ОБЪЕКТНО-ОРИЕНТИРОВАННЫЕ СВОЙСТВА SQL3

ШВЕЦ Иван
Coordonator: САРАНЧУК Дориан
Технический Университет Молдовы

Аннотация: *В данной статье описаны отличительные черти стандарта SQL3. Рассмотрены новые типы данных, их особенности и назначение. Описаны возможности языка, которые введены для корректной работы с объектными типами данных. Приведен пример использования синтаксиса для трёх таблиц с различными данными, включая строчные типы, BLOB и другие.*

Ключевые слова: *SQL3, тип данных, объект, функции, процедуры.*

1. ВВЕДЕНИЕ

Чисто реляционные базы данных обладают рядом ограничений, которые затрудняют их использование в приложениях, требующих богатого типового окружения. Это относится и к категорическому требованию использовать в столбцах таблиц только атомарные значения встроенных типов, и к невозможности определить новые типы данных (возможно, с атомарными значениями) с дополнительными или переопределёнными операциями. Понятно, что ослабление этих ограничений приводит к потребности существенного пересмотра архитектуры серверных продуктов баз данных.

SQL3 был принят как стандарт в 1999 году после 7-ми лет разработки. По сути, SQL3 включает в себя инструменты определения и управления данными из объектно-ориентированных СУБД, сохраняя при этом реляционные свойства СУБД.

Части SQL3, предоставляющие основу для поддержки работы с объектно-ориентированными структурами, являются: пользовательские типы (abstract data type, named row types, and distinct types), конструкторы для типов row type и reference type, для типов массивов (set, list и multiset), пользовательские функции и процедуры, поддержка “больших объектов” (large objects – BLOB и CLOB).

2. СЛОЖНЫЕ ТИПЫ ДАННЫХ

Эта разновидность типов ближе всего к абстрактным типам данных в языках программирования. Идея состоит в том, что сначала специфицируется определяемый пользователем тип данных (переименовывается некоторый встроенный тип, определяется строчный тип или тип коллекции). Затем для этого типа можно специфицировать ряд определяемых пользователем функций. После полной спецификации объектного типа его можно использовать как встроенный или любой ранее определённый тип.

Определение пользовательского абстрактного типа данных ADT (*abstract data type*) включает в себя атрибуты и операции в одно целое. Подтип ADT наследует структуру и поведение типа-родителя (множественное наследование поддерживается).

В реляционном подходе отсутствует возможность отдельно определить именованную схему таблицы, а затем - одну или несколько таблиц с той же самой схемой.

Строчный тип (*row type*) – последовательность пар типа (имя поля: данные). Данный тип похож на определение таблицы. Строчный тип представляет собой тип данных, который может отражать типы строк/рядов в таблицах, таким образом, чтобы целые ряды можно было хранить в виде переменных, передавать как аргументы процедурам (*routines*) и получать в виде возвращаемых значений от вызовов функций.

Были определены и коллекции как типы данных (set, list и multiset). Столбцы таблиц могут содержать наборы, списки и мультимножества помимо индивидуальных значений.

Таблицы были дополнены понятием подтаблицы (*subtable*). Таблица может быть определена как подтаблица для одной или нескольких супертаблиц при помощи команды UNDER в определении таблицы и подтаблица наследует все столбцы(схему) от своих супертаблиц и может вносить свои новые атрибуты.

Типы BLOB (Binary Large Object) и CLOB (Character Large Object) были введены для поддержки работы с объектами очень больших размеров. Такие объекты хранятся прямо в базе данных, избегая использования внешних файлов.

“Большие” типы данных не могут быть частью некоторых операций, например, сравнений с операторами “больше” (>) или “меньше” (<), но могут работать в других командах, таких как выборка дынных и предикат LIKE. SQL3 как реляционный язык, использующий строгий поиск по критериям, не содержит понятия степеней актуальности и соответственно не поддерживает возможности упорядочивания таких данных по семантическим сходствам с запросом.

3. СПЕЦИФИКАЦИЯ ПОВЕДЕНЧЕСКОЙ СЕМАНТИКИ

Когда определяется абстрактноый тип данных ADT, функция-конструктор автоматически прописывается для создания новых экземпляров объекта. Она возвращает новый экземпляр объекта данного типа, атрибутам которого присвоены значения по умолчанию их соответствующих типов. Для каждого атрибута автоматически определяются функции обозревателя(observer) и мутатора (mutator), хотя они могут быть определены и пользователям.

Функции могут быть либо SQL функциями, полностью описанными в определении схемы в SQL, либо могут быть внешними вызовами функций, написанных на стандартных языках программирования.

Определение состоит из имени, параметров, региона RETURNS (если это функция) и тела. Параметр определяется именем, типом данных, а также одним из трёх категорий: IN, OUT или INOUT. Метод может быть определён в SQL или быть внешним методом, при этом работает механизм перегрузки.

4. АТРИБУТЫ

Существует два типа ADT атрибутов, *хранимые атрибуты* и *виртуальные атрибуты* (*stored attributes* и *virtual attributes*). *Хранимый* создаётся, дав атрибуту имя и тип данных. Тип данных хранимого атрибута может быть любым известным типом данных, в том числе и ADT. Каждый атрибут неявно задаёт пару функций, для получения (функция-наблюдатель) и установки (функция-мутатор) значения атрибута. *Виртуальный* атрибут имеет производное или вычисляемое с помощью пользовательской функции-наблюдателя значение.

5. ОБЪЕКТНЫЕ СВОЙСТВА

В SQL3 был добавлен набор команд для того, чтобы сделать его вычислительно-полным для обеспечения полного описания поведения объектов.

- Команда для присваивания, позволяющая присвоить результат SQL выражения свободной переменной, атрибуту или атрибуту ADT;
- Команда CALL для вызова SQL процедуры;
- Команда RETURN, позволяющая вернуть результат SQL запроса как значение поля RETURNS некой SQL функции;
- Команда CASE для выбора пути выполнения кода на основе альтернативного выбора;
- IF команда вместе с THEN, ELSE и ELSEIF для выбора пути выполнения кода на основе истинности одного или нескольких условий.
- Команды LOOP, WHILE и REPEAT для повторяющегося выполнения части SQL кода. WHILE проверяет <поисковое условие> перед выполнением блока кода, а REPEAT - после.

Дополнительные средства управления включают составные операторы и обработку исключений. Составной оператор – это утверждение, позволяющее набору операторов SQL, быть сгруппированы в «блок». Составной оператор может объявлять свои локальные переменные и описывать обработку исключений, выявленных во время выполнения любой команды в группе. Для обработки исключений, блок CONDITION устанавливает соответствие “1 к 1-му” между состоянием ошибки SQLSTATE и пользовательским именем исключения. Блок HANDLER ассоциирует определяемые пользователем обработчики исключений с определёнными исключениями.

Стандарт SQL92 определяет языковые привязки для ряда стандартных языков. Одним из ключевых аспектов отдельных языковых привязок является определение соответствий между типами данных SQL и типами данных языка-хозяина. В некоторых случаях, это относительно просто; например, SQL тип CHARACTER привязывается к типу *char* в языке C. В других случаях, привязка не так проста. Например, SQL92 содержит тип данных TIMESTAMP, но стандартные языки программирования не содержат соответствующего встроенного типа. В этих случаях, SQL требует использования функции CAST для преобразования данных TIMESTAMP базы данных в символьные

данные в программе, и наоборот. В SQL92 эти соответствия типов определены только на уровне элементарных скалярных типов данных. Нет соответствий типов, определённых как структурированные типы, например, между рядом таблицы SQL и плоской записи или структуры в языке программирования (хотя некоторые такие соответствия будет относительно просто определить).

В настоящее время в SQL3 нет привязки между расширениями ADT (или строк, содержащих их) и классами объектов или типов в объектно-ориентированных языках программирования, таких как C++ или Smalltalk, хотя работы в этом направлении ведутся.

Далее приведён пример запроса, источником которого будет служить часть БД, показанная на рис.1.

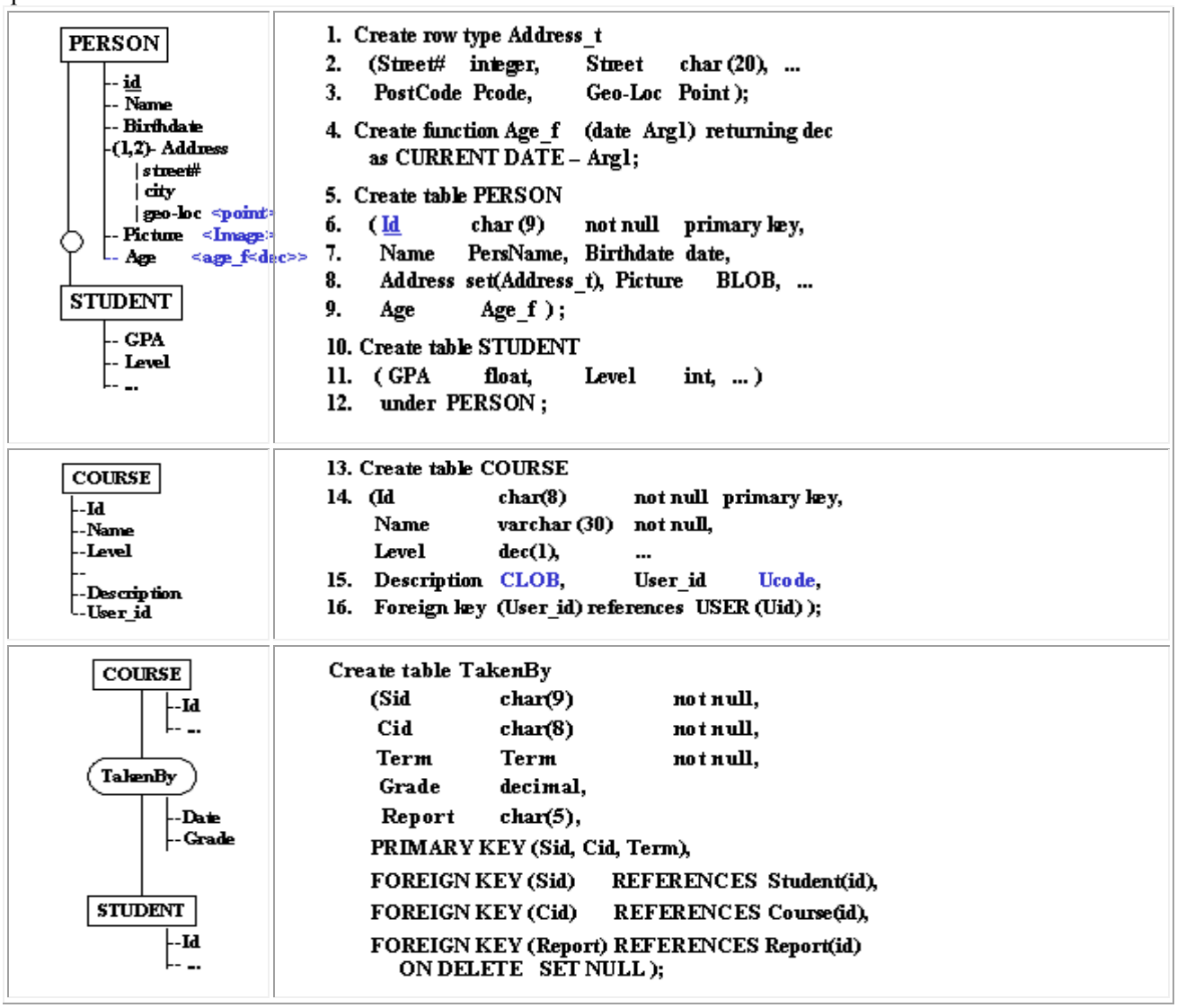

Рисунок 1 – Исходные таблицы для запроса

Запрос объединяет в себе основные нововведения стандарта SQL3 (виртуальный атрибут *Age*, BLOB *Picture*, CLOB *Description*, наследование между таблицами *STUDENT* и *PERSON*, строчный тип *Address_t* и массив *Address* типа *set* и др.). SELECT S.Name, Age, C.Name, C.Level, C.Description

```
FROM Student S, Course C, TakenBy T
WHERE S.Id = T.Sid AND T.Cid = C.Id AND C.Level > 1
AND C.Description LIKE '%data%'
AND C.Description LIKE '%management%'
AND S.Age > 25
AND (CURRENT DATE – T.Date) < 3 YEARS
ORDER BY C.LEvel, C.Name;
```

ЗАКЛЮЧЕНИЕ

Наиболее важным характеристиками SQL3 можно считать частичный отказ от чисто реляционной модели структурирования и хранения данных. С появлением стандарта SQL3 отпала строгая необходимость использования атомарных типов данных в качестве доменов атрибутов при создании таблиц. Такое изменения базовых требований позволило переосмыслить свойства баз данных и создавать БД с новыми предназначениями и задачами, ранее являвшимися недоступными.

За счёт создания БД со сложными типами данных в качестве доменов атрибутов возросла полезная нагрузка и количество задач, которые может решать та или иная база данных. Таким образом, появляется возможность создавать мультимедийные базы данных в тех же СУБД. Введение составных и “больших” типов данных позволяет строить новые йерархии отношений, в запросах к которым заметно снижается количество случаев, когда необходимо использовать соединения (JOIN) таблиц, что значительно сокращает вычислительную нагрузку и, соответственно, время выполнения запроса. С введением иерархий таблиц появляется возможность точнее описывать фактическое положение объектов в жизни.

Преимуществами объектно-реляционных БД является их функциональная многогранность, простота формирования запросов (основной принцип их создания не изменился и запросы больше похожи на широко распространённые языки программирования), их простая масштабируемость и достаточно простая привязка к функционалу других объектно-ориентированных языков программирования.

Обратной стороной данных нововведений становится резко возросший размер баз данных. Даже в тех случаях, когда крупные объекты хранятся в отдельных файлах (external files) фактический размер базы данных не сократится. Время выполнения сложных запросов к большим сложным объектам заметно возрастает. Объекты типа CLOB и BLOB могут сильно сократить производительность БД, так как содержащаяся в них информация структурирована непредсказуемо и такие объекты не поддаются простой сортировке и не позволяют индексировать таблицы. Такие объекты могут быть практически любого формат и размера, что требует отдельного описания их поведения в БД (поиск, сравнение и т.д.).

ЛИТЕРАТУРА

1. Nelson Mattos, "An Overview of the SQL3 Standard", presentation foils, Database Technology Institute, IBM Santa Teresa Lab., San Jose, CA, July 1996, ftp://speckle.ncsl.nist.gov/isowg3/dbl/BASEdocs/descriptions/SQL3_foils.ps.

2. Krishna G. Kulkarni, "Object-Orientation and the SQL Standard", *Computer Standards & Interfaces 15* (1993), 287-300.

3. Krishna Kultarni, Mike Carey, Linda DeMichiel, Nelson Mattos, Wei Hong, and Mike Ubell, "Introducing Reference Types and Cleaning Up SQL3's Object Model", SQL3 Change Proposal X3H2-95-456, November26, 1995.

4. Jim Melton and Alan R. Simon, *Understanding the New SQL: A Complete Guide*, Morgan Kaufmann, San Francisco, 1993.

5. www.objs.com : электронный портал[Электронный ресурс]. — Object Services and Consulting, Inc - Режим доступа : http://www.objs.com/x3h7/sql3.htm . (20.5.2014).