4.13 Типы баз данных. Реляционные БД. Нормальные формы РБД. Язык SQL

База Данных (БД) — структурированный организованный набор данных, описывающих характеристики какой-либо физической или виртуальной системы.
«Базой данных» часто упрощённо или ошибочно называют Системы Управления Базами Данных (СУБД). Нужно различать набор данных (собственно БД) и программное обеспечение, предназначенное для организации и ведения баз данных (СУБД).

Структура БД

Организация структуры БД формируется исходя из следующих соображений:
Адекватность описываемому объекту/системе — на уровне концептуальной и логической модели.

- Удобство использования для ведения учёта и анализа данных — на уровне так называемой физической модели.

Виды концептуальных и логических моделей БД — сетевая модель, иерархическая модель, реляционная модель (ER-модель), многомерная модель, объектная модель. По виду модели БД разделяются на: картотеки, сетевые, иерархические, реляционные, объектно-ориентированные, многомерны, дедуктивные.

На уровне физической модели электронная БД представляет собой файл или их набор в формате TXT, CSV, Excel, DBF, XML либо в специализированном формате конкретной СУБД. Также в СУБД в понятие физической модели включают специализированные виртуальные понятия, существующие в её рамках — таблица, табличное пространство, сегмент, куб, кластер и т.д. В настоящее время наибольшее распространение получили реляционные базы данных.

Картотеками пользовались до появления электронных баз данных. Сетевые и иерархические базы данных считаются устаревшими, объектно-ориентированные пока никак не стандартизированы и не получили широкого распространения. Некоторое возрождение получили иерархические базы данных в связи с появлением и распространением XML.

Реляционная база данных — база данных, основанная на реляционной модели. Слово «реляционный» происходит от английского «relation» («отношение», только не связей таблиц, а заимствованное из теории множеств). Для работы с реляционными БД применяют Реляционные СУБД.

Теория реляционных баз данных была разработана доктором Коддом из компании IBM в 1969 году. В реляционных базах данных все данные представлены в виде простых таблиц, разбитых на строки и столбцы, на пересечении которых расположены данные. Запросы к таким таблицам возвращают таблицы, которые сами могут становиться предметом дальнейших запросов. Каждая база данных может включать несколько таблиц. Кратко особенности реляционной базы данных можно сформулировать следующим образом:

- Данные хранятся в таблицах, состоящих из столбцов ("атрибутов") и строк ("записей", "кортежей");
- На пересечении каждого столбца и строчки стоит в точности одно значение;
- У каждого столбца есть своё имя, которое служит его названием, и все значения в одном столбце имеют один тип;
- Запросы к базе данных возвращают результат в виде таблиц, которые тоже могут выступать как объект запросов;

Строки в реляционной базе данных неупорядочены - упорядочивание производится в момент формирования ответа на запрос.

Общепринятым стандартом языка работы с реляционными базами данных является язык SQL.

Нормализация

4.13

Целью нормализации является устранение недостатков структуры базы данных, приводящих к вредной избыточности в данных, которая в свою очередь потенциально приводит к различным аномалиям и нарушениям целостности данных.
Теоретики реляционных баз данных в процессе развития теории выявили и описали типичные примеры избыточности и способы их устранения. Нормальная форма -- формальное свойство отношения, которое характеризует степень избыточности хранимых данных и возможные проблемы. Наиболее известные и важные нормальные формы:

Первая нормальная форма (1НФ, 1NF) - Таблица находится в первой нормальной форме, если каждый её атрибут атомарен и все строки различны. Под выражением «атрибут атомарен» понимается, что атрибут может содержать только одно значение. Таким образом, не соответствуют 1NF таблицы, в полях которых могут храниться списки значений. Для приведения таблицы к 1NF обычно требуется разбить таблицу на несколько отдельных таблиц.

Вторая нормальная форма (2НФ, 2NF) - Таблица находится во второй нормальной форме, если она находится в первой нормальной форме, и при этом любой её атрибут, не входящий в состав первичного ключа, функционально полно зависит от первичного ключа. Функционально полная зависимость означает, что атрибут функционально зависит от всего первичного ключа, но при этом не находится в функциональной зависимости от какой-либо его части.

Третья нормальная форма (3НФ, 3NF) - Таблица находится в третьей нормальной форме, если она находится во второй нормальной форме, и при этом любой её неключевой атрибут функционально зависит *только* от первичного ключа.
При решении практических задач в большинстве случаев третья нормальная форма является достаточной. Процесс проектирования реляционной базы данных, как правило, заканчивается приведением к 3NF.

Третья усиленная нормальная форма, или нормальная форма Бойса-Кодда (НФБК, BCNF) - Таблица находится в BCNF, если она находится в третьей нормальной форме, и при этом отсутствуют функциональные зависимости атрибутов первичного ключа от не-ключевых атрибутов. Данная нормальная форма — это модификация третьей нормальной формы. Таблица может находиться в 3NF, но не в BCNF, только в одном случае: если она имеет, помимо первичного ключа, ещё по крайней мере один составной возможный ключ, и по крайней мере один из атрибутов таблицы входит и в первичный, и в возможный ключи. Такое бывает достаточно редко, в остальном 3NF и BCNF эквивалентны.

Четвёртая нормальная форма (4НФ, 4NF) - Таблица находится в 4NF, если она находится в BCNF и не содержит нетривиальных многозначных зависимостей. Многозначная зависимость не является функциональной, она существует в том случае, когда из факта, что в таблице содержится некоторая строка X, следует, что в таблице обязательно существует некоторая определённая строка Y.
То есть, таблица находится в 4NF, если все ее многозначные зависимости являются функциональными.

Пятая нормальная форма (5НФ, 5NF)- Таблица находится в 5NF, если она находится в 4NF и любая многозначная зависимость соединения в ней является тривиальной.
Пятая нормальная форма в большей степени является теоретическим исследованием, и практически не применяется при реальном проектировании баз данных. Это связано со сложностью определения самого наличия зависимостей «проекции — соединения», поскольку утверждение о наличии такой зависимости должно быть сделано для *всех* возможных состояний БД.

Доменно-ключевая нормальная форма (ДКНФ, DKNF).

Каждая следующая нормальная форма в этом списке (кроме ДКНФ) в некотором смысле является более совершенной, чем предыдущая, с точки зрения устранения избыточности.

4.13

Денормализация (denormalization) — намеренное приведение структуры базы данных в состояние, не соответствующее критериям нормализации, обычно проводимое с целью ускорения операций чтения из базы за счет добавления избыточных данных.

SQL (*Structured Query Language* — язык структурированных запросов) — универсальный компьютерный язык, применяемый для создания, модификации и управления данными в реляционных базах данных. Вопреки существующим заблуждениям, SQL является информационно-логическим языком, а не языком программирования.SQL основывается на реляционной алгебре.Язык SQL делится на три части:

- операторы определения данных (*Data Definition Language, DDL*)
- операторы манипуляции данными (*Data Manipulation Language, DML*)
- операторы определения доступа к данным (*Data Control Language, DCL*)

История

В начале 1970-х годов в компании IBM была разработана экспериментальная СУБД «System R» на основе языка *SEQUEL* (*Structured English Query Language* — структурированный английский язык запросов). Позже по юридическим соображениям язык *SEQUEL* был переименован в *SQL*. Когда в 1986 году первый стандарт языка SQL был принят ANSI (American National Standards Institute), официальным произношением стало— *эс-кью-эл*. Несмотря на это, даже англоязычные специалисты по прежнему часто называют SQL *сиквел*. Целью разработки было создание простого непроцедурного языка, которым мог воспользоваться любой пользователь, даже не имеющий навыков программирования.

Собственно разработкой языка запросов занимались Чэмбэрлин (Chamberlin) и Рэй Бойс (Ray Boyce). Пэт Селинджер (Pat Selinger) занималась разработкой стоимостного оптимизатора (cost-based optimizer), Рэймонд Лори (Raymond Lorie) занимался компилятором запросов.

В 1981 году IBM объявила о своём первом основанном на SQL программном продукте — SQL/DS. Чуть позже к ней присоединились Oracle, Relational Technology и другие производители.

Преимущества

Независимость от конкретной СУБД

Несмотря на наличие диалектов и различий в синтаксисе, в большинстве своём тексты SQL-запросов, содержащие DDL и DML, могут быть достаточно легко перенесены из одной СУБД в другую. Существуют системы, разработчики которых изначально закладывались на применение по меньшей мере нескольких СУБД (например: система электронного документооборота Documentum может работать как с Oracle Database, так и с Microsoft SQL Server и IBM DB2)

Наличие стандартов

Наличие стандартов и набора тестов для выявления совместимости и соответствия конкретной реализации SQL общепринятому стандарту только способствует «стабилизации» языка.

Декларативность

С помощью SQL программист описывает только какие данные нужно извлечь или модифицировать. То, каким образом это сделать решает СУБД непосредственно при обработке SQL запроса.

Недостатки

Несоответствие реляционной модели данных

Создатель реляционной модели данных Эдгар Кодд, Кристофер Дейт и их сторонники указывают на то, что SQL не является истинно реляционным языком. В частности они указывают на следующие проблемы SQL:

- Повторяющиеся строки
- Неопределённые значения (nulls)

4.13

- Явное указание порядка колонок слева направо
- Колонки без имени и дублирующиеся имена колонок
- Отсутствие поддержки свойства «=»
- Использование указателей
- Высокая избыточность

В опубликованном Кристофером Дейтом и Хью Дарвеном Третьем Манифесте они излагают принципы СУБД следующего поколения и предлагают язык Tutorial D, который является подлинно реляционным.

Сложность

Хотя SQL и задумывался как средство работы конечного пользователя, в конце концов он стал настолько сложным, что превратился в инструмент программиста.

Отступления от стандартов

Несмотря на наличие международного стандарта ANSI SQL-92, многие компании, занимающиеся разработкой СУБД (например, Oracle, Sybase, Microsoft, MySQL AB, Borland), вносят изменения в язык SQL, применяемый в разрабатываемой СУБД, тем самым отступая от стандарта. Таким образом появляются специфичные для каждой конкретной СУБД диалекты языка SQL.

В частности, для управления планом выполнения запроса в некоторые реализации SQL добавлены подсказки.

Сложность работы с иерархическими структурами

Ранее SQL не предлагал стандартного способа манипуляции древововидными структурами. Некоторые поставщики СУБД предлагали свои решения. Например, Oracle использует выражение «CONNECT BY». В настоящее время в качестве стандарта принята рекурсивная конструкция «WITH».

Стандарты

986	SQL-86, SQL-87	Впервые опубликован ANSI. Ратифицирован ISO в 1987.
1989	SQL-89	Небольшие изменения.
1992	SQL-92, SQL-2	Существенные изменения.
1999	SQL:1999, SQL-3	Добавлены регулярные выражения, рекурсивные запросы, триггеры и некоторые обьекто-ориентированные нововведения.
2003	SQL:2003	Объявлены XML-зависимые нововведения.

Процедурные расширения

Поскольку SQL не является языком программирования (то есть не предоставляет средств для автоматизации операций с данными), вводимые разными производителями расширения касались в первую очередь процедурных расширений. Это хранимые процедуры (*stored procedures*) и процедурные языки-"надстройки". Практически в каждой СУБД применяется свой процедурный язык. Например, в Oracle — PL/SQL, основанный на языке Ada; в Sybase SQL Server и MS SQL Server — *Transact-SQL*.