Базы данных и Информационные системы

4/15 Ссылочная целостность

Кузиков Б.О.
Сумы, СумГУ
2013

Задачи занятия

- После завершения занятия вы должны уметь и знать следующее:
  - Ограничения на значения атрибутов
  - Создавать и поддерживать ограничения
  - Создавать первичные и внешние ключи

Что есть ограничение?

- Определяет правило на уровне таблицы.
- Предотвращает удаление таблицы в случае наличия зависимостей.
- В Oracle существуют следующие виды ограничений:
  - NOT NULL
  - UNIQUE Key
  - PRIMARY KEY
  - FOREIGN KEY
  - CHECK

Рекомендации по использованию

- Именуйте ограничения, иначе Oracle Server создаст имя в виде SYS_C*n*.
- Создать ограничение можно:
  - Во время создания таблицы
  - После того, как таблица была создана
- Ограничение можно задать на уровне таблицы и на уровне столбца.
- Ограничения можно просмотреть в каталоге данных.

Определение ограничений

```
CREATE TABLE [schema.]table
             (column datatype [DEFAULT expr]
              [column_constraint],
             …
              [table constraint]);
```

```
CREATE TABLE emp(
              empno   NUMBER(4),
              ename   VARCHAR2(10),
             …
              deptno   NUMBER(7,2) NOT NULL,
              CONSTRAINT emp_empno_pk
                           PRIMARY KEY (EMPNO));
```

Определение ограничений

- На уровне столбца

```
column [CONSTRAINT constraint_name] constraint_type,
```

- На уровне таблицы

```
column,...
   [CONSTRAINT constraint_name] constraint_type
   (column, ...),
```

```
CREATE TABLE NEW_EXAM_MARKS(

      STUDENT_ID INTEGER NOT NULL,

      SUBJ_ID INTEGER,

      MARK INTEGER,

CONSTRAINT EXAM_PR_KEY PRIMARY KEY (STUDENT_ID,
SUBJ_ID));
```

Ограничение NOT NULL

- Гарантирует, что в столбце не могут быть использованы значения Null

NULL

- NULL – обозначение пустоты (недостатка информации).
- С NULL нельзя сравнивать

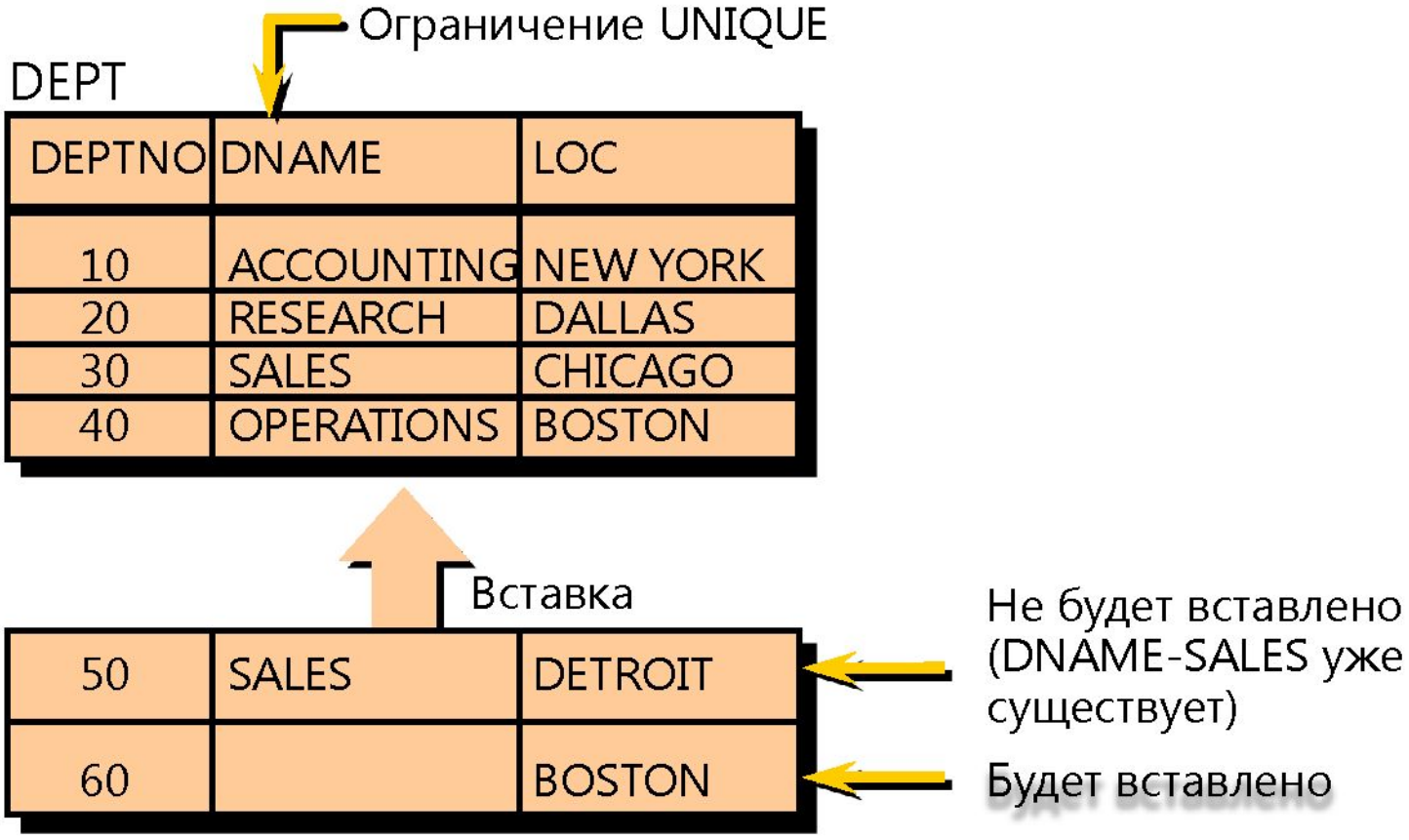
Условие	Значение A	Результат
a IS NULL	10	FALSE
a IS NOT NULL	10	TRUE
a IS NULL	NULL	TRUE
a IS NOT NULL	NULL	FALSE
a = NULL	10	UNKNOWN
a != NULL	10	UNKNOWN
a = NULL	NULL	UNKNOWN
a != NULL	NULL	UNKNOWN
a = 10	NULL	UNKNOWN
a != 10	NULL	UNKNOWN

Ограничение NOT NULL

- Определяется только на уровне столбца

```
SQL> CREATE TABLE emp(
  2       empno     NUMBER(4),
  3       ename     VARCHAR2(10) NOT NULL,
  4       job       VARCHAR2(9),
  5       mgr       NUMBER(4),
  6       hiredate  DATE,
  7       sal       NUMBER(7,2),
  8       comm      NUMBER(7,2),
  9       deptno    NUMBER(7,2) NOT NULL);
```

Ограничение UNIQUE (уникальный ключ)



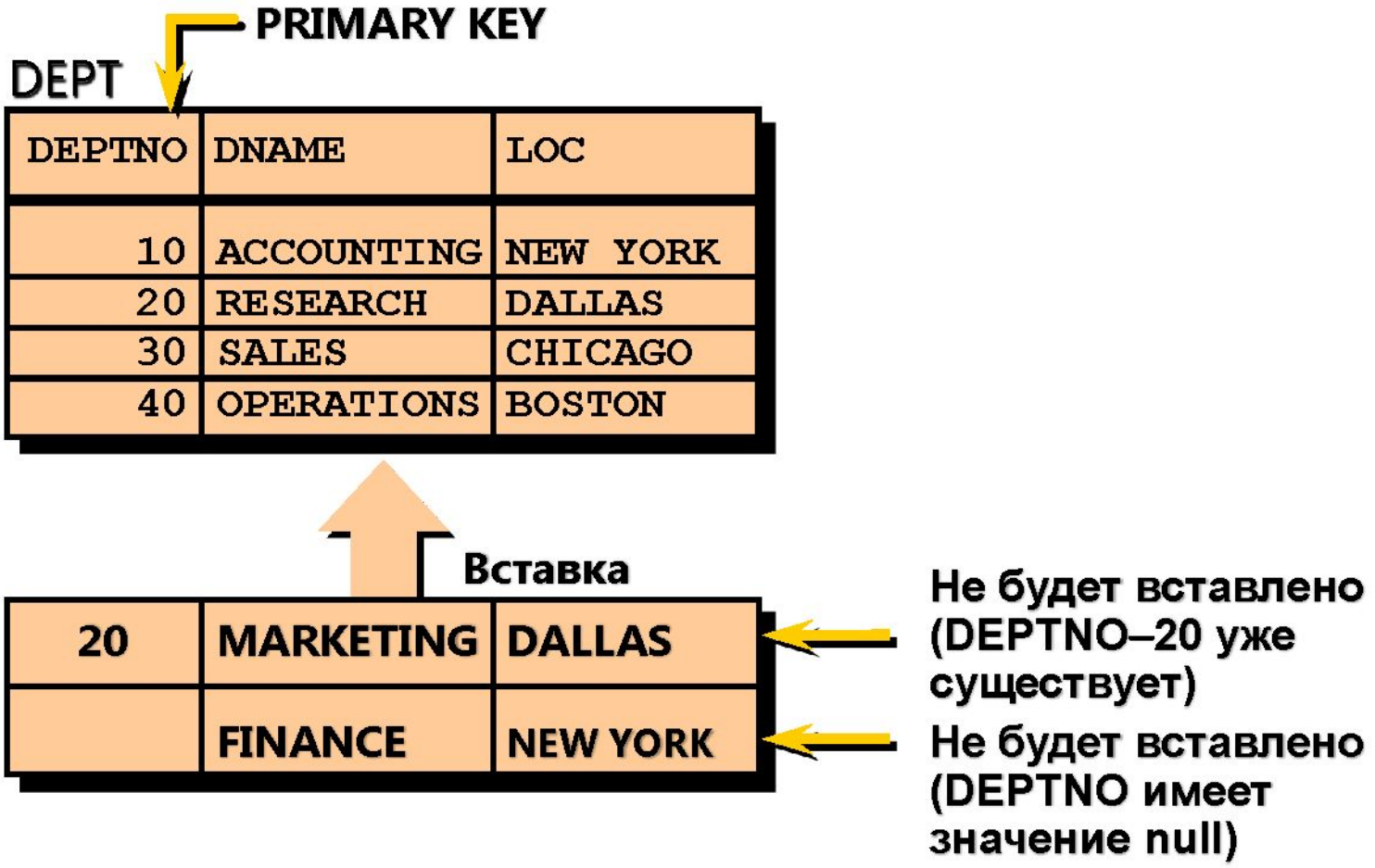
Ограничение UNIQUE

- Определяется как на уровне столбца, так и на улице таблицы

```
SQL> CREATE TABLE    dept(
  2       deptno      NUMBER(2),
  3       dname       VARCHAR2(14),
  4       loc         VARCHAR2(13),
  5       CONSTRAINT dept_dname_uk UNIQUE(dname));
```

```
CREATE TABLE Student(
      STUDENT_ID INTEGER,
      Name VARCHAR(20),
      Surname VARCHAR(20),
CONSTRAINT STUDENT_UNIQ UNIQUE (Name, Surname));
```

Ограничение PRIMARY KEY



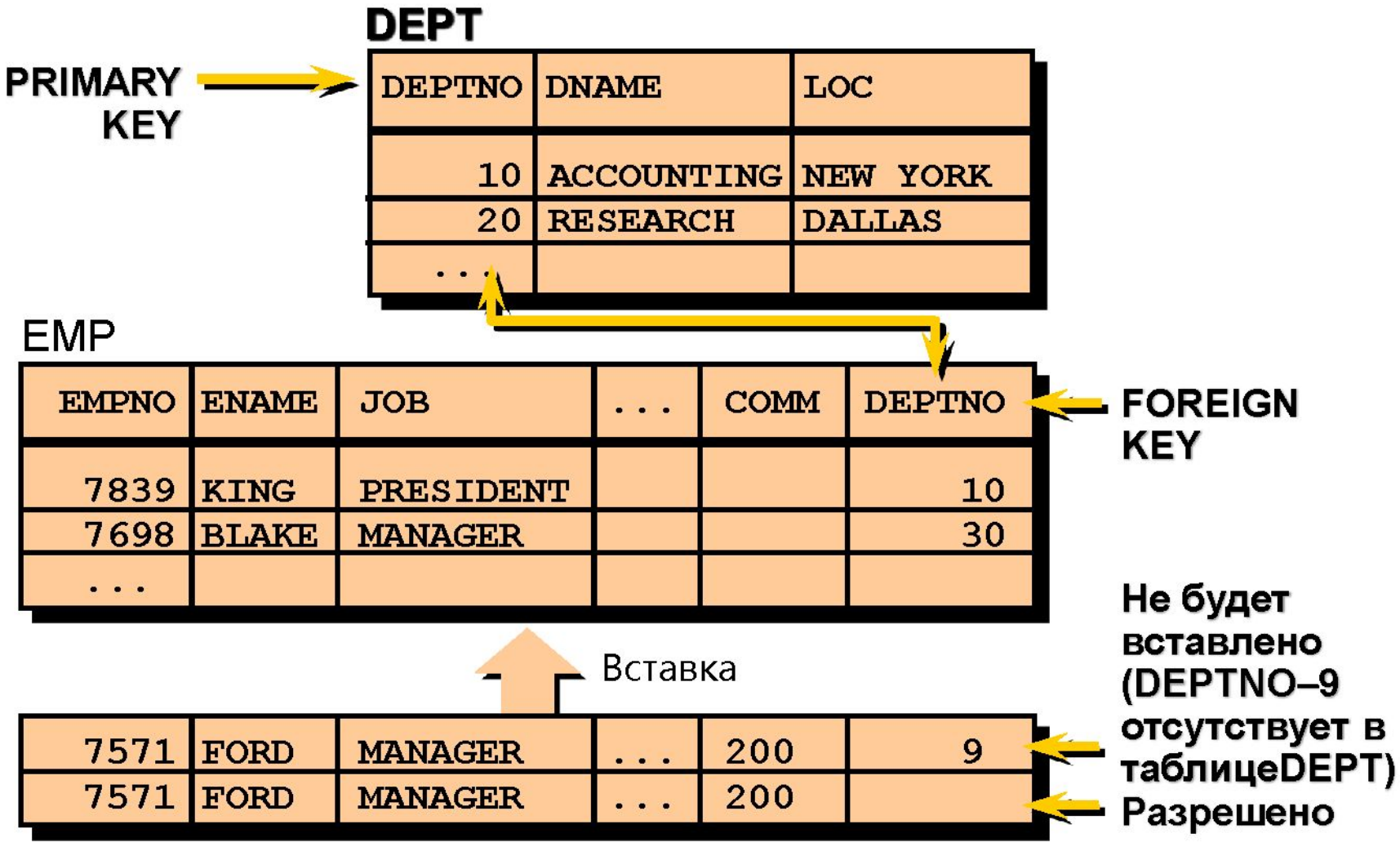
Ограничение PRIMARY KEY

- Определяется как на уровне столбца, так и на улице таблицы

```
SQL> CREATE TABLE    dept(
  2        deptno      NUMBER(2),
  3        dname       VARCHAR2(14),
  4        loc         VARCHAR2(13),
  5        CONSTRAINT dept_deptno_pk PRIMARY KEY(deptno));
```

```
SQL> CREATE TABLE    dept(
  2        deptno      NUMBER(2) PRIMARY KEY,
  3        dname       VARCHAR2(14),
  4        loc         VARCHAR2(13)
  );
```

Ограничение FOREIGN KEY



Ограничение FOREIGN KEY

- Определяется как на уровне столбца, так и на улице таблицы

```
SQL> CREATE TABLE emp(
  2       empno     NUMBER(4),
  3       ename     VARCHAR2(10) NOT NULL,
  4       job       VARCHAR2(9),
  5       mgr       NUMBER(4),
  6       hiredate  DATE,
  7       sal       NUMBER(7,2),
  8       comm      NUMBER(7,2),
  9       deptno    NUMBER(7,2) NOT NULL,
 10       CONSTRAINT emp_deptno_fk FOREIGN KEY (deptno)
 11                  REFERENCES dept (deptno));
```

Ключевые слова ограничения FOREIGN KEY

- FOREIGN KEY

  Определяет столбец дочерней таблицы на уровне ограничений таблицы
- REFERENCES

  Определяет родительскую таблицу и столбец в ней
- ON DELETE CASCADE

  Позволяет удалять строки родительской таблицы вместе со связанными элементами дочерней таблицы
- ON DELETE SET NULL

  При удалении из родительской таблицы в дочерней таблице значение полей с внешними ключами зануляется

Ограничение CHECK

- Определяет условие, которому должны удовлетворять все строки в таблице
- Не допускаются следующие выражения:
  - Ссылки на псевдостолбцы CURRVAL, NEXTVAL, LEVEL, и ROWNUM
  - Вызовы функций SYSDATE, UID, USER, и USERENV
  - Запросы к значениям в других строках

```
..., deptno   NUMBER(2),
      CONSTRAINT emp_deptno_ck
            CHECK (DEPTNO BETWEEN 10 AND 99),...
```

Добавление ограничений

```
ALTER TABLE  table
ADD [CONSTRAINT constraint] type (column);
```

- Ограничение можно добавить или убрать, но не модифицировать
- Ограничение можно активировать и деактивировать
- Ограничение NOT NULL добавляется с помощью предложения MODIFY

Добавление ограничений

- Добавление ограничения FOREIGN KEY в таблицу EMP, регламентирующего, что указываемый менеджер должен уже существовать в таблице EMP.

```
SQL> ALTER TABLE      emp
  2  ADD CONSTRAINT  emp_mgr_fk
  3           FOREIGN KEY(mgr) REFERENCES emp(empno);
Table altered.
```

Удаление ограничений

- Удаление ограничения на менеджера из таблицы EMP.

```
SQL> ALTER TABLE       emp
  2  DROP CONSTRAINT  emp_mgr_fk;
Table altered.
```

- Удаление ограничения PRIMARY KEY в таблице DEPT с удалением всех зависимых ограничений FOREIGN KEY.

```
SQL> ALTER TABLE    dept
  2  DROP PRIMARY KEY CASCADE;
Table altered.
```

Деактивация ограничений

- Используйте предложение DISABLE выражения ALTER TABLE для деактивации ограничения.
- Используйте параметр CASCADE для удаления всех зависимых ограничений.

```
SQL> ALTER TABLE              emp
  2  DISABLE CONSTRAINT       emp_empno_pk CASCADE;
Table altered.
```

Активация ограничений

- Для активации ограничения, деактивированного в данный момент, используйте предложение ENABLE.

```
SQL> ALTER TABLE              emp
  2  ENABLE CONSTRAINT        emp_empno_pk;
Table altered.
```

- Индексы UNIQUE и PRIMARY KEY автоматически формируются при активации ограничений UNIQUE или PRIMARY KEY.

Просмотр ограничений

- Запросы к таблице USER_CONSTRAINTS позволяют просмотреть все ограничения.

```
SQL>  SELECT  constraint_name, constraint_type,
  2           search_condition
  3   FROM    user_constraints
  4   WHERE   table_name = 'EMP';
```

```
CONSTRAINT_NAME              C SEARCH_CONDITION
---------------------------- - ------------------------
SYS_C00674                   C EMPNO IS NOT NULL
SYS_C00675                   C DEPTNO IS NOT NULL
EMP_EMPNO_PK                 P
...
```

Просмотр столбцов, связанных с ограничениями

- Просмотреть столбцы, задействованные в ограничении, по его имени, можно с помощью запроса к USER_CONS_COLUMNS

```
SQL> SELECT   constraint_name, column_name
  2  FROM     user_cons_columns
  3  WHERE    table_name = 'EMP';
```

```
CONSTRAINT_NAME              COLUMN_NAME
---------------------------- --------------------
EMP_DEPTNO_FK                DEPTNO
EMP_EMPNO_PK                 EMPNO
EMP_MGR_FK                   MGR
SYS_C00674                   EMPNO
SYS_C00675                   DEPTNO
```

Выводы

- Создавайте следующие типы ограничений:
  - NOT NULL
  - UNIQUE key
  - PRIMARY KEY
  - FOREIGN KEY
  - CHECK
- Запросы к таблице USER_CONSTRAINTS позволяют просмотреть все ограничения и их имена.

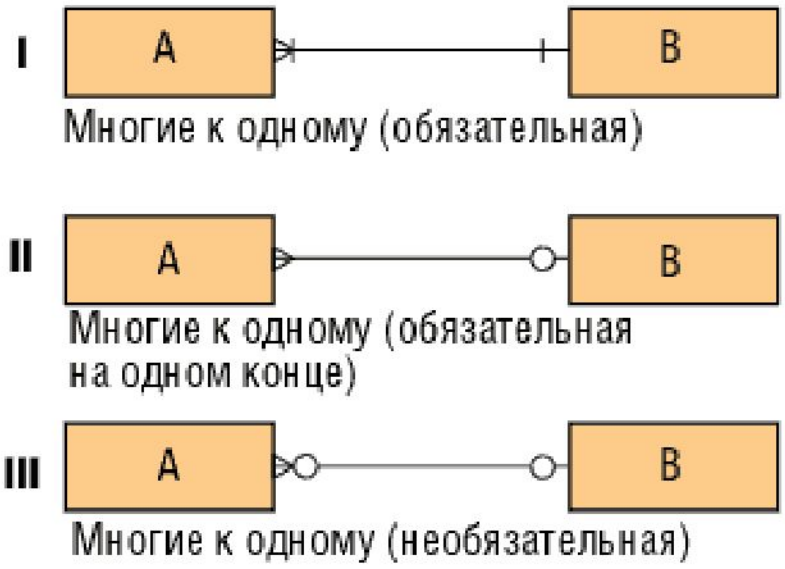
Домашнее чтение

- Типичные виды связей на ER-диаграммах и их применимость
  - Один-к-одному
  - Один-к-многим
  - Многие-ко-многим
  - Рекурсивные связи

Один-к-одному

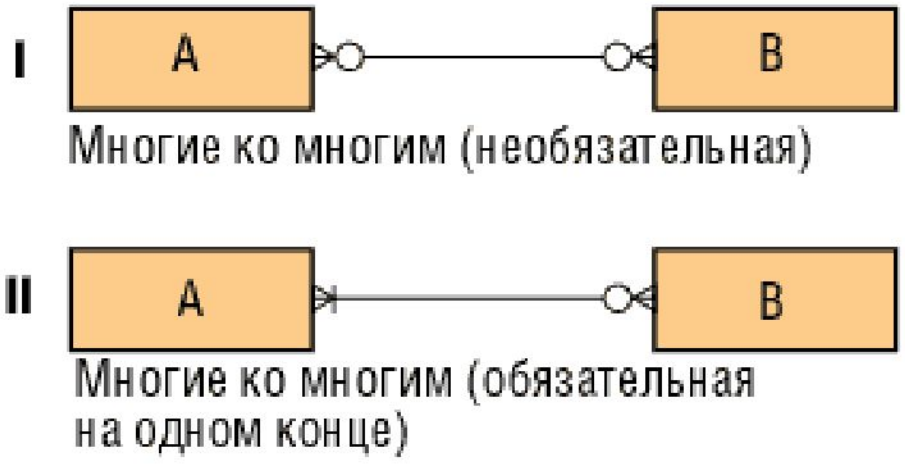
- При ближайшем рассмотрении связи типа "один к одному" почти всегда оказывается, что A и B представляют собой в действительности разные подмножества одного и того же предмета или разные точки зрения на него, просто имеющие отличные имена и по-разному описанные связи и атрибуты.

Один-ко-многим



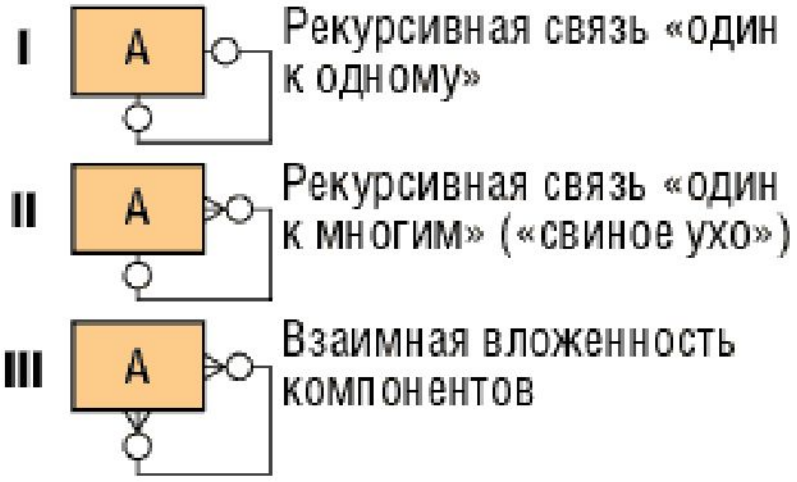
- I - достаточно сильная конструкция, предполагающая, что вхождение сущности B не может быть создано без одновременного создания по меньшей мере одного связанного с ним вхождения сущности A.
- II - это наиболее часто встречающаяся форма связи. Она предполагает, что каждое и любое вхождение сущности A может существовать только в контексте одного (и только одного) вхождения сущности B. В свою очередь, вхождения B могут существовать как в связи с вхождениями A, так и без нее.
- III - применяется редко. Как A, так и B могут существовать без связи между ними.

Многие-ко-многим



- I - такая конструкция часто имеет место в начале этапа анализа и означает связь - либо понятую не до конца и требующую дополнительного разрешения, либо отражающую простое коллективное отношение - двунаправленный список.
- II - применяется редко. Такие связи всегда подлежат дальнейшей детализации.

Рекурсивные связи



- I - редко, но имеет место. Отражает связи альтернативного типа.
- II - достаточно часто применяется для описания иерархий с любым числом уровней.
- III - имеет место на ранних этапах. Часто отражает структуру "перечня материалов" (взаимная вложенность компонентов). Пример: каждый КОМПОНЕНТ может состоять из одного и более (других) КОМПОНЕНТОВ и каждый КОМПОНЕНТ может использоваться в одном и более (других) КОМПОНЕНТОВ.

Рекурсивные связи