Кратко

1. Понятие файловой системы обработки данных (ФСОД). Организация обработки данных в ФСОД. Недостатки ФСОД.

ФСОД — отдельная автономная программа; обрабатывает и получает доступ к данным; поддерживает одну функцию; всё посредством ФС ОС.

ООД — автономный отдел, который содержит рабочую группу; изучает потребности в автоматизации и обработке данных; создаёт прикладные программы (ПП), решающие проблемы отделов предприятия.

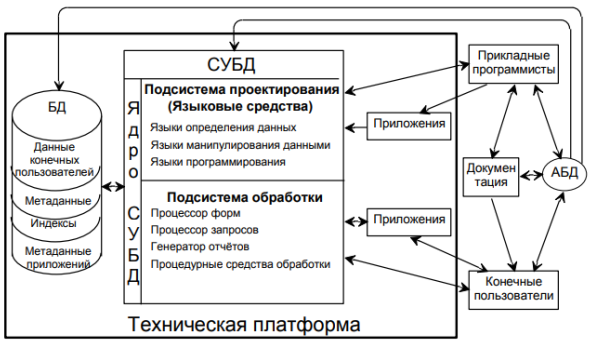
Организация обработки данных: программа содержит определения своих файлов, сканирует файлы для извлечения записей, использует методы доступа конкретной ОС.

Виды организации ФСОД:
Данные вокруг программ — программы независимо ведут свои файлы; форматы определяются автором программы.
Программы вокруг данных — форматы согласовываются между авторами ПП; создаётся общее пространство данных; отдельная ПП обрабатывает только часть общих данных; проблема связана с параллельным доступом к данным и разграничением доступа.

Недостатки ФСОД:
Данные вокруг программ: Неконтролируемая избыточность данных (возможная противоречивость) — одна ПП обновила свои файлы, но они дублируются у другой ПП (несогласованность). Решение: программы вокруг данных
Программы вокруг данных: Зависимость ПП от физических форматов данных — изменение форматов файлов ведёт к перекомпиляции всех ПП, даже если этим ПП не нужны атрибуты изменённого формата.
Разделение и изоляция данных — для формирования отчётов ПП необходима поддержка синхронной обработки файлов, чего ФСОД не поддерживает (нет сведений о структурах записей).
Невозможность обработки произвольных запросов — для каждого запроса нужно создавать отдельную ПП.

2. Система баз данных: компоненты, категории пользователей, компоненты приложений.

Компоненты СБД:



Ядро — взаимодействие между БД и подсистемами проектирования (операторы ЯОД, ядро

преобразует запросы ЯМД в последовательность команд ОС);
Аппаратура БД (АБД) — тех. средства для работы ПП;
Документация — инструкции и правила для проектирования и эксплуатации БД;
Конечные пользователи.

Категории пользователей:
Конечные пользователи (КП) — работники предприятия; взаимодействуют только с частью БД для выполнения его функций;
Прикладные программисты — создают ПП, интерфейс КП; не взаимодействуют с данными;
Администратор БД — проектировщик системы; сопровождает и несёт ответственность за функ. системы.

Компоненты приложений (программные средства для поддержки служебных функций КП):
Запрос — требование на выборку/обновление из БД подмножества данных по условию на ЯМД с последующим использованием ПП;
Форма — способ отображения/ввода/редактирования данных КП;
Отчёт — результат обработки запросов в виде документа;
Меню — средства доступа к функциям приложения;
ПП — программа, созданная для КП на ЯП, дополняющая функции СУБД (контроль email, тел.)

3. База данных и СУБД. Понятие БД. Структурные единицы БД. Ключи. Метаданные. Индексы. Назначение СУБД. Основные подсистемы СУБД.

БД — это самодокументированная (вместе с данными в БД есть описание её структуры) интегрированная (вместе с записями есть сведения о связях записей) совокупность записей.

Структурные единицы БД:
Поле — неделимый элемент данных в логической структуре, соответствующий атрибуту. Характеризуется именем, типом данных, длиной, точностью. Типы полей: семантически значимые (отображаемые), служебные (системные).
Запись — совокупность связанных полей, структурный тип данных. Состоит из имени записи и описания полей.
Экземпляр записи — реализация записи.
Таблица — совокупность экземпляров записи.

Ключ — поле (группа полей) для идентификации и быстрого поиска экземпляров записи**:**
Первичный ключ (PK) — уникальный идентификатор, значение которого не может быть одинаковым у различных экземпляров записи;
Вторичный ключ (SK) — поисковое поле, которому соответствует несколько экземпляров записи.
Внешний ключ (FK) — SK, являющейся ссылкой на существующий PK другой таблицы.

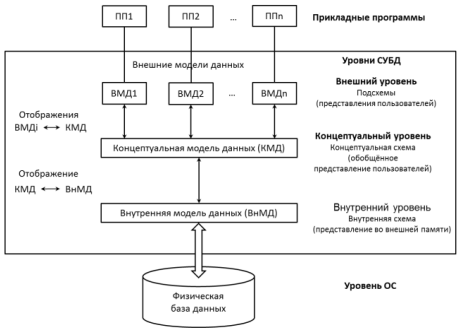
Метаданные — информация об объектах БД, хранимая в системных таблицах из системного каталога.
Содержит: описание полей/типов записей, сведения об отношениях таблиц, индексы, процедуры, триггеры и т.д.

Индекс — избыточные данные для производительности БД в виде служебной структуры, сопоставленной базовой таблице (с привязкой к физическим указателям) для доступа к записям по ключу; упорядочения записей. Виды: уникальные (избежание дублирования), не уникальные (быстрая сортировка и фильтрование); простые, составные (для базовой таблицы; для поиска и сортировки).

Назначение СУБД
СУБД — комплекс программ, позволяющих создать БД и манипулировать данными (вставлять, обновлять, удалять и выбирать). Система обеспечивает безопасность, надёжность хранения и целостность данных, а также предоставляет средства для администрирования БД.

Подсистемы СУБД:
Подсистема проектирования — для создания БД, интерфейсов категорий пользователей и проектирования приложений. Содержит: ЯОД, ЯМД, ЯП;
Подсистема обработки — обработка компонентов приложений. Содержит: процессор форм, запросов, генератор отчётов, процедурные средства обработки;
Ядро СУБД — взаимодействие между БД и подсистемами проектирования (операторы ЯОД, ядро преобразует запросы ЯМД в последовательность команд ОС).

4. Архитектура ANSI/SPARC. Уровни представления данных. Отображения. Независимость от данных. Системный каталог СУБД (назначение, сохраняемые данные, владелец, пользователи).



Архитектура ANSI/SPARC (American National Standard Institute - Национальный институт стандартизации / Standards Planning and Requirement Commitee - Комитет планирования стандартов и норм)
— архитектура СБД, состоящая из трёх уровней представления данных.

Уровни представления данных:
Концептуальный уровень — множество экземпляров концептуальных записей, соответствующих логическому представлению, которые физически не существуют, но воспроизводятся СУБД из экземпляров внутренних записей. Состоит из определений: типов записей, связей, ограничений целостности. Определяется на ЯОД и хранится в метаданных.
Внутренний уровень — описывает организацию данных на физическом уровне. Состоит из определений: типов внутренних записей, схем, файлов внешней памяти, индексов, служебных полей. Определяется на входном ЯВУ (язык высокого уровня) СУБД.
Внешний уровень — описывает внешнее представление данных конечного пользователя. Внешняя схема — подмножество концептуальной схемы. Содержит определения интерфейсов: форм, отчётов, запросов.

Отображения — взаимно-однозначное соответствие между представлениями данных соседних уровней:
концептуальный ↔ внутренний — соответствие концептуальным полям ↔ хранимых записей. Изменение внутреннего уровня не меняет концептуальную схему.
внешний ↔ концептуальный.

Независимость от данных — никакие изменения на нижних уровнях не меняют на верхние уровни представления данных.
Типы независимости:
Логическая независимость — изменение концептуальной схемы не меняет внешнюю схему.
Физическая независимость — изменение внутренней схемы не меняет концептуальную схему.

Системный каталог СУБД
Назначение — хранит определение данных всех уровней и является внутренней БД СУБД для согласованной работы всех её компонентов.
Сохраняемые данные — определение типов записей всех уровней, описание отображений, сведения о пользователях и их привилегиях, о программных и аппаратных средствах.

5. Организация доступа приложений к данным. Типовые операции обработки данных. Организация работы с внешней памятью. Обеспечение целостности данных.

Организация доступа приложений к данным
Приложение направляет СУБД запрос на обработку данных (в терминах внешней модели). Производится отображение внешний ↔ концептуальный ↔ внутренний. СУБД (средствами ОС) извлекает нужные физические записи из физической БД, помещает их в свой рабочий буфер, выполняет в буфере требуемую обработку, помещает в системные буферы СУБД. СУБД преобразует по схеме внутренний ↔ концептуальный ↔ внешний и посылает в рабочую область ПП, передаёт сообщение об исполнении запроса.

Типовые операции обработки данных
RETRIEVE — извлечение записей — считывание в рабочий буфер записей ФБД и формирование внешних записей;
INSERT — добавление записей — создание в рабочем буфере новых записей;
UNDATE — изменение значений — замена существующих значений полей извлечённых записей новыми значениями.
DELETE — удаление записей — уничтожение в рабочем буфере извлечённых записей.
SUMMARIZE — группировка записей по указанным признакам и агрегирование.
ROLLBACK — отмена выполненных обновлений.

Организация работы с внешней памятью
СУБД принимает запрос приложения и проверяет, есть ли записи для его исполнения. Если записи есть, то они отправляются приложению. Иначе: система проверяет наличие места для размещения физических записей. Если места достаточно, нужные записи извлекаются из ФБД и запрос исполняется. Иначе: рабочий буфер частично очищается, удаляются редко используемые данные, нужные записи извлекаются из ФБД и запрос исполняется.

Обеспечение целостности данных
Бизнес-правила (ограничивают допустимые значения элементов данных и свойства отношений) определяют ограничения целостности данных.
Обеспечение: триггерами БД.
СУБД проверяет запросы и при нарушении ОЦ: сообщает пользователю об ошибке и ждёт реакции или автоматически устраняет разгласованность данных. СУБД не гарантирует истинность данных.

6. Управление доступом к данным. Принципы ограничения доступа. Авторизация пользователей. Привилегии пользователей.

Управление доступом к данным
Аутентификация — связь идентификатора пользователя с паролем, который сохраняется в зашифрованном виде в системном каталоге, используется при подключении к системе и обладает ограничениями (минимальная длина, состав символов)

Принципы ограничения доступа
Служащий имеет право доступа и манипуляций только с теми данными, которые обусловлены его служебными обязанностями.

Авторизация пользователей
Реализует принципы ограничений доступа и обеспечивает предоставление прав доступа к системе и её объектам.
Пользователь (зарегистрированный в системном каталоге идентификатор авторизации с типом пользователя и описанием; оформлен в учётной записи) выполняет определённое действие (запускать приложение, просматривать таблицу, вставлять строки) с объектами БД, имея на это привилегию (право пользователя).

Привилегии пользователей
Сохраняются в системном каталоге в виде матрицы управления доступом
Системная привилегия — право создания и модификации объектов БД (схем, таблиц, приложений, правил)
Объектная привилегия — право использования объекта в операциях определённого типа (просмотр таблицы/столбцов/подмножества строк, вставка строк, запуск приложений)
retrieve(0001), update(0010), insert(0100), delete(1000), all(1111).
Код набора привилегий — поразрядная сумма типов привилегий (0101)

7. Транзакции в БД: понятие транзакции, свойства транзакции.

Понятие транзакции
Транзакция — последовательность операций над БД в виде логически неделимой единицы управления и манипулирования над БД, которая начинается с согласованного начального состояния и завершается согласованным конечным состоянием. Обеспечивает целостность БД.
Может завершиться в зависимости от удовлетворения всех ОЦ:
фиксацией (COMMIT TRANSACTION) — при удовлетворении всех ОЦ;
откатом (ROLLBACK TRANSACTION) — отмена всех проведённых транзакций, восстановление буфера на момент согласованного начального состояния.

Свойства транзакции (АСИД)
Атомарность — транзакция есть неделимая единица работы БД;
Согласованность — начальное и конечное состояние БД согласованны;
Изолированность — гарантия исполнения транзакции при параллельном исполнении;
Долговечность — фиксация транзакции гарантирует сохранение состояния БД.

8. Виды конфликтов параллельного доступа транзакций к данным: потеря обновлений, «грязные» чтения, несогласованные изменения.

Потеря обновлений
При параллельном обновлении одного и того же объекта двумя транзакциями.
Решение: до завершения транзакции, изменяющей объект, никакая другая транзакция не должна изменять этот объект.

«Грязные» чтения
При доступе транзакцией к промежуточным результатам другой транзакции

Решение: чтения должны быть заблокированы, пока транзакция, изменяющая некоторый объект, не освободит этот объект для другой транзакции.

Несогласованные изменения
Когда одна из двух параллельно исполняемых транзакций обновляет состояние БД, а другая извлекает данные, и выполняет какие-либо преобразования.
Решение: транзакция невозможна, если до завершения транзакции, читающей некоторый объект, никакая другая транзакция не сможет изменять его.

9. Управление параллелизмом. Конфликты параллельного доступа. Принцип изолированности транзакций. Двухфазный протокол синхронизационных блокировок.

Принцип изолированности транзакций
Из свойств АСИД: Изолированность транзакций — режим обработки транзакций, при котором результат их параллельного исполнения совпадает с результатов последовательного исполнения в каком-либо порядке — никакая из транзакция не влияет на результаты работы других транзакций.

Реализация

Двухфазный протокол синхронизационных блокировок
Принцип: прежде чем выполнять какую-либо операцию над объектов A, транзакция T должна запросить блокировку (захват) объекта A (который запрашивается неявными запросами блокировок по типу операции).
Режимы захвата:
S (Shared lock) — разделяемый захват (чтение). Объект A может захватить другая транзакция в режиме S.
X (eXclisive lock) — монопольный захват (добавление, удаление, модификация). Объект A не может захватить никто.
При запросе к заблокированному объекту, транзакция ожидает завершения заблокировавшей её транзакции (оператором COMMIT или ROLLBACK).
Фазы выполнения транзакции:
Фаза наложения блокировок — накопление захватов;
Фаза снятия блокировок — освобождение всех захваченных объектов.

10. Восстановление базы данных. Виды аварийных ситуаций. Системный журнал СУБД (назначение, сохраняемые данные, протокол WAL). Индивидуальный откат транзакции. Восстановление после мягкого сбоя. Восстановление после жёсткого сбоя.

Виды аварийных ситуаций
Локальный сбой — аварийное прекращение одной транзакции. Причины: попытка деления на ноль, нарушение ограничений целостности, явное завершение транзакции оператором ROLLBACK, временная блокировка транзакций.
Мягкий сбой — аварийное отключение питания, неустранимый сбой процессора. Теряется содержимое оперативной памяти, аварийно прерываются все существующие транзакции, могут оказаться незафиксированными в ФБД результаты транзакций, завершившихся оператором COMMIT.
Жёсткий сбой — физическое разрушение базы данных, последствие которой катастрофические для организации-владельца данных.

Системный журнал СУБД

Назначение: хранит детальные сведения обо операциях каждой транзакции.

Сохраняемые данные:

Для каждой транзакции: идентификатор транзакции, идентификатор пользователя, время начала транзакции, время и способ завершения транзакции;

Для каждой операции: идентификатор транзакции, в которой выполнялась операция, время начала операции, идентификатор обрабатываемого элемента данных, тип операции, копия элемента данных до операции (обновления или удаления), копия элемента данных после операции (обновления или вставки);

(COMMIT или ROLLBACK): запись об окончании транзакции

Протокол WAL

Write Ahead Log — протокол предварительной записи в журнал — обновления, выполненные транзакцией, сначала попадают в СЖ, и только потом во внешнюю память.

Программы формируют записи файлов с рабочих буферах → в буфере СЖ размещаются записи о транзакциях, которые принудительно выталкиваются во внешнюю память при завершении каждой транзакции → транзакция считается завершённой.

Индивидуальный откат транзакции

Также является транзакцией (записывается в СЖ) на случай мягкого сбоя в процессе отката.

ROLLBACK или локальный сбой: создание обратного хронологического списка записей из СЖ данной транзакции → последовательный просмотр записей и выполнение противоположных по смыслу операций. Начальное состояние процедуры — состояние буферов БД в момент прекращения транзакции.

Восстановление после мягкого сбоя

Создаётся 2 списка транзакций: отменяемые (UNDO, включаются все транзакции, указанные в записи контрольной точки (КТ)) и исполняемые (REDO, остаётся пустым).

Анализируется журнал регистрации, начиная с КТ:

записи о начале транзакции Т добавляются в список UNDO,

записи о завершении транзакции (COMMIT) добавляются в список REDO.

Из списка UNDO исключаются транзакции, попавшие в список REDO.

СЖ просматривается от конца до записи контрольной точки:

отменяются транзакции из списка UNDO,

выполняются повторно транзакции из списка REDO.

Восстановление после жёсткого сбоя

В СЖ устанавливается “жёлтая зона” → создание резервной копии при переполнении СЖ → система дожидается окончания всех транзакций → рабочие буферы СЖ и БД выталкиваются во внешнюю память → созданное состояние ФБД копируется на резервный носитель, файл журнала очищается.

Восстановление после жёсткого сбоя — восстановление ФБД на момент последнего создания резервной копии → по журналу регистрации повторно исполняются все транзакции до момента сбоя.

11. Функции СУБД (восемь сервисов Кодда)

СУБД должна:

1. Предоставлять пользователям возможность сохранять, извлекать и обновлять данные в БД (сокрытие физической реализации)

2. Поддерживать доступный пользователям каталог с описанием элементов и структур данных: определение смысла элементов и структур данных, обнаружение избыточности и противоречивости описания элементов, протоколы всех уровней, поддержка целостности, сведения о владельцах и привилегии, просмотр изменений до их применения.
3. Поддерживать атомарность — выполнение либо всех транзакций, либо ни одной их них
4. Поддерживать изолированность — обновление БД при параллельном выполнении операций многими пользователями
5. Поддерживать долговечность — средства восстановления БД в случае повреждения или разрушения
6. Обеспечивать безопасность — механизм санкционированного доступа пользователей
7. Поддерживать интеграцию с менеджерами обмена данными — для получения сообщений от приложений на клиентском узле через сеть с серверами БД
8. Предоставлять набор вспомогательных утилит для функций администратора БД.

12. Назначение и составные части реляционной модели данных (РМД)

РМД — набор понятий и языковых конструкций, предназначенных для описания структур данных, ограничений целостности данных и операций манипулирования данными на логическом уровне.

Составные части:
Структурная часть — определяет набор абстрактных языковых конструкций (ЯОД) для описания отношений РБД на логическом уровне
Целостная часть — определяет правила ограничений целостности для обеспечения непротиворечивости и правдоподобия данных
Манипуляционная часть — определяет набор непроцедурных операций над отношениями и набор языковых средств (ЯМД) для точного формулирования требований к изъятию необходимых данных (выборки/обновления)

13. Структурные понятия реляционной модели данных: домен, атрибут, схема отношения, кортеж, отношение. Свойства отношений.

Домен — подмножество элементов типа данных — пара (тип данных, предикат). На одном и том же домене можно определить произвольное число атрибутов.

Атрибут — имя, поставленное в соответствие домену и наследующее от домена его свойства.

Схема отношения (R·) — множество пар (домен, атрибут) (заголовок таблицы) $R = (D_1, A_1), (D_2, A_2), \ldots, (D_n, A_n)$

Кортеж (экземпляр *сущности*; **запись**) — множество пар S_R, соответствующих схеме (R·) (строка таблицы с заданным заголовком):
$S_R = (A_1, a_1), (A_2, a_2), \ldots, (A_n, a_n),\ a_i \in D_i,\ \ i = 1, \ldots, n$

Отношение (R) (единственный структурированный тип данных) — множество кортежей S_R, соответствующих одной и той же схеме (R·).
(в РМД) — набор записей (экземпляров сущности) обо всех существующих экземплярах соответствующего объекта ПО.

Экземпляр (значение) отношения — это набор кортежей с заданной схемой, существующий в некоторый фиксированный момент времени.
Сущность — потенциальное множество кортежей одной схемы.

Характеризуется
— арностью (степенью) (*фиксирована*) — числом пар (домен, атрибут) в схеме — (характеристика типа)
Тип — задаётся схемой отношения. Все кортежи — значения типа — удовлетворяют одной схеме.
— мощностью (может изменяться со временем) — числом кортежей, составляющих тело отношения (характеристика экземпляра отношения)

Свойства отношения:
Атомарность значений атрибутов — 1НФ — схема отношения включает атрибуты, определённые на домене, являющемся подмножеством простого типа данных.
Неупорядоченность атрибутов — порядок следования атрибутов несущественен.
Уникальность атрибутов — одноимённые атрибуты недопустимы.
Уникальность и неупорядоченность кортежей — кортеж можно найти по его ключу, отношение — множество кортежей, порядок следования кортежей несущественен.
Изменяемость отношений — набор кортежей (тело отношений) может меняться со временем (значения атрибутов кортежей, добавление/удаление кортежей)

14. Понятие целостности данных. Внутренние ограничения целостности РМД. Требования целостности домена и атрибута. NULL-значения и целостность атрибута.

Внутренние ограничения целостности РМД
Набор средств описания бизнес-правил ПО.

Виды внутренних ОЦ РМД:
Домена — прямым перечислением всех его значений; указанием предиката, определяющего условие принадлежности значения домену.
Требование целостности домена: должны быть явно определены все необходимые домены.
Атрибута — требование целостности атрибута — значения атрибута должны выбираться только из его домена.
Сущности — для каждого базового отношения должен быть объявлен первичный ключ.
(в БД не должно быть неидентифицируемых кортежей)
Ссылочные — БД не должна содержать значений FK, не совпадающих с каким-либо значением PK в существующих кортежах родительского отношения.

NULL-значения и целостность атрибута
Для каждого атрибута каждого отношения следует указывать допустимость/недопустимость NULL-значения
Атрибут-первичный ключ не может быть NULL-значением.

15. Определение потенциального ключа отношения. Понятие первичного ключа и требование целостности сущности. NULL-значения атрибутов и целостность сущности. Практический смысл требования целостности сущности.

Определение потенциального ключа отношения
— это подмножество атрибутов схемы, значения которого не повторяются в различных кортежах
— это
уникальнозначное (в любой момент времени в текущем значении R не существует двух кортежей с одинаковым значением K)
неизбыточное (никакое подмножество $L \in K$ не обладает свойством уникольнозначности — 2НФ) подмножество атрибутов его схемы.

Понятие первичного ключа и требование целостности сущности
Первичный ключ отношения — выделенный из практических соображений *потенциальный ключ отношения*, являющийся уникальным идентификатором кортежей отношения.
Требование целостности сущности — для каждого базового отношения должен быть объявлен первичный ключ отношения.

NULL-значения атрибутов и целостность сущности
Если в состав возможного ключа входит атрибут, который может принимать NULL-значение, то этот ключ нельзя использовать для идентификации кортежей, иначе нарушится целостность сущности.
Требование целостности сущности — для каждого базового отношения должен быть определён первичный ключ, ни один компонент которого не может приниматься NULL-значения.

Практический смысл требования целостности сущности
в БД не должно быть неидентифицируемых кортежей, т.к. отсутствие первичного ключа приводит к информационному мусору и нарушениям ограничений целостности.

16. Определение внешнего ключа. Роль внешних ключей в реляционной базе данных. Типы связей отношений, поддерживаемые реляционной моделью данных. Требование ссылочной целостности. NULL-значения атрибутов и ссылочная целостность. Типовые правила ссылочной целостности.

Определение внешнего ключа
Внешний ключ — подмножество атрибутов отношения-потомка (R1), эквивалентное первичному ключу родительского отношения (R2). Значение внешнего ключа — ссылка на кортеж родительского отношения.

Роль внешних ключей в реляционной базе данных
РБД не содержит какой-либо специальной структуры для представления связей, поэтому связь трактуется моделью как ассоциация кортежей, т.е. средствами РМД (внешними ключами) можно описать только ассоциативные отношения сущностей.

Типы связей отношений, поддерживаемые реляционной моделью данных
Механизм внешнего ключа поддерживает только
бинарные ассоциации $1:1$ и $1:M$
Бинарные связи типа $M:N$ и ассоциации высших степеней представляются отношениями-ассоциациями, которые содержат внешние ключи, эквивалентные первичным ключам ассоциируемых отношений.

Требование ссылочной целостности
БД не должна содержать значений FK, не совпадающих с каким-либо значением PK в существующих кортежах родительского отношения.

NULL-значения атрибутов и ссылочная целостность
БД не должна содержать NULL-значений внешнего ключа, не существующих среди значений первичного ключа родительского отношения.

Типовые правила ссылочной целостности
Правила внешних ключей —- определяют реакцию РСУБД на появление значений внешнего ключа, которых нет среди существующих значений родительского ключа
—— добавление кортежа в отношение-потомок — запрет на добавление в отношение-потомок кортежа, ссылающегося на несуществующий кортеж отношения-родителя.
—— удаление кортежа отношения-родителя:

— RESTRICTED — ограниченное удаление — отложить удаление родительского кортежа, если существует хотя бы один кортеж потомка, содержащий ссылку на этот родительский кортеж.
— CASCADE — каскадировать — распространить операцию удаления на все кортежи потомка, ссылающихся на этот кортеж-родителя.
—— обновление значения первичного ключа в родительском кортеже — аналогично случаю удаления кортежа родителя.

17. Реляционный язык определения данных РМД

Язык определения данных (ЯОД) объявляет все компоненты схемы: определения доменов, отношений, первичных и альтернативных ключей, внешних ключей и правил ссылочной целостности. Определения объектов сохраняются ***в системном каталоге*** и используются РСУБД в процессах обработки данных.

Объявление домена
```
CREATE DOMAIN имя_домена тип [(длина)]
   { VALUES (список) }      |
   { FOR ALL VALUE (предикат) };
```

Удаление домена
```
DESTROY DOMAIN имя_домена;
```

Объявление базового отношения
```
CREATE BASE RELATION имя_отношения
(
   список_определений_атрибутов
   список_определений_возможных_ключей
   список_определения_внешних_ключей
);
```

Объявление атрибута
```
строка_определения_атрибута ::= имя_атрибута DOMAIN имя_домена
```

Объявление возможного ключа
```
строка_определения_возможного_ключа ::=
   PRIMARY KEY (список_атрибутов) |
   CANDIDATE KET (список_атрибутов)
```

Объявление внешнего ключа
```
FOREIGN KEY (список_атрибутов)
   REFERENCES имя_отношения
      ON DELETE правило_удаления
      ON UPDATE правило_обновления
```

Удаление базового отношения
```
DESTROY BASE RELATION имя_отношения;
```

18. Определения теоретико-множественных операций реляционной алгебры: пересечение, объединение, разность, расширенное прямое произведение

Реляционная алгебра (РА) (выражение) — это специальная разновидность алгебры множеств, которая представляет собой набор операций над отношениями и правил комбинирования

операций в выражениях — описывает процедуру вычисления значения производного отношения по заданным значениям отношений-операндов.

X, Y, … — множество атрибутов

R(X) — схема отношения

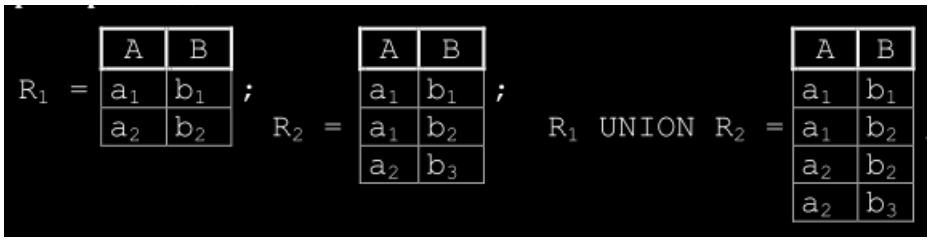
R — отношение

X:x — кортеж отношения - набор значений атрибутов из X

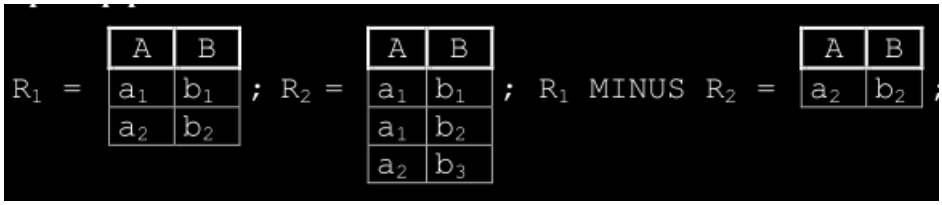
O — пустое множество

Группы операций РА:

- Специальные реляционные операции: селекция (ограничение, выборка), проекция, соединение по условию, естесственное соединение, взятие реляционного частного (реляционное деление), переименование
- Теоретико-множественные операции:
  - Операндами могут быть только отношения с эквивалентными множествами:
    - объединение — $UNION$
      — объединение отношений со схемами
      $R_1(X)$ и $R_2(X) \Rightarrow \quad R = R_1\ UNION\ R_2$,
      имеющее схему $R(X)$, эквивалентную схемам R_1, R_2,
      и тело, составленное из *всех* кортежей $X : x$,
      принадлежащих хотя бы одному из операндов

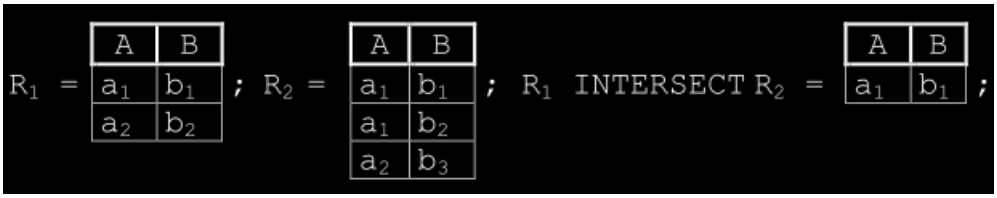


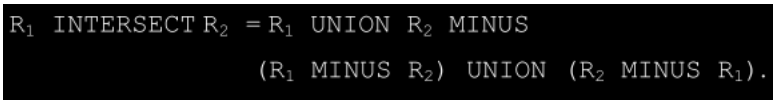
    - вычитание — MINUS — разность отношений со схемами
      $R_1(X)$ и $R_2(X) \Rightarrow \quad R = R_1\ MINUS\ R_2$,
      имеющее схему $R(X)$, эквивалентную схемам R_1, R_2,
      и тело, составленное *только из тех* кортежей $X : x \in R_1$,
      которые ***не принадлежат*** R_2



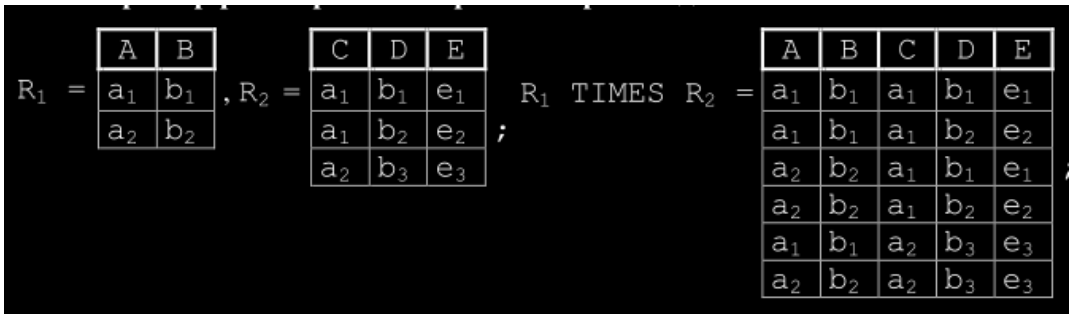
    - пересечение — INTERSECT —
      — пересечением отношений со схемами
      $R_1(X)$ и $R_2(X) \Rightarrow \quad R = R_1\ INTERSECT\ R_2$,
      имеющее схему $R(X)$, эквивалентную схемам R_1, R_2,

и тело, составленное *только из тех* кортежей $X : x \in R_1$,
которые ***принадлежат*** R_2





- Операндами могут быть только отношения с непересекающимися схемами:
  - расширенное прямое произведение — TIMES — отношений со схемами $R_1(X)$ и $R_2(X) \Rightarrow \quad R = R_1\ TIMES\ R_2$,
    имеющее схему $R(X, Y)$ и тело, образованное кортежами $\{X : x, Y : y\}$, такими, что в R_1 существует кортеж $X : x$ и в R_2 существует кортеж $Y : y$.



19. Определение реляционного исчисления с переменными-кортежами.

Реляционное исчисление (РИ) (формула) — это разновидность исчисления предикатов первого порядка, которое представляет набор формальных правил записей условий, которым удовлетворяют кортежи производного отношения. Условия записываются в виде предиката, содержащего атрибуты отношений-источников данных.

20. Понятие жизненного цикла системы с базами данных. Краткая характеристика этапов.

Жизненный цикл СБД — непрерывный процесс, который начинается в момент принятия решения о создании СБД и заканчивается в момент полного изъятия системы из эксплуатации.

Этапы и краткая характеристика:

Планирование разработки — оцениваются требуемый объём работ, требуемые ресурсы, общая стоимость проекта.

Определение требований к системе — определяются бизнес-процессы (диапазон действия системы), её границы и состав пользователей

Анализ требований пользователей — собирается и анализируется информация о деятельности организации, выявляются функции пользователя системы, данные, необходимые для выполнения функций, бизнес-правила, ограничивающие функции

Проектирование базы данных — создание информационного ядра системы: определение данных и связей между ними, создание модели данных и предварительного варианта проекта СБД.

Проектирование приложений — создание интерфейса конечного пользователя для всех функций и прикладных программ, предназначенных для обработки данных

Реализация — создание пустой базы данных (ЯОД), определение всех специфических пользовательских представлений, реализация всех приложений (ЯВУ), средств защиты данных, поддержания целостности и транзакций обработки БД (ЯМД).

Первоначальная загрузка данных — перенос любых существующих данных в новую БД и модификация всех существующих приложений с целью обеспечения их совместной работы с новой БД.

Тестирование — процесс выполнения прикладных программ с целью поиска ошибок

Эксплуатация и сопровождение — это наблюдение за системой и поддержание её нормального функционирования по окончании развёртывания. Осуществляется контроль производительности системы, сопровождение и модерация приложений.

21. Модель «сущность-связь». Назначение, основные понятия, нотации. Пример диаграммы с интерпретацией.

Модель «сущность-связь»

Модель “Сущность-связь” — ER-модель — Entity-Relationship Model — (Питер Ченн, 1976 г.) — это графический язык для описания объектов и отношений объектов, где спецификации требований пользователя представляются в виде диаграммы, показывающей объекты ПО, их связи и свойства объектов и связей — ER-диаграммы.

Назначение

ER-диаграмма отражает представления автора о требованиях пользователя к данным

Основные понятия

Сущность (entity) — это некоторый объект, выделяемый (идентифицируемый) пользователем в предметной области, имеющее реальное (физическое) или концептуальное существование и выделяемое пользователем в окружающем мире.
Классы сущностей — сущности одного и того же типа — абстракция, понятие, выделяемое пользователем, которому сопоставляется некоторый символ — имя сущности.
Атрибут — это характеристика сущности или связи (свойство класса), значимая с точки зрения пользователя.
Идентификатор — это атрибут сущности, значение которого можно использовать для идентификации или именования экземпляра.
Связь — это отношение сущностей.
Класс связи — осмысленная с точки зрения пользователя ассоциация классов сущностей. Смысл ассоциации передаётся глагольным оборотом, связывающим имена классов сущностей.

Нотации

Класс сущности — прямоугольник, в котором пишут имя сущности прописными буквами.
Слабая сущность — прямоугольник со скруглёнными углами (или двойными линиями) — сущность, логически зависимая от какой-либо другой сущности (или сущностей).
Класс связи — ромб.
Связь, от которой зависит существование слабой сущности — ромб со скруглёнными углами (или двойными линиями).
Участвующие в связи сущности — дуги, соединяемые с ромбом.

Смысл ассоциации — глагольный оборот около дуги связи, передающий смысл ассоциации.
Атрибут — именованный эллипс, соединённый дугой с прямоугольником сущности. контур эллипса:
— простой атрибут — сплошной,
— производный атрибут — штриховой,
— многозначный атрибут — двойной.

Пример диаграммы с интерпретацией

Существует преподаватель, ассоциированный с одной или несколькими дисциплинами, и, соответственно, дисциплина, ассоциированная с одним или несколькими преподавателями

22. Модель «сущность-связь. Понятие связи. Типы связей. Свойства связей: степень (арность), мощность, обязательность/необязательность. Приведение связи высшей арности к совокупности бинарных связей.

Модель «сущность-связь»
Модель “Сущность-связь” — ER-модель — Entity-Relationship Model — (Питер Ченн, 1976 г.) — это графический язык для описания объектов и отношений объектов, где спецификации требований пользователя представляются в виде диаграммы, показывающей объекты ПО, их связи и свойства объектов и связей — ER-диаграммы.

Понятие связи
Связь — это отношение сущностей.
Класс связи — осмысленная с точки зрения пользователя ассоциация классов сущностей. Смысл ассоциации передаётся глагольным оборотом, связывающим имена классов сущностей.

Типы связей
— 1:1 — один экземпляр E1 может участвовать не более чем в одном экземпляре связи и один экземпляр E2 может участвовать не более чем в одном экземпляре связи.

— 1:N — экземпляр E1 может участвовать в нескольких (N) экземплярах связи, а экземпляр E2 - не более чем в одном.
— M:N — экземпляр E1 может участвовать в N экземплярах связи, а экземпляр E2 - в M экземплярах.

Свойства связей
Степень (арность) — число классов сущностей, участвующих в связи: унарная, бинарная, тернарная и т.д.
Мощность — число экземпляров связи, в которых может участвовать один экземпляр сущности
Обязательность/необязательность — степень участия сущности в связи
обязательность — связь R, которая является обязательной со стороны сущности Ei, если каждый экземпляр Ei должен участвовать хотя бы в одном экземпляре связи

Приведение связи высшей арности к совокупности бинарных связей
Если каждый экземпляр новой сущности связан точно с одним экземпляром каждой сущности, вступающей в n-арную связь, то рассматриваемая n-арная связь эквивалентна n бинарным связям типа 1:M.

23. Жизненный цикл системы баз данных. Основные этапы. Виды работ на этапах.

Жизненный цикл СБД — непрерывный процесс, который начинается в момент принятия решения о создании СБД и заканчивается в момент полного изъятия системы из эксплуатации.

Этапы и виды работ на этапах:

Планирование разработки — оцениваются требуемый объём работ, требуемые ресурсы, общая стоимость проекта.

Определение требований к системе — определяются бизнес-процессы (диапазон действия системы), её границы и состав пользователей

Анализ требований пользователей — собирается и анализируется информация о деятельности организации, выявляются функции пользователя системы, данные, необходимые для выполнения функций, бизнес-правила, ограничивающие функции
Работы: наблюдение за деятельностью организации, изучение документов, анкетирование будущих пользователей. Итоги: входные и выходные документы и сообщения, сведения о транзакциях, список требований с приоритетами.

Проектирование базы данных — создание информационного ядра системы: определение данных и связей между ними, создание модели данных и предварительного варианта проекта СБД.

Проектирование приложений — создание интерфейса конечного пользователя для всех функций и прикладных программ, предназначенных для обработки данных
Работы: разработка прототипа, обладающего частью функционала для уточнения требований к системе. Развивается в готовый продукт.

Реализация — создание пустой базы данных (ЯОД), определение всех специфических пользовательских представлений, реализация всех приложений (ЯВУ), средств защиты данных, поддержания целостности и транзакций обработки БД (ЯМД).

Первоначальная загрузка данных — перенос любых существующих данных в новую БД и модификация всех существующих приложений с целью обеспечения их совместной работы с новой БД.

Тестирование — процесс выполнения прикладных программ с целью поиска ошибок

Эксплуатация и сопровождение — это наблюдение за системой и поддержание её нормального функционирования по окончании развёртывания. Осуществляется контроль производительности системы, сопровождение и модерация приложений.

24. Проектирование базы данных. Цели проектирования. Фазы (этапы) процесса проектирования. Виды работ на этапах.

Проектирование БД — это процесс создания БД, предназначенной для поддержки функционирования организации и способствующей достижению её целей.

Цели проектирования:
Создать структуры хранения, обеспечивающие накопление данных организации и выполнение необходимых видов обработки данных
Создать все приложения, обеспечивающие взаимодействие КП с базой данных и специфическую обработку в соответствие с требованиями.

Фазы процесса проектирования
Концептуальное моделирование — процесс анализа информационных потребностей КП.
Виды работ:
— изучение деятельности организации (документы, наблюдения за организацией, беседы с КП, анкетирование, опыт предыдущих разработок)
— анализ информации и выводы о структуре и связях объектов ПО в документированном виде
— обсуждение выводов с КП для согласования и выбора модели данных
— составление документированной концептуальной модели представлений КП об информационных потребностях бизнеса
Логическое моделирование — описание концептуальной схемы БД с учётом выбранного способа структурирования данных.
Виды работ:
— определение структуры целевой СУБД: реляционная, сетевая, объектно-ориентированная, иерархическая.
— описание структур данных в терминах выбранной модели данных: РМД
— описание ограничений целостности
Физическое моделирование — проектирование ФБД — создание внутренней схемы, определение структуры хранимых файлов и методов доступа.
Виды работ:
— выбор конкретной СУБД: РБД
— РБД: создание набора таблиц и ограничений в логической модели
— определение структур хранения данных и методов доступа к данным
— разработка средств защиты системы.

25. Определение функциональной зависимости атрибутов отношения. Основания для заключения о наличии ФЗ между атрибутами отношения. Способы объявления функциональной зависимости в реляционной модели данных.

Определение функциональной зависимости атрибутов отношения
A функционально определяет B — если атрибут A кортежа принял некоторое значение, то атрибут B кортежа автоматически принимает соответствующее значение
(пример, связка: К_ном → К_тел)

Основания для заключения о наличии ФЗ между атрибутами отношения
Взаимно однозначная ФЗ — Пусть A и B — подмножества атрибутов отношения. Говорят, что A и

B связаны взаимно-однозначной ФЗ A ↔ B, если и только если существует пара ФЗ A → B и B → A
Неприводимая ФЗ. Пусть A и B — подмножества атрибутов отношения. Говорят, что B неприводимо зависит от A, если и только если A → B и не существует такого C ⊂ A, что C → B
Транзитивная ФЗ. Пусть A, B и C — подмножества атрибутов отношения. Говорят, что C транзитивно зависит от A, если и только если существуют ФЗ A → B и B → C
Взаимная независимость атрибутов. Пусть A1, A2,..., An — атрибуты. Говорят, что эти атрибуты взаимно независимы, если и только если для любого значения i атрибут Ai не зависит функционально ни от какого подмножества остальных атрибутов

Способы объявления функциональной зависимости в реляционной модели данных
Механизм контроля уникальности значений — Нужно создать отношение, потенциальным ключом которого является детерминант ФЗ, а неключевым атрибутом — зависимая часть.
Множество хранимых атрибутов должно быть структурировано, т.е., представлено в виде схем нескольких отношений так, чтобы все ФЗ, заданные бизнес-правилами, могла поддерживать реляционная СУБД
Для этого отношения должны удовлетворять определённым требованиям. Эти требования ограничивают множество допустимых ФЗ в схеме отношения и называются **нормальными формами отношений**

26. Понятия взаимной независимости, транзитивной и неприводимой функциональной зависимости. Требования 1НФ, 2НФ, 3НФ, НФБК. Определение многозначной зависимости. Требование 4НФ.

Понятия взаимной независимости, транзитивной и неприводимой функциональной зависимости
Взаимная независимость атрибутов. Пусть A1, A2,..., An — атрибуты. Говорят, что эти атрибуты взаимно независимы, если и только если для любого значения i атрибут Ai не зависит функционально ни от какого подмножества остальных атрибутов
Транзитивная ФЗ. Пусть A, B и C — подмножества атрибутов отношения. Говорят, что C транзитивно зависит от A, если и только если существуют ФЗ A → B и B → C
Неприводимая ФЗ. Пусть A и B — подмножества атрибутов отношения. Говорят, что B неприводимо зависит от A, если и только если A → B и не существует такого C ⊂ A, что C → B

Требования 1НФ, 2НФ, 3НФ, НФБК
Первая нормальная форма — Говорят, что отношение находится в 1НФ, если и только если каждый его атрибут определён на домене простого типа данных
Вторая нормальная форма — Говорят, что отношение находится в 2НФ, если и только если оно находится в 1НФ и не содержит ФЗ от части первичного ключа. (выполняется неприводимость ФЗ
Третья нормальная форма — Говорят, что отношение находится в 3НФ, если и только если оно находится в 2НФ и не содержит транзитивных ФЗ от первичного ключа
Нормальная форма Бойса-Кодда. Отношение находится в НФБК, если и только если каждый его детерминант является возможным ключом.

Определение многозначной зависимости
Пусть A, B и C — произвольные подмножества атрибутов отношения R. Говорят, что A →→ B, если и только если для любой реализации R множество значений B, соответствующее заданной паре значений атрибутов A и C, зависит только от значения A и не зависит от значения C. Многозначная зависимость A →→ B имеет место, если и только если A →→ C.

Требование 4НФ
Говорят, что отношение находится в 4НФ, если и только если оно находится в НФБК и не содержит

многозначных зависимостей

27. Нормализация отношения. Необходимость нормализации. Правила нормализации. Критерии выбора проекций. Требования к нормализованной структуре РБД

Нормализация отношения

Нормализация — это процедура декомпозиции без потерь; процесс преобразования универсального отношения к системе отношений, не обладающей аномалиями обновления данных,

Необходимость нормализации

Средствами ядра РСУБД могут поддерживаться только те ФЗ, детерминантами которых являются возможные ключи

Многозначные зависимости не поддерживаются ни в каком виде, т.к. РМД не допускает многозначных атрибутов

Правила нормализации

1. Отношение в 1НФ следует разбить на проекции для исключения тех ФЗ от первичного ключа, которые не являются неприводимыми. Результатом будет набор отношений, находящихся, по крайней мере, в 2НФ.
2. Отношения в 2НФ следует разбить на проекции для исключения любых транзитивных зависимостей. Результатом будет набор отношений, находящихся, по крайней мере, в 3НФ.
3. Отношения в 3НФ следует разбить на проекции для исключения любых ФЗ, детерминанты которых не являются потенциальными ключами. Результатом декомпозиции будет набор отношений, находящихся, как минимум, в НФБК.
4. Отношение, находящееся в НФБК, следует разбить на проекции для исключения всех многозначных зависимостей. Результатом будет набор отношений в 4НФ.

Критерии выбора проекций

Критерий — каждое отношение в системе должно быть таким, чтобы любой его детерминант был потенциальным ключом

Декомпозиция без потерь — удовлетворяется, если и только если исходное отношение содержит хотя бы одну ФЗ (теорема Хеза).

Практическое правило декомпозиции без потерь: все атрибуты первичного ключа одной из проекций должны входить в состав атрибутов другой проекции.

Возможность независимого обновления проекций — удовлетворяется, если и только если все ФЗ, существующие в исходном отношении, являются логическими следствиями определений потенциальных ключей его проекций.

Требования к нормализованной структуре РБД

— Для любой реализации исходного отношения R естественное соединение проекций должно быть эквивалентно R

(декомпозиция без потерь). (иначе естественное соединение проекций будет содержать кортежи, которых не было в исходном отношении)

— Все межатрибутные зависимости, существующие в исходном отношении, должны быть логическими следствиями определений потенциальных ключей проекций. (иначе проекции не смогут обновляться независимо)

Кому достались последние билеты — помянем…

28. Модель IDEF1X. Назначение, основные понятия, нотации. Уровни модели. Примеры диаграмм различных уровней с интерпретацией

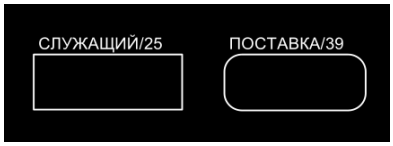
IDEX1x — Integrated DEFinition eXpanded — федеральный стандарт США проектирования реляционной модели данных; язык описания данных, подразумевающий
***Модель данных** — графическое и текстуальное представление, идентифицирующее потребности организации в данных для достижения ее целей, выполнения функций и определения стратегий управления. Модель данных идентифицирует сущности, домены, атрибуты и связи между данными и поддерживает единый концептуальный взгляд на данные и их взаимосвязи.*

Компоненты:

- **графический** — средства создания диаграмм, показывающих структуру и взаимосвязи данных;
- текстовый — правила создания и ведения текстовых документов, *уточняющих* и *поясняющих* графическую модель, — глоссариев, спецификаций, отчетов, заметок и т.п.

Уровни модели

- Уровень «сущность-связь» (ER level) — уровень наименее детального представления информации, который используется на начальной стадии моделирования, и на котором представлены лишь сущности и связи между ними.



- Уровень ключей (Key-Based Level, KB) — уровень представления первичных и внешних ключей сущностей, а также спецификаций связей. Предназначен для объявления уникальных идентификаторов экземпляров сущностей и ограничений ссылочной целостности.

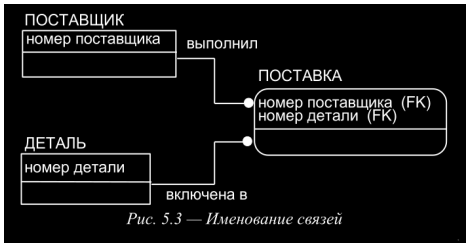


Рис. 5.3 — Именование связей

- Уровень атрибутов (Fully Attributed Level, FA) — отражает имена всех атрибутов сущностей и связей, полностью определяя структуру и взаимосвязи данных.

Нотации. Основные понятия.

- Сущность — отношение РМД, изображаемое именованным прямоугольником, в который вписываются имена атрибутов
  - ER-level IDEF1X — не различаются независимые и зависимые сущности.
  - На KB-level и FA — существуют зависимые и независимые сущности.
    Слабая сущность — это сущность, логически зависимая от какой-либо другой сущности (или сущностей)

- Атрибут — свойство или характеристика, общая для некоторых или всех экземпляров сущности;
  является конкретизацией домена в контексте сущности.

  - Присоединенный атрибут должен быть частью первичного ключа передавшей его родительской сущности. Присоединенный атрибут помечается символом «(FK)», следующим за именем атрибута.
  - Если атрибут является собственным атрибутом одной сущности и присоединенным атрибутом другой, то либо он имеет одинаковые имена в обеих сущностях, либо помечается именем роли как присоединенный.
  - По уровням (обязательна приписка -level, это не ER-,KB-,FA-модели):
    - ER: атрибуты не указываются
    - KB: показываются только атрибуты, входящие в состав первичных, альтернативных и внешних ключей:
      - Атрибуты альтернативных ключей обязательно помечаются символом «(AKn)», где n — номер альтернативного ключа.
        Все атрибуты, входящие в состав одного и того же альтернативного ключа, помечаются одним и тем же значением n.

- FA: показываются все атрибуты каждой сущности.
    - Атрибуты, не являющиеся частью первичного ключа, могут иметь неопределенные значения
      → помечаются символом «(O)», следующим за именем атрибута (Optional — необязательный).
- KB и FA: каждая сущность имеет не менее одного атрибута.
  Каждый атрибут может быть собственным атрибутом сущности или присоединенным (мигрировавшим), полученным от другой сущности через связь.

- Домен — специального синтаксиса не предусмотрено; определяется независимо от сущностей и диаграмм на основании требований ПО.

29. Модель IDEF1X. Понятие сущности. Сходства и различия понятий сущности в ER-модели и модели IDEF1X. Типы сущностей в модели IDEF1X. Изображения сущностей различных типов на диаграммах. Правила именования сущностей.

Понятие сущности

- Сущность — отношение РМД, изображаемое именованным прямоугольником, в который вписываются имена атрибутов
- Каждый экземпляр сущности должен иметь определенное (не NULL) значение каждого атрибута, являющегося частью первичного ключа.
- Не может быть экземпляра сущности, имеющего более чем одно значение какого-либо из атрибутов (1НФ).

Сходства и различия понятий сущности в ER-модели и модели IDEF1X

- ER: сущности не различаются как зависимые или независимые
- IDEF1X: различаются независимые и зависимые сущности
  Слабая сущность — это сущность, логически зависимая от какой-либо другой сущности (или сущностей)

Типы сущностей в модели IDEF1X

- IDEF1X: различаются независимые и зависимые сущности

Изображения сущностей различных типов на диаграммах

Правила именования сущностей

- Обязательно именуются.
- Именем может быть только имя существительное, возможно с определениями. В качестве имён допускаются аббревиатуры и акронимы.
- Имя должно быть уникальным и осмысленным. Однозначное описание смысла имени обязательно включается а глоссарий модели.
- Имя сущности должно иметь единственный смысл, и этот смысл всегда должен выражаться этим именем. Тот же смысл не может вкладываться в другое имя, если оно не является псевдонимом или синонимом основного.
- Сущности всегда именуются в единственном числе. Имя должно относиться к одному экземпляру сущности.

30. Модель IDEF1X. Понятие соединения. Типы соединений. Изображения соединений различных типов на диаграммах. Правила именования соединений. Маркировка свойств соединений.

Понятие соединения

Соединение — это один из двух видов связей — ассоциация между двумя сущностями или между экземплярами одной и той же сущности.

Типы соединений

- Неспецифическое соединение — это бинарная связь типа M:N, в которой нет родителя и потомка. Может быть показана только на диаграмме ER-уровня
  (На KB и FA — должна быть представлена эквивалентными парами специфических соединений).
- Специфическое соединение — это бинарная связь типа 1:M, в которой можно указать родителя (многосвязная сущность) и потомка (мощность 0 или 1).
  В состав атрибутов потомка обязательно входят все атрибуты первичного ключа родителя (внешние ключи потомка).
  - Идентифицирующее специфическое соединение — соответствующий ему внешний ключ полностью входит в первичный ключ потомка (идентификация экземпляров потомка возможна только через ссылку на соответствующий экземпляр родителя).
  - Неидентифицирующее специфическое соединение — соответствующий ему внешний ключ не входит в состав первичного ключа потомка или входит в него частично.

Изображения соединений различных типов на диаграммах

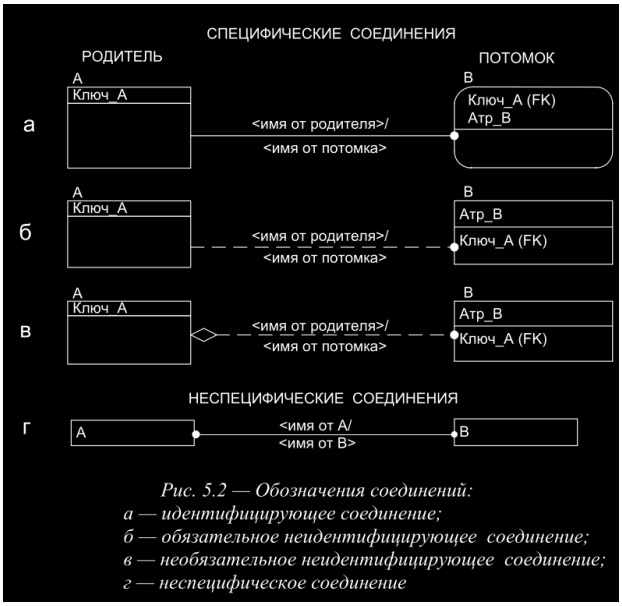


Рис. 5.2 — Обозначения соединений:
а — идентифицирующее соединение;
б — обязательное неидентифицирующее соединение;
в — необязательное неидентифицирующее соединение;
г — неспецифическое соединение

Правила именования соединений

- Соединению присваивается имя, выражаемое глагольным оборотом, которое зрительно привязывается к дуге, изображающей соединение.
- Имена соединений могут быть неуникальными в пределах диаграммы.
- Имя каждого соединения одной и той же пары сущностей должно быть уникальным во множестве имен связей этих сущностей.
- Имена специфических соединений выбираются так, чтобы можно было построить осмысленную фразу, составленную
  из имени родительской сущности, имени связи, выражения кардинальности и имени потомка.
- Связь может быть поименована «от родителя» и от «потомка».
  Имя «от родителя» обязательно.
- Если связь не именуется со стороны потомка, то имя «от родителя» должно выбираться так, чтобы связь легко читалась и со стороны потомка.
- Неспецифические соединения обязательно именуются с обеих сторон.

Маркировка свойств соединений

Таблица 5.1 — Спецификации кардинальности

Обозначение	Число возможных экземпляров связи
	0, 1 или более
p	1 или более
z	0 или 1
<N>	точно указанное число N
<N1> - <N2>	от N1 до N2

- Значение кардинальности (мощности) связи от родителя — около точки на конце дуги.

- Значение кардинальности связи со стороны потомка — указывается только для необязательного неидентифицирующего соединения.
- Для прочих типов — всегда равно 1.
- Ромб на конце дуги, примыкающий к родительской сущности — с каждым экземпляром потомка в связь вступает 0 или 1 экземпляр родителя.

31. Модель IDEF1X. Понятие атрибута. Сходства и различия понятий атрибута в ER-модели и модели IDEF1X. Правила именования атрибутов. Понятия первичного, альтернативного и внешнего ключей. Представление ключей на диаграммах.

Понятие атрибута. Представление ключей на диаграммах

- Атрибут — свойство или характеристика, общая для некоторых или всех экземпляров сущности;
  является конкретизацией домена в контексте сущности.
- Каждый атрибут является собственным атрибутом точно одной сущности.
- Присоединенный атрибут должен быть частью первичного ключа передавшей его родительской (или родовой) сущности. Присоединенный атрибут помечается символом «(FK)», следующим за именем атрибута.
- Каждый экземпляр сущности должен иметь определенное (*не NULL*) значение каждого атрибута, являющегося частью первичного ключа.
- Не может быть экземпляра сущности, имеющего более чем одно значение какого-либо из атрибутов (1НФ — простой тип данных).
- Атрибуты, не являющиеся частью первичного ключа, могут иметь неопределенные значения → помечаются символом «(O)», следующим за именем атрибута (Optional — необязательный).
- Если атрибут является собственным атрибутом одной сущности и присоединенным атрибутом другой, то либо он имеет одинаковые имена в обеих сущностях, либо помечается именем роли как присоединенный.
- Определение атрибута должно содержать ссылку на имя домена.

Сходства и различия понятий атрибута в ER-модели и модели IDEF1X

- ER-level: может показывать атрибуты и не должна показывать первичные, альтернативные или внешние ключи.
  (не путать с ER-моделью)
- Атрибут в ER-модели — это характеристика сущности или связи (свойство класса), значимая с точки зрения пользователя. Имеет имя и принимает значения:
  - Простой или составной (композитный):
    - Простой — значения атрибута в представлении пользователя не имеют внутренней структуры.
    - Композитный — имеет внутреннюю структуру.
  - Однозначный или многозначный:

        - Многозначный — атрибут может принимать более одного значения для одного экземпляра сущности.
        - Однозначный — может принимать только одно значение.
    - Первичный или производный:
        - Производный — значения атрибута могут быть определены по значениям других атрибутов
        - Первичный — значения определяются оригинально.

Правила именования атрибутов

- Обязательно именуются
- Именем может быть только имя существительное, возможно с определениями. В качестве имён допускаются аббревиатуры и акронимы.
- Имя должно быть уникальным и осмысленным. Однозначное описание смысла имени обязательно включается а глоссарий модели.
- Имя атрибута должно иметь единственный смысл, и этот смысл всегда должен выражаться этим именем. Тот же смысл не может вкладываться в другое имя, если оно не является псевдонимом или синонимом основного.
- Атрибуты всегда именуются в единственном числе. Имя должно относиться к одному значению атрибута.
- Никакие две сущности не могут иметь одноименных атрибутов, если они не связаны каким-либо отношением.

32. Модель IDEF1X. Понятия категории сущности и связи категоризации. Понятия категории, кластера категорий и дискриминатора кластера. Изображение связей категоризации на диаграмме.

Понятия категории сущности и связи категоризации

Связь категоризации (categorization relationship) — связь между родовой сущностью и категорией.

Понятия категории, кластера категорий и дискриминатора кластера

Каждая пара линий — от родовой сущности до окружности и от окружности до категории — представляет одну связь в кластере.

Спецификации кардинальности не указываются →
со стороны родовой сущность всегда (0 или 1)
со стороны категории всегда 1.

⇒ Категории одного и того же кластера не пересекаются.
⇒ Категории различных кластеров могут иметь общие экземпляры.

Связи категоризации не имеют явных имен.
От родовой сущности любую из них можно прочитать «может быть»,
а обратно — «является».

Изображение связей категоризации на диаграмме

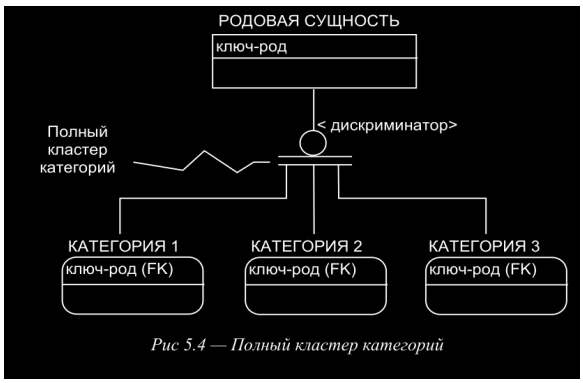


Рис 5.4 — Полный кластер категорий

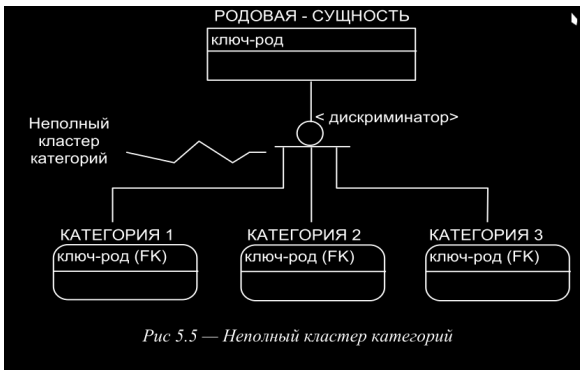


Рис 5.5 — Неполный кластер категорий