Лекция 2. SQL

SQL (аббр. от англ. Structured Query Language – «язык структурированных запросов») — декларативный язык программирования, применяемый для создания, модификации и управления данными в реляционной базе данных, управляемой соответствующей системой управления базами данных.

Является, прежде всего, информационно-логическим языком, предназначенным для описания, изменения и извлечения данных, хранимых в реляционных базах данных. В общем случае SQL (без ряда современных расширений) считается языком программирования неполным по Тьюрингу, но вместе с тем стандарт языка спецификацией SQL/PSM предусматривает возможность его процедурных расширений.

Изначально SQL был основным способом работы пользователя с базой данных и позволял выполнять следующий набор операций:

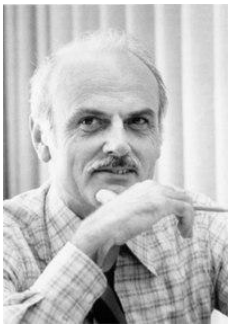
- создание в базе данных новой таблицы;
- добавление в таблицу новых записей;
- изменение записей;
- удаление записей;
- выборка записей из одной или нескольких таблиц (в соответствии с заданным условием);
- изменение структур таблиц.

Со временем SQL усложнился — обогатился новыми конструкциями, обеспечил возможность описания и управления новыми хранимыми объектами (например, индексы, представления, триггеры и хранимые процедуры) — и стал приобретать черты, свойственные языкам программирования.

При всех своих изменениях SQL остаётся самым распространённым лингвистическим средством для взаимодействия прикладного программного обеспечения с базами данных.

Современные СУБД, а также информационные системы, использующие СУБД, предоставляют пользователю развитые средства визуального построения запросов.

История языка (краткая)



Эдгар Кодд

Дональд Чемберлин (Donald D. Chamberlin) и Раймонд Бойс начали работу над языком реляционных баз данных после того, как узнали о **реляционной модели от Э. Кодда**, встретившись с ним на симпозиуме в Нью-Йорке в 1972 году. По воспоминаниям Чемберлина, это было «откровением». Авторы были впечатлены выразительностью и компактностью реляционной алгебры и реляционного исчисления, предложенных Э. Коддом, для представления сложных запросов. Кодд использовал символическую запись с математическими обозначениями операций, но Чемберлин и Бойс захотели спроектировать язык так, чтобы им мог воспользоваться любой пользователь, даже не имеющий навыков программирования и знаний математики.

Первый прототип языка SQL представила в 1979 году компания-разработчик Oracle. Сначала это был простейший инструмент для извлечения нужных данных, вроде фильтров в Excel-таблицах. С годами он усложнился, и теперь его применяют в качестве одного из основных инструментов для обработки данных.

После симпозиума Кодда, Чемберлин и Бойс провели следующий год в экспериментах над дизайном языка. Первой их попыткой был **язык SQUARE (Specifying Queries in A Relational Environment**), но он был сложен в практическом использовании из-за математической нотации с верхними и нижними индексами.

После переезда в Исследовательскую лабораторию Сан-Хосе в 1973 году, Чемберлин и Бойс начали работу над совершенно новым языком, который был назван **SEQUEL (от Structured English QUEry Language**, «английский язык структурированных запросов»). Авторы надеялись, что после небольшой практики даже пользователи-неспециалисты (например, бухгалтеры, инженеры, архитекторы, градостроители) смогут читать запросы так, словно последние написаны на обычном английском языке. Язык был назван «декларативным», поскольку он описывал желаемый результат, а не детальный план поиска этой информации. Предполагалось, что переводом декларативного запроса в детальный план исполнения будет заниматься специальный **оптимизирующий компилятор**.

Язык SEQUEL продолжил развитие как **часть проекта IBM System R** (первая реляционная СУБД). Пэт Селинджер (Pat Selinger) занималась разработкой **стоимостного оптимизатора** (cost-based optimizer), Рэймонд Лори (Raymond Lorie) занимался **компилятором запросов**.

В 1977 году авторы получили письмо от юриста британской авиастроительной группы компаний Hawker Siddeley, уведомляющее, что этой компании принадлежит торговая марка «SEQUEL». Было принято решение сократить название языка до **SQL (от Structured Query Language, «язык структурированных запросов»)**. Тем не менее,

первоначальное название SEQUEL повлияло на современное произношение аббревиатуры SQL

Первыми СУБД, поддерживающими новый язык, стали в 1979 году **Oracle** V2 для машин VAX компании Relational Software (впоследствии ставшей компанией Oracle) и **System/38** фирмы IBM, основанная на System/R.

Несмотря на то, что аббревиатура SQL формально должна произносится как «эс-кью-эл» ([ˈɛsˈkjuˈɛl]), первоначальная версия сокращённого названия языка — SEQUEL — совпадала по написанию со словом sequel и до сих пор часто произносится как «си́квел» (/ˈsiːkwəl/)

Стандартизация

Поскольку к началу 1980-х годов существовало **несколько вариантов СУБД** разных производителей, причём, каждый из них обладал собственной реализацией языка запросов, было принято решение разработать стандарт языка, который будет гарантировать переносимость ПО с одной СУБД на другую (при условии, что они будут поддерживать этот стандарт).

В 1983 году Международная организация по стандартизации (ISO) и Американский национальный институт стандартов (ANSI) приступили к разработке стандарта языка SQL. По прошествии множества консультаций и отклонения нескольких предварительных вариантов, в 1986 году ANSI представил свою первую версию стандарта, под названием **«Database Language — SQL» (Язык баз данных SQL)**. Неофициально этот стандарт SQL-86 получил название **SQL1**.

Со временем к стандарту накопилось несколько замечаний и пожеланий, особенно с точки зрения обеспечения целостности и корректности данных, в результате чего в 1989 году данный стандарт был расширен, получив название **SQL89.** В частности, в него была добавлена концепция первичного и внешнего ключей.

Сразу после завершения работы над стандартом SQL1 в 1987 году была начата работа над новой версией стандарта, который должен был заменить стандарт SQL89, получив название **SQL2**, поскольку дата принятия документа на тот момент была неизвестна. Таким образом, фактически **SQL89 и SQL2 разрабатывались параллельно**. Новая версия стандарта была принята в 1992 году, заменив стандарт SQL89. Новый стандарт, озаглавленный как **SQL92**, представлял собой, по сути, расширение стандарта SQL1, включив в себя множество дополнений, имевшихся в предыдущих версиях инструкций

История версий стандарта:

Год	Название	Иное название	Изменения
1986	SQL-86	SQL-87	Первый вариант стандарта, принятый институтом ANSI и одобренный ISO в 1987 году.
1989	SQL-89	FIPS 127-1	Немного доработанный вариант предыдущего стандарта.
1992	SQL-92	SQL2, FIPS 127-2	Значительные изменения (ISO 9075); уровень *Entry Level* стандарта SQL-92 был принят как стандарт FIPS 127-2.
1999	SQL:1999	SQL3	Добавлена поддержка регулярных выражений, рекурсивных запросов, поддержка триггеров, базовые процедурные расширения, нескалярные типы данных и некоторые объектно-ориентированные возможности.
2003	SQL:2003		Введены расширения для работы с XML-данными, оконные функции (применяемые для работы с OLAP-базами данных), генераторы последовательностей и основанные на них типы данных.
2006	SQL:2006		Функциональность работы с XML-данными значительно расширена. Появилась возможность совместно использовать в запросах SQL и XQuery.
2008	SQL:2008		Улучшены возможности оконных функций, устранены некоторые неоднозначности стандарта SQL:2003[13]
2011	SQL:2011		Реализована поддержка хронологических баз данных (PERIOD FOR), поддержка конструкции FETCH[14].
2016	SQL:2016		Защита на уровне строк, полиморфные табличные функции, JSON.
2023	SQL:2023		Операции над графами. Агрегатная функция ANY_VALUE(). Поддержка шестнадцатеричных/двоичных/восьмеричных литералов, улучшение поддержки JSON.[15]

Элементы языка

Язык SQL представляет собой совокупность операторов, инструкций, вычисляемых функций.

Согласно общепринятому стилю программирования, операторы (и другие зарезервированные слова) в SQL обычно **рекомендуется писать прописными буквами**.

Операторы SQL делятся на:

операторы определения данных (Data Definition Language, DDL):

- CREATE создаёт объект базы данных (саму базу, таблицу, представление, пользователя и так далее),
- ALTER изменяет объект,
- DROP удаляет объект;

операторы манипуляции данными (Data Manipulation Language, DML):

- SELECT выбирает данные, удовлетворяющие заданным условиям,
- INSERT добавляет новые данные,
- UPDATE изменяет существующие данные,
- DELETE удаляет данные;

операторы определения доступа к данным (Data Control Language, DCL):

- GRANT предоставляет пользователю (группе) разрешения на определённые операции с объектом,
- REVOKE отзывает ранее выданные разрешения,
- DENY задаёт запрет, имеющий приоритет над разрешением;

операторы управления транзакциями (Transaction Control Language, TCL):

- COMMIT применяет транзакцию,
- ROLLBACK откатывает все изменения, сделанные в контексте текущей транзакции,
- SAVEPOINT делит транзакцию на более мелкие участки.

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.

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 то, какие данные нужно извлечь или модифицировать. То, каким образом это сделать, решает СУБД непосредственно при обработке SQL-запроса. Однако программисту при этом полезно представлять, как СУБД будет разбирать текст его запроса. Чем сложнее сконструирован запрос, тем больше он допускает вариантов написания, различных по скорости выполнения, но одинаковых по итоговому набору данных.

Недостатки

Несоответствие реляционной модели данных. Создатели реляционной модели данных Эдгар Кодд, Кристофер Дейт и их сторонники указывают на то, что SQL не является истинно реляционным языком с точки зрения реляционной теории:

- допущение строк-дубликатов в таблицах и результатах выборок, что в рамках реляционной модели данных невозможно и недопустимо;
- поддержка неопределённых значений (NULL), создающая фактически многозначную логику;
- значимость порядка столбцов, возможность ссылок на столбцы по номерам (в реляционной модели столбцы должны быть равноправны);
- допущение столбцов без имени, дублирующихся имён столбцов.

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

Отступления от стандартов. Несмотря на наличие международного стандарта ANSI SQL-92, многие разработчики СУБД 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.

Сложность работы с иерархическими структурами. Ранее диалекты SQL большинства СУБД не предлагали способа манипуляции древовидными структурами. В настоящее время в ANSI стандартизована **рекурсивная конструкция.** Но в разных СУБД от разных производителей появились рекурсивные запросы (примерно 2005 г.).

Процедурные расширения

Поскольку SQL не является привычным процедурным языком программирования (то есть не предоставляет средств для построения циклов, ветвлений и так далее), 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, в частности:

- в Oracle Database а также в DB2 и Timesten используется **PL/SQL**,
- в СУБД Interbase и Firebird — **PSQL**,
- в СУБД DB2 — **SQL PL**,
- в СУБД MicroSoft SQL Server и Adaptive Server Enterprise — **Transact-SQL**,
- в СУБД PostgreSQL — **PL/pgSQL**.

Команда Select (SQL)

SELECT (от англ. select — «выбрать») — оператор запроса (DML/DQL) в языке SQL, возвращающий набор данных (выборку) из базы данных.

Select — многозначный термин.
select (Unix)[англ.] — системный вызов UNIX
SELECT — оператор запроса в SQL
Select (альбом) — второй студийный альбом британской певицы Ким Уайлд
MTV Select[англ.] — телевизионная программа
Select[англ.] — американская кредитная карта
<select></select> — элемент HTML
quick select[англ.] — алгоритм
Select Sport — датская спортивная фирма.
«Select» — британский музыкальный журнал.

Оператор **возвращает ноль или более строк**. Список возвращаемых столбцов задается в части оператора, называемой предложением SELECT. Поскольку SQL является декларативным языком, запрос SELECT определяет лишь требования к возвращаемому набору данных, но не является точной инструкцией по их вычислению. **СУБД транслирует запрос SELECT во внутренний план исполнения** («query plan»), который может различаться даже для синтаксически одинаковых запросов и от конкретной СУБД.

Оператор SELECT состоит из нескольких предложений (разделов):

- **SELECT** определяет список возвращаемых столбцов (как существующих, так и вычисляемых), их имена, ограничения на уникальность строк в возвращаемом наборе, ограничения на количество строк в возвращаемом наборе;
- **FROM** задаёт табличное выражение, которое определяет базовый набор данных для применения операций, определяемых в других предложениях оператора;
- **WHERE** задает ограничение на строки табличного выражения из предложения FROM;
- **GROUP BY** объединяет ряды, имеющие одинаковое свойство с применением агрегатных функций
- **HAVING** выбирает среди групп, определённых параметром GROUP BY
- **ORDER BY** задает критерии сортировки строк; отсортированные строки передаются в точку вызова. Сортировка может производиться как по возрастанию, так и по убыванию значений. Параметр ASC (по умолчанию) устанавливает порядок сортировки по возрастанию, от меньших значений к большим. Параметр DESC устанавливает порядок сортировки по убыванию, от больших значений к меньшим.

Оператор SELECT имеет следующую структуру:

SELECT

[**DISTINCT** | DISTINCTROW | **ALL**]

select_expression,...

FROM table_references

[**WHERE** where_definition]

[**GROUP BY** {unsigned_integer | col_name | formula}]

[**HAVING** where_definition]

[**ORDER BY** {unsigned_integer | col_name | formula} [**ASC** | **DESC**], ...]

Самый простой запрос вернёт все столбцы всех строк данной таблицы:

```
SELECT * FROM Table
```

А вот «посложнее»:

```
SELECT
  DeptID,
  SUM(SaleAmount)
FROM
  Sales
WHERE
  SaleDate = '01-Jan-2000'
GROUP BY
  DeptID
HAVING
  SUM(SaleAmount) > 1000
```

Возвращает список идентификаторов отделов, продажи которых превысили 1000 за 1 января 2000 года, вместе с суммами продаж за этот день:

Агрегатные функции

Агрегатные функции в SQL позволяют работать не только с отдельными строками, но и производить аналитику по всем строкам или по группе строк.

Агрегатная функция – это функция, которая выполняет вычисление на наборе значений и *возвращает одиночное значение*.

Примеры агрегатных функций:

- **AVG**(выражение) – арифметическое среднее;
- **MIN**(выражение) – минимальное значение выражения;
- **MAX**(выражение) – максимальное значение выражения;
- **SUM**(выражение) – сумма значений выражения;
- **COUNT**(*) – количество строк в результате запроса;
- **COUNT**(выражение) – количество значений выражения, не равных NULL.

Чтобы вызвать агрегатную функцию, необходимо указать её в списке выборки (в разделе SELECT).

Примеры:

Запрос с использованием агрегатной функции AVG:

```
SELECT HomeType, AVG(price) as AvgPrice FROM Rooms
GROUP BY HomeType
```

Запрос к таблице Employees, который будет считать количество сотрудников для каждой роли:

```
SELECT Role, COUNT(*)EmpoyeeNumber FROM Employees GROUP BY Role;
```