Язык UML (Unified Modeling Language)

Предложен: 1996 г., Г. Буч, Д. Румбах, А. Якобсон
Назначение: моделирование разных систем (аппаратных, программных, смешанных, с участием человека, и т.д.), описание их статических (структурных) и динамических (поведенческих) свойств
Стандарт: OMG (www.omg.org), текущая версия 2.5.1 (декабрь 2017 г.)
Нотация: графическая (диаграммы), стандарт определяет 14 типов диаграмм
CASE системы: IBM/Rational Software Architect, IBM Rapsody, Borland Together, Sparx Enterprise Architect, Objecteering, Umbrello

Диаграммы классов UML: основные понятия

Диаграмма классов описывает типы объектов моделируемой системы и статические отношения различного рода, которые существуют между ними. Диаграмма классов может включать комментарии и ограничения.
Класс – именованное описание совокупности объектов с общими атрибутами, операциями, связями и семантикой.
Атрибут класса – именованное свойство класса, описывающее множество значений, которые могут принимать экземпляры этого свойства (абстракция состояния объекта).
Операция класса – именованная услуга, которую можно запросить у любого объекта этого класса (абстракция поведения объекта).
Стереотип – механизм расширения семантики UML, позволяющий создавать новые элементы UML на основе существующих с учетом особенностей решаемой задачи.

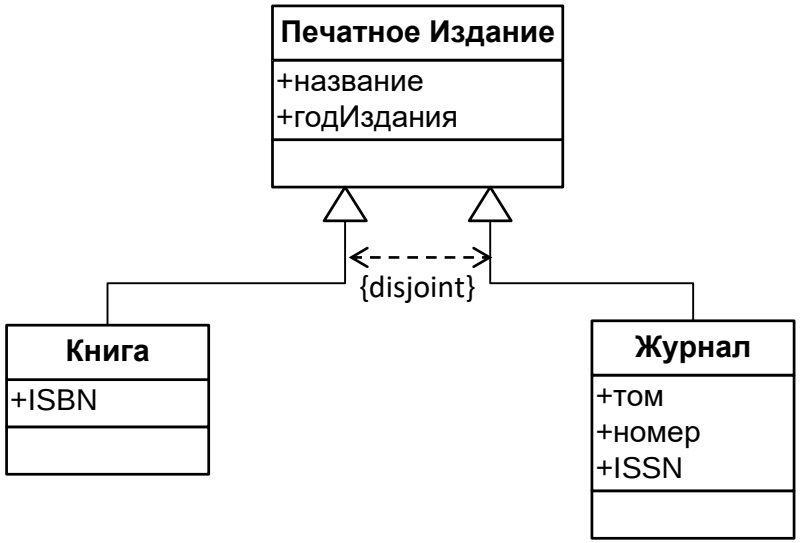
Диаграммы классов UML: пример

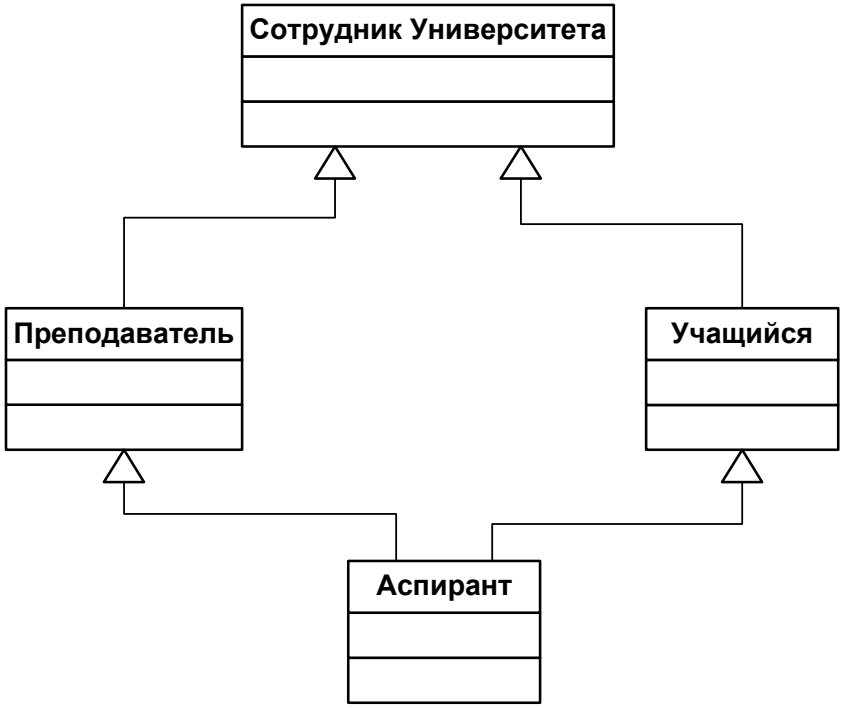
Категории связей UML

Три категории связей: зависимость, обобщение, ассоциация.
Зависимостью называется связь по применению, когда изменение в спецификации одного класса может повлиять на поведение другого класса, использующего первый. Изображение: пунктирная линия, направленная от зависимого класса.
Обобщением называется связь между общим классом (суперклассом) и более специализированной его разновидностью (подклассом). Изображение: сплошная линия с незакрашенным треугольником, направленным к суперклассу.
Правила наследования в ER-модели (см. предыдущую лекцию):

1. Включение ($\forall b \in B_i \rightarrow b \in A$, $i = 1, \ldots, n$) – **выполняется в UML**
2. Отсутствие собственных экземпляров у супертипа ($\forall a \in A \rightarrow a \in B_i$, $i = 1, \ldots, n$) – **не выполняется в UML по умолчанию** (исключение: объявление суперкласса абстрактным)
3. Разъединенность подтипов ($\forall b \in B_i \rightarrow b \notin B_j$, $i \neq j$) – **не выполняется в UML по умолчанию** (исключение: ограничение обобщения {disjoint})

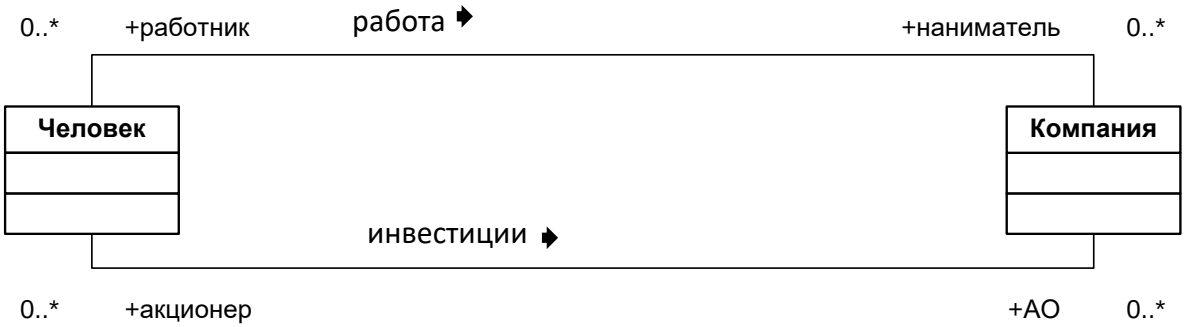
Пример с использованием связей-обобщений UML





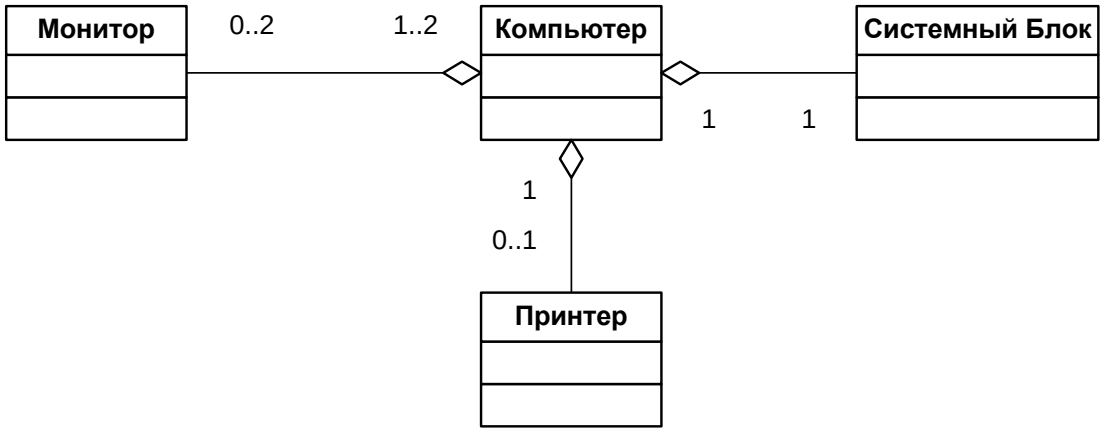
Связи-ассоциации в UML

Ассоциацией называется структурная связь между объектами одного класса и объектами другого или того же самого класса. Допускается создание n-арных ассоциаций, связывающих сразу несколько классов. Изображение: сплошная линия (в общем случае ненаправленная).
Ассоциации может быть присвоено имя, характеризующее природу связи (указывается над линией связи посередине).
Роль класса также характеризуется именем, помещаемым над линией ассоциации ближе к данному классу.
Кратность роли – характеристика, указывающая сколько объектов класса с данной ролью может или должно участвовать в каждом экземпляре ассоциации. Задается числом (1), диапазоном (0..1) или списком чисел и/или диапазонов (2, 4..6, 8..*).

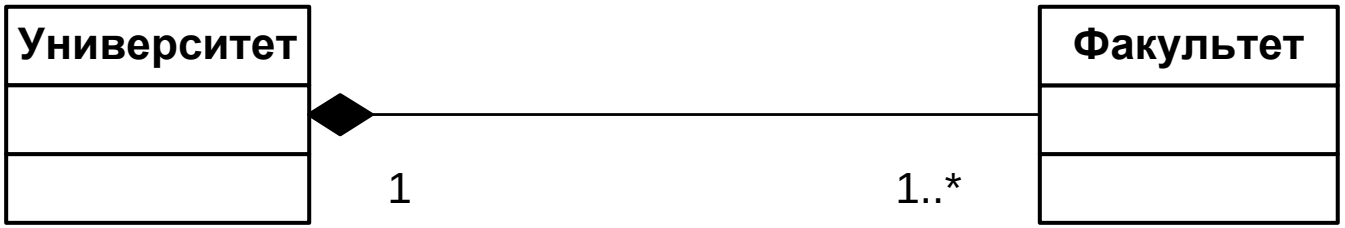


Агрегатные ассоциации в UML

Агрегатные ассоциации используются в UML для отображения отношений «часть—целое» (класс «целое» имеет более высокий концептуальный уровень, чем часть).

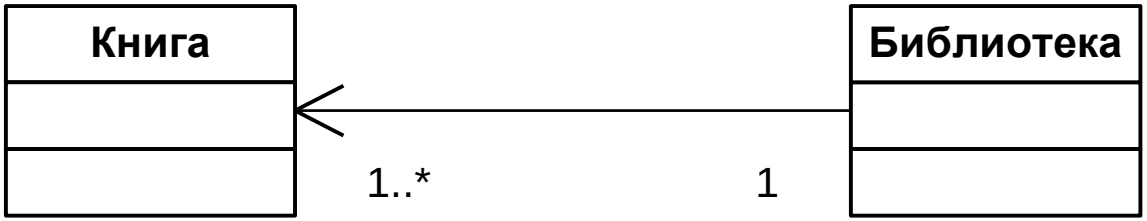


Бывают случаи, когда связь «части» и «целого» настолько сильна, что каждая часть может принадлежать одному и только одному целому и уничтожение целого приводит к уничтожению всех его частей. Агрегатные ассоциации, обладающие данным свойством, называются композициями.

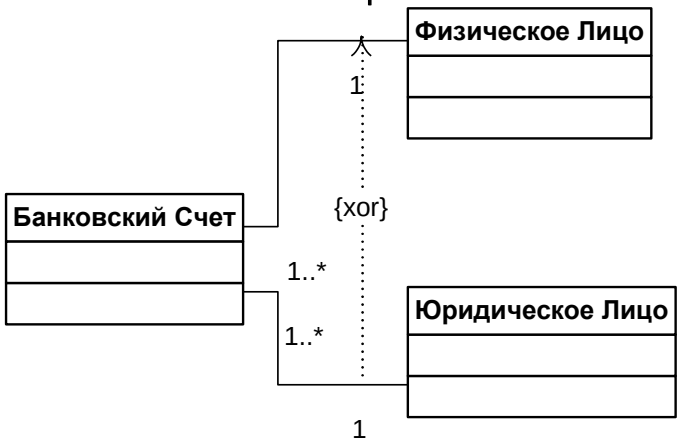


Навигация и взаимно исключающие связи

При наличии ассоциации предполагается возможность навигации между объектами, входящими в экземпляр ассоциации. По умолчанию в UML навигация может проводиться в обоих направлениях. В тех случаях, когда требуется ограничить направление навигации, на линии ассоциации ставится стрелка, указывающая требуемое направление.



Для организации взаимно исключающих связей используется ограничение ассоциации {xor}.



Рекомендации по использованию UML для концептуального проектирования реляционных БД

1. Алгоритм преобразования ER-модели в реляционную применим и к UML.
2. Операции в классах могут быть реализованы не во всякой РСУБД.
3. Связи-зависимости при преобразованиях в РСУБД не используются.
4. Старайтесь избегать множественного наследования и осторожно используйте одиночное наследование классов.
5. Эффективно в РСУБД реализуются только ассоциации «один-ко-многим».
6. Реализация ассоциаций с точно заданными кратностями ролей возможна, но требует определения дополнительных ограничений, что может привести к снижению эффективности.
7. Для РСУБД агрегатные ассоциации неестественны, композиции влияют, как правило, только на способ поддержки ссылочной целостности (каскадное удаление).
8. Для РСУБД поддержка однонаправленных ассоциаций вызывает дополнительные накладные расходы.
9. Определение большого числа ограничений может привести к снижению эффективности.

Язык OCL (Object Constraint Language)

Предложен: 1999 г., часть спецификации UML 1.3
Назначение: определение ограничений целостности данных, соответствующих моделям, представленным в терминах диаграмм UML
Стандарт: OMG (www.omg.org), текущая версия 2.4 (февраль 2014 г.)
Нотация: текстовая
Тип языка: декларативный, стандарт не определяет правила вычисления выражений, а также правила их отображения в императивные языки
CASE системы: Borland Together

Язык OCL (Object Constraint Language)

На диаграммах классов OCL может применяться для определения ограничений, описывающих пред- и постусловия операций классов, а также ограничений, являющихся инвариантами классов.

Инвариант класса – логическое выражение, при вычислении которого для любого объекта данного класса должно получаться значение true в течение всего времени существования этого объекта.

context <class_name> **inv**:
<OCL logical expression>

В выражениях OCL могут использоваться:
- Предопределенные типы данных OCL и операции над данными типами
- Типы данных, определяемые моделью UML (классы)
- Атрибуты классов и роли ассоциаций
- Операции классов, не изменяющие состояния моделируемой системы

Типы данных OCL

- Скалярные
  - **Integer**
  - **Real**
  - **Boolean**
  - **String**
- Коллекции
  - **set** (неупорядоченная коллекция, не содержащая одинаковых элементов)
  - **bag** (неупорядоченная коллекция, которая может содержать одинаковые элементы)
  - **sequence** (упорядоченная коллекция, которая может содержать одинаковые элементы)
  - **orderedSet** (упорядоченная коллекция, не содержащая одинаковых элементов)

Операции над скалярными типами OCL

Скалярный тип	Список операций
Boolean	**and**, **or**, **xor**, **not**, **implies**, **if-then-else-endif**
Integer	*, +, -, /, **abs**(), **div**(), **mod**(), **min**(), **max**(), операции сравнения
Real	*, +, -, /, **abs**(), **floor**(), **round**(), **min**(), **max**(), операции сравнения
String	**concat**(), **size**(), **substring**(), **toLower**(), **toUpper**()

Операции над объектными типами (классами UML)

- Получение значения атрибута

<объект>.<имя атрибута>

- Переход по экземпляру ассоциации

<объект>.<имя роли, противоположной по отношению к объекту>

- Вызов операции класса

<объект>.<имя операции>(<список фактических параметров>)

Литерал **self** – текущий объект класса, в котором определен инвариант

Операции над коллекциями OCL

Используемая нотация: <коллекция> -> <имя операции>(<список фактических параметров>)

Конструкторы коллекций:
select(<лог. выражение>)
reject(<лог. выражение>)
collect(<выражение>)

Кванторы:
exists(<лог. выражение>)
forAll(<лог. выражение>)

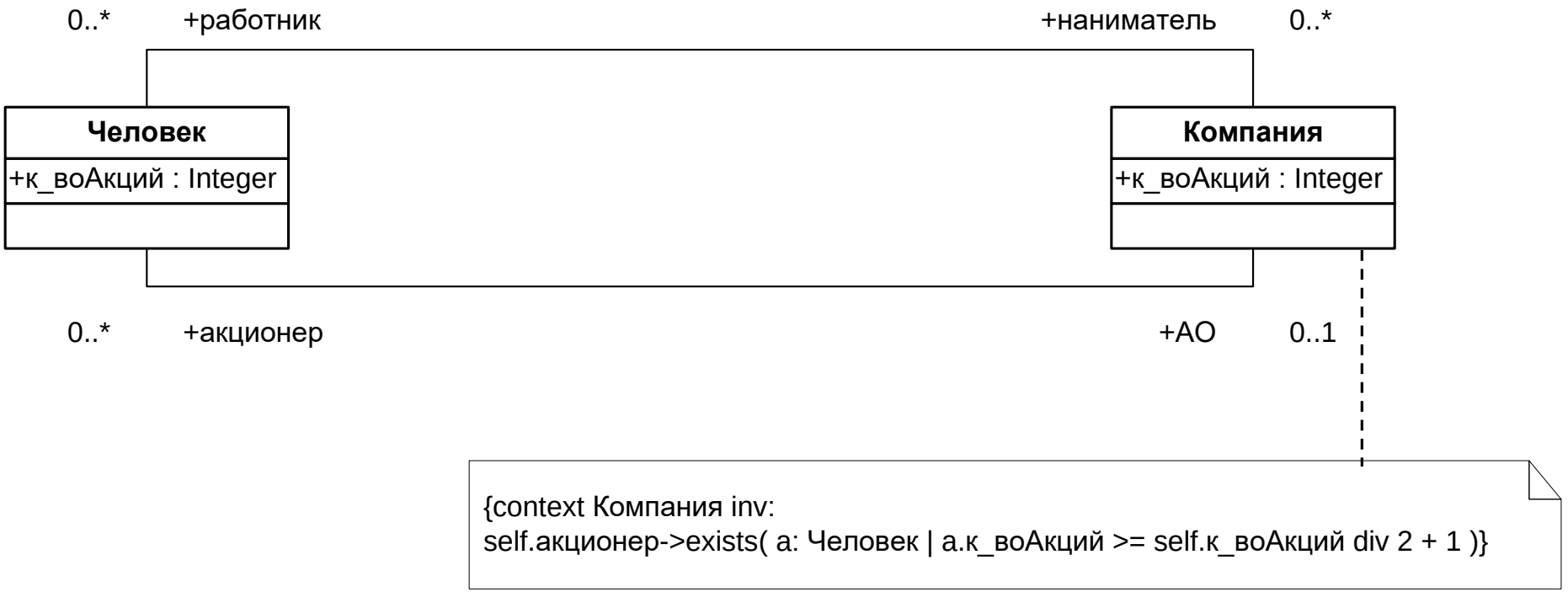
Теоретико-множественные операции:
union
intersect
-

Прочие операции:
count(<element>)
includes(<element>)
excludes(<element>)
size()
at(<индекс>)

В выражениях может использоваться конструкция iterator | body, где iterator – определение переменной, на каждом шаге принимающей значение текущего элемента коллекции, а body – выражение, в котором участвует iterator

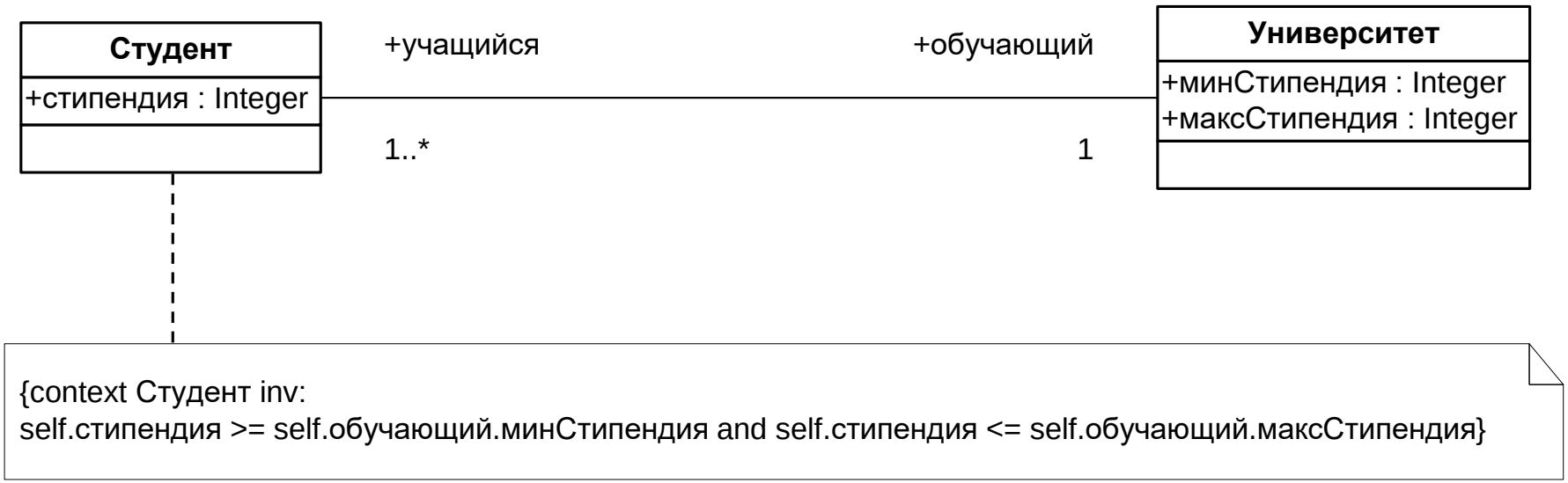
Примеры записи ограничений OCL

В компании должен быть акционер, обладающий контрольным пакетом акций



Примеры записи ограничений OCL

Размер стипендии студента должен находиться в диапазоне, допустимом в университете



Плюсы и минусы использования OCL при проектировании реляционных БД

Плюс:

1. OCL позволяет формально и однозначно определять ограничения целостности данных в терминах концептуальной модели БД

Минусы:

1. Строгость синтаксиса, неочевидность ряда конструкций и текстовая нотация языка OCL противоречат наглядности и интуитивной ясности диаграмм UML
2. OCL не поддерживается большинством CASE-систем, скорее всего придется преобразовывать инварианты OCL в ограничения целостности SQL вручную

Краткое сравнение ER-диаграмм и диаграмм классов UML

- ER-модель концептуально проще UML, в ней меньше понятий, терминов, вариантов применения.
- Поскольку UML может использоваться для унифицированного объектно-ориентированного моделирования всего чего угодно, в этом языке содержится масса различных понятий, терминов и вариантов использования, избыточных с точки зрения проектирования реляционных БД.
- В контексте проектирования реляционных БД структурные методы проектирования, основанные на использовании ER-диаграмм, и объектно-ориентированные методы, основанные на использовании языка UML, различаются, главным образом, лишь терминологией.
- Как правило, выбор средств для информационного моделирования – это вопрос вкуса и сложившихся обстоятельств.
- Неоспоримым преимуществом использования диаграмм классов UML при проектировании реляционных БД является то, что они позволяют оставаться в общем контексте UML и применять другие виды диаграмм для проектирования приложений БД.