Лабораторная работа №1

«Физическая модель данных MySQL»

Теоретический раздел

Общий синтаксис конструкций

В реляционных СУБД для выполнения операций над отношениями используются две группы языков, в качестве математической базы для которых используются теоретические языки запросов, предложенные Эдгаром Iддом:

- *реляционная алгебра*;
- *реляционное исчисление*.

В первом случае (реляционная алгебра) операнды и результаты всех действий являются отношениями. Такие языки — процедурные, поскольку отношение, которое является результатом запроса к базе данных, вычисляется при последовательном выполнении операторов, применяемым к отношениям. Сами операторы состоят из операндов (отношений) и реляционных операций (их результатом тоже является отношение).

Языки реляционного исчисления — непроцедурные, их называют *описательными* или *декларативными*. Они позволяют составлять запросы с помощью предиката первого порядка, которому должны удовлетворять кортежи или домены отношений. Таким образом, запрос к базе данных на таком языке содержит только информацию о желаемом результате. К языкам этой группы относится и SQL.

SQL (Structured Query Language, "язык структурированных запросов") — универсальный язык, применяемый для создания, модификации и управления данными в реляционных базах данных. В нем создается линейная последовательность операторов языка, которые выполняются СУБД. Операторы состоят из:

- имен операций и функций;
- имен таблиц и их столбцов;
- зарезервированных ключевых слов и специальных символов;
- логических, арифметических и строковых выражений.

Общий вид простого оператора в SQL:

```
ОПЕРАТОР параметры;
```

Если параметр не один, а несколько, то они перечисляются через запятую.

Все предложения на языке SQL оканчиваются точкой с запятой. Например:

```
USE `first_lab_data_base`;
SELECT `id`, `field1` FROM `mytable`;
```

Выражения в SQL не зависят от регистра, не требуют обязательного наличия кавычек при обозначении названий, дополнительные разделители (пробел, табуляция, переход на новую

строку) игнорируются. Для обозначения названий баз данных, таблиц, атрибутов таблиц, то есть названий, связанных с объектами СУБД, могут использоваться кавычки типа «тупое ударение» («`»). Например:

```
`элемент_бд`
```

Для текстовых данных, вводимых пользователем в базу и не связанных с элементами СУБД (например, обычных строковых значений), используются обычные или двойные кавычки: 'текст', "большой текст". Чтобы с помощью двойных кавычек указать строку, которая уже содержит двойные кавычки в конце и начале выражения, нужно записывать ее так (экранирование):

```
"\"заголовок\""
```

Существует общепринятый стиль «правильного» оформления выражений. Оно заключается в том, что при написании каких-либо выражений:

- после естественных разделителей выражений (например, запятых) ставится пробел;
- дополнительные разделители (пробелы, табы) не используются, если нет необходимости записать многостроковое выражение в удобном для чтения виде;
- системные обозначения (названия операторов, функций, ключевых слов и т.п.) пишутся заглавными буквами;
- при указании названий, связанных с объектами СУБД, обязательно используются кавычки в виде «тупого ударения».

Например:

```
INSERT INTO `news` (`id`, `post_date`) VALUES (42, '2015-02-02 04:13:15');
```

Далее **будут использоваться квадратные скобки для обозначения дополнительных (но необязательных) параметров**. Выражения, с параметром в квадратных скобках и без него, являются синтаксически правильными.

Например:

```
SELECT * FROM `table` [ WHERE условие ];
```

Такое выражение можно интерпретировать не только так:

```
SELECT * FROM `table` WHERE условие;
```

Но и как аналогичное выражение без необязательных параметров, т.е.:

```
SELECT * FROM `table`;
```

Типы данных в MySQL

В MySQL представлено множество типов данных, в соответствии с которыми хранятся и обрабатываются все данные в таблицах. Перечислим некоторые из них.

Числа

Числа разделяются на целые и дробные. Целые представлены следующими типами данных:

- TINYINT — 1 байт, т.е. принимает значения от -128 до 127 (в случае использования TINYINT UNSIGNED, т.е. без учета знака перед числом — 0..255);
- SMALLINT — 2 байта, -32768..32767 (0..65535);
- MEDIUMINT — 3 байта, -8388608..8388607 (0..2^24-1);
- INT — 4 байта, -2147483648..2147483647 (0..2^32-1);
- BIGINT — 8 байт, -2^32..2^32-1 (0..2^64).

Дробные числа представлены следующими типами данных:

- FLOAT (4 байта);
- DOUBLE (8 байт) — вдвое большая точность после запятой.

Строки

- CHAR — дополняет до заданной «ширины» (например, в случае использования CHAR(6) строка из пяти символов «hello» будет храниться как «hello », т.е. с пробелом на конце);
- VARCHAR — использует необходимый минимум памяти (в случае использования VARCHAR(6) строка из пяти символов «hello» будет храниться как «hello»);
- BLOB, TINYBLOB, MEDIUMBLOB, LONGBLOB — бинарные данные разных размеров;
- TEXT, TINYTEXT, MEDIUMTEXT, LONGTEXT — текстовые данные разных размеров;
- ENUM — одно из заданных значений (например, в ячейке типа ENUM(0,1,2) может храниться ноль, единица или двойка);
- SET — ноль или более заданных значений (в ячейке типа SET(0,1,2) может храниться любая комбинация из ноля, единицы и двойки — в том числе и пустое значение).

Другие типы

- булев: BOOL, BOOLEAN;
- уникальный автоматически увеличившийся идентификатор: SERIAL (== BIGINT UNSIGNED NOT NULL AUTO_INCREMENT UNIQUE);
- дата и время: DATETIME, DATE, TIMESTAMP, TIME, YEAR.

Работа с базами данных и таблицами

Команды для манипулирования базами данных

1. Создание базы данных:

```
CREATE DATABASE `db_name`;
```

Параметр команды создания баз данных — имя, выдаваемое создаваемой базе данных. Например, для создания базы данных под названием "my_database" нужно ввести команду:

```
CREATE DATABASE `my_database`;
```

Несмотря на то, что современная версия MySQL позволяет создавать БД с кириллическими и специальными символами в названии, принято использовать латинские буквы, цифры и знаки подчеркивания («_»).

2. При одновременной работе в нескольких базах данных в командах нужно уточнять, с данными какой БД вы работаете. Для этого используется разделитель точка — «.». Так, чтобы обратиться к атрибуту "attribute" таблицы "table", находящейся в базе данных "database1", нужно использовать запись:

```
`database1`.`table`.`attribute`
```

Если же вам понадобится обратиться к аналогичному атрибуту такой же таблицы, находящейся в БД "database2", запись станет такой:

```
`database2`.`table`.`attribute`
```

Для того, чтобы вводимые команды применялись к конкретной базе данных по умолчанию, можно воспользоваться командой USE и ввести название базы данных, с которой мы будем в дальнейшем работать:

```
USE `my_database`;
```

После выполнения команды USE следующие записи будут эквивалентны:

```
`my_database`.`table1`
`table1`
```

3. Удаление существующей базы данных выполняется командой DROP DATABASE, которая в качестве единственного аргумента принимает название удаляемой

базы данных. Например, чтобы удалить созданную вами в начале работы БД "my_database", нужно выполнить:

```
DROP DATABASE `my_database`;
```

После успешного удаления вы можете заново создать ее. **Учтите, что если бы в вашей базе данных были таблицы и данные, вся эта информация была бы утеряна навсегда.**

4. Для просмотра информации о базах данных, их таблиц, а также привилегий текущего пользователя, используется команда SHOW.

1. Увидеть список всех доступных пользователю баз данных можно с помощью команды:

```
SHOW DATABASES;
```

2. Увидеть список всех таблиц в используемой базе данных можно с помощью команды:

```
SHOW TABLES;
```

3. Для вывода информации о таблице "table_name" используйте команду:

```
SHOW CREATE TABLE `table_name`;
```

Обратите внимание, что результатом выполнения команды будет команда создания таблицы с учетом всех изменений, произведенных над таблицей в процессе работы с базой данных.

4. Увидеть список всех прав текущего пользователя СУБД можно с помощью команды:

```
SHOW GRANTS;
```

Команды для работы с таблицами БД

1. Для **создания таблицы** используется команда CREATE TABLE. В качестве аргумента ей передается название новой таблиц и перечисление всех атрибутов таблицы и их описаний (типы данных, значения по умолчанию, ограничения на применяемые значения и т.п.) — все перечисление берется в круглые скобки, а разделителем между атрибутами служит запятая. В общем виде команда выглядит так:

```
CREATE TABLE `table_name` (
 `название_первого_поля` его_тип [параметры],
 `название_второго_поля` его_тип [параметры],
 ...
 `название_последнего_поля` его_тип [параметры]
);
```

Обратите внимание, что после последнего атрибута запятая не требуется. Пример создания таблицы "news" с тремя атрибутами (идентификатор новости, время публикации, текст новости):

```
CREATE TABLE `news` (
  `id` MEDIUMINT(8) UNSIGNED PRIMARY KEY NOT NULL AUTO_INCREMENT,
  `posted` TIMESTAMP NOT NULL,
  `content` TEXT
);
```

2. Для **удаления таблицы** используется команда DROP TABLE с названием таблицы в качестве единственного аргумента. Например, для удаления созданной таблицы "news" команда будет выглядеть так:

```
DROP TABLE `news`;
```

3. Для **изменения таблиц** используется команда ALTER TABLE. Вид производимого изменения определятся последующими дополнительными командами:

3.1. **Переименование** таблицы осуществляется с помощью подкоманды RENAME. Например, чтобы переименовать таблицу "news" в "news_new", нужно выполнить следующую команду:

```
ALTER TABLE `news` RENAME TO `news_new`;
```

3.2. Для **добавления нового атрибута в таблицу** потребуется подкоманда ADD COLUMN, для которой нужно ввести название нового атрибута и указать его тип. Например, добавление к таблице "news" нового атрибута "author" (имя автора) будет выглядеть следующим образом:

```
ALTER TABLE `news` ADD COLUMN `author` VARCHAR(42);
```

Кроме того, можно задать положение добавляемого поля. Для этого в конец команды добавляется инструкция, указывающая, после какого столбца будет добавлено новое поле.

Например, команда добавления атрибута "author" в таблицу "news" с тем, чтобы "author" стал вторым полем, выглядит следующим образом:

```
ALTER TABLE `news` ADD COLUMN `author` VARCHAR(42) AFTER `id`;
```

Чтобы поле стало первым в таблице, нужно заменить конструкцию с "AFTER ..." на ключевое слово "FIRST". Команда добавления атрибута "author" в таблицу "news" в качестве первого поля будет выглядеть так:

```
ALTER TABLE `news` ADD COLUMN `author` VARCHAR(42) FIRST;
```

3.3. Для **изменения типа атрибута таблицы** служит подкоманда MODIFY, для которой нужно указать таблицу, название атрибута и заново перечислить все требуемые для него параметры. Например, чтобы изменить тип атрибута "author" таблицы "news" на CHAR(42), нужно выполнить следующую команду:

```
ALTER TABLE `news` MODIFY COLUMN `author` CHAR(42);
```

3.4. Для **удаления атрибута из таблицы** служит подкоманда DROP COLUMN. Например, команда удаления атрибута "author" из таблицы "news" выглядит так:

```
ALTER TABLE `news` DROP COLUMN `author`;
```

Команды для работы с данными таблиц баз данных

Все команды из двух предыдущих подразделов («Команды для манипулирования базами данных» и «Команды для работы с таблицами БД») относятся к языку DDL (Data Definition Language), являющемуся частью языка SQL, которая обеспечивает работу со схемой БД. Другая, не менее важная, часть SQL — язык DML (Data Manipulation Language), позволяющий манипулировать самими данными в таблицах баз данных. К нему относятся команды, речь о которых пойдет уже в этом подразделе.

1. Для **добавления строк в таблицу** используется команда INSERT. В качестве аргументов ей передается название таблицы и набор всех значений для одной из строк или набор из названий атрибутов и соответствующих им значений, подразумевая, что значения остальных атрибутов будут заполнены автоматически.

Пример добавления строки в таблицу "news" с 4 атрибутами ("id", "posted", "content", "author") с указанием всех значений:

```
INSERT INTO `news` VALUES (1, NOW(), "Текст новости", "Редактор");
```

Использованное в запросе выражение "NOW()" — это обращение к встроенной в MySQL функции, которая возвращает текущее время, Таким образом, в качестве значения "posted" для добавляемой строки запишется текущее время.

Пример добавления строки в таблицу "news" с 4 атрибутами ("id", "posted", "content", "author") с указанием только некоторых значений:

```
INSERT INTO `news` (`posted`, `content`) VALUES (NOW(), "Текст новости");
```

В данном случае для добавляемой строки мы задали только значения атрибутов "posted" и "content". В поле "id" автоматически запишется число, которое будет на единицу больше последнего значения "id" в таблице (или 1, если добавляемая строка — первая), а в поле "author" для этой строки запишется значение NULL — отметка об отсутствии значения (**не путайте ее с нулевым значением**).

2. Для **изменения данных в строке таблицы** используется команда UPDATE. В качестве аргументов ей передаются название таблицы, названия атрибутов, значения которых будут изменены, и новые значения этих атрибутов.

Например, для того, чтобы изменить значение поля "author" во всех строках таблицы "news" на "Пользователь", а значение поля "content" — на "Пустой текст", нужно выполнить следующую команду:

```
UPDATE `news` SET `author` = "Пользователь", `content` = "Пустой текст";
```

В данном случае будут изменены все строки таблицы, но зачастую нужно изменять значения только в определенных строках. Для этого предусмотрена возможность установления условия отбора тех строк (WHERE), для которых будут проведены изменения.

Например, чтобы теперь изменить значение поля "content" на "Новый текст" только для строки с "id", равным 1, в таблице "news", нужно выполнить следующую команду:

```
UPDATE `news` SET `content` = "Новый текст" WHERE id = 1;
```

Правила формирования условий запросов подробно описаны в следующем подразделе — «Команда выборки».

3. Для **удаления данных из таблицы** используется команда DELETE. В качестве аргумента ей передается только название таблицы. Кроме того, можно ограничить список удаляемых строк, задав условие с помощью WHERE (по аналогии с тем, как это делалось в команде UPDATE) — тогда СУБД выберет строки, соответствующие условию, и удалит из таблицы только их. Без указания условия будут удалены все строки из таблицы.

Например, чтобы удалить из таблицы "news" строку с "id", равным 1, нужно выполнить следующую команду:

```
DELETE FROM `news` WHERE id = 1;
```

Команда выборки данных

1. О выборке

Для получения значений атрибутов таблиц БД используется команда SELECT. Общий вид команды представляется следующим образом:

```
SELECT `my_field1`, `my_field2`, ..., `my_fieldN`
FROM `my_table`
WHERE условие;
```

Здесь:

- my_field1, my_field2 и т.д. — это перечисление названий атрибутов, значения которых мы "выбираем", т.е. в результирующей таблице будут выведены только значения указанных атрибутов. В случае, если требуется вывод значений всех атрибутов результирующей таблицы, для упрощения записи запроса используется символ звездочки («*»).
- my_table — это название таблицы, из которой будет сделана выборка.
- условие WHERE может иметь сложную структуру или отсутствовать.

Например, для выборки всех значений всех атрибутов из таблицы "news" достаточно ввести следующую команду:

```
SELECT * FROM `news`;
```

При выполнении запроса в оперативной памяти создается виртуальная таблица, состоящая из всех данных, удовлетворяющих условию отбора, а исходная таблица при этом никак не изменяется.

2. Условия выборки

Для решения некоторых задач условие отбора (WHERE) может иметь сложную структуру. Для построения таких условий используют логические операторы AND, OR, NOT и некоторые другие возможности языка SQL. Для группировки условий используются разделительные скобки («(» и «)»).

1. Для сравнения выражений предусмотрены: равенство («=»); больше или равно («>=»), меньше или равно («<=»), не равно («<>» или «!=»). Например, выборка значений атрибута "content" из таблицы "news", в строках которой значение атрибута "id" меньше 42, осуществляется следующим образом:

```
SELECT `content` FROM `news` WHERE `id` < 42;
```

2. Логическое умножение (И) записывается как AND и используется, когда требуется одновременное выполнение двух и более условий. Пример выборки значений атрибута "content" из таблицы "news", в строках которой значение атрибута "id" больше 21 и меньше 42:

```
SELECT `content` FROM `news` WHERE `id` > 21 AND `id` < 42;
```

3. Логическое сложение (ИЛИ) записывается как OR и используется, когда требуется, чтобы выполнялось хотя бы одно из нескольких условий. Пример выборки значений атрибута "content" из таблицы "news", в строках которой значение атрибута "id" больше 21 или меньше 10:

```
SELECT `content` FROM `news` WHERE `id` > 21 OR `id` < 10;
```

Логическое сложение имеет меньший приоритет чем логическое умножение, поэтому для корректной записи логических формул может потребоваться использование разделяющих скобок. Пример выборки значений атрибута "content" из таблицы "news", в строках которой значение атрибута "id" либо больше 21 и меньше 42, либо больше 84:

```
SELECT `content` FROM `news` WHERE ( `id` > 21 AND `id` < 42 ) OR `id` > 84;
```

4. Логическое отрицание (НЕ) записывается как NOT и используется для инвертирования последующего условия. Например, выборку значений атрибута "content" из таблицы "news", в строках которой значение атрибута "id" меньше 21 или больше 42, можно осуществить так:

```
SELECT `content` FROM `news` WHERE NOT ( `id` >= 21 AND `id` <= 42 );
```

5. Если у атрибута отсутствует значение, то оно записывается как NULL *(NULL — символ отсутствия значения, что не путать с пустым значением: пустое значение существует, а NULL указывает на его отсутствие)*. Для составления условий на выборку подобных отсутствующих значений предусмотрено специальное выражение IS NULL. Пример выборки значений атрибута "content" из таблицы "news", в строках которой значение атрибута "author" отсутствует:

```
SELECT `content` FROM `news` WHERE `author` IS NULL;
```

6. Для указания принадлежности значения атрибута какому-либо интервалу предусмотрено выражение BEETWEEN .. AND. Выражение "BETWEEN a AND b". Пример выборки значений атрибута "content" из таблицы "news", в строках которой значение атрибута "id" больше 21 и меньше 42:

```
SELECT `content` FROM `news` WHERE `id` BETWEEN 21 AND 42;
```

7. Для указания принадлежности значения атрибута какому-либо множеству предусмотрено выражение IN. Пример выборки значений атрибута "content" из таблицы "news", в строках которой значение атрибута "id" принимает значения 21, 42, 84 или 168:

```
SELECT `content` FROM `news` WHERE `id` IN (21, 42, 84, 168);
```

8. Для поиска строковых значений, содержащих заданную строку по шаблону, предусмотрена выражение LIKE. В качестве аргумента оператору LIKE передается шаблон в виде строки, в которой помимо текста могут содержаться метасимволы «_» (обозначает любой одиночный символ) и «%» (набор любых символов любой длины). Пример выборки значений атрибута "content" из таблицы "news", в строках которой значение атрибута "author" соответствует шаблону «_user%» (т.е. строка, которая строится как любой символ + user + любая последовательность символов):

```
SELECT `content` FROM `news` WHERE `author` LIKE "_user%";
```

Такому шаблону будут удовлетворять значения атрибута вроде «1user», «1user222», «xuser42» и т.д.

3. Сортировка и ограничение результатов

Для **ограничения количества результирующих строк**, удовлетворяющих условию отбора, используется выражение LIMIT, которое дописывается в конец запроса с максимальным числом строк для вывода в качестве простейшего аргумента.

Пример для выборки информации о 5 первых новостях из таблицы "news" со значениями атрибута "id" больше 5:

```
SELECT * FROM `news` WHERE `id` > 5 LIMIT 5;
```

Для **сортировки результата отбора** предусмотрен оператор ORDER BY. Он позволяет сортировать строки в результирующей таблице в соответствии со значениями выбранных атрибутов (если атрибутов несколько, то приоритет сортировки убывает при перечислении атрибутов слева направо). Поддерживается два типа сортировки: по возрастанию (ASC, этот режим используется по умолчанию) и по убыванию (DESC).

Пример для выборки информации о 10 первых новостях из таблицы "news" со значениями атрибута "id" больше 5, отсортированной сначала по авторам ("author") в алфавитном порядке, а затем — по идентификаторам ("id") в обратном порядке:

```
SELECT * FROM `news` WHERE `id` > 5 ORDER BY `author` ASC, `id` DESC LIMIT 10;
```

4. Числовые операции (агрегирующие функции)

1. Для получения среднего арифметического значения атрибута по всем результирующим строкам предусмотрена функция AVG(). В качестве единственного аргумента ей передается название атрибута, значения которого будут учитываться. Пример выборки для нахождения среднего значения идентификатора ("id") новости:

```
SELECT AVG(`id`) FROM `news`;
```

2. Для поиска минимального и максимального значений атрибута среди всех результирующих строк есть функции MIN() и MAX() соответственно. В качестве единственного аргумента им передается название атрибута, значения которого будут учитываться. Пример выборки для нахождения минимального и максимального значений идентификатора ("id") новости:

```
SELECT MIN(`id`), MAX(`id`) FROM `news`;
```

3. Для подсчета суммы числовых значений атрибута среди всех результирующих строк предусмотрена функция SUM(). В качестве единственного аргумента ей передается название атрибута, значения которого будут учитываться. Пример выборки для нахождения суммы значений идентификаторов ("id") всех новостей:

```
SELECT SUM(`id`) FROM `news`;
```

4. Для подсчета количества строк выборки предусмотрена функция COUNT(). В качестве единственного аргумента ей передается название атрибута, значения которого будут учитываться. Кроме того, поскольку обычно не важно, по какому атрибуту считать число результирующих строк, вместо названия атрибута в качестве аргумента может использоваться метасимвол звездочки («*»). Пример выборки для нахождения количества строк в таблице "news":

```
SELECT COUNT(*) FROM `news`;
```

Дополнительный параметр DISTINCT, указанный перед названием атрибута, позволяет вычесть из результата все вхождения по повторяющимся значениям выбранного

поля. Пример выборки для нахождения количества новостей, написанных разными авторами:

```
SELECT COUNT(DISTINCT `author`) FROM `news`;
```

5. Группировка результатов

Для разбиения результатов выборки по группам используется оператор GROUP BY. Разбиение строк по группам бывает необходимо для проведения каких-либо операций не над всеми строками по отдельности, а над группами, отобранными по какому-либо атрибуту и условию.

Группировка производится по указываемому атрибуту (или набору атрибутов), и строки распределяются по группам в соответствии со значением атрибута в этой строке (каждую группу образует набор строк с одним значением атрибута, по которому производится группировка).

Пример выборки значений всех атрибутов таблицы "news" с группировкой по атрибуту "author":

```
SELECT * FROM `news` GROUP BY `author`;
```

С помощью группировки можно, например, посчитать количество новостей, добавленных каждым автором:

```
SELECT `author`, COUNT(*) FROM `news` GROUP BY `author`;
```

Для добавления условий на результат группировки в конструкцию GROUP BY добавляют выражение HAVING, работающее по аналогии с WHERE, но с группами строк.

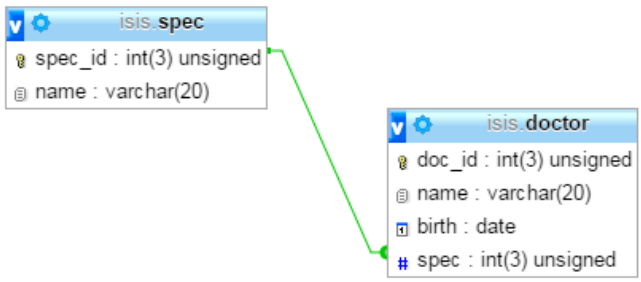
Например, к прошлому запросу с помощью HAVING можно добавить условие, по которому будут выводиться только те авторы, которые добавили более 3 новостей:

```
SELECT `author`, COUNT(*) AS `cnt` FROM `news` GROUP BY `author` HAVING `cnt` > 3;
```

Ключи

Рассмотрим базу данных из двух отношений.

- ***Специализация врача*** – ключ специализации *(первичный ключ)* и ее название;
- ***Доктор*** – ключевое поле, имя врача, дата рождения и специализация *(внешний ключ)*.



При создании таблицы ***spec*** укажем, что поле ***spec_id*** является *первичным ключом:*

```
CREATE TABLE `spec`(
`spec_id` INT(3) UNSIGNED PRIMARY KEY NOT NULL AUTO_INCREMENT,
`name` VARCHAR(20)
) ENGINE=InnoDB;
```

Обратите внимание, что таблица должна иметь тип InnoDB. Это связано с тем, что данный тип единственный в MySQL, поддерживающий механизмы транзакций и внешних ключей.

При создании таблицы ***doctor*** необходимо указать, что поле ***spec*** является внешним ключом.

В общем виде синтаксис создания внешнего ключа:

```
CONSTRAINT [symbol]] FOREIGN KEY
  [index_name] (index_col_name, ...)
  REFERENCES tbl_name (index_col_name,...)
  [ON DELETE reference_option]
  [ON UPDATE reference_option]
```

Где:

- **symbol** - имя ключа
- **index_name** - имя поля в таблице, которое мы хотим сделать ключом
- **tbl_name** - таблица, с которой осуществляем связывание
- **index_col_name** - имя поля, с которым связываем наш ключ
- **reference_option** – ограничения внешнего ключа

Ограничения reference_option:

RESTRICT | CASCADE | SET NULL | NO ACTION

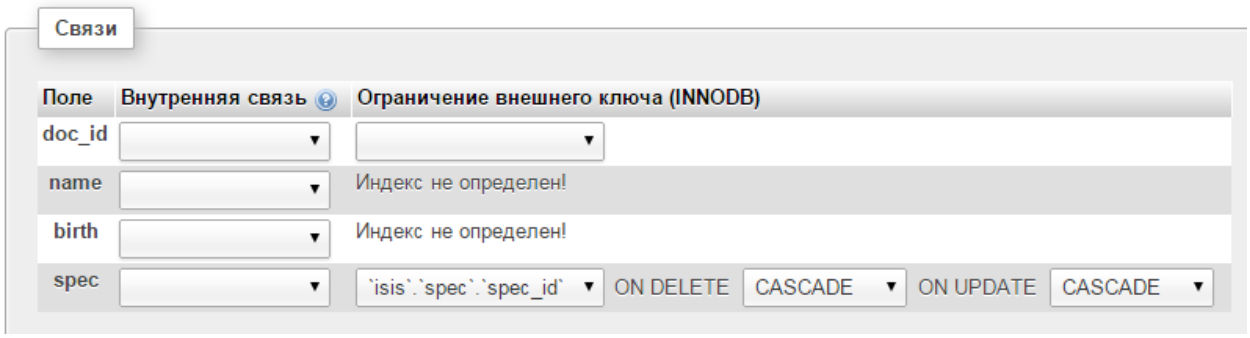
- **CASCADE**: если связанная запись родительской таблицы обновлена или удалена, и мы хотим чтобы соответствующие записи в таблицах-потомках также были обновлены или удалены. Что происходит с записью в родительской таблице, тоже самое произойдет с записью в дочерних таблицах.
- **SET NULL**:если запись в родительской таблице обновлена или удалена, а мы хотим чтобы в дочерней таблице некоторым занчениям было присвоено NULL (конечно если поле таблицы это позволяет)
- **NO ACTION**: смотри RESTRICT

- **RESTRICT**: если связанные записи родительской таблицы обновляются или удаляются со значениями, которые уже/еще содержатся в соответствующих записях дочерней таблицы, то база данных не позволит изменять записи в родительской таблице. Обе команды **NO ACTION** и **RESTRICT** эквивалентны отсутствию подвыражений **ON UPDATE** or **ON DELETE** для внешних ключей.

Итак, создадим таблицу ***doctor*** также типа InnoDB и укажем, что поле ***spec***-внешний ключ, а также зададим ограничения целостности ON DELETE CASCADE, ON UPDATE CASCADE.

```
CREATE TABLE `doctor` (
`doc_id` INT(3) UNSIGNED PRIMARY KEY NOT NULL AUTO_INCREMENT,
`name` VARCHAR(20),
`birth` DATE,
`spec` INT(3) UNSIGNED NOT NULL,
CONSTRAINT `spec` FOREIGN KEY(`spec`) REFERENCES `spec`(`spec_id`) ON DELETE CASCADE ON UPDATE CASCADE
) ENGINE=InnoBB;
```

В утилите phpMyAdmin ограничения внешнего ключа можно задать в редакторе связей:



Заполним таблицы данными:

Таблица ***spec***:

```
INSERT INTO `isis`.`spec` (`spec_id`, `name`)
VALUES (NULL, 'Терапевт'), (NULL, 'Кардиолог'), (NULL, 'Хирург');
```

Таблица ***doctor***:

```
INSERT INTO `isis`.`doctor` (`doc_id`, `name`, `birth`, `spec`)
VALUES (NULL, 'Иванов И.И.', '1972-02-02', '1'), (NULL, 'Жулин А.К.', '1949-03-19', '1'), (NULL, 'Голубев И.А.', '1968-06-29', '1'), (NULL, 'Терезникова Н.А.', '1955-09-17', '2'), (NULL, 'Самойлова О.Т.', '1978-12-13', '2');
```

spec_id	name
1	Терапевт
2	Кардиолог
3	Хирург

doc_id	name	birth	spec
1	Иванов И.И.	1972-02-02	1
2	Жулин А.К.	1949-03-19	1
3	Голубев И.А.	1968-06-29	1
4	Терезникова Н.А.	1955-09-17	2
5	Самойлова О.Т.	1978-12-13	2

Соединение таблиц

При разработке веб-проектов с участием базы данных нам часто необходимо в запросах объединять таблицы базы, чтобы получить необходимые данные.

Основные типы объединения таблиц в MySQL:

1. CROSS JOIN, он же INNER JOIN, он же JOIN
2. LEFT JOIN
3. RIGHT JOIN
4. NATURAL JOIN
5. Аналоги FULL OUTER JOIN для MySQL
6. Если не указать USING или ON в объединении (Декартова выборка)

В предложении **FROM** может быть указана **явная операция соединения** двух и более таблиц. Среди ряда операций соединения, описанных в стандарте языка SQL, многими серверами баз данных поддерживается только операция **соединения по предикату**. Синтаксис соединения по предикату имеет вид:

FROM <таблица 1>
[INNER]
{{LEFT | RIGHT | FULL } [OUTER]} JOIN <таблица 2>
[ON <предикат>]

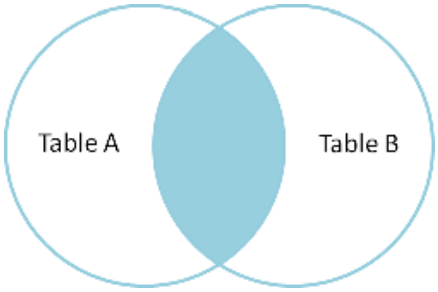
Соединение может быть либо внутренним (**INNER**), либо одним из внешних (**OUTER**). Служебные слова **INNER** и **OUTER** можно опускать, поскольку внешнее соединение однозначно определяется его типом — **LEFT** (левое), **RIGHT** (правое) или **FULL** (полное), а просто **JOIN** будет означать внутреннее соединение.

Предикат определяет условие соединения строк из разных таблиц. При этом **INNER JOIN** означает, что в результирующий набор попадут только те соединения строк двух таблиц, для которых значение предиката равно **TRUE**. Как правило, предикат определяет эквисоединение по внешнему и первичному ключам соединяемых таблиц, хотя это не обязательно.

INNER JOIN

- ✓ INNER JOIN производит выборку записей, которые только существуют в TableA и TableB одновременно.

- ✓ CROSS JOIN — это эквивалент INNER JOIN.
- ✓ INNER JOIN можно заменить условием объединения в WHERE.



Пример:

```
SELECT * FROM `spec` , `doctor`
WHERE `doctor`.`spec` = `spec`.`spec_id`
```

Идентичный запрос:

Условие соединения указывается при помощи ON.

```
SELECT * FROM `doctor`
INNER JOIN `spec`
ON `spec`.`spec_id` = `doctor`.`spec`
```

Если поле имеет одинаковое название в обеих таблицах, возможно использование **USING.**

```
…
INNER JOIN `таблица`
USING (`поле`);
```

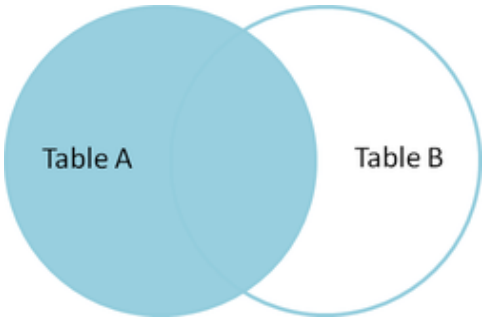
Результат:

spec_id	name	doc_id	name	birth	spec
1	Терапевт	1	Иванов И.И.	1972-02-02	1
1	Терапевт	2	Жулин А.К.	1949-03-19	1
1	Терапевт	3	Голубев И.А.	1968-06-29	1
2	Кардиолог	4	Терезникова Н.А.	1955-09-17	2
2	Кардиолог	5	Самойлова О.Т.	1978-12-13	2

Таким образом, соединяются только те строки таблицы, у которых в указанных столбцах находятся равные значения (***эквисоединение***). В таблице нет ни одного доктора со специализацией 3, т.е. хирург, поэтому **INNER JOIN** для него не сработал. Если разные таблицы имеют столбцы с одинаковыми именами, то для однозначности требуется использовать точечную нотацию, которая называется уточнением имени столбца: **<имя таблицы>.<имя столбца>.** В тех случаях, когда это не вызывает неоднозначности, использование данной нотации не является обязательным.

LEFT JOIN

LEFT OUTER JOIN (LEFT JOIN) указывает, что левая таблица управляющая (в нашем случае TableA) и производит из нее полную выборку, осуществляя поиск соответствующих записей в таблице TableB. Если таких соответствий не найдено, то база вернет пустой указатель - NULL. Указание OUTER - необязательно.



Запрос:

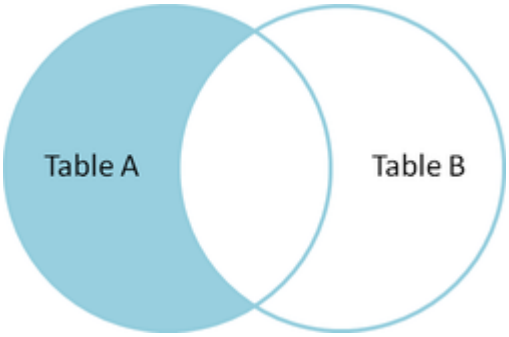
```
SELECT * FROM `spec`
LEFT JOIN `doctor`
ON `spec`.`spec_id` = `doctor`.`spec`
```

Результат:

spec_id	name	doc_id	name	birth	spec
1	Терапевт	1	Иванов И.И.	1972-02-02	1
1	Терапевт	2	Жулин А.К.	1949-03-19	1
1	Терапевт	3	Голубев И.А.	1968-06-29	1
2	Кардиолог	4	Терезникова Н.А.	1955-09-17	2
2	Кардиолог	5	Самойлова О.Т.	1978-12-13	2
3	Хирург	*NULL*	*NULL*	*NULL*	*NULL*

Для полей таблицы **spec** (терапевта и кардиолога) были найдены соответствия в таблице **doctor**, а для хирурга не было найдено ни одного соответствия, поэтому был возвращен NULL.

Чтобы произвести выборку записей из таблицы TableA, которых не существует в таблице TableB, мы выполняем LEFT JOIN, но затем из результата исключаем записи, которые не хотим видеть, путем указания, что TableB.id является нулем (указывая, что записи нет в таблице TableB).



Запрос:

```
SELECT *
FROM `spec`
LEFT JOIN `doctor` ON `spec`.`spec_id` = `doctor`.`spec`
WHERE `doctor`.`spec` IS NULL
```

Результат:

spec_id	name	doc_id	name	birth	spec
3	Хирург	*NULL*	*NULL*	*NULL*	*NULL*

RIGHT JOIN

RIGHT JOIN выполняет те же самые функции, что и LEFT JOIN, за исключением того, что правая таблица будет прочитана первой. Таким образом, если в предыдущих запросах LEFT заменить на RIGHT, то таблица результатов, грубо говоря, отразится по вертикали. То есть, в результате вместо значений TableA будут записи TableB и наоборот.

NATURAL JOIN

Суть этой конструкции в том, что база сама выбирает, по каким столбцам сравнивать и объединять таблицы. А выбор этот падает на столбцы с одинаковыми именами. С одной стороны, это весьма удобно, но с другой стороны, база может выбрать совершенно не те столбцы для объединения и запрос будет работать совершенно не так, как вы предполагали. Нет гарантии того, что столбцы с одинаковыми именами в таблицах будут именно ключевыми и предназначены для объединения. **NATURAL JOIN** ухудшает читаемость ida, так как разработчик не сможет по запросу определить, как объединяются таблицы. Поэтому, обращая внимание на такие факторы, **NATURAL JOIN использовать не рекомендуется.** Рассмотрим примеры.

Запрос:

```
SELECT * FROM `doctor`
NATURAL JOIN `spec`
```

В этом случае СУБД выполняет объединение таблиц по столбцу ***name***, так как он присутствует в обеих таблицах, что идентично следующему запросу:

```
SELECT * FROM `doctor`
INNER JOIN `spec`
USING ( `name` )
```

Результат:

Поскольку не существует записей с одинаковым полем name одновременно в обеих таблицах, запрос вернет пустой результат.

MySQL вернула пустой результат (т.е. ноль строк). (Запрос занял 0.0004 сек.)

Если попробовать изменить запрос следующим образом.

Запрос:

```
SELECT `spec` . * , `doctor` . *
FROM `doctor`
NATURAL RIGHT JOIN `spec`
```

Такой запрос приводится к следующему:

```
SELECT `spec` . * , `doctor` . *
FROM `doctor`
RIGHT JOIN `spec`
USING ( `name` )
```

Результат:

spec_id	name	doc_id	name	birth	spec
1	Терапевт	*NULL*	*NULL*	*NULL*	*NULL*
2	Кардиолог	*NULL*	*NULL*	*NULL*	*NULL*
3	Хирург	*NULL*	*NULL*	*NULL*	*NULL*

Происходит это так: так как правая таблица управляющая (т.е. таблица ***spec***), то она читается первой и полностью выбирается, независимо от левой (***doctor)*** таблицы; когда начинается поиск соответствующих записей в левой таблице, то СУБД не находит ни одной записи, которая была бы идентична по ***name***, поэтому возвращаются пустые указатели.

Для более подробного понимания работы NATURAL JOIN изменим имя доктора на Терапевт, а затем выполним те же самые запросы.

```
UPDATE `isis`.`doctor` SET `name` = 'Терапевт' WHERE `doctor`.`doc_id` =1;
```

Запрос:

```
SELECT * FROM `doctor`
NATURAL JOIN `spec`
```

Результат:

name	doc_id	birth	spec	spec_id
Терапевт	1	1972-02-02	1	1

Выводится одна запись, поскольку теперь в обеих таблицах существует запись с одинаковым значением поля ***name.***

Запрос:

```
SELECT `spec` . * , `doctor` . *
FROM `doctor`
NATURAL RIGHT JOIN `spec`
```

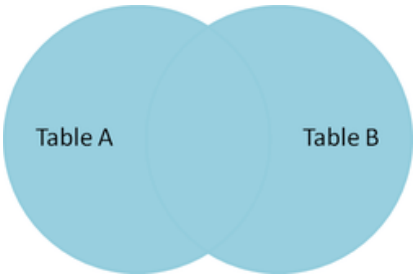
Возвращает результат:

spec_id	name	doc_id	name	birth	spec
1	Терапевт	1	Терапевт	1972-02-02	1
2	Кардиолог	*NULL*	*NULL*	*NULL*	*NULL*
3	Хирург	*NULL*	*NULL*	*NULL*	*NULL*

FULL OUTER JOIN

***Недоступно в MySQL**

FULL OUTER JOIN производит выборку всех записей из TableA и TableB, вне зависимости есть ли соответствующая запись в соседней таблице. Если таковой нет, то недостающая сторона будет содержать пустой указатель и результатом будет выводится NULL.



Декартова выборка

Если при объединении таблиц не указать условие объединения через ON или USING, то база произведет так называемую Декартову выборку, когда значению одной таблицы приравнивается каждое значение другой. Таким образом, СУБД, в нашем случае, возвращает 3x5 = 15 строк.

Запрос:

```
SELECT *
FROM `doctor`
JOIN `spec`
```

Результат:

doc_id	name	birth	spec	spec_id	name
1	Иванов И.И.	1972-02-02	1	1	Терапевт
1	Иванов И.И.	1972-02-02	1	2	Кардиолог
1	Иванов И.И.	1972-02-02	1	3	Хирург
2	Жулин А.К.	1949-03-19	1	1	Терапевт
2	Жулин А.К.	1949-03-19	1	2	Кардиолог
2	Жулин А.К.	1949-03-19	1	3	Хирург
3	Голубев И.А.	1968-06-29	1	1	Терапевт
3	Голубев И.А.	1968-06-29	1	2	Кардиолог
3	Голубев И.А.	1968-06-29	1	3	Хирург
4	Терезникова Н.А.	1955-09-17	2	1	Терапевт
4	Терезникова Н.А.	1955-09-17	2	2	Кардиолог
4	Терезникова Н.А.	1955-09-17	2	3	Хирург
5	Самойлова О.Т.	1978-12-13	2	1	Терапевт
5	Самойлова О.Т.	1978-12-13	2	2	Кардиолог
5	Самойлова О.Т.	1978-12-13	2	3	Хирург

Объединение запросов

UNION

Объединением двух множеств называется множество всех элементов, принадлежащих какому-либо одному или обоим исходным множествам. Результатом **объединения** будет множество, состоящее из всех строк, входящих в какое-либо одно или в оба первоначальных отношения. Однако, если этот результат сам по себе должен быть другим отношением, а не просто разнородной смесью строк, то два исходных отношения должны быть совместимыми по объединению, т. е. строки в обоих отношениях должны быть одной и той же формы. Что касается **SQL**, то две таблицы совместимы по объединению (и к ним может быть применен оператор объединения **UNION**) тогда и только тогда, когда:

- Они имеют одинаковое число полей (например, m);
- Для всех i (i = 1, 2, …, m) i-е поле первой таблицы и i-е поле второй таблицы имеют в точности одинаковый тип данных.

Пример:

```
(
SELECT * FROM `doctor`
WHERE `name` LIKE '%ов%'
)
UNION
(
SELECT * FROM `doctor`
WHERE `name` LIKE 'Ж%'
)
```

В данном абстрактном примере **UNION** используется для объединения результатов работы нескольких команд SELECT в один набор результатов. Круглые скобки используются для визуального разделения запросов и обязательны при использовании операторов ORDER BY и др.

Результат:

doc_id	name	birth	spec
1	Иванов И.И.	1972-02-02	1
4	Терезникова Н.А.	1955-09-17	2
5	Самойлова О.Т.	1978-12-13	2
2	Жулин А.К.	1949-03-19	1

UNION ALL и UNION DISTINCT

Если не используется ключевое слово ALL для UNION, все возвращенные строки будут уникальными, так как по умолчанию подразумевается DISTINCT для всего результирующего набора данных. Если указать ключевое слово ALL, то результат будет содержать все найденные строки из всех примененных команд SELECT.

Пример:

```
(
SELECT *
FROM `doctor`
WHERE `name` LIKE '%ов%'
)
UNION ALL (
SELECT *
FROM `doctor`
WHERE `name` LIKE 'И%'
)
```

Результат:

doc_id	name	birth	spec
1	Иванов И.И.	1972-02-02	1
4	Терезникова Н.А.	1955-09-17	2
5	Самойлова О.Т.	1978-12-13	2
1	Иванов И.И.	1972-02-02	1

Подзапросы

При работе с базой данных может возникнуть потребность в запросе, который зависел бы от результата другого запроса. Подзапрос - это запрос, результат которого используется в условии другого запроса. Основные преимущества подзапросов:

- Они позволяют писать *структурированные* запросы таким образом, что можно изолировать части оператора.
- Они представляют альтернативный способ выполнения операций, которые требуют применения сложных соединений и слияний (JOIN и UNION).
- По мнению многих, они более читабельны.

Обратите внимание, что такого рода подзапросы не могут использоваться с SELECT *, поскольку они выбирают лишь одиночный столбец.

Пример:

```
SELECT * FROM `doctor`
WHERE `spec` = ( SELECT `spec_id` FROM `spec` WHERE `name` LIKE 'К%' )
```

Результат:

	doc_id	name	birth	spec
Изменить Копировать Удалить	4	Терезникова Н.А.	1955-09-17	2
Изменить Копировать Удалить	5	Самойлова О.Т.	1978-12-13	2

Таким образом, выполняется выборка всех врачей со специализацией, начинающейся на букву К. Подзапрос используется для поиска spec_id, т.е. идентификатора специализации на букву К. Затем

найденный идентификатор служит условием поиска врачей с такой специализацией. На подзапросы можно налагать дополнительные ограничения, такие как:

- EXISTS
- SOME/ANY
- ALL

EXISTS

Используется, чтобы указать предикату, производит ли подзапрос вывод. Возвращает булево значение.

Пример:

```
SELECT * FROM `doctor`
WHERE EXISTS (
SELECT *
FROM `spec`
WHERE `name` LIKE 'Хирург'
)
```

Результат:

	doc_id	name	birth	spec
Изменить Копировать Удалить	1	Иванов И.И.	1972-02-02	1
Изменить Копировать Удалить	2	Жулин А.К.	1949-03-19	1
Изменить Копировать Удалить	3	Голубев И.А.	1968-06-29	1
Изменить Копировать Удалить	4	Терезникова Н.А.	1955-09-17	2
Изменить Копировать Удалить	5	Самойлова О.Т.	1978-12-13	2

Оператор EXISTS во внешнем предикате отмечает, что извлекать данные о врачах следует только в случае, если существует запись Хирург. В противном случае будет возвращен пустой результат.

SOME/ANY

Операторы SOME/ANY (взаимозаменяемые — различие в терминологии состоит в том, чтобы позволить людям использовать тот термин, который наиболее однозначен).

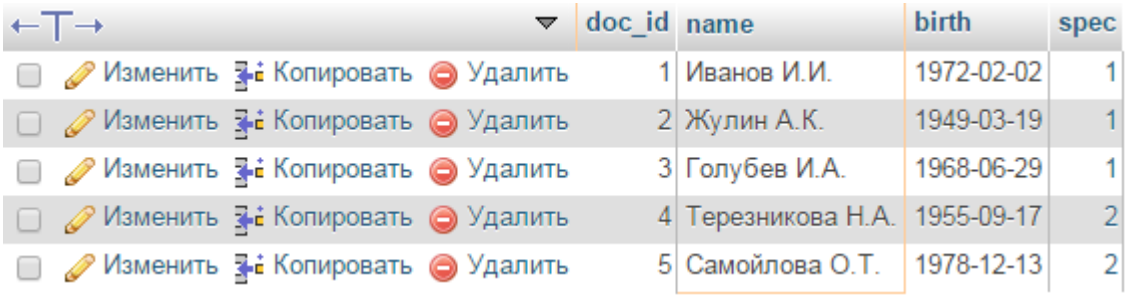
Оператор ANY берет все значения выведенные подзапросом, причем такого же типа, как и те, которые сравниваются в основном предикате. В этом его отличие от EXISTS, который просто определяет, производит ли подзапрос результаты, и фактически не использует эти результаты.

Пример:

```
SELECT * FROM `doctor`
WHERE `spec` = ANY(

SELECT `spec_id`
FROM `spec`
WHERE `spec_id` <3
)
```

Результат:



	doc_id	name	birth	spec
Изменить Копировать Удалить	1	Иванов И.И.	1972-02-02	1
Изменить Копировать Удалить	2	Жулин А.К.	1949-03-19	1
Изменить Копировать Удалить	3	Голубев И.А.	1968-06-29	1
Изменить Копировать Удалить	4	Терезникова Н.А.	1955-09-17	2
Изменить Копировать Удалить	5	Самойлова О.Т.	1978-12-13	2

В данном случае оператор ANY берет все значения, выведенные подзапросом (все записи с идентификатором < 3) и оценивает их как верные, если любой из них равняется идентификатору текущей строки внешнего запроса.

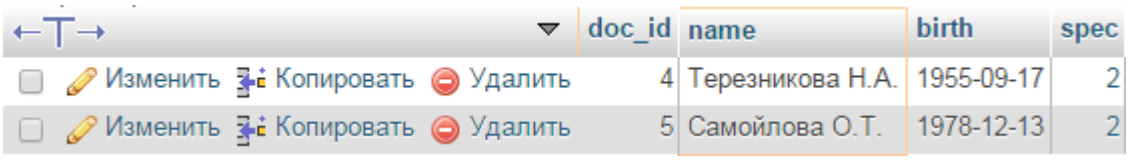
ALL

Предикат является верным, если каждое значение, выбранное подзапросом, удовлетворяет условию в предикате внешнего запроса.

Пример:

```
SELECT * FROM `doctor`
WHERE `spec` = ALL (
SELECT `spec_id`
FROM `spec`
WHERE `name` LIKE '%or'
)
```

Результат:



	doc_id	name	birth	spec
Изменить Копировать Удалить	4	Терезникова Н.А.	1955-09-17	2
Изменить Копировать Удалить	5	Самойлова О.Т.	1978-12-13	2

Данный запрос позволяет найти всех врачей, у которых название специализации заканчивается на «ог».

Представления

Представление — запрос на выборку, сохраненный в базе данных под каким-то названием, виртуальная таблица. Данные представления динамически рассчитываются по запросу из данных реальных таблиц, но структура (поля) результата запроса не меняются при изменении исходных таблиц.

Плюсы использования виртуальных таблиц:

1. Повышение скорости работы. Когда прикладной программе требуется таблица с определённым набором данных, она делает простейший запрос из подготовленного представления. Поскольку SQL-запрос, выбирающий данные представления, зафиксирован на момент его создания, СУБД получает возможность применить к этому запросу

оптимизацию или предварительную компиляцию, что положительно сказывается на скорости обращения к представлению по сравнению с прямым выполнением того же запроса из прикладной программы.

2. Независимая модификация прикладной программы и схемы хранения данных.
3. Повышение безопасности. За счет предоставления пользователю только тех данных, на которые он имеет права — от него скрыта реальная структура таблиц базы данных.

Может возникнуть такая ситуация, что в исходных таблицах есть обязательные и не пустые (NOT NULL) поля, а в представлении их нет. Тогда операция добавления строки данных не может быть выполнена за одно действие. Во избежание потерь данных подобных действий следует избегать.

Работа с представлениями:

Создание:

```
CREATE VIEW `name` [ FIELDS ] AS { запрос };
```

Удаление:

```
DROP VIEW `name`;
```

Изменение:

```
ALTER VIEW `name` [ FIELDS ] AS { новый запрос }
```

Задания к лабораторной работе

Общие правила выполнения заданий

Требования к ПО:

- Рекомендуется установка в качестве среды разработки openServer.
- Для работы с БД рекомендуется использовать MySQL Workbench, phpMyAdmin, консоль сервера MySQL и др.

Ход выполнения работы:

1. Выполните проектирование логической и физической модели данных.
2. Создайте необходимые отношения
3. Укажите ограничения атрибутов при необходимости
4. Реализуйте связи между таблицами с использованием PRIMARY и FOREIGN KEYS. Укажите корректные ограничения внешнего ключа.
5. Заполните таблицы данными.
6. Выполните запросы.

Требования и рекомендации по выполнению:

- Все запросы выполняются на SQL.
- Первичный ключ каждой таблицы должен иметь имя **id** для корректного выполнения последующих лабораторных работ.
- В описании проектов при перечислении атрибутов не полностью указана информация о ключах таблицы. Необходимо руководствоваться правилами и рекомендациями по выполнению заданий.
- Структура базы данных, набор отношений и перечень атрибутов могут быть дополнены и скорректированы проектировщиком;
- В перечне запросов указан общий вид запроса. В случае отсутствия указанных позиций, наименований в таблицах БД при задании условия поиска, они могут быть заменены на любые другие аналогичные.
- При выводе результатов запросов не должно выводиться идентификаторов.
- Удаление должно быть исключительно логическим, а не физическим. Физически данные из базы данных не удаляются, а «удаляемые» записи лишь помечаются специальным флагом состояния.
- Запросы на создание представлений будут использованы в последующих заданиях, на данный момент они могут содержать избыточные или дублирующие данные.
- Все выполненные запросы сохраняются в виде скриншотов (особенно запросы по созданию отношений, по наполнению данными и организации связей).

Правила оценивания:

- Лабораторная работа оценивается из 100%. В случае пересдачи максимально возможный балл – 60%.
- В случае нарушения дедлайнов лабораторная работа оценивается в 1%.

Вариант 1. Проект ПОСТАВКА ТОВАРОВ

Завод "Прогресс" поставляет товары (изделие А, изделие В, изделие С и др.) заказчикам по договорам. Для каждого товара определены планы поставок. Необходимо спроектировать базу данных **ПОСТАВКА ТОВАРОВ**, информация которой будет использоваться для анализа выполнения заводом планов поставок. В БД должна храниться информация о:

— ТОВАРАХ: *id товара*, *наименование товара*, *цена товара* (тыс. руб.);
— ЗАКАЗАХ на поставку товаров: *id заказа*, *наименование заказчика*, *адрес заказчика*, *телефон, номер договора, дата заключения договора*, *наименование товара, плановая поставка* (шт.);
— фактических ОТГРУЗКАХ товаров: *id отгрузки*, *id заказа*, *дата отгрузки, отгружено товара* (шт.).

При проектировании БД необходимо учитывать следующее:

— товар имеет *несколько* заказов на поставку. Заказ соответствует *одному* товару;
— товару могут соответствовать *несколько* отгрузок. В отгрузке могут участвовать *несколько* товаров.

Кроме того, следует учесть:

— товар *не обязательно* имеет заказ. Каждому заказу *обязательно* соответствует товар;
— товар *не обязательно* отгружается заказчику. Каждая отгрузка *обязательно* соответствует некоторому товару.

Перечень запросов для реализации (общий вид):

1. Вывести перечень заказчиков, в названии которых содержится «ООО», «АОА» или «ИП»
2. Вывести список товаров, отгруженных за текущий месяц
3. Определить количество отгруженного товара (шт) за первый квартал текущего года
4. Вывести товар, заказанный по адресу пл. Гагарина 1
5. Найти товар, который не был отгружен
6. Определить самый дорогой товар, заказанный компанией с наименованием ООО «Альянс-ЮГ»
7. Вывести список товаров, заказанный компанией с наименованием ООО «Альянс-ЮГ», отсортированный в порядке совершения заказа (от самого нового к самому первому)
8. Выведите таблицу вида «Наименование товара», «Цена товара», «Наименование заказчика», «Адрес заказчика». Товар, который не был заказан, также должен присутствовать в итоговой выборке.
9. Создайте представление на хранение данных «Ведомость поставок товаров заводом «Прогресс»: «Наименование заказчика», «Цена товара», «Плановая поставка», «Фактически отгружено», «Отклонение».
10. Удалите товар, стоимость которого 5000-12000 рублей

Вариант 2. Проект РОЗНИЧНАЯ ТОРГОВЛЯ

Магазин розничной торговли продает персональные компьютеры, средства связи и периферийное оборудование: принтеры, накопители и др.

Необходимо спроектировать базу данных **РОЗНИЧНАЯ ТОРГОВЛЯ**, информация которой будет использоваться для анализа продаж в магазине.

В БД должна храниться информация:

— о ТОВАРАХ: *id товара*, *наименование товара, дата поступления в магазин,количество товара, цена закупки* (руб.);
— ПОСТАВЩИКАХ товаров: *id поставщика*, *наименование поставщика*, *адрес,телефон, к кому обращаться*;
— ПРОДАЖАХ товаров в магазине: *id продажи*, *id товара*, *дата продажи*, *количество проданного товара* (шт.), *цена розничная* (руб.).

При проектировании БД необходимо учитывать следующее:

— поставщик поставляет *несколько* товаров. Товар поступает на склад магазина от *нескольких* поставщиков;
— товар имеет *несколько* продаж. Продажа относится к *одному* товару.

Кроме того, следует учесть:

— поставщик *не обязательно* поставляет товар (может временно не работать).Каждый товар *обязательно* поставляется;
— товар *не обязательно* продается. Каждая продажа *обязательно* связана с товаром.

Перечень запросов для реализации (общий вид):

1. Вывести перечень поставщиков, расположенных по адресу в г. Москва
2. Вывести список товаров, проданных за сегодняшний день
3. Определить выручку проданного товара за февраль текущего года
4. Вывести товар, который поступил в первом квартале этого года и был продан хотя бы 1 раз
5. Найти товар, который ни разу не был продан
6. Определить самый популярный товар (т.е. тот товар, который продавался чаще всего)
7. Вывести список товаров, поставляемый Mi SHOP, отсортированный от самого дорого до самого дешевого
8. Выведите таблицу вида «Наименование поставщика», «Наименование товара». Поставщики, которые временно не работают (не поставляют товар), также должны присутствовать в итоговой выборке
9. Создайте представление на хранение данных «Отчет о продажах товаров розничным магазином»: «Наименование товара», «Наименование заказчика», «Дата продажи», «Цена», «Количество», «Сумма»
10. Удалите поставщика ООО «Рога и копыта»

Вариант 3. Проект БАНКОВСКИЕ ВКЛАДЫ

Клиентам банка предлагается размещать денежные средства на срочные банковские вклады различных видов

Таблица «Срочные банковские вклады»

Наименование вклада	Срок хранения, мес.	Ставка, % годовых
Накопительный (в рублях)	13	10 %
Капитал (в долларах США)	18	7 %
Победа (в рублях)	13	Ставка рефинансирования + 2 %
К отпуску (в рублях)	12	Ставка рефинансирования – 1 %

Необходимо спроектировать базу данных **БАНКОВСКИЕ ВКЛАДЫ**, информация которой будет использоваться для анализа работы с клиентами по вкладам.

В БД должна храниться информация:

— о ВКЛАДАХ, которые предоставляет банк: *id вклада; наименование вклада; срок хранения* (месяцев); *ставка, % годовых*;
— КЛИЕНТАХ, которые помещают денежные средства на вклады: *id клиента,Ф.И.О. клиента, номер паспорта, адрес, телефон*;
— СЧЕТАХ клиентов банка: *№ счета, id клиента, id вклада, дата открытиясчета, дата закрытия счета, сумма вложенная* (руб.).

При проектировании БД необходимо учитывать следующее:

— клиент банка может помещать свои средства на *несколько* счетов. Счет открывается на *одного* клиента;
— каждый вид вклада связан с *несколькими* счетами клиентов. Счет относитсяк *одному* виду вклада.

Кроме того, следует учесть:

— каждый клиент *обязательно* имеет счет в банке. Каждый счет *обязательно* принадлежит клиенту;
— вклад некоторого вида *не обязательно* может быть связан со счетами клиентов. Каждый счет клиента *обязательно* связан с некоторым видом вклада.

Перечень запросов для реализации (общий вид):

1. Вывести перечень вкладов, накопительная ставка по которым более 10%
2. Определить клиентов, сумма на счете у которых более 1000000 рублей
3. Найти клиентов, закрывших счет в течение 10 дней после открытия
4. Вывести наименования вкладов, открытых Ивановым И.И.
5. Определить наименования вкладов, которые не были открыты ни одним клиентом
6. Отсортировать клиентов по вложенной сумме (от меньшей к большей)
7. Выведите таблицу вида «Наименование вклада», «Номер счета», «id клиента». Вклады, которые не связаны со счетами, также должны быть в таблице.
8. Создайте представление на хранение данных «Ведомость получения доходов клиентами банков по закрытым счетам»: «Наименование вклада», «Срок хранения», «Ставка % годовых», «Сумма вложенная», «Сумма накопления», «ФИО клиента» *(Примечание. "Сумма накопления, руб." = " Сумма вложенная, руб." x ("Срок хранения, месяцев" x "Ставка, % годовых" **:** 12) **:** 100.*
9. Измените процент ставки для вклада с id=3.
10. Удалите клиентов, закрывших счет более года назад.

Вариант 4. Проект АУДИТ

Сотрудники разных категорий (1-й, 2-й, 3-й) аудиторской фирмы осуществляют проверку предприятий в течение года. Работа сотрудников оплачивается за каждый час в зависимости от категории сотрудника.

Необходимо спроектировать базу данных **АУДИТ,** информация которой будет использоваться для автоматизации начисления зарплаты сотрудникам.

В БД должна храниться информация:

— о СОТРУДНИКАХ аудиторской фирмы: *id сотрудника*, *Ф.И.О. сотрудника*, *номер паспорта*, *дата рождения, рабочий телефон*;
— КАТЕГОРИЯХ сотрудников: *категория*, *ставка за 1 час* (тыс. руб.);
— выполненной сотрудниками РАБОТЕ на предприятиях: *название предприятия*, *id сотрудника*, *дата выполнения работы, количество отработанных часов.*

При проектировании БД необходимо учитывать следующее:

— определенную категорию могут иметь *несколько* сотрудников. Сотрудник имеет квалификацию только *одной* категории;
— сотрудник может выполнять *несколько* работ по проверке на разных предприятиях. Работу по проверке предприятия могут проводить *несколько* сотрудников.

Кроме того, следует учесть:

— каждый сотрудник *обязательно* имеет категорию. Определенная категория *не обязательно* может быть у сотрудников аудиторской фирмы;
— сотрудник *не обязательно* выполняет работу по проверке на предприятии. Каждая работа по проверке на предприятии *обязательно* выполняется сотрудниками.

Перечень запросов для реализации (общий вид):

1. Вывести список категорий, ставка за 1 час по которым составляет от 500 до 1000 рублей.
2. Подсчитать количество сотрудников каждой категории.
3. Найти категории, в которых нет ни одного сотрудника.
4. Вывести все работы, которые выполнял сотрудник Петров В.В.
5. Подсчитать количество часов, которые отработал Петров В.В.
6. Определить виды работ, которые Петров В.В. осуществлял в текущем месяце
7. Вывести таблицу вида «Ф.И.О. сотрудника», «Категория». В списке должны быть категории, даже если к ним не относится ни один сотрудник.
8. Создайте представление на хранение данных «Ведомость начислений сотрудникам аудиторской фирмы»: «ФИО», «Категория», «Ставка», «Название предприятия», «Дата выполнения работы», «Количество отработанных часов», «Начислено за выполненную работу»
9. Увеличьте ставку за 1 час старшего аудитора на 10%
10. Удалите сотрудников, родившихся до 1955 года.

Вариант 5. Проект ФАБРИКА

На склад готовой продукции фабрики по пошиву одежды в течение года поступают производимые ею товары разных моделей.

Необходимо спроектировать базу данных **ФАБРИКА**, информация кото рой будет использоваться для учета готовой продукции, хранящейся на складе.

В БД должна храниться информация:

— о видах ТОВАРА (пальто женское, костюм женский и др.): *id товара, наименование товара*;
— МОДЕЛЯХ одежды, выпускаемой фабрикой: *id модели, наименование модели, id товара, цена модели* (тыс. руб.);
— ПОСТУПЛЕНИЯХ на склад: *id поступления, id модели, дата поступления модели товара, количество* (шт.), *кто принял товар*.

При проектировании БД необходимо учитывать следующее:

— товар имеет *несколько* моделей. Модель связана с *одним* товаром;
— модель связана с *несколькими* поступлениями. Поступление связано с *одной* моделью товара.

Кроме того, следует учесть:

— каждый вид товар *обязательно* имеет модели. Каждая модель *обязательно* связана с определенным товаром;
— модель *обязательно* связана с поступлениями на склад. Поступление *обязательно* связано с моделью товара.

Перечень запросов для реализации (общий вид):

1. Вывести список моделей категории «пальто женское», цена которых от 8000 до 13000 тыс. рублей.
2. Определить количество моделей каждого вида
3. Определить виды, в которых не менее 5 моделей
4. Вывести модели товара категории футболка, поступившие в текущем месяце
5. Найти виды товара, в которых нет ни одной модели.
6. Найти модель с самой высокой стоимостью
7. Вывести список моделей категории «пальто женское», отсортированный от самого дорогого к самому дешевому
8. Выведите таблицу вида «Наименование товара», «Наименование модели», «Цена модели». Виды товара, к которым не относится ни одна модель, также должны быть отражены.
9. Создайте представление на хранение данных «Ведомость поступления на склад готовой продукции фабрики по пошиву одежды»: «Вид товара», «Модель», «Дата поступления модели», «Цена модели», «Количество», «Сумма»
10. Удалите все товары вида «носки»

Вариант 6. Проект ТУРАГЕНСТВО

Работники турагенства продают путевки путешествий по разным странам. В каждую страну организуются несколько маршрутов. По каждому маршруту указывается цель путешествия (отдых, экскурсия, лечение, шоп-тур, обучение и др.).

Необходимо спроектировать базу данных **ТУРАГЕНСТВО**, информация которой позволит определять наиболее популярные маршруты за текущий год, отслеживать обращения клиентов и др.

В БД должна храниться информация:

— о СТРАНАХ: *id страны, название страны, стоимость визы* (руб.);
— МАРШРУТАХ: *id страны, id маршрута, наименование маршрута*;
— ПРОДАЖАХ: *id маршрута, цель путешествия, цена путевки* (руб.), *количество проданных путевок по маршруту, дата продажи*.

При проектировании БД необходимо учитывать следующее:

— в каждую страну организуются *несколько* маршрутов. Маршрут имеет отношение только к *одной* стране;
— маршрут участвует в *нескольких* продажах. Продажа связана только с *одним* маршрутом.

Кроме того следует учесть:

— по каждой стране *обязательно* организуется маршрут. Каждый маршрут *обязательно* имеет отношение к некоторой стране;
— маршрут *не обязательно* может участвовать в продаже (может быть невостребован). Каждая продажа *обязательно* связана с одним маршрутом.

Перечень запросов для реализации (общий вид):

1. Найти список стран, стоимость визы в которые от 8000 до 12000 тыс. рублей
2. Найти страну, виза в которую является самой дорогой
3. Определить наименования маршрутов, реализуемых турагенством в Италии
4. Узнать, сколько маршрутов организуется в Испанию
5. Вывести список стран, в которые были проданы маршруты в 1 квартале текущего года
6. Определить, какие маршруты являются невостребованными
7. Вывести таблицу вида «Цель маршрута», «Цена путевки», «Наименование маршрута». Должно быть в том числе и невостребованные маршруты
8. Узнать, в какие страны организуются шоп-туры
9. Создайте представление на хранение данных «Ведомость реализации путевок турагентством»: «Страна», «Стоимость визы», «Дата продажи», «Наименование маршрута», «Цена путевки», «Количество проданных путевок», «Стоимость проданных путевок»
10. Увеличьте стоимость визы в Италию на 15%

Вариант 7. Проект ТОРГОВЛЯ

Отделы крупного торгового дома ежедневно продают различные виды товаров и ведут учет сведений о проданных товарах.

Необходимо спроектировать базу данных **ТОРГОВЛЯ**, информация которой будет использоваться для анализа выполнения плана реализации продукции в отделах; определения товаров, пользующихся наибольшим спросом и др.

В БД должна храниться информация:

— об ОТДЕЛАХ: *id отдела, наименование отдела, Ф.И.О. заведующего отделом, телефон, объем реализации в день* (руб.);
— ТОВАРАХ: *артикул товара, наименование товара, единица измерения, розничная цена товара* (руб.);
— ПРОДАЖАХ: *артикул товара, дата продажи, количество проданного товара.*

При проектировании БД необходимо учитывать следующее:

— отдел ежедневно осуществляет *несколько* продаж. Каждая продажа имеет отношение только к *одному* отделу;
— товар участвует в *нескольких* продажах. Каждая продажа соотносится только с *одним* товаром.

Кроме того, следует учесть:

— каждый отдел *обязательно* осуществляет продажу. Каждая продажа *обязательно* осуществляется отделом;
— товар *не обязательно* может участвовать в продаже (может быть невостребован). В продаже *обязательно* участвует товар.

Перечень запросов для реализации (общий вид):

1. Вывести список отделов, объем реализации которых менее 5000 рублей в день
2. Найти отдел с самым низким объемом реализации товара за день
3. Найти отделы, в которых было совершен менее 2 продажи в 1 первом квартале этого года
4. Вывести товары, проданные в текущем месяце
5. Посчитать количество товаров, проданных за январь месяц
6. Определить невостребованные товары
7. Подсчитать в течение какого количества дней не продавался товар определенного наименования
8. Вывести таблицу вида «Наименование товара», «Дата продажи». В списке должны быть и невостребованные товары
9. Создайте представление на хранение данных «Выручка от продажи товаров»: «Наименование отдела», «Наименование товара», «Единица измерения», «Цена розничная», «Количество товара», «Выручка»
10. Увеличьте розничную цену самого дешевого товара на 12%

Вариант 8. Проект КОСМЕТИЧЕСКАЯ ПРОДУКЦИЯ

ЗАО "Витекс" выпускает различную косметическую продукцию: кремы, шампуни, бальзамы и др. Некоторые наборы косметических средств составляют одну линию-бренд: "Кислородная линия", "Красота от природы", "Афродита" и др.

Предприятия торговли и сервиса осуществляют заказы у ЗАО "Витекс" на поставку им определенных видов продукции с указанием необходимого количества и даты поставки по мере продаж и расходования предыдущих партий.

Необходимо спроектировать базу данных **КОСМЕТИЧЕСКАЯ ПРОДУКЦИЯ**, информация которой будет использоваться для хранения информации о номенклатуре выпускаемых косметических средств, анализа спроса на отдельные виды и линии косметических средств, учета поступления и исполнения заказов предприятий в заданные сроки и др.

В БД должна храниться информация:

— о ТОВАРАХ: *id товара, наименование товара, id бренда, единица измерения, цена* (руб.);
— БРЕНДАХ: *id бренда, наименование бренда;*
— ПРЕДПРИЯТИЯХ: *id предприятия, наименование предприятия, адрес, телефон;*
— ЗАКАЗАХ: *id заказа, id предприятия, id товара, количество товара в заказе, дата заказа, дата к исполнению;*

При проектировании БД необходимо учитывать следующее:

— бренд включает *несколько* наименований товаров. Товар может принадлежать только *одному* бренду;
— товар может содержаться в *нескольких* заказах. Заказ связан с *одним* видом товара;
— предприятие может осуществлять *несколько* заказов. Заказ связан с *одним* предприятием.

Кроме того, следует учесть:

— каждый бренд *обязательно* включает несколько наименований товаров. Товар *не обязательно* принадлежит какому-либо бренду;
— товар *не обязательно* должен быть заказан. Каждый заказ *обязательно* связан с определенным товаром;
— предприятие *не обязательно* осуществляет заказы. Каждый заказ *обязательно* осуществляется *некоторым* предприятием.

Перечень запросов для реализации (общий вид):

1. Найти предприятия, расположенные в городах Краснодар, Сочи и Ессентуки.
2. Найти все товары бренда «Афродита»
3. Подсчитать количество товаров каждого бренда.
4. Вывести перечень товаров с указанием их бренда, заказанных в текущем году
5. Найти самый дорогой товар бренда «Кислородная линия»
6. Определить бренд, к которому относится товар с наименованием «Крем-масло OL»
7. Определить перечень товаров, который не был заказан
8. Найти предприятия, которые не осуществляли заказы в июле месяце
9. Создайте представление на хранение данных «Сведения об исполненных заказах на поставку косметической продукции предприятиям торговли и сервиса»: «Предприятия», «Наименование товара», «Наименование бренда», «Дата исполнения заказа», «Количество товара в доставке», «Цена», «Стоимость»
10. Удалите предприятия в г. Москва

Вариант 9. Проект ПОДПИСНЫЕ ИЗДАНИЯ

Отделение Почты России каждое полугодие осуществляет подписку граждан (в дальнейшем получателей) на различные издания (газеты, журналы) на один, три или шесть месяцев.

Необходимо спроектировать базу данных **ПОДПИСНЫЕ ИЗДАНИЯ**, информация которой будет использоваться для учета получателей и выписанных ими изданий.

В БД должна храниться информация:

- об ИЗДАНИЯХ, на которые можно оформить подписку: *id издания, вид издания* (газета, журнал)*, название издания, стоимость подписки на издание на 1 месяц* (руб.);
- ПОЛУЧАТЕЛЯХ: *id получателя, Ф.И.О. получателя, адрес получателя* (улица, дом, квартира);
- ПОДПИСКАХ, осуществленных получателями: *id получателя, id издания,срок подписки* (в месяцах), *месяц начала доставки издания, год начала доставки издания*.

При проектировании БД необходимо учитывать следующее:

- получатель может осуществить подписку *несколько* раз (подписаться на несколько изданий). Каждая подписка осуществляется *одним* получателем;
- издание может быть использовано для *нескольких* подписок (на издание могут подписаться несколько получателей). Каждая подписка соответствует*одному* изданию.

Кроме того, следует учесть:

- каждый получатель *обязательно* осуществляет хотя бы одну подписку. Каждая подписка *обязательно* соответствует получателю;
- на издание не обязательно может быть подписка (оно может быть не востребовано). Подписка обязательно соответствует некоторому изданию.

Перечень запросов для реализации (общий вид):

1. Вывести перечень изданий, подписка на которые составляет менее 3000 рублей
2. Определить издание, на которое самая дорогая подписка
3. Найти перечень изданий, на которые подписан Иванов А.А.
4. Узнать, на какое количество изданий подписан каждый получатель
5. Определить получателей, подписанных на издания с 2010 года
6. Найти получателей, подписанных на более чем 3 издания
7. Найти получателей, подписка на издания в месяц которых от 2000 до 4000 рублей
8. Узнать, какие издания необходимо доставлять в дома на улице Доватора
9. Создайте представление на хранение данных «Отчет о проведении подписки»: «ФИО получателя», «Адрес получателя», «Издание», «Срок подписки», «Стоимость подписки на 1 месяц», «С какого месяца доставка», «Итого к оплате» *("Итого к оплате, руб." = "Срок подписки, месяцев" х "Стоимость подписки на 1 месяц, руб.")*
10. Отмените возможность подписки на журнал «Сад»

Вариант 10. Проект ПОЛИКЛИНИКА

Хозрасчетная поликлиника оказывает различные медицинские услуги. Прием пациентов осуществляется врачами строго по талонам. Для врача каждой специальности определен набор талонов, используемый ежедневно. На каждого пациента заводится медицинская карта.

Оплата услуги осуществляется после приема и постановки диагноза. Стоимость визита к врачу зависит от категории врача (1-я, 2-я, 3-я) и цели посещения: консультация, обследование, лечение и др. Некоторым пациентампредоставляется скидка на обслуживание.

Необходимо спроектировать базу данных **ПОЛИКЛИНИКА**, информация которой позволит хранить сведения о заболеваниях пациентов, частоте их обращения, загрузке врачей, выручке от оказания медицинских услуг и др.

В БД должна храниться информация:

— о ВРАЧАХ: *Ф.И.О. врача, специальность, категория;*
— ПАЦИЕНТАХ: *номер медкарты, Ф.И.О. пациента, дата рождения, адрес, пол, скидка на обслуживание* (%);
— ежедневном ПРИЕМЕ пациентов: *номер талона на прием к врачу, дата визита, цель посещения, стоимость визита* (руб.);
— ДИАГНОЗАХ: *id диагноза, наименование диагноза.*

При проектировании БД необходимо учитывать следующее:

— врач осуществляет по талонам ежедневно *несколько* приемов. Каждый прием осуществляется *одним* врачом;
— пациент может приходить на прием к одному врачу *несколько* раз. На прием по талону приходит только *один* пациент;
— один и тот же диагноз выставляется на приеме *нескольким* пациентам. На одном приеме выставляется *один* диагноз.

Кроме того, следует учесть:

— каждый врач *обязательно* принимает пациентов, которые взяли талон. Каждый прием *обязательно* осуществляется врачом;
— каждый пациент *обязательно* приходит на прием по талону. На каждый прием *обязательно* приходит пациент;
— возможный диагноз *не обязательно* выставляется на приеме (его может не быть у принятых врачом пациентов). На приеме *обязательно* выставляется диагноз.

Перечень запросов для реализации (общий вид):

1. Найти пациентов, рожденных 1955-1957 гг.
2. Определить самого молодого пациента поликлиники
3. Узнать, к каким врачам на прием приходил пациент Сергеев В.В.
4. Узнать, сколько приемов осуществил врач Петров В.А.
5. Определить, какое количество приемов провел врач Сергеев В.В. в текущем году
6. Определить, каким пациентам не был поставлен диагноз
7. Подсчитать, какую сумма за посещение поликлиники потратили супруги Михайлов А.А. и Михайлова А.С.
8. Создайте представление на хранение данных «Ведомость учета приема пациентов поликлиникой»: «ФИО пациента», «Наименование диагноза», «Врач», «Стоимость визита», «Скидка», «Оплаченная сумма за визит»
9. Увеличьте процент скидки пациентам до 7%, если потратили более 7000 р, до 10% - если потратили более 10000 р.
10. Удалите врача, который не осуществлял приемы последний месяц

Вариант 11. Проект СПЕЦОДЕЖДА

Работники цехов предприятия получают со скидкой спецодежду (халаты, тапочки, комбинезоны и др.) для выполнения производственных функций. Процент скидки зависит от занимаемой должности и может составлять от 30 до 50 % стоимости единицы вида спецодежды. Спецодежда имеет разный срок носки, по истечении которого она подлежит замене. Ежедневно кладовщик цеха ведет учет выдачи спецодежды.

Необходимо спроектировать базу данных **СПЕЦОДЕЖДА**, информация которой будет использоваться для получения оперативных сведений о наличии спецодежды у работников; формирования списка работников, нуждающихся в замене спецодежды; планирования закупок спецодежды и др.

В БД должна храниться информация:

- о СПЕЦОДЕЖДЕ: *id спецодежды, вид спецодежды, срок носки, стоимость единицы* (руб.);
- ЦЕХАХ, работники которых пользуются различными видами спецодежды: *id цеха, наименование цеха, Ф.И.О. начальника цеха*;
- РАБОТНИКАХ: *id работника, Ф.И.О. работника, должность, скидка на спецодежду* (%);
- ПОЛУЧЕНИИ: *id работника, id спецодежды, дата получения, роспись*.

При проектировании БД необходимо учитывать следующее:

- в цеху работают *несколько* работников. Работник работает только в *одном* цеху;
- работник цеха участвует в получении *нескольких* видов спецодежды. Каждое получение имеет отношение только к *одному* работнику;
- один и тот же вид спецодежды поступает *несколько* раз для получения. Каждое получение относится к *одному* виду спецодежды.

Кроме того, следует учесть:

- каждый работник *обязательно* работает в цеху. В каждом цехе *обязательно* работают работники;
- работники некоторых должностей *не обязательно* участвуют в полученииспецодежды. В каждом получении *обязательно* участвует работник;
- каждый вид спецодежды *обязательно* поступает для получения. Каждое получение *обязательно* относится к некоторому виду спецодежду**.**

Перечень запросов для реализации (общий вид):

1. Найти работников, скидка на спецодежду которых составляет 30-45%
2. Вывести перечень видов спецодежды, получаемых работником Ивановым И.И.
3. Определить, какое количество видов спецодежды получает каждый сотрудник
4. Вывести перечень сотрудников, которые получили спецодежду в январе этого года
5. Подсчитать, сколько работников получили спецодежду в этом квартале
6. Определить, работники каких должностей не получают спецодежду
7. Определить, у кого из работников самая высокая скидка
8. Узнать начальника цеха, в котором работает сотрудник Иванов И.И.
9. Создайте представление на хранение данных «Отчет о получении спецодежды»: «ФИО работника», «Цех», «Вид спецождежды», «Стоимость единицы», «Скидка», «Стоимость с учетом скидки».
10. Увеличьте процент скидки сотрудникам в должности инженер-технолог до 35%

Вариант 12. Проект ГАИ

ГАИ города производит регистрацию автомобилей. Инспектора дорожно-патрульной службы следят за безопасностью дорожного движения. В случае нарушения правил дорожного движения к водителям применяются меры взысканий. Виды нарушений и меры взысканий определяются кодексом об административных правонарушениях.

Необходимо спроектировать базу данных **ГАИ**, информация которой будет использоваться для подведения статистики совершаемых водителями нарушений правил дорожного движения; выявления водителей, многократно совершающих нарушения правил дорожного движения; определения наиболееаварийных районов города, размера штрафа за совершенное нарушение и др.

В БД должна храниться информация:

— о ВОДИТЕЛЯХ: *номер водительского удостоверения*, *Ф.И.О.*, *адрес*, *телефон*;
— АВТОМОБИЛЯХ: *номер автомобиля*, *марка*, *модель*, *цвет*, *год выпуска*, *дата регистрации в ГАИ*;
— НАРУШЕНИЯХ правил дорожного движения: *id нарушения*, *вид нарушения* (превышение скорости, управление автомобилем в состоянии алкогольного опьянения и др.), *штраф* за нарушение (диапазон долей базовой величины. Например, штраф за превышение скорости составляет 0,5-10 базовых величин), *предупреждение* сделать или не сделать (Да/Нет, "Да" означает, что инспектор должен сделать водителю предупреждение за совершенное нарушение), *срок лишения права управления автомобилем* (диапазон месяцев. Например, срок за управление автомобилем в состоянии алкогольного опьянения составляет 12 – 36 месяцев);
— ВЗЫСКАНИЯХ с водителей-нарушителей: *id нарушения*, *дата и время нарушения*, *номер водительского удостоверения*, *район совершения нарушения*, *размер штрафа* (доля базовой величины, определяемая инспектором по кодексу об административных правонарушениях), *оплачен штраф или не оплачен* (Да/Нет), *срок лишения права управления автомобилем* (количество месяцев, определяемое инспектором по кодексу об административных право- нарушениях), *базовая величина* (на дату совершения нарушения, тыс. руб.), *личный номер инспектора ДПС*, установившего нарушение.

При проектировании БД необходимо учитывать следующее:

— водитель может иметь *несколько* автомобилей. Автомобиль принадлежит *одному* водителю;
— водитель может получить *несколько* взысканий (он может совершить не- сколько нарушений). Взыскание применяется к *одному* водителю;
— одному и тому же нарушению могут соответствовать *несколько* взысканий (взыскания к водителям могут применяться за один и тот же вид нарушения).Взысканию соответствует *единственное* нарушение.

Кроме того, следует учесть:

— каждый водитель *обязательно* имеет автомобиль (ГАИ хранит сведения только о тех водителях, которые зарегистрировали автомобиль). Каждый автомобиль *обязательно* принадлежит водителю (ГАИ хранит сведения только о зарегистрированных автомобилях);
— водитель не обязательно получает взыскания (водитель может не совершить ни одного нарушения). Каждое взыскание *обязательно* применяется к водителю;
— нарушению *не обязательно* соответствует взыскание (нарушение может ни разу никем не совершаться). Каждому взысканию *обязательно* соответствует нарушение.

Перечень запросов для реализации (общий вид):

1. Найти нарушения, за которые предусмотрено лишение прав не менее чем на 1 год
2. Найти автомобиль, который принадлежит Иванову И.И.

3. Подсчитайте количество автомобилей каждого водителя
4. Найдите водителей, которым принадлежит не менее 2 автомобилей
5. Узнайте, за какое нарушение полагается самый высокий штраф
6. Определите, кем и какие были совершены нарушения в летний период этого года
7. Выведите полный перечень правонарушений водителя Иванова И.И. с указанием того, был ли оплачен штраф или нет
8. Найдите тех водителей, которые не совершили ни одного правонарушения
9. Создайте представление на хранение данных «Ведомость ГАИ»: «Вид нарушения», «Время нарушения», «Номер водительского удостоверения», «Размер штрафа», «Базовая величина», «Сумма штрафа» («Сумма штрафа» = «Размер штрафа» х «Базовая величина»)
10. Удалите автомобиль с госномером А111ТТ 161

Вариант 13. Проект ЖЭС

ЖЭС города производит начисления за коммунальные услуги. Тарифы, установленные на них, не меняются. Квартиросъемщики должны оплачивать коммунальные услуги до 15 числа каждого месяца. За несвоевременную оплату взимается пеня за каждый день просрочки в размере 0,1 % общей суммы, подлежащей оплате за месяц.

Необходимо спроектировать базу данных **ЖЭС**, информация которой будет использоваться для выявления неплательщиков за коммунальные услуги, определения ежемесячной суммы оплаты квартиросъемщиками за коммунальные услуги, пени за несвоевременную оплату и др.

В БД должна храниться информация:

— о КВАРТИРОСЪЕМЩИКАХ: *лицевой счет*, *Ф.И.О.*, *телефон*;
— КВАРТИРАХ: *адрес* (улица, дом, квартира), *количество проживающих*, *площадь* ($м^2$);
— УСЛУГАХ: *id услуги*, *вид услуги* (отопление, горячее водоснабжение, каналы ТВ и др.), *единица измерения* (гКал, $м^3$, шт. и др.), *тариф* (руб.);
— ОПЛАТЕ ЗА УСЛУГУ: *лицевой счет*, *id услуги*, *фактически расходовано*, *оплатить по дату* (15.ММ.ГГ), *оплачена своевременно или не своевременно* (Да/Нет), *дата оплаты* (указывается в случае, если оплата произведена не своевременно).

При проектировании БД необходимо учитывать следующее:

— квартиросъемщик снимает *одну* квартиру. Квартира имеет *одного* квартиросъемщика;
— квартиросъемщик производит оплату за *несколько* услуг. Оплата за услугу производится *одним* квартиросъемщиком;
— одна и та же услуга может быть связана с *несколькими* оплатами (она оплачивается квартиросъемщиками в разные месяцы). Оплата относится к *одной* услуге.

Кроме того следует учесть:

— каждый квартиросъемщик *обязательно* снимает квартиру. Каждая квартира *обязательно* имеет квартиросъемщика;
— каждый квартиросъемщик *обязательно* производит оплату за услугу. Оплата за каждую услугу *обязательно* производится квартиросъемщиком;
— услуга *не обязательно* подлежит оплате (услуга может быть ни разу никому не оказана). Оплата за каждую услугу *обязательна*.

Перечень запросов для реализации (общий вид):

1. Вывести информацию о квартирах, площадь которых от 48 до 65 $м^2$
2. Определить квартиру с максимальным количеством проживающих по адресу ул. Малиновского 117ф
3. Определить, сколько электроэнергии нагорело по адресу ул. Ларина 2, кв.11
4. Узнать Ф.И.О. квартиросъемщика, проживающего по адресу ул. Ларина 2, кв.11
5. Определить, кто из квартиросъемщиков по адресу ул. Ларина 2, кв.11 задержал оплату ТВ-каналов с указанием даты внесения средств
6. Определить квартиросъемщиков, не своевременно оплативших услуги в прошлом месяце
7. Определить, какие услуги не заключены у съемщиков с лицевым счетом=44700112
8. Определить, кто из квартисъемщиков уже оплатил услуги в текущем месяце
9. Определить, сколько квартисъемщики по адресу ул. Серова 23, кв.119 заплатили за холодное водоснабжение за первый квартал текущего года *"Пеня, руб." = ("Дата оплаты" – "Оплатить по дату") х "Оплата за месяц (без учета пени), руб." х 0,001.)*
10. Измените ФИО квартиросъемщика по адресу ул. Ларина 2, кв.11

Вариант 14. Проект ОБЩЕПИТ

Предприятие общественного питания "Бистро" ежедневно занимается приготовлением различных блюд по заказам клиентов. Технология приготовления каждого блюда указана в рецепте.

Закуска "Лобио по-грузински"

Состав:

фасоль стручковая 200 г, лук зеленый 40 г, масло сливочное 30 г,зелень 10 г.

Технология приготовления:

ломаную очищенную фасоль, нашинкованный лук посолить, посы- пать перцем и припустить в масле с небольшим количеством воды; добавить зелень и довести до готовности. Затем запечь в духовке.

Выход – 210 г. Калорий – 725.

Ежедневно собираются сведения о приготовленных блюдах.

Необходимо спроектировать базу данных **ОБЩЕПИТ**, информация которой будет использоваться для приготовления блюд и анализа их калорийности, составления меню, определения расходов предприятия и др.

В БД должна храниться информация:

— о БЛЮДАХ, для описания которых нужны данные, входящие в их кулинарные рецепты: *номер блюда*, *название блюда*, *вид блюда* (закуска, суп, горячее и т.п.), *выход* (вес порции, г), *изображение блюда*;
— ежедневном ПРИГОТОВЛЕНИИ блюд: *номер блюда*, *количество порций*, *дата приготовления*;
— РЕЦЕПТАХ: *номер блюда*, *время приготовления блюда* (мин), *технология приготовления*;
— ПРОДУКТАХ, из которых приготавливаются блюда: *id продукта*, *название продукта*, *калорийность* (ккал в 100 г продукта), *вес продукта* (г), *цена* (руб.за 1 кг).

При проектировании БД необходимо учитывать следующее:

— блюдо может состоять из *нескольких* продуктов. Продукт может входить в состав *нескольких* блюд;
— у блюда может быть *несколько* приготовлений (оно может приготавливаться в разные дни в некотором количестве порций). Приготовление соотносится с*одним* блюдом;
— блюдо имеет *один* рецепт. Рецепт соответствует *одному* блюду.

Кроме того следует учесть:

— каждое блюдо *обязательно* состоит из одного или нескольких продуктов. Каждый продукт *обязательно* входит в состав одного или нескольких блюд;
— блюдо *не обязательно* приготавливается (оно может быть не востребовано клиентами). Каждое приготовление *обязательно* соотносится с некоторым блюдом;
— каждое блюдо *обязательно* имеет рецепт. Каждый рецепт *обязательно* соответствует некоторому блюду.

Перечень запросов для реализации (общий вид):

1. Вывести все супы из меню "Бистро"
2. Определить перечень продуктов, необходимых для приготовления шаурмы
3. Подсчитать количество продуктов для каждого блюда из меню
4. Определить, какие блюда были приготовлены в текущем месяце
5. Найти самый калорийный продукт в меню
6. Определить, для приготовления каких блюд используется курица терияки
7. Вывести стоимость всех блюд предприятия общественного питания

8. Определить перечень невостребованных блюд
9. Создайте представление на хранение данных «Отчет о продажах блюд»: «Дата», «Название блюда», «Количество порций», «Стоимость одного блюда», «Стоимость проданных блюд»
10. Увеличьте стоимость красной рыбы из меню на 30 рублей

Вариант 15. Проект ИНТЕРНЕТ-ПРОДАЖИ

Интернет-магазины реализуют потребителям бытовую технику (утюги, электрочайники, кухонные комбайны и др.) разных моделей известных фирм- производителей (Philips, Bosh, Mulinex и др.). Заказы осуществляются клиентами в интернет-магазинах в любое время суток. После подтверждения заказа клиентом по телефону курьер доставляет ему товар по указанному адресу.

Необходимо спроектировать базу данных **ИНТЕРНЕТ-ПРОДАЖИ**, информация которой будет использоваться для анализа спроса потребителей на конкретные модели товаров разных производителей, динамики реализации товаров в интернет-магазинах за определенные интервалы времени, сравнения условий доставки товаров в разных магазинах и др.

В БД должна храниться информация:

- об ИНТЕРНЕТ-МАГАЗИНАХ: *id магазина, название магазина, электронный адрес, оплата доставки* (Да/Нет);
- ТОВАРАХ: *id товара, название товара, фирма, модель, технические характеристики, цена* (руб.), *гарантийный срок, изображение*;
- ЗАКАЗАХ: *id заказа, id магазина, id товара, дата заказа, время заказа, количество, Ф.И.О. клиента, контактный телефон, подтверждение заказа* (Да/Нет)*;*
- ДОСТАВКЕ: *id заказа, дата доставки, время доставки, адрес доставки, Ф.И.О. клиента, Ф.И.О. курьера.*

При проектировании БД необходимо учитывать следующее:

- товар может продаваться в *нескольких* интернет-магазинах. Интернет-магазин может предлагать к продаже *несколько* товаров;
- товар может быть связан с *несколькими* заказами. Заказ связан с *одним* товаром;
- в интернет-магазине могут осуществляться *несколько* заказов. Заказ связан с *одним* интернет-магазином;
- заказ подлежит *одной* доставке. Доставка связана с *одним* заказом.

Кроме того, следует учесть:

- каждый товар *обязательно* реализуется через интернет-магазины. Каждый интернет-магазин *обязательно* реализует товары;
- товар *не обязательно* может быть заказан. Каждый заказ *обязательно* связан с товаром;
- магазин *не обязательно* может иметь заказы. Каждый заказ *обязательно* связан с определенным интернет-магазином;
- заказ *не обязательно* может быть доставлен. Каждая доставка товара *обязательно* связана с заказом.

Перечень запросов для реализации (общий вид):

1. Вывести список мультиварок и их цены
2. Вывести список товаров, реализуемый через интернет-магазин ТехноТочка
3. Определить перечень товаров, заказанных в июне месяце этого года
4. Подсчитать, сколько раз был заказан телефон Apple iphone 12 фиолетового цвета на 256 ГБ
5. Узнать, в каких интернет-магазинах Иванов И.И. совершал покупки
6. Вывести список интернет-магазинов, в которых не были совершены покупки в мае
7. Вывести полный перечень заказов, как доставленных, так еще и нет
8. Выведите в порядке убывания цены интернет-магазины, где продаются электрические чайники, поддерживающие работу с Алисой
9. Создайте представление на хранение данных «Сведения об исполненных заказах»: «Интернет-магазин», «Дата заказа», «Название товара», «Цена», «Количество», «ФИО клиента», «Стоимость заказа»
10. Удалите товар Huawei band 6

Вариант 16. Проект БАНКОМАТЫ

Банки предоставляют возможность своим клиентам осуществлять безналичные расчеты с помощью эмитируемых ими пластиковых карт и обналичивать деньги в банкоматах.

Каждый банк обслуживает свои банкоматы и своих клиентов по вопросам эксплуатации эмитируемых им пластиковых карт.

Если карточка клиента эмитирована банком, обслуживающим банкомат, то операция выдачи наличных денег банкоматом клиенту осуществляется бесплатно. Если же клиент некоторого банка обналичивает деньги в банкомате другого банка, то банкомат снимает комиссию (1,2 % суммы выдачи).

Клиенты осуществляют операции обналичивания денег в любое время суток и в любом банкомате.

Необходимо спроектировать базу данных **БАНКОМАТЫ,** информация которой будет использоваться для анализа операций обналичивания денег клиентами в банкоматах разных банков, частоты обслуживания банкоматами клиентов с взыманием комиссионных вознаграждений, динамики операций обналичивания денег клиентами отдельных банков в разных банкоматах за определенные интервалы времени и др.

В БД должна храниться информация:

- о БАНКАХ: *id банка, название банка, юридический адрес*;
- БАНКОМАТАХ: *id банкомата, адрес банкомата, id банка* (обслуживающего банкомат);
- КЛИЕНТАХ: *id карточки клиента, Ф.И.О. клиента, адрес клиента, id банка* (обслуживающего клиента);
- ОПЕРАЦИЯХ выдачи наличных денег клиентам: *id карточки клиента, id банкомата, дата, время, комиссия* (Да/Нет), *сумма выдачи* (руб.).

При проектировании БД необходимо учитывать следующее:

- банк обслуживает *несколько* банкоматов. Банкомат обслуживается *одним* банком;
- банк обслуживает *несколько* клиентов. Клиент обслуживается *одним* банком;
- банкомат обслуживает *несколько* клиентов. Клиент обслуживается *несколькими* банкоматами;
- банкомат осуществляет *несколько* операций обналичивания денег. Операция обналичивания денег связана с *одним* банкоматом;
- клиент осуществляет *несколько* операций обналичивания денег. Операция обналичивания денег связана с *одним* банкоматом.

Кроме того, следует учесть:

- каждый банк *обязательно* имеет в обслуживании банкоматы. Каждый банкомат *обязательно* обслуживается банком;
- каждый банк *обязательно* имеет клиентов. Каждый клиент *обязательно* обслуживается банком;
- каждый банкомат *обязательно* обслуживает клиентов. Каждый клиент *обязательно* обслуживается банкоматами;
- банкомат *не обязательно* осуществляет постоянно операции выдачи наличных денег. Каждая операция выдачи наличных денег *обязательно* связана с банкоматом;
- клиент *не обязательно* осуществляет операции обналичивания денег. Каждая операция обналичивания денег *обязательно* связана с клиентом.

Перечень запросов для реализации (общий вид):

1. Найти банкоматы, расположенные на ул. Соколова
2. Узнать, какой банк обслуживает банкомат по адресу пл. Гагарина, 1
3. Определить, какое количество банкоматов обслуживает каждый банк
4. Определить перечень операций банкомата Сбербанка по адресу пл. Гагарина, 1
5. Узнать, на какую сумму и в каких банкоматах были сняты наличные Ли А.А.
6. Определить, какие клиенты не выполняли операцию обналичивания денег

7. Определить банкомат, где была снята максимальная сумма денег и когда
8. Узнать, в каких банках обслуживается клиент, снявший 02.02.2022 10200 рублей в банкомате Сбербанка по адресу пл. Гагарина, 1
9. Создайте представление на хранение данных «Сведения об операциях выдачи наличных денег клиентам банкоматами с взыманием комиссионных вознаграждений»: «Номер карточки», «Номер банкомата», «Дата», «Сумма выдачи», «Сумма комиссии*» (Сумма комиссии, руб." = "Сумма выдачи, руб." x 0,012.)*
10. Измените юридический адрес банка Центр-Инвест

Вариант 17. Проект РЕМОНТ БЫТОВОЙ ТЕХНИКИ

Сервисный центр осуществляет ремонт и гарантийное обслуживание бытовой техники фирм-производителей Philips, Brown, Bosh.

Клиенты осуществляют заказы на ремонт товаров по гарантии и без нее. Сотрудники центра специализируются на ремонте и обслуживании отдельных товаров и выполняют соответствующие заказы.

В день исполнения заказа сервисный центр сообщает об этом клиенту. Срок бесплатного хранения отремонтированного изделия в сервисном центре составляет один месяц. После его истечения клиент лишается права бесплатного гарантийного ремонта изделия и оплачивает затраты центра на хранение данного товара (5 % стоимости ремонта за каждый дополнительный день).

Необходимо спроектировать базу данных **РЕМОНТ БЫТОВОЙ ТЕХНИКИ**, информация которой будет использоваться для учета услуг по ремонту и гарантийному обслуживанию товаров, анализа сроков исполнения заказов, видов неисправностей и др.

В БД должна храниться информация:

— о ТОВАРАХ: *id товара, наименование товара, фирма, модель, технические характеристики, гарантийный срок, изображение;*
— СОТРУДНИКАХ: *id сотрудника, Ф.И.О. сотрудника, должность;*
— ЗАКАЗАХ: *id заказа, Ф.И.О. клиента, id товара, гарантия* (Да/Нет), *дата поступления заказа;*
— ИСПОЛНЕНИЯХ заказов: *id заказа, вид ремонта, стоимость ремонта, дата исполнения заказа, сообщение клиенту* (Да/Нет), *дата получения товара, сумма оплаты услуг* (руб.).

При проектировании БД необходимо учитывать следующее:

— товар может быть отремонтирован *несколькими* сотрудниками. Сотрудникможет выполнять ремонт *нескольких* товаров;
— товар может быть связан с *несколькими* заказами. Заказ связан с *одним* товаром;
— заказ соотносится с *одним* исполнением. Исполнение связано с *одним* заказом;
— сотрудник осуществляет *несколько* исполнений заказов. Исполнение заказа связано с одним сотрудником.

Кроме того, следует учесть:

— каждый товар *обязательно* может быть отремонтирован сотрудниками. Каждый сотрудник *обязательно* выполняет ремонт товаров;
— товар *не обязательно* может быть связан с заказами. Каждый заказ *обязательно* связан с товаром;
— заказ *не обязательно* подлежит исполнению. Исполнение *обязательно* связано с заказом;
— сотрудник *не обязательно* осуществляет исполнения заказов. Каждое исполнение заказа *обязательно* связано с сотрудником.

Перечень запросов для реализации (общий вид):

1. Узнайте ФИО директора фирмы по ремонту бытовой техники
2. Выведите перечень товара, поступившего за последние сутки
3. Узнайте, ремонт какого товара заказал Иванов И.И.
4. Выведите статистику по количеству ремонтов каждым сотрудником
5. Узнайте, сколько ремонтов произвел сотрудник Алексеев А.А.
6. Узнайте, на какую сумму произвела фирма ремонт в этом месяце
7. Узнайте, какие виды ремонта микроволновых печей выполнила фирма в этом году
8. Выведите перечень товаров, ремонт которого осуществляется или уже был осуществлен по гарантии
9. Создайте представление на хранение данных «Сведения об исполненных заказах по гарантийному ремонту бытовой техники»: «ФИО клиента», «Телефон клиента», «Наименование товара», «Дата поступления заказа», «Дата исполнения заказа», «Срок ремонта» *("Срок ремонта, дней" = "Дата исполнения заказа" – "Дата поступления заказа".)*
10. Удалите сотрудника Иванова А.А.

Вариант 18. Проект МЕБЕЛЬ

Фирма специализируется на продаже офисной мебели разных видов заказчикам – школам, техникумам, вузам, фирмам, предприятиям, организациям. Это компьютерные столы различных моделей (СК-1, СК-2 и др.), тумбы (Т-1, Т-2 и др.), шкафы (Ш-1, Ш-2 и др.).

Необходимо спроектировать базу данных **МЕБЕЛЬ**, информация которой будет использоваться для учета продаж мебели.

В БД должна храниться информация:

— о МОДЕЛЯХ мебели: *название мебели, модель, характеристики модели, стоимость модели*;
— ПОКУПАТЕЛЯХ: *id покупателя, название покупателя, адрес покупателя, телефон покупателя;*
— ДОГОВОРАХ на продажу мебели: *id договора, id покупателя, дата оформления договора, дата исполнения договора;*
— ПРОДАЖАХ: *id договора, id мебели, количество* (шт.).

При проектировании БД необходимо учитывать следующее:

— покупатель может заключить *несколько* договоров. Договор заключается *одним* заказчиком;
— модель может быть связана с *несколькими* продажами (по разным договорам). Продажа имеет отношение к *одной* модели;
— по договору могут быть проданы *несколько* моделей мебели. Каждая продажа имеет отношение к *одному* договору.

Кроме того, следует учесть:

— каждый покупатель *обязательно* заключает договор. Каждый договор *обязательно* имеет отношение к покупателю;
— модель мебели *не обязательно* может быть продана (может не иметь спроса). Каждая продажа *обязательно* соответствует некоторой модели мебели;
— каждому договору *обязательно* соответствует хотя бы одна продажа. Каждая продажа *обязательно* соответствует некоторому договору.

Перечень запросов для реализации (общий вид):

1. Найти компьютерные кресла, изготовленных из материала кожа, по цене не более 55000 рублей
2. Узнать перечень покупателей, заключивших договор в июне месяце
3. Найти самую дорогую мебель
4. Узнать, какая мебель и сколько раз была продана
5. Узнать, какую мебель купил клиент Петров А.С.
6. Узнать, сколько договоров заключил на покупку Петров А.С.
7. Узнайте, сколько лет Петров А.С. является клиентом фирмы
8. Найти мебель, которая не имеет спроса
9. Создайте представление на хранение данных «Отчет о выполнении договоров на продажу мебели»: «Номер договора», «Название мебели», «Модель», «Количество», «Цена модели», «Стоимость»
10. Увеличьте стоимость всей мебели на 15%

Вариант 19. Проект ТИПОГРАФИЯ

Типография изготавливает полиграфическую продукцию различного рода: визитки, календари, буклеты и др.

Необходимо спроектировать базу данных **ТИПОГРАФИЯ**, информация которой будет использоваться для учета заказов на изготовление полиграфической продукции.

В БД должна храниться информация:

— о ЦЕХАХ типографии: *номер цеха, название цеха, начальник цеха, телефон цеха;*
— ПРОДУКЦИИ: *id продукции, название продукции, номер цеха, стоимость единицы печатной продукции* (руб.);
— ДОГОВОРАХ на изготовление полиграфической продукции: *номер договора, название заказчика, адрес заказчика, дата оформления договора, дата выполнения договора;*
— ЗАКАЗАХ: *номер договора, id продукции, количество продукции* (шт.).

При проектировании БД необходимо учитывать следующее:

— в цехе могут изготавливаться *несколько* видов печатной продукции. Каждый вид печатной продукции изготавливается только *одним* цехом;
— каждый вид печатной продукции может быть заказан *несколько* раз (по разным договорам). Заказ соответствует *одному* виду продукции;
— договору могут соответствовать *несколько* заказов. Заказ имеет отношение к *одному* договору.

Кроме того, следует учесть:

— каждый цех *обязательно* изготавливает хотя бы один вид продукции. Каждый вид продукции *обязательно* изготавливается в некотором цехе;
— продукция некоторого вида *не обязательно* может быть заказана (может не иметь спроса). Каждый заказ *обязательно* соответствует продукции некоторого вида;
— каждому договору *обязательно* соответствует хотя бы один заказ. Каждый заказ *обязательно* имеет отношение к некоторому договору

Перечень запросов для реализации (общий вид):

1. Найти продукцию, стоимость единицы которой более 100 рублей
2. Определить продукцию, за изготовление которой отвечает цех под руководством Пасечника А.А.
3. Определить, какое количество видов продукции изготавливается каждым цехом
4. Найти цеха, в которых изготавливается более 3 видов продукции
5. Определить, какая продукция была заказана в январе этого года
6. Определить товар, который не имеет спроса
7. Найти самый дешевый товар за единицу продукции
8. Вывести продукцию, заказанную «ИП Селезнева А.С.»
9. Создайте представление на хранение данных «Отчет о выполнении заказов на изготовление полиграфической продукции»: «Название продукции», «Название цеха», «Номер договора», «Количество», «Стоимость единицы», «Стоимость всей продукции»
10. Изменить начальника первого цеха

Вариант 20. Проект АПТЕЧНЫЙ СКЛАД

Аптечный склад РУП "ЮгФармация" осуществляет оптовую продажу лекарственных препаратов различным аптекам юга России.

Необходимо спроектировать базу данных **АПТЕЧНЫЙ СКЛАД**, информация которой будет использоваться для учета продаж аптекам лекарственных препаратов.

В БД должна храниться информация:

— о ЛЕКАРСТВАХ: *id лекарства*, *название лекарства, производитель, цена* (руб.);
— АПТЕКАХ: *id аптеки, название аптеки, адрес аптеки, номер телефона*;
— ЗАЯВКАХ: *id заявки, дата составления заявки, id аптеки, дата выполнения заявки;*
— ЗАКУПКАХ лекарственных препаратов: *номер заявки, id лекарства, количество* (шт.).

При проектировании БД необходимо учитывать следующее:

— аптека может составить *несколько* заявок. Заявка имеет отношение к *одной* аптеке;
— лекарство может иметь отношение к *нескольким* закупкам. Закупка соответствует *одному* лекарству;
— заявке могут соответствовать *несколько* закупок. Закупка соответствует *одной* заявке.

Кроме того, следует учесть:

— каждая аптека *обязательно* составляет хотя бы одну заявку. Каждая заявка *обязательно* имеет отношение к некоторой аптеке;
— лекарство *не обязательно* может входить в закупку. Каждая закупка обязательно соответствует лекарству;
— каждой заявке *обязательно* соответствует хотя бы одна закупка. Каждая закупка *обязательно* имеет отношение к заявке.

Перечень запросов для реализации (общий вид):

1. Найти аптеки, расположенные на пр. Ворошиловский и пр. Буденовский
2. Узнать, сколько заявок составила каждая аптека
3. Определить, сколько заявок составили аптеки за первый квартал этого года
4. Найти самое дорогое лекарство
5. Определить, какие лекарства не входили ни в одну закупку
6. Узнать, какое максимальное количество времени заняло выполнения заявки от момента ее составления для аптеки «ФармаПлюс»
7. Узнать, какие лекарства были в заявке аптеки «ФармаПлюс» от 11.02.2022
8. Узнайте, сколько лет аптека «ФармаПлюс» взаимодействует с аптечным складом РУП "ЮгФармация"
9. Создайте представление на хранение данных «Ведомость отпуска лекарственных препаратов аптеками»: «Номер заявки», «Название лекарства», «Производитель», «Количество», «Цена», «Сумма оплаты»
10. Увеличьте стоимость всех лекарств на 11%

Вариант 21. Проект ДИПЛОМНЫЙ ПРОЕКТ

Студенты высших учебных заведений на последнем курсе сдают госэкзамены (количество варьируется в зависимости от вуза), пишут и защищают дипломную работу. При написании дипломной работы выбирают тему дипломной работы и руководителя.

Необходимо спроектировать базу данных **ДИПЛОМНЫЙ ПРОЕКТ**, информация которой будет использоваться для хранения и поиска данных о научных темах, которые предложены студентам-дипломникам, успеваемости студентов и др.

В БД должна храниться информация:

— о СТУДЕНТАХ: *номер зачетной книжки, Ф.И.О. студента, факультет, группа*;
— ТЕМАХ: *id преподавателя*, *тема дипломной работы*;
— ОТМЕТКАХ: *номер зачетной книжки; оценка, полученная на госэкзамене, оценка, полученная на защите дипломной работы*;
— ПРЕПОДАВАТЕЛЯХ*: id преподавателя, Ф.И.О. преподавателя, степень, звание, кафедра, телефон, e-mail*.

При проектировании БД необходимо учитывать следующее:

— преподаватель для руководства студентами-дипломниками предлагает *несколько* тем дипломных работ. Тема дипломной работы может быть предложена только *одним* преподавателем;
— студент выбирает *одну* тему дипломной работы. Тема может быть выбрана только *одним* студентом.
— студент получает *одну* отметку. Отметка соответствует *одному* студенту.

Кроме того следует учесть:

— преподаватель *не обязательно* предлагает тему дипломной работы (он может не иметь научной степени или научного звания, необходимых для руководителя дипломной работы). Каждая тема *обязательно* предлагается преподавателем для написания дипломной работы;
— каждый студент *обязательно* выбирает тему для написания дипломной работы. Тема *не обязательно* выбирается студентом;
— каждый студент *обязательно* получает отметку. Каждая отметка *обязательно* соответствует студенту.

Перечень запросов для реализации (общий вид):

1. Найти преподавателей, которые не имеют ученой степени
2. Найти темы, которые предложил преподаватель Петров А.С.
3. Узнать, сколько тем предложил каждый преподаватель
4. Найти темы, которые не выбрал ни один из студентов
5. Вывести общий перечень тем и студентов, которые эту тему выбрали. Если тема не выбрана студентом, она все равно должна выводиться в этом списке.
6. Найти студентов, которые получили оценки «хорошо» и «отлично»
7. Найти студентов, средний балл которых от 4,5 до 5
8. Найти студентов, у которых руководителем был Петров А.С.
9. Создайте представление на хранение данных «Ведомость успеваемости студентов»: «ФИО студента», «Группа», «Оценка на госэкзамене», «Оценка на защите», «Средний балл»
10. Измените оценку на защите студента Иванов А.С.

Вариант 22. Проект НАУЧНО-ПРОИЗВОДСТВЕННОЕ ПРЕДПРИЯТИЕ

Сотрудники научно-производственного предприятия "Новые аналитические системы" занимаются разработкой и поставкой программного обеспечения для разных организаций.

Программное обеспечение позволяет управлять аналитическим оборудованием следующих типов: аналого-цифровым преобразователем (АЦП) NM с AM1, АЦП NM без усилителя, АЦП NM с U2, АЦП NM с AM2, АЦП NM с AM1.

При поставке программного обеспечения заключается договор с организацией. Если заключается договор на поставку программного обеспечения, которое разработал сотрудник, то он поощряется премией.

Необходимо спроектировать БД **НАУЧНО-ПРОИЗВОДСТВЕННОЕ ПРЕДПРИЯТИЕ**, информация которой будет использоваться для хранения данных о поставках программного обеспечения; организациях, в которые выполнены поставки; анализа деятельности сотрудников научно- производственного предприятия и др.

В БД должна храниться информация:

- о СОТРУДНИКАХ: *id сотрудника, номер отдела, Ф.И.О. сотрудника, должность, оклад, премия, месяц*;
- ПОСТАВКАХ: *номер договора, тип оборудования, комментарий пользователя о работе программного обеспечения* (может отсутствовать), *id сотрудника*;
- ДОГОВОРАХ: *номер договора; наименование организации*, с которой заключен договор; *дата заключения договора*;
- ОРГАНИЗАЦИЯХ: *номер договора, id страны, город, адрес, телефон, e-mail, адрес web-сайта* (может отсутствовать).

При проектировании БД необходимо учитывать следующее:

- сотрудник заключает *несколько* договоров с разными организациями на поставку программного обеспечения. Договор на поставку программного обеспечения заключает *один* сотрудник предприятия;
- поставка программного обеспечения предусматривает заключение *одного* договора. Договор соответствует *одной* поставке;
- поставка программного обеспечения производится *нескольким* организациям. Организации соответствует *одна* поставка.

Кроме того, следует учесть:

- сотрудник предприятия *не обязательно* заключают договор на поставку программного обеспечения (не устраивает договорная цена и др.). Каждый договор *обязательно* заключается сотрудником;
- каждый договор *обязательно* заключается при поставке программного обеспечения различным организациям. Каждая поставка *обязательно* предусматривает заключение договора;
- каждая поставка программного обеспечения *обязательно* производится организациям. Каждой организации *обязательно* соответствует поставка программного обеспечения.

Перечень запросов для реализации (общий вид):

1. Найти сотрудников с окладом более 45000 рублей
2. Вывести перечень организаций, с которым заключит договор Иванов И.И.
3. Узнать, какой тип оборудования поставляется организации ДГТУ
4. Вывести товар, заказанный по адресу пл. Гагарина 1
5. Узнать, на поставку какого оборудования заключены договора в сентябре этого года
6. Найти организации, с которыми был заключен договор в 2021 году
7. Узнать, сколько программного обеспечения заказала каждая организация
8. Определить, какие организации заказали более 2 видов программного обеспечения

9. Создайте представление на хранение данных «Ведомость выдачи заработной платы сотрудникам»: «ФИО сотрудника», «Оклад», «Премия», «Подоходный налог», «К выдаче»
10. Удалите сотрудника Иванова И.А.

Вариант 23. Проект КОММЕРЧЕСКАЯ ФИРМА

Коммерческие фирмы (Авто, Атлант-М и др.) занимаются поставкой различных моделей автомобилей. Модель имеет свои технические характеристики и может быть отгружена нескольким клиентам.

Необходимо спроектировать базу данных **КОММЕРЧЕСКАЯ ФИРМА**, которая должна обеспечивать хранение и поиск данных о наличии необходимых моделей автомобилей; сделках, совершаемых сотрудниками фирмы; выявлять модели автомобилей, пользующихся наибольшим спросом у клиентов, и др.

В БД должна храниться информация:

- о ПОСТАВЩИКАХ: *id фирмы*, *название фирмы, телефон, e-mail, адрес web-сайта* (может отсутствовать)*;*
- МОДЕЛЯХ: *id модели*, *наименование модели*, *цвет, обивка, мощность двигателя* (например, 100/139 кВт), *количество дверей, коробка передач* (ручная или автоматическая);
- КЛИЕНТАХ: *Ф.И.О.* клиента*, номер договора, дата покупки, телефон, адрес, id модели*;
- ПРЕЙСКУРАНТЕ ЦЕН: *id модели, год выпуска, цена* (у.е.)*, предпродажная подготовка* (стоимость услуг по подготовке к продаже, у.е.), *транспортные из держки* (у.е.).

При проектировании БД необходимо учитывать следующее:

- поставщик предлагает *несколько* моделей автомобилей. Модель автомобиляможет быть предложена *нескольким* поставщикам;
- модель автомобиля соответствует *одному* прейскуранту цен. Прейскурантцен соответствует только *одной* модели;
- клиент покупает *несколько* моделей автомобилей. Модель покупают *несколько* клиентов.

Кроме того, следует учесть:

- каждый поставщик *обязательно* предлагает модели автомобилей. Каждая модель *обязательно* предлагается поставщиком;
- модель автомобиля *не обязательно* покупается клиентом (не понравился цвет модели и т.д.). Клиент *не обязательно* покупает модель (не устраивают технические характеристики автомобиля);
- каждая модель автомобиля *обязательно* имеет прейскурант цен. Каждый прейскурант цен *обязательно* соответствует модели автомобиля.

Перечень запросов для реализации (общий вид):

1. Вывести перечень поставщиков, расположенных в г. Краснодар
2. Узнать цены на автомобиль Ford Focus с автоматической коробкой передач
3. Вывести технические характеристики автомобилей, которые можно купить от 500000 до 80000 рублей
4. Узнать, сколько моделей автомобилей предлагает каждый из поставщиков и вывести их в порядке от наибольшего к наименьшему
5. Узнать, сколько автомобилей было продано в первом квартале этого года
6. Узнать, какую модель автомобиля и по какой цене купил Иванов А.А.
7. Вывести перечень автомобилей с указанием цены, проданных в январе этого года
8. Вывести полный перечень моделей автомобилей с указанием даты заключения договора, в случае если автомобиль был продан.

9. Создайте представление на хранение данных «Отчет о реализации автомобилей»: «Фирма», «Наименование автомобиля», «Цена», «Предпродажная подготовка», «Транспортная подготовка», «Стоимость»
10. Удалите поставщиков в г. Воронеж

Задание 24. Проект ИЗДАТЕЛЬСКИЙ ЦЕНТР

Издательский центр "Печать" заключает с писателями контракты на издание книг. В течение срока контракта писатели работают только с этим центром и могут объединяться в творческие группы для совместного написания книг. Издательский центр может расторгнуть или перезаключить контракт с писателем на новый срок.

Центр издает написанные книги и продает их заказчикам: организациям, магазинам, библиотекам и др. За изданные книги писатели получают гонорары. Необходимо спроектировать базу данных ИЗДАТЕЛЬСКИЙ ЦЕНТР, информация которой будет использоваться для определения гонораров писателям за изданные книги, исчисления срока контракта с писателями, количества написанных книг писателем за время контракта, затрат на издание книг, прибыли от продажи книг и др.

В БД должна храниться информация:

- о ПИСАТЕЛЯХ: *номер паспорта, фамилия, имя, отчество, домашний адрес, телефон;*
- КОНТРАКТАХ: *номер контракта, дата заключения контракта, срок контракта (*лет*), контракт расторгнут или не расторгнут (*Да/Нет*), дата расторжения контракта;*
- КНИГАХ: *шифр книги, название, тираж, дата выхода из печати, себестоимость (*руб.*), цена продажи (руб.), гонорар (на всех авторов книги, тыс. руб.);*
- ЗАКАЗАХ: *номер заказа, название заказчика, дата поступления заказа, дата выполнения заказа, количество экземпляров заказываемой книги.*

При проектировании БД необходимо учитывать следующее:

- писателю соответствует *один* контракт (в случае перезаключения контракта прежний контракт заменяется новым). Контракт заключается с *одним* писателем;
- писатель может написать *несколько* книг. Книга может быть написана несколькими писателями;
- на книгу может быть сделано *несколько* заказов. Заказ оформляется на *одну* книгу;

Кроме того, следует учесть:

- каждому писателю *обязательно* соответствует контракт. Каждый контракт *обязательно* заключается с писателем;
- писатель *не обязательно* пишет книгу (он заключил контракт, но по какой- то причине не пишет книгу). Каждая книга *обязательно* пишется одним или несколькими писателями;
- на книгу *не обязательно* делается заказ (она может быть не востребована заказчиками). Каждый заказ *обязательно* оформляется на книгу;

Перечень запросов для реализации (общий вид):

1. Вывести перечень писателей, телефон которые проживают в одном из перечисленных городов: г. Ростов-на-Дону, г. Ставрополь, г. Краснодар
2. Найти писателей, которые заключили контракт сроком на 3-5 лет
3. Узнать, сколько времени прошло от момента заключения до момента расторжения контракта писателя А.А. Иванова
4. Узнать, сколько книг издал каждый писатель в издательском центре
5. Найти авторов книги «Кому в ДГТУ жить хорошо»
6. Найти книги, которые были заказаны менее 3 раз
7. Найти книгу с максимальным тиражом
8. Найти книги, которые были заказаны в первом полугодии этого года

9. Создайте представление на хранение данных «Прибыль от продаж книг издательского центра»: «Название книги», «Себестоимость», «Цена продажи», «Количество экземпляров», «Прибыль от продажи» *("Прибыль от продажи книги, руб." = ("Цена продажи, руб." – "Себестоимость, руб.") x "Количество экземпляров")*
10. Удалить писателей, срок заключения контракта с которыми истек

Вариант 25. Проект АВТОЗАПРАВКИ

Фирмы – поставщики автомобильного топлива – "Роснефть", "Газпромнефть", "Лукойл" имеют сеть заправочных автостанций в ЮФО. На автозаправках реализуется за безналичный расчет с помощью специальных пластиковых карт автомобильное топливо всех видов –бензин-92, бензин-95, бензин-100 дизельное топливо, газ.

Необходимо спроектировать базу данных **АВТОЗАПРАВКИ**, информация которой будет использоваться для анализа продаж автомобильного топлива за безналичный расчет клиентам по видам топлива в сети заправок конкретной фирмы-производителя, продаж различных видов топлива клиентам по всем заправочным станциям, спроса на автомобильное топливо в динамике за определенные промежутки времени и др.

В БД должна храниться информация:

- о КЛИЕНТАХ: *карт-счет клиента, Ф.И.О. клиента, адрес, телефон*;
- АВТОЗАПРАВКАХ: *id автозаправки, название фирмы, адрес автозаправки*;
- ФИРМАХ: *название фирмы, юридический адрес, телефон;*
- ТОПЛИВЕ: *id топлива, вид топлива, единица измерения, цена* (руб.);
- ежедневной ПРОДАЖЕ топлива клиентам: *дата продажи, карт-счет клиента,id автозаправки, id топлива, количество* (в натуральном выражении).

При проектировании БД необходимо учитывать следующее:

- фирма имеет *несколько* автозаправок. Автозаправка принадлежит только *одной* фирме;
- фирма предоставляет *несколько* видов топлива. Каждый вид топлива предоставляется *несколькими* фирмами;
- каждый вид топлива связан с *несколькими* продажами. Продажа связана с *одним* видом топлива;
- на автозаправке осуществляются *несколько* продаж. Продажа топлива связана с *одной* автозаправкой;
- с клиентом могут быть связаны *несколько* продаж. Продажа связана только с *одним* клиентом.

Кроме того следует учесть:

- каждая фирма *обязательно* имеет автозаправки. Каждая автозаправка *обязательно* принадлежит определенной фирме;
- каждая фирма *обязательно* предоставляет топливо. Каждый вид топлива *обязательно* поставляется фирмами;
- топливо некоторого вида *не обязательно* связано с продажами. Каждая продажа *обязательно* связана с топливом некоторого вида;
- на каждой заправке *обязательно* осуществляются продажи топлива. Каждая продажа *обязательно* связана с автозаправкой;
- с клиентом *не обязательно* связаны продажи топлива (клиент может не осуществлять покупки топлива). Каждая продажа топлива *обязательно* связанас клиентом.

Перечень запросов для реализации (общий вид):

1. Найти автозаправки, расположенные на трассе М4 ЮФО
2. Подсчитать, сколько заправок имеет каждая фирма
3. Вывести адреса автозаправок, где есть дизельное топливо, фирмы Роснефть
4. Найти фирму, предоставляющую самый дорогой бензин-95
5. Подсчитать, сколько топлива продал Лукойл в текущем месяце

6. Вывести, какое топливо, какой фирмы, в каком объеме и на каких заправках покупал клиент Иванов А.А.
7. Отсортировать в порядке убывания по количеству продаж топливо фирмы Лукойл
8. Создайте представление на хранение данных «Продажи автомобильного топлива»: «Фирма», «Дата продажи», «Карта-счет клиента», «Вид топлива», «Цена», «Количество», «Стоимость»
9. Измените адрес местоположения заправки, расположенной по адресу г. Ростов-на-Дону, ул. Серова 2
10. Удалите заправки «Газпромнефть»

Вариант 26. Проект БИБЛИОТЕКА

Библиотека располагает фондом книг, который постоянно пополняется. Книги, находящиеся в библиотеке, изданы различными издательствами. Каждый читатель может взять на абонемент не более пяти книг на срок до 20 дней.

Необходимо спроектировать базу данных **БИБЛИОТЕКА**, информация которой будет использоваться для получения оперативной информации о наличии книг в библиотеке, наличии книг у читателя, для контроля своевременности возврата книг и др.

В БД должна храниться информация:

— об ИЗДАТЕЛЬСТВАХ: *id издательства, наименование издательства, город;*
— КНИГАХ: *шифр книги, название книги, первый автор, год издания, цена книги* (руб.), *количество экземпляров* (шт.);
ЧИТАТЕЛЯХ: *id читателя, Ф.И.О. читателя, адрес, телефон;*
— ВЫДАЧАХ: *id читателя, шифр книги, дата выдачи, роспись.*

При проектировании БД необходимо учитывать следующее:

— в фонде библиотеки могут храниться *несколько* книг одного и того же издательства. Книга издается только *одним* издательством;
— книга может быть затребована *несколько* раз на выдачу. Каждая выдача относится к *одной* книге;
— читатель может быть задействован в выдаче *нескольких* книг. Каждая выдача относится к *одному* читателю.

Кроме того, следует учесть:

— каждая книга, находящаяся в фонде библиотеки, *обязательно* издается издательством. Издательство *обязательно* издает книги;
— книга *не обязательно* может быть затребована на выдачу. Каждая выдача *обязательно* связана с книгой;
— каждый читатель *обязательно* задействован в выдаче. В каждой выдаче *обязательно* задействован читатель.

Перечень запросов для реализации (общий вид):

1. Отсортировать и вывести читателей, зарегистрированных в г. Ростов-на-Дону, ул. Малиновского
2. Узнать, сколько книг взял каждый читатель
3. Найти книги, которые взял в библиотеке Степанько А.С.
4. Найти читателей, которые взяли только одну книгу в библиотеке
5. Найти книги издательства «Азбука», которые ни разу не были выданы
6. Найти читателей, которые брали книгу «1984»
7. Определить какие книги и какого издательства были выданы весной этого года
8. Вывести перечень книг, которые брали в библиотеке Ли М.А. и Петров А.К.
9. Создайте представление на хранение данных «Сведения о читателях, у которых наступил срок возврата»: «Читатель», «Автор», «Название книги», «Дата выдачи»
10. Удалите читателя Семенова А.С.

Вариант 27. Проект РОСТОВЗЕЛЕНСТРОЙ

Предприятие "Ростовзеленстрой" в весенне-летний период осуществляет выполнение заказов на цветочное оформление парков, площадей, прилегающих территорий предприятий и организаций города (заказчиков).

Для цветочного оформления используется рассада различных цветов: агератум, гортензия, петуния и др. Предприятию "Ростовзеленстрой" рассаду цветов поставляют селекционеры различных предприятий: "Цветы юга", "Цветочные композиции", "Цветники" и др.

Необходимо спроектировать базу данных **РОСТОВЗЕЛЕНСТРОЙ**, информация которой будет использоваться для учета на предприятии поставляемой рассады цветов, расчета стоимости заказов на цветочное оформление и др.

В БД должна храниться информация:

- о ПОСТАВЩИКАХ рассады цветов: *id поставщика, название поставщика, адрес поставщика;*
- ЦВЕТАХ: *id цветка*, *название цветка, id поставщика, цена рассады* (руб.);
- ЗАКАЗЧИКАХ: *id заказчика, название заказчика, адрес заказчика, телефон;*
- ДОГОВОРАХ: *номер договора, id заказчика, дата оформления договора, дата исполнения договора*;
- ЗАКАЗАХ (о цветах, используемых для выполнения договора): *номер договора,id цветка, количество рассады* (шт.).

При проектировании БД необходимо учитывать следующее:

- поставщик поставляет *несколько* видов цветов. Цветок некоторого вида поставляется только *одним* поставщиком;
- заказчик может заключить *несколько* договоров. Каждый договор заключается *одним* заказчиком;
- договору могут соответствовать *несколько* заказов. Каждый заказ имеет отношение к *одному* договору;
- каждый вид цветка может участвовать в *нескольких* заказах. Каждый заказ имеет отношение к *одному* виду цветка.

Кроме того следует учесть:

- каждый поставщик *обязательно* поставляет рассаду хотя бы одного цветка. Каждый вид цветка *обязательно* поставляется одним из поставщиков;
- каждый заказчик *обязательно* заключает хотя бы один договор. Каждый договор *обязательно* имеет заказчика;
- каждому договору *обязательно* соответствует хотя бы один заказ. Каждый заказ *обязательно* соответствует некоторому договору;
- поставляемый цветок *не обязательно* может иметь заказ (на него нет спроса). Каждый заказ *обязательно* соответствует некоторому виду цветка.

Перечень запросов для реализации (общий вид):

1. Найти цветы, цена которых 200, 300 или 700 рублей
2. Узнать, кто является поставщиком самого дорого цветка
3. Подсчитать количество цветков, которые поставляет каждый поставщик
4. Узнать, цветы какого поставщика были заказаны в июне месяце
5. Узнать, какие цветки были заказаны в этом году более 3 раз
6. Узнать, какие цветы и какого поставщика были заказаны и уже поставлены
7. Найти перечень цветков, на которые нет спроса
8. Найти заказчиков цветов по цене рассады 300-800 рублей
9. Создайте представление на хранение данных «Отчет о выполнении заказов

на цветочное оформление»: «Номер договора», «Название цветка», «Количество рассады», «Цена за единицу», «Стоимость рассады» *(Стоимость рассады, руб." = "Количество рассады, шт." x "Цена за 1 шт. рассады, руб." x 1,35.)*

10. Удалите поставщиков из г. Краснодар

Вариант 28. Проект АВИАЛИНИИ

Авиакомпания "Полет" занимается авиаперевозками пассажиров. Авиакомпанией установлены маршруты полетов: Ростов-Франкфурт, Ростов-Рига и др. Рейсы осуществляются по установленным маршрутам согласно расписанию. За каждым самолетом закреплен командир корабля.

Необходимо спроектировать базу данных **АВИАЛИНИИ**, информация которой будет использоваться для определения доходов, приносимых рейсами, даты и времени прибытия самолета в аэропорт назначения, истечения срока эксплуатации самолета и др.

В БД должна храниться информация:

- о МАРШРУТАХ: *номер маршрута*, *аэропорт вылета*, *аэропорт назначения*, *цена билета на маршрут* (руб.), *продолжительность полета* (мин.);
- РЕЙСАХ: *номер рейса*, *дата и время вылета*, *рейс отменен или не отменен* (Да/Нет);
- ПАССАЖИРАХ: *номер паспорта*, *Ф.И.О. пассажира*, *адрес*, *телефон*;
- САМОЛЕТАХ: *бортовой номер*, *модель* (Боинг-747, Ту-134, Ил-62 и др.), *дата изготовления*, *срок эксплуатации* (лет), *готовность или не готовность к вылету* (Да/Нет, "Да" означает, что самолет находится в исправном состоянии готов к вылету);
- КОМАНДИРАХ КОРАБЛЯ: *личный номер*, *Ф.И.О. командира*, *адрес*, *телефон*, *налет* (часов).

При проектировании БД необходимо учитывать следующее:

- по маршруту могут осуществляться *несколько* рейсов. Рейс осуществляется по *одному* маршруту;
- пассажир может купить билеты на *несколько* рейсов. Одним и тем же рейсом могут лететь *несколько* пассажиров;
- самолет может назначаться на *несколько* рейсов. На рейс назначается *один* самолет;
- у самолета *один* командир корабля. Командир корабля закреплен за *одним* самолетом.

Кроме того, следует учесть:

- по каждому маршруту *обязательно* осуществляется один или несколько рейсов. Каждый рейс *обязательно* осуществляется по некоторому маршруту;
- каждый пассажир *обязательно* летит рейсом. Рейс *не обязательно* имеет пассажиров (ни один пассажир не купил билет на рейс. В этом случае рейс отменяется);
- каждый самолет *обязательно* назначается на рейс. Каждому рейсу *обязательно* назначается самолет;
- у каждого самолета *обязательно* есть командир корабля. Каждый командир корабля *обязательно* закрепляется за самолетом.

Перечень запросов для реализации (общий вид):

1. Найти самолеты, которые находятся в исправном состоянии
2. Найти самолет с самой ранней датой изготовления
3. Найти сколько рейсов осуществляется по каждому маршруту
4. Узнать, по какому маршруту осуществляется только 1 рейс
5. Найти командиров корабля, который осуществляли полет в мае месяце
6. Найти отмененные рейсы, на которые не были куплены билеты
7. Определить, куда летал Петров А.С. в первом квартале этого года
8. Вывести полный перечень всех рейсов. Отмененные рейсы также должны

быть в списке.

9. Создайте представление на хранение данных «Прибыль авиакомпании»: «Номер рейса», «Цена билета», «Число пассажиров», «Прибыль от полета»
10. Удалить командиров корабля, с налетом часов более 12000

Вариант 29. Проект ПРОКАТ ВЕЛОСИПЕДОВ ФИТНЕС-КЛУБА

Фитнес-клубы расширяют свою деятельность, предоставляя в прокат велосипеды клиентам. Пункт проката велосипедов в фитнес-клубе производит выдачу велосипедов клиентов и их прием. Клиенты могут брать велосипеды для своих родственников и друзей. Прокат велосипедов имеет несколько точек проката в разных фитнес-клубах. В пункте проката находится определенный перечень велосипедов. Велосипеды выдаются на одинаковый срок – одну неделю.

Необходимо спроектировать базу данных **Прокат велосипедов фитнес-клуба**, которая должна обеспечивать хранение данных о выдаче и возврате велосипедов в пункт проката; анализе данных о должниках.

В БД должна храниться информация:

— о ВЕЛОСИПЕДАХ: *id велосипеда*, *марка*, *модель*, *цвет, цена* (руб.);
— ЭКЗЕМПЛЯРАХ: *id велосипеда; количество экземпляров; инвентарный номер; дата выдачи; дата возврата; id пункта проката*, к которому относится велосипед;
— КЛИЕНТАХ: *id клиента*, *Ф.И.О. клиента*, *дата рождения,телефон*;
— ПУНКТАХ ПРОКАТА, где содержатся данные о *id пункта, ФИО обслуживающего сотрудника* и *его адресе*.

При проектировании БД необходимо учитывать следующее:

— клиент может брать несколько велосипедов. Велосипед привязан к одному клиенту;
— велосипед определенно марки и модели имеется на пункте проката в *нескольких* экземплярах. Экземпляр соответствует *одному велосипеду*;
— велосипед может содержаться в *нескольких* пунктах проката. В пункте проката содержатся сведения о *нескольких* велосипедах.

Кроме того, следует учесть:

— клиент *не обязательно* берет велосипед (отсутствует необходимая модель и др.). Велосипед *не обязательно* востребуется клиентом;
— каждый велосипед *обязательно* имеет экземпляр. Каждый экземпляр *обязательно* соответствует велосипеду;
— каждый велосипед *обязательно* привязан к пункту проката Каждый пункт проката *обязательно* содержит сведения о велосипеде.

Перечень запросов для реализации (общий вид):

1. Вывести цену велосипедов марки Cube
2. Узнать, на каком пункте проката хранятся велосипеды марки Cube
3. Вывести статистику по выдаче велосипедов марки Cube за апрель месяц
4. Найти сколько раз брал велосипед Иванов А.А.
5. Узнать, кто из клиентов брал велосипеды более 2 раз
6. Узнать, кто из клиентов не вернул велосипед
7. Найти велосипеды, которые находятся в пункте проката в фитнес-клубе на пр. Доломановском и пр. М. Нагибина
8. Найти велосипеды, которые не были востребованы клиентами
9. Создайте представление на хранение данных «Модель велосипеда», «Марка», «Дата выдачи», «Дата возврата», «Должник (да/нет) (*Должником является тот клиент, у которого срок пользования велосипедом превысил 7 дней.)*
10. Удалить клиента Иванова А.С.

Вариант 30. ПРОЕКТ КАНЦЕЛЯРСКАЯ ПРОДУКЦИЯ

Фирма специализируется на оптовой продаже канцелярской продукции. Необходимо спроектировать базу данных **Канцелярская продукция**, информация которой будет использоваться для учета продаж.

В БД должна храниться информация:

— о ПРОДУКЦИИ: *наименование, характеристики продукции, стоимость*;
— ОРГАНИЗАЦИИ: *id организации, наименование, адрес, телефон;*
— ДОГОВОРАХ на продажу: *id договора, id организации, дата оформления договора, дата исполнения договора;*
— ПРОДАЖАХ: *id договора, id продукции, количество* (шт.).

При проектировании БД необходимо учитывать следующее:

— Организация может заключить *несколько* договоров. Договор заключается *одним* заказчиком;
— продукция может быть связана с *несколькими* продажами (по разным договорам). Продажа имеет отношение к *одной* определенной продукции;
— по договору могут быть проданы *несколько* видов продукции. Каждая продажа имеет отношение к *одному* договору.

Кроме того, следует учесть:

— каждая организация *обязательно* заключает договор. Каждый договор *обязательно* имеет отношение к организации-заказчику;
— продукция определенного типа *не обязательно* может быть продана (может не иметь спроса). Каждая продажа *обязательно* соответствует некоторому типа продукции;
— каждому договору *обязательно* соответствует хотя бы одна продажа. Каждая продажа *обязательно* соответствует некоторому договору.

Перечень запросов для реализации (общий вид):

1. Найти и вывести цену ручек с характеристиками: черная, гелевая
2. Узнать перечень продукции, заказанной по договорам в сентябре месяце
3. Узнать какую продукцию заказала фирма «ИП Малиновский А.С.»
4. Узнать, какие организации заказали продукцию в сентябре, и договор по которым еще не был исполнен на текущей момент
5. Подсчитать, сколько раз совершали заказ все организации-заказчика
6. Найти организации, которые заключили более 2 договоров
7. Узнайте, сколько лет фирма «ИП Малиновский А.С.» заказывает канцелярскую продукцию
8. Найти продукцию, которая не имеет спроса
9. Создайте представление на хранение данных «Отчет о выполнении договоров на продажу канцелярской продукции»: «Номер договора», «Наименование» «Количество», «Цена», «Стоимость»
10. Удалите заказчика - фирму «ИП Малиновский А.С.»