28. Системы базы данных. Аппаратное и программное обеспечение БД. СУБД. Уровни архитектуры БД. Реляционная модель данных. Операции над отношениями. Неопротиворечивость данных (целостный аспект БД).*База данных* - совместно используемый набор логически связанных данных. Это единое хранилище данных, которое однократно определяется, а затем используется одновременно многими пользователями.

Система управления базами данных (СУБД) - это программное обеспечение, с помощью которого пользователи могут определять, создавать и поддерживать базу данных, а также осуществлять к ней контролируемый доступ.

Система баз данных (СБД) – это компьютеризированная система хранения структурированных данных, основная цель которой – хранить информацию и предоставлять ее по требованию.

Системы БД существуют и на малых, менее мощных компьютерах, и на больших, более мощных. На больших применяют в основном многопользовательские системы, на малых – однопользовательские.

Однопользовательская система (single-user system) – это система, в которой в одно и то же время к БД может получить доступ не более одного пользователя.

Многопользовательская система (multi-user system) - это система, в которой в одно и то же время к БД может получить доступ несколько пользователей.

Основная задача большинства многопользовательских систем – позволить каждому отдельному пользователю работать с системой как с однопользовательской.

Различия однопользовательской и многопользовательской систем – в их внутренней структуре, конечному пользователю они практически не видны.

Система баз данных содержит четыре основных элемента: ***данные, аппаратное обеспечение, программное обеспечение*** и***пользователи***.

Данные в БД являются *интегрированными* и *общими*.

- ***Интегрированные*** – значит, данные можно представить как объединение нескольких, возможно перекрывающихся, отдельных файлов данных. (Например, имеется файл, содержащий данные о студентах – фамилию, имя, отчество, дату рождения, адрес и т.д., а другой – о спортивной секции. Необходимые данные о студентах, посещающих секцию, можно получить путем обращения к первому файлу.)
- ***Общие*** – значит, отдельные области данных могут использовать различные пользователи, т.е. каждый из этих пользователей может иметь доступ к одной и той же области данных, даже одновременно. (Например, одни и те же данные БД о студентах может одновременно использовать студенческий отдел кадров и деканат.)

1. Аппаратное и программное обеспечение БД.

К **аппаратному обеспечению** относятся:

- Накопители для хранения информации вместе с подсоединенными устройствами ввода-вывода, каналами ввода-вывода и т.д.
- Процессор (или процессоры) вместе с основной памятью, которая используется для поддержки работы программного обеспечения системы.

Физическая модель данных зависит от выбранной СУБД. Например, если вы планируете использовать СУБД Oracle, то физическая база данных будет состоять из файлов данных, областей таблиц, сегментов отката, таблиц, столбцов и индексов.

К реляционным СУБД относится целый ряд программных продуктов, среди них*Microsoft Access из пакета Microsoft Office, MySQL* или более мощные системы промышленного уровня, таких как *Microsoft SQL Server* или *Oracle*.

Примеры объектных *СУБД*: Cache, GemStone (от Servio Corporation), ONTOS (ONTOS).
Примеры реляционных *СУБД*: *MySql,* PostgreSql.

2. СУБД.

Для работы с данными используются системы управления базами данных (СУБД). Основные функции СУБД:

- определение данных (описание структуры баз данных);
- обработка данных;
- управление данными.

Разработка структуры БД - важнейшая задача, решаемая при проектировании БД. Структура БД (набор, форма и связи ее таблиц) - это одно из основных проектных решений при создании приложений с использованием БД. Созданная разработчиком структура БД описывается на языке определения данных СУБД.

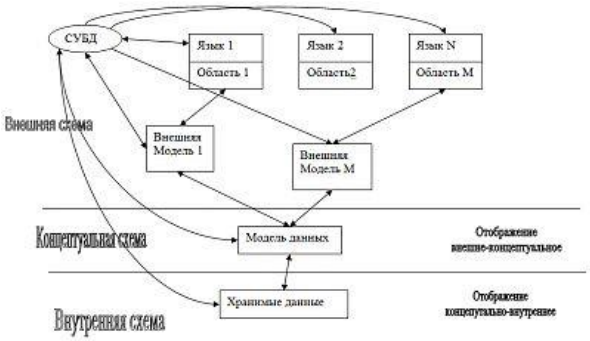
Любая СУБД позволяет выполнять следующие операции с данными:

- добавление записей в таблицы;
- удаление записей из таблицы;
- обновление значений некоторых полей в одной или нескольких записях в таблицах БД;
- поиск одной или нескольких записей, удовлетворяющих заданному условию.

Для выполнения этих операций применяется механизм запросов. Результатом выполнения запросов является либо отобранное по определенным критериям множество записей, либо изменения в таблицах. Запросы к базе формируются на специально созданном для этого языке, который так и называется «язык структурированных запросов» (SQL - Structured Query Language).

Под управлением данными обычно понимают защиту данных от несанкционированного доступа, поддержку многопользовательского режима работы с данными и обеспечение целостности и согласованности данных.

3. Уровни архитектуры БД.



Архитектура СУБД должна обеспечивать, в первую очередь, разграничение пользовательского и системного уровней. В настоящее время чаще всего поддерживается трехуровневая архитектура описания БД с тремя уровнями абстракции, на которых можно рассматривать базу данных. Такая архитектура включает: внешний уровень, внутренний урове¬нь, концептуальный уровень.Описание структуры данных на любом уровне называется схемой.

Основным назначением трехуровневой архитектуры является обеспечение незави¬симости от данных. Суть этой независимости заключается в том, что изменения на нижних уровнях никак не влияют на верхние уровни. Различают два типа независимости от данных: логиче¬скую (означает полную защищенность внешних схем от изменений, вносимых в концептуальную схему) и физическую (защищенность концептуальной схемы от изменений, вносимых во внутреннюю схему).

На внешнем уровне пользователи воспринимают данные, где отдельные группы пользователей имеют свое представление (ПП) на базу данных. Каждый тип пользователей может применять для работы с БД свой язык общения. Конечные пользователи употребляют либо язык запросов, либо специальный язык, поддерживаемый приложениями и вызывающий определенные для пользователя экранные формы и пользовательские меню. Прикладные программисты чаще применяют либо языки высокого уровня, например, C, Pascal и так далее, либо специальные языки СУБД.

Концептуальный уровень является промежуточным уровнем в трехуровневой архитектуре и обеспечивает представление всей информации базы данных в абстрактной форме. Описание базы данных на этом уровне называется концептуальной схемой, которая включает объекты и их атрибуты, связи между объектами, ограничения, накладываемые на данные, семантическую информацию о данных, обеспечение безопасности и поддержки целостности данных. Концептуальная схема — это единое логическое описание всех элементов данных и отношений между ними, логическая структура всей базы данных.

Внутренняя схема описывает физическую реализацию базы данных и предна¬значена для достижения оптимальной производительности и обеспечения экономно¬го использования дискового пространства. На внутреннем уровне осуществляется взаимодействие СУБД с ме¬тодами доступа операционной системы с целью размещения данных на запоминающих уст-ройствах, создания индексов, извлечения данных и т. д. На внутреннем уровне хранится следующая информация: распределение дискового пространства для хранения данных и индексов, описание подробностей сохранения записей (с указанием реальных разме¬ров сохраняемых элементов данных), сведения о размещении записей, сведения о сжатии данных и выбранных методов их шифрования. Ниже внутреннего уровня находится физический уровень, кото¬рый контролируется операционной системой, но под руководством СУБД. Физический уровень учитывает, каким образом данные будут представлены в машине.

Реализация трехуровневой архитектуры БД требует, чтобы СУБД переводила информацию с одного уровня на другой, то есть преобразовывала адреса и указатели в соответствующие логические имена и отношения и наоборот. Выгодой такого перевода является независимость логического и физического представления данных, но и плата за эту независимость не малая — большая системная задержка. Типичная алгоритм управления в управлениях данной схемы:
1. Пользователь выдает запрос на доступ, используя конкретный подъязык данных.
2. СУБД воспринимает запрос и интерпретирует его
3. СУБД обследует по очереди внешнюю схему, отображение внешне-концепутальное, концепутальную схему, отображение концептуально-внутреннее и определяет структуру хранения
4. СУБД выполняет необходимые операции над хранимой БД

Построение БД по этой схеме, предполагает, что интерфейс пользователя является границей, за которой пользователь ничего не видит. В частности одним из признаков хорошей базы данных написанной в АКСЦЕСС является то, что все свои задачи пользователь решает только через формы и не обращается к конструкторам.

4. Реляционная модель данных.

Почти все современные системы основаны на **реляционной** (relational) модели управления базами данных. Название **реляционная**связано с тем, что каждая запись в такой базе данных содержит информацию, относящуюся только к одному конкретному объекту.

В **реляционной** СУБД все обрабатываемые данные представляются в виде плоских таблиц. Информация об объектах определенного вида представляется в табличном виде: в столбцах таблицы сосредоточены различные атрибуты объектов, а строки предназначены для сведения описаний всех атрибутов к отдельным экземплярам объектов.

Модель, созданная на этапе инфологического моделирования, в наибольшей степени удовлетворяет принципам реляционности. Однако для приведения этой модели к реляционной необходимо выполнить процедуру, называемую **нормализацией**.

Теория нормализации оперирует с пятью **нормальными формами**. Эти формы предназначены для уменьшения избыточности информации, поэтому каждая последующая нормальная форма должна удовлетворять требованиям предыдущей и некоторым дополнительным условиям. При практическом проектировании баз данных четвертая и пятая формы, как правило, не используются. Мы ограничились рассмотрением первых четырех нормальных форм.

Введем понятия, необходимые для понимания процесса приведения модели к реляционной схеме.

Отношение - абстракция описываемого объекта как совокупность его свойств. Проводя инфологический этап проектирования, мы говорили об абстракции объектов и приписывали им некоторые свойства. Теперь же, проводя концептуальное проектирование, мы переходим к следующему уровню абстракции. На данном этапе объектов, как таковых, уже не существует. Мы оперируем совокупностью свойств, которые и определяют объект.

Экземпляр отношения - совокупность значений свойств конкретного объекта.

Первичный ключ - идентифицирующая совокупность атрибутов, т.е. значение этих атрибутов уникально в данном отношении. Не существует двух экземпляров отношения содержащих одинаковые значения в первичном ключе.

Простой атрибут - атрибут, значения которого неделимы.

Сложный атрибут - атрибут, значением которого является совокупность значений нескольких различных свойств объекта или несколько значений одного свойства.

Требования к реляционным моделям

Рациональные варианты концептуальной схемы базы данных должны удовлетворять третьей нормальной форме, а также следующим требованиям:

- Выбранный перечень отношений должен быть минимален. Отношение используется, если только его необходимость обусловлена задачами.
- Выбранный перечень атрибутов должен быть минимален. Атрибут включается в отношение только в том случае, если он будет использоваться.
- Первичный ключ отношения должен быть минимальным. То есть невозможно исключить ни один атрибут из идентифицирующей совокупности атрибутов, не нарушив при этом однозначной идентификации.
- При выполнении операций над данными не должно возникать трудностей.

Графическая интерпретация реляционной схемы

Концептуальная модель, реализованная в виде реляционной схемы, имеет свои правила графического представления.

- Отношение представляется в виде полоски, содержащей имена всех атрибутов. Имя отношения пишется над ней.
- Первичный ключ отношения должен быть выделен жирной рамкой.
- Связи, определенные между отношениями, должны быть показаны линиями, проведенными между связующими атрибутами. Значения экземпляров связующих атрибутов должны совпадать.

Операции над отношениями.

Для манипулирования отношениями используют операции реляционной алгебры. Отношения реляционной алгебры - это множества, поэтому средства работы с отношениями базируются на традиционных операциях теории множеств, которые дополняются некоторыми операциями, специфичными для баз данных. Чаще всего выделяют следующие операции реляционной алгебры:

- объединение отношений;
- пересечение отношений;
- разность отношений;
- произведение отношений;

- деление отношений;
- ограничение отношения;
- проекция отношения;
- соединение отношений.

Кроме перечисленных выше, в СУБД, как правило, реализуются так же операция присваивания, позволяющая сохранять в базе данных результаты обработки, операция переименования атрибутов и операция агрегации.

Не сильно вдаваясь в детали, операции реляционной алгебры могут быть описаны следующим образом.

1)Операция объединения двух отношений позволяет создать отношение, включающее все строки отношений-операндов. Отношения операнды должны иметь одинаковый набор атрибутов.

2) Результат операции пересечения отношений - отношение, содержащее строки, которые входят одновременно в оба отношения-операнда. Отношения-операнды должны иметь одинаковый набор атрибутов.

3) Разность отношений используется для выделения строк, которые входят в первое отношение-операнд и не входят во второе. Операнды должны иметь одинаковый набор атрибутов.

4) При выполнении операции произведения двух отношений каждая строка первого отношения-операнда сцепляется (конкатенируется) с каждой строкой второго отношения-операнда. Сцепленные строки образуют отношение-результат. Число строк в отношении-результате равно произведению числа строк в отношениях-операндах. Множества атрибутов отношений-операндов не должны пересекаться.

5) Операция деления "обратна" операции умножения. Описать ее в "житейских" терминах довольно сложно, поэтому придется прибегнуть к некоторым обозначениям. Пусть имеются два отношения: делимое A с атрибутами $\{a_r\ a_v \dots a_{if}\ b_r\ b_v \dots b_m\}$ и делитель B. Результат деления A на B - отношение C с атрибутами $\{ a_r\ a_v \dots a_{if} \}$. Формируется результат так. Если в строке отношения A атрибуты $b_r\ b_v \dots b_m$ совпадают с одной из строк отношения B, то эти атрибуты $(b_r\ b_v \dots b_m\ \}$ отсекаются и строка без них включается в отношение-результат C.

6) Операция ограничения (селекции) - это выбор из отношения подмножества кортежей, удовлетворяющих заданному условию. Например, если задано отношение R[XYZ], то операция селекции определяется как:

Sel (R) = { R[XYZ] / R.B = b }, где B - атрибут R. B=b

Пример операции селекция: вывести список служащих ростом 175 из отношения, приведенного на рис. 1.4. Sel (EHW)= EHW.РосТ=175 Результат:

Номер Рост Вес;
101 175 95
303 175 80
801 175 87

7) Операция проекции позволяет выбрать из отношения-операнда определенные столбцы (атрибуты отношения), исключая повторения. Например, для отношения R[XYZ] операция проекции определяется как:

Pr (R)={R.B|, где B - множество атрибутов R. B

Пример операции проекция: вывести веса все служащих из отношения, приведенного на Результате выше. Pr (EHW)= EHW. Вес атрибута A (обозначается A->B), если в любой момент времени каждому значению атрибута A соответствует не более одного значения атрибута B. Обратите внимание, что термин функциональная зависимость соответствует понятию функции в математике. Если неключевой атрибут зависит от всего составного ключа и не зависит от его частей, то говорят о полной функциональной зависимости атрибута от составного ключа. Если атрибут A зависит от атрибута B, а B зависит от атрибута C, но обратная зависимость отсутствует, то говорят, что атрибут C зависит от A транзитивно.

5. Неопротиворечивость данных (целостный аспект БД).

Применение СУБД для работы с интегрированными БД выявило особую важность проблемы целостности БД. Под целостностью БД понимают правильность и непротиворечивость ее содержимого. Нарушение целостности может быть вызвано, например, ошибками и сбоями, так как в этом случае система не в состоянии обеспечить нормальную обработку или выдачу правильных данных.

Рассмотрим два аспекта целостности – на уровне отдельных объектов и операций и на уровне базы данных в целом.

Первый аспект целостности обеспечивается на уровне структур данных и отдельных операторов языковых

средств СУБД. При нарушении такой целостности (например, ввод значения больше 10 в столбец «Семестр» таблицы «Учебный план» БД «Сессия») соответствующий оператор отвергается.

Некоторые ограничения целостности не нужно выражать в ясном виде, поскольку они встроены в структуры данных. Например, в СУБД, поддерживающей структуры, составленные из записей, каждый экземпляр записи в БД должен отображать спецификацию типа записи. Это означает, что все поля, специфицированные в описании типа, должны быть представлены в каждом экземпляре записи, а значение, заносимое в отдельное поле, должно иметь соответствующий описанию тип данных.

Часто же база может иметь такие ограничения целостности, которые требуют обязательного выполнения не одной, а нескольких операций. Для иллюстрации примеров рассмотрим функциональные возможности учебной БД «Сессия», добавив в таблицу «Кадровый состав» столбец Нагрузка для решения дополнительной задачи – расчета общей годовой нагрузки преподавателей (в часах учебной работы). Тогда любая операция по внесению изменений или добавления данных в столбец ID_Преподаватель таблицы «Учебный план» должна сопровождаться соответствующим изменением данных в столбце Нагрузка. Если после внесения изменений в столбец ID_преподаватель произойдет сбой, то БД окажется в нецелостном состоянии.

Для обеспечения целостности в случае ограничений на базу данных, а не какие-либо отдельные операции, служет аппарат транзакций.

Транзакция – неделимая с точки зрения воздействия на БД последовательность операторов манипулирования данными (чтения, удаления, вставки, модификации), такая, что:
1) либо результаты всех операторов, входящих в транзакцию, отображаются в БД;
2) либо воздействие всех операторов полностью отсутствует.
При этом для поддержания ограничений целостности на уровне БД допускается их нарушение внутри транзакции так, чтобы к моменту завершения транзакции условия целостности были соблюдены.

Для обеспечения контроля целостности каждая транзакция должна начинаться при целостном состоянии БД и должна сохранить это состояние целостным после своего завершения. Если операторы, объединенные в транзакцию, выполняются, то происходит нормальное завершение транзакции, и БД переходит в обновленное (целостное) состояние. Если же происходит сбой при выполнении транзакции, то происходит так называемый откат к исходному состоянию БД.

Модели транзакций. Рассмотрим две модели транзакций, используемые в большинстве коммерческих СУБД: модель автоматического выполнения транзакций и модель управляемого выполнения транзакций, обе основаны на инструкциях языка SQL – COMMIT и ROLLBACK.

Автоматическое выполнение транзакций.
В стандарте ANSI/ISO зафиксировано, что транзакция автоматически начинается с выполнения пользователем или программой первой инструкции SQL. Далее происходит последовательное выполнение инструкций до тех пор, пока транзакция не завершается одним из двух способов:
• инструкцией COMMIT, которая выполняет завершение транзакции: изменения, внесенные в БД, становятся постоянными, а новая транзакция начинается сразу после инструкции COMMIT;
• инструкцией ROLLBACK, которая отменяет выполнение текущей транзакции и возвращает БД к состоянию начала транзакции, новая транзакция начинается сразу после инструкции ROLLBACK.
Такая модель создана на основе модели, принятой в СУБД DB2.

Управляемое выполнение транзакций.
Отличная от модели ANSI/ISO модель транзакций используется в СУБД Sybase, где применяется диалект Transact-SQL, в котором для обработки транзакций служат четыре инструкции:
• инструкция BEGIN TRANSACTION сообщает о начале транзакции, т.е. начало транзакции задается явно;
• инструкция COMMIT TRANSACTION сообщает об успешном выполнении транзакции, но при этом новая транзакция не начинается автоматически;
• инструкция SAVE TRANSACTION позволяет создать внутри транзакции точку сохранения и присвоить сохраненному состоянию имя точки сохранения, указанное в инструкции;
• инструкция ROLLBACK отменяет выполнение текущей транзакции и возвращает БД к состоянию, где была выполнена инструкция SAVE TRANSACTION (если в инструкции указана точка сохранения – ROLLBACK TO имя_точки_сохранения), или к состоянию начала транзакции.