MySQL 学习记录

## 连接/断开服务器

·· mysql -h *host* -u *user* -p

Enter password: \*\*\*\*\*\*\*\*

***host***并 ***user***代表你的MySQL服务器运行的主机名和你的MySQL帐户的用户名。

如果正在运行MySQL的同一台计算机上登录，则可以省略主机

一些MySQL安装允许用户作为匿名（未命名）用户连接到在本地主机上运行的服务器。如果你的机器上是这种情况，你应该可以通过调用[**mysql**](https://dev.mysql.com/doc/refman/5.7/en/mysql.html)来连接到那个服务器，而不需要任何选项：shell> mysql

连接成功后，可以通过在提示符处键入QUIT（或\q）来断开连接mysql>：

## sql初步查询

可以在一行中输入多个语句。只要用分号结尾：mysql> SELECT VERSION(); SELECT NOW();

一个查询不需要在一行中全部提供，因此需要多行的冗长查询不成问题。 **[mysql](https://dev.mysql.com/doc/refman/5.7/en/mysql.html" \o "4.5.1 mysql  -  MySQL命令行工具)**通过查找终止分号来确定语句的结束位置，而不是通过查找输入行的结尾。（换句话说，**[mysql](https://dev.mysql.com/doc/refman/5.7/en/mysql.html" \o "4.5.1 mysql  -  MySQL命令行工具)** 接受自由格式的输入：它收集输入行，但直到看到分号才执行它们。

mysql> SELECT

-> USER()

-> ,

-> CURRENT\_DATE;

在这个例子中，请注意提示符从 mysql>对->您输入一个多行查询的第一行后。这就是[**mysql**](https://dev.mysql.com/doc/refman/5.7/en/mysql.html)如何 表示它还没有看到一个完整的声明，正在等待其余的。提示是你的朋友，因为它提供了有价值的反馈。如果你使用这个反馈，你可以随时了解[**mysql**](https://dev.mysql.com/doc/refman/5.7/en/mysql.html) 正在等待什么。

如果您决定不想执行正在输入的查询，请通过键入\c以下命令来取消它 ：

mysql> SELECT

-> USER()

-> \c

mysql>

| **提示** | **含义** |
| --- | --- |
| mysql> | 准备好新的查询 |
| -> | 等待多行查询的下一行 |
| '> | 等待下一行，等待完成以单引号（'）开始的字符串 |
| "> | 等待下一行，等待以双引号开头的字符串完成（"） |
| `> | 等待下一行，等待完成一个以反引号开头的标识符（`） |
| /\*> | 等待下一行，等待完成一个评论开始 /\* |

你认为你已经输入了一个声明，但唯一的回应是一个->提示，很可能[**mysql**](https://dev.mysql.com/doc/refman/5.7/en/mysql.html)正在等待分号。

在'>和">字符串集合（说，MySQL是在等待一个字符串完成的另一种方式）中的提示信息出现。在MySQL中，你可以编写由字符'或 "字符（例如，'hello'或"goodbye"）包围的字符串，而 **[mysql](https://dev.mysql.com/doc/refman/5.7/en/mysql.html" \o "4.5.1 mysql  -  MySQL命令行工具)**允许你输入跨越多行的字符串。当您看到'>或 ">提示符时，意味着您输入的行中包含以一个' 或"引号字符开头的字符串，但尚未输入用于终止该字符串的匹配引号。这往往表明你无意中遗漏了一个引号字符。

出现这种情况时取消查询。但是，你不能只输入\c这种情况，因为[**mysql**](https://dev.mysql.com/doc/refman/5.7/en/mysql.html)将它解释为它正在收集的字符串的一部分。相反，输入结束引号字符（所以[**mysql**](https://dev.mysql.com/doc/refman/5.7/en/mysql.html)知道你已经完成了字符串），然后键入\c：

### 创建和选择一个数据库

mysql> CREATE DATABASE menagerie;

在Unix下，数据库名称是区分大小写的（不像SQL关键字），所以你必须总是指到你的数据库 menagerie，而不是 Menagerie，MENAGERIE或一些其他变种。表名也是如此。（在Windows下，这个限制不适用，尽管你必须在给定的查询中使用相同的字母表来引用数据库和表。但是由于各种原因，推荐的最佳做法总是使用与该数据库已创建。）

//如果在尝试创建数据库时遇到错误1044（42000）：用户'micah'@'localhost'拒绝访问数据库'menagerie'，这意味着您的用户帐户没有必要的权限来执行所以

创建数据库不会选择使用; 你必须明确地做到这一点。要制作menagerie当前数据库，请使用以下语句：

mysql> USE menagerie

Database changed

或在开始登陆时使用该数据库

shell> mysql -h *host* -u *user* -p menagerie

Enter password: \*\*\*\*\*\*\*\*

### 4.创建一个表

mysql> CREATE TABLE pet (name VARCHAR(20), owner VARCHAR(20),

-> species VARCHAR(20), sex CHAR(1), birth DATE, death DATE);

一旦你创建了一个表，[SHOW TABLES](https://dev.mysql.com/doc/refman/5.7/en/show-tables.html" \o "13.7.5.37 SHOW TABLES语法)应该产生一些输出：

mysql> SHOW TABLES;

+---------------------+

| Tables in menagerie |

+---------------------+

| pet |

+---------------------+

要验证您的表是否按照您的预期创建，请使用以下[DESCRIBE](https://dev.mysql.com/doc/refman/5.7/en/describe.html" \o "13.8.1 DESCRIBE语法)语句：

mysql> DESCRIBE pet;

+---------+-------------+------+-----+---------+-------+

| Field | Type | Null | Key | Default | Extra |

+---------+-------------+------+-----+---------+-------+

| name | varchar(20) | YES | | NULL | |

| owner | varchar(20) | YES | | NULL | |

| species | varchar(20) | YES | | NULL | |

| sex | char(1) | YES | | NULL | |

| birth | date | YES | | NULL | |

| death | date | YES | | NULL | |

+---------+-------------+------+-----+---------+-------+

[DESCRIBE](https://dev.mysql.com/doc/refman/5.7/en/describe.html)例如，如果您忘记了表格中列的名称或其类型， 您可以随时使用。

## 向表中添加数据

1. 从本地问价添加数据

本地创建txt文件，文件内容格式：

Whistler Gwen bird \N 1997-12-07 \N //中间空格是tab键单个制表符，mysql可以自动识别单个制表符和换行符，也可以自己设置中止符

LINES TERMINATED BY '\r\n'; //将\r\n设置为中止符

这些足以让语句pet.txt正确地读取文件。

1. INSERT 添加

mysql> INSERT INTO pet

-> VALUES ('Puffball','Diane','hamster','f','1999-03-30',NULL);

## 5.从表中检索数据

该[SELECT](https://dev.mysql.com/doc/refman/5.7/en/select.html" \o "13.2.9 SELECT语法)声明用于从表格中提取信息。声明的一般形式是：

SELECT *what\_to\_select*

FROM *which\_table*

WHERE *conditions\_to\_satisfy*;

***what\_to\_select***表明你想看到什么。这可能是列的列表，或 \*指示“ 的所有列。“ ***which\_table***表示您要从中检索数据的表格。该WHERE 条款是可选的。如果存在，则***conditions\_to\_satisfy***指定行必须满足以符合检索条件的一个或多个条件。

(1) 检索所有数据

Select \* from pet;

(2) 检索特定的行

mysql> SELECT \* FROM pet WHERE name = 'Bowser';

mysql> SELECT \* FROM pet WHERE birth >= '1998-1-1';

mysql> SELECT \* FROM pet WHERE species = 'dog' AND sex = 'f';

mysql> SELECT \* FROM pet WHERE species = 'snake' OR species = 'bird';

mysql> SELECT \* FROM pet WHERE (species = 'cat' AND sex = 'm')

-> OR (species = 'dog' AND sex = 'f'); //AND 优先级大于 OR

(3) 检索特定的列

mysql> SELECT name, birth FROM pet;

查询只是owner从每条记录中检索 列，其中一些不止一次出现。为了最小化输出，通过添加关键字来检索每个唯一的输出记录 DISTINCT：

mysql> SELECT DISTINCT owner FROM pet; //消除相同行

mysql> SELECT name, species, birth FROM pet

-> WHERE species = 'dog' OR species = 'cat';

(4) 排序行（ORDER BY） 默认排序为升序排序 DESC为降序排序

mysql> SELECT name, birth FROM pet ORDER BY birth;

在字符类型的列上，像所有其他比较操作一样，排序通常以不区分大小写的方式执行。这意味着对于除了大小写之外相同的列，订单是未定义的。您可以通过使用强制的列区分大小写的排序[BINARY](https://dev.mysql.com/doc/refman/5.7/en/cast-functions.html#operator_binary)，像这样： 。 ORDER BY BINARY ***col\_name***

mysql> SELECT name, birth FROM pet ORDER BY birth DESC;//降序排序 DESC关键字仅适用于紧挨这他的列明

（5）日期计算（TIMESTAMPDIFF）

mysql> SELECT name, birth, CURDATE(),

-> TIMESTAMPDIFF(YEAR,birth,CURDATE()) AS age

-> FROM pet ORDER BY name;

提取birth列的月份部分 ：

mysql> SELECT name, birth, MONTH(birth) FROM pet;

mysql> SELECT name, birth FROM pet WHERE MONTH(birth) = 5;

DATE\_ADD(date,INTERVAL expr type) // DATE\_ADD() 函数向日期添加指定的时间间隔。

mysql> SELECT name, birth FROM pet

-> WHERE MONTH(birth) = MONTH(DATE\_ADD(CURDATE(),INTERVAL 1 MONTH));//在当前月份基础上添加一个月

MOD(N,M) 该函数返回N除以M后的余数

mysql> SELECT name, birth FROM pet

-> WHERE MONTH(birth) = MOD(MONTH(CURDATE()), 12) + 1;

Eg：

怎样才能找到下周五的日期：

SELECT DATE\_ADD（CURRENT\_DATE，INTERVAL mod（6-DAYOFWEEK（current\_date）+7,7）DAY）;

（6）使用NULL值

使用 IS NULL 和 IS NOT NULL 进行空值测试

mysql> SELECT 1 IS NULL, 1 IS NOT NULL;

+-----------+---------------+

| 1 IS NULL | 1 IS NOT NULL |

+-----------+---------------+

| 0 | 1 |

+-----------+---------------+

（7）模式匹配

MySQL提供了标准的SQL模式匹配，以及基于扩展正则表达式的模式匹配形式，类似于Unix工具（如**vi**，**grep**和 **sed）**所使用的正则表达式 。

使用\_ 匹配任何单个字符并%匹配任意数量的字符（包括零字符），使用[LIKE](https://dev.mysql.com/doc/refman/5.7/en/string-comparison-functions.html#operator_like)或 [NOT LIKE](https://dev.mysql.com/doc/refman/5.7/en/string-comparison-functions.html#operator_not-like)比较运算符。

为了找到名字开头b：mysql> SELECT \* FROM pet WHERE name LIKE 'b%';

找到以下列结尾的名字fy：mysql> SELECT \* FROM pet WHERE name LIKE '%fy';

要找到名称包含w：mysql> SELECT \* FROM pet WHERE name LIKE '%w%';

要查找包含正好五个字符的名称，请使用\_模式字符的五个实例：

mysql> SELECT \* FROM pet WHERE name LIKE '\_\_\_\_\_';

MySQL提供的其他类型的模式匹配使用扩展正则表达式。当你测试这种模式的匹配时，使用 [REGEXP](https://dev.mysql.com/doc/refman/5.7/en/regexp.html#operator_regexp)和[NOT REGEXP](https://dev.mysql.com/doc/refman/5.7/en/regexp.html" \l "operator_not-regexp)运算符（或 [RLIKE](https://dev.mysql.com/doc/refman/5.7/en/regexp.html#operator_regexp)和 [NOT RLIKE](https://dev.mysql.com/doc/refman/5.7/en/regexp.html#operator_not-regexp)）

* . 匹配任何单个字符。
* 一个字符类[...]匹配括号内的任何字符。例如， [abc]匹配a， b或c。要命名一系列字符，请使用短划线。[a-z] 匹配任何字母，而[0-9] 匹配任何数字。
* \*匹配之前的事物的零个或多个实例。例如，x\* 匹配任意数量的x字符， [0-9]\*匹配任意数量的数字，并.\*匹配任何数量的任何内容。
* 甲[REGEXP](https://dev.mysql.com/doc/refman/5.7/en/regexp.html#operator_regexp)如果该图案在被测试的值相匹配的任何位置的图案匹配成功。（这与[LIKE](https://dev.mysql.com/doc/refman/5.7/en/string-comparison-functions.html" \l "operator_like)模式匹配不同 ，只有模式匹配整个值才能成功。）
* 要锚定一个模式，以便它必须匹配被测试值^的开始或$结束，请在模式的开始或结束处使用。

要查找以名字开头的名字b，请使用 ^匹配名称的开头：

mysql> SELECT \* FROM pet WHERE name REGEXP '^b';

查询仅b在名称的开头匹配小写字母：

mysql> SELECT \* FROM pet WHERE name REGEXP BINARY '^b'; //binary使mysql区分大小写

找到名字结尾的名字fy， $用来匹配名字的结尾：

mysql> SELECT \* FROM pet WHERE name REGEXP 'fy$';

要查找包含名称w：

mysql> SELECT \* FROM pet WHERE name REGEXP 'w';

要查找包含恰好五个字符的名称，请使用 ^和$匹配名称的开始和结尾以及.两者之间的五个实例 ：

mysql> SELECT \* FROM pet WHERE name REGEXP '^.....$'; 或

mysql> SELECT \* FROM pet WHERE name REGEXP '^.{5}$';

（8）计数行

计算您拥有的动物总数与“ 表中有多少行 pet？

mysql> SELECT COUNT(\*) FROM pet;

找出每个拥有者有多少宠物:

mysql> SELECT owner, COUNT(\*) FROM pet GROUP BY owner;

mysql> SELECT species, sex, COUNT(\*) FROM pet

-> WHERE species = 'dog' OR species = 'cat'

-> GROUP BY species, sex;

+---------+------+----------+

| species | sex | COUNT(\*) |

+---------+------+----------+

| cat | f | 1 |

| cat | m | 1 |

| dog | f | 1 |

| dog | m | 2 |

+---------+------+----------+

（9）使用多个表

mysql> SELECT pet.name,

-> TIMESTAMPDIFF(YEAR,birth,date) AS age,

-> remark

-> FROM pet INNER JOIN event

-> ON pet.name = event.name

-> WHERE event.type = 'litter';

因为name列出现在两个表中，所以在引用列时必须具体说明哪个表是您的意思。这是通过将表名添加到列名来完成的

在pet表中选择种类相同的而且没死亡的宠物交配：

mysql> SELECT p1.name，p1.sex，p2.name，p2.sex，p1.species   
- > FROM pet AS p1，pet AS p2   
- > WHERE p1.species = p2.species AND p1.sex =“f” AND p2.sex =“m”   
- > AND p1.death IS NULL and p2.death IS NULL;