ВОПРОСЫ
ПО ПРЕДЫДУЩЕЙ ЛЕЦИИ

1. Что такое «нормализация» ? (применительно к реляционным базам данных)

2. В какой форме находится отношение?

СОТР_НОМЕР	СОТР_УРОВЕНЬ	СОТР_ЗАРПЛАТА
2934	2	12000
2935	3	13000
2936	1	10000
2937	1	10000

Единственный потенциальный ключ: СОТР_НОМЕР

Функциональные зависимости атрибутов:

СОТР_НОМЕР → СОТР_УРОВЕНЬ

СОТР_НОМЕР → СОТР_ЗАРПЛАТА

СОТР_УРОВЕНЬ → СОТР_ЗАРПЛАТА

3. Что такое модель ER ? (применительно к предметной области проектирования баз данных)

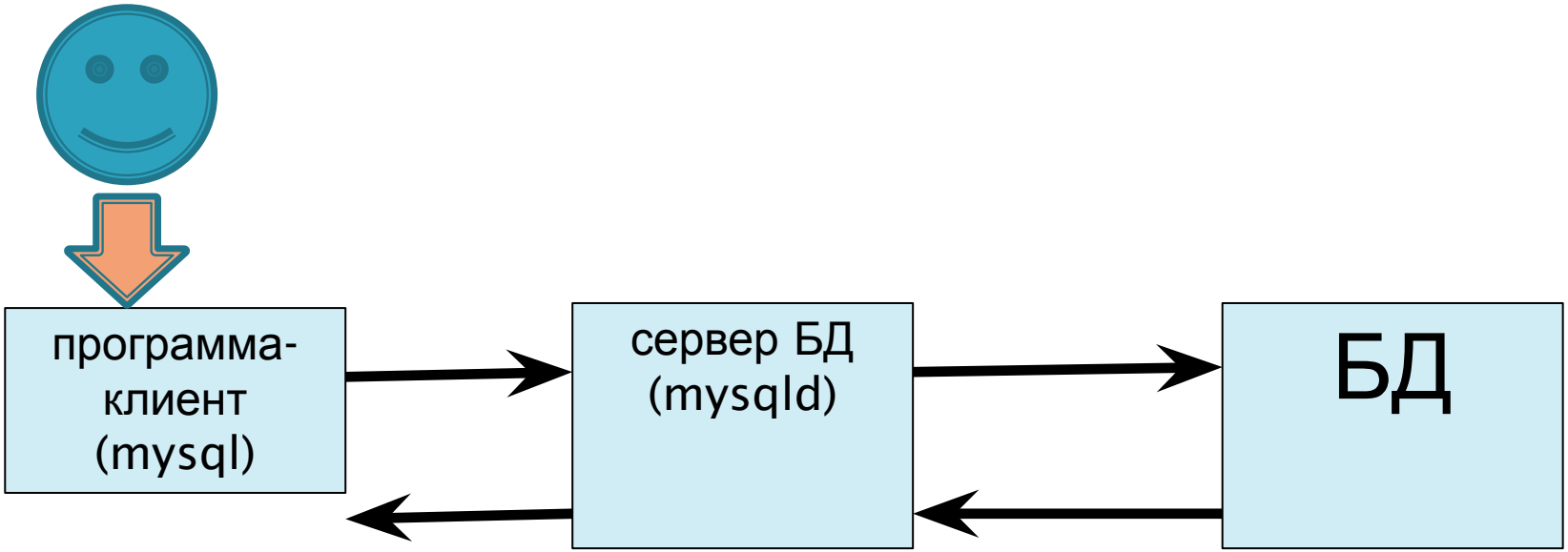
Лекция 3.

1. Общие сведения о СУБД MySQL.
2. Общие сведения о языке SQL.
3. Наборы символов и порядки сопоставления.
4. Некоторые команды языка определения данных (DDL).

1. Общие сведения о СУБД MySQL

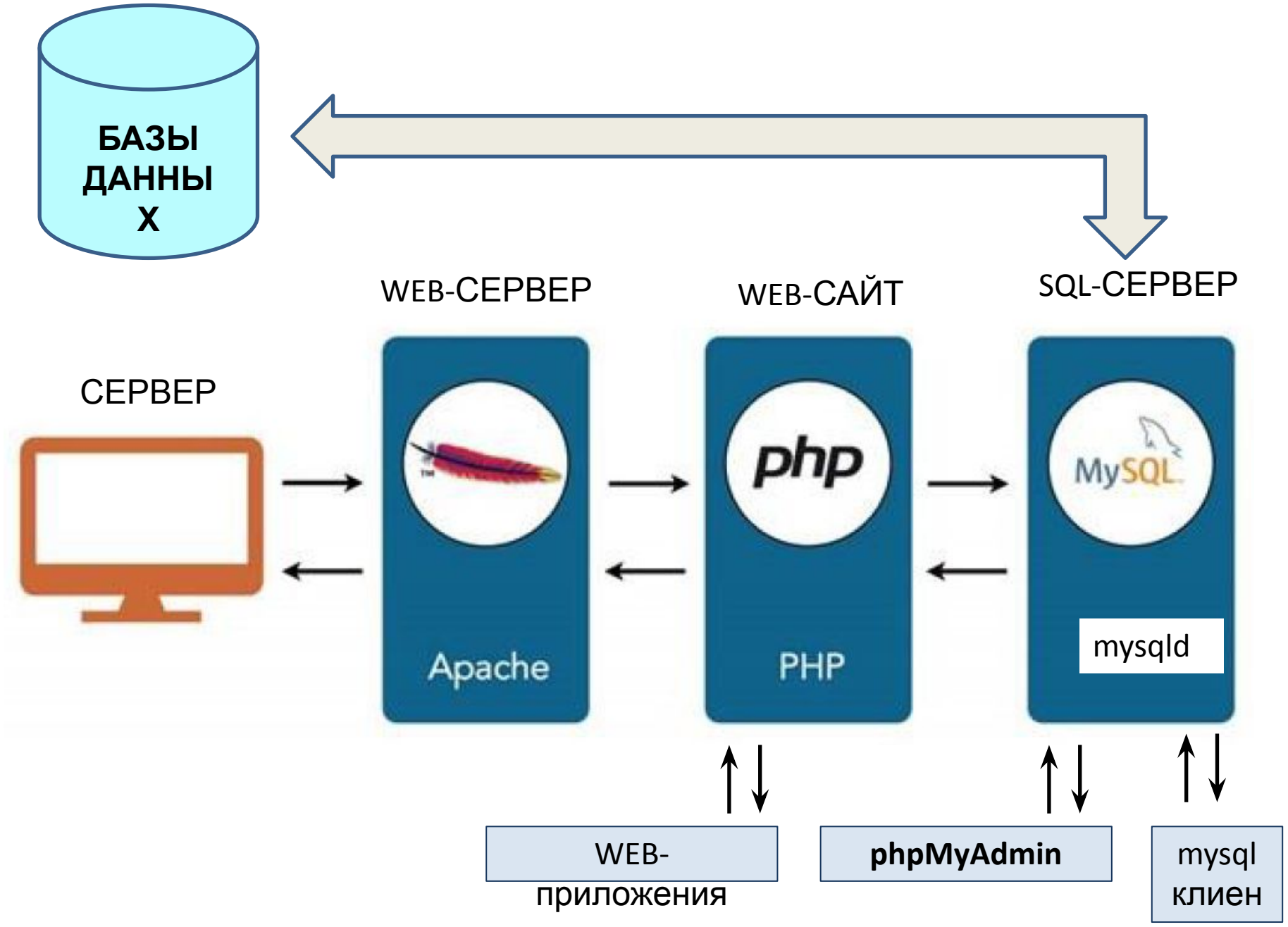
- **система управления реляционными базами данных**
- **открытый исходный код**
- **имеет клиент-серверную архитектуру (но может быть встроена в приложение)**
- **в состав СУБД включены:**
  - ***сервер MYSQL,***
  - ***набор клиентских программ ,***
  - ***инструменты администрирования ,***
  - ***программные интерфейсы приложений (API)***
- **использует ряд расширений языка SQL (относительно стандартов языка SQL)**

Схема взаимодействия пользователя с СУБД при работе за терминалом



- **Сервер баз данных** — управляет данными, хранящимися в **базе данных.**
- **Клиенты** отправляют серверу запросы.

Сервер обрабатывает каждый полученный запрос
и отправляет результаты соответствующему клиенту.


БАЗЫ ДАННЫХ
WEB-СЕРВЕР
WEB-САЙТ
SQL-СЕРВЕР
СЕРВЕР
Apache
PHP
mysqld
WEB-приложения
phpMyAdmin
mysql клиент

2. Общие сведения о языке SQL

- SQL (язык структурированных запросов) - язык, используемый для доступа к реляционным базам данных. Основан на теории реляционных отношений.

- Стандарты: ANSI/ISO
  1986→1992→1999→2003→2008

Основные подмножества SQL

Обозн.	Описание	Действия	Основные команды
DDL	(Data Definition Language) язык определения данных	Создание, изменение и удаление объектов БД (баз данных , таблиц и индексах в базах)	CREATE, ALTER, DROP
DML	(Data Manipulation Language) язык манипулирования данными	Вставка, изменение и удаление строк таблицы	INSERT, UPDATE, DELETE
DQL	(Data Query Language) язык запросов	Выборка строк таблицы	SELECT
DCL	(Data Control Language) язык управления данными	Управление доступом	GRANT, REVOKE
TPL	(Transaction Processing Language) язык обработки транзакций	Фиксация или откат транзакции, сохранение промежуточной точки	COMMIT, ROLLBACK, SAVEPOINT

Структура команд SQL

- Команда SQL всегда начинается с *действия* – слова или группы слов, описывающих выполняемую операцию
- Команда обычно содержит одну или несколько *секций (разделов)*, уточняющих ее смысл

```
SELECT Name FROM Student

INSERT INTO Student  VALUES (1, 'Иванов', 7333)

DELETE * FROM Student
```

Отличие от процедурных языков

- В отличие от команд обычных языков программирования, команды SQL указывают СУБД *какие* данные необходимо найти, но не сообщают, *как* это должно происходить
- Хорошо написанный SQL -запрос должен читаться почти также легко, как обычное предложение (на английском языке)

Типы лексем (минимальных смысловых элементов языка)

- Ключевое слово (зарезервированные слова: названия команд, функций и т.п.)
- Идентификатор (имена таблиц, столбцов и др. объектов БД)
- Константа (литерал, т.е. данные, интерпретируемые буквально)
- Специальный символ (скобки и т.п.)

Символы-разделители

- Пробел
- Символ табуляции
- Разрыв строки

Из последнего следует, что команда не обязательно должна располагаться на одной строке

Комментарии

- Однострочные комментарии начинаются
  с двух дефисов -- или знака #
- Многострочные – начинаются с последовательности /* и завершаются последовательностью */

Идентификаторы

- Идентификаторы могут начинаться с любой допустимой буквы или с символа подчеркивания, далее следует любое сочетание букв, цифр и символов подчеркивания
- *Регистр* ввода незащищенных идентификаторов не играет роли – в соответствии со стандартом SQL/92 при выполнении команды они должны по умолчанию преобразовываться к верхнему регистру

Ограничения

- Идентификаторы *не могут* содержать символов-разделителей
- Идентификаторы *не могут* начинаться с цифры
- Идентификаторы *не должны* совпадать с ключевыми словами

Примеры

- Правильные:

  Group7333

  g7334

  group_7334

- Неправильные:

  7333

Защищенные идентификаторы

- Идентификаторы, заключенные в двойные кавычки, называют *защищенные идентификаторами*. Например:

  "Группа 7334"

- Использование защищенных идентификаторов позволяет преодолеть рассмотренные ограничения
- Защищенные идентификаторы интерпретируются буквально – т.е. являются регистрозависимыми

Числовые константы

Числовые константы (целочисленные и вещественные) задаются в SQL так же, как и большинстве обычных языков программирования:

12

–134

1.5

Строковые константы

- *Строковые константы* представляют собой произвольную совокупность символов, заключенную апострофы (в одинарные кавычки)
- Если в самой константе встречается апостроф, при её определении следует удвоить его
- Если строковая константа представлена в кодировке Unicode, то ей должна предшествовать буква N

Примеры

'SQL/92'

'O''Reilly'

N'Иванов'

Типы данных

- целые
- вещественные
- строковые
- дата/время
- NULL

NULL

NULL — специальное значение (псевдозначение), которое может быть записано в поле таблицы БД.

Введено для того, чтобы различать в полях БД пустые (визуально не отображаемые) значения (например, строку нулевой длины) и отсутствующие значения (когда в поле не записано вообще никакого значения, даже пустого).

NULL

Трёхзначная логика

Таблица истинности

A	B	Not A	A ^ B	A v B
TRUE	TRUE	FALSE	TRUE	TRUE
TRUE	FALSE	FALSE	FALSE	TRUE
TRUE	Null	FALSE	Null	TRUE
FALSE	TRUE	TRUE	FALSE	TRUE
FALSE	FALSE	TRUE	FALSE	FALSE
FALSE	Null	TRUE	FALSE	Null
Null	TRUE	Null	Null	TRUE
Null	FALSE	Null	FALSE	Null
Null	Null	Null	Null	Null

Тип данных: целые

Тип	Диапазон	Память (байт)	Описание
`TINYINT[(M)]`	-127..128 или 0..255	1	Очень маленькие целые числа
`BIT`		1	Синоним TINYINT
`BOOL`		1	Синоним TINYINT
`SMALLINT[(M)]`	-32768..32767 или 0..65535	2	Маленькие целые числа
`MEDIUMINT[(M)]`	-8388608..8388607 или 0..16777215	3	Целые числа средней величины
`INT[(M)]` `ITEGER[(M)]`	$-2^{31}..2^{31}-1$ или $0..2^{32}-1$	4	Обычное целые числа
`BIGINT[(M)]`	$2^{63}..2^{63}$ или $0..2^{64}-1$	8	Большие целые числа

Операции с целыми числами

Обычные операции с целыми, как в любом языке программирования

```
mysql>
mysql> select * from testtable;
+------+------+
| A    | B    |
+------+------+
| 2    | 2    |
| 3    | 5    |
+------+------+
2 rows in set (0.00 sec)

mysql> select A, B, (A + B), (A - B), (A * B), (A / B), (A DIV B), (A % B) from testtable;
+------+------+---------+---------+---------+---------+----------+---------+
| A    | B    | (A + B) | (A - B) | (A * B) | (A / B) | (A DIV B) | (A % B) |
+------+------+---------+---------+---------+---------+----------+---------+
| 2    | 2    |       4 |       0 |       4 |       1 |        1 |       0 |
| 3    | 5    |       8 |      -2 |      15 |     0.6 |        0 |       3 |
+------+------+---------+---------+---------+---------+----------+---------+
2 rows in set (0.00 sec)
```

Тип данных: вещественные (1/2)

Тип	Диапазон	Память (байт)	Описание
FLOAT(точность)	Зависит от точности	Различна	Числа с плавающей точкой одинарной или двойной точности
FLOAT [(M,D)]	±1.175494351E-38 ±3.402823466E+38	4	Эквивалентно FLOAT, но с указанной шириной отображения и количеством десятичных разрядов
DOUBLE [(M,D)]	±1.797693134 8E+308 ±2.2250738585E-308	8	Эквивалентно FLOAT, но с указанной шириной отображения и количеством десятичных разрядов
DOUBLE PRECISION[(M,D)]	±1.797693134 8E+308 ±2.2250738585E-308	8	Синоним DOUBLE[(M,D)]
REAL [(M,D)]	±1.797693134 8E+308 ±2.2250738585E-308	8	Синоним DOUBLE[(M,D)]

Тип данных: вещественные (2/2)

DECIMAL [(M,D)]	**Различный**	**M+2**	**Число с плавающей точкой, хранимое как char. Диапазон зависит от M.**
NUMERIC [(M,D)]	**Различный**	**M+2**	**Синоним DECIMAL[(M,D)]**
DEC [(M,D)]	**Различный**	**M+2**	**Синоним DECIMAL[(M,D)]**
FIXED [(M,D)]	**Различный**	**M+2**	**Синоним DECIMAL[(M,D)]**

Функции для работы с числовыми данными (1/2)

- **ROUND**: округляет число. В качестве первого параметра передается число. Второй параметр указывает на длину. Если длина представляет положительное число, то оно указывает, до какой цифры после запятой идет округление. Если длина представляет отрицательное число, то оно указывает, до какой цифры с конца числа до запятой идет округление

```
SELECT ROUND(1342.345, 2),      -- 1342.35
(SELECT ROUND(1342.345, -2));   -- 1300;
```

- **TRUNCATE**: оставляет в дробной части определенное количество символов

```
SELECT TRUNCATE(1342.345, 2);       -- 1342.34
```

- **ABS**: возвращает абсолютное значение числа.

```
SELECT ABS(-123)    -- 123
```

- **CEILING**: возвращает наименьшее целое число, которое больше или равно текущему значению.

```
SELECT CEILING(-123.45),        -- -123
(SELECT CEILING(123.45));       -- 124
```

- **FLOOR**: возвращает наибольшее целое число, которое меньше или равно текущему значению.

```
SELECT FLOOR(-123.45),      -- -124
(SELECT FLOOR(123.45));     -- 123
```

Функции для работы с числовыми данными (2/2)

- **POWER**: возводит число в определенную степень.

```
SELECT POWER(5, 2),     -- 25
(SELECT POWER(5, 3));       -- 125
```

- **SQRT**: получает квадратный корень числа.

```
SELECT SQRT(225);       -- 15
```

- **SIGN**: возвращает -1, если число меньше 0, и возвращает 1, если число больше 0. Если число равно 0, то возвращает 0.

```
SELECT SIGN(-5),        -- -1
(SELECT SIGN(7));       -- 1
```

- **RAND**: генерирует случайное число с плавающей точкой в диапазоне от 0 до 1.

```
SELECT RAND();      -- 0.707365088352935
SELECT RAND();      -- 0.173808327956812
```

Тип данных: строки

```
CHAR(20)     /* строка фиксированной длины */
VARCHAR(20)  /* строка переменной длины */
```

Тип	Максимальная длина (в символах)	Описание
TINYBLOB	2^8-1 (255)	Маленькое поле BLOB
TINYTEXT	2^8-1 (255)	Маленькое поле TEXT
BLOB	2^{16}-1 (65 535)	Нормальное поле BLOB
TEXT	2^{16}-1 (65 535)	Нормальное поле TEXT
MEDIUMBLOB	2^{24}-1 (16 777 215)	Среднее поле BLOB
MEDIUMTEXT	2^{24}-1 (16 777 215)	Среднее поле TEXT
LONGBLOB	2^{32}-1 (4 294 967 295)	Большое поле BLOB
LONGTEXT	2^{32}-1 (4 294 967 295)	Большое поле TEXT

ENUM		Перечисление.
SET		Набор

Некоторые функции для работы со строками (1/3)

- **CONCAT**: объединяет строки. В качестве параметра принимает от 2-х и более строк, которые надо соединить:

```
SELECT CONCAT('Tom', ' ', 'Smith')  -- Tom Smith
```

При этом в функцию можно передавать не только непосредственно строки, но и числа, даты - они будут преобразовываться в строки и также объединяться.

- **CONCAT_WS**: также объединяет строки, но в качестве первого параметра принимает разделитель, который будет соединять строки:

```
SELECT CONCAT_WS(' ', 'Tom', 'Smith', 'Age:', 34)  -- Tom Smith Age: 34
```

- **LENGTH**: возвращает количество символов в строке. В качестве параметра в функцию передается строка, для которой надо найти длину:

```
SELECT LENGTH('Tom Smith')  -- 9
```

- **LTRIM**: удаляет начальные пробелы из строки. В качестве параметра принимает строку:

```
SELECT LTRIM('  Apple')
```

- **RTRIM**: удаляет конечные пробелы из строки. В качестве параметра принимает строку:

```
SELECT RTRIM(' Apple    ')
```

Некоторые функции для работы со строками (2/3)

- **TRIM**: удаляет начальные и конечные пробелы из строки. В качестве параметра принимает строку:

```
SELECT TRIM('  Tom Smith   ')
```

С помощью дополнительного оператора можно задать где имеено удалить пробелы: `BOTH` (в начале и в конце), `TRAILING` (только в конце), `LEADING` (только в начале):

```
SELECT TRIM(BOTH FROM '  Tom Smith   ')
```

- **LOCATE(find, search [, start])**: возвращает позицию первого вхождения подстроки find в строку search. Дополнительный параметр start позволяет установить позицию в строке search, с которой начинается поиск подстроки find. Если подстрока search не найдена, то возвращается 0:

```
SELECT LOCATE('om', 'Tom Smith');       -- 2
SELECT LOCATE('m', 'Tom Smith');        -- 3
SELECT LOCATE('m', 'Tom Smith', 4);     -- 6
SELECT LOCATE('mig', 'Tom Smith');      -- 0
```

- **LEFT**: вырезает с начала строки определенное количество символов. Первый параметр функции - строка, а второй - количество символов, которые надо вырезать сначала строки:

```
SELECT LEFT('Apple', 3) -- App
```

- **RIGHT**: вырезает с конца строки определенное количество символов. Первый параметр функции - строка, а второй количество символов, которые надо вырезать сначала строки:

```
SELECT RIGHT('Apple', 3)    -- ple
```

- **SUBSTRING(str, start [, length])**: вырезает из строки str подстроку, начиная с позиции start. Третий необязательный параметр передает количество вырезаемых символов:

Некоторые функции для работы со строками (3/3)

- **REPLACE(search, find, replace)**: заменяет в строке find подстроку search на подстроку replace. Первый параметр функции - строка, второй - подстрока, которую надо заменить, а третий - подстрока, на которую надо заменить:

```
SELECT REPLACE('Galaxy S8 Plus', 'S8 Plus', 'Note 8')   -- Galaxy Note 8
```

- **INSERT(str, start, length, insert)**: вставляет в строку str, заменяя length символов с позиции start подстрокой insert. Первый параметр функции - строка, второй - позиция, с которой надо заменить, третий - сколько символов с позиции start надо заменить вставляемой подстрокой, четвертый параметр - вставляемая подстрока:

```
SELECT INSERT('Galaxy S9', 8, 3, 'Note 9');   -- Galaxy Note 9
```

- **REVERSE**: переворачивает строку наоборот:

```
SELECT REVERSE('123456789') -- 987654321
```

- **LOWER**: переводит строку в нижний регистр:

```
SELECT LOWER('Apple')   -- apple
```

- **UPPER**: переводит строку в верхний регистр

```
SELECT UPPER('Apple')   -- APPLE
```

- **SPACE**: возвращает строку, которая содержит определенное количество пробелов

- **REPEATE(str, count)**: возвращает строку, которая содержит определенное количество повторов подстроки str. Количество повторов задается через параметр count.

```
SELECT REPEAT('ab', 5);   -- abababababab
```

Типы данных: дата/время

Временные типы данных MySQL

Тип	Формат по умолчанию	Допустимые значения
Date	YYYY-MM-DD	от 1000-01-01 до 9999-12-31
Datetime	YYYY-MM-DD HH:MI:SS	от 1000-01-01 00:00:00 до 9999-12-31 23:59:59
Timestamp	YYYY-MM-DD HH:MI:SS	от 1970-01-01 00:00:00 до 2037-12-31 23:59:59
Year	YYYY	от 1901 до 2155
Time	HHH:MI:SS	от -838:59:59 до 838:59:59

Некоторые функции для работы с дата/время

Получение даты и времени

- Функции **NOW()**, **SYSDATE()**, **CURRENT_TIMESTAMP()** возвращают текущую локальную дату и время на основе системных часов в виде объекта datetime. Все три функции возвращают одинаковый результат

```
1 SELECT NOW();                -- 2018-05-25 21:34:55
2 SELECT SYSDATE();            -- 2018-05-25 21:34:55
3 SELECT CURRENT_TIMESTAMP(); -- 2018-05-25 21:32:55
```

- Функции **CURDATE** и **CURRENT_DATE** возвращают текущую локальную дату в виде объекта date:

```
1 SELECT CURRENT_DATE();       -- 2018-05-25
2 SELECT CURDATE();            -- 2018-05-25
```

- Функции **CURTIME** и **CURRENT_TIME** возвращают текущее время в виде объекта time:

```
1 SELECT CURRENT_TIME();   -- 20:47:45
2 SELECT CURTIME();        -- 20:47:45
```

Некоторые функции для работы с дата/время

Парсинг даты и времени

- **DAYOFMONTH(date)** возвращает день месяца в виде числового значения
- **DAYOFWEEK(date)** возвращает день недели в виде числового значения
- **DAYOFYEAR(date)** возвращает номер дня в году
- **MONTH(date)** возвращает месяц даты
- **YEAR(date)** возвращает год из даты
- **QUARTER(date)** возвращает номер квартала года
- **WEEK(date [, first])** возвращает номер недели года. Необязательный параметр позволяет задать стартовый день недели. Если этот параметр равен 1, то первым днем считается понедельник, иначе воскресенье
- **LAST_DAY(date)** возвращает последний день месяца в виде даты
- **DAYNAME(date)** возвращает название дня недели
- **MONTHNAME(date)** возвращает название текущего месяца
- **HOUR(time)** возвращает час времени
- **MINUTE(time)** возвращает минуту времени
- **SECOND(time)** возвращает секунду времени

Примеры функций:

Функция	Результат
DAYOFMONTH('2018-05-25')	25

Некоторые функции для работы с дата/время

Функции для манипуляции с датами

Ряд функций позволяют производить операции сложения и вычитания с датами и временем:

- **DATE_ADD(date, INTERVAL expression unit)** возвращает объект DATE или DATETIME, который является результатом сложения даты date с определенным временным интервалом. Интервал задается с помощью выражения `INTERVAL expression unit`, где `INTERVAL` предоставляет ключевое слово, `expression` - количество добавляемых к дате единиц, а `unit` - тип единиц (часы, дни и т.д.) Параметр `unit` может иметь те же значения, что и в функции EXTRACT, то есть DAY, HOUR и т.д.
- **DATE_SUB(date, INTERVAL expression unit)** возвращает объект DATE или DATETIME, который является результатом вычитания из даты date определенного временного интервала
- **DATEDIFF(date1, date2)** возвращает разницу в днях между датами date1 и date2
- **TO_DAYS(date)** возвращает количество дней с 0-го года
- **TIME_TO_SEC(time)** возвращает количество секунд, прошедших с момента полуночи

Примеры применения:

Вызов	Результат
DATE_ADD('2018-05-25', INTERVAL 1 DAY)	2018-05-26
DATE_ADD('2018-05-25', INTERVAL 3 MONTH)	2018-08-25

3. Символьные наборы и порядки сопоставления.

Символьный набор (character set)– совокупность изображений символов и соответствующих этим изображениям числовых кодов

Порядок сопоставления (collation) - совокупность правил сравнения символов в наборе, т.е. правил их упорядочивания.

КАКИЕ СИМВОЛЬНЫЕ НАБОРЫ ПОДДЕРЖИВАЕТ СЕРВЕР MYSQL ?

```
mysql> SHOW CHARACTER SET;
+----------+---------------------------------+---------------------+
| Charset  | Description                     | Default collation   |
+----------+---------------------------------+---------------------+
| big5     | Big5 Traditional Chinese        | big5_chinese_ci     |
| dec8     | DEC West European               | dec8_swedish_ci     |
| cp850    | DOS West European               | cp850_general_ci    |
| hp8      | HP West European                | hp8_english_ci      |
| koi8r    | KOI8-R Relcom Russian           | koi8r_general_ci    |
| latin1   | ISO 8859-1 West European        | latin1_swedish_ci   |
| latin2   | ISO 8859-2 Central European     | latin2_general_ci   |
...
```

Кириллические наборы символов

- cp1251 (Windows, кириллица):
  - cp1251_bin
  - cp1251_bulgarian_ci
  - cp1251_general_ci (по умолчанию)
  - cp1251_general_cs
  - cp1251_ukrainian_ci
- cp866 (DOS, русский):
  - cp866_bin
  - cp866_general_ci (по умолчанию)
- koi8r (KOI8-R, русский, Релком):
  - koi8r_bin
  - koi8r_general_ci (по умолчанию)
- koi8u (KOI8-U, украинский):
  - koi8u_bin
  - koi8u_general_ci (по умолчанию).

Задание символьного набора и порядка сопоставления на уровне СОЕДИНЕНИЯ

1. Какой набор символов имеет запрос, когда он покидает клиента? (переменная character_set_client)

2. В какой набор символов сервер должен транслировать запрос после его получения? (переменные character_set_connection и collation_connection).

3. В какой символьный набор должен сервер транслировать результирующий набор или сообщение об ошибке перед отправкой их клиенту? (переменная character set results)

SET NAMES *'имя__набора_символов'*

SET NAMES xxx;

эквивалентны

SET character_set_client = xxx;
SET character_set_results = xxx;
SET **character_set_connection** = xxx;

4. Некоторые конструкции языка определения данных (DDL)

Создание базы данных

CREATE DATABASE [IF NOT EXISTS] db_name [create_specification]

CREATE DATABASE SecScaner CHARACTER SET CP1251;

Команда создания таблицы

CREATE TABLE <table name>
({ <column name> <data type> [(<size>)]
[<column constraint> ...] } .,..
[<table constraint >] .,..
)

<table name>	Имя создаваемой таблицы
<column name>	Имя столбца в таблице
<data type>	Тип данных для значений столбца
<size>	Размер (зависит от <data type>)
<column constraint>	Ограничение целостности на уровне столбца
<table constraint>	Ограничение целостности на уровне таблицы

Примеры

```
CREATE TABLE Student
    ( ID int, Fio varchar(50), Gender char(1) )

CREATE TABLE Student
    (ID int PRIMARY KEY,
     Fio varchar(50) NOT NULL,
     GroupID smallint REFERENCES Group(ID),
     Gender char(1) DEFAULT ‘м’
     )
```

Ограничения целостности таблицы

CHECK(<predicate>)	Проверка условия
PRIMARY KEY	Первичный ключ
[FOREIGN KEY] REFERENCES <table name> [(<column name>)]	Внешний ключ

```
CREATE TABLE Student
      (GroupID  smallint REFERENCES Group(ID),
       ID int, -- номер в журнале группы
       Fio varchar(50),
       Gender char(1) DEFAULT ‘м’ ,
       PRIMARY KEY (GroupID, ID)
           );
```

Команда изменения структуры таблицы

```
ALTER TABLE <table name>
   ( [ADD <column>.,..] .,..
     [ALTER COLUMN <column name>
       <data type> [NULL | NOT NULL] ] .,..
     [DROP COLUMN <column>] .,..
   )
```

<table name>	Имя изменяемой таблицы
<column name>	Имя столбца в таблице
<column>	Определение столбца в таблице
<data type>	Тип данных для значений столбца

Пример

```
ALTER TABLE Student
    ADD Birthdate datetime NULL
```

Команда удаления

DROP TABLE <table name>

DROP DATABASE <db_name>

Заключение

На лекции рассмотрены следующие вопросы:

Общие сведения о СУБД MySQL, в т.ч.
- общая характеристика СУБД
- схема взаимодействия пользователя с БД

Общие сведения о языке SQL:
- основные группы команд
- типы данных и операции с ними

Наборы символов и порядки сопоставления.

Некоторые команды языка определения данных:
создание/удаление БД
создание/модификация/удаление таблиц БД