Информатика для СПО

Базы данных. Системы управления базами данных (СУБД). Основные понятия.

Базы данных и СУБД (системы управления базами данных). Основные понятия. Объекты

Большие массивы данных вместе с *программно-аппаратными* средствами для их обработки называют информационными системами (ИС)

В основе любой *информационной системы* лежит *база данных*

***База данных-* совокупность *данных*, организованных по определенным *правилам*, предусматривающая *общие принципы* описания, хранения и манипулирования данными, независимая от прикладных программ (ГОСТ 20886-85)**

Характерный *признак* баз данных: *Базы данных* – набор данных находящихся *под управлением СУБД*

Системы управления базами данных (СУБД)

СУБД - это ***специализированные программные продукты***, позволяющие:

1) постоянно хранить ***сколь угодно большие*** (но ***не бесконечные***) объемы данных

2) ***извлекать и изменять*** эти хранящиеся данные в том или ином аспекте, используя при этом так называемые ***запросы***

3) создавать ***новые базы данных***, т. е. описывать ***логические структуры*** данных и задавать их ***структуру***, т. е. предоставляют ***интерфейс программирования***

4) ***обращаться*** к хранящимся ***данным*** со стороны ***нескольких*** пользователей ***одновременно*** (т. е. предоставляют ***доступ*** к механизму управления ***транзакциями***)

Системы управления базами данных (СУБД)

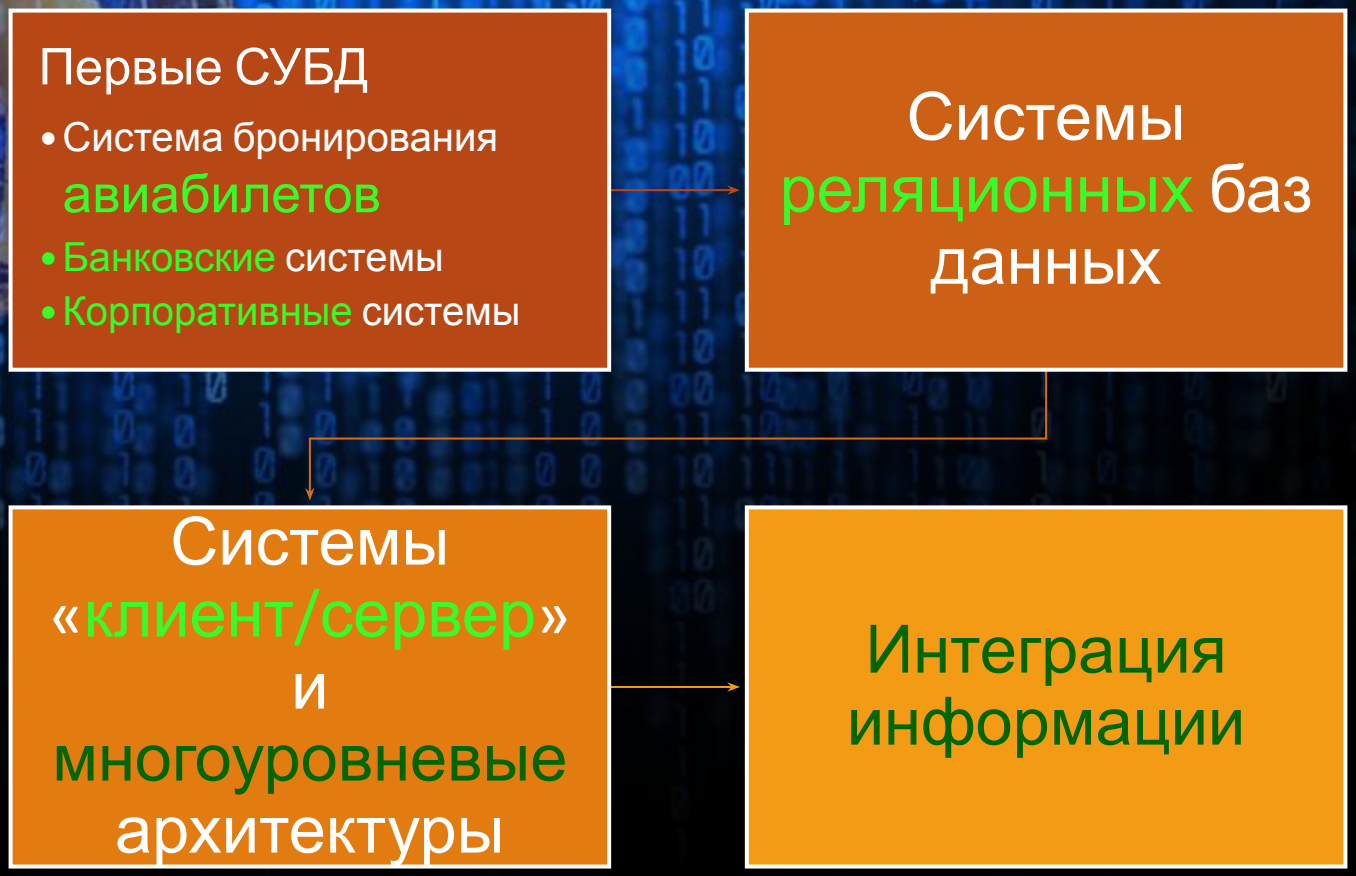
Изначально СУБД были основаны на иерархических и сетевых моделях данных, т. е. позволяли работать только с древовидными и графовыми структурами. В процессе развития в 1970 г. появились системы управления базами данных, предложенные Коддом (Codd), основанные на *реляционной модели данных*

Сейчас системы управления базами данных являются наиболее сложными программными продуктами на рынке и составляют его основу. В дальнейшем предполагается вести разработки по сочетанию обычных систем управления базами данных с объектно-ориентированным программированием (ООП) и интернет-технологиями

Что такое *база данных*?

По существу, это не что иное, как набор порций информации, существующий в течение длительного периода времени.

Термином базы данных (database) в соответствии с принятой традицией обозначают набор данных, находящихся под контролем СУБД.



Эволюция систем баз данных

Первые СУБД

Появление первых коммерческих систем управления базами данных датируется концом 1960-х годов. Непосредственным предшественником таких систем были файловые системы, удовлетворяющие только некоторым из требований, предъявляемых к современным СУБД.

Первые СУБД

Файловые системы действительно пригодны для хранения обширных фрагментов данных в течение длительного времени, но они не способны гарантировать, что данные, не подвергшиеся резервному копированию, не будут испорчены или утрачены, и не поддерживают эффективные инструменты доступа к элементам данных, положение которых в определенном файле заранее не известно.

Первые СУБД

К числу первых серьезных программных приложений СУБД относились те, в которых предполагалось, что данные состоят из большого числа элементов малого объема; для обработки таких элементов требовалось выполнить множество элементарных запросов и операций модификации.

Система подобного типа имеет дело со следующими элементами данных:

1) системы о резервировании конкретным пассажиром места на определенный авиа рейс, включая информацию о номере места и обеденном меню;

2) информацией о полетах – аэропортах отправления и назначения для каждого рейса, сроках отправки и прибытия, принадлежности воздушных судов тем или иным компаниям, экипажам и т.п.;

3) данными о ценах на авиабилеты, поступивших заявках и наличии свободных мест.

Системы бронирования авиабилетов

Системы бронирования авиабилетов

В типичных запросах требуется выяснить, какие рейсы из одного заданного аэропорта в другой близки по времени отправления к указанному календарному периоду, имеются ли свободные места и какова стоимость билета.

К числу характерных операций по изменению данных относятся бронирование места на рейс, определение номера места и выбор обеденного меню.

В любой момент к одним и тем же элементам данных могут обращаться несколько операторов-кассиров. СУБД обязана обеспечить подобную возможность, но исключить любые потенциальные проблемы, связанные, например, с продажей нескольких билетов на одно место, а также предотвратить опасные потери записей данных, если система внезапно выйдет из строя.

Банковские системы

Базы данных банковских систем содержат информацию об именах и адресах клиентов, лицевых счетах, кредитах, остатках и оборотах денежных средств, а также о связях между элементами бухгалтерской и персональной информации, т.е. о том, кто из клиентов владеет теми или иными счетами, кредитами и т.п. Весьма распространёнными являются запросы об остатке на счете, а также операции по изменению его состояния, связанные с приходом и расходом средств.

Как и в ситуации с системой бронирования авиабилетов, вполне естественной выглядит возможность одновременного доступа к информации со стороны многочисленных клиентов и служащих банков, пользующихся локальными терминалами, банкоматами или средствамиWeb. В этой связи жизненно важное значение приобретает требование о том, чтобы единовременное обращение к одному и тому же банковскому счету ни при каких условиях не приводило к потере отдельных транзакций. Какие бы то ни было ошибки в данном случае совершенно не допустимы.

Как только, например, деньги со счета выданы банкоматом, система должна немедленно сохранить информацию о выполненной расходной операции – даже тогда, когда в этот момент внезапно отключилось электропитание. Соответствующие решения, обладающие требуемым уровнем надежности, далеко не очевидны и могут быть отнесены к разряду «фигур высшего пилотажа» в сфере технологий СУБД.

Корпоративные системы

Многие из ранних приложений баз данных были предназначены для хранения корпоративной информации – записей о продажах и закупках, данных об остатках на счетах внутреннего бухгалтерского учета или персональные сведения о служащих компании (их имена, адреса проживания, уровнях фиксированной заработной платы, надбавках, отчислениях и т.д.). Запросы к таким базам данных позволяют получать информацию о состоянии счетов, выплатах сотрудникам и т.п.

Корпоративные системы

Каждая операция купли/продажи, прихода/расхода, приема сотрудника на работу его продвижении по службе или увольнения приводит к изменению соответствующих элементов данных.

Корпоративные системы

Самые первые СУБД, во многом унаследовавшие свойства файловых систем, оказывались способными представить результаты запросов практически только в том виде, который соответствовал структуре хранения данных.

Корпоративные системы

Одной из основных проблем, препятствовавших распространению и использованию таких моделей и систем, было отсутствие поддержки ими высокоуровневых языков запросов.

Системы реляционных баз данных

В 1970 году Э.Ф. Кодд предложил схему представления данных в виде таблиц, называемых отношениями (relations). Структуры таблиц могут быть весьма сложными, что не снижает скорости обработки самых различных запросов.

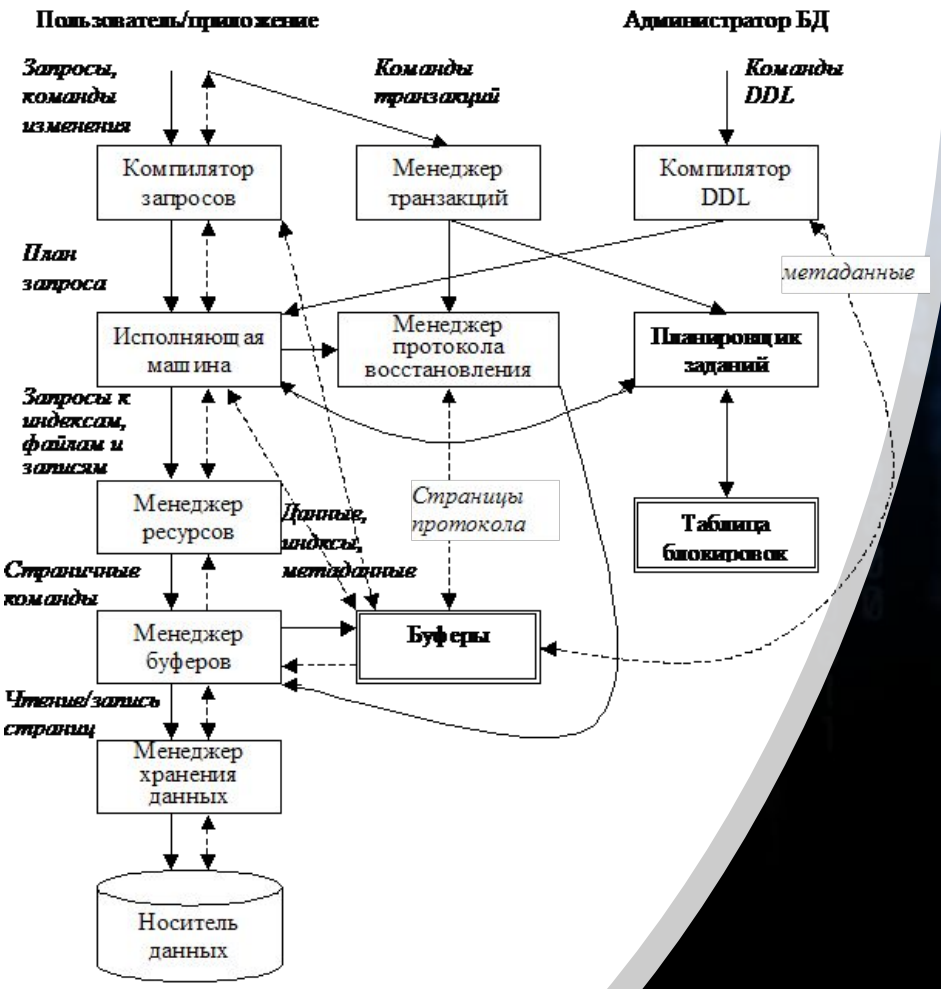
Системы реляционных баз данных

В отличие от ранних систем баз данных, рассмотренных выше, пользователю реляционной базы данных вовсе не требуется знать об особенности организации хранения информации о носителе. Запросы к такой базе данных выражаются средствами высокоуровневого языка, позволяющего значительно повысить эффективность работы программиста.

Современные тенденции развития систем баз данных

- Уменьшение и удешевление систем
- Тенденции роста систем
- Третичные устройства хранения
- Параллельные вычисления
- Системы «клиент/сервер» и многоуровневые архитектуры
- Данные мультимедиа
- Интеграция информации

Обзор структуры СУБД



Современная база данных включает в себя:

- компилятор запросов;
- исполняющую машину;
- менеджер ресурсов;
- менеджер буферов;
- менеджер хранения данных
- менеджер транзакций
- менеджер протоколирования и восстановления
- компилятор DDL
- планировщик заданий
- таблица блокировок
- и др.

Язык определения данных (Data Definition Language – DDL)

Язык определения данных решает задачи создания и удаления объектов базы данных. По стандартам SQL-92 к таким объектам относятся:

- схемы
- представления
- курсоры
- таблицы
- индексы

Каждый объект в базе однозначно описывается его именем и имеет владельца. Подавляющее число операторов DDL начинается с ключевых слов CREATE (создать) или DROP (удалить).

Обработка запросов

Запрос анализируется и оптимизируется компилятором запросов. Сформированный компилятором план запроса (query plan), или последовательность действий, подлежащих выполнению системой с целью получения ответа на запрос, передается исполняющей машине.

Обработка запросов

Исполняющая машина направляет группу запросов на получение небольших порций данных – как правило строк (кортежей) таблицы (отношения) – менеджеру ресурсов, который «осведомлен» об особенностях размещения информации в файлах данных (data files), содержащих таблицы, о форматах и размещения информации в файлах данных (data files), содержащих таблицы, о формулах и размерах записей в этих файлах и о структурах индексных файлов (index files), обеспечивающих существенное ускорение процессов поиска запрошенных данных.

Обработка запросов

Запросы на получение данных транслируется в адреса страниц и пересылаются менеджеру буферов (buffer manager). Менеджер буферов предназначен для обращения к соответствующим порциям данных на носителях вторичных устройств хранения, где они размещены постоянно с последующим переносом данных в буферы, размещаемые в оперативной памяти, и наоборот. Единицами потоков обмена данными между буферами в памяти и диском являются страница или «дисковый блок».

Обработка запросов

Чтобы получить информацию с диска менеджеру буферов приходится обращаться к услугам менеджера хранения данных (storage manager), который, решая возложенные на него задачи, может вызывать команды операционной системы, но чаще всего непосредственно инициирует инструкции дискового контроллера.

Обработка транзакций

Запросы и другие команды языка управления данными группируются в транзакции (transactions) – процессы, которые должны выполняться атомарным образом (atomically) и изолированно (in isolation) друг друга. Зачастую каждый отдельный запрос или операция по изменению данных является самостоятельной транзакцией. Транзакция обязана обладать свойством устойчивости (durability). Это значит, что результат каждой завершенной транзакции должен быть зафиксирован в базе данных даже в тех случаях, когда после окончания транзакции система по той или иной причине выходит из строя.

Обработка транзакций

В представленной схеме процессор транзакций (transaction processor) предназначен в виде двух основных компонентов:

планировщика заданий (scheduler), или менеджера параллельных заданий (concurrency-control manager), ответственного за обеспечение атомарности и изолирования транзакций;	менеджера протоколирования и восстановления (logging and recovery manager), гарантирующего выполнение требования устойчивости транзакций.

Менеджеры буферов и хранения данных

На менеджер хранения данных (storage manager) возлагается задача управления размещением информации на диске и обмена ею между диском и оперативной памятью.

Менеджер буферов (buffer manager) ответственен за разбиение доступной оперативной памяти на буферы (buffers) – участки-страницы, куда может быть помещено содержимое дисковых блоков. Все компоненты СУБД, обращающиеся к дисковой информации, взаимодействуют с буферами и менеджером буферов – либо непосредственно, либо при помощи исполняющей машины.

Обработка транзакций

Обычной практикой является оформление одной или нескольких операций над базой данных в виде транзакции (transaction) – единицы работы, которая должна быть выполнена атомарным образом и изолировано от других транзакций.

Обработка транзакций

СУБД должна удовлетворять требованию устойчивости транзакций:

результат выполнения завершенной операции не должен быть утрачен ни при каких условиях.

Обработка транзакций

Менеджер транзакций (transaction manager) воспринимает от приложения команды транзакций (transaction commands), которые свидетельствуют о начале и завершении транзакции, а также передают информацию о предпочтениях приложения в отношении параметров транзакции.

Функции процессора транзакций

- Протоколирование
- Управление параллельными заданиями
- Разрешение взаимоблокировок

Протоко-лирование

С целью удовлетворения требования устойчивости (durability) транзакций каждое изменение в базе данных фиксируется в специальных дисковых файлах.

Протоколирование

Менеджер протоколирования (logging manager) в своей работе руководствуется одной из нескольких стратегий, призванных исключить вредные последствия системных сбоев во время выполнения транзакции, а менеджер восстановления (recovery manager) в случае возникновения подобных ситуаций способен считать протокол изменений и привести базу данных в некоторое сообразное состояние.

Протоколирование

Информация протокола сначала сохраняется в буферах; затем менеджер протоколирования в определенные моменты времени взаимодействует с менеджером буферов, дабы убедиться, что содержание буферов действительно сохраняется на диске.

Управление параллельными заданиями (concurrency control)

Транзакции обязаны выполняться в полной изоляции друг от друга. Горькая истина, однако, заключается в том, что в реальных системах одновременно могут действовать несколько процессов-транзакций.

Управление параллельными заданиями (concurrency control)

Планировщик заданий (scheduler), или менеджер параллельных заданий (concurrency-control manager), должен обеспечить такой режим работы системы, чтобы результат выполнения отдельных перемежающихся во времени операций многочисленных транзакций оказался таким, как если бы транзакции в действительности инициировались, протекали и полностью завершались в строгой очередности, не «пересекаясь» одна с другой.

Управление парал-лельными заданиями (concurrency control)

Типичный планировщик заданий добивается поставленной перед ним цели, устанавливая признаки блокировки (lock) на соответствующие фрагменты содержимого базы данных.

Управление парал-лельными заданиями (concurrency control)

Блокировки препятствуют возможности единовременного обращения нескольких транзакций к порции данных такими способами, которые плохо согласуются друг с другом. Признаки блокировки обычно хранятся в таблице блокировок (lock table), размещенной в оперативной памяти.

Управление парал-лельными заданиями (concurrency control)

Планировщик заданий воздействует на процесс выполнения запросов и других операций, запрещая исполняющей машине обращаться к блокированным порциям данных.

Разрешение взаимоблокировок (deadlock resolution).

Поскольку транзакция состязаются за ресурсы, которые могут быть блокированы планировщиком заданий, возможно возникновение таких обстоятельств, когда ни одна из транзакций не в состоянии продолжить работу ввиду того, что ей необходим ресурс, находящийся в ведении другой транзакции.

Менеджер транзакций обладает прерогативой вмешиваться в ситуацию и прерывать (откатывать – «rollback») одну или несколько транзакций, чтобы позволить остальным продолжить работу.

Процессор запросов

Подсистема, в наибольшей степени определяющая показатели производительности СУБД, носит название процессора запросов (query processor).

Типичная структура процессора запросов включает в себя следующие компоненты:

Компилятор запросов (query compiler)

Исполняющая машина (execution engine)

Компилятор запросов (query compiler)

- Транслирует запрос во внутренний формат системы – план запросов (query plan). Последний описывает последовательность инструкций, подлежащих выполнению. Часто инструкции плана запроса представляет собой реализацию операций реляционной алгебры.

Компилятор запросов состоит из трех основных частей:

синтаксического анализатора запросов (query parser), создающего на основе текста запроса древовидную структуру данных;

Компилятор запросов состоит из трех основных частей:

препроцессора запросов (query preprocessor), выполняющего семантический анализ запроса (проверку того, все ли отношения и их атрибуты, упомянутые в тексте запроса действительно существуют) и функции преобразования дерева, построенного анализатором, в дерево алгебраических операторов, отвечающих исходному плану запроса;

Компилятор запросов состоит из трех основных частей:

оптимизатора запросов (query optimizer), осуществляющего трансформацию плана запроса в наиболее эффективную последовательность фактических операций надданными.

Компилятор запросов (query compiler)

Компилятор запросов, принимая решения о том, какая из последовательностей операций с большей вероятностью окажется самой оптимальной по быстродействию, пользуется метаданными и статистической информацией, накопленной СУБД.

Например, наличие индекса (index) – специальной структуры данных, обслуживающей процессы доступа к информации отношений посредством хранения определенных значений, которые соответствуют порциям содержимого отношения,- способно существенным образом повлиять на выбор наиболее эффективного плана.

Исполняю-щая машина (execution engine)

Несет ответственность за осуществление каждой из операций, предусмотренных выбранным планом запроса. В процессе своей работы она взаимодействует с большинством других компонентов СУБД.

Исполняю-щая машина (execution engine)

Чтобы получить возможность обрабатывать данные, исполняющая машина обязана считать их с носителем и перенести в буферы. При этом ей необходимо «общаться» с планировщиком заданий, чтобы избежать опасности обращения к блокированным порциям информации, а также с менеджером протоколирования, обеспечивающим гарантии того, что все изменения, внесенные в базу данных, должным образом зафиксированы в протоколе.