

Урок 6.1. Взаимосвязи между таблицами ER диаграмма

Основные типы связей между таблицами	2
Задание для закрепления	4
Ключи для связи таблиц	5
Графическое отображение связей. ER-диаграммы	7
Задание для закрепления	10

Основные типы связей между таблицами



Важно!

База данных с доступом на чтение:

hostname: ich-db.edu.itcareerhub.de

username: ich1

password: password

Зачем хранить информацию в разных таблицах?

1. **Избежание дублирования данных:** Разделение данных на несколько таблиц позволяет исключить избыточное дублирование информации. Например, вместо хранения информации о клиенте в каждом заказе, можно создать отдельную таблицу "Customers", где будет храниться информация о каждом клиенте.
2. **Удобство управления:** Разделение данных упрощает обновление, удаление и вставку информации. Например, если нужно обновить контактные данные клиента, это можно сделать в одной таблице, и изменения отразятся во всех связанных данных.
3. **Поддержка целостности данных:** Связи между таблицами помогают гарантировать, что данные остаются актуальными и непротиворечивыми.

Связь один к одному (One-to-One):



Связь один к одному (One-to-One) — это связь, в которой каждой записи в одной таблице соответствует ровно одна запись в другой таблице.

Пример:

Допустим, у нас есть таблица **employees** и таблица **employee_privileges**. Мы знаем, что каждый сотрудник может иметь только одну привилегию или вовсе ее не иметь. Таким образом, одной записи о сотруднике в таблице **employees** соответствует одна запись в таблице **employee_privileges**.

Связь один ко многим (One-to-Many):



Связь один ко многим (One-to-Many) — это связь, в которой связи одной записи в таблице может соответствовать несколько записей в другой таблице. Этот вид связи самый популярный.

Пример:

Таблица **customers** и таблица **orders**. Один клиент может делать несколько заказов, но каждый заказ принадлежит только одному клиенту.

Связь многие ко многим (Many-to-Many):



Связь многие ко многим (Many-to-Many) — это связь, в которой одной записи в одной таблице могут соответствовать несколько записей в другой таблице и наоборот.

Пример:

Таблицы **products** и **orders**. Один заказ может включать несколько товаров, и один товар может быть частью нескольких заказов. Для реализации такой связи обычно используется промежуточная таблица, например **order_details**, где хранятся связи между **products** и **orders**.



Задание для закрепления

1. Внимательно изучите таблицы базы данных northwind.
2. Приведите по одному примеру разных типов связей между таблицами.

Ключи для связи таблиц



Первичный ключ — это одно или несколько полей (столбцов) в таблице, которые уникально идентифицируют каждую запись в этой таблице.

Основные характеристики первичного ключа:

- **Уникальность:** Каждое значение первичного ключа должно быть уникальным. Это означает, что в таблице не может быть двух строк с одинаковым значением первичного ключа.
- **Не допускает NULL:** Значение первичного ключа не может быть пустым (NULL). Оно всегда должно быть определено.
- **Неизменяемость:** Значения первичного ключа редко изменяются, так как это может нарушить связи с другими таблицами.

Пример:

В таблице **customers** в базе данных Northwind **id** является первичным ключом. Каждый клиент имеет уникальный идентификатор, который используется для однозначной идентификации клиента.



Внешний ключ (Foreign Key) — это поле (или набор полей) в одной таблице, которое ссылается на первичный ключ другой таблицы.

Внешний ключ используется для создания связи между двумя таблицами и обеспечения целостности данных.

Основные характеристики внешнего ключа:

- **Связывает таблицы:** Внешний ключ устанавливает связь между двумя таблицами. Он обеспечивает связь "многие к одному" или "один к одному".
- **Обеспечение целостности данных:** Внешние ключи помогают поддерживать целостность данных, гарантируя, что записи в связанной таблице существуют. Например, если заказ в таблице **orders** ссылается на клиента, то такой клиент должен существовать в таблице **customers**.

Примеры использования:

1. Связь заказов и клиентов:

Таблица `orders` содержит поле `customer_id`, которое является внешним ключом, ссылающимся на таблицу `customers`.

Это позволяет связать каждый заказ с конкретным клиентом.

2. Связь заказов и продуктов:

Таблица `order_details` содержит поля `order_id` и `product_id`, которые являются внешними ключами и ссылаются на таблицы `orders` и `products` соответственно.

Это позволяет связать конкретный заказ с определенным набором продуктов.

Графическое отображение связей. ER-диаграммы

Как вы уже знаете, видов связей бывает несколько.

Для каждого вида связи можно использовать свое условное обозначение — и таким образом отобразить взаимосвязи графически.



ER-диаграмма (ERD, Entity-Relationship Diagram) — это графическое представление структуры базы данных, которое показывает, как данные в системе связаны между собой.

ER-диаграммы используются для проектирования и визуализации базы данных до её создания.

Основные компоненты ER-диаграммы

1. Сущности (Entities):

- **Сущность** представляет собой объект или понятие, которое имеет значение для системы и о котором необходимо хранить информацию. Иными словами это таблицы.
- Примеры: **customers** (Клиенты), **orders** (Заказы), **products** (Продукты).
- В ER-диаграммах сущности изображаются в виде прямоугольников.

2. Атрибуты (Attributes):

- **Атрибуты** — это свойства или характеристики сущностей.
- Примеры: у сущности **customers** могут быть атрибуты **id**, **company_name**, **contact_name**. Это набор столбцов таблицы.

Customers

Id
company_name
contact_name

3. Связи (Relationships):

- **Связи** показывают, как сущности связаны друг с другом. Это могут быть связи "один к одному", "один ко многим" или "многие ко многим".
- Примеры: связь между **customers** и **orders** (один клиент может делать много заказов).



ER-диаграммы, как и другую документацию о структуре и связях в базе данных, разрабатывают на этапе проектирования. Если такая документация под рукой, работать с базой будет удобно.

Документацию обычно получают у администратора базы или инженера данных. А без документации придётся самостоятельно просматривать таблицы и искать связи.

Пример ER диаграммы



Задание для закрепления

1. Отрисуйте ER диаграмму с взаимосвязями между следующими таблицами.

- customers
- orders
- products
- Order_details

2. Используйте <https://www.drawio.com/>, либо листок и ручку.

3. Выбирайте SoftWare и пример самой первой диаграммы.

