Теория баз данных

Лекция 2. Реляционная модель

Е. П. Моргунов
Сибирский федеральный университет
г. Красноярск
Институт космических и информационных технологий
emorgunov@mail.ru

Версия от 15.02.2018

2.1. История реляционной модели

Реляционная модель была впервые предложена Эдгаром Ф. Коддом (E. F. Codd) в статье «A relational model of data for large shared data banks» в 1970 г.

Целью было:

1. Добиться высокой степени независимости приложений от данных (т. е. от изменений внутреннего представления данных: организации файлов, упорядочивания записей и изменения путей доступа).
2. Добиться согласованности данных и решить проблему избыточности данных (было введено понятие нормализованных отношений, о котором речь пойдет в следующих лекциях).

Исследовательские проекты:

- System R в компании IBM, вторая половина 1970-х гг.
  - Разработка языка SQL и первых коммерческих реляционных СУБД
- Проект INGRES в Калифорнийском университете в Беркли (University of California at Berkeley).
  INGRES – INteractive Graphics REtrieval System
  Майкл Стоунбрейкер (Michael Stonebraker)

2.2. Основные положения реляционной модели

В реляционной системе выполняются как минимум три условия:

- **Структурный аспект.** Данные в базе воспринимаются пользователем, как таблицы (и никак иначе).
- **Аспект целостности.** Эти таблицы отвечают определенным условиям целостности. (См. лекцию 1)
- **Аспект обработки.** В распоряжении пользователя имеются операторы манипулирования таблицами, которые генерируют новые таблицы на основании уже имеющихся и среди которых есть, по крайней мере, операторы сокращения (restrict), проекции (project) и объединения (join).

- Операция *сокращения* извлекает указанные строки из таблицы (в названии этой операции подразумевается, что число строк ее результата меньше или равно числу строк исходной таблицы). Операцию сокращения иногда называют *выборкой.*
- Операция *проекции* предназначена для извлечения определенных столбцов из таблицы.
- Операция *соединения* предназначена для получения комбинации двух таблиц на основе общих значений в общих столбцах.
- Эти определения являются нестрогими!

2.2.1. Модельная база данных

Таблица «Студенты» (students)

Номер зачетной книжки (record_book)	Ф. И. О. (name)	Серия паспорта (psp_ser)	Номер паспорта (psp_num)
55500	Иванов Иван Петрович	0402	645327
55800	Климов Андрей Иванович	0402	673211
55865	Новиков Николай Юрьевич	0202	554390

Таблица «Успеваемость» (progress)

Номер зачетной книжки (record_book)	Предмет (discipline)	Учебный год (acad_year)	Семестр (term)	Оценка (mark)
55500	Физика	2017/2018	1	5
55500	Математика	2017/2018	1	4
55800	Физика	2017/2018	1	4
55800	Физика	2017/2018	2	5

2.2.2. Операторы манипулирования таблицами

Сокращение Таблица «Студенты» (students)

Номер зачетной книжки	Ф. И. О.	Серия паспорта	Номер паспорта
55500	Иванов Иван Петрович	0402	645327

Проекция Таблица «Успеваемость» (progress)

Предмет	Учебный год	Семестр	Оценка
Физика	2017/2018	1	5
Математика	2017/2018	1	4
Физика	2017/2018	1	4
Физика	2017/2018	2	5

Соединение Таблицы «Студенты» (students) и «Успеваемость» (progress)

Номер зачетной книжки	Ф. И. О.	Серия паспорта	Номер паспорта	Предмет	Учебный год	Семестр	Оценка
55500	Иванов Иван Петрович	0402	645327	Физика	2017/2018	1	5
55500	Иванов Иван Петрович	0402	645327	Математика	2017/2018	1	4
55800	Климов Андрей Иванович	0402	673211	Физика	2017/2018	1	4
55800	Климов Андрей Иванович	0402	673211	Физика	2017/2018	2	5

2.2.3. Ряд замечаний

- Определение реляционной системы требует, чтобы база данных только *воспринималась пользователем* как набор таблиц. Таблицы в реляционной системе являются **логическими,** а не физическими структурами. На самом деле, на физическом уровне система может использовать любую из существующих структур памяти (последовательный файл, индексирование, хэширование, цепочку указателей и т. п.), лишь бы существовала возможность отображать эти структуры в виде таблиц на логическом уровне.
- Данное положение можно сформулировать и по-другому: таблицы представляют собой *абстракцию* способа физического хранения данных, в которой все нюансы реализации на уровне физической памяти (размещение хранимых записей, упорядочение хранимых записей, кодировка хранимых данных, префиксы хранимых записей, хранимые структуры доступа, такие как индексы и т.д.) *скрыты от пользователя.*
- В данном случае термин *логическая структура* в терминологии ANSI/SPARC относится как к концептуальному, так и ко внешнему уровням. Дело в том, что и концептуальный, и внешний уровни в реляционной системе являются одинаково реляционными, и лишь внутренний или физический уровень не является таковым. На самом деле реляционная теория вообще не может определить внутренний уровень. Она определяет лишь то, как база данных представлена *пользователю.*

2.2.3. Ряд замечаний (продолжение)

- У реляционных баз данных есть одно замечательное свойство, определяемое так называемым **информационным принципом:** *все информационное наполнение базы данных представлено одним и только одним способом, а именно — явным заданием значений, помещенных в позиции столбцов в строках таблицы.* Этот метод представления — *единственно* возможный для реляционных баз данных (естественно, на логическом уровне). В частности, **нет никаких** *указателей,* связывающих одну таблицу с другой.
- Примечание. Не имеется в виду, что в ней не может быть указателей *на физическом уровне;* наоборот, они могут быть предусмотрены на этом уровне и в действительности наверняка существуют. Но сведения об организации физического хранения в реляционных системах скрыты от пользователя.

2.2.4. Более формальное определение реляционной модели

Реляционная модель — это абстрактная теория данных, основанная на некоторых областях математики (в основном на теории множеств и логике предикатов). В первом приближении **реляционная** модель состоит из следующих пяти компонентов.

1. Неограниченный набор **скалярных типов** (включая, в частности, *логический тип* или *истинностное значение).*
2. Генератор **типов отношений** и соответствующая интерпретация для сгенерированных типов отношений.
3. Возможность определения **переменных отношения** (relation variables) для указанных сгенерированных типов отношений.
4. Операция **реляционного присваивания** для присваивания реляционных значений указанным переменным отношения.
5. Неограниченный набор общих **реляционных операторов** (*реляционная алгебра* – см. лекцию 3*)* для получения значений отношений из других значений отношений.

2.2.5. Кортежи

- Если дана коллекция типов T_i (i = 1, 2, . . ., n), которые не обязательно все должны быть разными, то значением кортежа (или кратко *кортежем),* определенным с помощью этих типов (назовем его t), является множество упорядоченных троек в форме $<A_i,T_i,v_i>$, где A_i — имя атрибута, T_i — имя типа и v_i — значение типа T_i. Кроме того, кортеж t должен соответствовать требованиям:
- Значение n – это **степень,** или **арность** t.
- Упорядоченная тройка $<A_i, T_i, v_i>$ является **компонентом** t.
- Упорядоченная пара $<A_i, T_i>$ представляет собой **атрибут** t и однозначно определяется именем атрибута A_i.
- Значение v_i — это **значение атрибута**, соответствующее атрибуту A_i кортежа t. Тип T_i — это соответствующий **тип атрибута**. Полное множество атрибутов составляет **заголовок кортежа** t.
- **Тип кортежа** t определен заголовком t, а сам заголовок и этот тип кортежа имеют такие же атрибуты (и поэтому такие же имена и типы атрибутов) и такую же степень, как t.

MAJOR_P# : P#	MINOR_P# : P#	QTY : QTY
P2	P4	7

2.2.6. Отношения

- Отношение это математический термин. Необходимо различать переменную отношения (relation variable) и значение отношения.
- Каждое отношение, точнее, каждое *значение* отношения, состоит из двух частей: набора пар «*имя_столбца : имя_типа*» (**заголовка**) и множества кортежей, согласованных с этим заголовком (**тела**). Кардинальность **r** отношения определяется как кардинальность этого множества (*кардинальностью множества* называется количество элементов множества).
- Отношение r имеет такие же атрибуты (следовательно, такие же имена и типы атрибутов) и такую же степень, как заголовок.
- Тело отношения r представляет собой множество кортежей, имеющих один и тот же заголовок.

2.2.6.1. Смысл отношения

1. В определенном отношении r заголовок отношения r представляет собой определенный **предикат** (под *предикатом* подразумевается просто функция с истинностными значениями, которая, как и все функции, имеет ряд *формальных параметров).*

- Например, для отношения progress предикат будет таким: Студент, имеющий зачетную книжку с номером record_book, сдал экзамен по дисциплине discipline в семестре term учебного года acad_year и получил оценку mark.
- Здесь формальные параметры: record_book, discipline, term, acad_year, mark.

2. Каждая строка в теле отношения r представляет собой определенное **истинное высказывание,** полученное из предиката путем подстановки определенных значений *фактических параметров* соответствующего типа вместо *формальных параметров* этого предиката, т. е. путем конкретизации.

- Пример истинного утверждения: Студент, имеющий зачетную книжку с номером 55500, сдал экзамен по дисциплине «Физика» в 1 семестре 2017/2018 учебного года и получил оценку 5.

2.2.6.2. Свойства отношений

Каждое отношение обладает следующими свойствами:

1. Каждый кортеж содержит точно одно значение (соответствующего типа) для каждого атрибута. Атрибуты могут иметь **значения в виде отношений**.

2. Атрибуты не характеризуются каким-либо упорядочением (например, слева направо).

3. Кортежи не характеризуются каким-либо упорядочением (например, сверху вниз).

4. В отношении отсутствуют дубликаты кортежей.

2.2.7. Замкнутость реляционной системы

- Свойство **замкнутости** реляционных систем означает, что результат выполнения операции имеет тот же тип, что и объекты, над которыми проводилась операция (все они являются отношениями). Но это значит, что к *результатам операций можно снова применить какую-либо операцию.* Например, можно сформировать проекцию соединения, соединение двух сокращений, сокращение проекции и т.д. Другими словами, можно использовать **вложенные реляционные выражения**, т.е. выражения, в которых операнды представлены выражениями, а не простыми именами таблиц.
- Значения переменных отношения изменяются с помощью операций **реляционного присваивания,** причем привычные нам операции обновления **INSERT, UPDATE** и **DELETE** можно считать сокращенной формой записи операций реляционного присваивания определенных типов.

2.2.8. Базовые переменные отношения и представления

- Исходные (заданные) переменные отношения называются **базовыми переменными отношения,** а присвоенные им значения называются **базовыми** отношениями. Отношение, которое получено или может быть получено из базового отношения в результате выполнения каких-либо реляционных выражений, называется **производным** отношением.
- Реляционные системы обычно поддерживают еще один вид именованных переменных отношения, называемых **представлениями**. В любой конкретный момент их значение является *производным* отношением.
- Выражение, фактически **определяющее представление,** не вычисляется, а просто *запоминается* системой.
- Реляционным аналогом *внешнего представления* ANSI/SPARC обычно служит множество из нескольких переменных отношения, каждая из которых является представлением в реляционном смысле. *Внешняя схема* состоит из определений таких представлений. Из этого следует, что в реляционной модели представления являются одним из способов обеспечения **логической независимости от данных**.
- Базовые переменные отношения «реально существуют» в том смысле, что они воплощают в себе данные, которые действительно хранятся в базе данных.
- Представления, наоборот, «реально не существуют», а просто предоставляют различные способы просмотра «реальных» данных.

2.2.9. Различная терминология

Формальные термины	Альтернатива 1	Альтернатива 2
Relation (отношение)	Table (таблица)	File (файл)
Tuple (кортеж)	Row (строка)	Record (запись)
Attribute (атрибут)	Column (столбец)	Field (поле)

Литература

1. Гарсиа-Молина, Г. Системы баз данных. Полный курс : пер. с англ. / Гектор Гарсиа-Молина, Джеффри Ульман, Дженнифер Уидом. – М. : Вильямс, 2003. – 1088 с.
2. Грофф, Дж. SQL. Полное руководство : пер. с англ. / Джеймс Р. Грофф, Пол Н. Вайнберг, Эндрю Дж. Оппель. – 3-е изд. – М. : Вильямс, 2015. – 960 с.
3. **Дейт, К. Дж. Введение в системы баз данных : пер. с англ. / Крис Дж. Дейт. – 8-е изд. – М. : Вильямс, 2005. – 1328 с.**
4. **Коннолли, Т. Базы данных. Проектирование, реализация и сопровождение. Теория и практика : пер. с англ. / Томас Коннолли, Каролин Бегг. – 3-е изд. – М. : Вильямс, 2003. – 1436 с.**
5. Кузнецов, С. Д. Основы баз данных : учеб. пособие / С. Д. Кузнецов. – 2-е изд., испр. – М. : Интернет-Университет Информационных Технологий ; БИНОМ. Лаборатория знаний, 2007. – 484 с.
6. Лузанов, П. PostgreSQL для начинающих / П. Лузанов, Е. Рогов, И. Лёвшин ; Postgres Professional. – М., 2017. – 146 с.
7. Моргунов, Е. П. Язык SQL. Базовый курс : учеб.-практ. пособие. / Е. П. Моргунов ; под ред. Е. В. Рогова, П. В. Лузанова ; Postgres Professional. – М., 2017. – 257 с.
8. PostgreSQL [Электронный ресурс] : официальный сайт / The PostgreSQL Global Development Group. – http://www.postgresql.org.
9. Postgres Professional [Электронный ресурс] : российский производитель СУБД Postgres Pro : официальный сайт / Postgres Professional. – http://postgrespro.ru.

Задание

Для выполнения практических заданий необходимо использовать книгу:

Моргунов, Е. П. Язык SQL. Базовый курс : учеб.-практ. пособие / Под ред. Е. В. Рогова, П. В. Лузанова ; Postgres Professional. – М., 2017. – 257 с. https://postgrespro.ru/education/books/sqlprimer

1. Прочитать введение и главу 1.
2. Установить ОС Linux (Debian или другой). Указания по установке СУБД PostgreSQL приведены в главе 2, параграф 2.1. Можно воспользоваться виртуальной машиной VirtualBox или аналогичной. Можно использовать уже настроенную ОС Debian (в виде виртуальной машины), полученную у преподавателя.
3. Развернуть учебную базу данных «Авиаперевозки» https://postgrespro.ru/education/demodb (см. главу 2, параграф 2.3). Использовать версию БД от 13.10.2016.
4. Изучить материал главы 4. Запросы к базе данных выполнять с помощью утилиты psql, описанной в главе 2, параграф 2.2.