Управление данными

Лекции

- "Ну и запросы у вас...“ – сказала база данных и повисла

ВВЕДЕНИЕ

- Управление ресурсами в общем случае означает способность к выполнению над ними процедур
  - планирования
  - распределения
  - поддержки и сохранения
  - экономного расходования
  - правильного потребления и интеграции (возможности использования в различных целях).
- Возможность моделирования в памяти ЭВМ любого хранилища данных позволяет рассмотреть данные как один из видов ресурсов.

Данные, информация

- Данные – это информация, фиксированная в определенной форме, пригодной для последующей обработки, хранения и передачи.
- Под информацией понимают любые сведения о каком-либо событии, сущности, процессе, являющиеся объектом некоторых операций восприятия, передачи, преобразования или использования.

Области использования ВТ

- выполнение математических вычислений, которые невозможно провести вручную
- использование средств ВТ в автоматизированных информационных системах
  - основные функции средств ВТ
    - поддержка надежного хранения информации
    - выполнение специфических для данного приложения преобразований информации
    - предоставление пользователям удобного и легко осваиваемого интерфейса

Плоские файлы

- Плоский файл — это именованный набор данных на внешнем носителе. Сама ОС никакой структурой плоский файл не наделяет и трактует его просто как набор байт.
- Недостатки:
  - Все предположения о структуре записей файла делает прикладная программа. Ошибка приведет к непредсказуемым последствиям.
  - Невозможно заблокировать на чтение/модификацию отдельную запись файла средствами ОС
  - Для поиска нужной записи по условию придется считывать все записи последовательно с начала файла

Индексно-последовательные файлы

- При создании файла создается индекс, который служит для быстрого доступа к записям по значению ключевого поля.
- Делит файл на записи и поддерживает индекс сама ОС, а не прикладная программа
- Недостатки:
  - В одном файле могут храниться только сущности одного вида.
  - Отсутствие средств для описания взаимосвязей между сущностями, хранящимися в различных файлах.

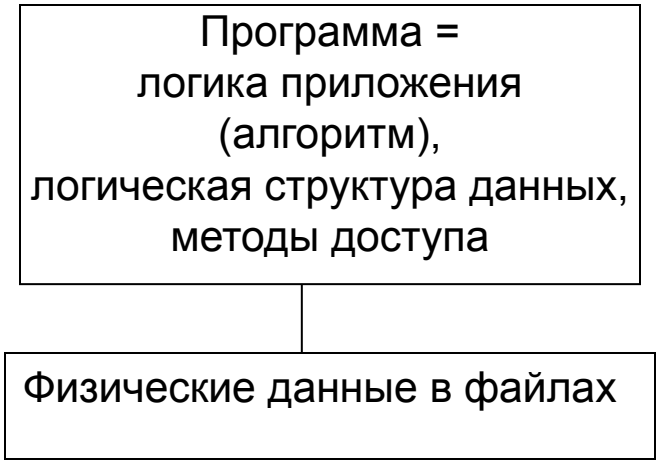
Достоинства и недостатки файловых систем

- ☺ Возможность легко реализовать структуры хранения, наиболее естественно соответствующие специфике соответствующей прикладной области.
- ☺ Высокая скорость обработки данных.

- ☹ *Избыточность данных*
- ☹ *Возможность наличия противоречивости данных*
- ☹ *Частое нарушение целостности данных*
- ☹ *Трудно получить ответ на интегрированные запросы*

Понятие о базе данных

- Структура программы при файловой организации данных
  Структура программы при использовании технологии баз данных



- СУБД – DBMS – DataBase Management System

Ключевые моменты концепции баз данных

- Информация не скрыта в сочетании «файл-программа», а хранится явным образом в БД. БД ориентирована на интегрированные запросы, а не на одну задачу.
- Возможность выделения по запросу из всех данных, хранящихся в БД, только необходимых и в требуемой форме (структуре и форматах).

Определение БД

- 1) База данных – предназначенная для машинной обработки совокупность интегрированных данных, т.е. данных, предназначенных для удовлетворения нужд многих пользователей.
- 2) База данных – совокупность данных, организованных по определенным правилам, предусматривающим общие принципы описания, хранения и манипулирования данными, независимо от прикладных программ (ГОСТ 34,003 ЕКС АС)
- 3) База данных – совокупность взаимосвязанных, хранящихся вместе данных при наличии такой минимальной избыточности, которая допускает их использование оптимальным образом для одного или нескольких приложений.
- Данные запоминаются так, чтобы они были независимы от использующих их программ. Для добавления новых или модификации существующих данных, а также для поиска данных в БД применяется общий способ.

- БД
  - Однопользовательская
  - Многопользовательская
- Концепции сетевой обработки данных
  - концепция файл-сервер – предполагает наличие ЭВМ, выделенной под файловый сервер, на котором находится ядро сетевой ОС и централизованно хранимые файлы. Запрошенные пользователем данные транспортируются с файлового сервера на рабочие станции, где выполняется их обработка.
  - концепция клиент-сервер – подразумевает разделение функций обработки данных между рабочей станцией и машиной-сервером. По запросу пользователя осуществляется поиск на сервере БД и обработанные данные транспортируются по сети от сервера к клиенту.

Определение СУБД

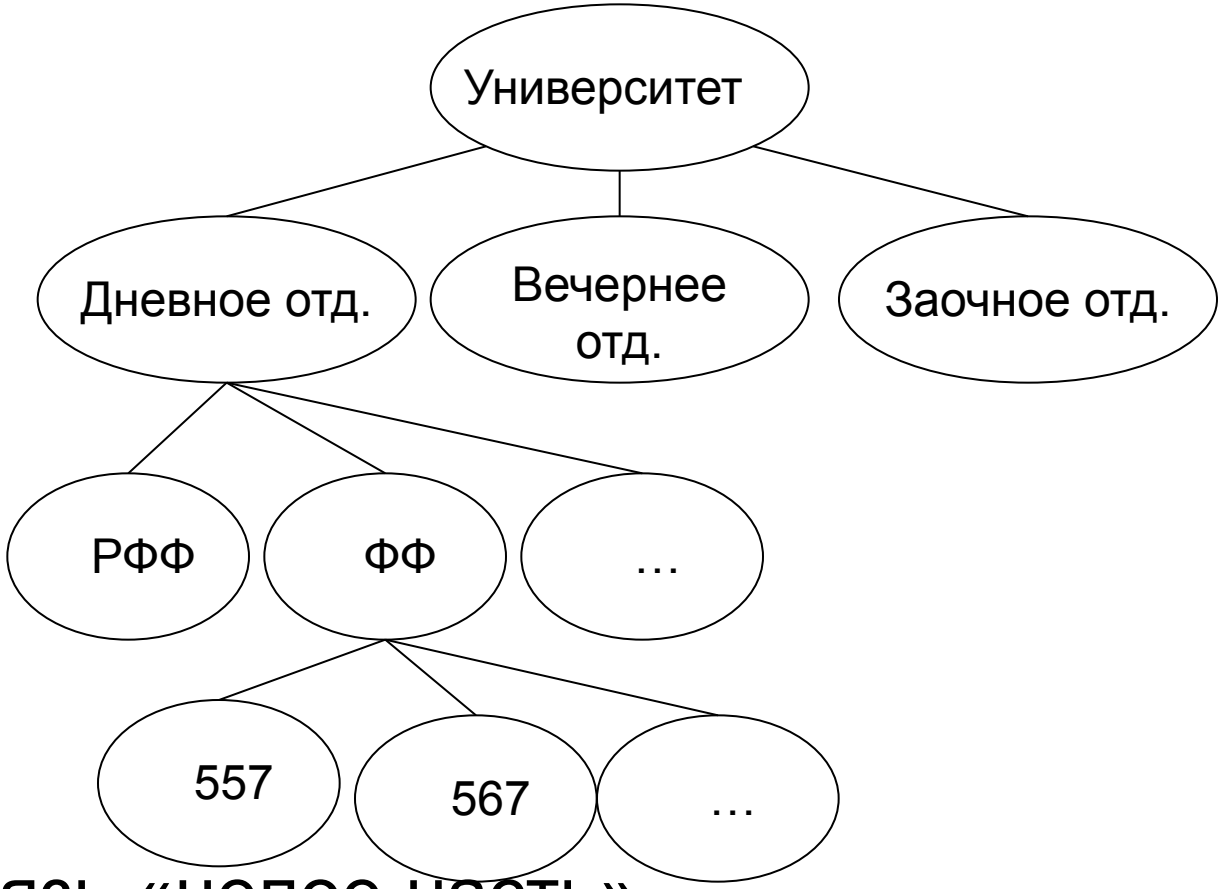
- СУБД – специализированное программное обеспечение, с помощью которого реализуется централизованное управление данными, хранимыми в базе, доступ к ним и поддержание их в состоянии, соответствующим состоянию предметной области.
- Свойства:
  - Поддержание логически согласованного набора файлов
  - Обеспечение языка манипулирования данными
  - Восстановление информации после сбоев
  - Обеспечение параллельной работы нескольких пользователей

МОДЕЛИ ДАННЫХ

- Модель данных – совокупность структур данных и операций их обработки.
- Классические модели:
  - Иерархическая
  - Сетевая
  - Реляционная
- Новые модели:
  - Постреляционная
  - Многомерная
  - Объектно-ориентированная

Иерархическая модель

- Корень
- Ветви
- Листья



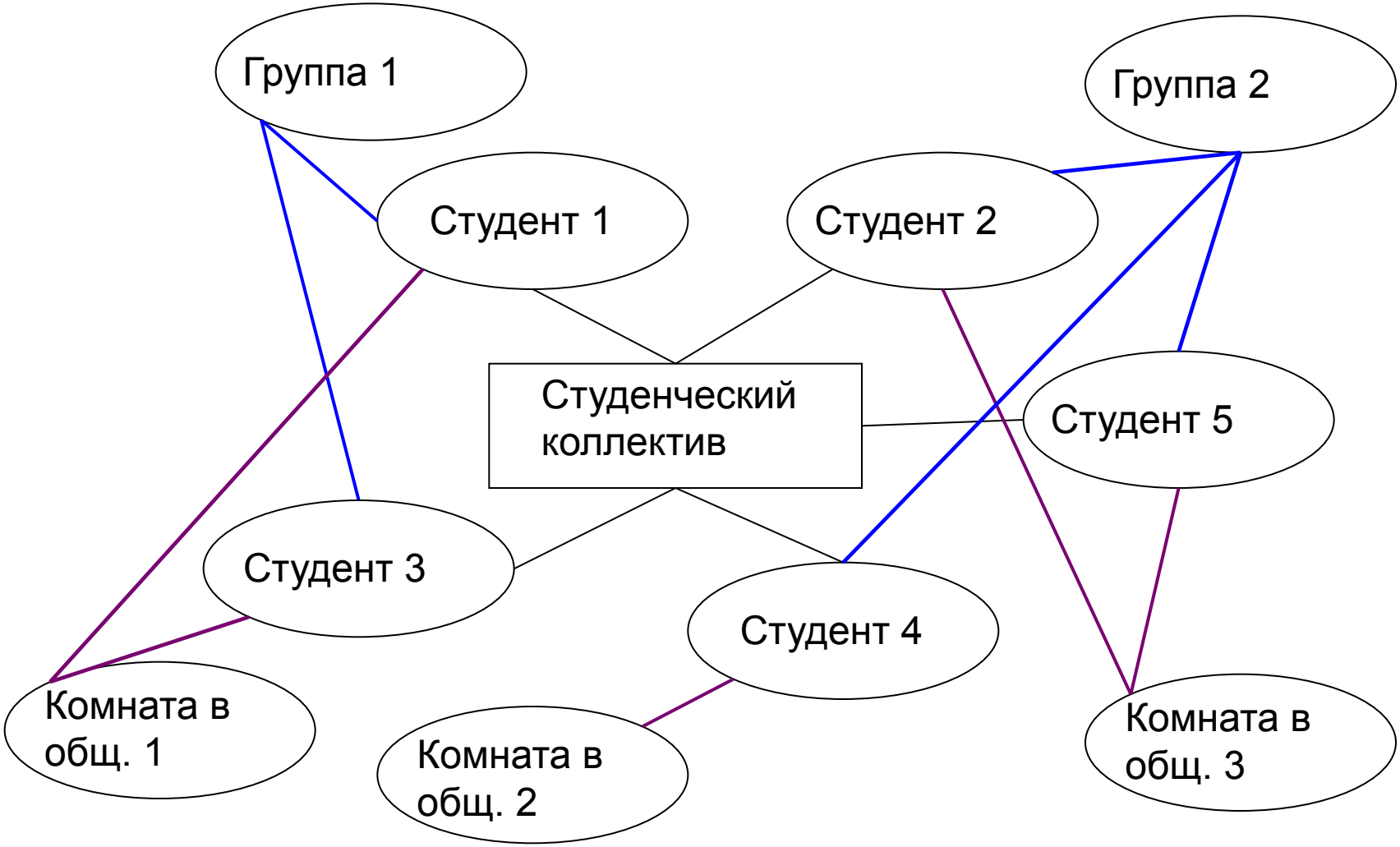
Логическая связь «целое-часть»

- Эффективное использование памяти
- Удобна для работы с иерархически упорядоченной информацией

- Громоздкость для обработки информации со сложными логическими связями
- Сложность понимания для обычного пользователя

- Примеры СУБД: IMS, PC/Focus, Ока, ИНЭС, МИРИС.

Сетевая модель



- Эффективное использование памяти
- Оперативность
- Бóльшая возможность образования произвольных связей

- Высокая сложность схемы БД
- Сложность для понимания обычным пользователем
- Слабый контроль целостности

- Примеры СУБД: IDMS, db_VistaIII, СЕТЬ, СЕТОР, КОМПАС

Достоинства и недостатки СУБД первого поколения

- ☹ *Сложность использования*
- ☹ *Необходимость знания физической организации БД*
- ☹ *Отсутствие средств автоматизации проектирования БД*
- ☹ *Очень высокая стоимость*

- ☺ Наличие развитых средств управления данными во внешней памяти на низком уровне
- ☺ Возможность построения эффективных прикладных систем вручную
- ☺ Возможность экономии памяти за счет совместного использования объектов

Реляционная модель

СТУДЕНТ

№ личного дела	ФИО	Дата Рожд	…
16493 16597 16695 …	Иванов Петров Сидоров		

СЕССИЯ

№ личного дела	Оценка 1	Оценка 2	…
16493 16597 16695 …			

- Простота, понятность для пользователя
- Удобство реализации на ЭВМ

- Сложность описания иерархических и сетевых связей

- Примеры СУБД: dBaseIII Plus, dBaseIV, FoxPro, FoxBase, Paradox, Access, Clarion, Ingres, Oracle (до 7.x реляционные, с 8.x объектно-реляционные), ПАЛЬМА, HyTech

Постреляционная модель

- Постреляционная модель данных - расширенная реляционная модель, снимающая ограничение неделимости данных, хранящихся в записях таблиц.
- Постреляционная модель допускает многозначные поля. Набор значений многозначных полей считается самостоятельной таблицей, встроенной в основную таблицу.

а) Реляционная модель

Номер накладной	номер покупателя
0373	8723
8374	8232
7364	8723

название товара	количество
Сыр	3
Рыба	2
Лимонад	1
Сок	6
Печенье	2
Йогурт	1

б) Постреляционная модель

Номер накладной	покупателя	название товара	количество
0373	8723	Сыр	3
		Рыба	2
8374	8232	Лимонад	1
		Сок	6
		Печенье	2
7364	8723	Йогурт	1

Постреляционная модель

- Достоинства:
  - высокая наглядность
  - повышение эффективности обработки информации.
- Недостаток:
  - Сложность решения проблемы обеспечения целостности и непротиворечивости хранимых данных.
- Примеры СУБД:
  - uniVers, Bubba, Dasdb.

Многомерная модель

- Многомерные СУБД являются узкоспециализированными СУБД, предназначенными для интерактивной аналитической обработки информации.

а) реляционная модель

Модель	Месяц	Объем
«Жигули»	июнь	12
«Жигули»	июль	24
«Жигули»	август	5
«Москвич»	июнь	2
«Москвич»	июль	18
«Волга»	июль	19

б) двумерная модель

Модель	Июнь	Июль	Август
«Жигули»	12	24	5
«Москвич»	2	18	
«Волга»		19	

Многомерная модель

- Достоинства:
  - высокая наглядность и информативность;
  - удобство и эффективность аналитической обработки больших объемов данных, связанных со временем.
- Недостаток:
  - громоздкость для простейших задач обычной оперативной обработки информации.
- Примеры СУБД:
  - Essbase(Arbor Software), Media Multi-matrix (Speedware), Oracle Express Server (Oracle), Cache (InterSystem).

Объектно-ориентированная модель

- уктура ентированной БД едстав ева, узлами отся об ва объектов стандар и типом, ым поль ss).
- class е ляющийся экземпляром соответствующего класса объект-эк а считается по объекта, пределен как с
- Объект-э са принадлежит классу и одителя. Родов в БД обра ю иерархию об

БИБЛИОТЕКА

Свойство	тип
Район	string
АБОНЕНТ	class
КАТАЛОГ	class
Билет	string
…	

АБОНЕНТ

Свойство	тип
билет	string
имя	string
адрес	string
…	

КАТАЛОГ

Свойство	тип
удк	string
название	string
автор	string
КНИГА	class
…	

КНИГА

Свойство	тип
номер	string
стеллаж	string
издание	string
…	

Основные понятия ОО БД

- Инкапсуляция ограничивает область видимости свойства пределами того объекта, в котором оно определено.
  - Н-р телефон абонента и автора книги
- Наследование, наоборот, распространяет область видимости свойства на всех потомков объекта.
  - Н-р билет
- Полиморфизм – способность одного и того же программного кода работать с разнотипными данными. (Допустимость в объектах разных типов иметь методы с одинаковыми именами)
  - Объекты класса КНИГА, имеющие разных родителей из класса КАТАЛОГ, могут иметь разный набор свойств

Объектно-ориентированная модель

- Достоинство
  - возможность отображения информации о сложных взаимосвязях объектов. Объектно-ориентированная модель данных позволяет идентифицировать отдельную запись базы данных и определять функции их обработки.
- Недостатки
  - высокая понятийная сложность,
  - неудобство обработки данных,
  - низкая скорость выполнения запросов.
- Примеры:
  - POET (POET Software), Jasmine (Computer Associates), Versant (Versant Technologies), O2 (Ardent Software), ODB-Jupiter (научно-производственный центр «Интелтек Плюс»).

Основные функции СУБД

- непосредственное управление данными во внешней памяти
- управление буферами оперативной памяти
- управление транзакциями
- протоколирование (журнализация)
- поддержка языков баз данных.

Непосредственное управление данными во внешней памяти

- Обеспечение необходимых структур внешней памяти для хранения
  - данных, непосредственно входящих в БД,
  - данных для служебных целей (например, индексы для ускорения доступа к данным)

Управление буферами ОП

- Объем информации в БД достаточно велик и превышает доступный объем ОП
- Время доступа к ОП, существенно меньше, чем к внешней памяти

- Система будет работать со скоростью устройства внешней памяти

- Увеличения скорости обмена данными можно достичь, используя буферизацию данных в ОП
- в СУБД обычно поддерживается собственный набор буферов ОП с собственным механизмом замены буферов

Управление транзакциями

- *Транзакцией* называется последовательность операций над БД, рассматриваемых СУБД как единое целое.
  - Если все операции успешно выполнены, то транзакция также считается успешно выполненной и СУБД *фиксирует* все изменения данных, произведенные этой транзакцией.
  - Если хотя бы одна операция транзакции заканчивается неудачей, то транзакция считается невыполненной и производится *откат* — отмена всех изменений данных, произведенных в ходе выполнения транзакции, и возврат БД к состоянию до начала выполнения транзакции.
- Свойства
  - Атомарность
  - Сериализуемость
  - Долговечность

Протоколирование (журнализация)

- Надежность хранения: СУБД должна быть в состоянии восстановить последнее согласованное состояние БД после любого аппаратного или программного сбоя.
- Аппаратные сбои:
  - *мягкие* сбои связаны с внезапной остановкой работы компьютера.
  - *жесткие* сбои характеризуются потерей информации на носителях внешней памяти.
- Программные сбои – последствия ошибок в программах:
  - в самой СУБД = мягкий аппаратный сбой
  - в пользовательской программе.

Протоколирование (журнализация)

- Для восстановления информации в БД необходимо иметь *журнал изменений* БД.
- *Журнал* – особая часть БД, недоступная пользователям СУБД и поддерживаемая с особой тщательностью, в которую поступают записи обо всех изменениях основной части БД.
- Стратегия «упреждающей» записи в журнал (протокол Write Ahead Log — WAL)
- Восстановление
  - индивидуальный откат транзакции
  - мягкий сбой
  - жесткий сбой

Поддержка языков баз данных

- Специализированные языки:
  - *язык определения схем данных*
    (Schema Definition Language, SDL)
    определение логической структуры БД
  - *язык манипулирования данными*
    (Data Manipulation Language, DML)
    заносить данные в базу, удалять, модифицировать, выбирать существующие данные.
- Интегрированные языки:
  - SQL (Structured Query Language) – структурированный язык запросов
  - QBE (Query By Example) – язык запросов по образцу

Реляционная модель данных

- Эдгар Кодд, 1970 г., статья «A Relational Model of Data for Large Shared Data Banks».
- Достоинства:
  - возможность сравнительно просто моделировать большую часть распространенных предметных областей
  - наличие простого и в то же время мощного математического аппарата, опирающегося главным образом на теорию множеств и математическую логику
  - возможность манипулирования данными без необходимости знания конкретной физической организации баз данных во внешней памяти.
- Недостатки:
  - ограниченность при использовании в нетрадиционных областях, в которых требуются предельно сложные структуры данных
  - нет достаточных средств для представления смысла данных

Базовые понятия

- тип данных
- домен
- атрибут
- кортеж
- ключ
  - Пример отношения СТУДЕНТЫ

№_студ_билета	Имя	Дата_ рождения	Курс	Специальность
23980282	Алексеев Д. А.	12.03.1982	2	Биология
22991380	Яковлев Н. В.	25.12.1979	4	Физика
22657879	Михайлов В. В.	29.02.1979	5	Математика
24356783	Афанасьев А. В.	19.08.1983	1	Иностранный язык
24350283	Кузнецов В. И.	03.10.1982	1	Физика
23125681	СмирновА. Д.	26.03.1981	3	История

Тип данных

- Понятие тип данных в реляционной модели данных эквивалентно соответствующему понятию в алгоритмических языках.
  - целочисленные
  - вещественные
  - строковые
  - денежные
  - временные (дата и/или время)
  - двоичные
  - гиперссылки.

Домен

- *Доменом* называется множество атомарных значений одного и того же типа.
- *Атомарное* (неразложимое) для данной модели значение данных — это наименьшая единица данных реляционной модели.
- Пример

Домен	**Базовый тип**	**Ограничение**
Имена Специальности	строковый	строки, которые могут изображать имя и название специальности
Даты_рождения	временной	допустимый диапазон дат рождения студентов
Номера_курсов Номера_студ_ билетов	целочислен-ный	целые числа, которые могут обозначать номер курса университета (обычно от 1 до 6) и номер студенческого билета (обязательно положительное число)

- **Атрибуты** — столбцы отношения.
  - Им присваиваются имена, по которым к ним затем производится обращение.
- **Схема отношения** — список имен атрибутов отношения с указанием имен доменов (или типов, если домены не поддерживаются).
  - Схема отношения СТУДЕНТ:

    СТУДЕНТ (№_студ_билета Номера_студ_билетов,
    Имя Имена,
    Дата_рождения Даты_рождения,
    Курс Номера_курсов,
    Специальность Специальности)
- **Степень отношения** — это число его атрибутов.
  - унарное отношение
  - бинарное отношение
  - тернарное отношение
  - ...
  - n-арное отношение.
- **Схема базы данных** — множество именованных схем отношений.

Кортеж

- **Кортеж**, соответствующий данной схеме отношения, представляет собой множество пар (имя атрибута, значение), которое содержит одно вхождение каждого имени атрибута, принадлежащего схеме отношения.
- **Степень кортежа** — число элементов в нем — совпадает со степенью соответствующей схемы отношения.
- Отношение — множество кортежей, соответствующих одной схеме отношения.
- ***Кардинальное число* (*мощность отношения)*** — число его кортежей.
- В отличие от степени отношения кардинальное число отношения изменяется во времени.

Пустые значения

- В отношении СТУДЕНТЫ может храниться информация о посещающих подготовительные курсы вуза.
  - атрибуты «№_студенческого_билета» и «Курс» — неприменимы
  - специальность — может быть неизвестна
- **Пустое значение** — это *неизвестное* значение атрибута, которое *не определено* в данный момент времени и в принципе может быть определено позднее (это не ноль и не пустая строка).
- Обозначение: NULL

Ключи отношения

- Если R — отношение с атрибутами A_1, A_2,..., A_n, то множество атрибутов К = (A_i, A_j,..., A_k) отношения R является **первичным ключом** этого отношения тогда и только тогда, когда удовлетворяются два независимых от времени условия:
- – *уникальность*: в произвольный момент времени никакие два различных кортежа отношения R не имеют одного и того же значения для A_i, A_j,..., A_k;
- – *минимальность*: ни один из атрибутов A_i, A_j,..., A_k не может быть исключен из К без нарушения уникальности.

- Первичный ключ однозначно определяет каждый кортеж отношения и обеспечивает уникальность строк таблицы.
- В зависимости от количества атрибутов, входящих в ключ, различают **простые** и **сложные** (или составные) ключи.
- В общем случае операции объединения выполняются быстрее в том случае, когда в качестве ключа используется самый короткий и самый простой из возможных типов данных.
- **Суперключ** — сложный (составной) ключ с бóльшим числом столбцов, чем необходимо для того, чтобы быть уникальным идентификатором.

- В зависимости от того, содержит ли атрибут, являющийся первичным ключом, какую-либо информацию, различают **искусственные** (суррогатные) и **естественные** ключи.
- Достоинства естественных ключей
  - они несут вполне определенную информацию
  - нет необходимости добавлять в таблицы атрибуты, значения которых не имеют никакого смысла и используются лишь для связи между отношениями.
- Недостатки естественных ключей
  - их использование весьма затруднительно в случае изменчивости предметной области, т.к. значения атрибутов первичного ключа не должны изменяться.
  - как правило, уникальные естественные ключи являются составными и содержат строковые атрибуты.

- В любой из таблиц может оказаться несколько наборов атрибутов, которые можно выбрать в качестве ключа. Такие наборы называются *потенциальными* или *альтернативными* ключами.
- **Вторичный ключ** представляет собой комбинацию атрибутов, отличную от комбинации, составляющей первичный ключ. Причем вторичные ключи не обязательно обладают свойством уникальности.

Индексы

- Индекс - указатель на данные, размещенные в реляционной таблице.
- Индекс – средство ускорения операции поиска записей в таблице.
- Индекс хранит информацию о местонахождении записей, относящихся к индексируемому столбцу таблицы. При добавлении в таблицу новых записей или удалении существующих индекс модифицируется.
- При выполнении запроса к БД, в условие поиска которого входит индексированный столбец, поиск значений производится в первую очередь в индексе.

Пример индекса

Имя	Расположение
Алексеев Д. А.	1
Афанасьев А. В.	4
Кузнецов В. И.	5
Михайлов А. И.	1000
Михайлов В. В.	3
Смирнов А. Д.	6
...	...
Яковлев Н. В.	2

Расположение	...	Имя	...	Курс	...
1		Алексеев Д. А.		2	
2		Яковлев Н. В.		4	
3		Михайлов В. В.		5	
4		Афанасьев А. В.		1	
5		Кузнецов В. И.		1	
6		Смирнов А. Д.		3	
...		...		...	
1000		Михайлов А. И.		3	

- Обращение к индексу выполняется быстрее, чем к таблице.
- В индексе записи хранятся в упорядоченном виде (н-р, в алфавитном порядке)
  ⇒ можно организовать эффективный поиск.

- Типы индексов:
  - Простые индексы
  - Составные индексы.
- Условия оптимальности следования столбцов в составном индексе:
  - первым следует помещать столбец, содержащий наиболее ограничивающее значение
  - первым следует помещать столбец, содержащий данные, которые наиболее часто задаются в условиях поиска.
- Не следует индексировать:
  - столбцы, данные в которых часто изменяются
  - столбцы, содержащие большое количество пустых значений
  - столбцы, содержащие мало уникальных значений
  - небольшие таблицы
  - поля большого размера.

Связанные отношения

- В реляционной модели данные представляются в виде совокупности взаимосвязанных таблиц.
- Взаимоотношение между таблицами называется *связью*.

Главная (основная) таблица

№_студ_билета	Имя	Дата_ рождения	Курс	Специальность
23980282	Алексеев Д. А.	12.03.1982	2	Биология
22991380	Яковлев Н. В.	25.12.1979	4	Физика
22657879	Михайлов В. В.			
24356783	Афанасьев А. В.			
24350283	Кузнецов В. И.			
23125681	СмирновА. Д.			

Внешний ключ

Подчиненная (дополнительная) таблица

Студ_билет	Предмет	Оценка
…	…	…
23980282	Высшая математика	4
23980282	Философия	5
22991380	Высшая математика	3
22991380	Философия	NULL
22657879	Общая физика	5
24356783	Общая физика	NULL

Внешние ключи

- *Внешний ключ* — это атрибут (множество атрибутов) одного отношения, являющийся ключом другого (или того же самого) отношения.
  - используются для установления логических связей между отношениями.
- Часто связь между отношениями устанавливается по первичному ключу,
  - т.е. значениям внешнего ключа одного отношения присваиваются значения первичного ключа другого отношения.

Условия целостности данных

- Целостность – свойство БД, означающее, что она содержит полную, непротиворечивую и адекватно отражающую предметную область информацию.
- Ограничения целостности – правила, определяющие возможные значения в БД:
  - **категорийная целостность**
    - никакой атрибут первичного ключа не может быть пустым
  - **ссылочная целостность**
    - Каждой записи основной таблицы соответствует нуль или более записей дополнительной таблицы
    - Каждая запись дополнительной таблицы имеет только одну родительскую запись основной таблицы

Методы контроля ссылочной целостности

- **Ввод новых записей**
  - Данные сначала вводятся в основную таблицу, а потом в дополнительную.
- **Модификация записей**
  - При редактировании полей связи дополнительной таблицы запись может сменить родителя, но остаться без него не должна.
  - При редактировании полей связи основной таблицы:
    - блокировать модификацию записей, имеющих подчиненные записи или
    - каскадное обновление подчиненных записей

Методы контроля ссылочной целостности

- **Удаление записей**

Три подхода при удалении записей основной таблицы:

- Запрещается удалять запись, имеющую подчиненные записи
- При удалении записи, на которую имеются ссылки, во всех ссылающихся записях значение внешнего ключа = NULL
- Каскадное удаление
  - вместе с основной записью удаляются все подчиненные записи

Типы связей между таблицами

- Характеристика видов связей:

	1 : 1	1 : M	M : 1	M : M
Поля связи основной таблицы	являются ключом	являются ключом	НЕ являются ключом	НЕ являются ключом
Поля связи дополнитель-ной таблицы	являются ключом	НЕ являются ключом	являются ключом	НЕ являются ключом

Связь вида 1 : 1

Поля связи

ключ

- Пример1

Таблица О1

Поле 11	Поле 12
а	10
б	40
в	3

Таблица Д1

Поле 21	Поле 22
а	Стол
в	Книга

- Пример2

БД «Выполняемые научно-исследовательские работы».

Все работы имеют гриф «секретно».

Связь вида 1 : М

Таблица О2 Таблица Д2

ключ

ключ

Код	Вид устройства
а	CD-ROM
б	CD-Recorder
в	Sound Blaster

Код	Фирма-производитель	наличие
а	Acer	да
а	Mitsumi	нет
а	NEC	да
а	Panasonic	да
а	Sony	да
б	Philips	нет
б	Sony	нет
б	Yamaha	да
в	Creative Labs	да

Поля связи

Связь вида М : 1

Поля связи

Таблица О3 Таблица Д3

ключ

ключ

Поле 11	Поле 12	Поле 13
деталь1	чугун	марка1
деталь1	чугун	марка2
деталь2	сталь	марка1
деталь2	сталь	марка2
деталь2	сталь	марка3
деталь3	алюминий	-
деталь4	чугун	марка2

Поле 21	Поле 22	Поле 23
деталь1	4.03.99	90
деталь2	3.01.99	35
деталь3	17.02.99	90
деталь4	6.05.99	240

Связь вида М : М

Поля связи

ключ

ключ

Таблица О4 Таблица Д4

Работает	На станке
Иванов	Станок 1
Иванов	Станок 2
Петров	Станок 1
Петров	Станок 3
Сидоров	Станок 2

Обслуживает	Станок
Голубев	Станок 1
Голубев	Станок 3
Зыков	Станок 2
Зыков	Станок 3

Таблица О4 + Д4

Работает	Станок	Обслуживает
Иванов	Станок 1	Голубев
Иванов	Станок 2	Зыков
Петров	Станок 1	Голубев
Петров	Станок 3	Голубев
Петров	Станок 3	Зыков
Сидоров	Станок 2	Зыков

Основные свойства отношений

- 1. Отсутствие кортежей-дубликатов
- 2. Отсутствие упорядоченности кортежей
- 3. Отсутствие упорядоченности атрибутов
- 4. Атомарность значений атрибутов

Базисные средства манипулирования реляционными данными

- Базовые механизмы:
  - Реляционная алгебра
  - Реляционное исчисление:
    - Исчисление доменов
    - Исчисление кортежей
- Их свойства:
  - Замкнутость относительно понятия отношение
  - Эквивалентность

Примеры языков запросов

- Язык, основанный на реляционной алгебре:
  - ISBL (Information System Base Language) - базовый язык информационных систем
  - В современных СУБД не применяются
- Язык, основанный на реляционном исчислении:
  - SQL (Structured Query Language) - структурированный язык запросов

Реляционная алгебра

- Основная идея:
  Т.к. отношения являются множествами, то средства манипулирования отношениями могут основываться на традиционных теоретико-множественных операциях, дополненных некоторыми специальными операциями, специфичными для баз данных.

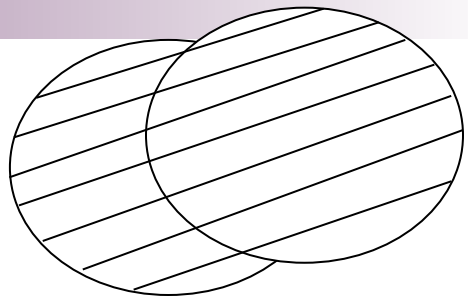
Основные операции

- Теоретико-множественные операции:
  - объединение отношений;
  - пересечение отношений;
  - взятие разности отношений;
  - прямое произведение отношений (декартово произведение).
- Специальные реляционные операции:
  - ограничение отношения (выборка);
  - проекция отношения;
  - соединение отношений;
  - деление отношений.
- Дополнительные операции:
  - операция присваивания,
  - операция переименования атрибутов.

Совместимость отношений по объединению

- Два отношения совместимы по объединению в том и только в том случае, когда в заголовках обоих отношений содержится один и тот же набор имен атрибутов, и одноименные атрибуты определены на одном и том же домене.
- Операции объединение, пересечение и разность выполняются над двумя совместимыми отношениями.

Объединение



- R1 - поставщики из Москвы.

Код поставщика	Имя	Статус	Город_П
S1	Сергей	20	Москва
S4	Николай	20	Москва

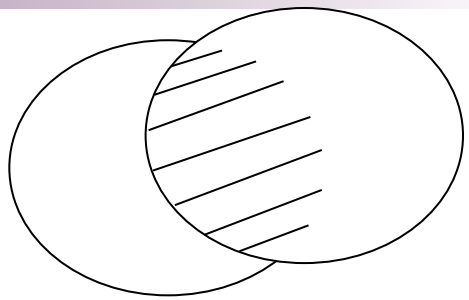
- R2 - поставщики, которые поставляют деталь P1.

Код поставщика	Имя	Статус	Город_П
S1	Сергей	20	Москва
S2	Иван	10	Киев

- R (R1 UNION R2) - поставщики, находящиеся в Москве, или поставщики, выпускающие деталь P1, либо те и другие.

Код поставщика	Имя	Статус	Город_П
S1	Сергей	20	Москва
S2	Иван	10	Киев
S4	Николай	20	Москва

Пересечение



- R1 - поставщики из Москвы.

Код поставщика	Имя	Статус	Город_П
S1	Сергей	20	Москва
S4	Николай	20	Москва

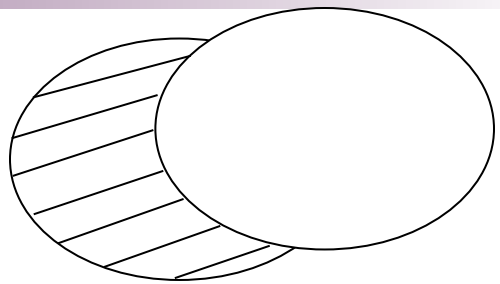
- R2 - поставщики, которые поставляют деталь P1.

Код поставщика	Имя	Статус	Город_П
S1	Сергей	20	Москва
S2	Иван	10	Киев

- R (R1 INTERSECT R2) – поставщики из Москвы, выпускающие деталь P1.

Код поставщика	Имя	Статус	Город_П
S1	Сергей	20	Москва

Разность



- R1 - поставщики из Москвы.

Код поставщика	Имя	Статус	Город_П
S1	Сергей	20	Москва
S4	Николай	20	Москва

- R2 - поставщики, которые поставляют деталь P1.

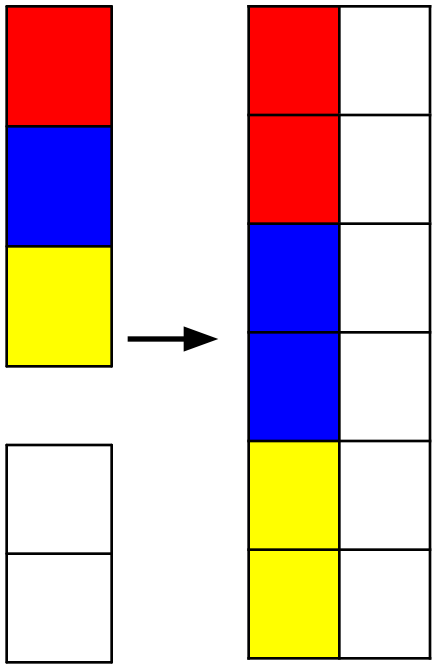
Код поставщика	Имя	Статус	Город_П
S1	Сергей	20	Москва
S2	Иван	10	Киев

- R (R1 MINUS R2) – поставщики из Москвы, НЕ выпускающие деталь P1.

Код поставщика	Имя	Статус	Город_П
S4	Николай	20	Москва

Прямое произведение

- Совместимость по взятию прямого произведения. Два отношения совместимы в том и только в том случае, если множества имен атрибутов этих отношений не пересекаются.
- Переименование RENAME.
- **Произведение** отношения $\mathbf{R_1}$ степени k_1 и отношения $\mathbf{R_2}$ степени k_2 есть такое отношение R степени k_1+k_2, заголовок которого представляет сцепление заголовков R_1 и R_2,
а тело имеет кортежи такие, что первые k_1 элементов кортежей принадлежат R_1,а последние k_2 элементов - множеству R_2.



Пример прямого произведения

R1 Студент

N	ФИО
1	Иванов
2	Петров
3	Сидоров

R2 Предмет

Код	Название
П1	Математика
П2	Физика

R1 TIMES R2

N	ФИО	Код	Название	Оценка
1	Иванов	П1	Математика	
2	Петров	П1	Математика	
3	Сидоров	П1	Математика	
1	Иванов	П2	Физика	
2	Петров	П2	Физика	
3	Сидоров	П2	Физика	

Свойства операций

- Все четыре операции являются ассоциативными.

  (A op B) op C = A (B op C) = A op B op C

  где op - любая из четырех операций

  A, B и C - отношения, обладающие свойствами, требуемыми для корректного выполнения соответствующей операции.

- Все операции, кроме взятия разности, являются коммутативными

  A op B = B op A.

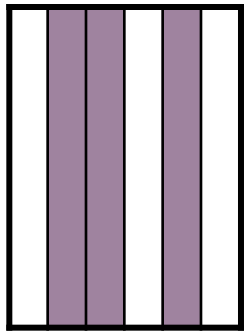
Выборка

- **Выборка** – новое отношение с таким же заголовком, и телом, состоящим из тех кортежей отношения R, которые удовлетворяют истинности логического выражения f.

R WHERE f

- Пример.
  - Студент WHERE Группа=555
  - Студент WHERE Специальность=«Физика» AND Курс>2

Проекция



- **Проекция** отношения А на атрибуты $a_1, a_2, ..., a_n$ - отношение, с заголовком $a_1, a_2, ..., a_n$, и с телом, содержащим кортежи отношения А, за исключением повторяющихся кортежей.

$$A\,[a_1, a_2, ..., a_n]$$

Пример проекции

- Отношение P

Код	Название	Тип	Вес	Город
P1	Гайка	Каленый	12	Москва
P2	Болт	Мягкий	17	Киев
P3	Винт	Твердый	17	Ростов
P4	Винт	Каленый	14	Москва
P5	Палец	Твердый	12	Киев
P6	Шпилька	Каленый	19	Москва

- P [Тип, Город]

Тип	Город
Каленый	Москва
Мягкий	Киев
Твердый	Ростов
Твердый	Киев

Соединение

- Результатом операции соединения является отношение, получаемое путем выполнения операции ограничения по данному условию прямого произведения отношений-операндов.

(R1 TIMES R2) WHERE f

Частные случаи соединения

- Операция соединения называется операцией <u>эквисоединения</u>, если условие соединения имеет вид (a = b), где a и b - атрибуты разных операндов соединения.

R1 [a=b] R2

- Если атрибуты одноименные, то операция называется <u>естественное соединение</u>.

R1 JOIN R2

Естественное соединение

A	
B	
C	

	X
	Y
	Z
	U

A		X
B		X
C		Z

Пример эквисоединения

- Books (id_book, author, title, givento)
- Readers (id_reader, name, address)

Readers [id_reader=givento] Books

Какие книги кому выданы.

Деление

- Результатом деления R1(X,Y) на R2(X) является отношение R с заголовком (Y), и телом, образованным множеством кортежей (y), таких что для всех кортежей (x) из R2 в отношении R1 найдется кортеж (x, y).

R1 R2R1 DIVIDEBY R2

Имя	№отдела
Петр	1
Мария	1
Иван	1
Петр	2
Иван	2

X Y

Имя
Петр
Мария
Иван

X

№отдела
1

Пример

- Books (id_book, author, title)
- Readers (id_reader, name, address)
- ReaderBook(id_book, id_reader)
  – регистрация выдачи книг, после возвращения книги записи не удаляются.

(Readers JOIN ReaderBook) DIVIDEBY Books [id_book]

Список читателей, прочитавших все книги.

Дополнительные операции

- Операция **переименования** производит отношение, тело которого совпадает с телом операнда, но имена атрибутов изменены.

RENAME <отношение> <старое имя атрибута> AS <новое имя атрибута>

- Операция **присваивания** позволяет сохранить результат вычисления реляционного выражения в существующем отношении БД.

Задача

- Дано:
  - База данных имеет следующие схемы отношений
    СОТРУДНИКИ (СОТР_НОМ, СОТР_ИМЯ, СОТР_ЗАРП, ОТД_НОМ)
    ОТДЕЛЫ (ОТД_НОМ, ОТД_КОЛ, ОТД_НАЧ)
- Найти:
  - Имена и номера сотрудников, являющихся начальниками отделов с количеством сотрудников больше 50.
- Решение (реляционное исчисление):
  - Выдать СОТР_ИМЯ и СОТР_НОМ для сотрудников таких, что существует отдел с таким же значением ОТД_НАЧ и значением ОТД_КОЛ > 50.
- Решение (реляционная алгебра):
  - A := СОТРУДНИКИ [СОТР_НОМ = ОТД_НАЧ] ОТДЕЛЫ
  - B := A where ОТД_КОЛ > 50
  - C := B [СОТР_ИМЯ, СОТР_НОМ].

Различия между Р.А. и Р.И.

- Реляционная алгебра – процедурный подход (задает правила выполнения запроса)
- Реляционное исчисление – описательный подход (описывает свойства желаемого результата)
- Алгоритм редукции Кодда – преобразование выражений Р.А. в эквивалентные выражения Р.И. и наоборот.

Реляционное исчисление

- Базисные понятия
  - Переменная
  - Область допустимых значений переменной
  - Правильно построенная формула
- В зависимости от допустимых значений используемых переменных различают
  - исчисление кортежей
  - исчисление доменов

Определение кортежной переменной

- Языки ALPHA → QUEL
- RANGE OF <переменная> IS <список>
  - список – последовательность отношений или выражений над отношениями
- Пример
  - RANGE OF Сотрудник IS СОТРУДНИКИ
- Использование переменной
  - Сотрудник.СОТР_ИМЯ

Правильно построенные формулы

- WFF (Well-Formed Formula) служат для выражения условий, накладываемых на кортежные переменные.
- WFF содержит
  - Операции сравнения =, ≠, >, ≥, <, ≤
  - Логические операции NOT, AND, OR
  - Ключевые слова IF и THEN
  - Ключевые слова EXISTS и FORALL

Примеры WFF

- Простое сравнение:
  <операнд1> Θ <операнд2>
  где Θ - операция сравнения
  - Сотрудник.СОТР_НОМ = 140

- Пусть form - WFF, comp - простое сравнение
  - NOT form
  - comp AND form
  - comp OR form
  - IF comp THEN form,

Кванторы

- Квантор существования

  EXISTS x (form)

  Существует по крайней мере одно такое значение x, что вычисление формулы form дает значение истина

- Квантор всеобщности

  FORALL x (form)

  Для всех значений переменной x вычисление формулы form дает значение истина

Примеры

- Пусть

RANGE OF Сотр1 IS СОТРУДНИКИ

RANGE OF Сотр2 IS СОТРУДНИКИ

- Тогда WFF

EXISTS Сотр2 (Сотр1.СОТР_ЗАРП > Сотр2.СОТР_ЗАРП)

  - для текущего кортежа переменной Сотр1 принимает значение true в том и только в том случае, если во всем отношении СОТРУДНИКИ найдется кортеж (связанный с переменной Сотр2) такой, что значение его атрибута СОТР_ЗАРП удовлетворяет условию сравнения.

FORALL Сотр2 (Сотр1.СОТР_ЗАРП > Сотр2.СОТР_ЗАРП)

  - для текущего кортежа переменной Сотр1 принимает значение true тогда и только тогда, когда для всех кортежей отношения СОТРУДНИКИ (связанных с переменной Сотр2) значения атрибута СОТР_ЗАРП удовлетворяют условию сравнения.

Целевой список (Target_list)

- определяет набор и имена столбцов результирующего отношения.
- состоит из элементов вида:
  - <переменная>.<атрибут>
  - <переменная>
  - <пер>.<атрибут> AS <новое_имя_атрибута>

Выражение реляционного исчисления

- target_list WHERE wff

- Значением выражения является отношение, тело которого определяется WFF, а набор атрибутов и их имена - целевым списком
- Пример
  - СОТР1_ИМЯ WHERE FORALL Сотр2 (Сотр1.СОТР_ЗАРП >= Сотр2.СОТР_ЗАРП)

Задача

- Дано:
  - База данных имеет следующие схемы отношений
    СОТРУДНИКИ (СОТР_НОМ, СОТР_ИМЯ, СОТР_ЗАРП, ОТД_НОМ)
    ОТДЕЛЫ (ОТД_НОМ, ОТД_КОЛ, ОТД_НАЧ)
- Найти:
  - Имена и номера сотрудников, являющихся начальниками отделов с количеством сотрудников больше 50.

Реляционное исчисление доменов

- Переменная домена – скалярная переменная, значения которой охватывают элементы некоторого домена.
  - Пример: ИМЯ
- Условие принадлежности (членства)

$$R(A_1{:}v_1, A_2{:}v_2, \ldots)$$

где A_i – атрибут отношения R, v_i – доменная переменная или константа.

  Условие истинно тогда и только тогда, когда в отношении R существует кортеж, содержащий указанные значения указанных атрибутов.
- Пример выражения:
  - ИМЯ WHERE СОТРУДНИКИ(СОТР_ИМЯ:ИМЯ, ОТД_НОМ:50)

- На исчислении доменов основаны языки
  - ILL (разраб. Лакроикс и Пиротте)
  - FQL
  - DEDUCE
  - QBE (Query By Example)

Языки запросов

- Запрос – специальным образом описанное требование, определяющее состав производимых над БД операций по выборке, удалению или модификации хранимых данных.
- Основные языки запросов:
  - QBE (Query By Example)
  - SQL (Structured Query Language)

Язык запросов по образцу (QBE)

- Злуфф М.М. 1975-1977
- Язык QBE позволяет задавать сложные запросы к БД путем заполнения предлагаемой СУБД запросной формы.
- Можно выполнить операции:
  - Выборка данных
  - Вычисление над данными

    новая таблица
  - Вставка новых записей
  - Удаление записей
  - Модификация данных

    обновленная исходная таблица

Язык SQL

- Разработан в середине 70-х годов.
- 1986 г. – ANSI принял язык SQL в качестве стандарта.
- 1989 г. – в стандарт внесены незначительные изменения.
- 1992 г. – стандарт SQL существенно расширен.
- ANSI SQL-92 или SQL/92.

Типы команд SQL

- DDL (Data Definition Language) — язык определения данных.
- DML (Data Manipulation Language) — язык манипулирования данными.
- DCL (Data Control Language) — язык управления данными.
- DQL (Data Query Language) — язык запросов.
- Команды администрирования базы данных
- Команды управления транзакциями.

Типы данных SQL/92

Строковые типы	символьные строки фиксированной длины	CHARACTER(n)
	символьные строки переменной длины	VARCHAR(n)
Числовые типы	целочисленные типы	SMALLINT INTEGER
	вещественные типы с фиксированной точкой	DECIMAL(n.m)
	вещественные типы с плавающей точкой	FLOAT DOUBLE
	двоичные строки фиксированной длины	BIT(n)
	двоичные строки переменной длины	BIT VARYING(n)

Типы данных SQL/92

Темпоральные типы (представление даты и времени)	DATE TIME TIMESTAMP INTERVAL

Создание таблиц

- CREATE TABLE имя__таблицы
  (имя_поля_1 тип_данных,
  имя_поля_2 тип_данных,
  . . .
  имя_поля_N тип_данных)

- При создании таблицы для полей могут оговариваться ограничительные условия и значения, задаваемые по умолчанию.

Задание ограничений

- ограничение NOT NULL
- ограничение первичного ключа
- ограничение UNIQUE
- ограничение внешнего ключа
- ограничение CHECK.

Ограничение NOT NULL

- CREATE TABLE имя__таблицы
  (имя_поля_1 тип_данных NOT NULL,
  имя_поля_2 тип_данных NULL,
  . . .
  имя_поля_N тип_данных NOT NULL)

Ограничение первичного ключа

- **Способ 1**

```
CREATE TABLE имя__таблицы
(имя_поля_1 тип_данных NOT NULL PRIMARY KEY,
 имя_поля_2 тип_данных,
 . . .
 имя_поля_N тип_данных NOT NULL)
```

- **Способ 2**

```
CREATE TABLE имя__таблицы
(имя_поля_1 тип_данных NOT NULL,
 имя_поля_2 тип_данных,
 . . .
 имя_поля_N тип_данных NOT NULL,
 PRIMARY KEY (имя_поля_1))
```

- **Задание составных первичных ключей**

```
CREATE TABLE имя__таблицы
(имя_поля_1 тип_данных NOT NULL,
 имя_поля_2 тип_данных,
 имя_поля_3 тип_данных NOT NULL,
 . . .
имя_поля_N тип_данных NOT NULL,
PRIMARY KEY (имя_поля_1, имя_поля_3))
```

Ограничение UNIQUE

- **Способ 1**

  CREATE TABLE имя__таблицы
  (имя_поля_1 тип_данных NOT NULL PRIMARY KEY,
  имя_поля_2 тип_данных UNIQUE,
  . . .
  имя_поля_N тип_данных NOT NULL UNIQUE)

- **Способ 2**

  CREATE TABLE имя__таблицы
  (имя_поля_1 тип_данных NOT NULL PRIMARY KEY,
  имя_поля_2 тип_данных,
  имя_поля_3 тип_данных NOT NULL,
  . . .
  имя_поля_N тип_данных NOT NULL UNIQUE
  UNIQUE (имя_поля_2, имя_поля_3))

Ограничение внешнего ключа

- Ограничение внешнего ключа задается в дополнительной таблице
  - в операторе CREATE TABLE
  - в операторе модификации таблиц ALTER TABLE
- FOREIGN KEY имя_внешнего_ключа(список полей внешнего ключа)

  REFERENCES имя_родительской_таблицы (список полей родительского ключа)
- Списки полей должны быть совместимы
- Пример.
  - CREATE TABLE сотрудники

    (…

    FOREIGN KEY Должн_ВК (Код_должности)

    REFERENCES Должности (Код_должности)

)

Каскадное удаление и обновление (не входит в стандарт)

- Каскадное обновление и удаление:
  UPDATE OF имя_родительской_таблицы CASCADES
  DELETE OF имя_родительской_таблицы CASCADES
- Запрет каскадного обновления и удаления:
  UPDATE OF имя_родительской_таблицы RESTRICTED
  DELETE OF имя_родительской_таблицы RESTRICTED

  - CREATE TABLE сотрудники
    (…
    FOREIGN KEY Должн_ВК (Код_должности)
    REFERENCES Должности (Код_должности)
    UPDATE OF Должности CASCADES
    DELETE OF Должности RESTRICTED)

Ограничение CHECK

- Используется для проверки допустимости данных, вводимых в поле таблицы.
- Задается при создании таблицы.

CONSTRAINT имя_ограничения
CHECK (ограничение)

- Примеры.
  - CONSTRAINT CHK_RATE CHECK (Разряд<=20)
  - CONSTRAINT chk_rate CHECK (Разряд<=20 AND Зарплата>=1000)

Задание значений по умолчанию

CREATE TABLE

(. . .

имя_поля тип_данных DEFAULT = значение_по_умолч

. . .)

Модификация таблиц

- ALTER TABLE имя_таблицы
  [MODIFY имя_поля тип_данных]
  [ADD имя_поля тип_данных]
  [DROP имя_поля]

Удаление таблиц

- DROP TABLE имя_таблицы [RESTRICT | CASCADE]

Создание индексов

- CREATE INDEX имя_индекса
  ON имя_таблицы (имя_поля_1, [имя_поля_2, ...])
- Примеры.
  - CREATE INDEX name_idx
    ON Сотрудник (Фамилия)
  - CREATE INDEX fullname_idx
    ON Сотрудник (Фамилия, Имя)

Удаление индексов

- DROP INDEX имя_индекса

Манипулирование данными

- DML — Data Manipulation Language
- Основные операторы DML:
  - INSERT - ввод данных
  - UPDATE - изменение данных
  - DELETE - удаление данных из таблицы

Добавление к таблице новой записи

- INSERT INTO имя_таблицы
  VALUES (значение_1, значение_2, ..., знач_N)
  - Кол-во значений = кол-ву полей таблицы.
  - Тип данных каждого из значений совпадает с типом данных поля в таблице.
  - Порядок полей определяется порядком полей в CREATE TABLE
- Пример.
  - INSERT INTO Должности
    VALUES (12, 'Ведущий программист', 30000.00)

Ввод данных в отдельные поля таблицы

- INSERT INTO имя_таблицы (имя_поля_1, имя_поля_2, ..., имя_поля_N) VALUES (значение_1, значение_2, ..., значение_N)

- Пример.
  - INSERT INTO Сотрудники (Код_сотрудника, Фамилия, Имя, Отчество)
  - VALUES (234,'Иванов','Федор','Михайлович')

Изменение данных

- UPDATE имя_таблицы
  SET имя_поля_1 = значение_1,
  [имя_поля_2 = значение_2,
  . . .
  имя_поля_N = значение_ N]
  [WHERE условие]

- Пример.
  - UPDATE Сотрудники
    SET Телефон = '(3822) 234789'
    WHERE Код_сотрудника = 16

Удаление данных из таблицы

- DELETE FROM имя_таблицы [WHERE условие]

Команды формирования запросов

- SELECT * | { [DISTINCT | ALL] *выражение*, ...}
  FROM *имя_таблицы1* [, ... *имя_таблицыN*]
  [WHERE *условие*]
  [GROUP BY {*имя_поля* | *число*},...]
  [HAVING *условие*]
  [ORDER BY {*имя_поля* | *число*},…]

- Пример.
  - SELECT * FROM Сотрудники
  - SELECT Фамилия, Имя FROM Сотрудники

Объединение таблиц

- ***Полное имя столбца таблицы***
  - имя_таблицы.имя_столбца

Занесение в таблицу данных, содержащихся в другой таблице

- INSERT INTO имя_таблицы (имя_поля_1, имя_поля_2, имя_поля_N)

  SELECT {* | имя_поля_1, имя_поля_2 имя_поля_N}

  FROM имя_таблицы

  [WHERE условие]