NOT NULL,

PRIMARY KEY (`idschet`),

UNIQUE KEY `idschet` (`idschet`)

) ENGINE=InnoDB AUTO\_INCREMENT=8 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

CREATE TABLE IF NOT EXISTS `sklad` (

`Nsklad` int unsigned NOT NULL AUTO\_INCREMENT,

`adress` varchar(5000) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL,

`naimen` varchar(500) NOT NULL,

PRIMARY KEY (`Nsklad`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

CREATE TABLE IF NOT EXISTS `tovar` (

`idtovar` int NOT NULL AUTO\_INCREMENT,

`naimen\_tovara` varchar(500) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL DEFAULT 'Не указано',

`ed\_izm` varchar(50) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL DEFAULT 'Не указано',

`cena` int NOT NULL,

`idpostav` int NOT NULL,

`sklad` int unsigned NOT NULL,

`Kolvo` int NOT NULL,

`sort` int NOT NULL,

PRIMARY KEY (`idtovar`),

UNIQUE KEY `idtovar` (`idtovar`),

KEY `FKpostavt` (`idpostav`),

KEY `FK\_tovar\_sklad` (`sklad`),

CONSTRAINT `FK\_tovar\_sklad` FOREIGN KEY (`sklad`) REFERENCES `sklad` (`Nsklad`),

CONSTRAINT `FKpostavt` FOREIGN KEY (`idpostav`) REFERENCES `postav` (`idpostav`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci COMMENT='Список товаров нормализованный';

CREATE TABLE IF NOT EXISTS `transport` (

`idtransport` int NOT NULL AUTO\_INCREMENT,

`nom\_auto` varchar(45) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL,

`marka\_auto` varchar(45) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL,

`tonnag` float NOT NULL,

`god\_vipuska` int NOT NULL,

`vladelec` varchar(450) CHARACTER SET utf8mb4 COLLATE utf8mb4\_0900\_ai\_ci NOT NULL,

`data\_perevozki` date DEFAULT NULL,

`nsklad` int unsigned DEFAULT NULL,

PRIMARY KEY (`idtransport`) USING BTREE,

KEY `FK\_transport\_sklad` (`nsklad`),

CONSTRAINT `FK\_transport\_sklad` FOREIGN KEY (`nsklad`) REFERENCES `sklad` (`Nsklad`)

) ENGINE=InnoDB AUTO\_INCREMENT=6 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci;

CREATE TABLE IF NOT EXISTS `zakaz` (

`id\_zakaz` int NOT NULL AUTO\_INCREMENT,

`data` date DEFAULT NULL,

`id\_client` int NOT NULL,

`id\_tovar` int NOT NULL,

`kol\_vo\_zak\_tov` int NOT NULL DEFAULT '0',

`cena\_tovara` float NOT NULL,

`sum\_zakaza` float NOT NULL,

PRIMARY KEY (`id\_zakaz`),

UNIQUE KEY `id\_zakaz` (`id\_zakaz`),

KEY `FKtovar` (`id\_tovar`),

KEY `FKclientz` (`id\_client`),

CONSTRAINT `FKclientz` FOREIGN KEY (`id\_client`) REFERENCES `klient` (`idklient`),

CONSTRAINT `FKtovar` FOREIGN KEY (`id\_tovar`) REFERENCES `tovar` (`idtovar`)

) ENGINE=InnoDB AUTO\_INCREMENT=4 DEFAULT CHARSET=utf8mb4 COLLATE=utf8mb4\_0900\_ai\_ci COMMENT='Список заказов п 10';

## 2.2 Внешние ключи

Внешние ключи позволяют установить связи между таблицами. Внешний ключ устанавливается для столбцов из зависимой, подчиненной таблицы, и указывает на один из столбцов из главной таблицы. Как правило, внешний ключ указывает на первичный ключ из связанной главной таблицы. можно установить действия, которые выполняются соответственно при удалении и изменении связанной строки из главной таблицы. В качестве действия могут использоваться следующие опции:

CASCADE: автоматически удаляет или изменяет строки из зависимой таблицы при удалении или изменении связанных строк в главной таблице.

SET NULL: при удалении или обновлении связанной строки из главной таблицы устанавливает для столбца внешнего ключа значение NULL. (В этом случае столбец внешнего ключа должен поддерживать установку NULL)

RESTRICT: отклоняет удаление или изменение строк в главной таблице при наличии связанных строк в зависимой таблице.

NO ACTION: то же самое, что и RESTRICT

Потенциальными ключами отношения Поставщики являются атрибуты ФИО и Идентификатор поставщика. Идентификатор занимает меньше места, а ФИО к тому же может измениться, поэтому мы выбираем Идентификатор в качестве первичного ключа. Точно такая же ситуация с кадрами и клиентами, ФИО может изменяться а Идентификатор и короче и останется постоянным. Аналогичным образом поступим с остальными таблицами.

## 2.3 Индексы

Индекс — объект базы данных, создаваемый с целью повышения производительности поиска данных. Таблицы в базе данных могут иметь большое количество строк, которые хранятся в произвольном порядке, и их поиск по заданному критерию путём последовательного просмотра таблицы строка за строкой может занимать много времени. Индекс формируется из значений одного или нескольких столбцов таблицы и указателей на соответствующие строки таблицы и, таким образом, позволяет искать строки, удовлетворяющие критерию поиска. Ускорение работы с использованием индексов достигается в первую очередь за счёт того, что индекс имеет структуру, оптимизированную под поиск.

# 3 СОЗДАНИЕ ПРЕДСТАВЛЕНИЙ И ГОТОВЫХ ЗАПРОСОВ

## Задание а.

В каком количестве и по какой цене имеется данный товар на данном складе?

В задании есть входные данные: наименование товара и наименование склада, и имеются данные, которые нам необходимо выяснить, это количество товара и его цена. Для получения необходимой информации нам необходимо задействовать две таблицы, таблицу товаров, из которой выбрать записи необходимого товара, и таблицу складов, из которых необходимо выбрать указанный склад. Из таблицы товаров по номеру склада хранения товара мы получим искомый склад, если на нем есть товар.

([Товар.склад=Склад.номер&Товар.наименование=’Ноутбук’&СкладюНаименование=’ 'Бобр Гоголя’])[Наименование товара, адрес склада, цена товара, количество товара]

SELECT t.naimen\_tovara, s.adress, t.cena, t.kolvo

from tovar t, sklad s

WHERE

t.sklad=s.Nsklad

AND t.naimen\_tovara='Ноутбук'

AND s.naimen='Бобр Гоголя'

Таблица товаров

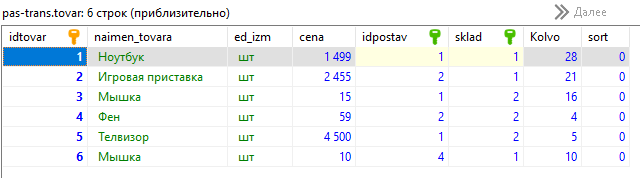
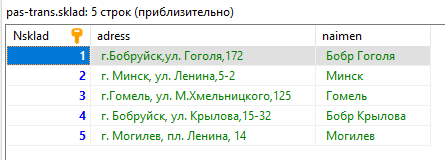
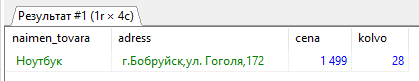


Таблица складов



Результат выполнения запроса



## Задание b

Кто из поставщиков не поставляет данный товар по данной цене.

В задании есть входные данные: наименование товара и его цена, и имеются данные, которые нам необходимо выяснить, это поставщик, который не поставляет указанный товар по этой цене. Для получения необходимой информации нам необходимо задействовать две таблицы: таблицу товаров и таблицу поставщиков. Из таблицы товаров мы выберем указанный товар с ценой не равной данной, и по этим записям выберем поставщиков.

([Товар.ИДпоставщика=Поставщик.ИДпоставщика&Товар.наименование=’Мышка’&Товар.цена<>10=’])[Поставщик.ФИО]

SELECT p.fio FROM

postav p, tovar t

WHERE

t.idpostav=p.idpostav

AND t.naimen\_tovara='Мышка'

AND t.cena<>10

Таблица поставщиков

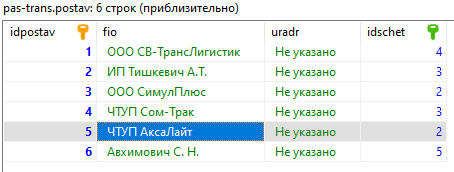
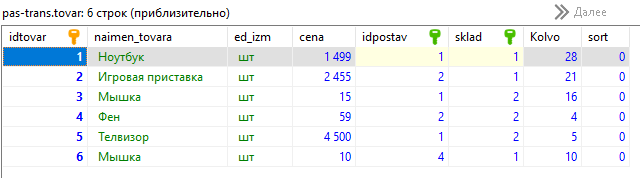
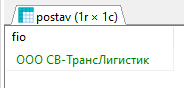


Таблица товаров



Результат выполнения запроса



## Задание c.

Какой товар и в каком количестве был отпущен на данную сумму по данной накладной.

В задании есть входные данные: номер накладной, и имеются данные, которые нам необходимо выяснить, это товар, его количество и сумма, который был отпущен. Для получения необходимой информации нам необходимо задействовать две таблицы: таблицу товаров и таблицу накладных. Из таблицы накладных мы выберем указанную накладную по её номеру, и по этой записи выберем товар из таблицы товаров.

([Накладная.ИДтовара=Товар.ИДтовара])[Товар.Наименование, Накладная.Количество]

SELECT t.naimen\_tovara, n.kolvo FROM

nakladnaya n, tovar t

WHERE

n.idtovar=t.idtovar

AND n.idnakladnaya=2

AND n.summa=100

Таблица накладных

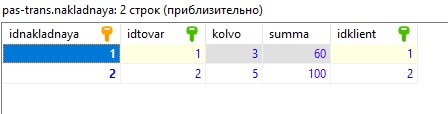
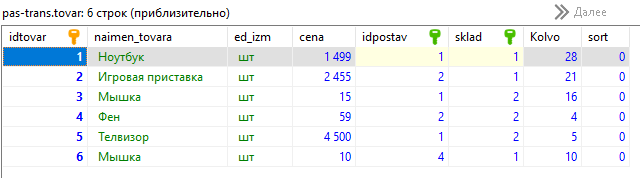
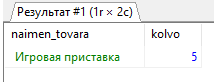


Таблица товаров



Результат выполнения запроса



## Задание d.

Кто из сотрудников данной фирмы является поставщиком товаров.

В задании необходимо найти записи у которых совпадает ФИО в таблице поставщиков и в таблице сотрудников.

([Поставщики.ФИО=Кадры.ФИО])[Кадры.ФИО, Кадры.Должность]

SELECT k.fio, k.dolgznost FROM

kadry k, postav p

WHERE

k.fio=p.fio

Таблица поставщиков

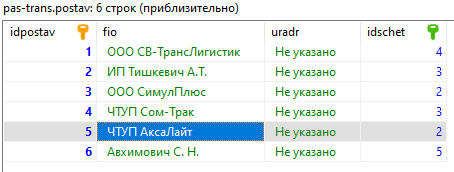
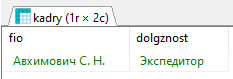


Таблица сотрудников



Результат выполнения запроса



## Задание e.

В каком банке открыл счет данный поставщик.

Входными данными для нас является название поставщика, выходными – банк и номер счета в банке.

([Поставщик.ФИО=’Название поставщика’&&Поставщик.ИДСчет=Счета.ИДСчет])[Поставщик.Наименование, Счет.Банк, Счет.Номер]

SELECT p.fio, s.bank, s.numсhet FROM

schet s, postav p

WHERE

p.fio= 'ЧТУП АксаЛайт'

AND p.idschet=s.idschet

Таблица банковских счетов

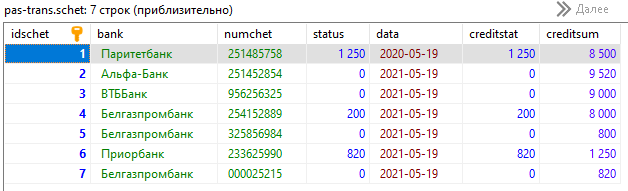
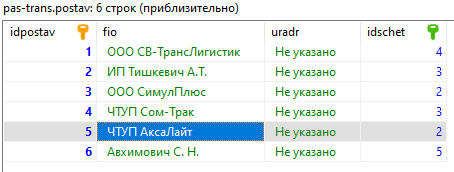
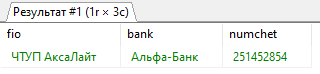


Таблица поставщиков



Результат выполнения запроса

****

# ЗАКЛЮЧЕНИЕ

Целью создания любой БД является упрощение использования больших массивов информации. БД позволяют собирать, хранить, обновлять и выводить информацию в понятной пользователю форме.

В ходе курсовой работы была разработана и создана база данных "Фирма", которая значительно упрощает работу с данными и во многом экономит время сотрудников. Созданная база данных позволяет вести учет товаров на складах, выданных накладных, управление персоналом фирмы, контроль заказов, в нее могут быть введены дополнительные средства формирования данных.

# СПИСОК ИСПОЛЬЗОВАННЫХ ИСТОЧНИКОВ