Лекция 2

Семантическая модель Entity-Relationship (ER)

Семантические модели данных

Широкое распространение SQL-ориентированных СУБД и их использование в самых разнообразных приложениях показывает, что реляционная модель данных достаточна для моделирования разнообразных предметных областей.

Однако реляционная модель проявляет ограниченность в следующих аспектах:

- Модель не обеспечивает достаточных средств для представления смысла данных
- Реляционная модель не предоставляет какие-либо формализованные средства для представления внутренних зависимостей.

Семантические модели данных

- Потребность проектировщиков баз данных в более удобных и мощных средствах моделирования предметной области привела к появлению семантических моделей данных.
- Основным назначением семантических моделей данных является обеспечение возможности выражения семантики данных.
- Чаще всего семантическое моделирование используется на первой стадии проектирования базы данных. В терминах семантической модели производится концептуальная схема базы данных, которая затем вручную преобразуется к реляционной схеме.

Семантические модели данных

Достоинства подхода:

- Построение мощной и наглядной концептуальной схемы БД позволяет более полно оценить специфику моделируемой предметной области и избежать возможных ошибок на стадии проектирования схемы реляционной БД.
- На этапе семантического моделирования производится важная документация, которая может оказаться очень полезной не только при проектировании схемы реляционной БД, но и при эксплуатации, сопровождении и развитии уже заполненной БД.

Семантическая модель Entity-Relationship

На основе ER-модели (модели "сущность-связь") создано большинство современных подходов к проектированию БД. Эта модель была предложена Питером Ченом (Peter Chen) в 1976г.

Моделирование предметной области базируется на использовании графических диаграмм, включающих небольшое число разнородных компонентов. Простота и наглядность представления концептуальных схем баз данных в ER-модели привели к её широкому распространению в CASE-системах, поддерживающих автоматизированное проектирование баз данных.

Основные понятия ER-модели

Основными понятиями ER-модели являются *сущность*, *связь* и *атрибут*.

Сущность – это реальный или представляемый объект, информация о котором должна сохраняться и быть доступной.

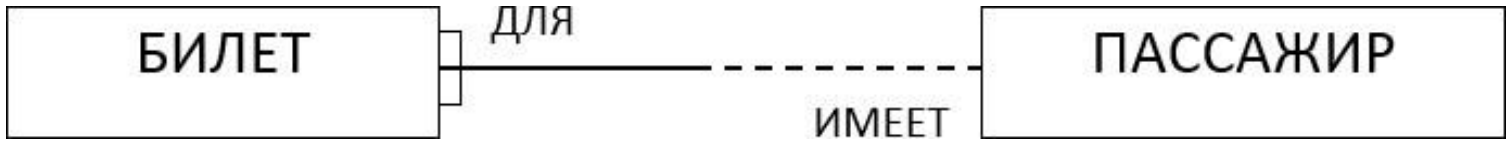
В диаграммах ER-модели сущность представляется в виде прямоугольника, содержащего имя сущности. При этом имя сущности – это имя типа, а не некоторого конкретного экземпляра этого типа.

Основные понятия ER-модели

Основными понятиями ER-модели являются *сущность*, *связь* и *атрибут*.

Сущность – это реальный или представляемый объект, информация о котором должна сохраняться и быть доступной.

В диаграммах ER-модели сущность представляется в виде прямоугольника, содержащего имя сущности. При этом имя сущности – это имя типа, а не некоторого конкретного экземпляра этого типа.

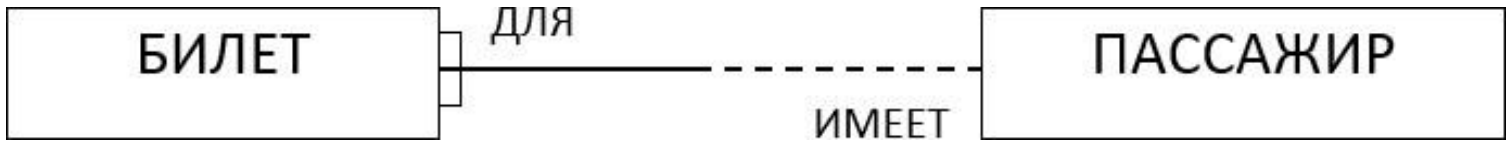


Основные понятия ER-модели

Связь – это графически изображаемая ассоциация, устанавливаемая между двумя типами сущностей.

В любой связи выделяются два конца (в соответствии с существующей парой связываемых сущностей), на каждом из которых указываются:

- имя конца связи
- степень конца связи
- обязательность связи

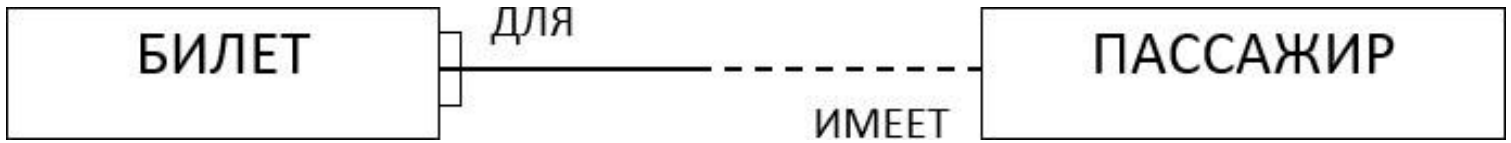


Основные понятия ER-модели

Связь представляется в виде линии, соединяющей две сущности или ведущей от сущности к ней же самой. В месте «стыковки» связи с сущностью используются:

- трёхточечный вход в прямоугольник сущности, если для этой сущности в связи могут (или должны) использоваться много экземпляров сущности
- одноточечный вход, если в связи может (или должен) участвовать только один экземпляр сущности

Обязательный конец связи изображается сплошной линией, а необязательный – прерывистой линией.



Основные понятия ER-модели

Здесь изображена рекурсивная связь, связывающая сущность МУЖЧИНА с ней же самой. Конец связи с именем «сын» определяет тот факт, что несколько мужчин могут быть сыновьями одного отца. Конец связи с именем «отец» означает, что не у каждого мужчины должны быть сыновья.

Основные понятия ER-модели

Атрибут сущности - это любая деталь, которая служит для уточнения, идентификации, классификации, числовой характеристики или выражения состояния сущности. Имена атрибутов заносятся в прямоугольник, изображающий сущность, под именем сущности и изображаются малыми буквами, возможно, с примером.

Основные понятия ER-модели

Атрибуты типа сущности в ER-модели похожи на атрибуты отношения в реляционной модели данных. Введение именованных атрибутов позволяет относиться к сущности как к типовой структуре данных, которой должен удовлетворять каждый экземпляр данной сущности.

Но имеется и важное отличие. В реляционной модели данных атрибут определяется как упорядоченная пара <имя_атрибута, имя_домена>, а при определении атрибутов типа сущности в ER-модели указание домена атрибута не является обязательным, хотя это и возможно.

Уникальные идентификаторы типов сущности

При определении типа сущности необходимо гарантировать, что каждый экземпляр сущности отличим от любого другого экземпляра той же сущности.

Уникальным идентификатором может быть:

- атрибут (или комбинация атрибутов)
- связь (или комбинация связей)
- комбинация атрибутов и связей

Уникальные идентификаторы типов сущности

На примерах:

КНИГА
автор
название
номер издания
издательство
год издания
isbn
число книг на складе

Уникальные идентификаторы типов сущности

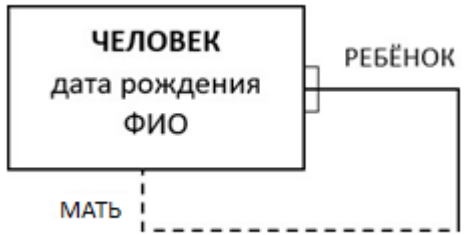
На примерах:

Уникальные идентификаторы типов сущности

На примерах:

Уникальные идентификаторы типов сущности

На примерах:



Более сложные элементы ER-модели

К числу некоторых более сложных элементов модели относятся:

- Подтипы и супертипы сущностей (наследование)
- Уточняемые степени связи
- Взаимно-исключающие связи
- Каскадные удаления экземпляров сущностей
- Домены

Наследование

Тип сущности при проектировании РБД может быть расщеплен на два или более взаимно-исключающих подтипов, каждый из которых включает общие атрибуты и/или связи.

Эти общие атрибуты и/или связи явно определяются один раз на более высоком уровне. В подтипах могут определяться собственные атрибуты и/или связи.

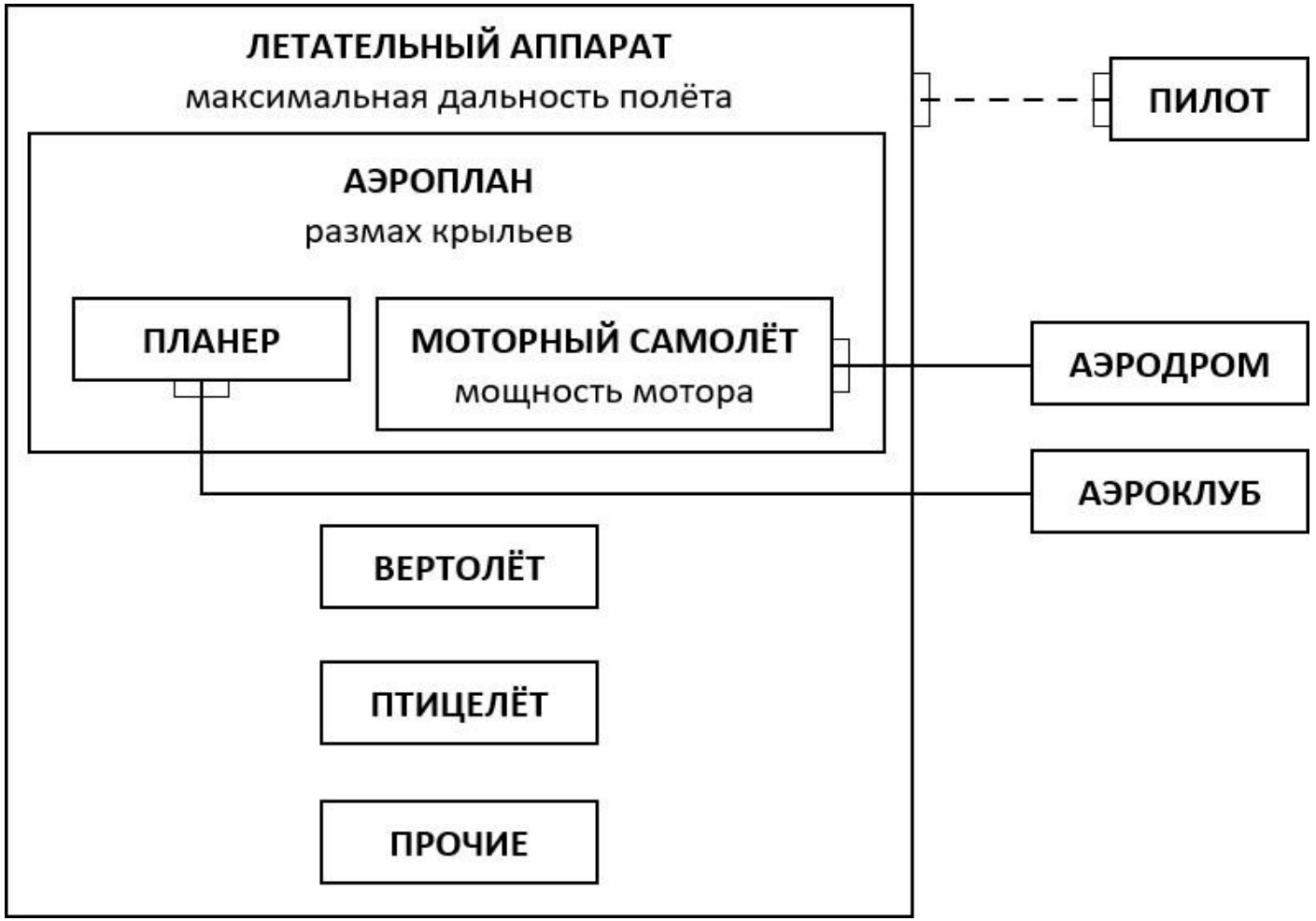
Тип сущности, на основе которого определяются подтипы, называется супертипом.

Наследование

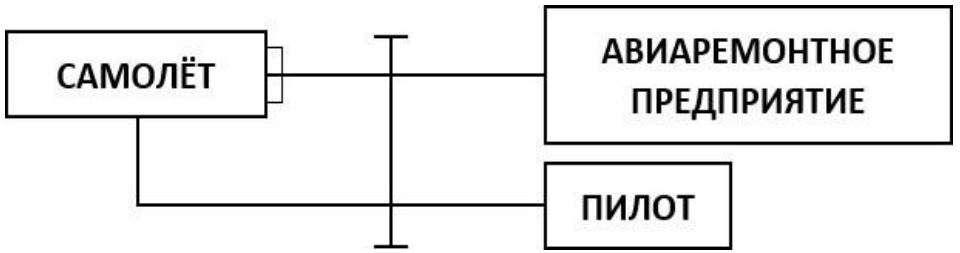
Особенности механизма наследования в ER-модели определяются следующими правилами. Если у типа сущности A имеются подтипы $B_1, B_2, \ldots, B_n$ то:

- любой экземпляр типа сущности $B_1, B_2, \ldots, B_n$ является экземпляром типа сущности A (включение)
- если a является экземпляром типа сущности A, то a является экземпляром некоторого подтипа B_i, где $i = 1,2,\ldots,n$ (отсутствие собственных экземпляров у супертипа)
- ни для каких подтипов B_i и B_j, где $i,j = 1,2,\ldots,n$ не существует экземпляра, типом которого одновременно являются типы сущности B_i и B_j (разъединённость подтипов)

Наследование



Взаимно-исключающие связи



Взаимно-исключающие связи требуют, чтобы существовал экземпляр только одной связи из заданного набора связей. В данном случае для каждого экземпляра типа сущности САМОЛЕТ должен существовать экземпляр одной из указанных связей.

Взаимно-исключающие связи

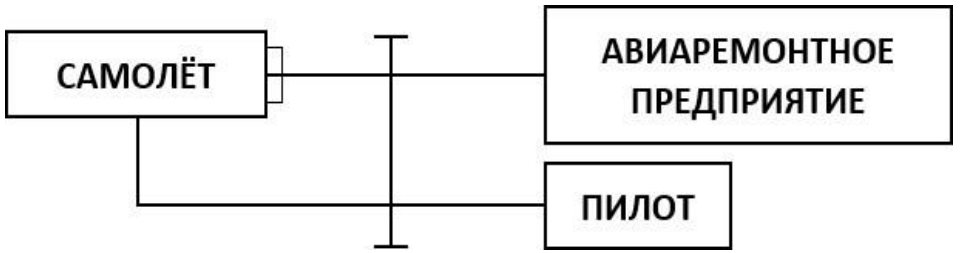
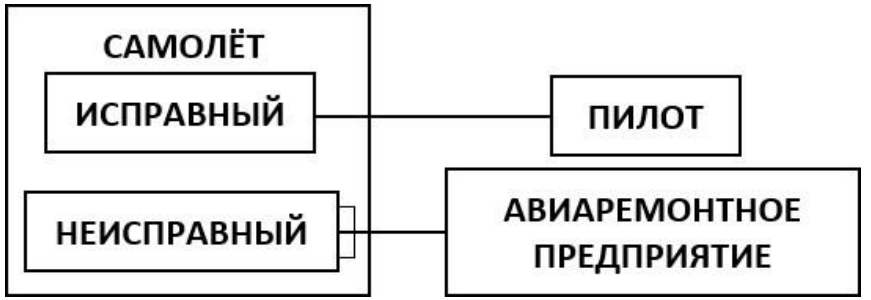


Диаграмма с взаимно-исключающими связями может быть преобразована к диаграмме без взаимно-исключающих связей путем введения подтипов.



Получение реляционной схемы из ER-диаграммы

Простым типом сущности называется тип сущности, не являющийся подтипом и не имеющий подтипов.

Каждый простой тип сущности превращается в таблицу:

- Имя сущности становится именем таблицы.
- Каждый атрибут становится столбцом таблицы; может выбираться более точный формат представления данных.
- Экземпляр типа сущности становится строкой таблицы.
- Уникальный идентификатор сущности становятся первичным ключом таблицы.

Получение реляционной схемы из ER-диаграммы

- Связи «многие к одному» (и «один к одному») становятся внешними ключами, т.е. образуется копия уникального идентификатора сущности на конце связи «один», и соответствующие столбцы составляют внешний ключ таблицы, соответствующей типу сущности на конце связи «многие».
- Если между двумя типами сущности *A* и *B* имеется связь «один к одному», то соответствующий внешний ключ по желанию проектировщика может быть объявлен как в таблице *A*, так и в таблице *B*.

Получение реляционной схемы из ER-диаграммы

- Чтобы отразить в определении таблицы ограничение, которое заключается в том, что степень конца связи равна единице, соответствующий (возможно, составной) столбец должен быть дополнительно специфицирован как возможный ключ таблицы.
- Для поддержки связи «многие ко многим» между типами сущности *A* и *B* создается дополнительная таблица *AB* с двумя столбцами, один из которых содержит уникальные идентификаторы экземпляров сущности *A*, а другой – уникальные идентификаторы экземпляров сущности *B*.

Получение реляционной схемы из ER-диаграммы

- Обозначим через УИД(*c*) уникальный идентификатор экземпляра некоторого типа сущности *C*. Тогда, если в экземпляре связи «многие ко многим» участвуют экземпляры $a_1,a_2,...,a_n$ типа сущности *A* и экземпляры $b_1,b_2,...,b_m$ типа сущности *B*, то в таблице *AB* должны присутствовать все строки вида <УИД(a_i), УИД(b_j)>, где $i = 1,2,...,n; j = 1,2,...,m$.

Супертипы и подтипы

Существует два способа представить наследование сущностей в реляционной схеме.

Первый способ заключается в создании единой таблицы для всех подтипов. Эта таблица содержит столбцы, соответствующие каждому атрибуту (и связям) каждого подтипа, а также один специальный столбец «код подтипа». Для каждой строки таблицы значение этого столбца определяет конкретный подтип, которому соответствует строка.

К достоинствам такого способа можно отнести:

- обеспечение простого доступа к экземплярам супертипа и не слишком сложный доступ к экземплярам подтипов
- возможность обойтись небольшим числом таблиц

Супертипы и подтипы

Существует два способа представить наследование сущностей в реляционной схеме.

Первый способ заключается в создании единой таблицы для всех подтипов. Эта таблица содержит столбцы, соответствующие каждому атрибуту (и связям) каждого подтипа, а также один специальный столбец «код подтипа». Для каждой строки таблицы значение этого столбца определяет конкретный подтип, которому соответствует строка.

К недостаткам можно отнести:

- приложения, работающие с одной таблицей супертипа, должны содержать дополнительный программный код для работы с разными наборами столбцов и разными ограничениями целостности
- общая для всех подтипов таблица потенциально может стать узким местом при многопользовательском доступе по причине возможности блокировки таблицы целиком
- потенциально в общей таблице будет содержаться много неопределенных значений, что может привести к непроизводительному расходу внешней памяти

Супертипы и подтипы

Второй способ предполагает *создание отдельной таблицы для каждого подтипа.* В этом случае для получения всех кортежей супертипа нужно объединить проекции таблиц подтипов, на заголовок таблицы супертипа. Другими словами, из всех таблиц подтипов выбираются общие столбцы супертипа.

Достоинства:

- действуют более понятные правила работы с подтипами (каждому подтипу соответствует отдельная таблица)
- упрощается логика приложений; каждая программа работает только с нужной таблицей

Супертипы и подтипы

Второй способ предполагает *создание отдельной таблицы для каждого подтипа.* В этом случае для получения всех кортежей супертипа нужно объединить проекции таблиц подтипов, на заголовок таблицы супертипа. Другими словами, из всех таблиц подтипов выбираются общие столбцы супертипа.

Недостатки:

- в общем случае требуется слишком много отдельных таблиц
- работа с экземплярами супертипа на основе представления, объединяющего таблицы супертипов, может оказаться недостаточно эффективной
- поскольку множество экземпляров супертипа является объединением множеств экземпляров подтипов, не все РСУБД могут обеспечить выполнение операций модификации экземпляров супертипа

Взаимно-исключающие связи

Существуют два способа формирования схемы реляционной БД при наличии взаимно-исключающих связей (имеются в виду связи «один ко многим», причем конец связи «многие» находится на стороне сущности, для которой связи являются взаимно-исключающими):

- определение таблицы с одним столбцом для представления всех взаимноисключающих связей, т.е. общее хранение внешних ключей
- определение таблицы, в которой каждой взаимно-исключающей связи соответствует отдельный столбец, т.е. раздельное хранение внешних ключей

Взаимно-исключающие связи

Преимущество первого подхода состоит в том, что в таблице, соответствующей сущности с взаимно-исключающими связями, появляется всего два дополнительных столбца. Недостатком является усложнение выполнения операции соединения.

При использовании второго подхода соединения являются явными (и естественными). Недостаток состоит в том, что требуется иметь столько столбцов, сколько имеется альтернативных связей. Кроме того, в каждом из таких столбцов будет содержаться много неопределенных значений, хранение которых может привести к излишнему расходу внешней памяти.