

Основные понятия и определения

База данных (БД) – унифицированная совокупность данных, совместно используемую различными приложениями в рамках некоторой единой автоматизированной информационной системы.

Программное обеспечение, осуществляющее операции над БД, получило название **СУБД** – система управления базами данных.

Запрос – специальным образом описанное требование, определяющее состав производимых над БД операций по выборке или модификации хранимых данных.

Для подготовки запросов чаще всего используют структурированный язык запросов – SQL (Structured Query Language). Этот язык стал фактическим стандартом языка работы с реляционными БД. Он является непроцедурным языком и не содержит операторов управления, организации подпрограмм, ввода-вывода и т. д. Поэтому SQL автономно не используется, а обычно погружен в среду встроенного языка программирования СУБД или процедурного языка (типа C++).

Современные СУБД позволяют создавать запросы, не применяя SQL. Однако его применение позволяет расширить возможности СУБД.

Категории SQL-запросов:

- ✓ определение данных (Data Definition Language, DDL) – SQL-запросы, позволяющие пользователям создавать и модифицировать структуру объектов БД (таблицы, представления и индексы); команды DDL влияют на контейнеры, содержащие данные, а не на данные;
- ✓ запросы данных (Data Query Language, DQL) – включает выражения SQL для получения данных из базы;
- ✓ манипуляции с данными (Data Manipulation Language, DML) – SQL-запросы, позволяющие пользователю добавлять и удалять данные (в форме строк), а также модифицировать имеющиеся в БД;
- ✓ контроль данных (Data Control Language, DCL) – SQL-запросы, позволяющие администраторам контролировать доступ к данным в базе и использовать различные системные привилегии СУБД;
- ✓ контроль транзакций – набор команд, которые пользователь применяет для того, чтобы вся транзакция либо была успешно выполнена, либо нет; команды контроля транзакций не вполне соответствуют синтаксису SQL-запросов, но положительно влияют на выполнение запросов, включенных в транзакцию.

Архитектура современных профессиональных СУБД базируется на принципах клиент-серверного взаимодействия программных компонентов. Сервер – процесс, обслуживающий информационную потребность клиента. Клиент – приложение, посылающее запрос на обслуживание сервером. Клиент инициирует связь с сервером, определяет вид запроса, получает от сервера результат обслуживания, подтверждает окончание обслуживания. Поскольку стандартом

интерфейса «клиент-сервер» в этом случае является язык SQL, СУБД называют SQL-сервером.

На клиентском компьютере может выполняться SQL-клиент – программа, предоставленная поставщиком СУБД и обеспечивающая пользователю возможность вводить SQL-запросы, посыпать их в СУБД и просматривать результат.

По пользовательскому интерфейсу SQL-клиенты разделяются на три типа:

- ✓ клиент с интерфейсом командной строки – команды вводятся с клавиатуры как текст, клиент можно использовать в любой операционной системе;
- ✓ клиент с графическим интерфейсом пользователя (GUI, Graphical User Interface) – выполняется в оконной системе (Microsoft Windows) и отображает данные, используя графические элементы (значки, кнопки и диалоговые окна);
- ✓ клиент с Web-интерфейсом – выполняется на сервере БД, а для взаимодействия с пользователем используется Web-браузер на клиентском компьютере.

Одним из наиболее популярных SQL-серверов БД является MySQL – небольшая и надежная реляционная СУБД с возможностью отката и восстановления после сбоя, многопользовательская, многопоточная, с высокой производительностью. Сервер MySQL предназначен как для критических по задачам производственных систем с большой нагрузкой, так и для встраивания в программное обеспечение массового распространения.

MySQL – открытое программное обеспечение (распространяется с открытым исходным кодом). Благодаря высокой производительности и простоте настройки, богатому выбору API-интерфейсов, а также функциональным средствам работы с сетями, сервер MySQL стал одним из самых удачных вариантов для разработки Web-приложений, взаимодействующих с БД. Система MySQL может быть реализована как:

- ✓ автономная настольная система;
- ✓ клиент-серверная система.

Если MySQL используется как автономная настольная система, то клиентское приложение исполняется на том же компьютере, на котором хранится программное обеспечение MySQL и БД. Сетевые соединения от клиента к серверу не устанавливаются. Настольные системы полезны в следующих случаях:

- ✓ при доступе к БД лишь одного пользователя;
- ✓ при небольшом числе пользователей, работающих с БД не одновременно.
- ✓ Клиент-серверная система может иметь:
- ✓ двухзвенную установку;
- ✓ трехзвенную установку.

Независимо от варианта установки, программное обеспечение и базы данных MySQL размещаются на центральном компьютере (сервере баз данных). Пользователи работают на компьютерах-клиентах. Доступ пользователей к серверу БД производится при помощи:

- ✓ приложений с компьютеров-клиентов (в двухзвенных системах);
- ✓ приложений, выполняющихся на специальном компьютере – сервере приложений (в трехзвенных системах).

В двухзвенных системах клиенты исполняют приложения, осуществляющие доступ к серверу БД непосредственно через сеть. Клиенты называются толстыми, поскольку выполняют два вида работы:

- ✓ исполняют программный код, соответствующий функциональным задачам;
- ✓ исполняют код, отображающий результаты доступа к БД.

Двухзвенная установка полезна при небольшом количестве пользователей, потому что для соединения с каждым из пользователей расходуются системные ресурсы (память и блокировки). Чем больше количество соединений с пользователями, тем хуже производительность системы из-за соперничества за ресурсы [4-5].

В трехзвенных системах в задачи компьютеров-клиентов входит лишь исполнение программного кода по вызову функций сервера приложений и отображение результатов. Такие клиенты называются тонкими. Сервер приложений исполняет многопотковые приложения, с которыми могут работать много пользователей одновременно. Сервер приложений соединяется с сервером БД, осуществляет доступ к данным и возвращает результаты клиенту.

С распространением Интернета клиенты и серверы стали взаимодействовать в глобальной сети. Web-среда предоставила пользователям дружественный интерфейс, за формирование которого отвечает Web-сервер. Такой подход позволил использовать для работы с удаленными БД Web-браузер, не прибегая к услугам специфических клиентских программ. Например, клиенты торговой компании, желающие ознакомиться со списком товаров, используют браузер для посещения сайта компании. Web-страницу со списком товаров формирует специальный модуль (скрипт), выполняющийся на Web-сервере компании. Для получения информации этот скрипт посылает SQL-запросы СУБД, находящейся на сервере БД. Таким образом, в трехуровневой архитектуре Интернета выделяются:

- ✓ клиент – Web-браузер (клиентское приложение), который взаимодействует с Web-сервером, посылая ему запросы на отображение той или иной Web-страницы;
- ✓ Web-сервер – на котором выполняется Web-приложение, формирующее SQL-запрос к СУБД (которая должна вернуть необходимые данные из БД); сервер баз данных – на котором размещены СУБД и база данных.

Подключение к серверу MySQL

Программный продукт MySQL WorkBench предназначен для визуального проектирования баз данных, а также обладает возможностью по управлению сервером MySQL. Чтобы приступить к выполнению лабораторной работы, необходимо создать подключение к серверу MySQL используя учетные данные, которые вы придумали ранее (В случае, если вы забыли свой логин и пароль обратитесь к преподавателю). Для этого, необходимо нажать в стартовом окне MySQL Workbench на соответствующую пиктограмму:

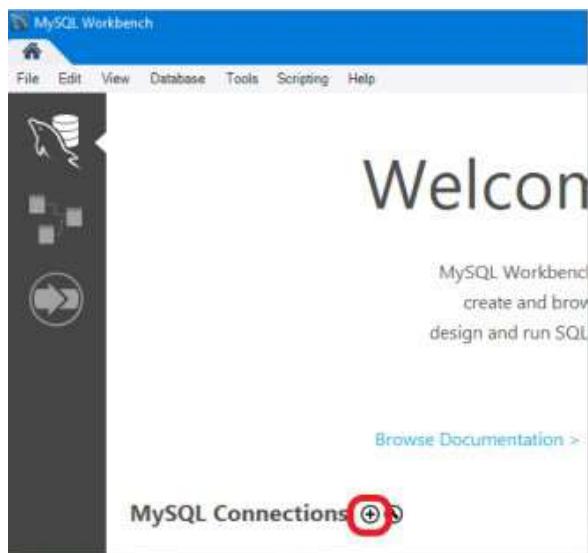


Рисунок 1 – Стартовое окно MySQL Workbench

После нажатия откроется «Менеджер соединений с сервером» (рисунок 2), где

необходимо создать новое подключение нажатием на кнопку «New» к серверу MySQL, для этого Вам необходимо указать:

- «Имя подключения» - любое логичное название;
- «Адрес сервера» - 95.131.149.21;
- «Порт» - стандартный порт 3306;
- «Имя пользователя» - логин, который был выдан преподавателем;
- «Пароль» - пароль, который был выдан преподавателем. MySQL обеспечивает

безопасное хранение пароля в специальном защищенном хранилище.

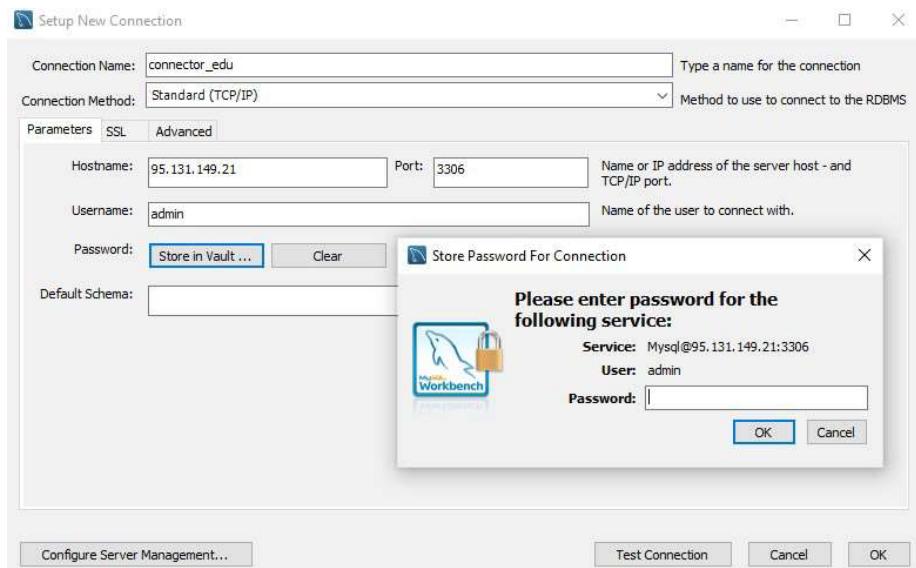


Рисунок 2 – Менеджер подключений к серверу

Для проверки корректности введенных данных, необходимо нажать на кнопку «Протестировать соединение», если тест прошел успешно можно закрыть окно менеджера рис.3.

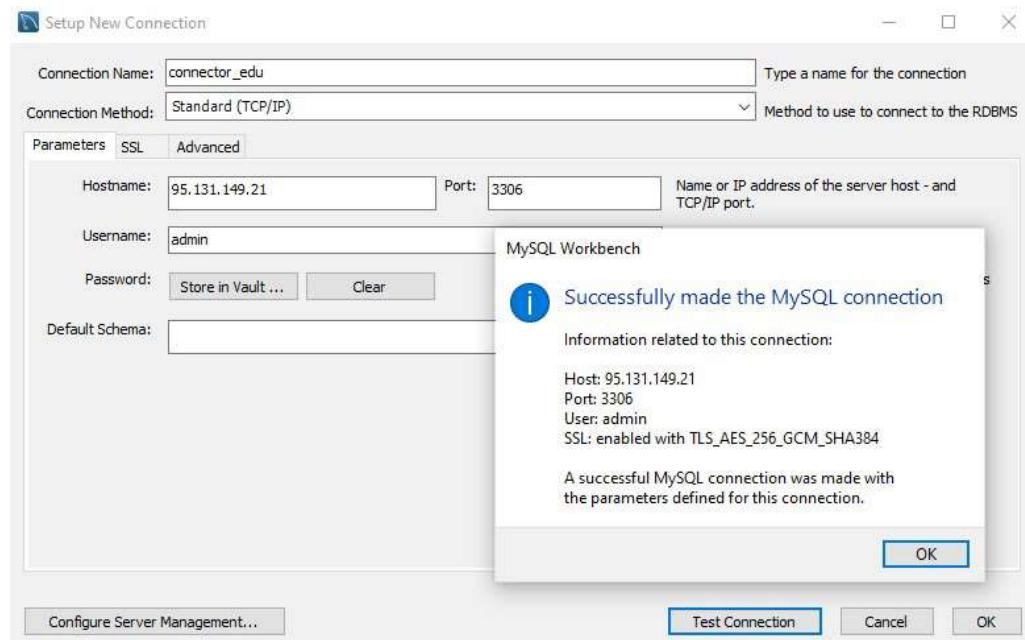


Рисунок 3 – Тестирование соединения в «Менеджер соединений с сервером»

Практическая работа №1 Построение ER-модели

Цель работы: проектирование информационно-логической модели базы данных при помощи case-средства mySQLWorkbench

Теоретические сведения

1. Создание ER-диаграммы

Среда **mySQL Workbench** предназначена для визуального проектирования баз данных и управления сервером **mySQL**.

Для построения моделей предназначена секция Data Modeling:



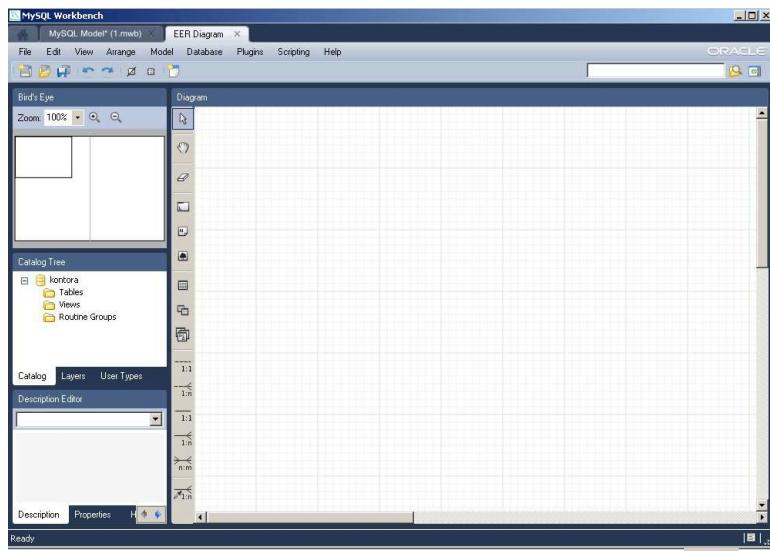
Выберем пункт **Create new EER Model**.

EER model расшифровывается как **Extended Entity-Relationship Model** и переводится как **Расширенная модель сущностей-связей**.

По умолчанию имя созданной модели **myDB**. Щелкните правой кнопкой мыши по имени модели и выберите в появившемся меню пункт **Edit schema**. В появившемся окне **можно** изменить имя модели. Назовем ее, например, **kontora**. В именах таблиц и столбцов нельзя использовать русские буквы.

В этом окне также **нужно** настроить так называемую «кодовую страницу» для корректного отображения русских букв **внутри** таблиц. Для этого выберите из списка пункт **«utf-8_general_ci»**. Окно свойств можно закрыть.

Диаграмму будем строить с помощью визуальных средств. Щелкнем по пункту **Add diagram**, загрузится пустое окно диаграммы:



Создать новую таблицу можно с помощью пиктограммы . Нужно щелкнуть по этой пиктограмме, а потом щелкнуть в рабочей области диаграммы. На этом месте появится таблица с названием по умолчанию **table1**. Двойной щелчок по этой таблице открывает окно редактирования, в котором можно изменить имя таблицы и настроить её структуру.

Будем создавать таблицу **Отделы** со следующими столбцами: `номер_отдела`, `полное_название_отдела`, `короткое_название_отдела`. Переименуем **table1** в **k_dept** и начнем создавать столбцы.

Каждый столбец имеет:

- имя (не используйте русские буквы в имени!),
- тип данных. Самые распространенные типы данных:
 - INT – целое число;
 - VARCHAR(размер) – символьные данные переменной длины, в скобках указывается максимальный размер;
 - DECIMAL(размер, десятичные_знаки) – десятичное число;
 - DATE – дата;
 - DATETIME – дата и время.

Далее располагаются столбцы, в которых можно настроить дополнительные свойства поля, включив соответствующий флагок:

- PK (primary key) – первичный ключ;
- NN (not null) – ячейка не допускает пустые значения;
- UQ (unique) – значение должно быть уникальным в пределах столбца;
- AI (auto incremental) – это свойство полезно для простого первичного ключа, оно означает, что первичный ключ будет автоматически заполняться натуральными числами: 1, 2, 3, и т.п.;
- DEFAULT – значение по умолчанию, т.е., значение, которое при добавлении новой строки в таблицу автоматически вставляется в ячейку сервером, если пользователь оставил ячейку пустой.

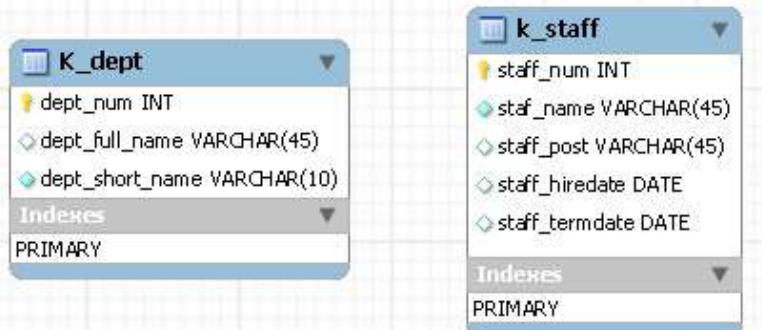
Таблица **Отделы** имеет следующий вид:

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
dept_num	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>					
dept_full_name	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
dept_short_name	VARCHAR(10)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					

Затем создадим таблицу **Сотрудники** со следующими столбцами: номер_сотрудника, имя_сотрудника, должность, дата_начала_контракта, дата_окончания_контракта

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
staff_num	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>					
staf_name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
staff_post	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
staff_hiredate	DATE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	
staff_termdate	DATE	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	

Созданные таблицы выглядят следующим образом:



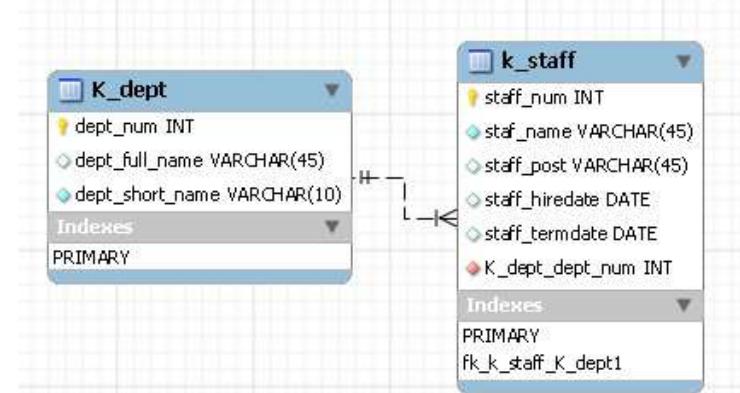
Обратите внимание, что при создании первичного ключа автоматически создается **индекс** по этому первичному ключу. **Индекс** представляет собой вспомогательную структуру, которая служит, прежде всего, для **ускорения поиска и быстрого доступа** к данным.

Теперь свяжем эти таблицы. Сначала создадим связь «Работает» между **Сотрудником** (дочерняя таблица) и **Отделом** (родительская таблица), степень связи M:1. Для создания связей M:1 служит пиктограмма на панели инструментов



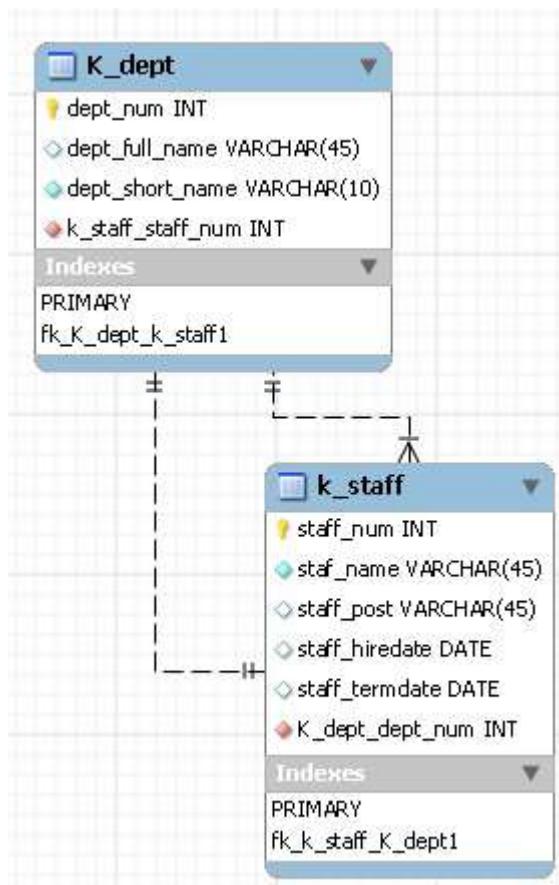
(с пунктирной линией). С ее помощью создается так называемая «неидентифицирующая связь», т.е. обычновенный внешний ключ, при этом первичный ключ родительской таблицы добавляется в список столбцов дочерней таблицы.

Итак, щелкнем на пиктограмме, затем щелкнем на дочерней таблице **Сотрудники**, затем на родительской таблице **Отделы**:



Обратите внимание, что при этом произошло. Между таблицами образовалась пунктирная линия; в сторону «к одному» она отмечена двумя черточками, в сторону «ко многим» - «куриной лапкой». Кроме того, в таблице **Сотрудники** образовался дополнительный столбец, которому автоматически присвоено имя *k_dept_dept_num* (т.е., имя родительской таблицы плюс имя первичного ключа родительской таблицы). А в группе **Индексы** создан индекс по внешнему ключу.

Теперь добавим связь между этими же таблицами «Руководит» 1:1. Выберем пиктограмму 1:1, затем щелкнем по **Отделам**, затем по **Сотрудникам**.



Чтобы 2 связи на картинке не «заязывались узлом», мы их разместили друг под другом.

Обратите внимание, что в таблицу Отделы был автоматически добавлен столбец *k_staff_staff_num*, а также индекс по внешнему ключу.

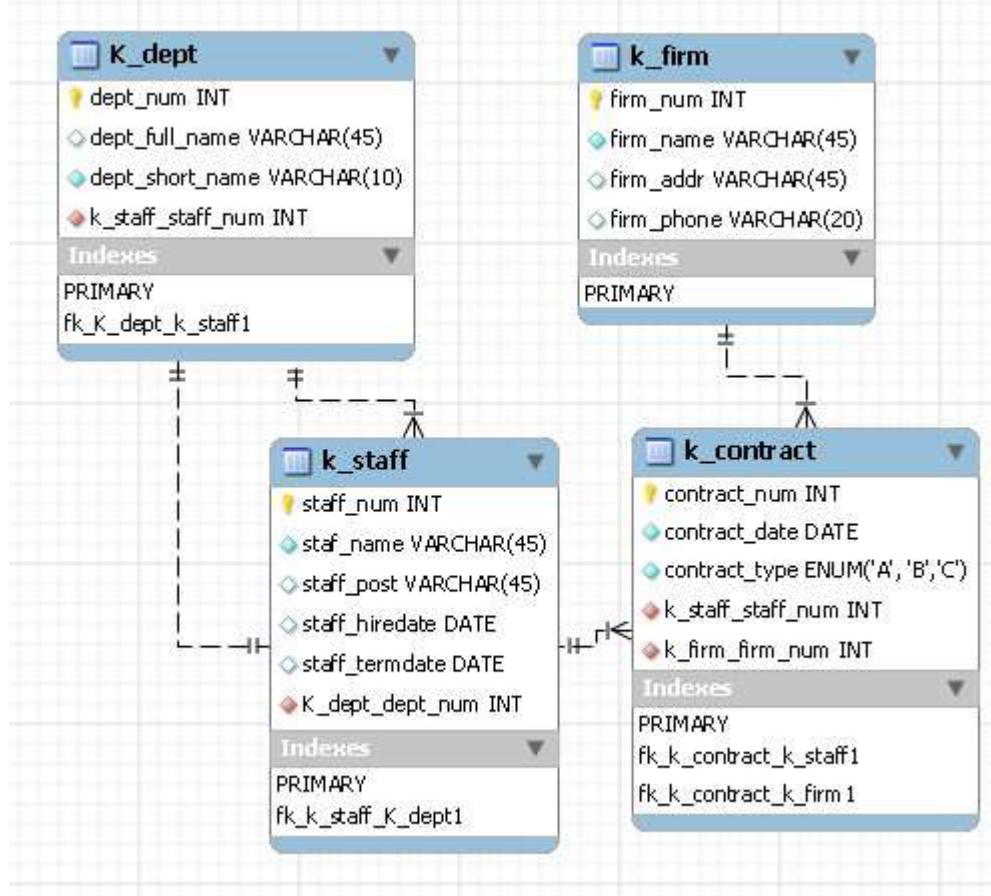
Создадим таблицу **Предприятия**:

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
firm_num	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
firm_name	VARCHAR(45)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				
firm_addr	VARCHAR(45)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>
firm_phone	VARCHAR(20)	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>

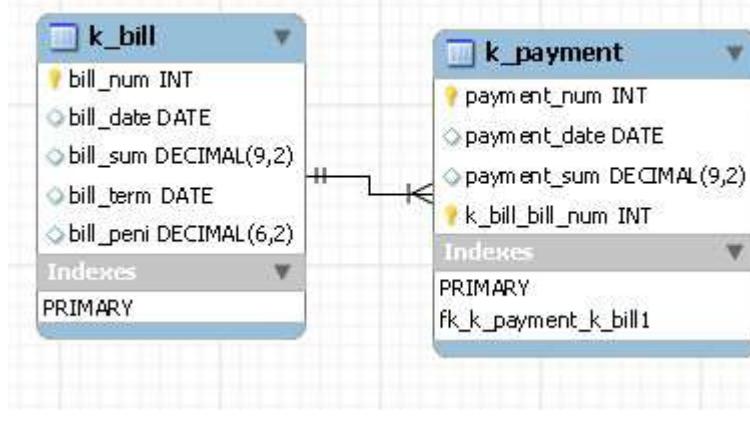
Создадим таблицу **Договоры**. У столбца Тип_договора зададим следующий формат: это буква из списка ‘A’, ‘B’, ‘C’.

Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
contract_num	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>				
contract_date	DATE	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				
contract_type	ENUM('A', 'B', 'C')	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>	<input type="checkbox"/>				

Свяжем **Договоры** с **Сотрудниками** и **Предприятиями** связями M:1.



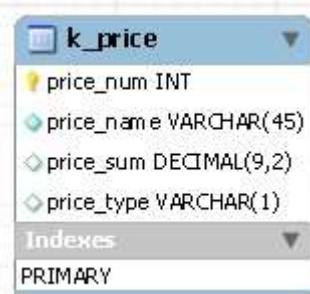
Затем создадим **Счета и Платежи**:



Платежом. Идентифицирующая связь создается с помощью пиктограммы (со сплошной линией). При этом новый столбец *k_bill_bill_num* становится не только внешним ключом в таблице **Платеж**, но и частью первичного ключа.

Далее создадим таблицу **Прайс-лист** со (номер_товара, название_товара, цена и тип_товара).

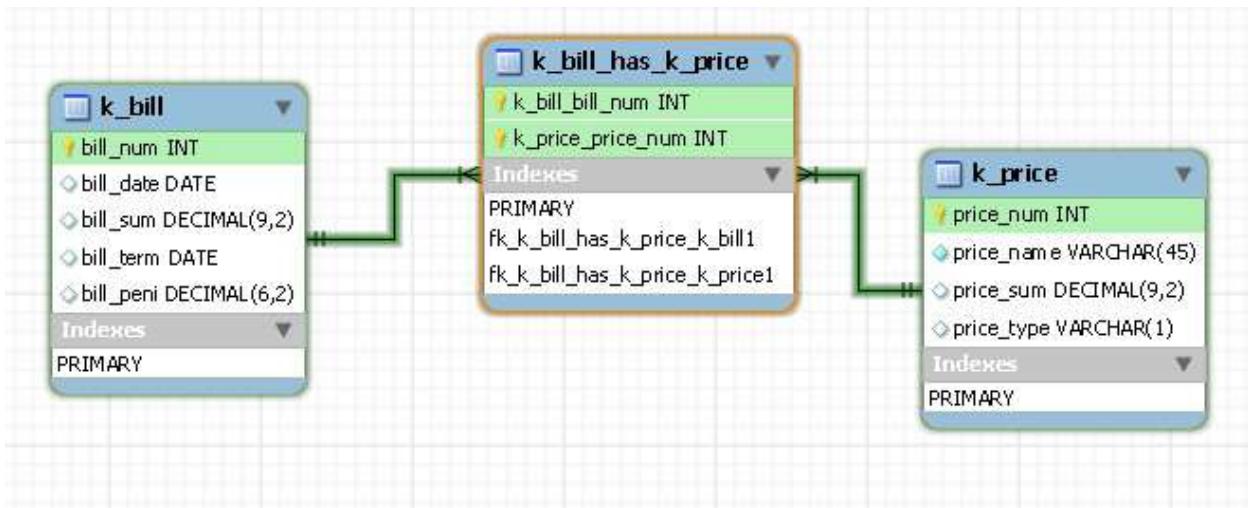
Поскольку сущность **Платеж** была «слабой», у нее нет полноценного первичного ключа, и каждый платеж однозначно идентифицируется группой атрибутов (номер_счета, номер_платежа). Отметим в качестве ключевого поля *payment_num*, а затем создадим **идентифицирующую** связь между **Счетом** и



столбцами на_товара

Между объектами **Счет** и **Прайс-лист** имеется связь «многие - ко многим».

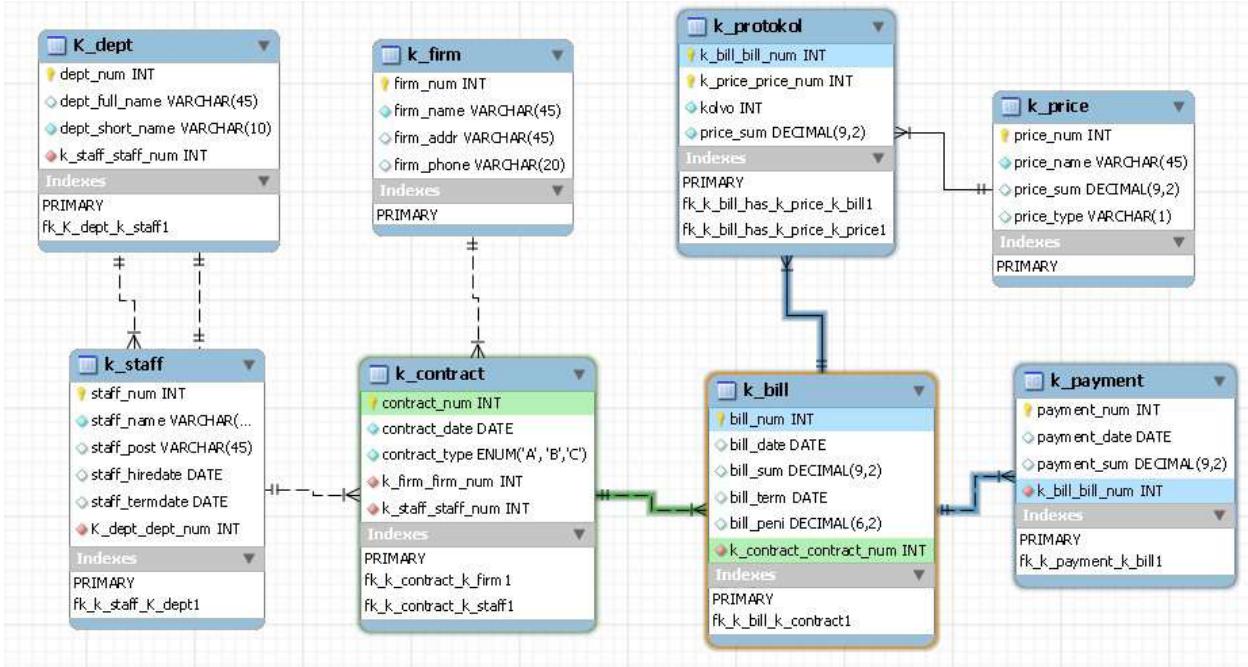
Для создания этой связи нужно использовать пиктограмму  . Следует щелкнуть мышью по этой пиктограмме, а затем последовательно щелкнуть по связываемым таблицам. Между ними появится новая таблица, обратите внимание на ее столбцы, первичный ключ и внешние ключи:



Для удобства переименуем эту таблицу в *k_protokol* (**ПротоколСчета**), добавим столбцы *kolvo* и *price_sum*.

k_protokol									
Column Name	Datatype	PK	NN	UQ	BIN	UN	ZF	AI	Default
k_bill_bill_num	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
k_price_price_num	INT	<input checked="" type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
kolvo	INT	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					
price_sum	DECIMAL(9,2)	<input type="checkbox"/>	<input checked="" type="checkbox"/>	<input type="checkbox"/>					

Теперь EER-диаграмма имеет такой вид:



Задание. Создайте в MySQL WORKBENCH ER-диаграмму для своей задачи.
Номер варианта соответствует номеру задачи.

- Институт.** Таблицы: студенты, преподаватели, предметы, группы, кафедры, книги по предметам. Представление: по студенту найти книги, по предметам, которые он проходит. Роли: Студент — может просматривать, но не может вносить изменения; Преподаватель — может просматривать и вносить изменения в базу данных.
- Магазин.** Таблицы: покупатели, продавцы, товары, покупки (связывает покупателей с товарами и продавцами, продавшими товар), товарные группы, отделы. Представление: по отделам найти покупателей. Роли: Покупатель — видит товары и свои покупки; Продавец: обладает всеми правами.
- Банк.** Таблицы: клиенты, договора(между клиентом и операционистом, на конкретный тип вклада), типы вкладов, отделения банка, операционисты. Представление: найти клиентов, с которыми заключил договора выбранный операционист.. Роли: Клиент, Операционист.
- Библиотека.** Таблицы: читательские билеты, книги, заказы книг (сопоставление книг и читательских билетов), авторы, жанры. Представление: определить любимые жанры выбранного читателя. Роли: Библиотекарь, Читатель.
- Сотовый оператор.** Таблицы: клиенты, записи разговоров (записи о клиенте, времени разговора, тариф), счета клиентов, тарифы, области (где обслуживает оператор). Представление: вывести разговоры, сделанные в выбранной области. Роли: Клиент, Оператор.

6. **Агентство недвижимости.** Таблицы: недвижимость, клиент, агенты, договор аренды, отделы агентов, цены аренды. Представление: сколько аренды платит выбранный клиент. Роли: Клиент, Агент.
7. **Школа.** Таблицы: ученики, учителя, оценки, классы, предметы. Представление: по оценкам найти учеников. Роли: Ученики, Учителя.
8. Автосервис. Таблицы: клиенты, машины, мастера. Роли: Клиент, Мастер.
9. **Железнодорожная касса.** Таблицы: маршруты, поезда, билеты, клиенты, заказы. Представление: вывести заказы билетов по известному поезду. Роли: Кассир, Администратор — имеет полные права.
10. **Служба поддержки.** Таблицы: объекты, сотрудники, заявки на выполнение работ, неполадки, расходные материалы. Представление: вывести расходные материалы, использованные на выбранном объекте. Роли: Администратор, Техник.
11. **Оптовая база.** 1-я таблица : Код товара, название товара, количество на складе, единица измерения, стоимость единицы товара, примечания - описание товара; 2-я таблица: Номер, адрес, телефон и ФИО поставщика товара, срок поставки и количество товаров в поставке, номер счета. Один и тот же товар может доставляться несколькими поставщиками, и один и тот же поставщик может доставлять несколько видов товаров.
12. **Производство.** 1-я таблица: Код изделия, название изделия, является ли типовым, примечание - для каких целей предназначено, годовой объем выпуска; 2-я таблица: код, название, адрес и телефон предприятий, выпускающих изделия; 3-я таблица: название, тип, единица измерения материала, цена за единицу, отметка об использовании материала в данном изделии; 4-я таблица: количество материала в спецификации изделия, дата установления спецификации, дата отмены; 5-я таблица: год выпуска и объем выпуска данного изделия предприятием. Одно изделие может содержать много типов материалов, и один и тот же материал может входить в состав разных изделий.
13. **Авторемонтные мастерские.**
Минимальный список характеристик:
Номер водительских прав, ФИО, адрес и телефон владельца автомобиля;
номер, ФИО, адрес, телефон и квалификация (разряд) механика;
номер, марка, мощность, год выпуска и цвет автомобиля;
номер, название, адрес и телефон ремонтной мастерской;
стоимость наряда на ремонт, дата выдачи наряда, категория работ, плановая и реальная дата окончания ремонта.
Один и тот же автомобиль может обслуживаться разными автомеханиками, и один и тот же автомеханик может обслуживать несколько автомобилей.

14. Деканат

Минимальный список характеристик:
Код группы, курс, количество студентов, общий объем часов;
ФИО преподавателя, вид контроля, дата;
Название дисциплины, категория, объем часов.

Одна группа изучает несколько дисциплин и одна дисциплина может преподаваться нескольким группам.

Категория дисциплины - гуманитарная, математическая, компьютерная, общеинженерная и т.д. Вид контроля - зачет, экзамен.

15.Договорная деятельность организации

Минимальный список характеристик:

Шифр работы, название, трудоемкость, дата завершения;

ФИО сотрудника, должность, табельный номер;

Дата выдачи поручения на работу, трудоемкость, плановая и реальная даты окончания.

Одна и та же работа может выполняться несколькими сотрудниками, и один и тот же сотрудник может участвовать в нескольких работах.

16.Поликлиника

Минимальный список характеристик:

Номер, фамилия, имя, отчество, дата рождения пациента, социальный статус, текущее состояние;

ФИО, должность, квалификация и специализация лечащего врача;

диагноз, поставленный данным врачом данному пациенту, необходимо ли амбулаторное лечение, срок потери трудоспособности, состоит ли на диспансерном учете, дата начала лечения.

Текущее состояние - лечится, вылечился, направлен в стационар, умер.

Социальный статус пациента - учащийся, работающий, временно неработающий, инвалид, пенсионер

Специализация врача - терапевт, невропатолог и т.п.

Квалификация врача - 1-я, 2-я, 3-я категория.

Один и тот же пациент может лечиться у нескольких врачей и один врач может лечить несколько пациентов.

17.Телефонная станция

Минимальный список характеристик:

Номер абонента, фамилия абонента, адрес, наличие блокиратора, примечание;

Код АТС, код района, количество номеров;

Номер спаренного телефона абонента, задолженность, дата установки.

Один спаренный номер одной АТС может использоваться несколькими абонентами, и один и тот же абонент может использовать телефоны разных АТС.

18.Городской транспорт

Минимальный список характеристик:

Вид транспорта, средняя скорость движения, количество машин в парке, стоимость проезда;

номер маршрута, количество остановок в пути, количество машин на маршруте, количество пассажиров в день;

начальный пункт пути, конечный пункт, расстояние.

Один и тот же вид транспорта может на разных маршрутах использовать разные пути следования.

19.Библиотека2

Минимальный список характеристик:

Автор книги, название, год издания, цена, количество экземпляров, краткая аннотация;

номер читательского билета, ФИО, адрес и телефон читателя, дата выдачи книги читателю и дата сдачи книги читателем, отметка о выбытии.

Книга имеет много экземпляров и поэтому может быть выдана многим читателям.

20. Университет

Минимальный список характеристик:

Номер, ФИО, адрес и должность преподавателя, ученая степень;

код, название, количество часов, тип контроля и раздел предмета (дисциплины);

код, название, номер заведующего кафедрой;

номер аудитории, где преподаватель читает свой предмет, дата, время, группа.

Один преподаватель может вести несколько дисциплин и одна дисциплина может вестись несколькими преподавателями.

Примечание: Циклы дисциплин: гуманитарный, общеинженерный, математический, компьютерный и т.д.

21. Оптовая база

Минимальный список характеристик:

Код товара, название товара, количество на складе, единица измерения, стоимость единицы товара, примечания - описание товара;

Номер, адрес, телефон и ФИО поставщика товара, срок поставки и количество товаров в поставке, номер счета.

Один и тот же товар может доставляться несколькими поставщиками, и один и тот же поставщик может доставлять несколько видов товаров.

22. Производство

Минимальный список характеристик:

Код изделия, название изделия, является ли типовым, примечание - для каких целей предназначено, годовой объем выпуска;

код, название, адрес и телефон предприятий, выпускающих изделия;

название, тип, единица измерения материала, цена за единицу, отметка об использовании материала в данном изделии;

количество материала в спецификации изделия, дата установления спецификации, дата отмены;

год выпуска и объем выпуска данного изделия предприятием.

Одно изделие может содержать много типов материалов, и один и тот же материал может входить в состав разных изделий.