Базы данных.

Ульман Основы систем бд

Видосы по бд мехмат мгу

Предметная область – это область реального мира, использующая конкретную инф систему.

Информация – абсолютно любые сведения о каком-либо событии, сфере, явлении, характерные для рассматриваемой предметной области.

Данные – это информация, зафиксированная в определенной форме, пригодной для последующей хранения, обработки и передачи

Файловая система – набор программ, которые выполняют для пользователя необходимые операции.

Проблемы:

1) Разделение и изоляция данных.
2) Дублирование данных.
3) Зависимости от данных.
4) Не совместимость форматов файлов.
5) Фиксированные запросы.

Факторы:

1) Хранение данных в самой программе
2) Не было инструментов для работы с данными

БД – это совокупность специальным образом организованных данных хранимых в памяти ВС и отображающих состояние объектов, и их взаимосвязей в рассматриваемой предметной области.

БД не может существовать сама по себе нужна СУБД

СУБД – программное обеспечение, с помощью которого пользователи могут определять, создавать и поддерживать БД, а также осуществлять к ней контролируемый доступ.

Строилась на ней две системы

Файл - сервер

Клиент – сервер

Архитектура БД

Фотка 2

Внешний и концептуальный – являются логическими

Внутренний – является физическим

Целью данной 3х уровневой архитектуры является отделение пользовательского преставления БД от ее физического.

Концептуальная модель включает:

- Сущности, атрибуты и связи между собой
- Накладываемы на данные ограничения
- Схематически информация о данных

На внутреннем уровне БД представляется в виде совокупности хранимых файлов, для которых известна структура хранимых записей, определены служебные поля реализующие связи между записями определены методы доступа СУБД к этим записям.

Внешний уровень, который разделяется на несколько преставлений для каждого пользователя имеется свое собственное представление БД и содержит только те сущности, атрибуты и связи между ними, которые нужны конкретному пользователю.

Логическая независимость от данных, то есть полную защищенность внешних схем от изменений концептуальной схемы.

Физическая независимость от данных, то есть защищенность концептуальной модели от изменений вносимых во внутреннею модель.

Функции СУБД

1) Непосредственное управление данными во внешней памяти;
2) Управление буферами ОЗУ;
3) Управление транзакциями;
   Транзакция – это последовательность операций над БД рассматриваемых СУБД как единое целое.
4) Журнализация – это особая часть БД не доступная пользователям СУБД и поддерживаемая особой тщательностью, в которую поступают записи обо всех изменениях БД;
   Под надёжностью хранения данных будем понимать, что СУБД должна быть в состоянии восстановить последнее согласованное состояние БД после любого программного или аппаратного сбоя.
5) Поддержка языков БД;

6) Поддержка словаря данных – метаданные.

Администратор БД – лицо или группа лиц, отвечающая за выработку требований к БД, ее проектирование, реализацию, эффективное использование и сопровождение. Включая управление учетными записями пользователей БД и защиту от несанкционированного доступа.

Задачи администратора БД:

1) Определение концептуальной модели;
2) Определение внутренней модели;
3) Взаимодействие с пользователями;
4) Определение правил безопасности и целостности данных;
5) Координация всех действий по проектированию БД;
6) Определение процедур резервного копирования и восстановления;
7) Ведение словаря данных;
8) Управление производительностью и реагирование на изменяющиеся потребности.

Лекция 2.

Инфологический аспект проектирования – какой-то логический аспект данных, не зависимый от физического расположения данных.

На этапе инфологического проектирования:

1) О каких объектах, событиях реального мира требуется накапливать и обрабатывать информацию в системе .
2) Какие их хар-ки и взаимосвязи необходимо учитывать в системе
3) Уточняются все вводимые в информационную систему понятия об объектах их характеристиках

Датологической аспект проектирования БД употребляется при рассмотрении вопросов представления данных в памяти инф системы.

При датологическом проектировании учитывая возможности имеющийся средств ввода, хранения и обработки информации разрабатываются соответствующие формы представления информации в системе, а также приводятся модели и методы представления и преобразования данных, формулируются правила их смысловой интерпретации.

Процесс проектирования

Модель сущность-связь — это неформальная модель предметной области, которая используется на инфологическом этапе.

Основным назначением модели является семантическое описание предметной области и представление информации для обоснования выбора видов моделей и структур данных.

Сущность – собирательное понятие, некоторая абстракция реально существующего объекта, процесса или явления, о котором необходимо хранить информацию в системе.

Виды сущностей – слабые и сильные

Слабые – сущность существование которой зависит от какой-то другой. - прямоугольник

Сильная – существование которой не зависит от другой. - Прямоугольник в прямоугольнике

Схема 1 – виды сущностей

Атрибут – поименовав хар-ка сущности, которая принимает значения из некоторого множества.

Доме - Множество значений, которое может принимать атрибут

Атрибуты могут быть:

- Однозначный – атрибут для данной конкретной сущности может принимать только одно значение. (Овал)
- Многозначный - атрибут для данной конкретной сущности может принимать множество значений. (Двойной овал)
- Производный – атрибут, который может принимать значения от другого связанного с ним атрибута. (Пунктирный овал)
- Ключи

Ключ – это элемент данных позволяющий уникально идентифицировать отдельные экземпляры некоторой сущности.

- Потенциальный ключ – уникальный атрибут или набор атрибутов, который идентифицирует сущность
- Составной ключ – потенциальный ключ, который состоит из двух или более атрибутов.
- Первичный ключ – потенциальный ключ, который выбран в виде первичного ключа.

Связь – это осмысленна ассоциация между двумя сущностями. (Ромбик) Если соединяет слабую сущность, то ромбик двойно.

Степень связи – кол-во связей между сущностями.

Показатель кардинальности связи – описывает кол-во возможных связей для каждой из сущности участниц. (1:1 1:N N:N)

1:1 – определяет такой тип связи между двумя сущностями, когда каждому экземпляру сущности А соответствует один и только один экземпляр сущности Б и наоборот.

1:N – определяет такой тип связи между сущностями А и Б, когда каждому экземпляру сущности А может соответствовать один или несколько экземпляров сущности Б.

N:N – определяет такой тип связи сущностей А и Б, при котором каждому экземпляру сущности может соответствовать один или более экземпляров сущности Б.

Модель данных – можно представить в виде:

1) Логическая структура данных – способ организации связей между элементами данных.
2) Набор допустимых операций по манипулированию данными для принятой логической структуры.
3) Правила поддержки целостности или не противоречивости данных для выбранной логической структуры.

Модели данных

1) Иерархическая – на основе древовидная
2) Сетевая – на основе графов
3) Реляционная – на основе таблиц
4) Постреляционная – таблицы с возможностью вложения таблиц в таблицы
5) ООПешная – на основе принципов ООП
6) Документно-ориентированные

Иерархическая модель.

В СУБД иерархическая типа информация представлена в виде деревьев, узлами которых являются записи. Корневая запись для каждого экземпляра дерева обязательно должная содержать ключ с уникальным значением. Ключи не корневых записей должны иметь уникальные значения только в рамках данного экземпляра дерева. Каждая запись идентифицируется полным сцепленным ключом, то есть совокупностью ключей всех записей начиная от корневой по иерархическом пути.

Каждый экземпляр дерева называется групповым отношение.

Корневая – владельцем группового отношения, остальные дочерние

Эта модель реализует отношение 1:N, между исходной и дочерней записью.

В этих БД поддерживается целостность связей между владельцами и членами группового отношения.

Недостатки:

- Возникает дублирование информации при реализации отношения N:N
- Не эффективный поиск от узла к корню

Сетевая модель.

Состоит из множества записей, которые могут быть владельцами или членами групповых отношений.

Основное различие от иерархической модели состоит в том, что запись может быть членом более одного группового отношения.

Достоинство данной модели является отсутствие дублирования информации.

Недостаток — это ее чрезмерная сложность, что приводит к сложностям администрирования. Например, сложность восстановления поврежденных данных. Медленные некоторые запросы.

Реляционная модель.

В реляционной модели достигается более высокий уровень абстракции данных чем сетевой или иерархической.

Реляционная модель представляет средства описания данных на основе только их естественной структуры без потребности введения какой-либо дополнительной структуры для целей машинного представления. То есть представление данных в ней не зависит от способа их физической организации.

Основным логическим объектом для хранения данных является таблица. Для организации связей между данными различных таблиц используются их общие столбцы.

Пост реляционная модель.

Содержит в основе своей реляционную модель дополненную возможность создания вложенных таблиц.

Однако использование используется довольно редко т. к. СУБД не гарантирует быструю работу с такими таблицами.

ОО-модель.

Отсутствие строгой математической модели ОО-БД.

Наличие огромного наличие данных в реляционных БД и существенных затраты на их конвертацию.

Многие СУБД на сегодняшний день позиционируются как объектно-реляционные. В их основе лежит реляционная модель, но дополненная возможность создания пользовательских типов столбцов с поддержки принципов инкапсуляцией и наследования.

Лекция 4

Принципы реляционный баз данных:

1. Вся инфа логически представлена в виде таблиц;
2. Логический доступ к данным осуществляется: по имени таблицы, имени столбца и значение первичного ключа;
3. Null значение которые в реляционной логике как отсутствие данных (NULL != “ ” != 0)

В первичных и вторичных null не может быть, во внешнем ключе может быть, но при создании таблицы null можно запретить;

4) Метаданные или описание структуры базы данных должны, как и все остальные значения должны храниться внутри БД;
5) Правило ссылочной целостности – все значение внешнего ключа должны быть согласованными со значениями первичного ключа;
6) Ссылочная целостность при выполнении любых управлений с данными должна контролироваться СУБД;

В реляционной модели выделяется 3 части:

Структурная

Целостная

Манипуляционная

Структурная часть

Атрибут определяется как именованный, атомарный элемент данных. Множество допустимых значений атрибута называется доменом. Сравнивать между собой атрибуты можно лишь определённых на одинаковых доменах.

Схема отношения – это именованное множество упорядоченных пар (имя атрибута: имя домена). Соответственно, степенью называется мощность этого множества.

Схема базы данных – это набор именованных схем отношений.

Кортеж – соответствующий данной схеме отношений это множество упорядоченных пар (имя атрибута: значение атрибута), которое содержит по одному вхождению каждого имени атрибута принадлежащего схеме отношения. Значение атрибута является допустимых значений определённого в домене.

Отношение — это множество картежей, соответствующих одной схеме отношения.

- В отношении не может быть двух одинаковых картежей
- Картежи не упорядочены

Имена всех атрибутов в пределах одного отношения должны быть уникальными.

Реляционная БД – это набор отношений, имена которых совпадают с именами схем отношений в схеме БД.

Потенциальных кличем отношения – называют подмножество атрибутов отношения, которые удовлетворяют следующим свойствам:

1) Уникальность
2) Без избыточности - (никакое подмножество потенциального ключа, не является в свою очередь потенциальным ключем)

В случае, когда невозможно выделить атрибут в качестве первичного ключа, используется дополнительный атрибут – суррогатный ключ, который автоматические заполняется уникальными значениями и никогда не изменяется.

Все потенциальные ключи, которые не являются основным – являются потенциальными и не могут быть NULL.

Внешний ключе – такое подмножество атрибутов дочернего отношения, что для любого его не пустого значения обязательно найдется равное значение первичного ключа главного отношения.

Целостная часть

Целостность данных – механизм поддержания БД в непротиворечивом состоянии, соответствующим динамично развивающийся предметной области.

СУБД должна контролировать операции вставки, удалении и обновлении данных, и отказываться в выполнении операции если в ней производится попытка нарушить целостность БД.

Набор определенных правил, устанавливающий допустимость значений данных и их связей называют правилами или ограничениями целостности. Они задаются и хранятся в словаре данных БД как один из элементов определения таблицы.

Любое приложение, которое обращается к этой таблице должно придерживаться заданных правил. Их изменение может быть произведено только на уровне БД в целом, а не для отдельного приложения.

Требование Целостность сущности заключается в следующем:

1) Каждый объект реального мира должен быть четко идентифицирован, каждое отношение должно иметь первичный ключ
2) Поддержание целостности сущности обеспечивается средствами СУБД и осуществляется с помощью следующих ограничений: при добавлении кортежей проверяется уникальность ПК и отсутствие в них NULL значений, не позволяется изменения значений атрибутов входящих в ПК.
3) Все значения внешних ключей должны быть согласованны.

При выполнении операции вставки ссылочная целостность (СЦ) контролируется только в случае наличии значения внешних ссылающихся на другие отношения. Если они отсутствуют, то операция вставки отклоняется.

При выполнении операции удаления, простое удаление картежа приведет к наличию несогласованных значений внешних ключей во всех дочерних отношениях.

Запретить удаление картежа в родительском отношении при наличии хотя бы одного ссылающегося картежа.

При удалении картежа родительского отношения, каскадом удалять все ссылающиеся картежи, ссылающиеся на него дочернего отношения.

При удалении картежа родительского отношения установить во всех ссылающихся картежах NULL значения во внешних ключах, если они разрешены объявленной структурой нашей таблицы

При удалении картежа родительского отношения установить во всех ссылающихся картежах дочернего отношения – значения по умолчанию, которые заданы при создании БД

Манипуляционная часть

Реляционная алгебра и реляционные исчисления различаются степенью их процедурности.

Запрос, представленный на языке реляционной алгебры, может быть вычислен, используя элементарные алгебраические операции с учетом их приоритетов и скобок.

Формула реляционного исчисления только устанавливает условия, которым должны удовлетворять картежи результирующего отношения.

Реляционные операции:

Объединение: R3 = помещаются все кортежи, которые есть R1 ИЛИ R2

Пересечение R3 = кортежи, которые есть в R1 И R2

Разность R3 = кортежи, которые есть в R1, но нет в R2

Декартово произведение

Проекция (отбор атрибутов отношения) R2 = подмножество R1

Операция селекция – это любое логическое выражение или условие отбора картежей, в состав которого входят имена атрибутов, операции и константы. В R2 включаются все картежи из R1 для которых это условие истина. При этом отношение R2 может не содержать ни одного картежа R1 или совпадать полностью.

Соединение данная операция определена над двумя отношениями, у которых есть общие подмножество атрибутов. В отличии от декартового произведения склеиваются только те картеже R1 R2 у которых есть одно и тоже значение общего атрибута. Она эквивалента операции выборки из декартового произведения отношений R1 R2

В результат операции сведения не входит картеж R1 R2 для которых не находиться общий атрибут.

В SQL могут существуют три вида соединения левая правая и полная

В левом внешнем соединении, результат внутреннего соединения дополняется картежами отношения, стоящего слева

В левом внешнем соединении, результат внутреннего соединения дополняется оставшимся картежами отношения, стоящего справа

В полном внешнем соединении, результат внутреннего соединения дополняется все не связанные картежами дополненные неопределенными значениями.

Лекция 5

SQL делится на DDL, DML, DCL.

Описание синтаксиса языка SQL:

- Все ключевые слова прописываются - ПРОПИСНЫМИ, а имена, формируемые пользователем – строчными.
- Все не обязательные элементы команды заключатся в квадратные скобки.
- В тех случаях, где пробел может трактоваться не однозначно он заменяется нижним подчеркиванием.
- При пропусках части команды используем многоточие.

Схем 1

Объекты БД

- DATABASE –
- SCHEME – часть базы данных в пределах которой все имена создаваемых объектов должны быть уникальны.
- TABLE – мультимножество строк.
- INDEX – вспомогательный объект БД который служит для ускорения поиска данных. Хотя при этом замедляет операции вставки, удаления, обновление таблицы. Для потенциальных ключей индексы создаются автоматически.
- VIEW – именованный запрос на выборку данных который хранится в БД и выполняется на сервере при любом обращении к нему по имени, создавая при этом виртуальную таблицу с результатами выборки.
- PROCEDURE / FUNCTION – данные объекты пишутся на языке процедурного расширения языка SQL который дополняется его такими управляющими конструкциями языков высокого уровня как ветвления и циклы, и позволяется реализовать любые алгоритмы обработки данных. Хранимый код хранится на сервере и выполняется по запросу пользователя из клиентских приложений.
- TRIGGER – особый вид хранимой процедуры которая запускается автоматически при наступлении определенных событий в БД. Стандартно триггеры привязываться к DML.
- USER / ROLE – хранятся пользователи и их права на выполнение различных действий в БД. Данные объекты служат для разграничения доступа к информации многочисленным пользователям к БД. Могут выполняться действия, которые прописаны.

Типы данных

Числовые типы данных:

Десятичные: SMALLINT, INT, BIGINT, NUMERIC(p,s), FLOAT, DOUBLE, REAL, DOUBLE PROCESTION

Строки: CHAR[(n)], VARCHAR(n)

Дата и время: DATATIME, DATE, TIME, TIMESTAMP

Для больших объектов: BLOB – для хранения не только структурированной информация, а для бинарной.

Список ограничений

Ограничения позволяет задавать дополнительные условия проверки вводимых данных, которые СУБД проверяется автоматически.

Механизм ограничений позволяет поддерживать данных вне противоречивого состояния с соответствующем бизнес-правилам предметной области.

PRIMIRE KEY – при задании ограничение первичного ключа любое значение проверяется на уникальность.

UNIQUE – ограничение уникальности.

REFERENCES (имя таблицы) – внешний ключ который задается для. Для значений внешнего ключа автоматически выполняется проверка на существование равного значение первичного ключа главной таблицы.

ON DELETE

RESTRICT

CASCADE

SET NULL

SET DEFULT

CHECK – проверка некоторого условия при изменении данного столбца.

Ограничения таблицы.

Используется, когда они затрагивают несколько столбцов

PRIMARY KEY (список столбцов) составной первичный ключ

UNIQUE (список столбцов)

И т.д.

Создание таблиц на основе других таблиц.

В данном способе новая таблица будет содержать такие же имена столбцов и их типы, но не будет иметь ее ограничения.

Лекция 27.10

Декартово произведение – операция, при которой каждая строка одной таблицы склеивается с каждой строкой другой таблицы. Обычно таблицы, связывающие с помощью этой операции, не имеют общих столбцов.

```
select * from students, marks
```

Результат – возвратить нам кол-во строк, равное произведению строк 1 табл на 2 табл.

Внутреннее (или естественное) соединение таблиц.

Таким способом, можно соединять только таблицы, имеющие общие столбцы. При выполнении данной операции соединяются строки таблиц, которые имеют общие значения в столбце связи. Такой способ соединения таблиц является реализацией отношения «1 ко многим», когда в качестве столбцов связи используется первичный ключ главной таблицы и внешний ключ подчиненной. Те строки главной таблицы, для которых нет связанных в подчиненной, при внутреннем соединении не попадут в результат запроса.

1. Выборка из декартова произведения
```
select students.name, marks.mark
        from students, marks
        where students.cod_st += marks. cod_st
```
2. Непосредственное соединение
```
select students.name_st, marks.mark
        from students join marks on
        students.cod_st = marks.cod_st
```

Соединение таблиц необязательно должно выполняться по первичному и внешнему ключу. Любые 2 столбца, совпадающие по типу, могут быть использованы в условии соединения таблиц.

Выборка из декартова произведения, пример:

```
select st.cod_st, st.name, s.name_sub, m.mark
from students st, subjects s, marks m
where st.cod_st = m.cod_st and s.cod_st = m.cod_sub
```

Записать непосредственно:

```
select st.cod_st, st.name, s.name_sub, m.mark
from students st join marks on st.cod_st = m.cod_st
join subjects s.cod_sub = m.cod_sub
```

При внешнем соединении таблиц, в отличие от внутреннего, в результат выборки попадают не только все связанные строки обеих таблиц, но и строки одной из таблиц или обеих, для которых нет связанных в другой таблице. При недостающих присваивается значение NULL

Левое внешнее соединение, правое внешнее и

…

Вложенные запросы.

Это запросы, которые находятся в теле другого запроса. Они могут использоваться в следующих секциях:

- После from
- В секции where
- having
- После select

1. Тело вложенного запроса заключаем в скобки
2. Вложенные запросы могут содержать другие вложенные запросы, при этом выполнение запроса начинается с выполнения самого глубокого по вложенности запроса, заканчивая внешним

Пример использования вложенного запроса

```
select MIN(avg_mark), MAX(avg_mark)
from (select AVF(mark)  avg_mark from marks group by cod_st)
```

Второй пример: Как альтернатива соединения таблиц в запросах

```
select mark from marks
where cod_st in #нашли оценку
(select cod_st from students where
name_st = ‘Иванов’) #Нашли id иванова
AND cod_sub in
(select cod_sub from subjects
where name_sub = ‘Математика’) #нашли ID математики
```

Попробуем найти студентов, у которых есть максимальный или минимальный балл:

```
select name_st from students where cod_st
(select cod_st from marks groupby cod_st having
avg(mark) = (select MAX(avg mark) from
(select AVG(mark) from marks groupby cod_st)))
```

Использование в качестве вычисляемых столбцов в команде *select*

```
select cod_st, name_st,
(select AVG(mark) from marks
```

where cod_st students.cod_st)

from students big_mark

Вложенный запрос должен возвращать 1 значение для каждой строки внешнего запроса.

Вложенные запросы, которые не используют данные внешнего запроса, называются **некореллированными.** Такие вложенные запросы выполняются 1 раз и их результаты используются во внешнем запросе в качестве констант.

А те, которые используют данные внешнего запроса – **кореллированные.** Особенность в том, что они выполняются каждый раз для каждой строки внешнего запроса. Они сильно влияют на производительность, их стоит избегать.

Есть конструкции, которые тоже можно использовать в select`ах. Это ключевые слова *ALL* и *ANY.* Употребляются в секциях where и having.

ALL проверяет что все результаты запросы (например) равны чему-то. В нашем случае, у которых все оценки равны 4.

ANY – будут выбраны студенты, у которых есть хотя бы 1 четвертка.

Пример: Нужно вывести студентов, у которых 4ки

```
select cod_st, name_st from students
where ALL(select mark from marks
        where cod_st = students, cod_st) = 4
```

Это кореллированный запрос, приме как сделать его некор.:

```
select st.cod_st, st.name_st from students.st, marks.m
where st.code_st = m.cod_st
groupby m.cod_st
having MAX(mark) = 4 AND MIN(mark) = 4
```

Есть функция *EXIST* – служит для проверки результатов вложенного запроса на пустоту. Использовав ее перед вложенным запросом, если запрос что-то возвращает, то *EXIST* возвращает true.