

# Лабораторная работа №1.

## Введение в проектирование реляционных БД.

Базы данных.

3 курс. 5 группа.

Кушнеров А.В. 2023-2024 г.

Уровень сложности: *Лёгкий*

Формат работы: Индивидуальная по вариантам.

Срок выполнения: **2 недели.**

### Цель работы

Получить начальные сведения о процессе проектирования БД. Приобретение навыков работы с ER-диаграммами и логическими схемами БД. Разработка простейшей БД в СУБД MySQL.

### Минимальные теоретические сведения

Этапы проектирования БД:

1. Концептуальная модель (ER-диаграмма)
2. Логическая модель (Определение ключевых атрибутов сущностей и связей между ними)
3. Физическая модель (Непосредственная реализация на СУБД)

Основные SQL-команды для создания БД на базе СУБД MySQL.

CREATE DATABASE *db\_name* – определение БД.

USE *db\_name* – переключение на работу с конкретной БД.

CREATE TABLE *table\_name* [(create\_definition, ...)] – создание таблицы.

### Пример 1:

Создадим первую таблицу, в которой будем хранить информацию о продуктах.

```
create table products
(
  product_id integer primary key auto_increment,
  product_name varchar(50),
  product_type varchar(30),
  product_price float
);
```

Следует обратить внимание на некоторые ключевые слова в этом примере.

PRIMARY KEY – помечает поле, как первичный ключ – уникальное для каждой строки таблицы значение.

AUTO\_INCREMENT – позволяет ввести автоматическую нумерацию значений первичного ключа. Это удобно и гарантирует уникальность.

После имени столбца (поля) таблицы указывают *тип данных* этого столбца.

При создании таблиц принципиальную роль играют связи между сущностями. Типы связей:

1. *Один к одному.* Один объект одной сущности соответствует одному объекту другой (паспорта – граждане).
2. *Один ко многим.* Один объект одной сущности может соответствовать нескольким объектам другой (кафедра – студенты).
3. *Многие ко многим.* Многие объекты одной сущности могут соответствовать нескольким объектам другой. Реализуется с помощью вспомогательной таблицы (преподаватели – студенты)

### **Пример 2:**

Ниже приведён код создания двух таблиц (продажи и работники), которые связаны отношением «один ко многим».

Один работник может совершить несколько продаж, но каждая продажа осуществляется только одним работником.

```
create table staff
(
  employee_id integer primary key auto_increment,
  employee_name varchar(50) not null,
  employee_surname varchar(50) not null,
  employee_position enum('barman', 'cook')
);

create table sells
(
  sell_id integer primary key,
  sell_date datetime not null,
  barman_id integer not null,
  sell_amount float not null,
  constraint cn2 foreign key (barman_id) references staff(employee_id)
);
```

Обратим внимание на некоторые моменты.

NOT NULL – в поле должно содержаться значение.

ENUM – тип данных, обозначающий ограниченный набор из значений.

FOREIGN KEY – ограничение внешнего ключа. Внешний ключ обеспечивает связь между таблицами. В данном примере в таблице с продажами хранится ссылка (barman\_id) на сотрудника, который осуществил продажу. Реализуется как разновидность оператора CONSTRAINT.

### **Пример 3:**

Для реализации связи «многие ко многим» создают вспомогательную таблицу.

```
create table products_sells
(
  sell_id integer not null,
  product_id integer not null,
  quantity integer,
  primary key(sell_id, product_id),
  constraint cn3 foreign key (sell_id) references sells(sell_id),
  constraint cn4 foreign key (product_id) references products(product_id)
);
```

Эта таблица закрепит отношение между продуктами и продажами.

Для заполнения таблиц данными используют команду INSERT.

INSERT INTO table\_name VALUES (values,...) – добавление записи в таблицу.

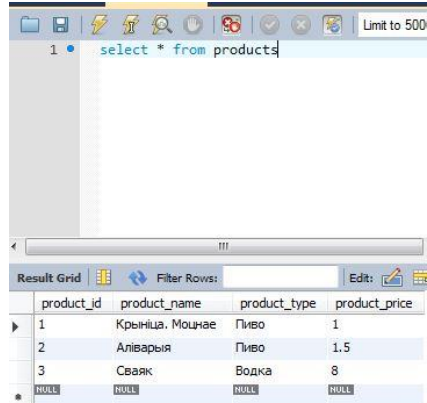
При заполнении будьте внимательны, особенно с типами данных и ограничениями на внешние ключи.

#### Пример 4:

Заполнение таблицы данными.

```
insert into products
(product_name,product_type,product_price)
values
('Аліварыя','Пиво',1.5),
('Сваяк','Водка',8);
```

Убедимся что данные добавлены.



The screenshot shows a database management interface. At the top, there's a toolbar with various icons. Below it, a SQL query is entered in a text area: `select * from products`. Below the query, there's a 'Result Grid' tab. The grid shows the results of the query, which are three rows of data. The columns are labeled 'product\_id', 'product\_name', 'product\_type', and 'product\_price'. The data rows are: 1. 'Крыніца, Моцнае' (Pivo) with price 1; 2. 'Аліварыя' (Пиво) with price 1.5; 3. 'Сваяк' (Водка) with price 8. There are also some 'NULL' values in the last row of the grid.

product_id	product_name	product_type	product_price
1	Крыніца, Моцнае	Пиво	1
2	Аліварыя	Пиво	1.5
3	Сваяк	Водка	8
NULL	NULL	NULL	NULL

Некоторые дополнительные полезные команды.

DESCRIBE table\_name – выводит описание таблицы.

SHOW TABLES – показывает все таблицы выбранной БД.

ALTER TABLE – команда для изменения созданной таблицы (будет рассмотрена в следующих ЛР подробнее).

DROP TABLE table\_name – удаление таблицы.

#### Пример 5:

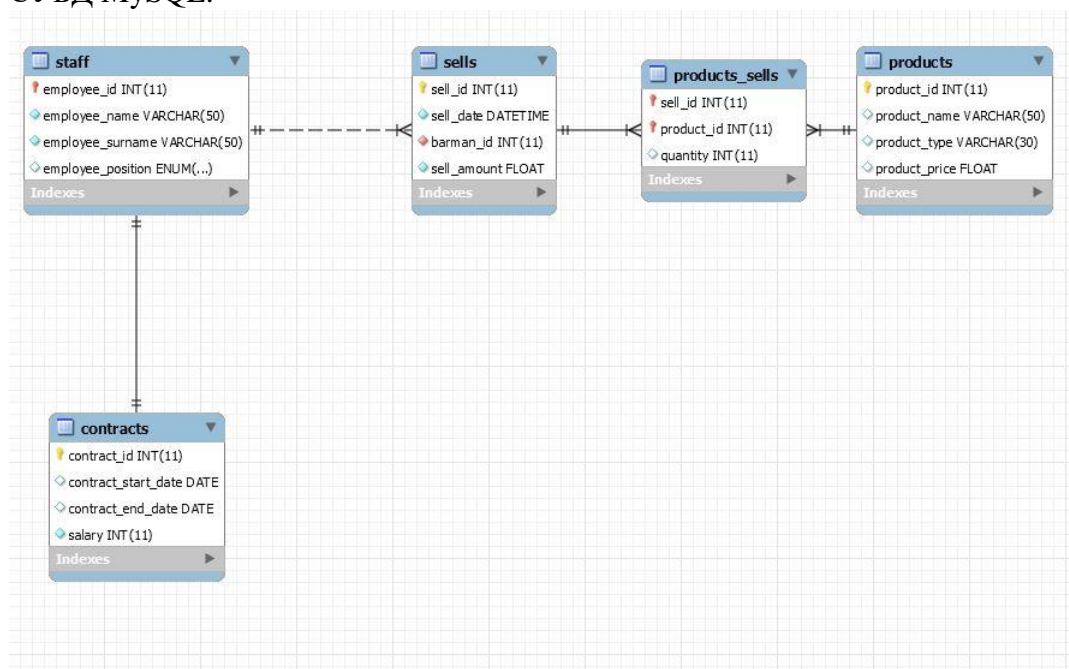
Оператор CHECK позволяющий наложить ограничения на значения столбцов для обеспечения корректности данных. По своей сути тоже является частью CONSTRAINT. Может быть реализован несколькими способами.

```
CREATE TABLE product (
    product_id INTEGER PRIMARY KEY AUTO_INCREMENT,
    product_name VARCHAR(50) CHECK (LEFT(product_name, 1) <> 'z'),
    product_type VARCHAR(30),
    product_price FLOAT CHECK (product_price >= 0)
);

CREATE TABLE contract (
    contract_id INTEGER PRIMARY KEY AUTO_INCREMENT,
    contract_start_date DATE,
    contract_end_date DATE,
    salary INTEGER NOT NULL DEFAULT '0',
    constraint cont_start_end_cons check (contract_start_date <> contract_end_date),
    constraint salary_max_cons check (salary < 10000)
);
```

### Задание для самостоятельной работы

1. Реализуйте и минимально заполните БД «Бар» согласно следующей физической схеме на базе СУБД MySQL.



Обратите внимание на то, что сущности «работники» и «контракты» связаны отношением «один к одному». Подумайте, как реализовать его программно.

2. Спроектируйте реляционную БД согласно вашему варианту (не менее 7 таблиц).
  - 2.1. Тщательно спроектируйте требования к вашей БД.
  - 2.2. Разработайте ER-модель для заданной БД
  - 2.3. Разработайте логическую модель БД
  - 2.4. Реализуйте физическую модель БД на СУБД MySQL. Реализацию производить с помощью SQL команд.

Вариант	База данных	Обязательные детали
1	Архив цитат Джейсона Стэтхэма.	Цитаты, мероприятия, окружение, пресса
2	Завод химического оружия	Продукция, заказы, склад, магазины.
3	Команда по кёрлингу	Игроки, игры, тренировочный процесс, тренеры.
4	Беларусь	Города, области, предприятия, реки, озёра.
5	Сообщество защитников природы.	Акции, состав, руководство.
6	Кладбище.	Работники, реклама, статистика, захоронения.
7	Учебная версия соц. сети.	Пользователи, группы, записи.
8	Тюрьма.	Заклученные, работники, группировки, встречи с родственниками.
9	Центр детского творчества.	Дети, группы, воспитатели, кружки.
10	Спортивная газета.	Новости, авторы, статьи, рубрики, доходы, реклама.

3. Выполните задания (разрешено использовать только оператор создания таблицы и входящие в него):

- 3.1. Предложите вариант создания таблицы «contracts» таким образом, чтобы при добавлении контракта в таблицу, автоматически устанавливалась текущая дата.
- 3.2. В тестовой БД приведите пример 15 типов данных в MySQL. Проведите с ними минимальные операции.
- 3.3. Предложите способ автоматического создания первичного ключа в виде уникальной строки для каждого работника.
- 3.4. Реализуйте возможность автоматической установки возраста кого-либо в вашей БД исходя из даты рождения и текущей даты.
- 3.5. Обеспечьте возможность автоматического создания никнейма для некоторого работника при его добавлении в БД. Никнейм формируется исходя из имени, фамилии и других данных по вашему выбору и должен быть уникальным в любом случае.
- 3.6. Попробуйте создать БД из задания 2. Только все те же данные разместить в одной или 2 таблицах. Сделайте выводы о работоспособности.

Литература:

1. Кузнецов, Симдянов – MySQL 5.0. (Глава 4)
2. Линн Бейли – Изучаем SQL.
3. Кронке – Теория и практика построения баз данных (Глава 3).
4. Коннолли, Берг – Базы данных.