***1. Назначение и элементы автоматизированных информационных систем [АИС]***

Информация - сведения (сообщения, данные), независимо от формы их представления.

Под это определение попадают все сведения о фактах, лицах... Информация может быть представлена в различной форме, включая звуковую, графическую, видео, текстовую и компьютерную форму.

Данные - представление информации в формальном виде (в виде, подчиненному определенным правилам) пригодном для передачи, интерпретации или обработки людьми или компьютером.

Для сбора, хранения и эффективного использования информации служат \*\*информационные системы\*\*, функционирующие в различных предметных областях.

АИС хранят и обрабатывают информацию, представленную в виде данных в памяти вычислительной системы.

\*\*Предметная область\*\* - это часть реального мира, подлежащая анализу с целью организации управления и автоматизации. Предметная область рассматривается на определенном уровне абстракции. Предметная область может состоять из множества фрагментов.

Развитие АИС началось с файловых систем хранения данных. Такие системы включают набор прикладных программ, которые представляют для пользователя определенные операции с заданным множеством данных (ввод данных, формирование отчетов, ...)

Каждая программа хранила собственные данные и управляла ими, при этом формат физического хранения данных в файле определялся самой программой.

\*\*База данных\*\* - это совместно используемый набор логически связанных данных (и описания этих данных), хранимый в соответствии с определенной схемой данных и предназначенный для решения определенных задач предметной области. АИС, спроектированной по концепции баз данных, называют СБД [Система баз данных].

***7. Модель «сущность-связь». Элементы модели***

ER-модель представляет собой один из вариантов систем описания проектов БД. Исходя из этого, ER-модель предназначена для того, чтобы на этапе проектирования БД описать предметную область, структурировав объекты, их свойства и связи между ними. Следует отметить, что ER-модель предназначены для моделирования предметной области без сосредоточения на представлении данных в конкретной БД. В ER-модели структура данных отображается графически в виде диаграммы «сущность-связь». ER-диаграмма включает элементы 3 основных типов:

Сущность (entity)

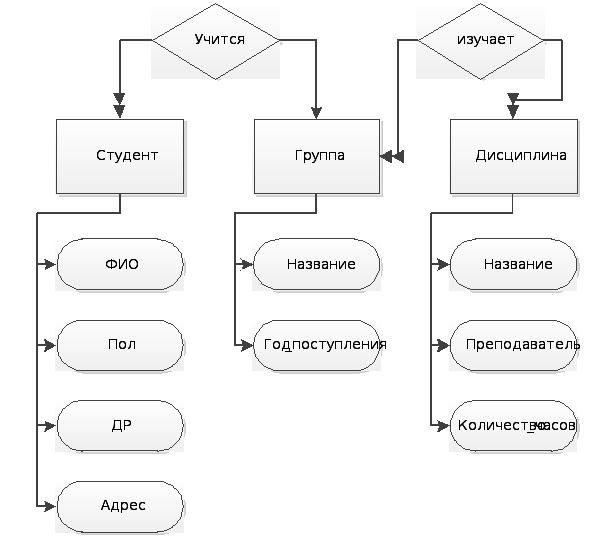
Атрибут (property)

Связь (relationship)

Сущность – это абстрактный объект определенного вида. Набор однородных сущностей называется множеством сущностей. Для простоты будем использовать термин «сущность» вместо термина «множество сущностей». Примерами сущностей являются: студенческая группа, студент, аудитория и т.п. На ER-диаграмме сущность обозначается прямоугольником.

Атрибуты – это свойства сущностей. Например, свойствами сущности «аудитория» являются: номер, количество мест, наличие проектора и т.п. На ER-диаграмме атрибуты обозначаются овалами.

Связи – это соединения между 2 или несколькими сущностями. На ER-диаграмме связи обозначаются в ромбах с соответствующими стрелочками.



***8. Модель «сущность-связь». Порядок построения. Нотации диаграмм***

ER-модель представляет собой один из вариантов систем описания проектов БД. Исходя из этого, ER-модель предназначена для того, чтобы на этапе проектирования БД описать предметную область, структурировав объекты, их свойства и связи между ними. Следует отметить, что ER-модель предназначены для моделирования предметной области без сосредоточения на представлении данных в конкретной БД. В ER-модели структура данных отображается графически в виде диаграммы «сущность-связь».

1) Определение сущностей:

Определите основные сущности (объекты), которые будут представлены в базе данных. Например, если вы создаете базу данных для учета сотрудников, сущности могут включать "Сотрудник", "Отдел", "Проект" и т.д.

2) Определение атрибутов:

Для каждой сущности определите ее атрибуты (свойства). Например, для сущности "Сотрудник" атрибутами могут быть "Имя", "Фамилия", "Должность" и т.д.

3) Определение отношений:

Определите отношения между сущностями. Отношения могут быть одним к одному, одним ко многим, многими к одному или многими ко многим. Например, сущность "Сотрудник" может иметь отношение "Работает в" к сущности "Отдел".

4) Определение первичных ключей:

Для каждой сущности определите первичный ключ (Primary Key), который уникально идентифицирует каждую запись в этой сущности.

5) Определение внешних ключей:

Если у вас есть отношения между сущностями, определите внешние ключи (Foreign Keys), которые связывают записи в одной таблице с записями в другой.

6) Создание диаграммы:

Используйте графические инструменты (например, круги для сущностей, линии для отношений) для создания диаграммы сущность-связь. Местоположение атрибутов, первичных ключей и внешних ключей также отображается на диаграмме.

Существует несколько нотаций для построения диаграмм модели сущность-связь (ERD). Некоторые из наиболее распространенных нотаций включают в себя:

**Нотация Чена (Chen's Notation):**

В этой нотации сущности представляются прямоугольниками, а отношения между сущностями - ромбами. Атрибуты сущностей указываются внутри соответствующих прямоугольников, а линии связи показывают, какие сущности связаны между собой.

**Нотация Кроу (Crow's Foot Notation):**

В нотации Кроу сущности также представляются прямоугольниками, но отношения между сущностями представлены символами, напоминающими лапы птицы (отсюда и название "Crow's Foot"). Одна лапа обозначает отношение "многие" (м:1 или 1:м), а несколько лап - отношение "многие ко многим" (м:м).

**UML-нотация:**

Unified Modeling Language (UML) также может использоваться для построения диаграмм модели сущность-связь. В этом случае используются блоки "классов" для представления сущностей, а линии связи - для отношений.

9.

В модели "сущность-связь" (Entity-Relationship, ER-модель) существуют стандартные элементы, такие как сущности, атрибуты и связи, но иногда требуется использовать особые схемы или расширения для описания более сложных структур данных. Вот несколько особенных и нестандартных схем, а также расширений модели "сущность-связь":

1) Иерархии:

Основная ER-модель иногда может быть расширена для описания иерархических отношений между сущностями. Например, модель "компания-подразделение-сотрудник" может содержать иерархию, где компания является корневой сущностью, подразделения - узлами следующего уровня, а сотрудники - листьями.

2) Специализация и обобщение:

Вводятся дополнительные концепции "специализация" и "обобщение", которые позволяют описывать отношения "является" и "является частным случаем". Например, сущность "Транспортное средство" может быть обобщена до "Автомобиль" и "Грузовик".

3) Ассоциативные сущности:

Введение ассоциативных сущностей для описания сложных отношений между другими сущностями. Например, если у вас есть отношение "Студент учится в Курсе", атрибуты этого отношения могут быть выделены в отдельную сущность "Регистрация", соединяющую студента и курс.

4) Многозначные атрибуты:

Расширение модели для поддержки многозначных атрибутов, т.е. атрибутов, которые могут содержать несколько значений. Это часто решается введением отдельной сущности, связанной с основной через отношение "имеет".

5) Метаданные:

Добавление сущностей и связей для описания метаданных. Например, для управления версиями или аудита можно ввести сущности "Версия" или "История изменений".

6) Денормализация:

Введение денормализации данных для оптимизации производительности. Например, предварительное вычисление и хранение агрегированных данных может уменьшить необходимость выполнения сложных запросов.

7) Событийные модели:

Введение сущностей и связей для моделирования событий. Например, сущности "Событие" и "Участник" могут быть связаны отношением "Участвует в".

Расширенные типы связей:

Расширение типов связей для учета сложных отношений, таких как связи "многие ко многим" с дополнительными атрибутами.

***10. Объединение моделей локальных представлений при проектировании базы данных***

Ну связи добавляем...

***11. Понятие модели данных. Ранние модели данных.***

Модель данных - это абстракция или формальное представление о том, как организованы и хранятся данные в информационной системе. Модели данных определяют структуру данных, их отношения и ограничения. Они служат основой для проектирования баз данных и обеспечивают понимание того, как данные могут быть использованы и обрабатываются в рамках конкретной системы.

Ранние модели данных:

1) Иерархическая модель данных:

Описание: Данные организованы в виде иерархии, представляющей собой древовидную структуру с корнем и ветвями. Каждая запись может иметь одну или несколько подчиненных записей. Примером является IMS (Information Management System).

+: Простота, эффективность для определенных структурированных данных.

-: Жесткость, сложность обновления и изменения структуры.

2) Сетевая модель данных:

Описание: Похожа на иерархическую модель, но допускает более сложные отношения между записями. Записи могут иметь несколько родителей и детей. Примером является CODASYL (Conference on Data Systems Languages) database.

+: Более гибкая, чем иерархическая модель, возможность представления сложных связей.

-: Сложность в реализации и использовании, менее интуитивна.

3) Реляционная модель данных:

Описание: Данные представляются в виде таблиц (отношений), где строки представляют кортежи, а столбцы - атрибуты. Отношения между таблицами устанавливаются ключами. Примером является модель, предложенная Эдгаром Коддом.

+: Простота, гибкость, легкость использования, отсутствие дублирования данных.

-: Изначально могла быть менее эффективной для некоторых типов запросов, но современные системы устраняют многие из этих ограничений.

Реляционная модель данных оказалась наиболее успешной и широко принятой, и большинство современных баз данных, таких как MySQL, PostgreSQL, Microsoft SQL Server, основаны на реляционной модели.

***12. Понятие модели данных. Реляционная модель данных***

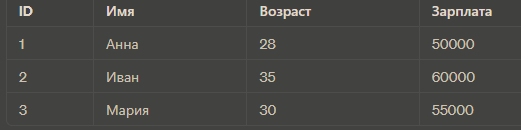
Модель данных - это абстракция, которая определяет структуру и организацию данных в информационной системе. Модели данных предоставляют средства для описания, хранения, обработки и манипулирования данными в рамках конкретной системы. Они являются основой для проектирования баз данных и обеспечивают структурированный способ представления информации.

Реляционная модель данных:

Реляционная модель данных (Relational Data Model) была предложена Эдгаром Коддом в 1970 году и является одной из самых широко используемых моделей данных в современных информационных системах. Основные концепции реляционной модели включают в себя:

Отношения (Таблицы): Данные представляются в виде двумерных таблиц, называемых отношениями. Каждая строка таблицы представляет собой кортеж (запись), а каждый столбец - атрибут (поле).

Пример отношения "Сотрудники":

******

***Ключи: Ключи используются для уникальной идентификации записей в отношении. Первичный ключ однозначно идентифицирует каждую запись, а внешний ключ связывает отношения между собой.***

***Пример: В отношении "Сотрудники" поле "ID" может быть использовано в качестве первичного ключа.***

***Целостность данных: Реляционная модель поддерживает целостность данных с использованием ограничений (constraints), таких как уникальность значений, проверки на null, ограничения целостности ссылок и другие.***

***Операции над данными: Для работы с данными в реляционной модели используются стандартные операции, такие как выборка (SELECT), вставка (INSERT), обновление (UPDATE) и удаление (DELETE). Также, для объединения данных используются операции объединения (JOIN).***

***Нормализация: Процесс нормализации в реляционной модели предназначен для устранения избыточности данных и обеспечения минимальных аномалий в хранении данных. Нормальные формы (1НФ, 2НФ, 3НФ и др.) помогают улучшить структуру базы данных.***

***13. Правила преобразования модели «сущность-связь» в реляционную модель данных.***

Преобразование модели "сущность-связь" (ER-модели) в реляционную модель данных, используемую в реляционных базах данных, осуществляется по определенным правилам. Вот основные шаги и правила для преобразования:

Сущности в таблицы:

Каждая сущность в ER-модели становится таблицей в реляционной модели. Имя таблицы обычно совпадает с именем сущности.

Атрибуты в столбцы:

Каждый атрибут сущности становится столбцом в таблице. Имя столбца обычно совпадает с именем атрибута.

Первичные ключи:

Идентифицируйте первичный ключ для каждой таблицы. Обычно используют атрибут сущности, который выделен как первичный ключ.

Связи внешними ключами:

Если существует связь между двумя сущностями в ER-модели, то в реляционной модели одна из таблиц получает внешний ключ, который ссылается на первичный ключ другой таблицы. Это создает внешнюю связь между таблицами.

Ассоциативные таблицы:

Если в ER-модели присутствуют ассоциации многие ко многим, они реализуются через создание дополнительной (ассоциативной) таблицы, которая содержит в себе внешние ключи, связывающие две связанные сущности.

Развитие атрибутов:

Атрибуты, которые являются производными или зависят от других атрибутов, должны быть вычислены на основе этих других атрибутов. Они не хранятся явно в таблице, если они могут быть легко вычислены.

Слабые сущности:

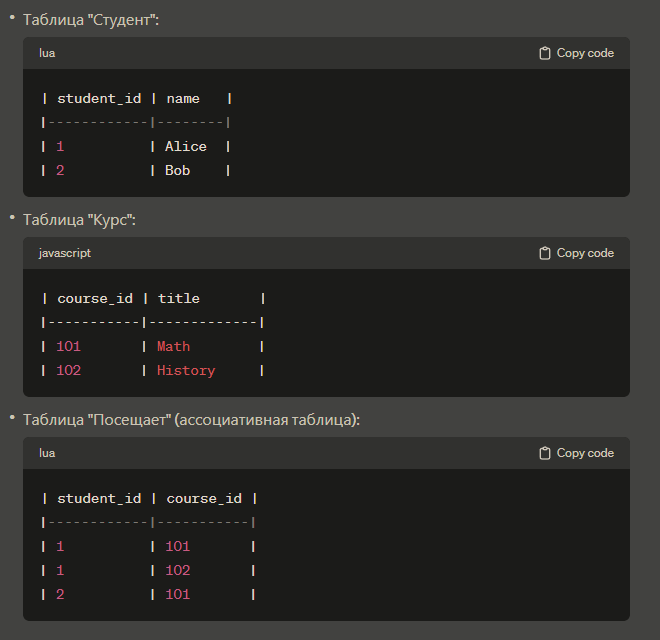
Если в ER-модели есть слабые сущности, то они могут быть реализованы как отдельные таблицы, а первичный ключ слабой сущности может включать в себя первичный ключ связанной сущности.

Нормализация:

Применяйте процесс нормализации для устранения избыточности и обеспечения структуры базы данных в соответствии с требованиями нормальных форм.

Пример:

Рассмотрим ER-модель сущностей "Студент" и "Курс" с ассоциацией многие ко многим "Посещает". Преобразование может выглядеть следующим образом:



***14. Операции реляционной алгебры***

Это коллекция операций, которые принимают в качестве операндов отношения и возвращают отношения в качестве результата. Все операции реляционной алгебры подразделяются на 2 класса: традиционные операции с множествами, а именно:

1. Объединение 
2. Пересечение 
3. Разность 
4. Декартово произведение 
5. Селекция (выборка) 
6. Проекция 
7. Соединение 
8. Частное 

Объединение – результатом объединения отношений , имеющих одинаковый заголовок, является отношение, включающее все кортежи отношений R и S.

Пересечением отношений R и S является отношение, кортежи которого входят в R и S одновременно.

Разность – R-S – отношение, включающее все кортежи отношения уменьшаемого кроме имеющихся в вычитаемом.

RS=R-(R-S)

Декартовым произведением отношений R,S является такое отношение, которое заголовком включает все атрибуты из R, S, а тело – все кортежи, являющиеся всеми возможными парами кортежей R и S.

Селекция – унарная операция - (R) – селекцией над отношением R является множество кортежей R, удовлетворяющих условию . – логический оператор.

Пример: 

Проекция - (R) – отношение, построенное на подмножестве атрибутов отношения R. Пример: (R)

Частное –, n – арность R, k – арность S. Данная операция применима только, если множество атрибутов делителя является подмножеством атрибутов делимого.

Соединение:

1. Естественное: представляет собой соединение двух отношений по совподающим значением атрибутов. |Какая-то формула|.
2. Тета-соединение: = , n - арность отношения R,  - логический оператор, i,j – номера отношений R и S - представляет собой выборку из декартового произведения 2 отношений по определенному условию.

## 15. Правила построения формул реляционного исчисления с переменными кортежами.

Реляционное исчисление позволяет описывать отношения и операции над ними в виде аналитического выражения или формулы. Если в реляционной алгебре определяется порядок выполнения операций над отношениями, приводящий к требуемому результату, то в реляционном исчислении описывается требуемое отношение, основываясь на существующих отношениях. Неформальный пример выражения реляционного исчисления может выглядеть следующим образом: получить фамилии студентов, для которых значение наименования факультета равно ФАИ. Несмотря на указанное различие формулировок алгебры исчисления логически, они эквивалентны, т.е. каждому выражению реляционной алгебры может быть поставлено в соответствие выражение с реляционными исчислением.

***Реляционное исчисление с переменными кортежами***

В общем виде формулы реляционного исчисления с переменными кортежами имеют вид:



 - кортеж фиксированной длины i

 - формула построенная из атомов и совокупности операторов

Атомы бывают 3-х видов:

1.  - кортеж t принадлежит отношению R
2.  - i-й компонент кортежа s находится в отношении с j-м компонента кортежа u
3.  - некоторое значение a находится в отношении  с j-м компонентом кортежа u

Существуют свободные и связные вхождения переменных в формулу. Вхождение в формулу называется связанным, если переменной предшествуют кванторы существования и всеобщности . В противном случае переменная является свободной. Формулы, а также свободные связанные вхождения переменных в эти формулы определяется рекурсивно следующим образом:

1. Каждый атом есть формула. Все переменные кортежей, входящие в атом, являются свободными в этой формуле.
2. Если  и  - формулы. Для формул определены операции И, ИЛИ, НЕ.
3. Если  - формула, то - существует кортеж s, удовлетворяющий условию .

 - для любого кортежа s выполняется условие .

1. Формулы при необходимости могут заключаться в скобки, при этом принят следующий приоритет выполнения операций
   1. Скобки
   2. Арифметические операторы
   3. Операторы сравнения
   4. Кванторы
   5. Логические операторы
2. Единственной, свободно входящей в формулу  является вхождение кортежей s, множество которой и определяется этой формулой. Все остальные переменные кортежа должны быть связаны кванторами
3. Ни что иное не является формулой

## 

## 17. Нормализация реляционных отношений

Проблемы проектирования заключаются в выделении типов записей и определение состава атрибутов в отношении. Для этой цели был разработан аппарат нормализации отношений. Сложность задач группировки атрибутов отношения обусловлена большим количеством возможных вариантов. Рациональный вариант группировки атрибутов отношения должны отвечать следующим требованиям:

1. Число выбранных атрибутов для первичных ключей должно быть минимально.
2. Выбранный состав отношений должен быть по возможности минимальным.

Первая нормальная форма: схема отношения R находится в первой нормальной форме, когда все атрибуты, входящие в состав R являются атомарными – на одном домене определен один атрибут. Преобразование ненормализованного отношения к нормализованному может привести к изменению арности отношения и его первичного ключа.

Рейс (номер, пункт\_отпр, пункт\_назн, РАСПИСАНИЕ)

РАСПИСАНИЕ(день\_нед, время\_отпр)

Р003 Липецк Москва Пн 9-00

Ср 10-00

Пт 9-00

З011 Липецк Сочи Сб 15-00

ВС 17-00

РЕЙС(номер, пункт\_отпр, пункт\_назн,день\_нед, время)

Р003 Липецк Москва Пн 9-00

Р003 Липецк Москва Ср 10-00

Р003 Липецк Москва Пт 9-00

З011 Липецк Сочи Сб 15-00

З011 Липецк Сочи ВС 17-00

Вторая нормальная форма

X,Y – атрибуты

X->Y

Х зависит от Y, если в любой момент времени каждому значению Х соответствует не более одного значения Y. Неключевой атрибут функционально полно зависит от составного ключа, если функционально зависит от этого ключа, но не находится в зависимости ни от какой части составного ключа.

Отношение находится во второй нормальной форме, если оно находится в первой нормальной форме, и каждый неключевой атрибут функционально полно зависит от составного ключа.

X->Y

AB->C

A\*>C

B\*>C

ПОСТАВКА(№\_П, товар, цена)

Для этого отношения не выполняется условие второй нормальной формы, т.к. цена зависит от товара. Неполная функциональная зависимость от неключевого атрибута цена от ключа приводит к следующим аномалиям:

1. аномалия включения – нельзя сохранить в БД информацию о товаре, если его еще не поставляли.
2. аномалия корректировки – при изменении цены товара необходим полный просмотр отношения с целью корректировки всех кортежей, где имеются сведения об этом товаре.

ПОСТАВКА(№\_П, Товар)

ТОВАР(Товар, Цена)

Третья нормальная форма

X,Y,Z – атрибуты некоторого отношения, при этом Y зависит от X, Z зависит от Y, но X не зависит от Y и Y не зависит от Z.

Тогда говорят, что Z транзитивно зависит от X.

Отношение находится в третьей нормальной форме, если оно находится во второй нормальной форме и каждый неключевой атрибут не транзитивно зависит от первичного ключа.

ХРАНЕНИЕ(ФИРМА, СКЛАД, ОБЪЕМ)

ФИРМА->СКЛАД->ОБЪЕМ

В результате имеются две аномалии:

1. аномалия включения – нельзя сохранить в БД информацию о складе, с которого еще не получает товар ни одна фирма
2. аномалия корректировки – при изменении объема склада необходимо корректировать все записи, относящиеся к ютому складу

ХРАНЕНИЕ(ФИРМА, СКЛАД)

СКЛАДЫ(СКЛАД, ОБЪЕМ)

Нормальная форма Бойса-Кодда (НФБД)

Отношение находится в НФДС, если оно находится в 3й нормальной форме и в нем отсутствуют зависимости первичных атрибутов от не первичных.

ПРОЕКТ(№\_ПР, №\_Д, №\_П)

Каждая деталь проекта поставляется одним поставщиком, а каждый поставщик обслуживает один проект.

№\_ПР,№\_Д -> №\_П

№\_П -> №\_ПР

Наблюдаются аномалии:

1. Факт поставки поставщиком детали не может быть сохранен в БД, пока деталь не будет использоваться в проекте.

ДЕТАЛИ(№\_Д, №\_ПР)

ПОСТАВКА(№\_П, №\_ПР)

4НФ

X многозначно определяет Y, если каждому значению Х соответствует множество значений Y, не связанных с другими атрибутами отношен

X ->> Y

ПРЕПОДАВАТЕЛЬ(№\_ПР, ДЕТИ, КУРСЫ, Должность)

№\_ПР ->> ДЕТИ

№\_ПР ->> КУРСЫ

R – схема отношения. B – множество зависимостей в R. R в 4НФ если оно в НФБК, и существует зависимость.

5НФ.

Замечания:

1. При проектировании схемы бБД с использованием морфологического подхода как правило получается нормализованная схема БД,
2. Нормализация отношений – не самоцель, кроме 1НФ