

# Базы данных. Тема 5.

Язык SQL.

# Определение. История развития

- *SQL (Structured Query Language)* — Структурированный Язык Запросов — стандартный язык запросов по работе с реляционными БД. Язык SQL появился после реляционной алгебры, и его прототип был разработан в конце 70-х годов в компании IBM Research. Он был реализован в первом прототипе реляционной СУБД фирмы IBM System R.
- В дальнейшем этот язык применялся во многих коммерческих СУБД и в силу своего широкого распространения постепенно стал стандартом "де-факто" для языков манипулирования данными в реляционных СУБД.

# История развития

- Первый международный стандарт языка SQL был принят в 1989 г. (далее мы будем называть его SQL/89 или SQL1). Иногда стандарт SQL1 также называют стандартом ANSI/ISO, и подавляющее большинство доступных на рынке СУБД поддерживают этот стандарт полностью. Однако развитие информационных технологий, связанных с базами данных, и необходимость реализации переносимых приложений потребовали в скором времени доработки и расширения первого стандарта SQL.
- В конце 1992 г. был принят новый международный стандарт языка SQL, который в дальнейшем будем называть SQL/92 или SQL2. И он не лишен недостатков, но в то же время является существенно более точным и полным, чем SQL/89. В настоящий момент большинство производителей СУБД внесли изменения в свои продукты так, чтобы они в большей степени удовлетворяли стандарту SQL2.

# Реализация

- По традиции, как и со многими стандартами в IT-индустрии, с языком SQL возникла проблема: на каком-то этапе многие производители использующего SQL программного обеспечения решили, что функционал в текущей (на тот момент времени) версии стандарта недостаточен, и его желательно расширить. В результате у разных производителей систем управления баз данных (СУБД) в ходу разные диалекты SQL, в общем случае между собой несовместимые.

Среди недостатков использования стандартов оказывается ограничение гибкости и функциональных возможностей конкретной реализации. **Под реализацией языка SQL** понимается программный продукт SQL соответствующего производителя. Для расширения функциональных возможностей многие разработчики, придерживающиеся принятых стандартов, добавляют к стандартному языку SQL

# КОМАНДЫ ЯЗЫКА SQL

- Реализация в SQL концепции операций, ориентированных на табличное представление данных, позволила создать компактный язык с небольшим набором предложений. Язык SQL может использоваться как для выполнения запросов к данным, так и для построения прикладных программ.
- Основные категории команд языка SQL предназначены для выполнения различных функций, включая построение объектов базы данных и манипулирование ими, начальную загрузку данных в таблицы, обновление и удаление существующей информации, выполнение запросов к базе данных, управление доступом к ней и ее общее администрирование.

# Команды языка SQL

- Основные категории команд языка SQL:
- DDL – язык определения данных;
- DML – язык манипулирования данными;
- DQL – язык запросов ;
- DCL – язык управления данными;
- команды администрирования данных;
- команды управления транзакциями

# Преимущества языка SQL

- стандартность – как уже было сказано, использование языка SQL в программах стандартизировано международными организациями;
- независимость от конкретных СУБД – все
- распространенные СУБД используют SQL,
- т.к. реляционную базу данных можно перенести с одной СУБД на другую с минимальными доработками;
- возможность переноса с одной вычислительной системы на другую – СУБД может быть ориентирована на различные вычислительные системы, однако приложения, созданные с помощью SQL, допускают использование как для локальных БД, так и для крупных многопользовательских систем;

# Преимущества языка SQL

- реляционная основа языка – SQL является языком реляционных БД, поэтому он стал популярным тогда, когда получила широкое распространение реляционная модель представления данных. Табличная структура реляционной БД хорошо понятна, а потому язык SQL прост для изучения;
- возможность создания *интерактивных запросов* – SQL обеспечивает пользователям немедленный доступ к данным, при этом в интерактивном режиме можно получить результат *запроса* за очень короткое время без написания сложной программы;



# Преимущества языка SQL

- возможность программного доступа к БД – язык SQL легко использовать в приложениях, которым необходимо обращаться к базам данных. Одни и те же операторы SQL употребляются
- как для интерактивного, так и программного доступа, поэтому части программ, содержащие обращение к БД, можно вначале проверить в интерактивном режиме, а затем встраивать в программу;
- обеспечение различного представления данных – с помощью SQL можно представить такую структуру данных, что тот или иной пользователь будет видеть различные их представления. Кроме того, данные из разных частей БД могут быть скомбинированы и представлены в виде одной
- простой таблицы, а значит, представления пригодны для усиления защиты БД и ее настройки под конкретные
- требования отдельных пользователей;

# Преимущества языка SQL

- возможность динамического изменения и расширения структуры БД – язык SQL позволяет манипулировать структурой БД, тем самым обеспечивая гибкость с точки зрения приспособленности БД к изменяющимся требованиям предметной области;
- поддержка архитектуры *клиент-сервер* – SQL – одно из лучших средств для реализации приложений на платформе *клиент-сервер*. SQL служит связующим звеном между взаимодействующей с пользователем клиентской системой и серверной системой, управляющей БД, позволяя каждой из них сосредоточиться на выполнении своих функций.

# Возможности SQL

- создавать *базы данных и таблицы* с полным описанием их структуры;
- выполнять основные операции манипулирования данными, в частности, вставку, модификацию и удаление данных из *таблиц* ;
- выполнять простые и сложные *запросы*, осуществляющие преобразование данных.

# Язык SQL

- *SQL не процедурный язык*, в нем необходимо указывать, *какая информация* должна быть получена, а не как ее можно получить. Иначе говоря, *SQL* не требует указания методов доступа к данным. Как и большинство современных языков, он поддерживает свободный формат записи операторов.

# Язык SQL

- Язык SQL – первый и пока единственный *стандартный язык* для работы с базами данных, который получил достаточно широкое распространение.

Практически все крупнейшие разработчики СУБД в настоящее время создают свои продукты с использованием языка SQL либо с SQL-интерфейсом.

# Типы данных языка SQL

| <i>Тип данных</i> | Объявления                                |
|-------------------|---|
| Символьный        | CHAR   VARCHAR                            |
| Битовый           | BIT   BIT VARYING                         |
| Точные числа      | NUMERIC   DECIMAL  <br>INTEGER   SMALLINT |
| Округленные числа | FLOAT   REAL   DOUBLE<br>PRECISION        |
| Дата/время        | DATE   TIME  <br>TIMESTAMP                |
| Интервал          | INTERVAL                                  |

# Символьные данные

• **Символьные данные** состоят из последовательности символов, входящих в определенный создателями СУБД набор символов. Поскольку наборы символов являются специфическими для различных диалектов языка SQL, перечень символов, которые могут входить в состав значений *данных символьного типа*, также зависит от конкретной реализации

# Битовые данные

- Битовый тип данных используется для определения битовых строк, т.е. последовательности двоичных цифр (битов), каждая из которых может иметь значение либо 0, либо 1 .



# Точные числа

- ***Тип точных числовых данных*** применяется для определения чисел, которые имеют точное представление, т. е. числа состоят из цифр, необязательной десятичной точки и необязательного символа знака.

# Числа с плавающей точкой

- Числа с плавающей точкой применяется для описания *данных*, которые нельзя точно представить в компьютере, в частности действительных чисел. *Округленные числа* или числа с плавающей точкой представляются в научной нотации, при которой число записывается с помощью мантиссы, умноженной на определенную степень десяти (порядок), например:  $10E3$ ,  $+5.2E6$ ,  $-0.2E-4$

# Дата/врем

- **Тип данных "дата/время"**  
используется для определения моментов времени с некоторой установленной точностью.

# Команда CREATE DATABASE

Синтаксис команды CREATE DATABASE  
имеет

ВИД:

*CREATE DATABASE [IF NOT EXISTS]  
имя\_базы\_данных [спецификация\_create[,  
спецификация\_create]...] спецификация\_create:*

- [DEFAULT] CHARACTER SET имя\_набора\_символов
- [DEFAULT] COLLATE имя\_порядка\_сопоставления

# Использование базы данных

SELECT

DATA

```
mysql> SELECT DATABASE();  
+-----+  
| DATABASE |  
+-----+  
|          |  
+-----+  
1 rows in set (0.00 sec)
```

USE employees;

# Синтаксис команды CREATE TABLE

- CREATE [TEMPORARY] TABLE [IF NOT EXISTS]
- ИМЯ
- [(спецификация, ...)]
- [опция, ...]
- [ [IGNORE | REPLACE] запрос]

# Создание таблицы

```
CREATE TABLE employee_data (  
emp_id int unsigned not null auto_increment primary key,  
f_name varchar(20),  
l_name varchar(20), title varchar(30), age int,  
salary int, perks int,  
email varchar(60)  
);
```

# Ограничения для столбца

- **NOT NULL** - в любой добавляемой или изменяемой строке столбец всегда должен иметь значение, отличное от NULL .
- **UNIQUE** - все значения столбца должны быть уникальны.
- **PRIMARY KEY** - устанавливает один столбец как первичный ключ

и одновременно подразумевает, что все значения столбца будут уникальны.

- **CHECK** ( condition ) - указываемое в скобках условие использует для сравнения значение столбца и возвращает TRUE, FALSE или UNKNOWN. Если при попытке выполнения SQL-оператора возвращаемое значение равно FALSE, то оператор выполнен не будет.

- **REFERENCES** table ( fields\_list ) - ограничение требует совпадения значений столбцов данной таблицы с указанными столбцами *родительской таблицы*.



# Ограничения на родительский КЛЮЧ

- **CASCADE** - распространение изменений, произведенных в родительском ключе, на совпадающие строки внешнего ключа (для MATCH PARTIAL - только на уникально совпадающие строки).
- **SET NULL** - значения внешнего ключа изменяются на NULL по следующим правилам: для MATCH FULL - заменяются все значения внешнего ключа; для MATCH PARTIAL - в уникально совпадающих строках заменяются значения только тех столбцов, значения которых в родительском ключе были изменены; если тип совпадения не указан, то заменяются значения только тех столбцов, значения которых в родительском ключе были изменены.
- **SET DEFAULT** - значения внешнего ключа изменяются на значение по умолчанию по тем же правилам, что и для фразы SET NULL, но при типе соответствия MATCH FULL заменяются значения только тех столбцов внешнего ключа, которые уникально соответствуют родительскому ключу  
(значение NULL внешнего ключа не заменяется).
- **NO ACTION** - никаких действий во внешнем ключе не выполняется, допускаются только изменения родительского ключа, не нарушающие ссылочную целостность.

# Синтаксис команды DROP TABLE

```
DROP TABLE [IF EXISTS] таблица [RESTRICT  
|  
CASCADE]
```

# Удаление таблиц

```
mysql> SHOW TABLES;  
+-----+  
| Tables in employees |  
+-----+  
| employee data      |  
+-----+  
1 rows in set (0.00 sec)
```

```
mysql> DROP TABLE employee_data;  
Query OK, 0 rows affected (0.01 sec)
```

# Типы данных

|  |  |
|--|--|
| <code>TINYINT [ (M) ]</code><br><code>[UNSIGNED] [ZEROFILL]</code>   | Очень малое целое число. Диапазон со знаком от -128 до 127. Диапазон без знака от 0 до 255                               |
| <code>BIT, BOOL</code>   | Синонимы <code>TINYINT (1)</code>  |
| <code>SMALLINT [ (M) ]</code><br><code>[UNSIGNED] [ZEROFILL]</code>  | Малое целое число. Диапазон со знаком от - 32768 до 32767. Диапазон без знака от 0 до 65535.                             |
| <code>MEDIUMINT [ (M) ]</code><br><code>[UNSIGNED] [ZEROFILL]</code> | Целое число среднего размера. Диапазон со знаком от -8388608 до 8388607.<br>Диапазон без знака от 0 до 16777215          |
| <code>INT [ (M) ] [UNSIGNED]</code><br><code>[ZEROFILL]</code>       | Целое число нормального размера. Диапазон со знаком от -2147483648 до 2147483647. Диапазон без знака от 0 до 4294967295. |

# Типы данных

`INTEGER [ (M) ]`      Синоним для `INT`  
`[ UNSIGNED ]`  
`[ ZEROFILL ]`

`BIGINT [ (M) ]`      Большое целое число.  
`[ UNSIGNED ]`      Диапазон со знаком от -  
`[ ZEROFILL ]`      9223372036854775808 до  
9223372036854775807. Диапазон  
без знака от 0 до  
18446744073709551615

# Типы данных

FLLOAT (точно  
сть)  
ТОЧНОСТИ  
[UNSIGNED]  
[ZEROFILL]

Число с плавающей точкой. Атрибут может иметь значение  $\leq 24$  для числа с плавающей точкой обычной (одинарной) точности и между 25 и 53 - для числа с плавающей точкой удвоенной точности. Эти типы данных сходны с типами `FLLOAT` и `DOUBLE`, описанными ниже. `FLLOAT (X)` относится к тому же интервалу, что и соответствующие типы `FLLOAT` и `DOUBLE`, но диапазон значений и количество десятичных знаков не определены.

# Типы данных

`FLOAT (M, D) ]`

обычной

`[ZEROFILL]`  
`[UNSIGNED]`

Малое число с плавающей точкой точности. Допустимые значения: от -3,402823466E+38 до -1,175494351E-38, 0, и от 1,175494351E-38 до 3,402823466E+38. Если указан атрибут `UNSIGNED`, отрицательные значения недопустимы. Атрибут `M` указывает количество выводимых пользователю знаков, а атрибут `D` - количество разрядов, следующих за десятичной точкой.

Обозначение `FLOAT` без указания аргументов или запись вида `FLOAT (X)`, где `X`  $\leq 24$ , справедливы для числа с плавающей точкой обычной точности.

# Типы данных

DOUBLE [ (M, D) ]  
[ UNSIGNED ]  
ТОЧНОСТИ  
[ ZEROFILL ]

Число с плавающей точкой удвоенной нормального размера. Допустимые значения: от - 1,7976931348623157E+308 до - 2,2250738585072014E-308, 0, и от 2,2250738585072014E-308 до 1,7976931348623157E+308. Если указан атрибут `UNSIGNED`, отрицательные значения недопустимы. Атрибут `M` указывает количество выводимых пользователю знаков, а атрибут `D` - количество разрядов, следующих за десятичной точкой. Обозначение `DOUBLE` без указания аргументов или запись вида `FLOAT (X)`, где  $25 \leq X \leq 53$ , справедливы для числа с плавающей точкой двойной точности.



# Типы данных

```
DECIMAL [ (M[, D]  
) ] [ UNSIGNED ]
```

Ведет  
[ ZEROFILL ]

или

```
DEC [ (M[, D]) ]  
[ UNSIGNED ]  
[ ZEROFILL ]
```

или

```
NUMERIC [ (M[, D]  
) ] [ UNSIGNED ]  
учитываются  
[ ZEROFILL ]
```

"Неупакованное" число с плавающей точкой. себя подобно столбцу `CHAR`, содержащему

цифровое значение. Термин "неупакованное" означает, что число хранится в виде строки и при этом для каждого десятичного знака используется один символ.

Разделительный знак десятичных разрядов, а также

знак '-' для отрицательных чисел не в `M` (но место для них зарезервировано). Если атрибут `D` равен 0, величины будут представлены без десятичного знака, т.е. без дробной части.

Максимальный интервал значений типа `DECIMAL` тот же, что и для типа `DOUBLE`, но действительный интервал для конкретного столбца `DECIMAL` может быть ограничен выбором значений атрибутов `M` и `D`. Если указан атрибут `UNSIGNED`, отрицательные значения недопустимы. Если атрибут `D` не указан, его значение по умолчанию равно 0. Если не указан `M`, его

# Типы данных

## DATE

Дата. Поддерживается интервал от '1000-01-01' до '9999-12-31'. MySQL выводит значения `DATE` в формате 'YYYY-MM-DD', но можно установить значения в столбец `DATE`, используя как строки, так и числа.

## DATETIME

Комбинация даты и времени.

Поддерживается интервал от '1000-01-01 00:00:00' до '9999-12-31

23:59:59'. MySQL выводит значения `DATETIME`

в

формате 'YYYY-MM-DD HH:MM:SS', но можно устанавливать значения в столбце `DATETIME`, используя как строки, так и числа.

# Типы данных

`TIMESTAMP [ (M) ]`

Временная метка. Интервал от '1970-01-01 00:00:00' до некоторого значения времени в 2037 году. MySQL выводит значения `TIMESTAMP` в форматах `YYYYMMDDHHMMSS`, `YYMMDDHHMMSS`, `YYYYMMDD` или `YYMMDD` в зависимости от значений `M`: 14 (или отсутствующее), 12, 8, или 6; но можно также устанавливать значения в столбце `TIMESTAMP`, используя как строки, так и числа.

Столбец `TIMESTAMP` полезен для записи даты и времени при выполнении операций `INSERT` или `UPDATE`, так как при этом автоматически вносятся значения даты и времени самой последней операции, если эти величины не введены программой. Можно также устанавливать текущее значение даты и времени, задавая

# Типы данных

`TIME`

Время. Интервал от '-838:59:59' до '838:59:59'. MySQL выводит значения `TIME` в формате 'HH:MM:SS', но можно устанавливать значения в столбце `TIME`, используя как строки, так и числа.

`YEAR [ (2 | 4) ]`

Год в двухзначном или четырехзначном форматах (по умолчанию формат четырехзначный). Допустимы следующие значения: с 1901 по 2155, 0000 для четырехзначного формата года и 1970-2069 при использовании двухзначного формата (70-69). MySQL выводит значения `YEAR` в формате YYYY, но можно задавать значения в столбце `YEAR`, используя как строки, так и числа.

# Типы данных

[NATIONAL]

CHAR (M)

[BINARY]

Строка фиксированной длины, при хранении всегда дополняется пробелами в конце строки до заданного размера. Диапазон аргумента **M** составляет от 0 до 255 символов. Концевые пробелы удаляются при выводе значения. Если не задан атрибут чувствительности к регистру **BINARY**, то величины **CHAR** сортируются и сравниваются как независимые от регистра в соответствии с установленным по умолчанию алгоритмом. Атрибут **NATIONAL CHAR** (или его эквивалентная краткая форма **NCHAR**) представляет собой принятый в ANSI SQL способ указания, что в столбце **CHAR** должен использоваться установленный по умолчанию набор символов (**CHARACTER**).

# Типы данных

CHAR

Это синоним для CHAR (1) .

[NATIONAL]

Строка переменной длины.

VARCHAR (M)

Примечание: концевые пробелы

[BINARY]

удаляются при сохранении

значения (в этом заключается

отличие от спецификации ANSI SQL).

Диапазон аргумента M составляет

от 0 до 255 символов. Если не

задан атрибут чувствительности к

регистру BINARY, то величины

VARCHAR сортируются и

сравниваются как независимые от

регистра.

# Типы данных

TINYBLOB,  
TINYTEXT

Столбец типа BLOB или TEXT с максимальной длиной 255 символов.

BLOB, TEXT

Столбец типа BLOB или TEXT с максимальной длиной 65535 символов.

MEDIUMBLOB,  
MEDIUMTEXT

Столбец типа BLOB или TEXT с максимальной длиной 16777215 символов.

LOBLOB,  
LONGTEXT

Столбец типа BLOB или TEXT с максимальной длиной 4294967295 символов.

# Типы данных

- `ENUM (`  
`'значение1'`, строки может иметь только одно  
`'значение2'`, значение, выбранное из заданного списка  
`...)` величин `'значение1'`, `'значение2'`,  
`...`, `NULL` или специальная величина  
ошибки `""`. Список `ENUM` может содержать  
максимум 65535 различных величин
- `SET (`  
`'значение1'`, или более значений, каждое из которых  
`'значение2'`, должно быть выбрано из заданного  
`...)` списка  
величин `'значение1'`, `'значение2'`,  
`...` Список `SET` может содержать  
максимум  
64 элемента.



# Наиболее часто используемые

Тип **числовые типы полей** Байт От До

|           |   |                |                 |
|-----------|---|----------------|-----------------|
| TINYINT   | 1 | -128           | 127             |
| SMALLINT  | 2 | -32768         | 32767           |
| MEDIUMINT | 3 | -8388608       | 8388607         |
| INT       | 4 | -2147483648    | 2147483647      |
| BIGINT    | 8 | -              | 922337203685477 |
|           |   | 92233720368547 | 5807            |
|           |   | 75808          |                 |

# Нулевые значения для типов данных

Тип столбца даты и времени "Ноль"

|           |   |
|-----------|---|
| DATETIME  | '0000-00-00 00:00:00'   |
| DATE      | '0000-00-00'  |
| TIMESTAMP | 0000000000000000 (длина зависит от количества выводимых символов) |
| TIME      | '00:00:00'  |
| YEAR      | 0000  |

# TIMESTAMP

- Тип столбца `TIMESTAMP` обеспечивает тип представления данных, который можно использовать для автоматической записи текущих даты и времени при выполнении операций `INSERT` или `UPDATE`. При наличии нескольких столбцов типа `TIMESTAMP` только первый из них обновляется автоматически.

# Символьные типы данных

| Тип             | Макс.размер        | Байт       |
|-----------------|--------------------|------------|
| TINYTEXT        | $2^8-1$            | 255        |
| <b>ИЛИ</b>      |                    |            |
| TINYBLOB        |                    |            |
| TEXT <b>или</b> | $2^{16}-1$ (64K-1) | 65535      |
| BLOB            |                    |            |
| MEDIUMTEXT      | $2^{24}-1$ (16M-1) | 16777215   |
| <b>ИЛИ</b>      |                    |            |
| MEDIUMBLOB      |                    |            |
| LONGBLOB        | $2^{32}-1$ (4G-1)  | 4294967295 |

# Типы данных CHAR и VARCHAR

| Величина   | CHAR(4) | Требуемая<br>память | VARCHAR<br>(4) | Требуемая<br>память |
|------------|---------|---------------------|----------------|---------------------|
| "          | ' '     | 4 байта             | "              | 1 байт              |
| 'ab'       | 'ab '   | 4 байта             | 'ab'           | 3 байта             |
| 'abcd'     | 'abcd'  | 4 байта             | 'abcd'         | 5 байтов            |
| 'abcdefgh' | 'abcd'  | 4 байта             | 'abcd'         | 5 байтов            |