Предметная область – некоторая область реального мира использующая конкретную информацинну систему.

Информация – абсолютно любые сведения о каком либо событии, сущности, процессе или явлении характерные для рассматриваемой предметной области.

Данные – это информация зафиксированная в какой то форме, и пригодная для обработки, хранения и передачи

Знания – это закономерности предметной области позволяющие ставить и решать задачи в этой области.

Информационный процесс – сбор , хранение и преобразование данных о предметной области

Информационная система – вычислительная среда обеспечивающая процесс обработки данных

Файловая система – набор программ которые выполняют для пользователй определенные операции, каждая программа определяет свои данные и управляет ими

Проблемы файловых систем: определение изоляция данных, дублирование данных, зависимость от данных, не современность форматов данных, фиксированные запросы

БД- это совокупность специальным образом организованных данных хранимых в памяти вычислительной системы, и отражающих состояние и взаимодействие объектов в предметной области

СУБД – специальное П. О. с помощью которого пользователи могут определять, создавать и поддерживать БД и осуществлять к ней контролируемый доступ.

Назначение СУБД

1. Хранение данных
2. Управление буфером оператора памяти
3. Управление транзакциями
4. Журналирование
5. Поддержка языков БД
6. Поддержка словаря данных

Администратор БД – лицо отвечающие за требования к БД ее проектирование, реализацию, эфф. Использоввание, а так же защиту от несанкц. Доступа к данным

Функции администратора

1. Определение концептуальной модели
2. Определение внутренней модели
3. Обеспечение взаимодействие с пользователем
4. Обеспечение целостности и безопасности данных
5. Координация всех устройств по бд
6. Обеспечение резервного копирование и востановления
7. Ведение словаря данных
8. Реагирование на изменение аспектов проектирования : инфологическое датологическое

Инфологическое (концептуальное) проектирование – процесс создания внешней (инфологической) модели данных о предметной области, не зависящее от любых физических аспектов ее представления.

Логическое (даталогическое) проектирование — создание схемы базы данных на основе конкретной модели данных, например, реляционной модели данных.

Сущность — это любой конкретный или абстрактный объект в рассматриваемой предметной области, информацию о котором необходимо хранить в базе данных.

Атрибут — это именованная характеристика сущности. Наименование атрибута должно быть уникальным для конкретного типа сущности, но может быть одинаковым для различного типа сущностей. Атрибуты используются для определения того, какая информация должна быть собрана о сущности.

Связь — это ассоциирование двух или более сущностей, которое указывает, каким образом связаны сущности. Эта информация необходима для поддержания целостности данных.

первичный ключ — минимальный набор столбцов таблицы, значения которых однозначно идентифицируют каждую строку этой таблицы.

суррогатный ключ — столбец, не являющийся атрибутом сущности, предназначенный для обеспечения уникальности каждой строки таблицы базы данных. В качестве типа данных обычно выбирается целочисленный тип данных

Значение суррогатного ключа для каждой строки таблицы генерируется автоматически средствами СУБД и называется идентификатором записи

внешний ключ — набор из одного или нескольких столбцов одной таблицы (подчинённой), которая имеет связь с другой таблицей (главной).

***Транзакция*** - Последовательные операции над БД

**Реляционная БД** – это тело связанной информации, сохраняемой в двумерных таблицах. Напоминает адресную и телефонную книгу.

**Виды первичных ключей.**

Обычно первичный ключ – числовое значение. Но он также может быть и любым другим типом данных. Не является обычной практикой использование строки в качестве первичного ключа (строка – фрагмент текста), но теоретически и практически это возможно.  
Составные первичные ключи.  
Часто первичный ключ состоит из одного поля, но он может быть и комбинацией нескольких столбцов, например, двух (трех, четырех…). Но вы помните, что первичный ключ всегда уникален, а значит нужно, чтобы комбинация n-го количества полей, в данном случае 2-х, была уникальна.

**Архитектура системы баз данных** — это совокупность ее основных функциональных компонентов,

а также средств обеспечения их взаимодействия друг с другом пользователями и системным персоналом.

**Атрибут** — свойство некоторой сущности. Часто называется полем таблицы.  
  
**Домен атрибута** — множество допустимых значений, которые может принимать атрибут.  
  
**Кортеж** — конечное множество взаимосвязанных допустимых значений атрибутов, которые вместе описывают некоторую сущность (строка таблицы).  
  
**Отношение**  — конечное множество кортежей (таблица).  
  
**Схема отношения** — конечное множество атрибутов, определяющих некоторую сущность. Иными словами, это структура таблицы, состоящей из конкретного набора полей.  
  
**Проекция** — отношение, полученное из заданного путём удаления и (или) перестановки некоторых атрибутов.  
  
**Функциональная зависимость** между атрибутами (множествами атрибутов) X и Y означает, что для любого допустимого набора кортежей в данном отношении: если два кортежа совпадают по значению X, то они совпадают по значению Y. Например, если значение атрибута «Название компании» — Canonical Ltd, то значением атрибута «Штаб-квартира» в таком кортеже всегда будет Millbank Tower, London, United Kingdom. Обозначение: {X} -> {Y}.  
  
**Метод нормальных форм (НФ)** состоит в сборе информации о объектах решения задачи в рамках одного отношения и последующей декомпозиции этого отношения на несколько взаимосвязанных отношений на основе процедур нормализации отношений.  
  
**Цель нормализации**: исключить избыточное дублирование данных, которое является причиной аномалий, возникших при добавлении, редактировании и удалении кортежей(строк таблицы).  
  
**Аномалией** называется такая ситуация в таблице БД, которая приводит к противоречию в БД либо существенно усложняет обработку БД. Причиной является излишнее дублирование данных в таблице, которое вызывается наличием функциональных зависимостей от не ключевых атрибутов.  
  
**Аномалии-модификации** проявляются в том, что изменение одних данных может повлечь просмотр всей таблицы и соответствующеев изменение некоторых записей таблицы.  
  
**Аномалии-удаления** — при удалении какого либо кортежа из таблицы может пропасть информация, которая не связана на прямую с удаляемой записью.  
  
**Аномалии-добавления** возникают, когда информацию в таблицу нельзя поместить, пока она не полная, либо вставка записи требует дополнительного просмотра таблицы.

**Трехуровненвая организация;**

* Внешний уровень - это тот, на котором представляют данные пользователи;
* Концептуальный уровень служит для отображения данных внешнего уровня на внутренний и обеспечивает необходимую независимость данных разных уровней друг от друга;
* На внутреннем уровне данные воспринимаются СУБД и операционной системой

**Внешний уровень -** это самый верхний уровень, который отражает представление конечного пользователя о конфигурации данных. Каждый пользователь представляет реальный мир по-своему, исходя из тго вида работы, которую он выполняет.

**Концептуальный уровень** - это объединение представления данных, используемых всеми пользовательскими приложениями, работающими с данной базой. На этом уровне база данных представляет собой общий згляд пользователя на данные проектируемой базы.

**Внутренний уровень** - служит для адаптации концептуальной модели к конкретной СУБД. Другими словами, внутренний уровень - это представление базы данных со стороны СУБД, и на этом уровне отписывается, как данные должны хранится в компьютере. А потому на этом уровне требуется что бы проектировщик привел свойства и ограничения концептуальной модели в соответствие с выбранной моделью реализации базы данных. Внутренний уровень предназначен для достижения оптимальной производительности и обеспечения оптимального использования дискового пространства. На этом уровне осуществляется заиможействие СУБД с методами доступа операционной системы. Здесь хранится такая информация:

* Распределение дискового пространства для хранения данных и индексов
* Описание подробностей хранения данных
* Сведения о размещении записей
* Сведения о сжатии данных и методах их шифрования

Команды для работы с базами данных

1. Просмотр доступных баз данных

SHOW DATABASES;

2. Создание новой базы данных

CREATE DATABASE;

3. Выбор базы данных для использования

USE <database\_name>;

4. Импорт SQL-команд из файла .sql

SOURCE <path\_of\_.sql\_file>;

5. Удаление базы данных

DROP DATABASE <database\_name>;

Работа с таблицами

6. Просмотр таблиц, доступных в базе данных

SHOW TABLES;

7. Создание новой таблицы

CREATE TABLE <table\_name1> (

<col\_name1> <col\_type1>,

<col\_name2> <col\_type2>,

<col\_name3> <col\_type3>

PRIMARY KEY (<col\_name1>),

FOREIGN KEY (<col\_name2>) REFERENCES <table\_name2>(<col\_name2>)

);

Ограничения целостности при использовании CREATE TABLE

Может понадобиться создать ограничения для определённых столбцов в таблице. При создании таблицы можно задать следующие ограничения:

ячейка таблицы не может иметь значение NULL;

первичный ключ — PRIMARY KEY (col\_name1, col\_name2, …);

внешний ключ — FOREIGN KEY (col\_namex1, …, col\_namexn) REFERENCES table\_name(col\_namex1, …, col\_namexn).

Можно задать больше одного первичного ключа. В этом случае получится составной первичный ключ.

Пример

Создайте таблицу «instructor»:

CREATE TABLE instructor (

ID CHAR(5),

name VARCHAR(20) NOT NULL,

dept\_name VARCHAR(20),

salary NUMERIC(8,2),

PRIMARY KEY (ID),

FOREIGN KEY (dept\_name) REFERENCES department(dept\_name)

);

8. Сведения о таблице

Можно просмотреть различные сведения (тип значений, является ключом или нет) о столбцах таблицы следующей командой:

DESCRIBE <table\_name>;

9. Добавление данных в таблицу

INSERT INTO <table\_name> (<col\_name1>, <col\_name2>, <col\_name3>, …)

VALUES (<value1>, <value2>, <value3>, …);

При добавлении данных в каждый столбец таблицы не требуется указывать названия столбцов.

INSERT INTO <table\_name>

VALUES (<value1>, <value2>, <value3>, …);

10. Обновление данных таблицы

UPDATE <table\_name>

SET <col\_name1> = <value1>, <col\_name2> = <value2>, ...

WHERE <condition>;

11. Удаление всех данных из таблицы

DELETE FROM <table\_name>;

12. Удаление таблицы

DROP TABLE <table\_name>;

Команды для создания запросов

13. SELECT

SELECT используется для получения данных из определённой таблицы:

SELECT <col\_name1>, <col\_name2>, …

FROM <table\_name>;

Следующей командой можно вывести все данные из таблицы:

SELECT \* FROM <table\_name>;

**Целостность базы данных** — соответствие имеющейся в базе данных информации её внутренней логике, структуре и всем явно заданным правилам.

**Нормализация –** процесс преобразования базы данных к виду, отвечающему нормальным формам.

**Нормальная форма –** совокупность требований, которым должно удовлетворять отношение.

**Типы нормальных форм**

• первая нормальная форма (1 NF);

• вторая нормальная форма (2 NF);

• третья нормальная форма (3 NF);

• нормальная форма Бойса-Кодда (BCNF);

• четвѐртая нормальная форма (4 NF);

• пятая нормальная форма (5 NF).

**Первая нормальная форма (1 NF)**

Отношение находится в 1 NF тогда и только тогда, когда все входящие в него атрибуты являются атомарными (неделимыми).

**Вторая нормальная форма (2 NF)**

Отношение находится в 2 NF, если оно находится в 1 NF и каждый неключевой атрибут функционально полно зависит от первичного ключа.

**Третья нормальная форма (3 NF)**

Отношение находится в 3 NF в том и только в том случае, если оно находится в 2 NF и каждый неключевой атрибут не- транзитивно зависит от первичного ключа.

Транзитивной зависимостью неключевых атрибутов от ключевых называется следующая: A → B и B → C, где A – набор ключевых атрибутов (ключ), B и С – различные множества неключевых атрибутов.

**Основные свойства нормальных форм**

каждая следующая нормальная форма в некотором смысле является более ограниченной, но более лучшей, чем предыдущая;

при переходе к следующей нормальной форме положительные свойства предыдущих нормальных свойств сохраняются.

**Операции над БД**

1. **Объединение R3= R1R2**

В результате R3 помещаются все кортежи, которые есть в R1 или в R2, причем кортежи, которые одновременно присутствуют в R1 и в R2, помещаются в результат один раз.

1. **Пересечение R3= R1R2**

В R3 помещаются кортеж, которые есть в R1 и R2.

1. **Разность R3= R1R2**

В R3 попадут кортежи, которые есть в R1 но нет в R2

Заметим что операция пересечения может быть вычислена через операции разности R3 = R1R2=R1-(R1-R2)

1. **Декартово произведение R3=R2xR2**

Результат получается путем клейки каждого кортежа отношения R1 c каждым кортежем R2

c=a\*b

a - количество кортежей в R1

b - количество кортежей в R2

c - количество кортежей в R3

При этом степень R3 равна сумме степеней R1 и R2

|  |  |
| --- | --- |
| R1 | |
| 1 | A |
| 2 | B |
| 3 | C |

|  |  |
| --- | --- |
| R2 | |
| 3 | C |
| 4 | D |
| 5 | E |

|  |  |  |  |
| --- | --- | --- | --- |
| Результат R3 | | | |
| 1 | A | 3 | C |
| 2 | B | 3 | C |
| 3 | C | 3 | C |
| 1 | A | 4 | D |
| 2 | B | 4 | D |
| 3 | C | 4 | D |
| 1 | A | 5 | E |
| 2 | B | 5 | E |
| 3 | C | 5 | E |

**Кодд вел дополнительные операции над отношениями**

1. **Проекция - отбор атрибутов отношения**

R2 = п1(R1)

I - подмножество атрибутов отношения R1

Степень результата R2 равна |I| - количество элементов в подмножестве I и может принимать целые значения от 1 до степени R1

Проекция R1 из предыдущих примеров по 2-му атрибуту

Результат R2 содержит всего один атрибут.

|  |
| --- |
| А |
| B |
| C |

1. **Операция выборки (селекции) - отбор кортежей**

R2=(R1)

И константы.

В R2 включает все кортежи из R1 для которых - истинно. При этом отношение R2 может не содержать ни одного кортежа или совпадать с отношением R1. Например, при условии Первый атрибут > 0

В R2 попадают все кортежи R1 а при условии первый атрибут > 2 только один кортеж (3 , С)

На практике операции в выборки и проекции часто сочетаются в одном запросе.

1. **Операция соединения R3 = R1 R2**

Данная операция определена над двумя отношениями, у которых есть общее подмножество атрибутов ( на практике это чаще всего один общий атрибут, по которому и выполняется операция соединения). В отличии от операции декартова произведения при соединении склеиваются только те кортежи R1 и R2, которые имею одно и то же значение общего атрибута ( а не каждый кортеж с каждым). При этом общий атрибут попадает в результат один раз.

Пусть выполняются операции соединения по первому из атрибутов ( содержащего числовые значения)

|  |  |
| --- | --- |
| R1 | |
| 1 | A |
| 2 | B |
| 3 | C |

|  |  |
| --- | --- |
| R2 | |
| 2 | C |
| 2 | D |
| 3 | E |
| 4 | F |

|  |  |  |
| --- | --- | --- |
| Результат R3 | | |
| 2 | B | C |
| 2 | B | D |
| 3 | C | E |

Операция соединения эквивалентна операции выборки из декартова произведения отношения R1 и R2

R3 = R1R2 = (R1xR2)

Однако учитывая большую важность этой операции для реляционной баз данных, где связь между отношениями устанавливаются при помощи общих ресурсов атрибутов, она включена в состав базовых операций реляционной алгебры.

Следует отметить, что в результат операции соединения не входят кортежи отношений R1 и R2 для которых не находится одинаковых значений в общем атрибуте. Так, в предыдущем примере в результат не вошел кортеж (1, А) из отношениями R1 и (4, F) из отношения R2. Ввиду этой особенности данную операцию называют операцией *внутреннего соединения*.

В языке SQL поддержат три операции *внешнего соединения* - левое, правое и полное.

В левом внешнем левом внешнем соединении результат внутреннего соединения дополняются оставшимися кортежами отношения, стоящего слева. Поскольку в результате должно быть 3 атрибута, незаполненный атрибут принимает значения NULL, т.е. в результате R3 добавляется кортеж (1,A,NULL)

В правом внешнем соединении результат внутреннего соединения дополняется оставшиеся кортежами отношения, стоящего справа.

Наконец в полном внешнем соединении в результат добавляются все несвязные кортежи, дополненные неопределенными значениями

1. **Операция деления R3 = R1 / R2**

Для выполнения операции деления отношения R1 и R2 должны иметь общее подмножество атрибутов (обычно один атрибут) причем в отношении R2 это подмножество является множеством его атрибутов (обычно R2 является унарным отношением).

|  |  |
| --- | --- |
| R1 | |
| 1 | A |
| 1 | B |
| 2 | C |
| 3 | A |
| 3 | B |
| 4 | A |

|  |
| --- |
| R2 |
| A |
| B |

В результате получаем отношение

|  |
| --- |
| R3 |
| 1 |
| 3 |

Операция деления явно не поддерживается в языке SQL, хотя имеется несколько способов выразить ее через другие операции