Лекция №1

Общая теория реляционных БД

1.Базы Данных (БД)

2.Системы управлений БД (СУБД)

3.Типы БД

Под БД понимается совокупность данных организованных по определенным законам, принципам, правилам, для централизованного хранения обработки данных и получения необходимых данных пользователем.

Под СУБД будем понимать комплекс Программных средств, обеспечивающих централизованное хранение, обработку данных и выдачу пользователю необходимой информации из БД.

Все данные используемые в БД, подразделяются на три основных типа: простые, структурированные, ссылочные.

Простой тип данных не имеет внутренней структуры (не делимый тип, данные).

Структурированный тип данных содержит несколько простых типов и представляют собой данные одного объекта. Например, запись, массив и вектор.

Ссылочный тип представляет собой ссылку на другие типы данных.

БД делятся на два типа: распределённая и централизованная.

Распределённая БД размещена на нескольких компьютерах, между которыми может и не быть связи.

Централизованное БД сохраняется в одном в компьютере, к которому подключается пользователь.

Как централизовать распределённую БД в централизованную

Лекция №2

Тема: технологии обработки данных

Существует две технологии обработки данных:  
Тонкий клиент ТонК (клиент – сервер)

Толстый клиент ТолК (файл – сервер)

Технология ТонК предусматривает, что клиент (клиентское приложение) посылает запрос серверу (серверному приложению); сервер обрабатывает свои данные и результат показывает клиенту. То есть в данном случаи сервер выполняет роль хранилища и обработчика данных.

Для ТолК характерно что сервер выполняет роль лишь хранилища данных. А клиент сам обрабатывает данные. Клинт посылает запрос серверу; сервер высылает ему данные; клиент обрабатывает их; сервер записывает их вместо старых.

Преимущества одной технологии является недостатком другой технологии, в данном случаи можно говорить о требованиях аппаратной части сервера или клиента. Также много нюансов возникает в передаче данных.

В таблице последний критерий связан со следующим фактом. Предположим к серверу обратился клиент; сервер выдал ему данные; клиент начал их обрабатывать. В это время другой клиент обращается к серверу выдать эти же данные. Если сервер выдает эти данные, при окончании обработки данных первым клиентам, он передает серверу, сервер записывает вместо старых. Однако после этого второй клиент заканчивает обработку и передаёт данные серверу, сервер записывает их вместо старых, при этом обновления первого клиента стерлись. Такое не допустимо, поэтому сервер обязан заблокировать данные после выдачи данных первому клиенту от всех остальных клиентов, пока первый клиент не вернет обработанные данные. Это отрицательно влияет на скорость обработки данных.

ТолК может использоваться в тех случаях, когда определенными данными может работать конкретная группа пользователей; решены вопросы скорости передачи, защиты данных. В этом случаи для этой группы пользователей можно создать отдельное приложения, который будет работать в режиме ТолК, это даст возможность разгрузить сервер и оперативно работать определенной группе пользователей.

Тема: модели БД

Существует след. модели организации БД:

Иерархическая

Сетевая

Объектно-ориентированная

Реляционная

Иерархическая модель предусматривает строгое подчинение одних данных другим. Данные могут быть предками и потомками. Потомок создается внутри предка.

Потомок не может существовать без своего предка. Потомок имеет одного определённого предка. Потомок в свою очередь может быть предком только своих потомков. Данные образуется в виде деревьев, дерево может содержать любое кол-во поддеревьев.

Сетевая модель является расширением иерархической. Здесь между двумя объектами может существовать несколько связей. Каждая из этих связей иерархия может меняться:

1. Предок – потомок
2. Потомок-предок

Объектно-ориентированная модель предусматривает работу с так называемыми объектами: документами, электр. Таблицами, аудио и видео фалами.

Реляционная модель БД является наиболее гибкой и адекватной предметной области. Реляционной модели нет понятия предка потомка, все данные располагаются в таблицах между которых существует отношения (relationship), обеспечивающие целостность данных.

\*Во кладке сервис –> параметры -> конструкторы: стоит галочка не менять таблицу

Лекция №3

Тема: Иерархическая модель данных

Как было сказана выше представляет собой дерево состоящие из поддеревьев.

Существует понятия тип дерева, тип предка, тип потомка.

В рассматриваемой примере рассматривается тип дерева Отдел, внутри которого тип предка – Отдел, типы потомков – Начальник и Сотрудники.

В иерархической модели происходит в порядке сверку-вниз-слева-направо.

\*Почему начальник – старший сын, а сотрудник – млад. сын.

Если рассматривать конкретное дерево, конкретного предка то вводится тип экземпляр.

Например, отдел 310 Иванов является экземпляр типа начальник, Сидоров экземпляр типа сотрудники.

Иванов и Сидоров близнецы.

В иерархической модели возможны след. виды манипулирования данными:  
Перейти от одного дерево к другому

Перейти от одной записи в другую

Перейти от одной записи к другой в порядке обхода

Вставить новую запись в указанную позицию;

Удалить текущею запись.

Тема: Сетевая модель данных

Связь идет от предка к потомку (P->C), без предка не может существовать связи.

В сетевой модели существует след. виды манипулирования данными:

Найти конкретную запись в наборе однотипных записей

Перейти от предка к первому потомку по некоторой связи

Перейти к след потомку в некоторой связи

Перейти от потомка к предку по некоторой связи

Создать новую запись

Уничтожить запись

Модифицировать запись

Включить в связь

Исключить из связи

Переставить в другую связь

Тема: Модель «Сущность – связь»

Сущностью (Entity) называется любой объект в реале действительности, который по своим данным (атрибутам) может быть идентифицирован от всех остальных данных.

Например, студент, преподаватель, товар и т.д.

Связь (relationship) – некоторая ассоциация между двумя сущностями.

В связи с этим модель «сущность – связь» (ER-model).

ID – идентификатор уникальный.

Сущности одного определенного типа называются набором сущности (Entity set).

Модель ER не является дополнительная моделью как иерархическая и т.д.

Это модель является наиболее общей для всех моделей. Тип определяет сколько? кардинальность обязательно ли?

**Лекция №4**

В связи «один к одному» одной сущности слева может соответствовать не более одной сущности справа, и наоборот.

Для уяснения ER – модели строится след. схема:

группа

В прямоугольники помещаются сущность

В ромбик – связь

Является старостой

Сущности записываются в единственном числе в именительном падеже.

Студент

Для определения типа и кардинальности связи, необходимо ответить на следующие 4 вопроса:

1. Для одной сущности слева сколько соответствует сущностей справа?

Ответ на этот вопрос, является тип сущности справа

1. Для одной сущности справа сколько имеется связи с сущностью слева?

Ответ на вопрос, является тип сущности слева

1. Обязательно ли сущность слева должна стоят в связи с сущностью справа?

Ответ на вопрос будет кардинальность сущности справа.

1. Обязательно ли сущность справа должна стоят в связи с сущностью слева?

Ответ на вопрос будет кардинальность сущности слева.

Вопросы для примера:

1. Один студент является старостой скольких групп?

Ответ, только в одной группе, следовательно, тип сущности Группа равен 1

1. Сколько старост может быть в одной группе?

Ответ, только один староста, следовательно, тип сущности Студент равен 1

1. Обязательно ли у группы должен быть староста?

Ответ, обязан, следовательно, кардинальность сущности Студент 1

1. Обязательно ли студент должен быть староста?

Ответ, не обязан, следовательно, кардинальность сущности Группа 0

(1,1:0,1) тип и кардинальность связи. 1 и 3 кардинальность, 2 и 4 связь.

Тема: Тип связи «один-ко-многим» (1:n)

В этой связи одной сущности слева может соответствовать множество сущности справа и одной сущности справа может соответствовать только одна сущность слева.

Улица состоит из домов, дома находятся на улице.

состоит

дом

улица

1. Одна улица состоит из скольких домов?

Ответ, из множества, следовательно, тип сущности Дом равен n

1. Один дом входит во сколько улиц?

Ответ, в одну, следовательно, тип сущности Улица равен 1

1. Обязательно ли улица состоит из домов?

Ответ, да, следовательно, кардинальность сущности Дом равна 1

1. Обязательно ли дом стоит на улице?

Ответ, да, следовательно, кардинальность сущности Улица равна 1

(1,1:1,n)

Тип связи «Много-к-одному»

В этой связи одной сущности слева может соответствовать одна сущность справа и одной сущности справа может соответствовать множество сущность слева.

Футболисты входят в клуб

входит

Клуб

Футболист

(1,n:1,1) футболисты входят и обязанные входить в клуб, клуб состоит и обязан иметь футболистов.

Тип связи «Много-ко-многим»

В этой связи одной сущности слева может соответствовать несколько сущностей справа, и наоборот.

Например, «Пользователь – приложение»

Один пользователь может пользоваться многими приложениями, одно приложение может иметь множество пользователей.

Приложения не обязан иметь пользователей, пользователь не обязан иметь приложение

(0,n:0,n)

Существуют понятия сильная сущность и слабая сущность

Если какая-либо сущность существует в существовании другой сущности, то это сущность называется слабой, а другая - сильной.

**Лекция №5**

Например, из числа студентов создается рабочая группа для выполнения некоторой задачи. Как только задача выполнена, группа расформировывается, поэтому сущность задача является сильной, а рабочая группа слабой сущностью.

Выполнение

Контракт

Рабочая группа

Слабая сущность заключается в двойной прямоугольники, к сильной сущности ставится стрелка. Как правило, тип данный связи один к одному и кардинальность один.

Тема: Построение реляционной модели по ER – модели.

Рассмотрим, как из ER модели построить реляционную модель, то есть построить таблицы и определить связи/отношения между ними.

Разберем каждый тип связи отдельно.

В случаи если одна из сущностей вступает в связь только данной сущностью, и не вступает в другие связи, то это сущность может вступать как атрибут/ы данной сущности.

Например, студент имеет ИНН и паспорт, сущности ИНН и Паспорт не вступает в другие связи то они входят как атрибуты сущности Студент.

|  |
| --- |
| ID |
| Name |
| … |
| INN(Not Null) Unique |
| … |

Если же обе сущности вступают в связь с другими сущностями, то это связь в реляционной модели реализуется с помощью двух таблиц, где в каждой таблице помещаются сущности, в одну из таблиц добавляет столбец, который является внешним ключом к другой таблице, в строку ставится ограничение уникальности, в зависимости от кардинальности ставится Not Null.

Тема: «Много к одному / один ко многим»

Реализуется с помощью двух таблиц, в каждой таблице помещается значение сущностей. В таблице с типом n добавляется поле, которая будет ссылаться на первичный ключ к сущности с типом 1.

Тема: «Много ко многим»

|  |
| --- |
| ID |
| Name |
| … |

Для реализации этой связи создаются 3 таблицы, в двух их которых помещаются связи, третья представляет собой связи между ними.

|  |
| --- |
| ID |
| Name |
| … |
| Group |

|  |
| --- |
| ID |
| Student |
| … |
| Sport |

Назовем ключом сущности один или несколько атрибутов, однозначно определяющих данную сущность. Если ключ состоит из одного атрибута, то он называется простым, а если из нескольких, то составным.

Например, для сущности студент, преподаватель, группа и т.п. имеют простые ключи, а в таблице, например, “ProgressInStudy”, “Lectures” имеют составной ключ. То есть не всегда первичный ключ является ключом сущности.

В SQLServer нужно обеспечить уникальность ключа сущности. Первичный ключ обеспечивает уникальность автоматически, а для составных кличей добавляется уникальный индекс,

**Лекция №6**

**Тема: Реляционная БД (РБД)**

Декартово произведение.

Данные два множество: A(a1,a2,a3 … an) и B(b1,b2,b3 … bn)

Декартовым произведением этих множеств является такое множество C которая получена попарным взаимодействием элементов множеств A и B.

С = A\*B = (a1\*b1, a1\*b2 … a1\*bn, a1\*b1, a2\*b1, … an\*b1 … an\*bn)

A(n), B(m), C(n\*m)

В РБД все сущности заключаются в таблице, при это будем называть кортежем множество атрибутов одной сущности.

Атрибут – это одно определённое свойство сущности.

Несколько атрибутов, определяющие одно общее свойство сущности может быть объединены в один домен.

Отношение (retio) называется декартово произведение атрибутов на кортеж

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| целое | строка | | целое | |
| номер | имя | должность | деньги | |
| Табельный номер | Имя | должность | оклад | Премия |
| 2934  2935  2936 | Иванов  Петров  Сидоров | Инженер  Вед. Инженер  бухгалтер | 112  144  92 | 40  50  35 |

Реляционная модель предполагает хранения сущности в таблицах, между которыми существует отношения. Отношения связывает PK с FK другой таблицы.

Отношения обладает следующими условиями:

Каждый картеж уникален. Для этого каждое отношение сущность должно иметь ключ отношений.

Картежи не упорядочены.

Атрибуты не упорядочены. Внутри одного отношения имя атрибута должно быть уникальным.

Атрибуты не делимы (атомарный). Не делимость понимается в плане семантической неделимости (целостное).

Советуется придерживаться принципам минимальности в выборе ключа отношения. То есть атрибуты входящие в ключ отношения не должны быть лишними в этом ключе. Например, в таблице “ProgressInStudy” нельзя включать в составной ключ поле Балы.

**Нормализация отношений (1НФ- Первая нормальная форма)**

Выше было указано что отношение удовлетворяет 4 условиям, однако в некоторых случаях эти условия не достаточны для корректной работы БД.

Определение: Отношение находится в 1НФ тогда и только тогда, когда (Т. и Т.Т.К) отношение удовлетворяет 4 условиям:

1) картежи уникальный

2)Картеж не упорядоченный

3)атрибуты не упорядочены и имеет уникальное имя

4) значение атрибутов атомарные.

Уникальность картежей обеспечивает ключ отношения.

Рассмотрим след. пример.

Пусть отношение хранит инфо о студентах, группах, проектах в которых участвует студенты.

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| Н\_Студ | Фамилия | Н\_Группы | группа | Н\_Про | проект | Номер задания |
| 2934 | Иванов | 1 | Пи-1-22 | 1 | космос | 1 |
| 2934 | Иванов | 1 | Пи-1-22 | 2 | климат | 1 |
| 2936 | Сидоров | 1 | Пи-1-22 | 1 | космос | 2 |
| 2934 | Иванов | 2 | Пи-2-22 | 1 | космос | 1 |

{Н\_СТУД, Н\_ПРО}- ключ отношения

{44} Данные отношения имеют составной ключ( сложный) номер студена и номер проекта, поскольку требования предметной области один студент не может дважды участвовать в одном проекте. В следствии этого выполняется 1НФ. Видим что остальные также выполняются поэтому данное отношение входит 1НФ.

С одной стороны из данного отношения мы получаем несколько видов инфо: данные студентов, данные групп, данные проектов, группа студента(какой студент в какой группе),участие студентов в проекте.

С другой стороны данное отношение приводит к след. трудностям: Например можно добавить существующего или имеющегося студента с другим номером, тоже самое с группой и проектами, можно для студента ввести не его группу.

Кроме этого данное отношение имеет аномалии:

1. Аномалия обновления. В случае если студент меняет фамилию , группа или проект меняет свое наименование, то это изменение необходимо провести во всех кортежах где они участвуют. При ручном изменении возможны ошибки. Поэтому для разрешения данной аномалии необходимо разработать доп. процедуры автоматически выполняющие данные обновления. То есть данная аномалия приводит к доп. трудностям.
2. Аномалия вставки. Предположим, что в отношение добавляется новый кортеж со студентом, который пока не участвует в проектах. Данный кортеж нельзя добавить в отношения поскольку атрибут номер проекта входит в сложный ключ и не может быть пустым. То есть студент есть, а в базе нет. Данное аномалия приводит некорректной работе БД.
3. Аномалия удаления. При удалении кортежа с одной сущностью мы можем потерять другую сущность. Например, при удаление сидорова теряем инфо о группе пи-2-22; при удалении проекта Космос теряем другие проекты с космосом. То есть аномалия удаление приводит к некорректной работе БД

Следовательно, условия 1НФ не достаточны для корректной работы БД. Требуется нормализация отношений.

Функциональная зависимость

Определение: Пусть R- отношение, X,Y – атрибуты данного отношения, от X,Y = R . Говорят что Y функционально зависит от X, если для A кортежей r1,r2=R из равенства r2X = r1X следует, что r2Y =r1Y. Символически функциональная зависимость записываться X->Y.s

В нашем примере мы имеем след функциональный зависимости

Н\_студ – ФАМ, Н\_про – ПРОЕКТ, Н\_гр – Группа

Все не ключевые атрибуты отношения зависят от ключа отношения, последние функциональные зависимости вытекают из того, что студент учится в одной определенной группе.

Отличие функциональной зависимости атрибутов от матем. понятия

Рассмотрим функцию у=F(x) а конкретно y=x^2 если х=-3 то у=9 то есть для любого значения аргумента х0 соответствует одно определенное значение функции у0. И это зависимость абсолютная не зависящее от обстоятельство времени.

Пусть студент Бактыбеков имеет номер 518, то есть имеем 518 – Бактыбеков. В случае если студент поменяет свое фамилия, тогда функциональное зависимость не зависимо. Зависимость относительно.

Причиной указанной 1НФ выше аномалий является избыточность данных в следствии того, что в отношениях хранится разнородная инфо. (в одном отношении хранятся несколько сущностей). Для избавления от аномалий требуется нормализация отношений.

Отношение R находится 2НФ (Т. и Т.Т.К) отношение находится в 1НФ и нет не ключевых атрибутов, зависящих от части сложного ключа.

Наше отношение находится в первой нормальной форме но не находится в 2НФ , поскольку есть зависимости не ключевых атрибутов которые зависят от части ключа.

Для того чтобы разрешить удовлетворение 2НФ необходимо произвести декомпозицию отношения на несколько отношений таким образом чтобы эти зависимые не ключевые атрибуты вместе со своими определителями детерминантами вынеслись в отдельные отношения.

Следовательно, не ключевые атрибуты ФИО, номер группы группа вместе со своими определителями выносятся в отдельное отношение, а проект со своим определителем номер проекта в отдельное

|  |  |  |  |
| --- | --- | --- | --- |
| Н\_Студ | Фамилия | Н\_Группы | группа |
| 2934 | Иванов | 1 | Пи-1-22 |
| 2936 | Сидоров | 1 | Пи-1-22 |
| 2934 | Петров | 2 | Пи-2-22 |

|  |  |  |
| --- | --- | --- |
| Н\_Студ | Н\_Про | Номер задания |
| 2934 | 1 | 1 |
| 2934 | 2 | 1 |
| 2936 | 1 | 2 |
| 2934 | 1 | 1 |

Отношения Студ\_группы и проекты находятся в 2НФ поскольку имеет простой ключ.

Отношение студенты и проекты находятся в 2НФ поскольку не ключевой атрибут номер задание не зависит от частей сложного ключа номер проекта и студента. Если проверить по отдельности, то не получится.

При анализе полученных отношений выесняется что некоторые аномалия разрешены (Аномалия вставки), но какие то остались (аномалия удаления)и частично аномалия обновления.

Причиной указанной 2НФ выше аномалий является избыточность данных в следствии того, что в отношениях хранится разнородная инфо. (в одном отношении хранятся несколько сущностей). Для избавления от аномалий требуется нормализация отношений.

Рассмотрим следующий пример .Пусть БД содержит инфо о поставках деталей различными поставщиками.

|  |  |  |  |
| --- | --- | --- | --- |
| Номер поставщика  PNUM | Наименования поставщика PNAME | Номер детали  DNUM | Постовляемое количество  VOLUME |
| 1 | Фирма 1 | 1 | 100 |
| 1 | Фирма 1 | 2 | 200 |
| 1 | Фирма 1 | 3 | 300 |
| 2 | Фирма 2 | 1 | 150 |
| 2 | Фирма 2 | 2 | 250 |
| 3 | Фирма 3 | 1 | 1000 |
|  |  |  |  |

При анализе этого отношения выясняем что отношение находится в 1НФ. В следствии того что название организации, как и номер организации является ее идентификатором. Поэтому мы имеем функциональные зависимости и тогда PNUM,DNUM и PNAME,DNUM равнозначные ключи отношения.

Следовательно, атрибуты PNUM, DNUM и PNAME входят в ключи отношения и атрибут VILUME единственный не ключевой атрибут.

Отношение находится в 2НФ так VОLUME не зависит PNUM, DNUM и PNAME.

Отношение находится в 3НФ так как находится в 2НФ и имеет 1 не ключевой атрибут.

В примере, рассмотренном выше приведение к 3НФ способствовало разрешению всех аномалий. Однако в данной отношении мы опять видим этим аномалия.

Аномалия обновления «если организация поменяет свое название, то нужно везде менять имя»

Аномалия вставки если мы добавим новую организацию невозможно пока не совершить поставку деталей. Поскольку поле DNUM не может быть пустым как часть сложного ключа, то есть аномалия приводит не корректной работе БД.

Аномалия удаления – при удалении детали 1 удаляется фирма (организация)теряется информация. Аномалия не разрешима и приводит не корректной работе БД.

Поскольку мы имеем 2 равнозначных ключа отношения, то имеем

Определение находится в НФБК т.и.т.т.,к отношение находится в 3НФ и все детерминанты (определители) функциональных зависимостей являются потенциальными ключами отношений.

Наше отношение находится в 3НФ но не находится в НФБК поскольку мы имеем функциональную зависимость PNAME и PNUM, но ни PNUM и ни PNAME не являются ключами отношений. Для того чтобы перейти к НФБК необходимо произвести декомпозию отношений таким образом чтобы эти зависимые атрибуты со своими определителями, которые не являются ключом вынеслись в отдельные отношения. Определитель остается в старом отношении.

|  |  |
| --- | --- |
| Номер поставщика  PNUM | Наименования поставщика PNAME |
| 1 | Фирма 1 |
| 2 | Фирма 2 |
| 3 | Фирма 3 |

Отношение «поставщики»

|  |  |  |  |
| --- | --- | --- | --- |
| Номер поставщика  PNUM | Номер детали  DNUM | Поставляемое количество  VOLUME | N |
| 1 | 1 | 100 | 1 |
| 1 | 2 | 200 | 2 |
| 1 | 3 | 300 | 3 |
| 2 | 1 | 150 | 4 |
| 2 | 2 | 250 | 5 |
| 3 | 1 | 1000 | 6 |

Отношение «поставки»

Отношение «поставщики» находятся в 3НФ и определитель PNUM является ключом отношения.

Отношения «поставки» также находится в НФБК поскольку находится в 3НФ и не имеет функциональных зависимостей.

При этом разрешены все указанные выше аномалия.

Не всегда ключ отношения является первичным ключом

В отношение поставки PNUM b DNUM являются ключом отношения, но не являются первичным ключом.

Советуется чтобы каждая таблица имела первичный ключ.

**Привести примеры НФ и НФБК из заданий по лаб работам.**