Информатика и информационно-коммуникационные технологии

Сафарьян Ольга Александровна

Лекция 6

ОСНОВНЫЕ ПОНЯТИЯ БАЗ ДАННЫХ

1. Задачи, решаемые с помощью баз данных
2. Классификация БД
3. Реляционная модель данных
4. Свойства полей базы данных
5. Объектно-ориентированные языки
6. Типы данных
7. Безопасность и объекты баз данных
8. Проектирование баз данных

Задачи, решаемые с помощью баз данных

Информационная система представляет собой аппаратно-программный комплекс, обеспечивающий выполнение следующих функций:

- ввод данных об объектах некоторой предметной области;
- надежное хранение и защита данных во внешней памяти вычислительной системы;
- дополнение, удаление, изменение данных;
- сортировка, выборка данных по запросам пользователей;
- выполнение специфических для данной предметной области преобразований информации;
- предоставление пользователям удобного интерфейса;
- обобщение данных и составление отчетов.

Совокупность взаимосвязанных данных, называемых *структурой* данных.

Совокупность структурированных данных, относящихся к одной предметной области, называется *базой данных* (БД). БД – организованная структура, предназначенная для хранения информации. Совокупность программ, реализующих в БД функции ИС в удобной для пользователя форме, называется системой управления базой данных (СУБД).

СУБД – комплекс программ, предназначенных для создания структуры новой базы, *ведению* данных и *визуализации* информации.

Ведение (сопровождение, поддержка) данных – термин, объединяющий действия по добавлению, удалению или изменению хранимых данных.

Под *визуализацией* информации БД понимается отбор отображаемых данных в соответствии с заданным критерием, упорядочение, оформление и последующая выдача на устройство вывода или передача по каналам связи.

Классификация БД

По *технологии обработки* данных БД подразделяются на *централизованные* и *распределенные*.
Централизованная БД хранится целиком в памяти одной вычислительной системы. Если система входит в состав сети, то возможен доступ к этой БД других систем.
Распределенная БД состоит из нескольких, возможно пересекающихся или дублирующих друг друга БД, хранимых в памяти разных вычислительных систем, объединенных в сеть.
По *способу доступа к данным* БД различают *локальный* (автономный) и *удаленный* (сетевой) доступ.
Локальный доступ предполагает, что СУБД обрабатывает БД, которая хранится на том же компьютере.
Удаленный доступ – это обращение к БД, которая хранится на одном из компьютеров, входящих в компьютерную сеть.

Классификация БД

Удаленный доступ может быть выполнен по принципу *файл-сервер* или *клиент-сервер*.

Архитектура *файл-сервер* (толстый клиент) предполагает выделение одного из компьютеров сети (сервер) для хранения централизованной БД. Все остальные компьютеры сети (клиенты) исполняют роль рабочих станций, которые копируют требуемую часть централизованной БД в свою память, где и происходит обработка. Однако при большой интенсивности запросов к централизованной БД увеличивается нагрузка на каналы сети, что приводит к снижению производительности ИС в целом.

Транзакция – элементарная операция по обработке данных, имеющую фиксированные начало, конец (успешное или неуспешное завершение) и ряд других характеристик

Классификация БД

Архитектура *клиент-сервер* (тонкий клиент) предполагает, что сервер, выделенный для хранения централизованной БД, дополнительно производит обработку клиентских запросов. Клиенты получают по сети уже обработанные данные. Учитывая широкое распространение БД в самых различных областях, в последнее время архитектура клиент-сервер применяется и на одиночных вычислительных системах.

Многозвенная архитектура предполагает наличие *сервера приложений*, на котором выполняется вся вычислительная работа, что позволяет уменьшить нагрузку на сервер БД.

Ещё более высокой надёжностью обладает *распределенная* архитектура, в которой вычислительная система состоит из нескольких компонент, распределённых по разным серверам. Специальные программы-мониторы следят за корректностью работы каждого из компонентов и, при необходимости, запускают дублирующие компоненты на других компьютерах.

Реляционная модель данных

Реляционная (от английского слова relation – отношение) модель данных является наиболее универсальной, к ней могут быть сведены другие модели (иерархическая и сетевая).

Важнейшим понятием реляционных моделей данных является *сущность*. Сущность — это объект любой природы, данные о котором хранятся в БД. Для представления данных об объектах и их взаимосвязях используются *отношения*. Каждое отношение – это реляционная таблица. Каждый конкретный экземпляр сущности представляется совокупностью элементов *строки*, которая называется *кортежем* (или *записью*). Каждый столбец есть *домен* (альтернативные названия – *атрибут* и *поле*) по имени которого группируются данные различных экземпляров сущности. Строка заголовков называется *схемой отношения*. Количество доменов определяет *степень* отношения, количество кортежей — его *мощность*.

Каждая реляционная таблица должна обладать следующими свойствами:

- один элемент таблицы — один элемент данных;
- все столбцы таблицы содержат однородные по типу данные (целочисленный, числовой, текстовый, и т.д.);
- каждый столбец имеет уникальное имя;
- число столбцов задается при создании таблицы;
- порядок записей в отношении может быть произвольным;
- записи не должны повторяться;
- количество записей в отношении не ограничено.

Структура простейшей базы, состоящей из одной таблицы, представлена *полями* (столбцами) и *записями* (строками). Даже если в базе нет ни одной записи (пустая база), это всё равно полноценная база – в ней содержится информация о методах хранения данных, хотя сами данные пока отсутствуют – её структура представлена набором полей.

Первичным ключом отношения называется поле или группа полей, однозначно определяющие запись.

Для связи реляционных таблиц необходимо ввести в обе таблицы одинаковые по типу поля, по которым определится связь между записями обеих таблиц.

Связи бывают нескольких типов:

• один к одному (1:1) – любая запись одной таблицы может быть связана только с одной записью другой и наоборот.

По сути, каждая пара записей является одной записью, поля которой разделены на две таблицы. То есть часть полей находится в одной таблице, а оставшаяся часть – в другой, являющейся продолжением первой;

• один ко многим (1:M или 1:∞) – любая запись одной таблицы может быть связана с несколькими записями другой, но любая запись второй таблицы связана только с одной записью первой таблицы.

• многие ко многим (M:M или ∞:∞) – любая запись одной таблицы может быть связана с несколькими записями другой и наоборот. В явном виде эта связь может не поддерживаться, обычно она организуется путём создания дополнительных таблиц.

Свойства полей базы данных

Рассмотрим основные *свойства* полей БД на примере СУБД Microsoft Access:

• имя поля – идентификатор, по которому происходит обращение к данным этого поля при автоматических операциях с базой (используется в качестве заголовка по умолчанию);

• подпись – определяет заголовок столбца, отличный от имени поля;

• тип поля – определяет тип данных, содержащихся в данном поле;

• обязательное поле – свойство, определяющее обязательность ввода данных;

• пустые строки – в отличие от предыдущего свойства, разрешает ввод пустых строк для некоторых (например, текстовых) типов данных;

• размер поля – задаёт предельную длину (в символах) данных, которые могут размещаться в данном поле;

• формат поля – определяет способ форматирования данных в ячейках, принадлежащих полю;

• маска ввода – определяет форму, в которой вводятся данные в поле;

• значение по умолчанию – автоматически вводимое в поле значение при создании новой записи;

• условие на значение – ограничение, используемое для проверки правильности ввода данных;

• сообщение об ошибке – текстовое сообщение, выдаваемое при попытке ввода ошибочных данных, если задано предыдущее свойство;

• индексированное поле – свойство, ускоряющее операции поиска и сортировки записей по значениям данного поля. Возможна автоматическая проверка и исключение дублирования данных.

Типы данных

Перечислим основные типы данных, которые используются в MS Access:

• Текстовый – тип данных для хранения текста длиной до 255 символов;

• Поле MEMO – тип данных для хранения больших объемов текста (до 65535 символов). Физически текст хранится в другом месте базы данных, а в поле помещается только указатель на него;

• Числовой – тип данных для хранения чисел, формат представления которых (целые, действительные и др.) задаётся свойством Размер поля;

• Дата/время – тип данных для хранения календарных дат и времени;

• Денежный – тип данных для хранения денежных сумм. Денежный тип удобнее, чем специальная настройка формата числового типа и имеет некоторые особенности, например, округления;

• Счетчик – специальное поле для натуральных чисел с автоматическим наращиванием. Используется для естественной нумерации записей;

• Логический – тип данных для хранения логических величин (вкл/выкл);

• Поле объекта OLE – тип данных для хранения объектов OLE (например, мультимедийных). Как и в поле MEMO, содержимое хранится в специальном месте БД, иначе работа с базой была бы замедленной;

• Гиперссылка – тип данных для хранения URL адресов Web-объектов Интернета. При щелчке на ссылке запускается браузер, выполняющий загрузку и воспроизведение указанного объекта; ;

• Вложение – тип данных для хранения внешних файлов, появившийся в версии Access 2007;

• Вычисляемый – тип данных, появившийся в версии Access 2010.

Безопасность и объекты баз данных

Базы данных – это особые структуры. Информация, которая в них содержится, имеет большую, часто общественную ценность (учёт автомобилей в ГИБДД, обслуживание счетов в банках, обработка результатов ЕГЭ…). Нередко с одной и той же базой работают тысячи пользователей по всей стране. От информации, её актуальности и сохранности, может зависеть благополучие множества людей. Поэтому целостность содержимого базы не должна зависеть ни от оператора, который может забыть сохранить введённые данные, ни от перебоев в электропитании.

Проблема безопасности решается тем, что в СУБД для сохранения данных используется двойной подход. С одной стороны, изменение структуры базы, добавление или удаление таблиц и другие, так сказать, глобальные операции происходят с сохранением файла базы данных. О таких операциях СУБД предупреждает пользователя (разработчика), их не проводят с базой, находящейся в эксплуатации.

С другой стороны, операции по наполнению и редактированию содержимого базы, не затрагивающие её структуру, максимально автоматизированы, выполняются без предупреждения, в обход операционной системы. Если мы добавим или удалим записи, отредактируем хотя бы одно поле, то все изменения тут же сохранятся на диске.

Каждая СУБД реализует свои типы объектов. Перечислим основные типы объектов, которые используются в БД на примере MS Access.

Таблицы – основные объекты любой реляционной БД. В таблицах *хранится* как структура базы (поля, их типы и свойства), так и все данные (записи).

Запросы – объекты, предназначенные для *доступа* к нужным данным из одной или нескольких таблиц. Результат выполнения запроса также представляется в табличной форме. Особенность запросов на выборку состоит в том, что из нужных полей и записей базовых таблиц в оперативной памяти создаётся временная результирующая таблица, которую ещё называют моментальным снимком. Основное назначение запросов – обеспечение *удобства* и *безопасности*.

Удобство работы с запросом состоит в том, что, во-первых, пользователю предоставляется доступ не ко всем данным, а только к тем, которые удовлетворяют указанным в запросе критериям. Во-вторых, отобранные по запросу данные могут быть упорядочены по указанному в запросе принципу, а в таблицах они хранятся в *естественном* порядке по мере поступления. В-третьих, применяемые по запросу операции сортировки и фильтрации выполняются гораздо быстрее, чем с базовыми таблицами, так как результирующая таблица находится на во внешней, а в оперативной памяти.

Формы – объекты, предназначенные для удобного и безопасного (аналогично запросам) *ввода* данных. В отличие от запросов, представляющих данные в виде таблиц, формы представляют собой окна с размещенными на них полями данных и элементами управления (кнопки, надписи, счетчики и прочие). К формам применяются специальные средства оформления, наглядно отображающие существующие данные и облегчающие ввод новых. Преимущества форм можно представить, если ввод данных осуществляется с заполненных бумажных бланков. В этом случае графическое представление экранной формы повторяет оформление бланка – это заметно упрощает работу наборщика, снижает его утомление и уменьшает количество ошибок, пропущенных и неверно заполненных полей.

Отчеты – объекты, предназначенные только для *вывода* данных. Отчеты позволяют на основе таблиц и запросов сформировать наглядные документы, которые могут быть распечатаны или включены в документ другого приложения. В отчетах могут быть не только отфильтрованные и отсортированные, но и сгруппированные данные. Добавляются специальные элементы оформления, характерные для печатных документов: колонтитулы, номера страниц, время создания отчёта и т.п.

Макросы и модули – объекты, предназначенные для автоматизации работы с базой. *Макросы* состоят из последовательности внутренних команд СУБД и предназначены для автоматизации повторяющихся операций. *Модули* создаются путём программирования на языке Visual Basic for Applications и предназначены для создания новых функций, реализации нестандартных возможностей, позволяющих удовлетворить специфические требования заказчика, повысить быстродействие системы управления и уровень её защищённости.

;

Проектирование баз данных

Проектирование БД представляет собой длительный, трудоёмкий, слабо формализованный процесс, от которого зависит жизнеспособность и эффективность проектируемой базы, её способность к развитию.
Проектирование БД выполняется, как правило, коллективом разработчиков и включает следующие этапы:

- анализ предметной области;
- проектирование и кодирование;
- тестирование и сопровождение.

Анализ предметной области необходим для составления технического задания на разработку базы данных.
Проектирование баз данных осуществляется на двух уровнях – *физическом* и *логическом*. На физическом уровне решаются вопросы размещения данных на внешних носителях. Во многом эта работа выполняется СУБД автоматически без участия разработчика. На логическом уровне создаётся структура базы, начиная с построения модели данных предметной области (инфологической, то есть информационно-логической модели) и заканчивая схемой данных (описанием таблиц и связей между ними).

Рпссмотрение информационной структуры приводит к разбиению основных таблиц на более мелкие с целью устранения повторяющихся данных в записях – *нормализации*, что уменьшает объем памяти, занимаемый базой данных на диске, и обеспечивает непротиворечивость данных в БД. Процесс нормализации носит итерационный (пошаговый) характер, осуществляется методом нормальных форм. Суть метода состоит в последовательном переводе таблицы из одной нормальной формы в другую, причем каждая последующая устраняет определенный вид функциональной зависимости между полями таблицы. Всего в теории описаны шесть нормальных форм, на практике чаще всего применяются первые три.

Первая нормальная форма. Отношение называется приведенным к первой нормальной форме, если все его атрибуты неделимы. Например, отношение, содержащее поле ФИО, не приведено к первой нормальной форме, если в запросах БД требуется выделить отдельно фамилию или имя. Разработчики БД изначально строят исходные отношения так, чтобы они были в первой нормальной форме.
Вторая нормальная форма. Для приведения отношений ко второй нормальной форме введем понятие функциональной зависимости.
Функциональная зависимость полей — это зависимость, при которой в строке определенному значению ключевого поля соответствует только одно значение не ключевого поля.

Третья нормальная форма позволяет устранить транзитивную зависимость. *Транзитивная зависимость* существует в том случае, если одно из двух описательных полей зависит от ключа, а второе зависит от первого. Отношение находится в третьей нормальной форме, если оно находится во второй нормальной форме, и каждое не ключевое поле не транзитивно зависит от ключа.

Процесс нормализации заканчивается созданием схемы данных, в которой указываются все нормализованные таблицы с их полями и взаимосвязями между ними. Дальнейшая работа над проектом – *кодирование* – связана с реализацией базы в среде конкретной СУБД, выбираемой с учётом требований заказчика и намеченной архитектуры ИС.

Тестирование должен проходить любой программный продукт, тем более такой, как БД. При тестировании с использованием реальных данных обнаруживаются возможные ошибки, собираются статистические данные для определения показателей качества и надёжности созданного программного обеспечения.

Сопровождение является самым продолжительным этапом жизненного цикла любой БД. Основные действия на этом этапе сводятся к наблюдению за созданной системой и поддержке её нормального функционирования.