Билет №23.2
Основы языка структурированных запросов SQL

Кузнецов Николай

5 курс 3 группа

ФТиП МПГУ

Москва 2012

Основные понятия

- **База данных (БД)** – совместно используемый набор логически связанных данных (и их описание), предназначенный для удовлетворения информационных потребностей организации.
- **СУБД (система управления базами данных)** – программное обеспечение, с помощью которого пользователи могут определять, создавать и поддерживать базу данных, а также получать к ней контролируемый доступ.
- Управление основными потоками информации осуществляется с помощью так называемых систем управления реляционными базами данных, которые берут свое начало в традиционных системах управления базами данных. Именно объединение реляционных баз данных и клиент-серверных технологий позволяет современному предприятию успешно управлять собственными данными, оставаясь конкурентоспособным на рынке товаров и услуг.
- В реляционной модели объекты реального мира и взаимосвязи между ними представляются с помощью совокупности связанных между собой таблиц (отношений).
- Даже в том случае, когда функции СУБД используются для выбора информации из одной или нескольких таблиц (т.е. выполняется запрос), результат также представляется в табличном виде. Более того, можно выполнить запрос с применением результатов другого запроса.
- Каждая таблица БД представляется как совокупность строк и столбцов, где строки (записи) соответствуют экземпляру объекта, конкретному событию или явлению, а столбцы (поля) – атрибутам (признакам, характеристикам, параметрам) объекта, события, явления.
- В каждой таблице БД необходимо наличие первичного ключа – так именуют поле или набор полей, однозначно идентифицирующий каждый экземпляр объекта или запись. Значение первичного ключа в таблице БД должно быть уникальным, т.е. в таблице не допускается наличие двух и более записей с одинаковыми значениями первичного ключа. Он должен быть минимально достаточным, а значит, не содержать полей, удаление которых не отразится на его уникальности.

Реляционные связи между таблицами баз данных

Между двумя или более таблицами базы данных могут существовать отношения подчиненности, которые определяют, что для каждой записи главной таблицы (называемой еще родительской) возможно наличие одной или нескольких записей в подчиненной таблице (называемой еще дочерней).

Выделяют три разновидности связи между таблицами базы данных:

- "один–ко–многим" (когда одной записи родительской таблицы может соответствовать несколько записей дочерней. Связь "один–ко–многим" иногда называют связью "многие–к–одному".);
- "один–к–одному" (когда одной записи в родительской таблице соответствует одна запись в дочерней. Это отношение встречается намного реже, чем отношение "один–ко–многим");
- "многие–ко–многим" (одной записи в родительской таблице соответствует более одной записи в дочерней; одной записи в дочерней таблице соответствует более одной записи в родительской).

Технология клиент-сервис

Технология клиент-сервер означает такой способ взаимодействия программных компонентов, при котором они образуют единую систему. Как видно из самого названия, существует некий клиентский процесс, требующий определенных ресурсов, а также серверный процесс, который эти ресурсы предоставляет. Совсем необязательно, чтобы они находились на одном компьютере. Обычно принято размещать сервер на одном узле локальной сети, а клиентов – на других узлах.

В контексте базы данных клиент управляет пользовательским интерфейсом и логикой приложения, действуя как рабочая станция, на которой выполняются приложения баз данных. Клиент принимает от пользователя запрос, проверяет синтаксис и генерирует запрос к базе данных на языке SQL или другом языке базы данных, соответствующем логике приложения. Затем передает сообщение серверу, ожидает поступления ответа и форматирует полученные данные для представления их пользователю. Сервер принимает и обрабатывает запросы к базе данных, после чего отправляет полученные результаты обратно клиенту. Такая обработка включает проверку полномочий клиента, обеспечение требований целостности, а также выполнение запроса и обновление данных. Помимо этого поддерживается управление параллельностью и восстановлением.

Технология клиент-сервис

Архитектура клиент-сервер обладает рядом преимуществ:

- обеспечивается более широкий доступ к существующим базам данных ;
- повышается общая производительность системы: поскольку клиенты и сервер находятся на разных компьютерах, их процессоры способны выполнять приложения параллельно. Настройка производительности компьютера с сервером упрощается, если на нем выполняется только работа с базой данных ;
- снижается стоимость аппаратного обеспечения; достаточно мощный компьютер с большим устройством хранения нужен только серверу – для хранения и управления базой данных ;
- сокращаются коммуникационные расходы. Приложения выполняют часть операций на клиентских компьютерах и посылают через сеть только запросы к базам данных, что позволяет значительно сократить объем пересылаемых по сети данных;
- повышается уровень непротиворечивости данных. Сервер может самостоятельно управлять проверкой целостности данных, поскольку лишь на нем определяются и проверяются все ограничения. При этом каждому приложению не придется выполнять собственную проверку;
- архитектура клиент-сервер естественно отображается на архитектуру открытых систем.

Дальнейшее расширение двухуровневой архитектуры клиент-сервер предполагает разделение функциональной части прежнего, "толстого" (интеллектуального) клиента на две части. В трехуровневой архитектуре клиент-сервер "тонкий" (неинтеллектуальный) клиент на рабочей станции управляет только пользовательским интерфейсом, тогда как средний уровень обработки данных управляет всей остальной логикой приложения. Третий уровень – сервер базы данных. Эта трехуровневая архитектура оказалась более подходящей для некоторых сред – например, для сетей Internet и intranet, где в качестве клиента может выступать обычный Web-браузер.

Типы команд SQL

Реализация в SQL концепции операций, ориентированных на табличное представление данных, позволила создать компактный язык с небольшим набором предложений. Язык SQL может использоваться как для выполнения запросов к данным, так и для построения прикладных программ.

Основные категории команд языка SQL предназначены для выполнения различных функций, включая построение объектов базы данных и манипулирование ими, начальную загрузку данных в таблицы, обновление и удаление существующей информации, выполнение запросов к базе данных, управление доступом к ней и ее общее администрирование.

Основные категории команд языка SQL:

- DDL – язык определения данных;
- DML – язык манипулирования данными;
- DQL – язык запросов ;
- DCL – язык управления данными;
- команды администрирования данных;
- команды управления транзакциями

Типы команд SQL

- **Определение структур базы данных (DDL)**

Язык определения данных (Data Definition Language, DDL) позволяет создавать и изменять структуру объектов базы данных, например, создавать и удалять таблицы. Основными командами языка DDL являются следующие: CREATE TABLE, ALTER TABLE, DROP TABLE, CREATE INDEX, ALTER INDEX, DROP INDEX.

- **Манипулирование данными (DML)**

Язык манипулирования данными (Data Manipulation Language, DML) используется для манипулирования информацией внутри объектов реляционной базы данных посредством трех основных команд: INSERT, UPDATE, DELETE.

- **Выборка данных (DQL)**

Язык запросов DQL наиболее известен пользователям реляционной базы данных, несмотря на то, что он включает всего одну команду SELECT. Эта команда вместе со своими многочисленными опциями и предложениями используется для формирования запросов к реляционной базе данных.

Типы команд SQL

- **Язык управления данными (DCL - Data Control Language)**

Команды управления данными позволяют управлять доступом к информации, находящейся внутри базы данных. Как правило, они используются для создания объектов, связанных с доступом к данным, а также служат для контроля над распределением привилегий между пользователями. Команды управления данными следующие: GRANT, REVOKE.

- **Команды администрирования данных**

С помощью команд администрирования данных пользователь осуществляет контроль за выполняемыми действиями и анализирует операции базы данных ; они также могут оказаться полезными при анализе производительности системы. Не следует путать администрирование данных с администрированием базы данных, которое представляет собой общее управление базой данных и подразумевает использование команд всех уровней.

- **Команды управления транзакциями**

Существуют следующие команды, позволяющие управлять транзакциями базы данных: COMMIT, ROLLBACK, SAVEPOINT, SET TRANSACTION.

Запись SQL-операторов

Оператор SQL состоит из зарезервированных слов, а также из слов, определяемых пользователем. Зарезервированные слова являются постоянной частью языка SQL и имеют фиксированное значение. Их следует записывать в точности так, как это установлено, нельзя разбивать на части для переноса с одной строки на другую. Слова, определяемые пользователем, задаются им самим (в соответствии с синтаксическими правилами) и представляют собой идентификаторы или имена различных объектов базы данных. Слова в операторе размещаются также в соответствии с установленными синтаксическими правилами.

Идентификаторы языка SQL предназначены для обозначения объектов в базе данных и являются именами таблиц, представлений, столбцов и других объектов базы данных. Символы, которые могут использоваться в создаваемых пользователем идентификаторах языка SQL, должны быть определены как набор символов. Стандарт SQL задает набор символов, который используется по умолчанию, – он включает строчные и прописные буквы латинского алфавита (A-Z, a-z), цифры (0-9) и символ подчеркивания (_). На формат идентификатора накладываются следующие ограничения:

- идентификатор может иметь длину до 128 символов;
- идентификатор должен начинаться с буквы;
- идентификатор не может содержать пробелы.
- <идентификатор>::=<буква> {<буква>|<цифра>}[,...n]

Большинство компонентов языка не чувствительны к регистру. Поскольку у языка SQL свободный формат, отдельные SQL-операторы и их последовательности будут иметь более читаемый вид при использовании отступов и выравнивания.

Запись SQL-операторов

Язык, в терминах которого дается описание языка SQL, называется метаязыком . Синтаксические определения обычно задают с помощью специальной металингвистической символики, называемой Бэкуса-Науэра формулами (БНФ). Прописные буквы используются для записи зарезервированных слов и должны указываться в операторах точно так, как это будет показано. Строчные буквы употребляются для записи слов, определяемых пользователем.

Применяемые в нотации БНФ символы и их обозначения показаны в таблице:

Символ	Обозначение
::=	Равно по определению
|	Необходимость выбора одного из нескольких приведенных значений
<…>	Описанная с помощью метаязыка структура языка
{…}	Обязательный выбор некоторой конструкции из списка
[…]	Необязательный выбор некоторой конструкции из списка
[,…n]	Необязательная возможность повторения конструкции от нуля до n-раз

Преимущества языка SQL

Основные достоинства языка SQL заключаются в следующем:

- Стандартность – как уже было сказано, использование языка SQL в программах стандартизировано международными организациями;
- Независимость от конкретных СУБД – все распространенные СУБД используют SQL, т.к. реляционную базу данных можно перенести с одной СУБД на другую с минимальными доработками;
- Возможность переноса с одной вычислительной системы на другую – СУБД может быть ориентирована на различные вычислительные системы, однако приложения, созданные с помощью SQL, допускают использование как для локальных БД, так и для крупных многопользовательских систем;
- Реляционная основа языка – SQL является языком реляционных БД, поэтому он стал популярным тогда, когда получила широкое распространение реляционная модель представления данных. Табличная структура реляционной БД хорошо понятна, а потому язык SQL прост для изучения;
- Возможность создания интерактивных запросов – SQL обеспечивает пользователям немедленный доступ к данным, при этом в интерактивном режиме можно получить результат запроса за очень короткое время без написания сложной программы;
- Возможность программного доступа к БД – язык SQL легко использовать в приложениях, которым необходимо обращаться к базам данных. Одни и те же операторы SQL употребляются как для интерактивного, так и программного доступа, поэтому части программ, содержащие обращение к БД, можно вначале проверить в интерактивном режиме, а затем встраивать в программу;
- Обеспечение различного представления данных – с помощью SQL можно представить такую структуру данных, что тот или иной пользователь будет видеть различные их представления. Кроме того, данные из разных частей БД могут быть скомбинированы и представлены в виде одной простой таблицы, а значит, представления пригодны для усиления защиты БД и ее настройки под конкретные требования отдельных пользователей;
- Возможность динамического изменения и расширения структуры БД – язык SQL позволяет манипулировать структурой БД, тем самым обеспечивая гибкость с точки зрения приспособленности БД к изменяющимся требованиям предметной области;
- Поддержка архитектуры клиент-сервер – SQL – одно из лучших средств для реализации приложений на платформе клиент-сервер. SQL служит связующим звеном между взаимодействующей с пользователем клиентской системой и серверной системой, управляющей БД, позволяя каждой из них сосредоточиться на выполнении своих функций.

Преимущества языка SQL

Любой язык работы с базами данных должен предоставлять пользователю следующие возможности:

- создавать базы данных и таблицы с полным описанием их структуры;
- выполнять основные операции манипулирования данными, в частности, вставку, модификацию и удаление данных из таблиц ;
- выполнять простые и сложные запросы, осуществляющие преобразование данных.

Язык SQL является основой многих СУБД, т.к. отвечает за физическое структурирование и запись данных на диск, а также за чтение данных с диска, позволяет принимать SQL- запросы от других компонентов СУБД и пользовательских приложений. Таким образом, SQL – мощный инструмент, который обеспечивает пользователям, программам и вычислительным системам доступ к информации, содержащейся в реляционных базах данных.

Язык SQL – первый и пока единственный стандартный язык для работы с базами данных, который получил достаточно широкое распространение. Практически все крупнейшие разработчики СУБД в настоящее время создают свои продукты с использованием языка SQL либо с SQL-интерфейсом. В него сделаны огромные инвестиции как со стороны разработчиков, так и со стороны пользователей. Он стал частью архитектуры приложений, является стратегическим выбором многих крупных и влиятельных организаций.