# MySQL快速入门

## 5、在批处理模式下使用mysql

在前面的部分中，以交互方式使用mysql输入语句并查看结果。也可以在批处理模式下运行mysql。为此，将要运行的语句放在一个文件中，然后告诉 mysql从文件中读取其输入：

shell> mysql < *batch-file*

如果需要在命令行中指定连接参数，命令可能如下所示：

shell> mysql -h *host* -u *user* -p < *batch-file*

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

当以这种方式使用**mysql时**，你将要创建一个脚本文件，然后执行该脚本。

如果你希望即使其中的某些语句产生错误脚本也继续执行，则应使用 [--force](https://dev.mysql.com/doc/refman/5.7/en/mysql-command-options.html" \l "option_mysql_force)命令行选项。

为什么要使用脚本？以下是几个原因：

* 反复（例如每天或每周）运行查询，使其成为一个脚本，可以在每次执行它时避免重新输入。
* 可以通过复制和编辑脚本文件来生成与现有查询类似的新查询。
* 在开发查询时，批处理模式也很有用，特别是对于多行语句或多语句序列。如果你犯了一个错误，你不需要重新输入一切。只需编辑你的脚本来纠正错误，然后告诉**mysql**再次执行它。
* 如果有一个产生大量输出的查询，你可以通过如下方式处理，而不是看着它从屏幕顶部滚动：

shell> mysql < *batch-file* | more

* 可以将输出捕获到文件中并进一步处理：

shell> mysql < *batch-file* > mysql.out

* 可以将脚本分发给其他人，以便他们也可以运行语句。
* 某些情况不允许交互式使用，例如，当你从**cron**作业运行查询时。在这种情况下，必须使用批处理模式。

当以批处理模式运行[mysql](https://dev.mysql.com/doc/refman/5.7/en/mysql.html" \o "4.5.1 mysql  -  MySQL命令行工具)时，默认的输出格式与交互式不同，例如，SELECT DISTINCT species FROM pet在[mysql](https://dev.mysql.com/doc/refman/5.7/en/mysql.html" \o "4.5.1 mysql  -  MySQL命令行工具)以交互方式运行时，输出如下所示 ：

+---------+

| species |

+---------+

| bird |

| cat |

| dog |

| hamster |

| snake |

+---------+

在批处理模式下，输出如下所示：

species

bird

cat

dog

hamster

snake

如果要以批处理方式获取交互式的输出格式，请使用**mysql -t**。要回传输出执行的语句，请使用**mysql -v**。

还可以在mysql启动后，在mysql提示符下，使用 source 命令或者**\.** 来执行脚本。

mysql> source *filename*;

mysql> \. *filename*

## 6、常见的查询示例

本节列出一些常用的查询示例。基础数据如下：

一张商品交易表，即：各种经销商（dealer）对文章的交易价格的表，每一篇文章可以被多个经销商销售，同一篇文章不同的经销商可以有不同的价格。因此该表的主键为（article, dealer）。

CREATE TABLE shop (

article INT(4) UNSIGNED ZEROFILL DEFAULT '0000' NOT NULL,

dealer CHAR(20) DEFAULT '' NOT NULL,

price DOUBLE(16,2) DEFAULT '0.00' NOT NULL,

PRIMARY KEY(article, dealer));

INSERT INTO shop VALUES

(1,'A',3.45),(1,'B',3.99),(2,'A',10.99),(3,'B',1.45), (3,'C',1.69),(3,'D',1.25),(4,'D',19.95);

SELECT \* FROM shop;

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

| article | dealer | price |

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

| 0001 | A | 3.45 |

| 0001 | B | 3.99 |

| 0002 | A | 10.99 |

| 0003 | B | 1.45 |

| 0003 | C | 1.69 |

| 0003 | D | 1.25 |

| 0004 | D | 19.95 |

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

**1）最大值**

SELECT MAX(article) AS article FROM shop;

+---------+

| article |

+---------+

| 4 |

+---------+

**2）查询最大列的行**

任务：找出价格最贵的文章对应的信息：

即：该表中哪一行数据的price最大，因此先要求出该表中price的最大值，然后以他为条件查询：

SELECT article, dealer, price FROM shop WHERE price=(SELECT MAX(price) FROM shop);

其他方案：

SELECT s1.article, s1.dealer, s1.price

FROM shop s1 LEFT JOIN shop s2 ON s1.price < s2.price

WHERE s2.article IS NULL;

SELECT article, dealer, price FROM shop ORDER BY price DESC LIMIT 1;

**注意：**如果有几个最贵的文章，每个价格为19.95，LIMIT解决方案将只显示其中之一。即：这个不通用。

**3）每个分组中的最大值**

任务：找到每篇文章最高的价格。

先按照文章分组，再求每组的最大值：

SELECT article, MAX(price) AS price FROM shop GROUP BY article;

**4）找出每组中包含特定列最大值的行**

任务：对于每篇文章，找到价格最昂贵的经销商。

即：以文章分组，找出每组中价格最大的行。可以如下完成：

SELECT article, dealer, price

FROM shop s1

WHERE price=(SELECT MAX(s2.price)

FROM shop s2

WHERE s1.article = s2.article);

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

| article | dealer | price |

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

| 0001 | B | 3.99 |

| 0002 | A | 10.99 |

| 0003 | C | 1.69 |

| 0004 | D | 19.95 |

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

即：对于每一行，处理where条件，首先计算出对于这一行数据中这一个文章ID的价格的最大值。然后选择出对应的列。

上述示例使用了相关子查询，MySQL可能不支持，也可以如下完成：

思路：首先找出每篇文章最大的价格对应的行：

SELECT article, MAX(price) AS price FROM shop GROUP BY article

然后与自己链接，目的是为了查询到其他的列数据。但是链接条件是文章ID和价格都相等。

SELECT s1.article, dealer, s1.price

FROM shop s1

JOIN

(

SELECT article, MAX(price) AS price

FROM shop

GROUP BY article

) AS s2

ON s1.article = s2.article AND s1.price = s2.price;

或者：

SELECT s1.article, s1.dealer, s1.price

FROM shop s1

LEFT JOIN shop s2 ON s1.article = s2.article AND s1.price < s2.price

WHERE s2.article IS NULL;

**5）使用用户定义的变量**

您可以使用MySQL用户变量记住结果，而不必将其存储在客户端中的临时变量中。

例如，要找到最高和最低价格的文章，您可以这样做：

mysql> SELECT @min\_price:=MIN(price),@max\_price:=MAX(price) FROM shop;

mysql> SELECT \* FROM shop WHERE price=@min\_price OR price=@max\_price;

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

| article | dealer | price |

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

| 0003 | D | 1.25 |

| 0004 | D | 19.95 |

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

**6）使用外键**

在MySQL中，InnoDB表支持检查外键约束。

外键约束涉及到两个表，但是，在定义时，对于除InnoDB之外的存储引擎而言，外键约束没有任何实际意义，仅仅作为备注或者注释。MySQL本身不会对外键进行任何检查，也不会进行任何操作，即：对于REFERENCES ***tbl\_name***(***col\_name***)，即使你在定义时写了ON DELETE或者ON UPDATE，但是却不会有任何实际行为，即对外键无感知。后边会详细学习，这里只是先说明一下，要使用外键功能还是要使用InnoDB。

**7）在两个字段上进行搜索**

使用单个字段的OR很容易优化，但是对于两个不同字段的OR就不好优化：

SELECT field1\_index, field2\_index FROM test\_table

WHERE field1\_index = '1' OR field2\_index = '1'

这就不好优化，其实你可以这么做：每个SELECT搜索只有一个字段，可以优化查询：

SELECT field1\_index, field2\_index

FROM test\_table WHERE field1\_index = '1'

UNION

SELECT field1\_index, field2\_index

FROM test\_table WHERE field2\_index = '1';

注意：UNION会去除结果集中重复的行，除非是有UNION ALL。

**8）计算每天的访问量**

计算一个Web页面，在各个月之中有几天被访问到：

CREATE TABLE t1 (year YEAR(4), month INT(2) UNSIGNED ZEROFILL,

day INT(2) UNSIGNED ZEROFILL);

INSERT INTO t1

VALUES(2000,1,1),(2000,1,20),(2000,1,30),(2000,2,2), (2000,2,23),(2000,2,23);

示例表中包含用户访问页面的年、月、日。要确定每月有几天被访问到，请如下查询：

SELECT year,month,BIT\_COUNT(BIT\_OR(1<<day)) AS days FROM t1 GROUP BY year,month;

将返回如下信息：

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

| year | month | days |

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

| 2000 | 01 | 3 |

| 2000 | 02 | 2 |

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

该查询可以计算每年/月组合中表格中显示的不同天数，自动删除重复的条目。

**9）使用AUTO\_INCREMENT**

AUTO\_INCREMENT属性可用于为新行生成唯一标识：

CREATE TABLE animals (

id MEDIUMINT NOT NULL AUTO\_INCREMENT,

name CHAR(30) NOT NULL,

PRIMARY KEY (id)

);

INSERT INTO animals (name) VALUES

('dog'),('cat'),('penguin'),

('lax'),('whale'),('ostrich');

SELECT \* FROM animals;

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

| id | name |

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

| 1 | dog |

| 2 | cat |

| 3 | penguin |

| 4 | lax |

| 5 | whale |

| 6 | ostrich |

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

可以看到默认从1开始，如果没有启用SQL模式NO\_AUTO\_VALUE\_ON\_ZER，那么可以明确地将0分配给列以生成序列号。可以使用LAST\_INSERT\_ID（）SQL函数或mysql\_insert\_id（）C API函数检索最近自动生成的AUTO\_INCREMENT值。这些功能是特定于连接的，所以它们的返回值不受另外一个也执行插入的连接的影响。

对于自增列，要合理的选用数据类型以保证可以容纳你的行。如果超过了范围，那么下一次自动生成序号时将失败。为了不让序号从1开始，你可以在CREATE TABLE或者ALTER TABLE中指定：

mysql> ALTER TABLE tbl AUTO\_INCREMENT = 100;