

Базы данных

ОСНОВЫ ИНФОРМАЦИОННЫХ СИСТЕМ

В развитии вычислительной техники (ВТ) можно выделить два основных направления:

- применение ВТ для больших численных расчетов, что способствовало развитию методов математического моделирования, языков программирования, численных методов;
- использование ВТ для создания, хранения, обработки, больших объемов данных. Такие задачи решают **информационные системы (ИС)** - поисковые, банковские справочные системы, автоматизированные системы управления предприятием.

Для 1-го типа характерны большие объемы вычислений при относительно небольших потребностях в памяти.

Второй тип требует большие объемы внешней памяти при относительно небольших расчетах.

Информационная система (ИС)

Информационная система – программно-аппаратный комплекс, обеспечивающий сбор, хранение, обработку и выдачу информации для решения задач.

Сначала в ИС применялась файловая модель, но затем стали использовать базы данных (БД).

Крупные ИС - банковские системы, системы заказов авиа и железнодорожных билетов и т. д.

Объем данных в ИС огромен, т. е. нужны устройства надежно хранящие большие объёмы данных во внешней памяти, число пользователей может достигать десятков тысяч.

Для успешного функционирования ИС необходимы эффективные алгоритмы, для создания которых данные прежде всего должны быть **структурированы**.

Общее определение базы данных

База данных – совокупность сведений о конкретных объектах реального мира в какой-либо предметной области.

Предметная область – часть реального мира, подлежащего изучению для организации управления и автоматизации (предприятие, вуз).

Структурирование данных

Пример неструктурированных данных:

Личное дело № 16493, Сергеев Петр Михайлович, дата рождения 1 января 1976; личное дело № 16593, Петрова Анна Владимировна, дата рождения 15 марта 1975; личное дело № 16693, Анохин Андрей Борисович, дата рождения 14 апреля 1976.

Данные в структурированной форме - таблица

№ личного дела	Фамилия	Имя	Отчество	Дата рождения
16493	Сергеев	Петр	Михайлович	01.01.76
16593	Петрова	Анна	Владимировна	15.03.75
16693	Анохин	Андрей	Борисович	14.04.76

Структурирование – введение соглашений о способах представления данных.

Определение базы данных

- **База данных (БД)** – поименованная совокупность структурированных данных, относящихся к определенной предметной области.
- **Система управления базами данных (СУБД)** – комплекс программных и языковых средств, необходимых для создания баз данных, поддержания их в актуальном состоянии и организации поиска в них информации.

Классификация баз данных

Технология обработки данных

ЦЕНТРАЛИЗОВАННЫЕ

БД хранится в памяти одной ВС (мэйнфрейм - доступ с терминалов или файловый сервер ЛС)

РАСПРЕДЕЛЕННЫЕ

состоит из нескольких частей, которые хранятся в различных ЭВМ вычислительной сети.

Классификация баз данных



Классификация баз данных

Централизованные БД с сетевым доступом могут иметь следующую архитектуру:

- **файл-сервер**
- **клиент-сервер**

Архитектура файл-сервер _



...предполагает выделение одного компьютера сети (сервер) для хранения централизованной БД. Остальные компьютеры сети (клиенты) - рабочие станции, которые копируют требуемую часть централизованной БД в свою память, где¹⁰ и происходит обработка.

Достоинства и недостатки файл-сервера

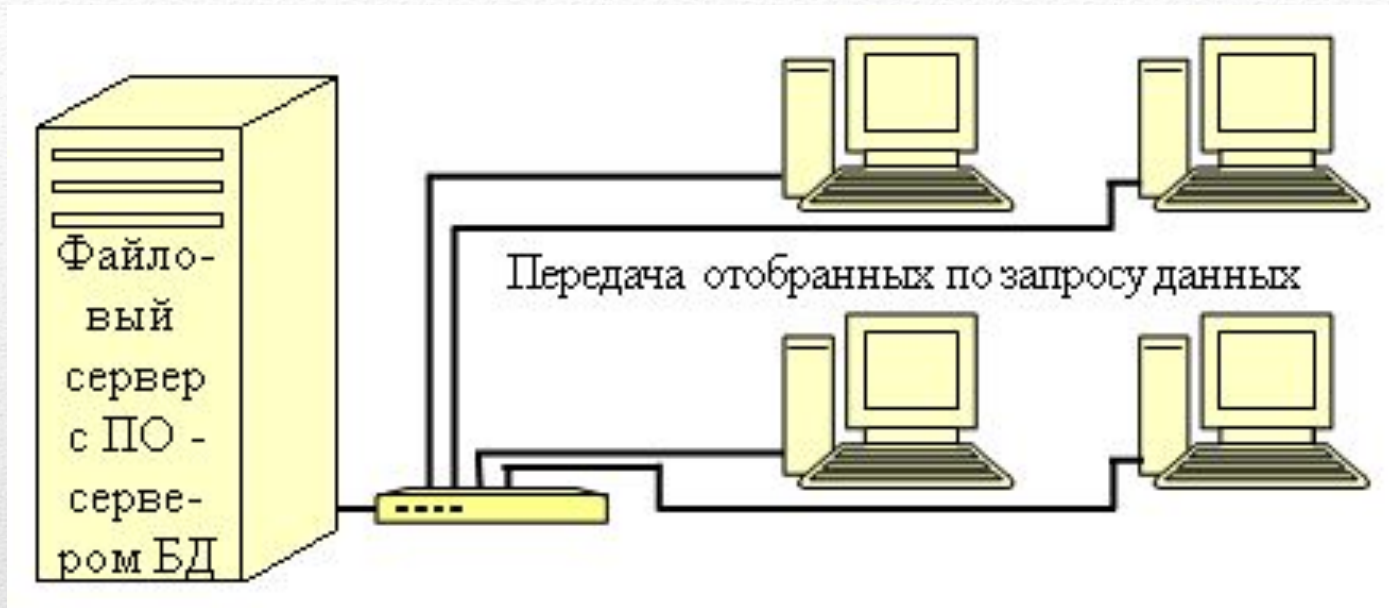
Достоинства

1. Нет высоких требований к производительности сервера (главное – требуемый объем дискового пространства);
2. На сервере СУБД не размещается, не устанавливается;

Недостатки

1. При большой интенсивности запросов к централизованной БД увеличивается нагрузка на каналы сети, что снижает производительность ИС в целом;
2. Отсутствие специальных механизмов безопасности файла (файлов) БД со стороны СУБД;

Архитектура клиент-сервер _



...предполагает, что сервер, выделенный для хранения централизованной БД, производит обработку клиентских запросов. Клиенты получают по сети уже обработанные данные.

Достоинства и недостатки клиент-сервера

Достоинства:

1. Более низкий трафик сети, чем в модели файл-сервер
2. SQL-сервер обеспечивает функции по обеспечению целостности и безопасности данных;

Недостатки:

1. В определенных случаях некоторые наборы данных могут занимать достаточно существенный объем;

Модели данных

Для реализации основных функций в ИС используются различные принципы описания данных. Ядром любой БД является **модель представления данных**.

Хранимые в базе данные имеют определенную логическую структуру – описываются некоторой моделью представления данных, поддерживаемой СУБД.

Модель данных определяет способ их организации, ограничения целостности и множество операций, допустимых над объектом.

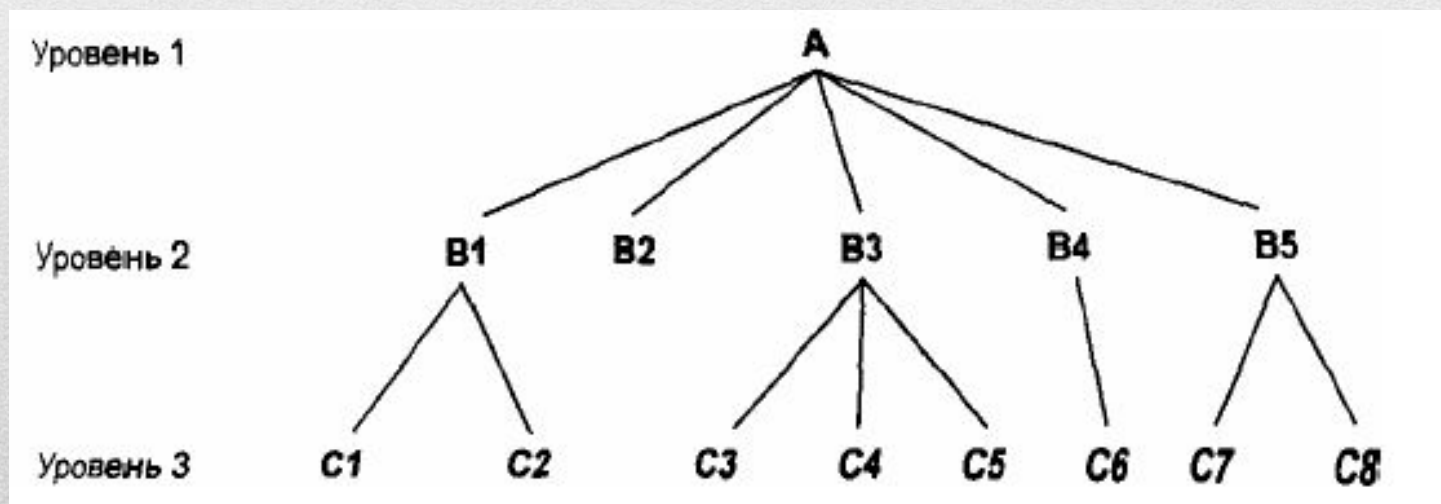
К числу классических относятся следующие модели:

- ☐ иерархическая
- ☐ сетевая
- ☐ реляционная

Иерархическая модель данных...

... была разработана одной из первых, на ее основе в конце 60 – начале 70 г.г. создана **первая профессиональная СУБД IMS фирмы IBM.**

Связи между данными описываются с помощью дерева или упорядоченного графа



Иерархическая модель данных

Основные понятия иерархической модели:

- ❑ **узел** – набор атрибутов данных, описывающих объект;
- ❑ **связь** - линия, связывающая узлы нижнего уровня (потомки) с одним узлом вышележащего (предки) уровня;
- ❑ **уровень** – номер слоя узлов, отсчитанный от корня.

Количество деревьев в БД определяется числом **корневых записей**.

К каждому узлу существует единственный путь от корня.

Достоинство и недостатки иерархической модели

Достоинство:

1. Достаточно высокие показатели времени выполнения операций над данными

Недостатки:

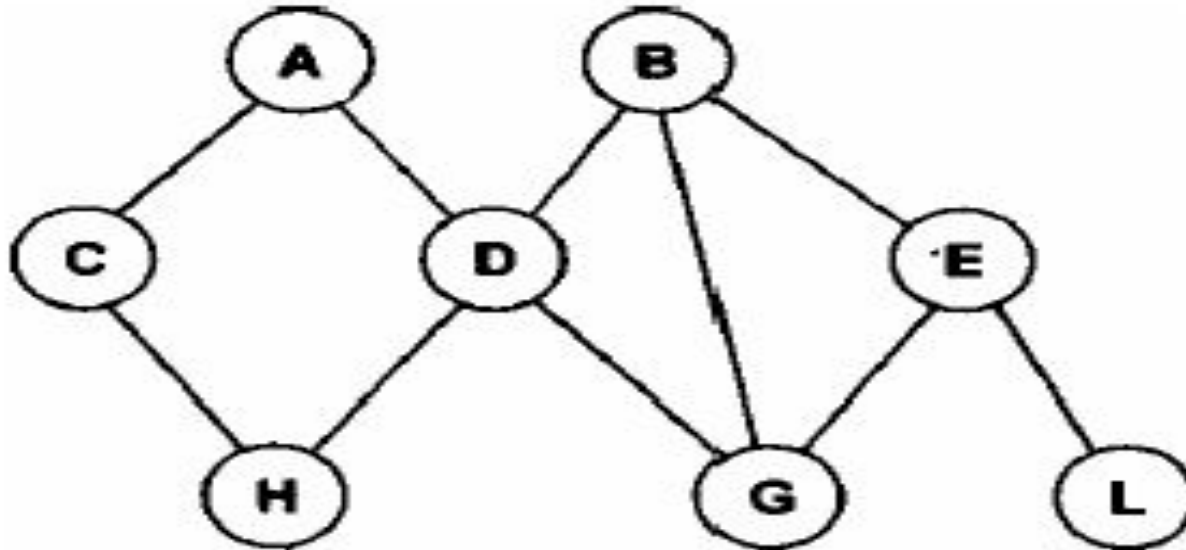
1. Сложность понимания для обычного пользователя
2. Присутствие избыточности

Иерархическая модель данных



Сетевая модель данных

Связи между данными описываются произвольным графом



Сетевая структура имеет те же составляющие, что и иерархическая, но каждый узел может быть связан с любым другим узлом. Этот подход - расширение иерархического. В сетевой структуре потомок может иметь любое число предков.

Сетевая модель данных

Студент (номер зачетной книжки, фамилия, группа)



Достоинства и недостатки сетевой модели

Достоинства:

1. Минимальная избыточность;
2. В сравнении с иерархической моделью сетевая имеет большие возможности по образованию новых связей;
3. Эффективная реализация по показателям затрат памяти.

Недостатки:

1. Сложность понимания для обычного пользователя;
2. Ослаблен контроль правильности образования связей
3. Сложно осуществить операции поиска

Реляционная модель данных

Наиболее универсальная. Впервые предложена Эдгаром Коддом в 1970 г.

Важнейшее понятие - **сущность** – объект любой природы, данные о котором хранятся в БД.

Данные о сущности хранятся в двумерных таблицах, которые называют **реляционными**.

Основывается на понятии отношение (relation). Графически отношение представляется двумерной таблицей.

В реляционной СУБД предполагается, что пользователь воспринимает БД как набор таблиц.

Примеры реляционных СУБД:

MicroSoft Access Paradox dBASE FoxPro Clarion DB2 Oracle

Свойства реляционной таблицы

- 1. Один элемент таблицы – один элемент данных;**
- 2. Все столбцы таблицы содержат однородные по типу данные (целые, вещ. числа, текст и т. д.);**
- 3. Каждый столбец имеет уникальное имя;**
- 4. Число столбцов задается при создании таблицы;**
- 5. Порядок записей в отношении может быть произвольным;**
- 6. Записи не должны повторяться;**
- 7. Количество записей в отношении не ограничено.**

Реляционная модель данных

№ личного дела	Фамилия	Имя	Отчество	Дата рождения	Группа
16493	Сергеев	Петр	Михайлович	01.01.76	111
16593	Петрова	Анна	Владимировна	15.03.75	112
16693	Петров	Андрей	Борисович	14.04.76	113

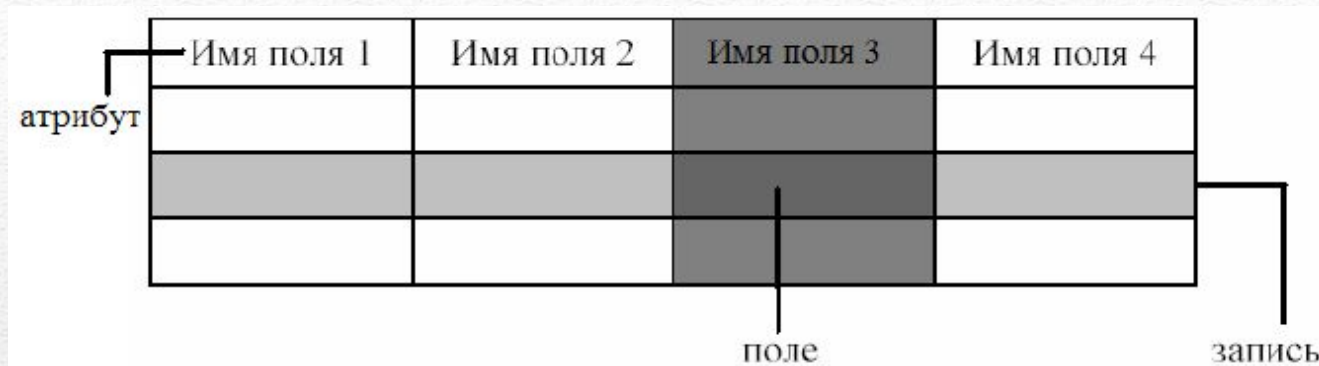
Достоинство:

Простота и понятность для широкого пользователя.

Недостаток:

Необходимая избыточность из-за связей между таблицами.

Реляционная модель данных



Альтернативные варианты терминов реляционной БД

Альтернативный вариант	Альтернативный вариант	Альтернативный вариант
Таблица	Файл	Отношение
Строка	Запись	Кортеж
Столбец	Поле	Домен

Строка заголовков называется **схемой**
отношения.

Первичный ключ

Первичным ключом, или **ключом отношения**, называется атрибут отношения (набор атрибутов), который однозначно идентифицирует каждый из его кортежей.

Студент(№ личного дела, Фамилия, Имя, Отчество, Д

№ личного дела	Фамилия	Имя	Отчество	Дата рождения	Группа
16493	Сергеев	Петр	Михайлович	01.01.76	111
16593	Петрова	Анна	Владимировна	15.03.75	112
16693	Петров	Андрей	Борисович	14.04.76	113

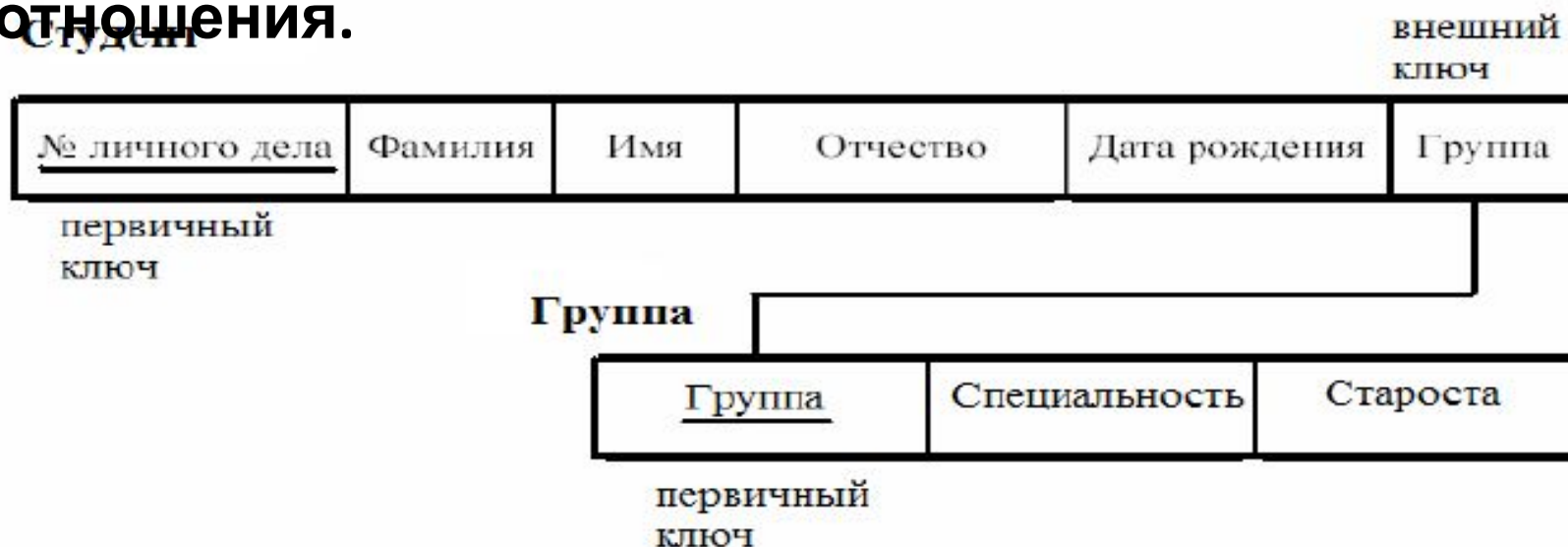
Свойства первичного ключа

- 1. Уникальность** – в таблице может быть только один первичный ключ, у составного ключа поля могут повторяться, но не все.
- 2. Неизбыточность** – не должно быть полей, которые будучи удаленными из первичного ключа, не нарушат его уникальность.
- 3.** В состав первичного ключа не должны входить поля типа, комментариев, графическое.

Внешний ключ

Логические связи между отношениями устанавливаются с помощью **внешних ключей**.

Внешний ключ – это атрибут (набор атрибутов) одного отношения, являющийся ключом другого отношения.



Индексы

ЗАДАНИЕ: осуществить в таблице поиск по
Фамилии

Расположени е	...	Фамилия	...	Группа	...
1		Алексеев		151	
2		Яковлев		152	
3		Михайлов		151	
4		Афанасьев		511	
...				...	
1000		Кузнецов		221	

Индексы

Индекс – средство ускорения операции поиска записей в таблице, а также выполнения операций, использующих поиск (извлечение, модификация, сортировка и т.д.)

Фамилия	Располо- жение		Располо- жение	...	Фамилия	...	Группа	...
Алексеев	1	→	1		Алексеев		151	
Афанасьев	4	→	2		Яковлев		152	
Кузнецов	1000	→	3		Михайлов		151	
Михайлов	3	→	4		Афанасьев		511	
...		→	...		...			
Яковлев	2	→	1000		Кузнецов		221	

Индексы

Индексы представляют собой механизмы быстрого доступа к данным в таблицах БД.

Сущность индексов состоит в том, что они хранят значения индексных полей (т.е. полей, по которым построен индекс) и указатель на запись в таблице.

При последовательном методе доступа для выполнения запроса к таблице БД просматриваются все записи таблицы, от первой до последней.

Индексы

При **индексно-последовательном** методе доступа для выполнения запроса к таблице БД указатель в индексе устанавливается на первую строку, удовлетворяющую условию запроса (или его части), и считывается запись из таблицы по хранящемуся на нее в индексе указателю.

Определение первичных и внешних ключей таблиц БД приводят к созданию индексов по полям, объявленным в составе первичных или внешних ключей.

Виды индексов

Индексы

```
graph TD; A[Индексы] --> B[ПЕРВИЧНЫЙ]; A --> C[ВТОРИЧНЫЙ];
```

ПЕРВИЧНЫЙ

Ключевое поле таблицы всегда индексируется, поэтому для него не требуется дополнительно определять индекс

ВТОРИЧНЫЙ

- Ускоряют выполнение запросов.
- Может быть несколько.
- Может входить любое количество полей.
- Одно и то же поле может входить в разные индексы.

Формирование отношений

Пример 1: для деканата создать БД о студентах, определим имена выделенных атрибутов и их краткие характеристики:

№ - номер личного дела студента

Фамилия – фамилия студента

Имя – имя студента

Дата рождения – дата рождения студента

Группа – номер группы, в которой учится студент

Специальность – специальность

Формирование отношений

№	Фамилия	Имя	Дата рождения	Группа	Специальность
15345	Иванов	Иван	15.04.1989	392	Информатика-менеджмент
15349	Медведева	Анна	13.02.1989	392	Информатика-менеджмент
15310	Петров	Михаил	12.11.1989	392	Информатика-менеджмент
15259	Сидоров	Николай	26.01.1987	591	Информатика-английский язык
15263	Санин	Александр	20.10.1987	591	Информатика-английский язык

Нормализация

При проектировании структуры БД определяют сущности (объекты, явления) предметной области, которые должны найти отражение в БД.

Анализ предметной области приводит к созданию **эскиза** БД. Сначала изображают сущности и связи между ними. Обычно сущности соответствует таблица. В эскизе второго порядка - для каждой таблицы БД приводится список атрибутов - полей записи.

До этих пор проектирование БД процесс неоднозначный, и узловые его моменты могут быть формализованы.

Одной из формализаций является требование, по которому реляционная БД должна быть нормализована.

Нормализация

Нормализация отношений – правила формирования отношений (таблиц), которые устраняют дублирование, противоречивость хранимых в БД.

Э. Коддом разработаны **три нормальные формы отношений** и предложен механизм, позволяющий любое отношение преобразовать к третьей нормальной форме (3НФ)



Разбиение отношения

Студент

№	Фамилия	Имя	Дата рождения	Группа
15345	Иванов	Иван	15.04.1989	392
15349	Медведева	Анна	13.02.1989	392
15310	Петров	Михаил	12.11.1989	392
15259	Сидоров	Николай	26.01.1987	591
15263	Санин	Александр	20.10.1987	591

Группа

Группа	Специальность
392	Информатика-менеджмент
591	Информатика-английский язык

Формирование отношений

Пример 2: для деканата создать БД о студентах
Имена атрибутов и их краткие характеристики:

№ - номер личного дела студента

Фамилия – фамилия студента

Имя – имя студента

Дата рождения – дата рождения студента

Группа – номер группы, в которой учится
студент

Специальность – специальность,

Семестр – номер семестра обучения

Предмет – предмет, изучаемый студентом

Оценка – экзаменационная оценка за предмет

Формирование отношений

№	Фамилия	Имя	Дата рождения	Группа	Специальность	Семестр	Предмет	Оценка
15345	Иванов	Иван	15.04.1989	392	ИМ	6	Англ.язык	Хорошо
15345	Иванов	Иван	15.04.1989	392	ИМ	6	Теор.алгоритмов	Отлично
15345	Иванов	Иван	15.04.1989	392	ИМ	6	Химия	Хорошо
15310	Петров	Михаил	12.11.1989	392	ИМ	6	Англ.язык	Удовлет.
15310	Петров	Михаил	12.11.1989	392	ИМ	6	Теор.алгоритмов	Удовлет.
15310	Петров	Михаил	12.11.1989	392	ИМ	6	Химия	Хорошо
15259	Сидоров	Николай	26.01.1987	591	ИА	10	Архитектура ЭВМ	Отлично
15259	Сидоров	Николай	26.01.1987	591	ИА	10	Комп.моделирование	Отлично

Первая нормальная форма (1НФ) _

... требует, чтобы каждое поле таблицы БД:

- было неделимым;**
- не содержало повторяющихся групп.**

Неделимость поля означает, что значение поля не должно делиться на более мелкие значения.

Например, в поле ФИО содержится Фамилия и Имя, требование неделимости не соблюдается и нужно из данного поля выделить Фамилию и отдельно поле Имя.

Повторяющимися являются поля, содержащие одинаковые по смыслу значения. Например, если требуется получить статистику сдачи экзаменов по предметам, можно создать поля для хранения данных об оценке по каждому предмету. Однако в этом случае мы имеем дело с повторяющимися группами.

Первая нормальная форма (1НФ)

Отношение называется приведенным к 1НФ, если все его атрибуты неделимы.

Пример. Отношение, с полем ФИО, не приведено к 1НФ, если в запросах БД требуется выделить отдельно фамилию или имя.

№	Фамилия	Имя	Дата рождения	Группа	Специальность	Семестр	Предмет	Оценка
15345	Иванов	Иван	15.04.1989	392	ИМ	6	Англ.язык Теор.алгоритмов Химия	Хорошо Отлично Хорошо
15289	Петров	Михаил	12.11.1988	392	ИМ	6	Англ.язык Теор.алгоритмов Химия	Удовлет. Удовлет. Хорошо
15259	Сидоров	Николай	26.01.1987	591	ИА	10	Архитектура ЭВМ Комп.моделирование	Отлично Отлично

Отношение НЕ находится в
1НФ

Первая нормальная форма (1НФ)

№	Фамилия	Имя	Дата рождения	Группа	Специальность	Семестр	Предмет	Оценка
15345	Иванов	Иван	15.04.1989	392	ИМ	6	Англ. язык	Хорошо
15345	Иванов	Иван	15.04.1989	392	ИМ	6	Теор. алгоритмов	Отлично
15345	Иванов	Иван	15.04.1989	392	ИМ	6	Химия	Хорошо
15310	Петров	Михаил	12.11.1989	392	ИМ	6	Англ. язык	Удовлет.
15310	Петров	Михаил	12.11.1989	392	ИМ	6	Теор. алгоритмов	Удовлет.
15310	Петров	Михаил	12.11.1989	392	ИМ	6	Химия	Хорошо
15259	Сидоров	Николай	26.01.1987	591	ИА	10	Архитектура ЭВМ	Отлично
15259	Сидоров	Николай	26.01.1987	591	ИА	10	Комп. моделирование	Отлично

**Отношение находится в
1НФ**

Вторая нормальная форма (2НФ)

Функциональная зависимость полей — зависимость, при которой в строке определенному значению ключевого поля соответствует только одно значение не ключевого поля.

Функционально не ключевое поле зависит от составного ключа, но не зависит от любого поля из составного ключа.

Пример. СТУДЕНТ(ФАМИЛИЯ, ИМЯ, ОТЧЕСТВО, ФАКУЛЬТЕТ, КУРС, ГРУППА). Поля ФАКУЛЬТЕТ, КУРС, ГРУППА функционально полно зависят от составного ключа.

Отношение находится во 2НФ, если оно находится в 1НФ , и каждое не ключевое поле функционально полно зависит от составного ключа.

Вторая нормальная форма (2НФ)

2НФ требует, чтобы все поля таблицы зависели от первичного ключа, то есть, чтобы первичный ключ однозначно определял запись и не был избыточен.

Поля, которые зависят только от части первичного ключа, должны быть выделены в составе отдельных таблиц.

Вторая нормальная форма (2НФ)

Пример. УСПЕВАЕМОСТЬ (НОМЕР ЗАЧЕТКИ, ФАМИЛИЯ, ДИСЦИПЛИНА, ОЦЕНКА) составным ключом является совокупность НОМЕР ЗАЧЕТКИ + ДИСЦИПЛИНА.

Отношение находится в 1НФ, но оно не находится во 2НФ, так как поле ФАМИЛИЯ не имеет полной функциональной зависимости от составного ключа.

Для перевода во 2НФ нужно исключить поле ФАМИЛИЯ, так как оно функционально зависит от НОМЕРА ЗАЧЕТКИ, т.е. исходное отношение необходимо разбить на два связанных отношения:
УСПЕВАЕМОСТЬ (НОМЕР ЗАЧЕТКИ, ДИСЦИПЛИНА, ОЦЕНКА)

СПИСОК (НОМЕР ЗАЧЕТКИ, ФАМИЛИЯ). 46

Связь осуществляется по полю НОМЕР ЗАЧЕТКИ.

Третья нормальная форма (3НФ)

Позволяет устранить транзитивную зависимость. Транзитивная зависимость существует в отношении, если существуют два описательных поля, в которых 1-е зависит от ключа, а 2-е зависит от 1-го.

Отношение находится в 3НФ, если оно находится во 2НФ, и каждое не ключевое поле не транзитивно зависит от ключа.

Пример. СТУДЕНТ (ФАМИЛИЯ, ФАКУЛЬТЕТ, НАЗВАНИЕ вуза, АДРЕС) поле АДРЕС транзитивно (через поле НАЗВАНИЕ вуза) зависит от ключа ФАМИЛИЯ. При заполнении экземплярами такого отношения поле АДРЕС будет многократно повторяться. Для устранения транзитивной зависимости используется расщепление отношения на несколько.

Формирование отношений

№	Фамилия	Имя	Дата рождения	Группа	Специальность	Семестр	Предмет	Оценка
15345	Иванов	Иван	15.04.1989	392	ИМ	6	Англ.язык	Хорошо
15345	Иванов	Иван	15.04.1989	392	ИМ	6	Теор.алгоритмов	Отлично
15345	Иванов	Иван	15.04.1989	392	ИМ	6	Химия	Хорошо
15310	Петров	Михаил	12.11.1989	392	ИМ	6	Англ.язык	Удовлет.
15310	Петров	Михаил	12.11.1989	392	ИМ	6	Теор.алгоритмов	Удовлет.
15310	Петров	Михаил	12.11.1989	392	ИМ	6	Химия	Хорошо
15259	Сидоров	Николай	26.01.1987	591	ИА	10	Архитектура ЭВМ	Отлично
15259	Сидоров	Николай	26.01.1987	591	ИА	10	Комп.моделирование	Отлично

Результат проектирования БД «Студент»

№	Фамилия	Имя	Дата рождения	Группа
15345	Иванов	Иван	15.04.1989	392
15310	Петров	Михаил	12.11.1989	392
15259	Сидоров	Николай	26.01.1987	591

Группа	Специальность
392	ИМ
591	ИА

№	Семестр	Предмет	Оценка
15345	6	Англ. язык	Хорошо
15345	6	Теор. алгоритмов	Отлично
15345	6	Химия	Хорошо
15310	6	Англ. язык	Удовлет.
15310	6	Теор. алгоритмов	Удовлет.
15310	6	Химия	Хорошо
15259	10	Архитектура ЭВМ	Отлично
15259	10	Комп. моделирование	Отлично

SQL

Доступ к информации, содержащейся в реляционных БД, для пользователей, программ и ВС обеспечивает язык запросов SQL (Structured Query Language)

Независимость от конкретных СУБД – все распространенные СУБД используют SQL.

- Приложения, созданные с помощью SQL, допускают использование как для локальных БД, так и для клиент-серверных систем.
- Операторы SQL употребляются для интерактивного и программного доступа, поэтому части программ, содержащие обращение к БД, можно вначале проверить в интерактивном режиме, а затем встраивать в программу.

Реляционная алгебра

**SQL основан на операциях реляционной алгебры.
Реляционная алгебра – набор операций,
выполняемых над отношениями.**

**Реляционная алгебра разработана Э.Коддом в
рамках реляционной модели**

**Применяя операции реляционной алгебры к одним
отношениям можно получить другие отношения**

Основные операции реляционной алгебры

Объединением двух совместимых по типу отношений A и B называют отношение с тем же заголовком, что и у A и B , и телом, состоящим из кортежей, принадлежащих или A , или B , или обоим отношениям.

Отношения называют совместимыми по типу, если их заголовки идентичны, и атрибуты с одинаковыми именами определены на одних и тех же доменах.

Пересечением двух совместимых по типу отношений A и B называют отношение с тем же заголовком, что и у A и B , и телом, состоящим из кортежей, принадлежащих сразу обоим отношениям A и B .

Вычитанием двух совместимых по типу отношений A и B называется отношение с тем же заголовком, что и у A и B , и телом, состоящим из кортежей, принадлежащих отношению A и не принадлежащих отношению B .

Основные операции реляционной алгебры

Объединение

Студенты 392

№	Фамилия	Имя	Дата рождения	Группа
15345	Иванов	Иван	15.04.1989	392
15349	Медведева	Анна	13.02.1989	392
15310	Петров	Михаил	12.11.1989	392

Студенты 591

№	Фамилия	Имя	Дата рождения	Группа
15259	Сидоров	Николай	26.01.1987	591
15263	Санин	Александр	20.10.1987	591

Объединение студентов 392 и 591 группы

№	Фамилия	Имя	Дата рождения	Группа
15345	Иванов	Иван	15.04.1989	392
15349	Медведева	Анна	13.02.1989	392
15310	Петров	Михаил	12.11.1989	392
15259	Сидоров	Николай	26.01.1987	591
15263	Санин	Александр	20.10.1987	591

Основные операции реляционной алгебры

Пересечение

Студенты 392 и 591 группы

№	Фамилия	Имя	Дата рождения	Группа
15345	Иванов	Иван	15.04.1989	392
15349	Медведева	Анна	13.02.1989	392
15310	Петров	Михаил	12.11.1989	392
15259	Сидоров	Николай	26.01.1987	591
15263	Санин	Александр	20.10.1987	591

Студенты 392

№	Фамилия	Имя	Дата рождения	Группа
15345	Иванов	Иван	15.04.1989	392
15350	Морозова	Ирина	21.12.1989	392
15359	Петрова	Мария	12.11.1989	392

Результат пересечения

№	Фамилия	Имя	Дата рождения	Группа
15345	Иванов	Иван	15.04.1989	392

Основные операции реляционной алгебры

Вычитание

Студенты 392 и 591 группы

№	Фамилия	Имя	Дата рождения	Группа
15345	Иванов	Иван	15.04.1989	392
15349	Медведева	Анна	13.02.1989	392
15310	Петров	Михаил	12.11.1989	392
15259	Сидоров	Николай	26.01.1987	591
15263	Санин	Александр	20.10.1987	591

Студенты 392

№	Фамилия	Имя	Дата рождения	Группа
15345	Иванов	Иван	15.04.1989	392
15350	Морозова	Ирина	21.12.1989	392
15359	Петрова	Мария	12.11.1989	392

Результат вычитания

№	Фамилия	Имя	Дата рождения	Группа
15349	Медведева	Анна	13.02.1989	392
15310	Петров	Михаил	12.11.1989	392
15259	Сидоров	Николай	26.01.1987	591
15263	Санин	Александр	20.10.1987	591

Основные операции реляционной алгебры

Выборка

Студенты 392 и 591 группы

№	Фамилия	Имя	Дата рождения	Группа
15345	Иванов	Иван	15.04.1989	392
15349	Медведева	Анна	13.02.1989	392
15310	Петров	Михаил	12.11.1989	392
15259	Сидоров	Николай	26.01.1987	591
15263	Санин	Александр	20.10.1987	591

Результат выборки студентов 591 группы

№	Фамилия	Имя	Дата рождения	Группа
15259	Сидоров	Николай	26.01.1987	591
15263	Санин	Александр	20.10.1987	591

Выборкой на отношении A с условием s называется отношение с тем же заголовком, что и у отношения A , и телом, состоящем из кортежей, значения атрибутов которых при подстановке в условие s дают значение ИСТИНА.

Оператор выбора языка SQL

SELECT [DISTINC] элементы FROM таблица(цы) [WHERE условие] [GROUP BY поле(я) [HAVING условие]] [ORDER BY поле(я)]

Производит выборку указанных элементов из указанных таблиц по указанными условиями. Результат - новая таблица.

где:

SELECT – выбрать

DISTINC – устранить в итоговой таблице одинаковые строки

FROM – из (таблиц)

WHERE – где

GROUP BY – выборка с точностью до группы строк

HAVING – условие выборки группы

57

ORDER BY – упорядочивание результата по указанным полям

Оператор выбора языка SQL

SELECT [DISTINC] элементы
FROM таблица(цы)
[WHERE условие]
[GROUP BY поле(я) [HAVING условие]]
[ORDER BY поле(я)]

где:

SELECT – выбрать

DISTINC – устранить витоговой таблице

одинаковые строки

FROM – из (таблиц)

WHERE – где

GROUP BY – выборка с точностью до группы

строк

HAVING – условие выборки группы

ORDER BY – упорядочивание результата по

указанным полям

Оператор выбора языка SQL

1. Выбрать фамилии всех студентов

SELECT Фамилия FROM Студенты

Студенты				
№	Фамилия	Имя	Дата рождения	Группа
15345	Иванов	Иван	15.04.1989	392
15349	Медведева	Анна	13.02.1989	392
15310	Петров	Михаил	12.11.1989	392
15259	Сидоров	Николай	26.01.1987	591
15263	Санин	Александр	20.10.1987	591



Фамилия
Иванов
Медведева
Петров
Сидоров
Санин

2. Вывести дату рождения студента Петрова

**SELECT Дата рождения FROM Студенты WHERE
Фамилия="Петров"**

3. Вывести все сведения о студентах 591 группы, упорядочив их по фамилии

SELECT * FROM Студенты WHERE Группа=591 ORDER BY Фамилия

4. Вывести дату рождения студента Петрова

SELECT Дата рождения FROM Студенты WHERE

Возможности языка SQL

- **Создание базы данных и таблицы с полным описанием их структуры**
 - **Выполнение основных операций манипулирования данными, в частности, вставки, модификации и удаления данных из таблиц.**
 - **Выполнение простых и сложных запросов, осуществляющих преобразование данных**

Тенденции развития СУБД

Направление развития реляционных СУБД в последние годы заметно меняется. Если предыдущее десятилетие они развивались, чтобы обеспечить быстрый доступ к данным, то теперь часто нужно хранить еще графику и звук. Существенно изменилась аппаратная среда - она стала сетевой. С развитием Web появилась необходимость поддерживать HTML - страницы.

Основные функции СУБД

- управление данными во внешней памяти (на дисках)
- управление данными в оперативной памяти
- журнализация изменений, резервное копирование и восстановление базы данных после сбоев
- поддержка языка манипулирования данными

В состав СУБД входят средства для:

- создания БД и модификации их структуры, создания индексных файлов
- работы с базами в табличном формате или в виде стандартной формы с расположением полей построчно
- разработки экранных форм
- генерации печатных форм
- генерации запросов очень сложной структуры
- в системах, ориентированных на разработчика, разработка меню, справочной системы и проекта, включающего все перечисленные выше компоненты и компилирующегося в исполняемую программу

СУБД

Система управления базами данных (СУБД) – комплекс программных и языковых средств, необходимых для создания БД, поддержания их в актуальном состоянии и организации поиска в них необходимой информации.

Термин "сервер баз данных" обычно используют для обозначения всей СУБД, основанной на архитектуре "клиент-сервер", включая и серверную, и клиентскую части.

Сервер баз данных - СУБД, которая принимает запросы по сети и возвращает информацию, соответствующую запросу. Наиболее распространенными серверами являются в настоящее время Interbase, Microsoft SQL Server, Oracle, IBM DB2, Informix

Классификация СУБД

**По типу управляемой базы данных СУБД
разделяются на:**

- **Сетевые (CronosPlus)**
- **Иерархические (IMS, System 2000)**
- **Реляционные (MS Access, Paradox, Interbase, FireBird, MySQL, DB2, Oracle, Ingres)**
- **Объектно-реляционные (Oracle Database, MicroSoft SQL Server 2005)**

Сравнение СУБД

Характеристики	ORACLE	IBM DB2	MS SQL SERVER	SYBASE 12.0
Язык программирования	Java, Delphi PL/SQL	Java, SQL 2000	Transact-SQL	Java, Transact-SQL
Объектно-ориентированное проектирование БД	Да	Да	Нет	Да (через Java)
Мультимедийные типы данных	Да	Да	Ограниченно	Ограниченно
Макс. размер таблиц	Не огр.	64 Гбайт		Не огр.
Макс. число таблиц в БД	Не огр.	Практически не ограничено	Не огр.	Не огр.
Макс. число таблиц на каждое соединение	Не огр.	31	Не огр.	Не огр.
Макс. число пользователей БД	Не огр.	Практически не ограничено	Не огр.	Не огр.
Рекомендуемая емкость ОП на одного пользователя	Изменяемая величина	Локальный пользователь: 550 Кб Удаленный - 250 Кб		50 Кбайт

Возможности СУБД

Производительность СУБД оценивается:

- скоростью поиска информации;**
- скоростью выполнения операций обновления, вставки, удаления данных;**
- временем выполнения операций импортирования базы данных из других форматов;**
- максимальным числом параллельных обращений к данным в многопользовательском режиме;**
- временем генерации отчета.**

Возможности СУБД

Обеспечение целостности данных на уровне БД

Характеристика средства, позволяющих удостовериться, что информация в БД всегда остается корректной и полной:

- **проверка уникальности первичных ключей,**
- **ограничение операций над данными,**
- **каскадное обновление и удаление данных.**

Обеспечение безопасности

Некоторые СУБД обеспечивают безопасность данных, т. е. возможно выполнение следующих операций:

- **шифрование прикладных программ;**
- **шифрование данных;**
- **защиту паролем;**
- **ограничение уровня доступа (к БД, к таблице).**

Возможности СУБД

Доступ к данным посредством языка SQL. Язык запросов SQL реализован в целом ряде популярных СУБД для различных типов ЭВМ либо как базовый, либо как альтернативный. В силу своего широкого использования является международным стандартом языка запросов.

