**Лабораторное занятие №4.**

**Создание связей в базе данных.**

**Выполнял: Шеменев**

**Цель:** Научиться создавать таблица с вторичным ключом, создавать связи между таблицами и строить схему данных .

## Исходные данные

Студент получает индивидуальный вариант исходных данных с кратким описанием предметной области, который используется при выполнении всех лабораторных работ. При этом каждая очередная лабораторная работа является продолжением выполненной ранее и поэтому они должны обязательно выполняться последовательно. Вариант задания приведен в лабораторной работе №1.

## Используемые программы

Все операции выполняются с помощью приложения «**SQLiteStudio**».

**Теоретические сведения**

**Связи в реляционной модели**

Если между некоторыми сущностями существует связь, то факты из одной сущности ссылаются, или некоторым образом связаны, с фактами из другой сущности. Поддержание непротиворечивости функциональных зависимостей между сущностями называется ссылочной целостностью. Поскольку связи содержатся «внутри» реляционной модели, реализация ссылочной целостности может выполняться как приложением, так и самой СУБД (с помощью механизмов декларативной ссылочной целостности – внешних и альтернативных ключей, ограничений, триггеров).

### Виды связей между таблицами

Связь осуществляется путем сопоставления данных в ключевых столбцах; обычно это столбцы, имеющие в обеих таблицах одинаковые названия. В большинстве случаев сопоставляются первичный ключ одной таблицы, содержащий для каждой из строк уникальный идентификатор, и внешний ключ другой таблицы. Например, с каждым из изданий, находящихся в продаже, можно связать объемы его продаж путем создания столбца "ИД\_издания" в таблице "Книги" (первичный ключ) и столбца "ИД\_издания" в таблице "Продажи" (внешний ключ).

Существует три вида связей между таблицами. Вид создаваемой связи зависит от того, как заданы связанные столбцы.

#### Связи "один ко многим"

Связь "один ко многим" - наиболее распространенный вид связи. При такой связи каждой строке таблицы А может соответствовать множество строк таблицы Б, однако каждой строке таблицы Б может соответствовать только одна строка таблицы А. Например, между таблицами "Издатели" и "Книги" установлена связь "один ко многим": каждый из издателей может опубликовать множество книг, однако каждая книга публикуется лишь одним издателем.

Связь "один ко многим" создается в том случае, когда только на один из связываемых столбцов наложено ограничение уникальности или он является первичным ключом.

В SQLiteStudio сторона связи "один ко многим", которой соответствует первичный ключ, обозначается символом ключа. Сторона связи, которой соответствует внешний ключ, обозначается символом бесконечности.

#### Связи "многие ко многим"

При установлении связи "многие ко многим" каждой строке таблицы А может соответствовать множество строк таблицы Б и наоборот. Такая связь создается при помощи третьей таблицы, называемой соединительной, первичный ключ которой состоит из внешних ключей, связанных с таблицами А и Б. Например, между таблицами "Авторы" и "Книги" установлена связь вида "многие ко многим", задаваемая с помощью связей вида "один ко многим" между каждой из этих таблиц и таблицей "АвторыКниг". Первичный ключ таблицы "АвторыКниг" - это сочетание столбцов "ИД\_автора" (первичного ключа таблицы авторов) и "ИД\_книги" (первичного ключа таблицы заголовков).

#### Связи "один к одному"

При установлении связи "один к одному" каждой строке таблицы А может соответствовать только одна строка таблицы Б и наоборот. Связь "один к одному" создается в том случае, когда оба связанные столбца являются первичными ключами или на них наложены ограничения уникальности.

Этот вид связи используется редко, поскольку в такой ситуации связываемые данные обычно можно хранить в одной таблице. Использовать связь вида "один к одному" можно в указанных ниже случаях.

* Чтобы разделить таблицу, содержащую слишком много столбцов.
* Чтобы изолировать часть таблицы по соображениям безопасности.
* Для хранения данных кратковременного использования, удалить которые проще всего путем очистки таблицы.
* Для хранения данных, имеющих отношение только к подмножеству основной таблицы.

В SQLiteStidio сторона связи "один ко многим", которой соответствует первичный ключ, обозначается символом ключа. Сторона связи, которой соответствует внешний ключ, также обозначается символом ключа.

**Пример создания связей**

Имеется база данных состоящая из 6 таблиц: DATA (главная таблица), остальные являются таблицами-справочниками (имеют префикс малой **s**). Примеры таблиц представлены на рисунках ниже.

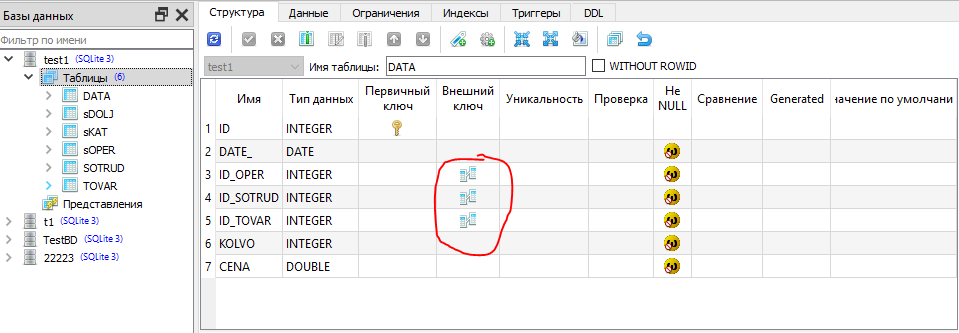
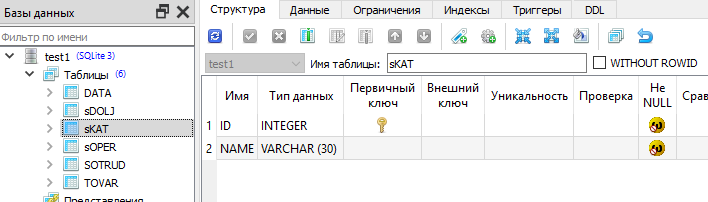


Рисунок 1 – таблица DATA

И пример одного из справочников   


Таблицы «**sDOLJ**», «**sKAT**», «**sOPER**» состоит из 2 полей:

- поле «**ID**» - идентификатор записи в таблице;

- поле «**NAME**» - наименование (должность, категория товара, вид операции).

Таблица «**DATA**» состоит из 5 полей:

- поле «**ID**» - идентификатор записи в таблице;

- поле «**DATE\_**» - дата совершения операции;

- поле «**ID**\_**OPER**» - идентификатор тип операции (поступление, продажа);

- поле «**ID\_SOTRUD**» - идентификатор сотрудника;

- поле «**ID\_TOVAR**» - идентификатор товара;

- поле «**KOLVO**» - количество выбранного товара.

- поле «**CENA**» - цена товара.

Для создания связей между таблицами в данной базе данной, необходимо открыть главную таблицу «DATA» выбрать соответствующее поле, где требуется поставить вторичный ключ (**ID**\_**OPER, ID\_SOTRUD, ID\_TOVAR**) и установить дополнительные ограничения, а именно «**Внешний ключ**» (рисунок 3).

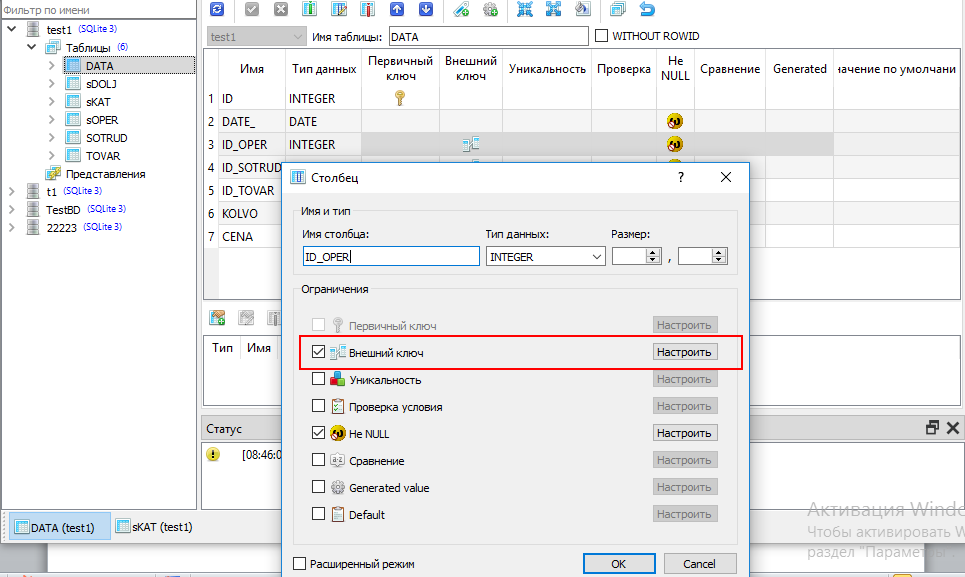


Рисунок 3 – создание связей

Что бы сознать новую связь требуется нажать на пункт «**Внешний ключ**» и настроить данный параметр. В открывшемся окне необходимо выполнить несколько действий:

1. Указать внешнюю таблицу, из которой будут браться данные.
2. Указать поле из внешней таблицы, по которому будет осуществляться связь. Связь всегда осуществляется по идентификатору (ID).
3. Далее необходимо указать условия обновления и удаления. На соответствующих пунктах ставим галочки (**ON UPDATE**, **ON DELETE**) и выставить состояние в тип **CASCADE**.
4. Применить изменения.

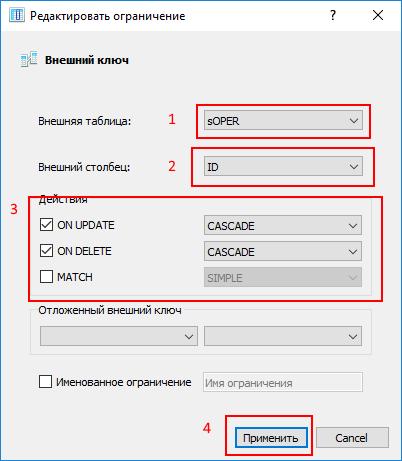


Рисунок 4 - создание новой связи

**Задание**

Создать связи со всеми таблицами, где это необходимо.

