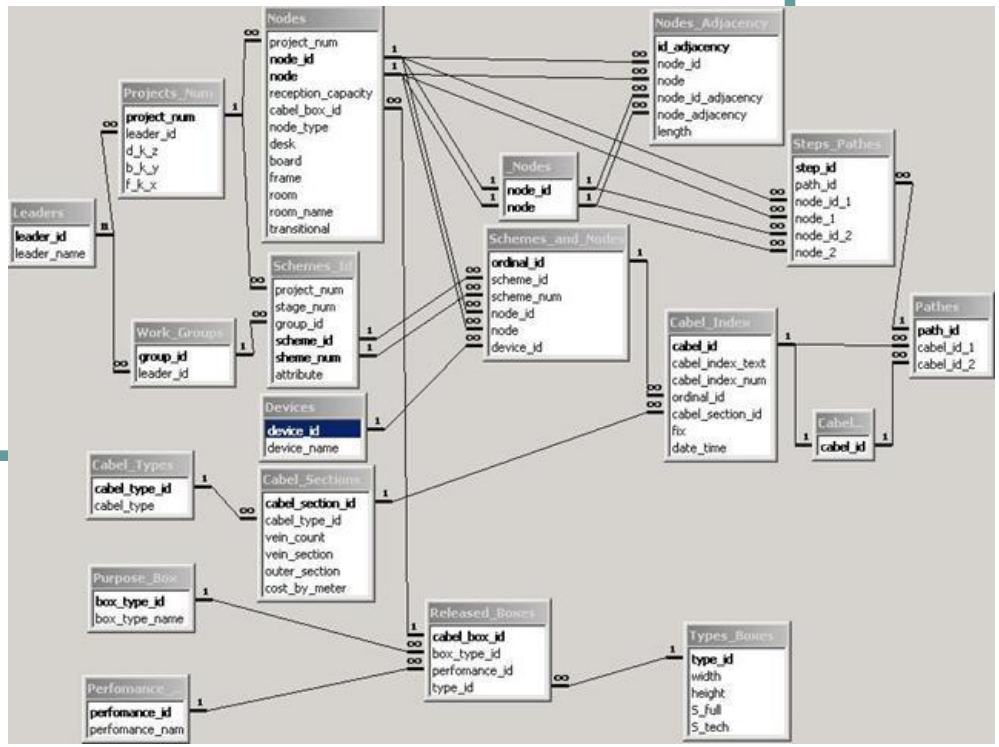
Multitable Databases



Контрольные вопросы

- Что такое выборка?
- Как реализуется сортировка выборки?
- Как реализуется построчный фильтр?
- Шаблоны LIKE
- Синтаксис INSERT
- Синтаксис UPDATE
- Синтаксис DELETE

Рост сложности реализации

Основной проблемой при разработке БД является проблема роста сложности реализации проекта с течением времени. Представить предметную область в том виде, в котором она существует на самом деле в реальном мире, с помощью одной таблицы практически невозможно.
http://www.mstu.edu.ru/study/materials/zelenkov/ch_5_1.html

Структура таблицы Products

id	name	category	price	discount	quantity	measurement	producer	country	supplier	date_of_delivery
1	Масло	Молочная	22	1	200	шт.	Коровушка	Украина	Кестер	2016-01-01
2	Молоко	Молочная	12	2	125	л	Семерка	Украина	Кестер	2016-01-01
3	Сливки	Молочная	20,5	0	250	шт.	Молочник	Украина	Веселов	2016-05-10
4	Платье	Одежда	480	15	100	шт.	Фея	Беларусь	Лукаш	2016-05-05
5	Блуза	Одежда	260	12	200	шт.	Фея	Беларусь	Лукаш	2016-06-06
6	Шорты	Одежда	210	11	100	шт.	Фея	Беларусь	Лукаш	2016-05-03
7	Носки	Одежда	50	10	300	шт.	Рест	Россия	Стандарт	2012-02-02
8	Платок	Аксессуары	130	13	200	шт.	Рест	Россия	Стандарт	2014-02-03
9	Кольцо	Аксессуары	120	5	100	шт.	Рест	Россия	Позитив	2005-04-03
10	Ремень черный	Аксессуары	200	3	450	шт.	Велес	Украина	Сикерта	2005-04-03
11	Ремень серый	Аксессуары	290	3	340	шт.	Велес	Украина	Сикерта	2006-04-03
12	Ремень плетеный	Аксессуары	340	5	390	шт.	Велес	Украина	Сикерта	2004-03-02
13	Сумка жен.	Сумки	1000	10	340	шт.	Велес	Украина	Сикерта	2005-04-03
14	Сумка дет.	Сумки	300	12	230	шт.	Велес	Украина	Сикерта	2006-04-03
15	Сумка хоз.	Сумки	120	0	400	шт.	Велес	Украина	Сикерта	2004-03-02
17	Печень куриная	Мясо	50	0	50	кг	Ряба	Украина	Найда	2016-06-03
18	Печень гусиная	Мясо	70	0	60	кг	Ряба	Украина	Найда	2006-05-04
19	Филе говяжье	Мясо	80	0	30	кг	Ряба	Украина	Найда	2007-06-05
20	Филе куриное	Мясо	60	0	56	кг	Ряба	Украина	Найда	2016-06-05
21	Мыло "Ромашка"	Хозтовары	23	2	34	шт.	Вега	Украина	Транс	2016-07-06
22	Мыло "Ваниль"	Хозтовары	22	2	45	шт.	Вега	Украина	Транс	2005-04-03
23	Мыло "Сирень"	Хозхтовары	34	3	34	шт.	Вега	Украина	Транс	2007-06-05
24	Мыло "Весеннее"	Хозтовары	22	3	56	шт.	Вега	Украина	Транс	2005-04-03
25	Мыло "Хозяйстве...	Хозтовары	12	3	4	шт.	Вега	Украина	Транс	2005-04-03
26	Шампунь	Хозтовары	44	3	67	шт.	Вега	Украина	Транс	2006-04-03

Избыточность

Обратите внимание на поля **category**, **producer**, **country**, **supplier**. В данной таблице значения в этих полях часто повторяются (например, для 26 записей только 3 разных названия стран), при этом поля текстовые, что влечёт значительные расходы дискового пространства («Украина» занимает либо 7, либо 14 Байт).

Потенциально неверный ввод

К тому же, неопытные пользователи при вводе информации для поля, допустим, **category**, могут запросто ввести название категории «Малочные», а не «Молочные», и в дальнейшем это может создать проблемы как с сортировкой получаемых из таблицы выборок, так и с поиском информации в этой таблице.

Решение этих двух проблем

Разумнее было бы создать для категорий **отдельную таблицу**, где каждое название категории хранилось бы только один раз. То же самое для поставщиков, стран и производителей. Создание многотабличных БД потребует некоторых дополнительных усилий, и работа с полученной многотабличной БД будет немного отличаться от работы с однотабличной БД.

Аномалия добавления

Также при работе с однотабличными БД часто возникают проблемы добавления информации, описывающей какие-то отдельные части предметной области. Например, в БД Products может потребоваться сохранить дополнительную информацию о магазинах, такую как адрес, телефоны, имена сотрудников и тп. Добавлять эту информацию в ту же таблицу Products, где описаны сами товары, очень неудобно.

Аномалия удаления

Также существует аномалия, связанная с удалением данных. Допустим, будет необходимо удалить всю информацию о некотором производителе, и все товары, которые этот производитель выпустил. Технически это вполне осуществимо, но логически это идет в разрез с идеологией однотабличных БД.

Подробнее про аномалии

- http://citforum.ck.ua/database/dblearn/dblearn06.shtml
- В книге Кристофера Дейта

Отдельная таблица

Итак, мы остановились на необходимости вынести названия категорий в отдельную таблицу:

Categories

id	name
1	Заморозка
2	Овощи, фрукты
3	Кондитерские изделия
4	Консервы
...	...

Внешний ключ

В таблице Products вместо текстового названия категории можно использовать числовой идентификатор этой категории из новой таблицы Categories. Если так сделать, то в таблице Products будет поле, ссылающееся на первичный ключ таблицы Categories. Такое поле называется **внешним ключом**. Таким полям принято давать «говорящие» названия. Например, добавлять перед именем поля префикс **id_**. Итак, **внешний ключ – это одно или несколько полей подчинённой таблицы, предназначенных для связи с главной таблицей. Там хранятся значения первичного ключа главной таблицы, за счёт этого и обеспечивается связь.**

Что получится

Products

id	name	id_category
1	Пельмени	1
2	Мороженое	1
3	Крабовые палочки	1
4	Авокадо	2
5	Руккола	2
6	Мармелад	3
7	Леденцы	3
8	Повидло	4
9	Шпроты	4
...	...	

Вопрос

Ну и как же пользователь будет работать с какими-то цифрами вместо понятных названий? В действительности, конечный пользователь сайта или приложения никогда не увидит внутреннюю структуру БД. Замена цифр на текстовые значения при их отображении для пользователя – это задача программиста, и её решение будет рассмотрено чуть позже.

Виды связей между таблицами

Из рассмотренного примера очевидно, что у нас не просто появилась ещё одна таблица, но и, благодаря внешнему ключу, эти две таблицы связаны друг с другом (как минимум, в нашем воображении ☺). Существует три вида связей между таблицами:

- один к одному
- один ко многим
- многие ко многим

Один к одному

Обозначение связи: **1:1**.

Таблицы связаны по схеме «один-к-одному», если каждой записи из первой таблицы соответствует 1 или 0 записей во второй таблице.

Например: таблица «Люди» и таблица «Внутренние паспорта». Действительно, по законодательству Украины, у каждого гражданина может быть только один внутренний паспорт (который можно нечаянно потерять ☺). Ещё пример: «Люди» и «Идентификационные коды», «Медведи» и «Ложки».

Эта связь используется реже, чем две остальные.

Один ко многим

Обозначение связи: **1:M**.

Между таблицами имеет место связь «один-ко-многим», если каждой записи из первой таблицы соответствует любое количество (0, 1 или более) записей во второй таблице.

Это самая часто используемая связь с современных реляционных БД. Например, так можно соединить таблицы «Матери» и «Дети», «Специализации» и «Группы».

Многие ко многим

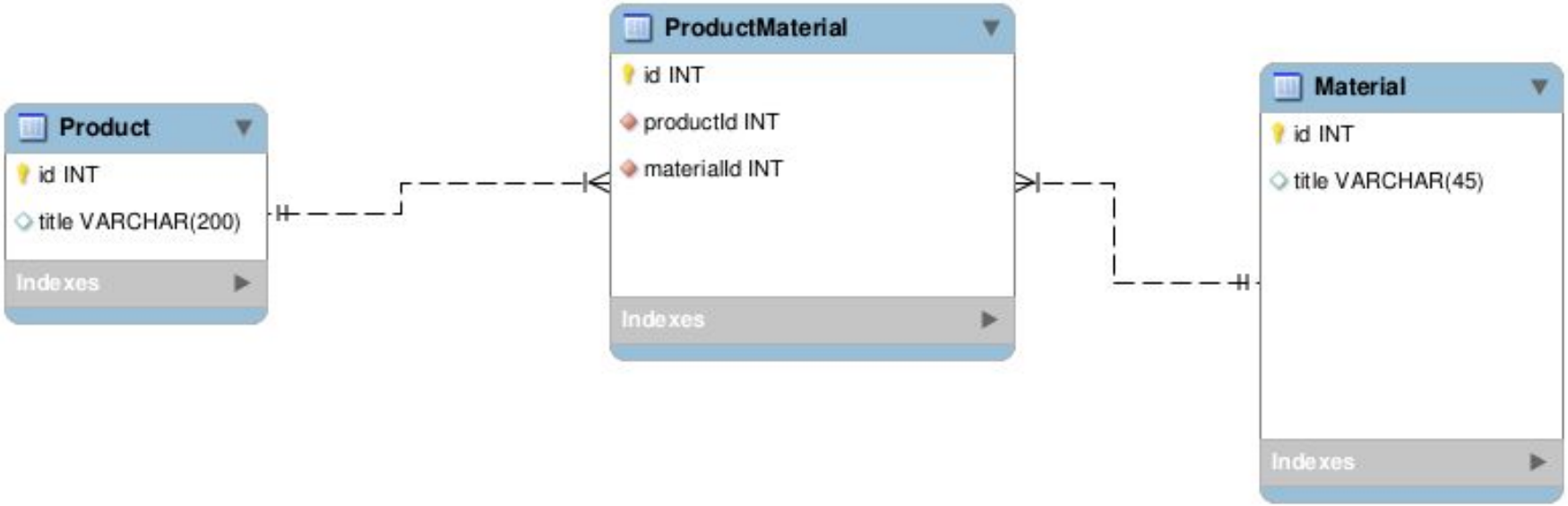
Обозначение связи: **N:M**.

Между таблицами имеет место связь «многие-ко-многим», если каждой записи из первой таблицы соответствует 0, 1 или более записей во второй таблице и наоборот.

Пример: таблица «Дисциплины» и таблица «Преподаватели». Каждая дисциплина может читаться несколькими преподавателями, но и каждый преподаватель может читать несколько разных дисциплин. Другой пример: «Люди» и «Увлечения».

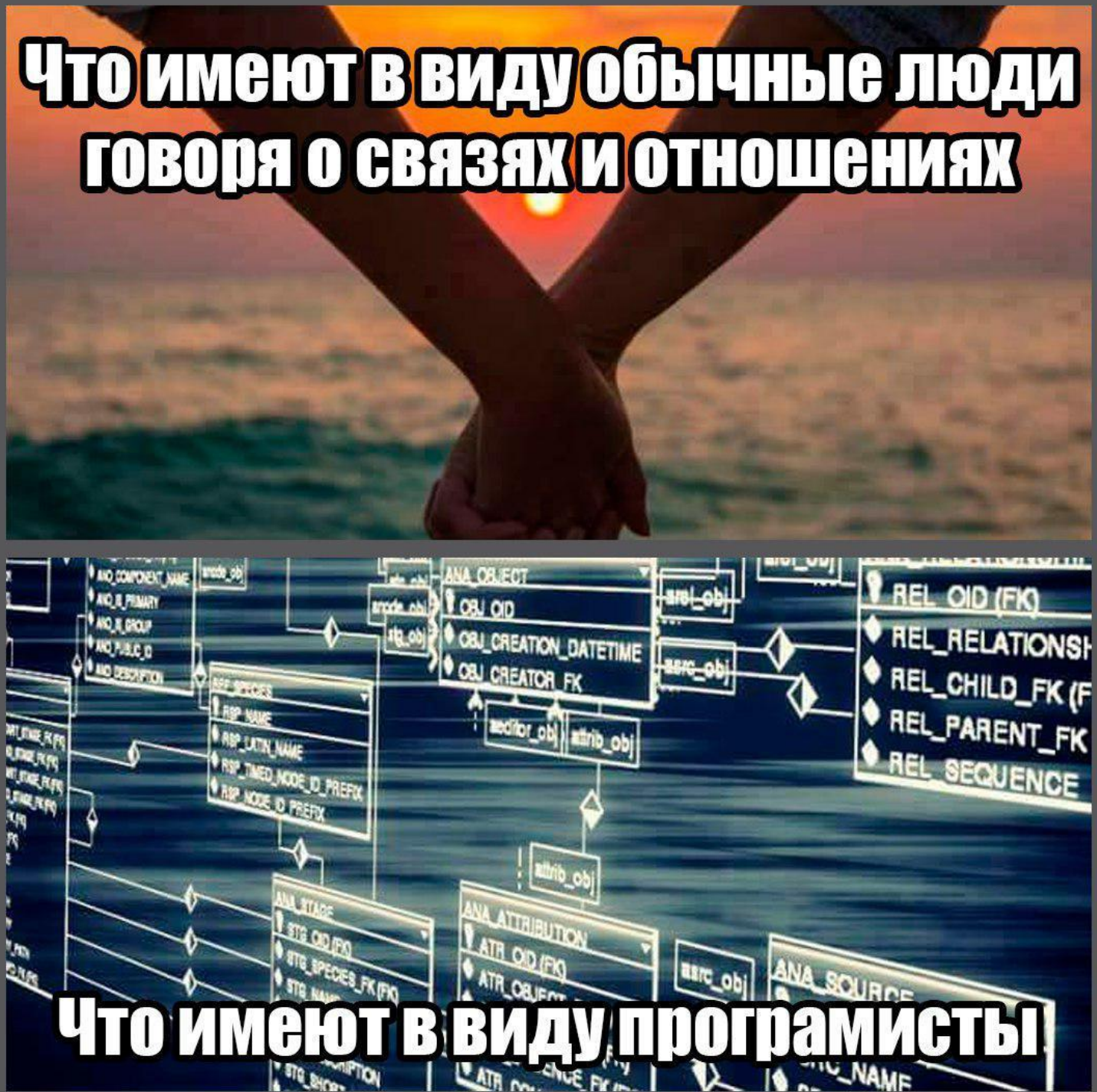
Таблица-тэг

Физически не существует способа реализовать связь «многие-ко-многим» непосредственно между двумя таблицами. Вместо этого такая связь реализуется при помощи дополнительной таблицы, называемой **тэгом**, и двух связей «один-ко-многим».



Связь «Родитель-Потомок»

Вообще, существует ещё одно обозначение связи между двумя таблицами: «**родитель-потомок**». Если первая таблица содержит внешний ключ, ссылающийся на вторую таблицу, то говорят, что первая таблица является потомком, а вторая – родителем. В примере с таблицами «Продукты» и «Категории», таблица «Категории» – родительская, а таблица «Продукты» – дочерняя.


Что имеют в виду обычные люди
говоря о связях и отношениях
ANA_OBJECT
OBJ_OID
OBJ_CREATION_DATETIME
OBJ_CREATOR_FK
REL_OID (FK)
REL_CHILD_FK (F
REL_PARENT_FK
REL_SEQUENCE
attrib_obj
ANA_ATTRIBUTION
ANA_SOURCE
Что имеют в виду програмисты

Ограничения внешнего ключа

Допустим, из таблицы «Категории» потребовалось удалить некоторую категорию. Однако в дочерней таблице «Продукты» есть продукты, относящиеся к этой категории. Если удалить категорию, то внешний ключ этих продуктов будет ссылаться в никуда… Что можно сделать в такой ситуации? Существует четыре варианта развития событий.

1. Restrict (No action)

Запрещено удалять записи из родительской таблицы, если в дочерней таблицы есть ссылки на удаляемую запись. Т.е. сначала необходимо удалить все записи из дочерней таблицы, и только затем удалить запись из родительской таблицы.

2. Cascade

Из дочерней таблицы автоматически удаляются все записи, ссылающиеся на удаляемую запись из родительской таблицы.

3. Set Null

Предполагает, что будет происходить автоматическая замена значений (на NULL) внешнего ключа дочерней таблицы, ссылающихся на удаляемую запись из родительской таблицы (это не всегда возможно, например, если для поля внешнего ключа не выставлена галочка Allow Nulls).

4. Set Default

Автоматически заменяет во внешнем ключе записей дочерней таблицы, ссылающихся на удаляемую запись из родительской таблицы, значение на установленное значение поля по умолчанию.

Изменение первичного ключа

Аналогичное поведение можно получить при изменении первичного ключа записи из родительской таблицы, на которую ссылаются записи из дочерней. В этом случае эти же четыре способа остаются актуальными с той разницей, что во втором случае записи в дочерней таблице не удаляются, а вместо этого автоматически изменяются на новые значения внешних ключей таких записей.

Целостность данных

Таким образом реализуется сохранение **целостности данных** в БД. Все современные СУБД содержат достаточно развитые средства, позволяющие следить за целостностью данных. Подобные проблемы могут возникнуть в ситуации, когда пользователь пытается ввести во внешний ключ дочерней таблицы значение, которое отсутствует среди значений первичного ключа родительской таблицы. В этом случае средства сохранения целостности данных также помогают, блокируя занесение такого ошибочного во внешний ключ.

Определение понятия

Целостность базы данных (database integrity) — соответствие имеющейся в базе данных информации её внутренней логике, структуре и всем явно заданным правилам. Каждое правило, налагающее некоторое ограничение на возможное состояние базы данных, называется ограничением целостности (integrity constraint). Примеры правил: вес детали должен быть положительным; количество знаков в телефонном номере не должно превышать 25; возраст родителей не может быть меньше возраста их биологического ребёнка и тд.

https://ru.wikipedia.org/wiki/%D0%A6%D0%B5%D0%BB%D0%BE%D1%81%D1%82%D0%BD%D0%BE%D1%81%D1%82%D1%8C_%D0%B1%D0%B0%D0%B7%D1%8B_%D0%B4%D0%B0%D0%BD%D0%BD%D1%8B%D1%85

Целостность и достоверность

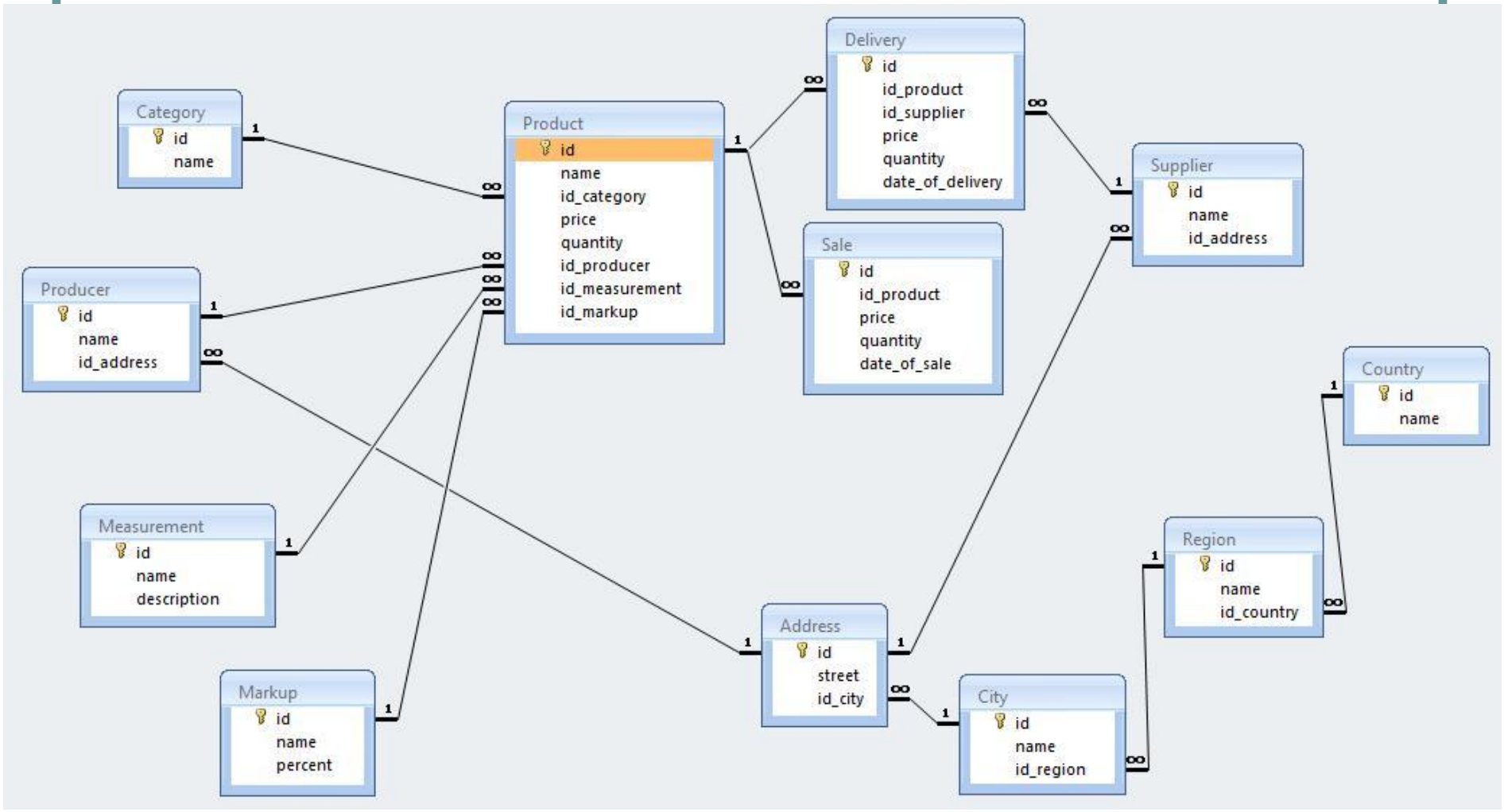
Задача аналитика и проектировщика базы данных — возможно более полно выявить все имеющиеся ограничения целостности и задать их в базе данных. Целостность БД не гарантирует достоверности содержащейся в ней информации, но обеспечивает по крайней мере правдоподобность этой информации, отвергая заведомо невероятные, невозможные значения. Таким образом, не следует путать целостность БД с достоверностью БД. Достоверность (или истинность) есть соответствие фактов, хранящихся в базе данных, реальному миру. Очевидно, что для определения достоверности БД требуется обладание полными знаниями как о содержимом БД, так и о реальном мире. Для определения целостности БД требуется лишь обладание знаниями о содержимом БД и о заданных для неё правилах. Поэтому СУБД может (и должна) контролировать целостность БД, но принципиально не в состоянии контролировать достоверность БД. Контроль достоверности БД может быть возложен только на человека, да и то в ограниченных масштабах, поскольку в ряде случаев люди тоже не обладают полнотой знаний о реальном мире.

Практика

Создание схемы данных для базы, которая позволяет не установить связи, а **активизировать средства сохранения целостности данных**.

На примере таблиц «Студенты» и «Группы», либо «Продукты» и «Категории».

Домашнее задание



Требования к схеме данных

- Названия таблиц и полей, все связи должны быть как на картинке
- В каждой таблице должно быть по 5-15 записей