Дополнительные главы баз данных

Сергей Дмитриевич Кузнецов
Институт системного программирования РАН
kuzloc@ispras.ru

Содержание курса (1)

- История стандарта языка SQL. Типы данных. Средства языка SQL для определения и изменения доменов, базовых таблиц и ограничений целостности
  - История стандарта SQL и структура языка
    - ✓ Этапы процесса стандартизации языка SQL
    - ✓ Структура языка SQL

Содержание курса (2)

- Типы данных SQL
  - Точные числовые типы, приближенные числовые типы, типы символьных и битовых строк, темпоральные типы, Булевский тип, типы коллекций, анонимные строчные типы, типы, определяемые пользователем, ссылочные типы
- Средства определения, изменения и ликвидации доменов
- Средства определения, изменения и ликвидации базовых таблиц
- Средства определения и отмены общих ограничений целостности

Содержание курса (3)

- Базовые возможности выборки данных в языке SQL
  - Общая структура оператора выборки в языке SQL
    - ✓ Семантика оператора выборки
    - ✓ Ссылки на таблицы раздела FROM
      - Табличное выражение, спецификация запроса и выражение запросов
      - Ссылки на базовые, представляемые и порождаемые таблицы
  - Представляемые таблицы, или представления

Содержание курса (4)

- Базовые возможности выборки данных в языке SQL
  - Логические выражения раздела WHERE
    - Предикат сравнения, предикат between, предикат null, предикат in, предикат like, предикат similar, предикат exists, предикат unique, предикат overlaps, предикат сравнения с квантором, предикат match, предикат distinct
  - Логические выражения раздела HAVING

Содержание курса (5)

- Базовые возможности модификации баз данных в языке SQL
  - Представления, над которыми возможны операции обновления
    - Представления, допускающие применение операций обновления, в стандарте SQL/92
    - Представления, допускающие применение операций обновления, в стандарте SQL:1999
    - Раздел WITH CHECK OPTION определения представления
    - Исторический очерк

Содержание курса (6)

- Базовые возможности модификации баз данных в языке SQL
  - Операции обновления баз данных и механизм триггеров
    - Понятие триггера в SQL:1999
    - Синтаксис определения триггеров и типы триггеров
    - Выполнение триггеров
    - Триггеры и ссылочные действия

Содержание курса (7)

- Механизмы авторизации доступа и управления подключениями, сессиями и транзакциями в языке SQL
  - Поддержка авторизации доступа к данным в языке SQL
    - ✓ Пользователи и роли
    - ✓ Использование идентификаторов пользователей и имен ролей
    - ✓ Создание и ликвидация ролей
    - ✓ Передача привилегий и ролей
    - ✓ Изменение текущих идентификаторов пользователей и имен ролей
    - ✓ Аннулирование привилегий и ролей

Содержание курса (8)

- Механизмы авторизации доступа и управления подключениями, сессиями и транзакциями в языке SQL
  - Управление транзакциями в SQL
    - ✓ Порождение транзакций в SQL
    - ✓ Уровни изоляции SQL-транзакции
    - ✓ Завершение транзакций
    - ✓ Транзакции и ограничения целостности
    - ✓ Точки сохранения

Содержание курса (9)

- Механизмы авторизации доступа и управления подключениями, сессиями и транзакциями в языке SQL
  - Подключения и сессии
    - ✓ Установление соединений
    - ✓ Операторы SQL для управления соединениями
      - Оператор CONNECT
      - Оператор SET CONNECTION
      - Оператор DISCONNECT

Лекция 1. Общее введение в SQL, типы данных и средства определения доменов

- После небольшой исторической справки и краткого введения в структуру языка SQL будут рассмотрены типы данных, допустимые в языке SQL и в SQL-ориентированных базах данных, а также языковые средства определения, изменения определения и отмены определения доменов.
- В начале лекции мы представим небольшой исторический обзор SQL
  - язык уже далеко не молод
  - в 2004 г. сообщество баз данных отметило его 30-летний юбилей
  - чтобы правильно понимать и трактовать современные варианты SQL, нужно знать историю языка хотя бы в общих чертах

Краткая история языка SQL (1)

- Язык SQL, предназначенный для взаимодействия с базами данных, появился в середине 70-х гг. (первые публикации датируются 1974 г.) и был разработан в компании IBM в рамках проекта экспериментальной реляционной СУБД System R
- Исходное название языка SEQUEL (Structured English Query Language) только частично отражало суть этого языка
- Конечно, язык был ориентирован главным образом на удобную и понятную пользователям формулировку запросов к реляционным БД.

Краткая история языка SQL (2)

- Но, в действительности, он почти с самого начала являлся полным языком БД, обеспечивающим помимо средств формулирования запросов и манипулирования БД следующие возможности:
  - средства определения и манипулирования схемой БД;
  - средства определения ограничений целостности и триггеров;
  - средства определения представлений БД;
  - средства определения структур физического уровня, поддерживающих эффективное выполнение запросов;
  - средства авторизации доступа к отношениям и их полям[67];
  - средства определения точек сохранения транзакции и выполнения фиксации и откатов транзакций

Краткая история языка SQL (3)

- В языке отсутствовали средства явной синхронизации доступа к объектам БД со стороны параллельно выполняемых транзакций: с самого начала предполагалось, что необходимую синхронизацию неявно выполняет СУБД.
- В настоящее время язык SQL реализован во всех коммерческих реляционных СУБД и почти во всех СУБД, которые изначально основывались не на реляционном подходе
- Все компании-производители провозглашают соответствие своей реализации стандарту SQL, и на самом деле реализованные диалекты SQL очень близки
- Этого удалось добиться не сразу.

Краткая история языка SQL (4)

- Наиболее близки к System R были две системы компании IBM – SQL/DS и DB2
- Разработчики обеих систем использовали опыт проекта System R, а СУБД SQL/DS напрямую основывалась на программном коде System R
- Отсюда предельная близость диалектов SQL, реализованных в этих системах, к SQL System R
- Из SQL System R были удалены только те части, которые были недостаточно проработаны (например, точки сохранения) или реализация которых вызывала слишком большие технические трудности (например, ограничения целостности и триггеры)
- Можно назвать этот путь к коммерческой реализации SQL движением сверху вниз.

Краткая история языка SQL (5)

- Другой подход применялся в таких системах, как Oracle, Informix и Sybase. Несмотря на различие в способах разработки систем, реализация SQL везде происходила «снизу вверх»
- В первых выпущенных на рынок версиях этих систем использовалось ограниченное подмножество SQL System R
- В частности, в первой известной нам реализации SQL в СУБД Oracle в операторах выборки не допускалось использование вложенных подзапросов и отсутствовала возможность формулировки запросов с соединениями нескольких отношений

Краткая история языка SQL (6)

- Особенностью большинства современных коммерческих СУБД, затрудняющей сравнение существующих диалектов SQL, является отсутствие единообразного описания языка
- Обычно описание разбросано по разным руководствам и перемешано с описанием специфических для данной системы языковых средств, не имеющих прямого отношения к SQL
- Тем не менее, можно сказать, что базовый набор операторов SQL, включающий операторы определения схемы БД, выборки и манипулирования данными, авторизации доступа к данным, поддержки встраивания SQL в языки программирования и операторы динамического SQL, в коммерческих реализациях устоялся и более или менее соответствует стандарту

Краткая история языка SQL (7)

- Деятельность по стандартизации языка SQL началась практически одновременно с появлением его первых коммерческих реализаций
- В 1982 г. комитету по базам данных Американского национального института стандартов (ANSI) было поручено разработать спецификацию стандартного языка реляционных баз данных
- Первый документ из числа имеющихся у автора проектов стандарта датирован октябрем 1985 г. и является уже не первым проектом стандарта ANSI
- Стандарт был принят ANSI в 1986 г., а в 1987 г. одобрен Международной организацией по стандартизации (ISO)
- Этот стандарт принято называть *SQL/86*

Краткая история языка SQL (8)

- Понятно, что в качестве основы стандарта нельзя было использовать SQL System R
- Во-первых, этот вариант языка не был должным образом технически проработан
- Во-вторых, его слишком сложно было бы реализовать (кто знает, как бы сложилась судьба SQL, если бы все идеи проекта System R были реализованы полностью)
- Поэтому за основу был взят диалект языка SQL, сложившийся в IBM к началу 1980-х гг.
- В сущности, этот диалект представлял собой технически проработанное подмножество SQL System R

Краткая история языка SQL (9)

- К 1989 г. стандарт SQL/86 был несколько расширен, и был подготовлен и принят следующий стандарт, получивший название ANSI/ISO *SQL/89*
- Анализ доступных документов показывает, что процесс стандартизации SQL происходил очень сложно с использованием не только научных доводов
- В результате SQL/89 во многих частях имеет чрезвычайно общий характер и допускает очень широкое толкование
- В этом стандарте полностью отсутствуют такие важные разделы, как манипулирование схемой БД и динамический SQL
- Многие важные аспекты языка в соответствии со стандартом определяются в реализации

Краткая история языка SQL (10)

- Наиболее важными достижениями стандарта SQL/89 являются четкая стандартизация синтаксиса и семантики операторов выборки данных и манипулирования данными и фиксация средств ограничения целостности БД
- Были специфицированы средства определения первичного и внешних ключей отношений и так называемых проверочных ограничений целостности, которые представляют собой подмножество немедленно проверяемых ограничений целостности SQL System R
- Средства определения внешних ключей позволяют легко формулировать требования так называемой ссылочной целостности БД
- Это распространенное в реляционных БД требование можно было сформулировать и на основе общего механизма ограничений целостности SQL System R, но формулировка на основе понятия внешнего ключа более проста и понятна

Краткая история языка SQL (11)

- Осознавая неполноту стандарта SQL, на фоне завершения разработки этого стандарта специалисты различных компаний начали работу над стандартом SQL2
- Эта работа также длилась несколько лет, было выпущено множество проектов стандарта, пока наконец в марте 1992 г. не был принят окончательный проект стандарта (SQL/92)
- Этот стандарт существенно полнее стандарта SQL/89 и охватывает практически все аспекты, необходимые для реализации приложений: манипулирование схемой БД, управление транзакциями (появились точки сохранения) и сессиями (сессия – это последовательность транзакций, в пределах которой сохраняются временные отношения), подключения к БД, динамический SQL
- Наконец, были стандартизованы отношения-каталоги БД, что вообще-то не связано непосредственно с языком, но очень сильно влияет на реализацию

Краткая история языка SQL (12)

- В 1995 г. стандарт был дополнен спецификацией интерфейса уровня вызова (Call-Level Interface – *SQL/CLI*). SQL/CLI представляет собой набор спецификаций интерфейсов процедур, вызовы которых позволяют выполнять динамически задаваемые операторы SQL
- По сути дела, SQL/CLI представляет собой альтернативу динамическому SQL
- Интерфейсы процедур определены для всех основных языков программирования: C, Ada, Pascal, PL/1 и т. д.
- Следует заметить, что стандарт SQL/CLI послужил основой для создания повсеместно распространенных сегодня интерфейсов ODBC (Open Database Connectivity) и JDBC (Java Database Connectivity).

Краткая история языка SQL (13)

- В 1996 г. к стандарту SQL/92 был добавлен еще один компонент – *SQL/PSM* (Persistent Stored Modules)
- Основная цель этой спецификации состоит в том, чтобы стандартизировать способы определения и использования хранимых процедур, т. е. специальным образом оформленных программ, включающих операторы SQL, которые сохраняются в базе данных, могут вызываться приложениями и выполняются внутри СУБД.

Краткая история языка SQL (14)

- Незадолго до завершения работ по определению стандарта SQL2 была начата разработка стандарта SQL3
- Первоначально планировалось завершить проект в 1995 г. и включить в язык некоторые объектные возможности: определяемые пользователями типы данных, поддержку триггеров, поддержку темпоральных свойств данных и т. д.
- Реально работу над новым стандартом удалось частично завершить только в 1999 г., и по этой причине (а также в связи с проблемой 2000 года) стандарт получил название SQL:1999

Краткая история языка SQL (15)

- Приведем краткую характеристику текущего состояния стандарта SQL:1999 и перспектив его развития
- Прежде всего, заметим, что каждый новый вариант стандарта языка SQL был существенно объемнее предыдущих версий. Так, если стандарт SQL/89 занимал около 600 страниц, то объем SQL/92 составлял на 300 с лишним страниц больше
- Самые первые проекты SQL3 занимали около 1500 страниц
- Это вполне естественно, потому что язык усложняется, а его спецификации становятся более детальными и точными
- Но разработчики SQL3 пришли к выводу, что при таких объемах стандарта вероятность его принятия и последующей успешной поддержки заметно уменьшается
- Поэтому было принято решение разбить стандарт на относительно независимые части, которые можно было бы разрабатывать и поддерживать по отдельности

Краткая история языка SQL (16)

- В 1999 г. были приняты пять первых частей стандарта SQL:1999
- Первая часть (SQL/Framework) посвящена описанию концептуальной структуры стандарта
- В этой части приводится развернутая аннотация следующих четырех частей и формулируются требования к реализациям, претендующим на соответствие стандарту

Краткая история языка SQL (17)

- Вторая часть SQL:1999 (SQL/Foundation) образует базис стандарта. Вводится система типов языка, формулируются правила определения функциональных зависимостей и возможных ключей, определяются синтаксис и семантика основных операторов SQL:
  - операторов определения и манипулирования схемой базы данных;
  - операторов манипулирования данными;
  - операторов управления транзакциями;
  - операторов управления подключениями к базе данных и т. д.

Краткая история языка SQL (18)

- Третью часть занимает уточненная по сравнению с SQL/92 спецификация SQL/CLI
- В четвертой части специфицируется SQL/PSM – синтаксис и семантика языка определения хранимых процедур
- Наконец, в пятой части – SQL/Bindings – определяются правила связывания SQL для стандартных версий языков программирования FORTRAN, COBOL, PL/1, Pascal, Ada, C и MUMPS

Краткая история языка SQL (19)

- В стандарт SQL:1999 должны были войти еще несколько частей. Среди них спецификации следующих средств:
  - управление распределенными транзакциями (SQL/Transaction);
  - поддержка темпоральных свойств данных (SQL/Temporal);
  - управление внешними данными (SQL/MED);
  - связывание с объектно-ориентированными языками программирования (SQL/OLB);
  - поддержка оперативной аналитической обработки (SQL/OLAP)

Краткая история языка SQL (20)

- В конце 2003 г. был принят и опубликован новый вариант международного стандарта *SQL:2003*
- Многие специалисты считали, что в варианте стандарта, следующем за SQL:1999, будут всего лишь исправлены неточности SQL:1999
- Но на самом деле, в SQL:2003 специфицирован ряд новых и важных свойств, часть из которых мы затронем в этом курсе

Краткая история языка SQL (21)

- Претерпела некоторые изменения общая организация стандарта
- Стандарт SQL:2003 состоит из следующих частей:
  - 9075-1, SQL/Framework;
  - 9075-2, SQL/Foundation;
  - 9075-3, SQL/CLI;
  - 9075-4, SQL/PSM;
  - 9075-9, SQL/MED;
  - 9075-10, SQL/OLB;
  - 9075-11, SQL/Schemata;
  - 9075-13, SQL/JRT;
  - 9075-14, SQL/XML

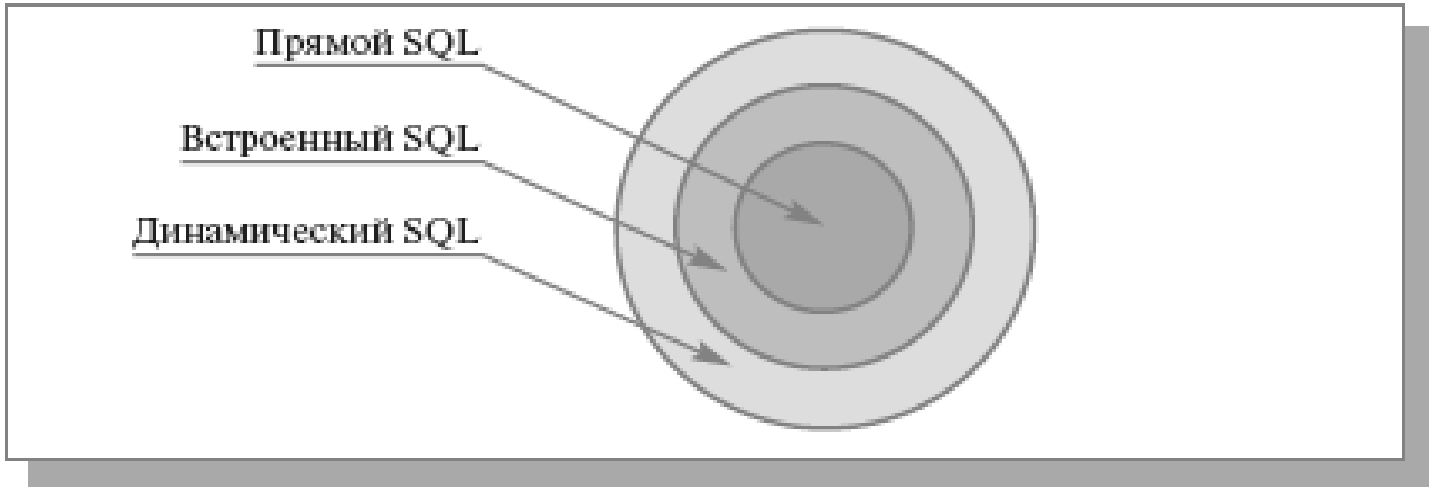
Краткая история языка SQL (22)

- Части 1-4 и 9-10 с необходимыми изменениями остались такими же, как и в SQL:1999
- Часть 5 (SQL/Bindings) перестала существовать; соответствующие спецификации включены в часть 2
- Раздел части 2 SQL:1999, посвященный информационной схеме, выделен в отдельную часть 11
- Появились две новые части – 13 и 14
- Часть 13 полностью называется «SQL Routines and Types Using the Java Programming Language» («Использование подпрограмм и типов SQL в языке программирования Java»)
  - появление такой части стандарта оправдано повышенным вниманием к языку Java со стороны ведущих производителей SQL-ориентированных СУБД
- Наконец, последняя часть SQL:2003 посвящена спецификациям языковых средств, позволяющих работать с XML-документами в среде SQL

Краткая история языка SQL (23)

- Текущее состояние процесса стандартизации языка SQL отражает текущее состояние технологии SQL-ориентированных баз данных
- Ведущие поставщики соответствующих СУБД (сегодня это компании IBM, Oracle и Microsoft) стараются максимально быстро реагировать на потребности и конъюнктуру рынка и расширяют свои продукты все новыми и новыми возможностями
- Очевидна потребность в стандартизации соответствующих языковых средств, но процесс стандартизации явно не поспевает за происходящими изменениями

Структура языка SQL (1)



- В данной лекции мы начинаем систематически описывать базовые механизмы языка SQL
- Чтобы пояснить, о какой части языка пойдет речь в этой и следующих лекциях, обратимся к рисунку

Структура языка SQL (2)

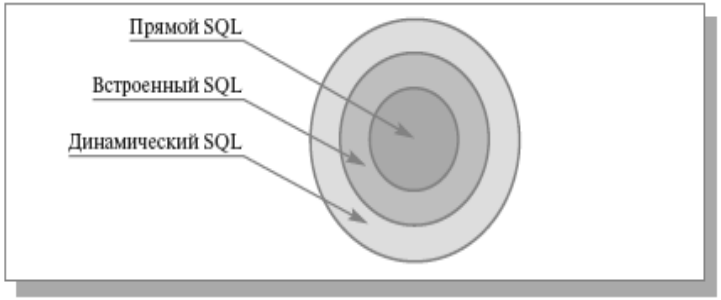
- Язык SQL, соответствующий последним стандартам SQL:2003, SQL:1999 (и даже SQL/92), это очень богатый и сложный язык, все возможности которого трудно сразу осознать и тем более понять
- Поэтому приходится разбивать язык на уровни, или слои, такие, что каждый уровень языка включает все конструкции, входящие в более низкие уровни
- В стандарте определяется несколько способов разбиения языка на уровни
- В одной из классификаций язык разбивается на *базовый (entry)*, *промежуточный (intermediate)* и *полный (full)* уровни

Структура языка SQL (4)

- Эта классификация ориентирована, прежде всего, на производителей СУБД, в которых поддерживается SQL
- Реализация базового уровня языка является обязательным условием хотя бы какого-то соответствия стандарту
- Реализация промежуточного уровня желательна, и обычно именно такой уровень языка поддерживается ведущими компаниями-производителями SQL-ориентированных СУБД
- Наконец, полный уровень языка является целью, к достижению которой следует стремиться
- Критерием отнесения той или иной возможности языка к некоторому уровню является оцениваемая создателями стандарта SQL (большая часть которых является сотрудниками ведущих компаний, производящих SQL-ориентированные СУБД) техническая сложность реализации этой возможности
- Конечно, такая классификация важна и для программистов приложений баз данных, но только для того, чтобы оценить реальные возможности конкретной СУБД
- Для понимания языка SQL это разбиение на уровни несущественно

Структура языка SQL (5)

• Другая классификация показана на рисунке
• Среди всех конструкций языка SQL можно выделить такие конструкции, которые можно было использовать при *прямом (direct)* взаимодействии конечного пользователя с СУБД (например, в интерактивном режиме)



• В некотором смысле этот уровень также является базовым, поскольку соответствующие средства языка в наибольшей степени отражают его ориентированность на работу с мультимножествами
• На следующем уровне, уровне *встраиваемого (embedded)* SQL, язык расширяется конструкциями, позволяющими использовать возможности прямого SQL в программах, написанных на традиционных языках программирования
• Наконец, на уровне *динамического (dynamic)* SQL во встраиваемый SQL добавляются конструкции, позволяющие приложениям обращаться к СУБД с конструкциями прямого SQL, которые динамически образуются во время выполнения программы

Структура языка SQL (6)

- Вторая классификация является более полезной для читателя, постигающего основы языка SQL
- Дополнительные возможности, присутствующие во встраиваемом и в динамическом SQL, не слишком сильно влияют на модельное представление языка
- Конечно, возможности встраиваемого и динамического SQL необходимо хорошо знать разработчикам приложений SQL-ориентированных баз данных
- Но поскольку задачей этого курса не является обучение использованию языка SQL при программировании приложений баз данных, мы не будем затрагивать эти темы
- Обратимся к прямому SQL, причем не в полном объеме стандартов SQL:2003 и SQL:1999 (этого не позволяет сделать объем курса)
- Обсудим только наиболее важные аспекты

Типы данных SQL (1)

- Данные, хранящиеся в столбцах таблиц SQL-ориентированной базы данных, являются типизированными, т. е. представляют собой значения одного из типов данных, предопределенных в языке SQL или определяемых пользователями путем применения соответствующих средств языка
- Для этого при определении таблицы каждому ее столбцу назначается некоторый тип данных (или домен), и в дальнейшем СУБД должна следить, чтобы в каждом столбце каждой строки каждой таблицы присутствовали только допустимые значения. В этом разделе мы обсудим систему типов языка SQL

Типы данных SQL (2)

- Все допустимые в SQL типы данных, которые можно использовать при определении столбцов, разбиваются на следующие категории:
  - точные числовые типы (exact numerics);
  - приближенные числовые типы (approximate numerics);
  - типы символьных строк (character strings);
  - типы битовых строк (bit strings);
  - типы даты и времени (datetimes);
  - типы временных интервалов (intervals);
  - булевский тип (Booleans);
  - типы коллекций (collection types);
  - анонимные строчные типы (anonymous row types);
  - типы, определяемые пользователем (user-defined types);
  - ссылочные типы (reference types)

Типы данных SQL (3)

- В столбцах таблиц, определенных на любых типах данных, наряду со значениями этих типов, допускается сохранение неопределенного значения, которое обозначается ключевым словом NULL
- В языке определено, что результатом выражений вида *x* a_op NULL, NULL a_op *x*, NULL a_op NULL является NULL для всех арифметических операций a_op (+, - и т. д.), допустимых для типа данных выражения *x*
  - выражение NULL a_op NULL является допустимым для любой арифметической операции a_op
- Также по определению полагается, что значением выражений *x* comp_op NULL, NULL comp_op *x*, NULL comp_op NULL для всех операций сравнения (=, , >, < и т. д.), определенных для типа выражения *x*, является третье логическое значение unknown
  - выражение NULL comp_op NULL является допустимым для любой операции сравнения comp_op

Типы данных SQL (4)
Точные числовые типы (1)

- К категории *точных числовых типов* в SQL относятся те типы, значения которых точно представляют числа
- Типы данных этой категории распадаются на две части: *истинно целые типы* (INTEGER и SMALLINT) и *типы, допускающие наличие дробной части* (NUMERIC и DECIMAL)
- Охарактеризуем эти типы данных более подробно

Типы данных SQL (5)
Точные числовые типы (2) Истинно целые типы

- Тип INTEGER служит для представления целых чисел. Точность чисел (число сохраняемых бит) определяется в реализации
  - при определении столбца данного типа достаточно указать просто INTEGER.
- Тип SMALLINT также служит для представления целых чисел
  - точность определяется в реализации, но она не должна быть больше точности типа INTEGER
  - При определении столбца указывается просто SMALLINT
- Литералы типов целых чисел представляются в виде строк символов, изображающих десятичные числа; в начале строки могут присутствовать символы «+» или «-» (если символ знака отсутствует, подразумевается «+»)
  - примеры литералов типов INTEGER и SMALLINT: 1826545, 876

Типы данных SQL (6)

Точные числовые типы (3)

Точные типы, допускающие наличие дробной части (1)

- Тип NUMERIC
- На самом деле, это не просто тип данных, а параметризуемый тип
  - При определении столбца можно указать спецификацию NUMERIC (p, s), где p и s – литералы истинно целого типа, и p задает точность значений (число сохраняемых бит), а s – шкалу (число десятичных цифр в дробной части)
  - Задаваемая шкала не должна быть отрицательной и не должна превышать значение точности
  - При определении столбца можно использовать сокращенные формы спецификации типа – NUMERIC и NUMERIC (p)
    - Первая форма предполагает использование точности, определяемое по умолчанию в реализации, и шкалы, равной нулю, а вторая – использование заданной точности и шкалы, равной нулю
    - Допустимые диапазоны значений p и s определяются в реализации

Типы данных SQL (7)

Точные числовые типы (4)

Точные типы, допускающие наличие дробной части (2)

- Тип DECIMAL
- Этот тип аналогичен типу NUMERIC
- Отличие состоит в том, что если при определении столбца типа DECIMAL задается точность p, то на самом деле используется точность m, определяемая в реализации, такая, что m > p
  - Шкала всегда устанавливается такой, как явно или неявно (по умолчанию) задается
  - При указании типа столбца можно использовать спецификации DECIMAL, DECIMAL (p) и DECIMAL (p, s)
- Литералы типов точных чисел, допускающих наличие дробной части, представляются в виде строк символов, изображающих десятичные числа, в начале которых могут присутствовать символы «+» или «-» (если символ знака отсутствует, подразумевается «+»), а внутри последовательности цифр может присутствовать символ «.».
- Примеры литералов типов NUMERIC и DECIMAL: 125, 26.36

Типы данных SQL (8)

Приближенные числовые типы (1)

- К категории *приближенных числовых типов* в SQL относятся те типы, значения которых представляют числа приближенным образом
- Приближенные числа представляются в виде пары <мантисса, порядок>, где мантисса состоит из значащих цифр числа, а порядок определяет реальный размер числа
- В реализациях приближенным числовым типам SQL обычно соответствуют типы с плавающей точкой
- В SQL поддерживаются три варианта приближенных числовых типов

Типы данных SQL (9)
Приближенные числовые типы (2)

- Тип REAL
  - Значения типа соответствуют числам с плавающей точкой одинарной точности
  - Точность определяется в реализации, но обычно совпадает с точностью одинарной плавающей арифметики, поддерживаемой на аппаратной платформе, которая используется реализацией
  - При определении столбца указывается просто REAL
- Тип DOUBLE PRECISION
  - Точность значений этого типа определяется в реализации, но она должна быть больше точности типа REAL
  - Обычно приближенным числам SQL с двойной точностью соответствуют поддерживаемые аппаратурой числа с плавающей точкой двойной точности
  - При определении столбца указывается просто DOUBLE PRECISION

Типы данных SQL (10)

Приближенные числовые типы (3)

- Тип FLOAT
- Это параметризуемый тип, значение
  - Требуется, чтобы реально обеспечиваемая реализацией точность значений была не меньше p
  - Допустимый диапазон значений параметра p определяется в реализации
  - При определении столбца можно указать либо FLOAT (p), либо просто FLOAT
    - В последнем случае подразумевается точность, определяемая реализацией по умолчанию
- Литералы приближенных числовых типов представляются в виде литерала точного числового типа, за которым могут следовать символ «E» и литерал целого числового типа
  - Примеры литералов приближенных числовых типов: 123, 123.12, 123E12, 123.12E12
  - Литеральное выражение xEy представляет значение x*(10y)

Типы данных SQL (11)

Типы символьных строк (1)

- В SQL определены три параметризуемых *типа символьных строк*: CHARACTER (или CHAR), CHARACTER VARYING (или CHAR VARYING, или VARCHAR) и CHARACTER LARGE OBJECT (или CLOB)
- Тип CHARACTER
  - Значениями типа являются символьные строки
  - Конкретный набор допустимых символов определяется в реализации, но, как правило, включает набор символов ASCII
  - При определении столбца допускается использование спецификаций CHARACTER (x) и просто CHARACTER
    - Последний вариант эквивалентен заданию CHARACTER (1)
  - После определения столбца типа CHARACTER (x) СУБД будет резервировать место для хранения x символов этого столбца во всех строках соответствующей таблицы
  - Если, например, определен столбец типа CHARACTER (8), и в некоторой строке таблицы в него заносится символьная строка длиной пять символов, то реально будут храниться восемь символов, последние три из которых будут пробелами

Типы данных SQL (12)

Типы символьных строк (2)

- Тип CHARACTER VARYING
  - При определении столбца допускается использование спецификаций CHARACTER VARYING (x) и просто CHARACTER VARYING
    - Последний вариант эквивалентен заданию CHARACTER VARYING (1)
  - Если в некоторой таблице определяется столбец типа CHARACTER VARYING (x), то в каждой строке этой таблицы значения данного столбца будут занимать ровно столько места, сколько требуется для сохранения соответствующей символьной строки (но ни одна такая строка не может состоять более чем из x символов).

Типы данных SQL (13)

Типы символьных строк (3)

- Определен ряд операций, которые можно выполнять над символьными строками
- Перечислим некоторые из них
  - Операция конкатенации (обозначается в виде «||») возвращает символьную строку, произведенную путем соединения строк-операндов в том порядке, в каком они заданы.
  - Функция выделения подстроки (SUBSTRING) принимает три аргумента – строку, номер начальной позиции и длину – и возвращает строку, выделенную из строки-аргумента в соответствии со значениями двух последних параметров.
  - Функция UPPER возвращает строку, в которой все строчные буквы строки-аргумента заменяются прописными. Функция LOWER, наоборот, заменяет в заданной строке все прописные буквы строчными.
  - Функция определения длины (CHARACTER_LENGTH, OCTET_LENGTH, BIT_LENGTH) возвращает длину заданной символьной строки в символах, октетах или битах (в зависимости от вида вычисляющей функции) в виде целого числа.
  - Функция определения позиции (POSITION) определяет первую позицию в строке S, с которой в нее входит заданная строка S1 (если не входит, то возвращается значение нуль)

Типы данных SQL (14)
Типы символьных строк (4)

- Тип CHARACTER LARGE OBJECT
  - Этот тип данных предназначен для определения столбцов, хранящих большие и разные по размеру группы символов
  - При определении столбца задается спецификация CLOB (z), где z задает максимальный размер соответствующей группы символов
  - Максимально возможное значение параметра z определяется в реализации, но, очевидно, что оно должно быть существенно больше максимально возможного значения параметра x, присутствующего в типах CHAR и CHAR VARYING
- Литералы типов символьных строк представляются в виде последовательностей символов, заключенных в одинарные или двойные кавычки
  - В первом случае среди набора символов литерала допускается наличие символов двойной кавычки, а во втором – символов одинарной кавычки
  - Примеры литералов символьных строк: 'ABCDEF', 'Ab"Ctd', "Fbcdef", "ab'cdtF"

Типы данных SQL (16)
Типы битовых строк (1)

- В SQL определены три параметризуемых *типа битовых строк*: BIT, BIT VARYING и BINARY LARGE OBJECT (или BLOB).
- Тип BIT
  - значениями типа являются битовые строки
  - при определении столбца допускается использование спецификаций BIT (x) и просто BIT
    - последний вариант эквивалентен заданию BIT (1)
  - после определения столбца типа BIT (x) СУБД будет резервировать место для хранения x бит этого столбца во всех строках соответствующей таблицы

Типы данных SQL (17)
Типы битовых строк (2)

- Тип BIT VARYING
- при определении столбца допускается использование только спецификации без умолчания вида BIT VARYING (x), где значение x определяет максимальную длину битовой строки, которую можно хранить в данном столбце
- Над битовыми строками определен ряд операций:
  - битовая конкатенация (обозначается в виде ||) возвращает результирующую битовую строку, полученную путем конкатенации строк-аргументов в том порядке, в котором они заданы

Типы данных SQL (18)
Типы битовых строк (3)

- функция извлечения подстроки из битовой строки
  - синтаксис и семантика этой функции идентичны синтаксису и семантике функции SUBSTRING для символьных строк, за исключением того, что первый аргумент и возвращаемое значение являются битовыми строками.
- функция определения длины (OCTET_LENGTH, BIT_LENGTH) возвращает длину заданной битовой строки в октетах или битах в зависимости от выбранной функции.
- функция определения позиции (POSITION) определяет первую позицию в битовой строке S, с которой в нее входит строка S1
  - если строка S1 не входит в строку S, возвращается значение нуль

Типы данных SQL (19)
Типы битовых строк (4)

- Тип BINARY LARGE OBJECT
- этот тип данных предназначен для определения столбцов, хранящих большие и разные по размеру группы байтов
- при определении столбца задается спецификация BLOB (z), где z задает максимальный размер соответствующей группы байтов
- с технической точки зрения типы CLOB и BLOB очень похожим
  - их разделение требуется для того, чтобы подчеркнуть, что значения типа CLOB состоят из символов
    - в частности, в них может осмысленно производиться текстовый поиск
  - а значения типа BLOB состоят из произвольных байтов, не обязательно кодирующих символы

Типы данных SQL (20)
Типы битовых строк (5)

- Литералы типов битовых строк представляются как заключенные в одинарные кавычки последовательности символов «0» и «1», предваряемые символом «B»; или предваряемые символом «X» последовательности символов, которые изображают шестнадцатеричные цифры
  - за цифрой «9» следуют «A», «B», «C», «D», «E» и «F»
- Примеры литералов типов битовых строк: B'011100111100011111111', X'78FBCD0012FFFFA'

Типы данных SQL (21)

Типы даты и времени (1)

- Возможность сохранения в базе данных информации о дате и времени очень важна с практической точки зрения
  - Достаточно вспомнить взбудоражившую весь мир «проблему 2000 года», одним из основных источников которой было некорректное хранение дат в базах данных
- В стандарте SQL поддержке средств работы с датой и временем уделяется большое внимание
- В частности, поддерживаются специальные *«темпоральные» типы данных*
  - DATE,
  - TIME,
  - TIMESTAMP,
  - TIME WITH TIME ZONE и
  - TIMESTAMP WITH TIME ZONE

Типы данных SQL (23)

Типы даты и времени (2)

- Тип DATE
  - значения этого типа состоят из компонентов-значений года, месяца и дня некоторой даты
  - значение года состоит из четырех десятичных цифр и соответствует летоисчислению от Рождества Христова до 9999 г.
  - значение месяца состоит из двух десятичных цифр и варьируется от 01 до 12
  - значение номера дня месяца состоит из двух десятичных цифр и варьируется от 01 до 31, хотя значение месяца даты может накладывать ограничения на возможность использования значений дня месяца 29, 30 и 31

Типы данных SQL (24)

Типы даты и времени (3)

- В стандарте SQL не накладываются какие-либо ограничения на внутренний способ представления дат, используемый в реализации
- При определении столбца типа DATE указывается просто DATE.

- Литералы типа DATE представляются в виде строки «'yyyy-mm-dd'», где символы y, m и d должны изображать десятичные числа
  - например, литерал DATE '1949-04-08' представляет дату 8 апреля 1949 г.

Типы данных SQL (25)

Типы даты и времени (4)

- Тип TIME
  - значения этого параметризованного типа состоят из компонентов-значений часа, минуты и секунды некоторого времени суток
  - значение часа состоит ровно из двух десятичных цифр и варьируется от 00 до 23
  - значение минуты состоит из двух десятичных цифр и варьируется от 00 до 59
  - основное значение секунды также состоит из двух цифр, но может включать дополнительные цифры, представляющие доли секунды
    - так что в целом значение секунды варьируется от 00 до 61.999…

Типы данных SQL (26)

Типы даты и времени (5)

- в значении времени присутствуют две лишние секунды, поскольку Всемирная служба времени иногда добавляет две секунды к последней минуте года для синхронизации мирового времени с реальным
  - решение о поддержке этих «високосных» секунд принимается на уровне реализации
- число цифр в доле секунды также определяется в реализации
  - в стандарте требуется только то, чтобы это число было не меньше шести
- при определении столбца типа TIME может указываться TIME (p)
  - значение p задает точность долей секунды
- или просто TIME
  - в этом случае доли секунды не учитываются

Типы данных SQL (27)

Типы даты и времени (6)

- Литералы типа TIME представляются в виде строки TIME ‘hh:mm-ss:f…f’, где символы h, m, s и f должны изображать десятичные числа
  - например, литерал TIME ’16:33-20:333’ представляет время суток 16 часов 33 минуты 20 и 333 тысячных секунды

Типы данных SQL (28)

Типы даты и времени (7)

- Тип TIMESTAMP
  - значения этого параметризованного типа состоят из компонентов
    - значений года, месяца и дня некоторой даты
  - а также компонентов
    - значений часа, минуты и секунды некоторого времени суток
  - т. е. каждое значение задает некоторую абсолютную временную метку – отсюда название типа TIMESTAMP).
- Число десятичных цифр в значениях-компонентах и ограничения этих значений такие же, как у значений типов DATE и TIME

Типы данных SQL (29)

Типы даты и времени (8)

- при определении столбца типа TIMESTAMP может указываться TIMESTAMP (p)
  - значение p задает точность долей секунды
- или просто TIMESTAMP
  - в этом случае, в отличие от типа данных TIME, по умолчанию принимается, что в доли секунды используются шесть десятичных цифр
- максимально допустимое значение p определяется в реализации

- Литералы типа TIMESTAMP представляются в виде строки TIMESTAMP 'yyyy-mm-dd hh:mm-ss:f...f', где символы y, m, d, h, m, s и f должны изображать десятичные числа
  - Например, литерал TIMESTAMP '1949-04-08 16:33-20:333' представляет временную метку 16 часов 33 минуты 20 и 333 тысячных секунды 8 апреля 1949 г.

Типы данных SQL (30)

Типы даты и времени (9)

- Тип TIME WITH TIME ZONE
  - этот тип данных похож на тип TIME с тем лишь отличием, что значения типа TIME WITH TIME ZONE включают дополнительный компонент — значение, характеризующее смещение соответствующего времени относительно гринвичского времени
    - теперь его называют UTC – universal time coordinated
    - деталей представления этого дополнительного компонента мы касаться не будем.
- Тип TIMESTAMP WITH TIME ZONE
  - этот тип данных отличается от типа TIMESTAMP тем, что значения типа TIMESTAMP WITH TIME ZONE включают дополнительный компонент-значение, характеризующее смещение соответствующего времени относительно гринвичского

Типы данных SQL (31)

Типы временных интервалов (1)

- *Временным интервалом* называется разность между двумя значениями даты или времени
- В SQL определены две категории *типов временных интервалов*: «год-месяц» и «день-время суток»
- Временные интервалы языка SQL не привязываются к начальному и/или конечному значению даты/времени, а описывают только протяженность во времени
- В общем случае при определении столбца типа временного интервала указывается INTERVAL start (p) [TO end (q)], где в качестве «start» и «end» могут задаваться YEAR, MONTH, DAY, HOUR, MINUTE и SECOND
  - параметр p задает требуемую точность лидирующего поля интервала (число десятичных цифр)
  - параметр q может задаваться только в том случае, когда в качестве end используется SECOND, и указывает точность долей секунды

Типы данных SQL (32)

Типы временных интервалов (2)

- Возможны следующие вариации типов временных интервалов.
- Типы категории «год-месяц»
  - можно определить столбцы следующих типов:
  - INTERVAL YEAR, INTERVAL YEAR (p)
    - значения этих типов – временные интервалы в годах
  - INTERVAL MONTH, INTERVAL MONTH (p)
    - значения этих типов – временные интервалы в месяцах
  - INTERVAL YEAR TO MONTH, INTERVAL YEAR (p) TO MONTH
    - значения этих типов – временные интервалы в годах и месяцах.
  - если значение параметра p не указывается явно, по умолчанию принимается его значение «2»

Типы данных SQL (33)

Типы временных интервалов (3)

- Типы категории «день-время суток»
  - при определении столбца можно использовать следующие комбинации:
    - INTERVAL DAY (p), INTERVAL DAY,
    - INTERVAL DAY (p) TO HOUR, INTERVAL DAY TO HOUR,
    - INTERVAL DAY (p) TO MINUTE, INTERVAL DAY TO MINUTE,
    - INTERVAL DAY (p) TO SECOND (q), INTERVAL DAY TO SECOND (q),
    - INTERVAL DAY (p) TO SECOND, INTERVAL DAY TO SECOND,
    - INTERVAL HOUR (p), INTERVAL HOUR,
    - INTERVAL HOUR (p) TO MINUTE, INTERVAL HOUR TO MINUTE,
    - INTERVAL HOUR (p) TO SECOND (q), INTERVAL HOUR TO SECOND (q),
    - INTERVAL HOUR TO SECOND, INTERVAL MINUTE (p),
    - INTERVAL MINUTE, INTERVAL MINUTE (p) TO SECOND (q),
    - INTERVAL MINUTE TO SECOND (q), INTERVAL MINUTE (p) TO SECOND,
    - INTERVAL MINUTE TO SECOND, INTERVAL SECOND (p, q), INTERVAL
    - SECOND (p), INTERVAL SECOND
  - если значение параметра p не указывается явно, по умолчанию принимается его значение «2»
  - значением параметра q по умолчанию является «6»

Типы данных SQL (34)

Типы временных интервалов (4)

- Приведем только один пример литерала одной из разновидностей типа INTERVAL:
  - INTERVAL '10:20' MINUTE TO SECOND – временной интервал в 10 минут и 20 секунд.
- Над значениями темпоральных типов могут выполняться арифметические операции, смысл которых определяется следующей таблицей:

Тип первого операнда	Операция	Тип второго операнда	Тип результата
Datetime	-	Datetime	Interval
Datetime	+ или -	Interval	Datetime
Interval	+	Datetime	Datetime
Interval	+ или -	Interval	Interval
Interval	* или /	Numeric	Interval
Numeric	*	Interval	Interval

Типы данных SQL (35)

Булевский тип (1)

- При определении столбца булевского типа указывается просто спецификация BOOLEAN
  - *булевский тип* состоит из трех значений: true, false и unknown
  - соответствующие литералы обозначаются TRUE, FALSE и UNKNOWN
- Поддерживается возможность построения булевских выражений, которые вычисляются в трехзначной логике

Типы данных SQL (36)

Булевский тип (2)

- Таблицы истинности основных логических операций показаны на рисунке

AND	TRUE	FALSE	UNKNOWN
TRUE	TRUE	FALSE	UNKNOWN
FALSE	FALSE	FALSE	FALSE
UNKNOWN	UNKNOWN	FALSE	UNKNOWN

OR	TRUE	FALSE	UNKNOWN
TRUE	TRUE	TRUE	TRUE
FALSE	TRUE	FALSE	UNKNOWN
UNKNOWN	TRUE	UNKNOWN	UNKNOWN

NOT	
TRUE	FALSE
FALSE	TRUE
UNKNOWN	UNKNOWN

Типы данных SQL (37)

Типы коллекций (1)

- Начиная с SQL:1999, в языке поддерживается возможность использования типов данных, значения которых являются коллекциями значений некоторых других типов
- Обычно под термином *коллекция* понимается одно из следующих образований: массив, список, множество и мультимножество
- В варианте SQL:1999, принятом в 1999 г., были специфицированы только типы массивов
- В стандарте SQL:2003 появилась спецификация типа мультимножества

Типы данных SQL (38)

Типы коллекций (2)

- Любой возможный *тип массива* получается путем применения конструктора типов ARRAY
- При определении столбца, значения которого должны принадлежать некоторому типу массива, используется конструкция dt ARRAY [mc], где
  - dt специфицирует некоторый допустимый в SQL тип данных, а
  - mc является литералом некоторого точного числового типа с нулевой длиной шкалы и определяет максимальное число элементов в значении типа массива
    - в терминологии SQL:1999 это значение называется максимальной кардинальностью массива
- В стандарте SQL:1999 многомерные массивы и массивы массивов не поддерживались
- Однако в стандарте SQL:2003 это ограничение было снято, и теперь типом элементов любого типа коллекций может быть любой допустимый в SQL тип данных, кроме самого конструируемого типа коллекции

Типы данных SQL (39)

Типы коллекций (3)

- Элементам каждого значения типа массива соответствуют их порядковые номера, называемые индексами
- Значение индекса всегда должно принадлежать отрезку [1, mc]
- Значениями типа массива dt ARRAY [mc] являются все массивы, состоящие из элементов типа dt, максимальное значение индекса которых cs не превосходит значения mc
- При сохранении в базе данных значения типа массива занимает столько памяти, сколько требуется для сохранения cs элементов
- Обеспечивается доступ к элементам массива по их индексам
  - в частности, можно объявить столбец типа INTEGER ARRAY [10] и при вставке строки в соответствующую таблицу задать значение только пятого элемента массива
  - тогда в строку будет занесен массив из пяти элементов, причем первые четыре элемента будут содержать неопределенное значение (NULL)

Типы данных SQL (40)

Типы коллекций (4)

- Основными операциями над массивами являются:
  - выборка значения элемента массива по его индексу,
  - изменение некоторого элемента массива или массива целиком и
  - конкатенация (сцепление) двух массивов
  - кроме того, для любого значения типа массива можно узнать значение его cs

Типы данных SQL (41)

Типы коллекций (5)

- При определении столбца таблицы *типа мультимножеств* используется конструкция dt MULTISET, где
  - dt задает тип данных элементов конструируемого типа мультимножеств
- Значениями типа мультимножеств являются мультимножества, т. е. неупорядоченные коллекции элементов одного и того же типа, среди которых допускаются дубликаты
  - например, значениями типа INTEGER MULTISET являются мультимножества, элементами которых — целые числа. Примером такого значения может быть мультимножество {12, 34, 12, 45, -64}

Типы данных SQL (42)

Типы коллекций (6)

- В отличие от массива, мультимножество является неограниченной коллекцией
  - при конструировании типа мультимножеств не указывается предельная кардинальность значений этого типа
- Однако это не означает, что возможность вставки элементов в мультимножество действительно не ограничена
  - стандарт всего лишь не требует явного объявления границы
- Ситуация аналогична той, которая возникает при работе с таблицами, для которых в SQL не объявляется максимально допустимое число строк

Типы данных SQL (43)

Типы коллекций (7)

- Для типов мультимножеств поддерживаются операции:
  - преобразования типа значения-мультимножества к типу массивов или другому типу мультимножеств с совместимым типом элементов (операция CAST),
  - удаления дубликатов из мультимножества (функция SET),
  - определения числа элементов в заданном мультимножестве (функция CARDINALITY),
  - выборки элемента мультимножества, содержащего в точности один элемент (функция ELEMENT)
- Кроме того, для мультимножеств обеспечиваются операции
  - объединения (MULTISET UNION),
  - пересечения (MULTISET INTERSECT) и
  - определения разности (MULTISET EXCEPT)
- Каждая из операций может выполняться в режиме с сохранением дубликатов (режим ALL) или с устранением дубликатов (режим DISTINCT)

Типы данных SQL (44)

Типы коллекций (8)

- Расширенные в SQL:2003 возможности работы с типами коллекций являются принципиально важными
- Даже при наличии определяемых пользователями типов данных и типов массивов SQL:1999 не предоставлял полных возможностей для преодоления исторически присущего реляционной модели данных вообще и SQL в частности ограничения «плоских таблиц»
- После появления конструктора типов мультимножеств и устранения ограничений на тип данных элементов коллекции это историческое ограничение полностью ликвидировано
- Мультимножество, типом элементов которого является анонимный строчный тип, представляет собой полный аналог таблицы
- Тем самым, в базе данных допускается произвольная вложенность таблиц
  - возможности выбора структуры базы данных безгранично расширяются

Типы данных SQL (45)

Анонимные строчные типы (1)

- *Анонимный строчный тип* – это конструктор типов ROW, позволяющий производить безымянные типы строк (кортежей)
- Любой возможный строчный тип получается путем использования конструктора ROW
- При определении столбца, значения которого должны принадлежать некоторому строчному типу, используется конструкция ROW (fld1, fld2, s, fldn),
  - где каждый элемент fldi, определяющий поле строчного типа, задается в виде тройки fldname, fldtype, fldoptions
  - подэлемент fldname задает имя соответствующего поля строчного типа
  - подэлемент fldtype специфицирует тип данных этого поля.

Типы данных SQL (46)

Анонимные строчные типы (2)

- В качестве типа данных поля строчного типа можно использовать любой допустимый в SQL тип данных, включая типы коллекций, определяемые пользователями типы и другие строчные типы
- Необязательный подэлемент fldoptions может задаваться для указания применяемого по умолчанию порядка сортировки, если соответствующий подэлемент fldtype указывает на тип символьных строк, а также должен задаваться, если fldtype указывает на ссылочный тип
- Степенью строчного типа называется число его полей

Типы данных SQL (47)

Типы, определяемые пользователем (1)

- *Индивидуальные типы (Distinct Types)*
- Можно определить долговременно хранимый, именованный тип данных, опираясь на единственный предопределенный тип
  - например, можно определить индивидуальный тип данных PRICE, опираясь на тип DECIMAL (5, 2)
  - тогда значения типа PRICE представляются точно так же, как значения типа DECIMAL (5, 2)
- Однако в SQL:1999 индивидуальный тип не наследует от своего опорного типа набор операций над значениями

Типы данных SQL (48)
Типы, определяемые пользователем (2)

- Например, чтобы сложить два значения типа PRICE требуется явно сообщить системе, что с этими значениями нужно обращаться как со значениями типа DECIMAL (5, 2)
- Другая возможность состоит в явном определении методов, функций и процедур, связанных с данным индивидуальным типом
  - нет наследования

- Похоже, что в будущих версиях стандарта появятся и другие, более удобные возможности

Типы данных SQL (49)
Типы, определяемые пользователем (3)

- *Структурные типы (Structured Types)*
- Соответствующие возможности SQL:1999 позволяют определять долговременно хранимые, именованные типы данных, включающие один или более атрибутов любого из допустимых в SQL типа данных, в том числе другие
  - структурные типы,
  - типы коллекций,
  - строчные типы и т. д.

Типы данных SQL (50)
Типы, определяемые пользователем (4)

- Стандарт SQL не накладывает ограничений на сложность получаемой в результате структуры данных, однако не запрещает устанавливать такие ограничения в реализации
- Дополнительные механизмы определяемых пользователями методов, функций и процедур позволяют определить поведенческие аспекты структурного типа

Типы данных SQL (51)
Ссылочные типы (1)

- Обеспечивается механизм конструирования типов
  - *ссылочных типов*
- которые могут использоваться в качестве типов столбцов некоторого вида таблиц
  - типизированных таблиц
- Фактически значениями ссылочного типа являются строки соответствующей типизированной таблицы

Типы данных SQL (52)
Ссылочные типы (2)

- Более точно, каждой строке типизированной таблицы приписывается уникальное значение
  - нечто вроде первичного ключа, назначаемого системой или приложением,
- которое может использоваться в методах, определенных для табличного типа, для уникальной идентификации строк соответствующей таблицы
- Эти уникальные значения называются *ссылочными значениями*, а их тип – ссылочным типом
- Ссылочный тип может содержать только те значения, которые действительно ссылаются на экземпляры указанного типа
  - т. е. на строки соответствующей типизированной таблицы

Средства определения, изменения определения и отмены определения доменов (1)

- При определении столбцов таблицы требуется явно указывать тип данных каждого столбца
- Для этого можно использовать средства спецификации типа
- Но в SQL поддерживается и другой механизм— механизм доменов
- Домен является долговременно хранимым, именованным объектом схемы базы данных
- Домены можно:
  - создавать (определять),
  - изменять (изменять определения) и
  - ликвидировать (отменять определение)
- Имена доменов можно использовать при определении столбцов таблиц

Средства определения, изменения определения и отмены определения доменов (2)

- Можно считать, что в SQL *определение домена* представляет собой вынесенное за пределы определения индивидуальной таблицы «родовое» определение столбца, которое можно использовать для определения различных реальных столбцов реальных базовых таблиц
- В языке SQL обеспечиваются средства
  - определения доменов,
  - изменения и
  - отмены существующих определений

Средства определения, изменения определения и отмены определения доменов (3)

- Определение домена
- Для определения домена в SQL используется оператор CREATE DOMAIN
- domain_definition ::= CREATE DOMAIN domain_name [AS] data_type
  [default_definition]
  [domain_constraint_definition_list]
- Здесь domain_name задает имя создаваемого домена, data_type есть спецификация определяющего типа данных
- В необязательных разделах default_definition и domain_constraint_definition_list специфицируются значение домена по умолчанию и набор ограничений целостности, которые будут применяться к любому столбцу, определенному на этом домене

Средства определения, изменения определения и отмены определения доменов (4)

- Раздел default_definition имеет вид
  DEFAULT { literal | niladic_function | NULL }
- Здесь literal представляет любое допустимое литеральное значение определяющего типа домена,
  - NULL обозначает неопределенное значение, а
  - niladic_function может задаваться в одной из следующих форм:
    - USER
    - CURRENT_USER
    - SESSION_USER
    - SYSTEM_USER
    - CURRENT_DATE
    - CURRENT_TIME
    - CURRENT_TIMESTAMP

Средства определения, изменения определения и отмены определения доменов (5)

- Если в операторе CREATE DOMAIN значение по умолчанию не специфицируется, считается, что такого значения нет
- Однако позже к определению домена можно добавить раздел значения по умолчанию с помощью оператора ALTER DOMAIN
  - Кроме того, этот оператор позволяет удалить раздел значения по умолчанию из существующего определения домена.
- Элемент списка domain_constraint_definition_list имеет вид [CONSTRAINT constraint_name] CHECK (conditional_expression)
- Необязательный раздел CONSTRAINT constraint_name позволяет определить имя нового ограничения целостности
- Если явное указание имени отсутствует, ограничению назначается имя, автоматически генерируемое системой

Средства определения, изменения определения и отмены определения доменов (6)

- Что касается вида условного выражения, служащего собственно ограничением целостности, то в стандарте запрещается лишь прямое или косвенное использование в нем домена, в определение которого входит данное условное выражение.
- Однако наиболее естественным (и наиболее распространенным) видом ограничения домена является следующий:
  CHECK (VALUE IN (list_of_valid_values))
- Такое ограничение запрещает появление в любом столбце, определенном на данном домене, любого значения определяющего типа, не входящего в список допустимых значений.

Средства определения, изменения определения и отмены определения доменов (7)

- Примеры определений доменов
- В примерах мы будем иметь дело с таблицами служащих (EMP), отделов (DEPT) и проектов (PRO)
- Каждый служащий обладает уникальным номером (EMP_NO) и получает заработную плату (SALARY)
- Определим домены EMP_NO и SALARY
- CREATE DOMAIN EMP_NO AS INTEGER
  CHECK (VALUE BETWEEN 1 AND 10000);
- Номера служащих являются целыми числами, поэтому базовый тип домена EMP_NO есть тип INTEGER
- Кроме того, на значения этого домена устанавливается следующее ограничение: они должны быть больше нуля и не превосходить целое значение 10000

Средства определения, изменения определения и отмены определения доменов (8)

- Домен SALARY определим следующим образом:
- CREATE DOMAIN SALARY AS NUMERIC (10, 2)
  DEFAULT 10000.00
  CHECK (VALUE BETWEEN 10000.00 AND 20000000.00)
  CONSTRAINT SAL_NOT_NULL CHECK (VALUE IS NOT NULL);
- Размер заработной платы является значением точного числового типа NUMERIC из десяти десятичных цифр, две из которых составляют дробную часть
- По умолчанию размер заработной платы составляет 10000 руб.
- Установлен диапазон допустимого размера зарплаты от 10000 руб. до 20000000 руб.
- Неопределенное значение зарплаты не допускается
  - на уровне определения домена

Средства определения, изменения определения и отмены определения доменов (9)

- Изменение определения домена
- Для изменения характеристик ранее определенного домена используется оператор SQL ALTER DOMAIN
- Синтаксис этого оператора выглядит следующим образом:
- domain_alternation ::= ALTER DOMAIN domain_name domain_alternation_action domain_alternation_action ::= domain_default_alternation_action | domain_constraint_alternation_action
- Как видно из синтаксических правил, при изменении определения домена можно
  - выполнить действие по изменению раздела значения по умолчанию либо
  - изменить ограничение домена.

Средства определения, изменения определения и отмены определения доменов (10)

- Для первого варианта действует следующий синтаксис:
- domain_default_alternation_action ::=
  SET default_definition | DROP DEFAULT
- В случае установки нового значения по умолчанию (SET) это значение автоматически применяется ко всем столбцам, определенным на данном домене
  - более точно, это значение становится новым значением по умолчанию
  - операция не оказывает влияния на состояние существующих строк таблиц базы данных
- В случае отмены раздела значения по умолчанию в определении домена (DROP) существовавшее значение домена по умолчанию становится значением по умолчанию каждого столбца,
  - который определен на данном домене и
  - для которого не специфицировано собственное значение по умолчанию

Средства определения, изменения определения и отмены определения доменов (11)

- Действие по изменению ограничения домена определяется следующим синтаксисом:
- domain_constraint_alternation_action ::=
  ADD domain_constraint_definition |
  DROP CONSTRAINT constraint_name
- Действие по добавлению нового определения ограничения домена (ADD) приводит к тому, что новое условие добавляется через AND к существующему ограничению домена
- Если к моменту выполнения соответствующего оператора ALTER DOMAIN существуют столбцы некоторых таблиц, текущие значения которых противоречат новому ограничению, то СУБД должна отвергнуть этот оператор ALTER DOMAIN

Средства определения, изменения определения и отмены определения доменов (12)

- Действие по отмене ограничения домена (DROP) приводит к исчезновению соответствующей части общего ограничения соответствующего домена, что, естественно, не влияет на существующие значения столбцов имеющихся таблиц

Средства определения, изменения определения и отмены определения доменов (13)

- Примеры изменения определения домена
- Немного поупражняемся с доменом SALARY
- Для изменения значения заработной платы по умолчанию с 10000 на 11000 руб. нужно выполнить оператор
  ALTER DOMAIN SALARY SET DEFAULT 11000.00;
- Для отмены значения по умолчанию в домене SALARY следует воспользоваться оператором
  ALTER DOMAIN SALARY DROP DEFAULT;
- Если к определению домена SALARY требуется добавить ограничение
  - например, запретить значение зарплаты, равное 15000 руб.
- необходимо выполнить оператор
  ALTER DOMAIN SALARY ADD CHECK (VALUE <> 15000.00);

Средства определения, изменения определения и отмены определения доменов (14)

- Наконец, если требуется отменить (именованное!) ограничение целостности, препятствующее наличию неопределенных значений в столбцах, которые определены на домене SALARY, то нужно выполнить оператор
ALTER DOMAIN SALARY DROP CONSTRAINT SAL_NOT_NULL;

Средства определения, изменения определения и отмены определения доменов (15)

- Отмена определения домена
- Чтобы отменить ранее созданное определение домена, нужно воспользоваться оператором DROP DOMAIN в следующем синтаксисе:
  DROP DOMAIN domain_name {RESTRICT | CASCADES}
- Если в операторе указано RESTRICT, и если соответствующий домен использован в определении некоторого столбца, в определении некоторого представления или в определении ограничения целостности , то оператор DROP DOMAIN отвергается
- В противном случае определение домена ликвидируется.

Средства определения, изменения определения и отмены определения доменов (16)

- Если в операторе DROP DOMAIN указано CASCADES, то оператор выполняется всегда
- При этом уничтожаются все представления и ограничения целостности, в определении которых использовалось имя данного домена
- Столбцы, определенные на этом домене, автоматически переопределяются следующим образом:
  - считается, что каждый такой столбец теперь относится к определяющему типу уничтожаемого домена;
  - если у столбца не было определено собственное значение по умолчанию, то считается, что теперь у него имеется такое значение по умолчанию, совпадающее со значением по умолчанию уничтожаемого домена;
  - каждый столбец наследует все ограничения уничтожаемого домена

Неявные и явные преобразования типа или домена (1)

- В SQL поддерживается совместимость некоторых типов данных за счет *неявного преобразования* значений одного типа к значениям другого типа данных
- например, при необходимости FLOAT неявно приводится к DOUBLE
- Опишем наиболее важные правила совместимости типов, принятые в SQL:1999
- Начнем с определения приводимости типов
- Тип данных A приводим к типу данных B в том и только в том случае, когда в любом месте, где ожидается значение типа B, может быть использовано значение типа A

Неявные и явные преобразования типа или домена (2)

- Основные правила приводимости типов состоят в следующем.
- Типы символьных строк
  - Тип CHARACTER (x) приводим к любому типу CHARACTER (y), если y ≥ x. Типы VARCHAR (x) и CHARACTER (x) приводимы к любому типу VARCHAR (y), если yx. Типы CHARACTER (x) и VARCHAR (x) приводимы к любому типу CLOB.
  - Типы битовых строк. Тип BIT (x) приводим к любому типу BIT (y), если y ≥ x. Типы BIT VARYING (x) и BIT (x) приводимы к любому типу BIT VARYING (y), если y ≥ x.
  - Типы BLOB. Тип BLOB (x) приводим к любому типу BLOB (y), если y ≥ x.
  - Типы точных чисел. Тип EN (p_1, s_1) приводим к любому типу EN (p_2, s_2), у которого $s_2 \geq s_1$ и p_2 определяется в реализации. Тип EN (p, s) приводим к любому типу приблизительных чисел AN (p_1), где p_1 определяется в реализации.
  - Типы приблизительных чисел. Тип AN (p_1) приводим к любому типу AN (p_2), если $p_2 \geq p_1$

Неявные и явные преобразования типа или домена (3)

- Неявные преобразования типов не всегда удобны, недостаточно гибки и иногда могут вызывать ошибки
- Поэтому число допустимых неявных преобразований типов в SQL весьма ограничено
- Однако в SQL существует специальный оператор CAST, с помощью которого можно *явно* преобразовывать типы или домены в более широких пределах допускаемых преобразований
- Конструкция имеет следующий синтаксис:
  CAST ({scalar-expression | NULL } AS {data_type | domain_name})

Неявные и явные преобразования типа или домена (4)

- Оператор преобразует значение заданного скалярного выражения к указанному типу или к базовому типу указанного домена
- Результатом применения оператора CAST к неопределенному значению является неопределенное значение
- Для значений, отличных от неопределенных, в стандарте приводятся подробные правила выполнения преобразований, которые интуитивно понятны

Неявные и явные преобразования типа или домена (5)

- Поясним действие оператора CAST в наиболее важных случаях; обозначим:
  - EN – точные числовые типы (Exact Numeric)
  - AN – приблизительные числовые типы (Approximate Numeric)
  - C – типы символьных строк (Character)
  - FC – типы символьных строк постоянной длины (Fixed-length Character)
  - VC – типы символьных строк переменной длины (Variable-length Character)
  - B – типы битовых строк (Bit String)
  - FB – типы битовых строк постоянной длины (Fixed-length Bit String)
  - VB – типы битовых строк переменной длины (Variable-length Bit String)
  - D – тип Date
  - T – типы Time
  - TS – типы Timestamp
  - YM – типы Interval Year-Month
  - DT – типы Interval Day-Time

Неявные и явные преобразования типа или домена (6)

- Пусть TD – это тип данных, к которому производится преобразование, а SD – тип данных операнда
- Тогда допустимы следующие комбинации
  - «да» означает безусловную допустимость, «нет» – безусловную недопустимость и «?» – допустимость с оговорками