第四章：检索数据

1、检索单个列

select prod\_name from products;(返回所有的行，没有排序)

（注意：

1. 多条SQL语句必须以分号结束；单条可用可不用；使用MYSQL命令行，必须加上分号结束；
2. SQL语句和大小写无关，但应保持大小写的惯例。
3. 语句中所有空格被忽略，可分多行。

）

2、检索多个列

select prod\_id, prod\_name, prod\_price from products;

3、检索所有列，使用符\*

select \* from products;(优点：检索出未知列名的列)

4、检索不同行

select vend\_id from products;

select distinct vend\_id from products;

(注意：返回不同的行,必须放在列名前面；不能部分使用；应用于所有列)

5、限制结果，使用limit 子句

Select prod\_name from products limit 5;(返回第一行开始的不多于五行)

Select prod\_name from products limit 5,5;(返回从行5开始的5 行)

（注意：行0为第一行，limit 1,1 返回第二行而不是第一行。

行数不够，返回全部；limit 4 offset 3）

6、使用完全限定的表名

Select products.prod\_name from products;

Select products.prod\_name from crashcourse.products;

第五章：排序检索数据

1、单列排序

Select prod\_name from products order by prod\_name;(字母顺序排序)

2、按多个列排序

Select prod\_id, prod\_price, prod\_name from products

Order by prod\_price, prod\_name;(主次关键字排序)

3、指定排序方向（作用于多列需要分别指定关键字）

1)Select prod\_id, prod\_price, prod\_name from products

Order by prod\_price DESC;

2)Select prod\_id, prod\_price, prod\_name from products

Order by prod\_price desc, prod\_name;(desc 只对前面的列有效)

4、最大值与最小值

Select prod\_price from products Order by prod\_price DESC LIMIT 1;

(**注意各个关键字的次序)**

第六章：过滤数据

1、使用WHERE子句(order by 子句应该放在WHERE子句之后)

Select prod\_price, prod\_name from products where prod\_price=2.50;

2、检查单个值

Select prod\_price, prod\_name from products where prod\_name=’fuses’;

Select prod\_price, prod\_name from products where prod\_price<10;

Select prod\_price, prod\_name from products where prod\_price<=10;

3、不匹配检查

Select vend\_id, prod\_name from products where vend\_id<>1003;(!=)

4、范围值检查

Select prod\_price, prod\_name from products where prod\_price between 5 and 10;(between and )

5、空值检查（NULL无值，与字段包含0，空字符串或仅仅包含空格不同）

Select prod\_price, prod\_name from products where prod\_price is null;

Select cust\_id from customers where cust\_email is null;

第七单：数据过滤

1、组合WHERE子句（多个WHERE子句,and的正确使用）

Select prod\_id, prod\_price, prod\_name

from products

where vend\_id=1003 and prod\_price<=10;

2、OR操作符

Select prod\_id, prod\_price, prod\_name

from products

where vend\_id=1003 or vend\_id=1002;

3、计算次序(比较以下两句，注意括号的使用)

Select prod\_id, prod\_price, prod\_name

from products

where vend\_id=1002 or vend\_id=1003 and prod\_price>=10;

(where vend\_id=1002 or( vend\_id=1003 and prod\_price>=10);)

Select prod\_id, prod\_price, prod\_name

from products

where (vend\_id=1002 or vend\_id=1003 )and prod\_price>=10;

4、IN操作符（用来指定条件范围，每个条件分别匹配，由逗号分开）

Select prod\_id, prod\_price, prod\_name

from products

where vend\_id IN (1002,1003) ORDER BY prod\_name;

(等价于where vend\_id＝1002 OR VEND\_ID=1003 ORDER BY prod\_name;)

5、NOT操作符（否定它之后的任何条件）

Select prod\_id, prod\_price, prod\_name

from products

where vend\_id NOT IN (1002,1003) ORDER BY prod\_name;(非1002，1003)

（MYSQL支持使用NOT对IN、BETWEEN和EXISTS子句取反）

第八章：用通配符进行过滤

1、LIKE操作符

（用来品配值的一部分的特殊字符,注意尾空格的匹配与NULL的匹配，P63）

1）百分号（%）通配符（表示任何字符出现任意次数）

Select prod\_id,prod\_name

from products

where prod\_name LIKE ‘jet%’;(注意：搜索区分大小写)

Select prod\_id, prod\_price, prod\_name

from products

where prod\_name LIKE ‘%anvil%’;

Select prod\_id, prod\_price, prod\_name

from products

where prod\_name LIKE ‘s%e’;

2）下划线（\_）通配符（匹配单个字符）

Select prod\_id, prod\_name

from products

where prod\_name LIKE ‘\_ ton anvil’;

Select prod\_id, prod\_name

from products

where prod\_name LIKE ‘% ton anvil’;

(注意使用通配符的技巧，适用于应用程序中的模糊查询。P49)

第九章：用正则表达式进行搜索

正则表达式是用来匹配文本的特殊的串（字符集合），它用正则表达式语言来建立。MYSQL 仅支持多数正则表达式实现的一个很小的子集。

1、基本字符匹配（注意与LIKE的区别，见P53,不区分大小写）

Select prod\_id, prod\_name

from products

where prod\_name REGEXP ‘1000’ ORDER BY prod\_name;

(与文字正文1000匹配的一个正则表达式)

Select prod\_id, prod\_name

from products

where prod\_name REGEXP ‘.000’ ORDER BY prod\_name;

(.是正则表达式中的一个特殊字符，匹配任意一个字符)

Select prod\_id, prod\_name

from products

where prod\_name REGEXP BINARY ‘jetPack .000’ ORDER BY prod\_name;

(使用BINARY区分大小写)

2、进行OR匹配（二者取其一）

Select prod\_id, prod\_name

from products

where prod\_name REGEXP ‘1000|2000’ ORDER BY prod\_name;

(|为正则表达式的OR操作符，表示匹配其中之一，类似于OR语句，可以给出多个OR条件)

3、匹配几个字符之一

Select prod\_id, prod\_name

from products

where prod\_name REGEXP ‘[123] Ton’ ORDER BY prod\_name;

（[123]定义一组字符，只能取其中一个，[]是另一种形式的OR语句，等价于[1｜2｜3] ton）

Select prod\_id, prod\_name

from products

where prod\_name REGEXP ‘1|2|3 Ton ’ ORDER BY prod\_name;

(除非把字符｜括在一个集合中，否则意义将大不一样)

Select prod\_id, prod\_name

from products

where prod\_name REGEXP ‘[^123] Ton ’ ORDER BY prod\_name;

([^123]匹配除123这些字符外的任何东西)

4、匹配范围（集合可以用来定义要匹配的一个或多个字符）

Select prod\_id, prod\_name

from products

where prod\_name REGEXP ‘[1-5] Ton ’ ORDER BY prod\_name;

（[1-5]等价于[12345]，[a-z]）

**\*5、匹配特殊字符（正则表达式由具有特殊意义的特殊字符构成）**

Select vend\_name from vendors

Where vend\_name REGEXP ‘\\.’(‘.’匹配任意字符，需要转义字符)

6、匹配字符类（预定义的字符集）

Select vend\_name from vendors

Where vend\_name REGEXP ‘[:alnum:]’;

**\*7、匹配多个实例（匹配数量上的重复，重复元字符）**

Select prod\_id, prod\_name

from products

where prod\_name REGEXP ‘\\([0-9] sticks?\\)’ ORDER BY prod\_name;

(\\( [0-9] sticks? \\) , ?匹配它前面的任意字符的0次或1次出现)

Select prod\_id, prod\_name

from products

where prod\_name REGEXP ‘[[:digit:]]{4}’ ORDER BY prod\_name;

(匹配连在一起的4个数字)

Select prod\_id, prod\_name

from products

where prod\_name REGEXP ‘[0-9] [0-9] [0-9] [0-9]’ ORDER BY prod\_name;

**\*8、定位符（匹配特定位置的文本，使用定位元字符，^的双重用途）**

Select prod\_id, prod\_name

from products

where prod\_name REGEXP ‘^[0-9\\.]’ ORDER BY prod\_name;

(匹配以数或小数点.开始的数，^匹配串的开始，只在为串中第一个字符时)

第十章：创建计算字段

1、拼接字段（CONCAT()需要一个或多个用逗号分隔的串）

1）select concat(vend\_name,' (',vend\_country,')') as 'sss' from vendors order by vend\_name;（sss作为别名）

2）select concat(rtrim(vend\_name),' (',rtim(vend\_country),')') as 'sss' from vendors order by vend\_name;（rtrim(),ltrim(),trim()）

2、使用别名（别名是一个字段或值的替换名，AS为关键字，任何客户机应用都可以按名引用这个计算的列，就像它是一个实际的表列一样）

select concat(rtrim(vend\_name),' (',rtim(vend\_country),')') as vend\_title from vendors order by vend\_name;

（指示SQL创建一个包含指定计算的名为vend\_title的计算字段，别名有时也称为导出列。）

3、执行算术计算

Select prod\_id,quantity,item\_price from orderitems where order\_num=20005;

Select prod\_id,quantity,item\_price,quantity\*item\_price as expanded\_price from orderitems where order\_num=20005;

4、测试计算\*

第十一章：使用数据处理函数

1、文本处理函数(P69)

1)Select vend\_name,upper(vend\_name) as vend\_name\_upcase from venders order by vend\_name;

2)select cust\_name,cust\_contact from customers where cust\_contact=’Y Lie’;

select cust\_name,cust\_contact from customers where soundex(cust\_contact)=soundex(’Y Lie’);

2、日期和时间处理函数(P71)

Select cust\_id,order\_num,order\_date from orders where order\_date=’2005-09-01’;

Select cust\_id,order\_num,order\_date from orders where date(order\_date)='2005-09-01';

Select cust\_id,order\_num,order\_date from orders where date(order\_date) between '2005-09-01' and '2005-09-30';

Select cust\_id,order\_num,order\_date from orders where year(order\_date)=2005 and month(order\_date)=9;

select year(curdate()), month(curdate());

3、数值处理函数（仅处理数值数据）

select sin(pi()/2);

第十二章：汇总数据

1、聚集函数（运行在行组上，计算和返回单个值的函数）

1）AVG()函数（忽略列值为NULL的行）

select avg(prod\_price) as avg\_price from products where vend\_id=1003;

2）COUNT()函数（计数，count(\*)与count(column)）

select count(\*) as num\_cust from customers;

select count(cust\_email) as num\_cust from customers;

3）MAX()函数（返回列中的最大值，要求指定别名，可对非数值列使用）

select max(prod\_price) as max\_price from products;

4）MIN()函数（返回列中的最小值，要求指定别名，可对非数值列使用）

select min(prod\_price) as min\_price from products;

5）SUM()函数（返回列的和，要求指定别名，也可用来合计计算值）

select sum(quantity) as items\_ordered fromorderitems where order\_num=20005;

select sum(item\_price\*quantity) as total\_price from orderitems where order\_num=20005;

2、聚集不同值（distinct）

select avg(prod\_price) as avg\_price from products where vend\_id=1003;

select avg(distinct prod\_price) as avg\_price from products where vend\_id=1003;

3、组合聚集函数

select count(\*) as num\_items,

avg(prod\_price) as price\_avg ,

max(prod\_price) as price\_max

from products;

第十三章：分组数据（对每个组或子集进行聚集）

**注意：**

**1）GROUP BY 子句可以包含任意数目的列，从面可以实现嵌套；**

**2）若嵌套了分组，数据将在最后规定的分组上进行汇总，此时指定的所有的列都一起计算，所以不再能从个别的列取回数据；**

**3）子句中列出的每个列都必须是检索列或有效的表达式，但不能是聚集函数，如果在SELECT 中使用表达式，则必须在GROUP BY 子句中指定相同的表达式，不能使用别名。**

**4）除聚集计算语句外，SELECT 语句中的每个列都必须在GROUP BY子句中给出。**

**5）若分组列中具有NULL值，则将作为一个分组返回，若列中有多行NULL值，它们将分为一组。**

**6）GROUP BY 子句必须出现在WHERE子句之后，ORDE BY 子句之前。**

1、数据分组（简单分组需要加WHERE子句）

Select count(\*) as num\_prods from products where vend\_id=1003;

2、创建分组（同步自动排序并创建多个分组）

Select vend\_id,count(\*) as num\_prods from products group by vend\_id;

Select vend\_id,count(\*) as num\_prods from products group by vend\_id order by vend\_id desc;（特别指出要降序排序）

\*使用WITH ROLLUP关键字，可以得到每个分组以及每个分组汇总级别（针对每个分组）的值。（分组对应级别的总计值）

Select vend\_id,count(\*) as num\_prods from products group by vend\_id with rollup;

3、过滤分组（HAVING 子句，必须基于完整的分组而不是个别的行进行过滤）

**注意：**

**1）事实上，所有简单类型的WHERE子句都可以用HAVING子句来代替，唯一的差别是WHERE过滤行，而HAVING过滤分组。**

**2）HAVING支持所有的WHERE操作符，句法相同，而关键字不同。**

**3）WHERE在数据分组前过滤，而HAVING在数据分组后过滤**

Select cust\_id,count(\*) as orders\_sum FROM orders group by cust\_id having count(\*)>=2;

Select vend\_id,count(\*) as num\_prods from products where prod\_price>=10 group by vend\_id having count(\*)>=2;

Select vend\_id,count(\*) as num\_prods from products group by vend\_id having count(\*)>=2;

4、分组和排序（GROUP BY 与ORDER BY ）

Select order\_num,sum(quantity\*item\_price) as ordertotal from orderitems group by order\_num having sum(quantity\*item\_price)>=50;

Select order\_num,sum(quantity\*item\_price) as ordertotal from orderitems group by order\_num having sum(quantity\*item\_price)>=50 order by ordertotal;

注意：SELECT子句顺序

SELECT FROM [WHERE[ GROUP BY [HAVING][ORDER BY]] LIMIT [NUM]

Select order\_num,sum(quantity\*item\_price) as ordertotal from orderitems group by order\_num having sum(quantity\*item\_price)>=50 order by ordertotal LIMIT 3;

附：样例库（crashcourse）关系图

