“互联网 + 教育”新形态一体化系列教材

MySQL 数据库技术应用与实战

主　编　徐红伟

北京工业大学出版社

编委会

主　编

徐红伟

副主编

杜　娟　胡　菠　徐文燕

参　编

曹利萍　张　鑫　高洪云　陈　莹

前言

PREFACE

党的二十大报告指出：“加快建设网络强国、数字中国，构建新一代信息技术、人工智能、生物技术、新能源、新材料、高端装备、绿色环保等一批新的增长引擎。”随着计算机的不断发展，需要计算机处理的数据越来越多，能够高效存储和处理的数据库技术应运而生。数据库技术已经成为IT领域不可缺少的一部分。数据库技术的设计和开发已经成为各类应用和网站中主要的组成部分，其需求也相应增多。因此数据库设计和维护技术作为一项知识技能受到大家越来越多的重视。在众多数据库中，MySQL数据库作为访问数据库常用的SQL语言，具备体积小、速度快、总体拥有成本低、开放源码等特点。所以，MySQL成为一般中小型网站开发的首选数据库。

本书以MySQL数据库为基础，围绕最新的MySQL数据库技术展开深入讲解，以清晰的思路、典型的实例使读者快速入门，并逐步掌握网络编程的知识。本书注重基础理论与实用开发相结合，突出数据库设计思想与数据库操作优化方法的介绍，所选实例都具有较强的概括性和实际应用价值。

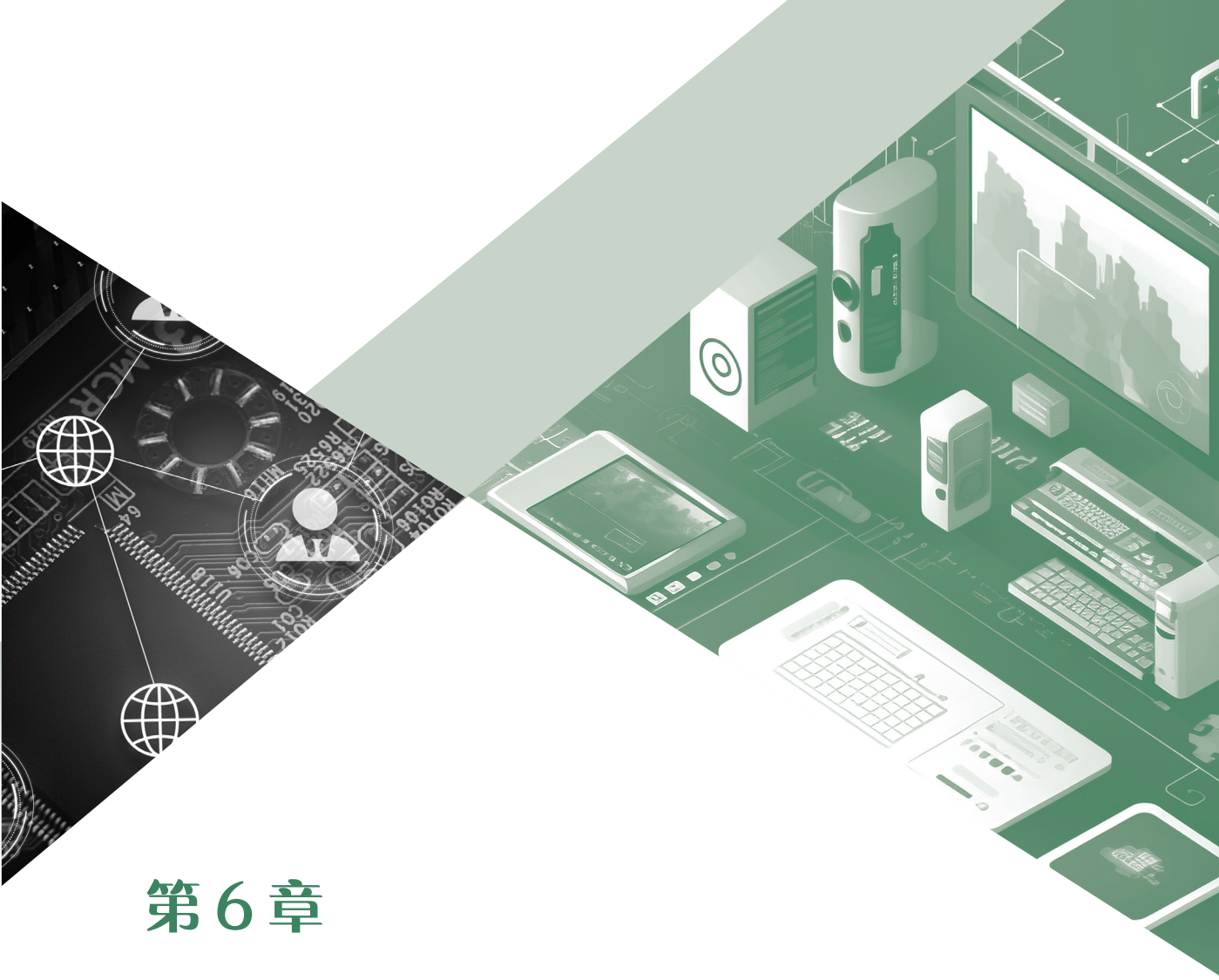
本书是编者根据多年从事数据库程序设计工作和讲授计算机专业相关课程的教学实践，在已编多部讲义和教材的基础上编写而成的；内容充实，循序渐进，选材上注重系统性、先进性和实用性；注重实践性，精选大量例题，增加了代码注释且所有例题已在MySQL数据库上调试通过，可直接引用，读者也可按照书中提示步骤自己动手完成。

由于编者水平有限，加之编写时间仓促，书中难免存在错误和疏漏之处，希望广大读者批评指正。

编　者

目录

CONTENTS



第6章

MySQL 运算符和系统函数

MySQL 提供了大量的运算符和系统函数用于实现对数据的管理。通过不同的运算符可以十分准确地对指定的数据进行操作，使用系统函数可以提高开发人员进行数据分析与统计的效率。

知识入门


MySQL运算符和系统函数

1）创建测试表

创建数据表operation，在表中新建一个字段含有一个INT类型的字段Number，效果如下所示。

```
mysql> CREATE TABLE operation (Number INT);
Query OK, 0 rows affected (1.61 sec)
```

向operation数据表中插入数字10，效果如下所示。

```
mysql> INSERT INTO operation  VALUES (10), (11);
Query OK, 2 row affected (0.23 sec)
```

查看operation数据表的具体数据，效果如下所示。

```
mysql>  SELECT * FROM operation;
+----------+
| Number   |
+----------+
|     10   |
|     11   |
+----------+
2 rows in set (0.00 sec)
```

2）进制

进制是指数字进位的方式。人们常用的是十进制，即逢十进一。计算机支持的进制包括二进制、八进制、十六进制和十进制。

二进制只包含0和1两个数字，进位规则为逢二进一。八进制包含0~7共8个数字，进位规则为逢八进一。十六进制包含0~9、A~F共16个字符，进位规则为逢十六进一。在内存中，所有的计算机数据都是以二进制进行存放的。每八位二进制表示一个字节。

3）十进制转换二进制

计算机在处理数据时会将十进制数据自动转换为二进制数据，其转换原理是通过对十进制数据进行除2取余的方式实现的。例如，将十进制数8转换为二进制的过程如下所示。

```
8/2              余0
4/2              余0
2/2              余0
1/2              余1
```

然后，将余数从下向上书写，即十进制数8的二进制数为1000。

4）二进制转换十进制

二进制数转换为十进制数的方法是将二进制数每八位为一组，依次从低位向高位（书写时

右侧为低位，左侧为高位）进行 2 的 N 次幂运算（N 包括 0~7，共 8 个数字，最低位为 0，最高位为 7）并乘以对应二进制位的值，最后相加所得。例如，将二进制数 1000 转换为十进制数，过程如下所示。

```
高位          1              0              0              0
低位
              1*2^3          0*2^2          0*2^1          0*2^0
              8       +      0       +      0       +      0
```

最终得到十进制的数字 8，因此，二进制数 1000 对应的十进制数为 8。

6.1 MySQL运算符


MySQL 运算符

MySQL 支持多种类型的运算符，实现对数据库中数据的运算。常用的运算符包括算术运算符、比较运算符、逻辑运算符和位运算符。

6.1.1 算术运算符

算术运算符主要用于数学运算，可以实现对数据的四则运算和求余运算。算术运算符如表 6.1 所示。

表 6.1　算术运算符

运算符	名称	作用	示例
+	加法运算	计算两个值或表达式的和	A + B
−	减法运算	计算两个值或表达式的差	A − B
*	乘法运算	计算两个值或表达式的乘积	A * B
/或DIV	除法运算	计算两个值或表达式的商	A / B或A DIV B
%或MOD	求模（求余）运算	计算两个值或表达式的余数	A % B或A MOD B

实例 6-1　对数据表operation中的Number字段的所有值进行算术运算。

```
mysql> SELECT Number,Number+2,Number-2,Number*2,Number/2,
Number%2 FROM operation;
+--------+----------+----------+---------+----------+----------+
| Number | Number+2 | Number-2 | Number*2| Number/2 | Number%2 |
+--------+----------+----------+---------+----------+----------+
|   10   |       12 |        8 |    20   |  5.0000  |     0    |
|   11   |       13 |        9 |    22   |  5.5000  |     1    |
+--------+----------+----------+---------+----------+----------+
2 rows in set (0.00 sec)
```

从运行结果可以看出，SELECT语句会提取数据表的Number字段进行算术运算，然后显示运算结果。由于SELECT语句是通过提取数据表中的数据进行运算，因此不会改变数据表中的源数据，查询数据表operation，效果如下所示。

```
mysql> SELECT * FROM operation;
+----------+
| Number   |
+----------+
|     10   |
|     11   |
+----------+
2 rows in set (0.00 sec)
```

从运行结果可以看出，数据表operation的源数据并没有受到影响。


比较运算符

6.1.2 比较运算符

比较运算符可以对数据进行比较。如果比较运算符的运算结果为真，返回1；比较的结果为假，返回0；其他情况返回NULL。使用SELECT查询语句和比较运算符可以快速查找到指定的数据内容。

1. 等号运算符（=）

等号运算符用于判断等号两边的数据是否相等，如果相等返回1，不相等则返回0。如果数据有一侧或全部为NULL，运算结果为NULL；如果数据为字符串，会比较每个字符的ASCII编码是否相等；如果数据为整数，会比较两个值的大小；如果两侧的数据类型不同，会转换为同一类型后进行比较。

2. 安全等于运算符（<=>）

安全等于运算符（<=>）和等号运算符（=）功能基本相同。只是安全等于运算符可以比较NULL值。如果比较的两个数据都为NULL，则结果返回1。

3. 不等于运算符（<>和!=）

不等于运算符用于判断两边的数据是否不相等。如果不相等返回1，相等返回0。如果数据有一侧或全部为NULL，运算结果为NULL。

4. 小于运算符（<）

小于运算符用于判断左边的数据是否小于右边的数据。如果小于返回1，否则返回0。如果数据有一侧或全部为NULL，运算结果为NULL。

5. 小于或等于运算符（<=）

小于或等于运算符用于判断左边的数据是否小于或等于右边的数据。如果小于或等于返回1，否则返回0。如果数据有一侧或全部为NULL，运算结果为NULL。

6. 大于运算符（>）

大于运算符用于判断左边的数据是否大于右边的数据。如果大于返回1，否则返回0。如

果数据有一侧或全部为NULL，运算结果为NULL。

7. 大于或等于运算符（>=）

大于或等于运算符用于判断左边的数据是否大于或等于右边的数据。如果大于或等于返回1，否则返回0。如果数据有一侧或全部为NULL，运算结果为NULL。

8. 空运算符（IS NULL或者ISNULL）

空运算符判断一个值是否为NULL。如果为NULL，返回1，否则返回0。

9. 非空运算符（IS NOT NULL）

非空运算符判断一个值是否不为NULL。如果不为NULL，返回1，否则返回0。

10. 最小值运算符（LEAST）

最小值运算符用于获取参数列表中的最小值。如果参数列表中有任意一个值为NULL，则结果返回NULL。

11. 最大值运算符（GREATEST）

最大值运算符用于获取参数列表中的最大值。如果参数列表中有任意一个值为NULL，则结果返回NULL。

12. BETWEEN AND运算符

BETWEEN AND运算符使用的格式通常为C BETWEEN A AND B。如果C在A和B之间时结果为1，否则结果为0。其中，C可以等于A或B。

13. IN运算符

IN运算符判断给定的值是否与IN列表中的某一个值相等。如果相等返回1，否则返回0。如果给定的值为NULL，或者IN列表中存在NULL，则结果为NULL。

14. NOT IN运算符

NOT IN运算符判断给定的值是否与IN列表中的任意一个值都不相等。如果不是IN列表中的一个值，则返回1，否则返回0。

15. LIKE运算符

LIKE运算符主要用于匹配字符串，通常用来模糊匹配。如果满足条件则返回1，否则返回0。如果给定的值或者匹配条件为NULL，则返回结果为NULL。

16. REGEXP运算符

REGEXP运算符用于匹配字符串，通常与正则表达式一起使用。如果满足条件则返回1，否则返回0。如果给定的值或者匹配条件为NULL，则结果返回NULL。

实例 6-2　显示比较运算符的运算结果。

```
mysql> SELECT 1 = 1 AS A, NULL <=> NULL AS B,'a' != NULL AS
C, 'a' < 'b' AS D, 2 <= 2 AS E, 1 > 1 AS F, 1 >= 1 AS G,
ISNULL(NULL) AS H, NULL IS NOT NULL AS I, LEAST(1,2,3) AS J,
GREATEST(1,2,3) AS K;
+---+---+----------+---+----+---+---+---+---+---+---+
```

```
| A | B | C         | D | E   | F | G | H | I| J| K|
+---+---+-----------+---+-----+---+---+---+--+--+--+
| 1 | 1 |  NULL     | 1 | 1   | 0 | 1 | 1 | 0| 1| 3|
+---+---+-----------+---+-----+---+---+---+--+--+--+
1 row in set (0.00 sec)
```

从运行结果可以看出，使用不同的比较运算符会显示不同的数据。


逻辑运算符

6.1.3 逻辑运算符

逻辑运算又称为布尔运算。布尔用数学方法研究逻辑问题，成功地建立了逻辑演算。逻辑运算的运算结果包括 1（真）、0（假）或者 NULL3 种结果。MySQL 支持 4 种逻辑运算符，如表 6.2 所示。

表 6.2　MySQL 支持的逻辑运算符

运算符	作用	示例
NOT 或 !	逻辑非运算符，运算的值为 0 时，返回 1；值为非 0 值时，返回 0；值为 NULL时，返回NULL	NOT A
AND 或 &&	逻辑与运算符，当运算的所有值均为非 0 值，且不为NULL时，返回 1，当运算的值为 0，则返回 0，否则返回NULL	A AND B A && B
OR 或 ||	逻辑或运算符，当运算的值都为非 0 值且不为NULL时返回 1，否则返回 0；当一个值为NULL且另一个值非 0 值时，返回 1，否则返回NULL；当两个值都为NULL时，返回NULL	A OR B A || B
XOR	逻辑异或运算符，当运算的值中任意一个值为NULL，返回NULL；如果两个非NULL的值都是 0 或者都不等于 0，则返回 0；如果一个值为 0，另一个值不为 0，则返回 1	A XOR B

实例 6-3　显示逻辑运算符的运算结果。

```
mysql> SELECT NOT 1,! 0,1 AND 1, 0 && 1,1 OR 1,0 || 1,1 XOR 1, 1
XOR 0;
+-------+------+----------+--------+--------+---------+--------+
------------+
| NOT 1 | ! 0  | 1 AND 1  | 0 && 1 | 1 OR 1 | 0       || 1     |
1 XOR 1     |
+-------+------+----------+--------+--------+---------+--------+
------------+
|     0 |    1 |        1 |      0 |      1 |       1 |       0|
       1    |
+-------+------+----------+--------+--------+---------+--------+
------------+
1 row in set, 3 warnings (0.00 sec)
```

从运行结果可以看出，不同逻辑运算符在处理相同数据时运算的结果不同。

6.1.4 位运算符


位运算符

位运算符主要对数据的二进制形式进行逻辑运算，从而得出相应的运算结果。位运算符包括按位与、按位或、按位异或、按位取反、按位右移和按位左移 6 种。

1. 按位与运算符（&）

将数据对应的二进制数逐位进行逻辑与运算。对应二进制位的数都为 1 时，该位返回 1，否则返回 0。

实例 6-4　将十进制数 1 和 2 按位与运算。

```
mysql> SELECT 1&2;
+-----+
| 1&2 |
+-----+
|   0 |
+-----+
1 row in set (0.00 sec)
```

其中，十进制数 1 转换为二进制为 1，十进制数 2 转换为二进制数为 10。按位与运算过程如下所示。

```
0      1
1      0
0      0
```

所以，得到二进制结果为 0，转换为十进制也为 0。

2. 按位或运算符（|）

将数据对应的二进制数逐位进行逻辑或运算。对应二进制位的数有一个或都为 1 时，该位返回 1，否则返回 0。

实例 6-5　将十进制数 1 和 2 按位或运算。

```
mysql> SELECT 1|2;
+-----+
| 1|2 |
+-----+
|   3 |
+-----+
1 row in set (0.00 sec)
```

按位或运算过程如下所示。

```
0      1
1      0
1      1
```

所以，得到二进制结果为 11，转换为十进制为 3。

3. 按位异或操作符（^）

将数据对应的二进制数逐位进行逻辑异或运算。对应的二进制位的数值不同时，该位返回1，否则返回0。

实例 6-6 将十进制数 1 和 2 按位异或运算。

```
mysql> SELECT 1^2;
+-----+
| 1^2 |
+-----+
|   3 |
+-----+
1 row in set (0.00 sec)
```

按位或运算过程如下所示。

```
0      1
1      0
1      1
```

所以，得到二进制结果为 11，转换为十进制为 3。

4. 按位取反运算符（~）

将数据的二进制数逐位进行取反操作，即将 1 变为 0，将 0 变为 1。

实例 6-7 将十进制数 1 和 2 按位异或运算。

```
mysql> SELECT ~0;
+----------------------------------+
| ~0                               |
+----------------------------------+
| 18446744073709551615             |
+----------------------------------+
1 row in set (0.00 sec)
```

由于MySQL的无法符号整数位最大值为 18446744073709551615，最小值为 0，因此将 0 全部按位取反后，得到MySQL可以表示的最大二进制整数，即 18446744073709551615。

5. 按位右移运算符（>>）

按位右移运算符左侧为要位移的数据，右侧为移动的位数。二进制数会向低位进行位移指定位数，左边高位空出的位置用 0 补齐。

实例 6-8 将十进制数 8 按位右移一位。

```
mysql> SELECT 8>>1;
+--------+
| 8>>1 |
+--------+
|    4 |
```

```
+-------+
1 row in set (0.00 sec)
```

十进制数 8 转换为二进制后为 1000，按位右移 1 位的运算过程如下所示。

```
1000
0100
```

所以，得到二进制结果为 0100，转换为十进制为 4。

6. 按位左移运算符（<<）

按位左移运算符左侧为要位移的数据，右侧为移动的位数。二进制数会向高位进行位移指定位数，右边低位空出的位置用 0 补齐。

实例 6-9　将十进制数 8 按位左移一位。

```
mysql> SELECT 8<<1;
+------+
| 8<<1 |
+------+
|   16 |
+------+
1 row in set (0.00 sec)
```

按位左移 1 位的运算过程如下所示。

```
1 0 0 0
1 0 0 0 0
```

所以，得到二进制结果为 10000，转换为十进制为 16。从左移和右移运算符的运算结果可以看出，位移的方式可以快速实现数据的次幂运算。

任务 6-1　使用加法运算符更新 author 表的年龄信息

使用加法运算符更新 author 表的年龄信息

【任务描述】

（1）为数据表 author 添加 age 字段。

（2）为 author 的 age 字段添加数据。

（3）使用加法运算符更新所有 age 数据。

【任务实施】

1. 为 author 数据表添加 age 字段

添加字段使用 ALTER TABLE 语句，效果如下所示。

```
mysql> ALTER TABLE author ADD COLUMN age INT;
Query OK, 0 rows affected (0.47 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

2. 为age字段添加数据

为age字段添加数据需要使用UPDATE更新语句，效果如下所示。

```
mysql> UPDATE author SET age = 32 WHERE id = 1;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> UPDATE author SET age=33 WHERE id=2;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> UPDATE author SET age=25 WHERE id=3;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> UPDATE author SET age=55 WHERE id=4;
Query OK, 1 row affected (0.10 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> UPDATE author SET age=27 WHERE id=5;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> UPDATE author SET age=38 WHERE id=6;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
mysql> UPDATE author SET age=40 WHERE id=7;
Query OK, 1 row affected (0.00 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

3. 查看数据表的当前数据

使用SELECT语句查看添加数据后的author表，效果如下所示。

```
mysql> SELECT * FROM author;
+-----+----------------+--------------------+-------+
| id  |  name          | tel                | age   |
+-----+----------------+--------------------+-------+
|  1  | 周志强         | 12332365911        |   32  |
|  2  | 章耐魂         | 12332365461        |   33  |
|  3  | 何恩承         | 12323563432        |   25  |
|  4  | 王小美         | 12325622357        |   55  |
|  5  | 张无奈         | 12349832321        |   27  |
|  6  | 黄华清         | 12250367654        |   38  |
|  7  | 李国良         | 12152566591        |   40  |
+-----+----------------+--------------------+-------+
7 rows in set (0.00 sec)
```

4. 使用加法运算符更新age字段的数据

更新数据还需要使用UPDATE语句，但是使用加法运算符可以将age字段中的所有数据一

次性进行更新，效果如下所示。

```
mysql> UPDATE author SET age= age +2;
Query OK, 7 rows affected (0.11 sec)
Rows matched: 7  Changed: 7  Warnings: 0
```

再次查询数据表，效果如下所示。

```
mysql> SELECT * FROM author;
+-----+----------+---------------+------+
| id  | name     |  tel          | age  |
+-----+----------+---------------+------+
|  1  | 周志强   | 12332365911   |  34  |
|  2  | 章耐魂   | 12332365461   |  35  |
|  3  | 何恩承   | 12323563432   |  27  |
|  4  | 王小美   | 12325622357   |  57  |
|  5  | 张无奈   | 12349832321   |  29  |
|  6  | 黄华清   | 12250367654   |  40  |
|  7  | 李国良   | 12152566591   |  42  |
+-----+----------+---------------+------+
7 rows in set (0.00 sec)
```

从运行结果可以看出，age字段中的所有数据都进行了加 2 运算。

6.2 系统函数


系统函数

MySQL提供了丰富的内置函数。这些函数使得数据的维护与管理更加方便，能够更好地提供数据的分析与统计功能。从函数的功能可以将系统函数分为数学函数、字符串函数、日期和时间函数、流程处理函数、获取MySQL信息函数、加密与解密函数等。

6.2.1 数学函数

数学函数主要对数字运算进行处理，主要包括绝对值函数、圆周率函数、获取整数的函数、返回列表中的最大值与最小值函数、角度与弧度互换函数、三角函数等，如表 6.3 所示。

表 6.3　数学函数

函数	功能
ABS(X)	获取X的绝对值
PI()	获取圆周率的值
CEIL(X)	对指定值进行向上取整，如果值为整数，则返回该值
CEILING(X)	对指定值进行向上取整，如果值为整数，则返回该值
FLOOR(X)	对指定值进行向下取整，如果值为整数，则返回该值

续表

函数	功能
LEAST(e1,e2,e3...)	获取列表中的最小值，列表中的数据既可以由数字组成，也可以由字符串组成
GREATEST(e1,e2,e3...)	获取列表中的最大值，列表中的数据既可以由数字组成，也可以由字符串组成
RADIANS(X)	将角度转化为弧度，X表示角度
DEGREES(X)	将弧度转化为角度，X表示弧度
SIN(X)	求X的正弦值
ASIN(X)	求X的反正弦值
COS(X)	求X的余弦值
ACOS(X)	求X的反余弦值
TAN(X)	求X的正切值
ATAN(X)	求X的反正切值
ATAN2(M, N)	返回两个参数的反正切值
POW(X, Y)	返回X的Y次方
POWER(X, Y)	返回X的Y次方
EXP(X)	返回e的X次方
SQRT(X)	返回X的平方根，当X的值为负数时，返回NULL
LN(X)	返回以e为底，X的对数，当X的值小于或者等于0时，返回的结果为NULL
LOG(X)	返回以e为底，X的对数，当X的值小于或者等于0时，返回的结果为NULL
LOG10(X)	返回以10为底，X的对数，当X的值小于或者等于0时，返回的结果为NULL
LOG2(X)	返回以2为底，X的对数，当X的值小于或等于0时，返回NULL
RAND()	返回一个0到1之间的随机数
RAND(X)	返回一个在0到1之间的随机数，多个函数的X值相同时产生的随机数相同
ROUND(X)	返回一个对X的值进行四舍五入后，最接近于X的整数
ROUND(X, Y)	返回一个对X的值进行四舍五入后，最接近X的值，保留到小数点后Y位
TRUNCATE(X, Y)	对X的值进行截断处理，保留到小数点后Y位
SIGN(X)	返回X的符号。X为正数结果返回1，为负数返回-1，为0返回0
M DIV N	获取M除以N的整数结果值，当N为0时，将返回NULL

续表

函数	功能
MOD(X, Y)	返回X除以Y后的余数。当X能被Y整除时，返回 0；当Y的值为 0 时，返回 NULL

实例 6-10　使用数学函数显示运算结果。

```
mysql> SELECT PI() AS PI,CEIL(3.6) AS CEIL,FLOOR(3.6) AS
FLOOR,LEAST(5,3,7,8) AS LEAST,POWER(2,2) AS POWER,RAND() AS
RAND,ROUND(5.789,2) AS ROUND,MOD(10,3) AS MO;
+-------------+-------+-----------+----------+------------+-----
----------------------+-----------+----------+
| PI          | CEIL  | FLOOR     | LEAST    | POWER      |
RAND                  | ROUND     | MO       |
+-------------+-------+-----------+----------+------------+-----
----------------------+-----------+----------+
| 3.141593    |     4 |         3 |        3 |          4 |
0.9912005448631948    |      5.79 |        1 |
+-------------+-------+-----------+----------+------------+-----
----------------------+-----------+----------+
1 row in set (0.00 sec)
```

从运行结果可以看出，使用不同的数学函数可以实现数学方面的数据运算。

6.2.2 字符串函数

字符串函数

字符串函数主要用于处理数据库中的字符串数据，如数据表中的姓名、家庭住址、兴趣爱好、产品介绍等字符串类型的数据。字符串函数如表 6.4 所示。

表 6.4　字符串函数

函数	功能
ASCII(S)	返回字符串S中的第一个字符的ASCII码值
CHAR_LENGTH(S)	返回字符串S中的字符个数
LENGTH(S)	返回字符串S的字节数
CONCAT(S1,S2,...,Sn)	将字符串S1,S2,...,Sn合并为一个字符串
CONCAT_WS(X, S1,S2,...,Sn)	将字符串S1,S2,...,Sn拼接成一个以X分隔的字符串
INSERT(oldstr, x, y, replacestr)	将字符串oldstr从第x位置开始，y个字符长度的子字符串替换为replacestr
LOWER(S)	将字符串S转化为小写
LCASE(S)	将字符串S转化为小写

续表

函数	功能
UPPER(S)	将字符串S转化为大写
LEFT(str, x)	返回字符串str最左边的x个字符组成的字符串，如果x的值为NULL，则返回NULL
RIGHT(str, x)	返回字符串str最右边的x个字符组成的字符串，如果x的值为NULL，则返回NULL
LPAD(str, n pstr)	使用字符串pstr对字符串str最左边进行填充，直到str字符串的长度达到n为止
RPAD(str, n, pstr)	使用字符串pstr对字符串str最右边进行填充，直到str字符串的长度达到n为止
LTRIM(S)	去除字符串S左边的空格
RTRIM(S)	去除字符串S右边的空格
TRIM(S)	去除字符串S两边的空格
TRIM(substr FROM str)	删除字符串str首尾的子字符串substr，如果未指定substr，则默认删除空格
REPEAT(str, x)	返回重复x次str的结果数据
REPLACE(S,A,B)	用字符串B替换字符串S中出现的所有字符串A，并返回替换后的字符串
STRCMP(S1, S2)	比较S1和S2的ASCII码值的大小，相等返回1，S1小于S2返回-1，S1大于S2返回1
SUBSTR(S, X, Y)	返回从字符串S中，从第X个位置开始，长度为Y的子字符串
MID(S, X, Y)	返回从字符串S中，从第X个位置开始，长度为Y的子字符串
SPACE(X)	返回一个由X个空格组成的字符串
LOCATE(substr, str)	返回字符串substr在字符串str中的位置
ELT(M, S1, S2, ..., Sn)	返回M指定位置的字符串
FIELD(S,S1,S2,...,Sn)	返回字符串S在字符串列表中第一次出现的位置，不存在S或S为NULL，返回0
FIND_IN_SET(S1, S2)	返回字符串S1在字符串S2中的位置。S1不存在，S2为空，S1或S2为NULL时，返回0
REVERSE(S)	返回与字符串S顺序完全相反的字符串
NULLIF(value1, value2)	比较两个字符串，如果value1与value2相等，则返回NULL，否则返回value1

实例 6-11　使用字符串函数实现字符串操作。

```
mysql> SELECT ASCII("abc"),CHAR_LENGTH('你好'),LENGTH('你
好'),INSERT('hello world',7,5,'MySQL'),LOWER('MYSQL');
+--------------+----------------------+----------------------+
--------------------------------------+----------------------+
|ASCII("abc")   |CHAR_LENGTH('你好')    | LENGTH('你好')        |
INSERT('hello world',7,5,'MySQL')      |LOWER('MYSQL')        |
+--------------+----------------------+----------------------+
--------------------------------------+----------------------+
|    97        |             2        |       6              |
         hello My                      |      mysql           |
+--------------+----------------------+----------------------+
--------------------------------------+----------------------+
1 row in set (0.00 sec)
```

从运行结果可以看出，使用不同的字符串函数可以实现对字符串的转换、提取和替换等操作。

6.2.3 日期和时间函数


日期和时间函数

数据表中的日期数据是一种十分常见的数据内容，它可以保证表的时效性和准确性。使用系统提供的日期和时间函数，能够灵活方便地处理日期和时间数据。系统提供的日期和时间函数如表 6.5 所示。

表 6.5　日期和时间函数

函数	功能
CURDATE()	返回当前日期，只包含年、月、日部分，格式为 YYYY-MM-DD
CURTIME()	返回当前时间，只包含时、分、秒部分，格式为 HH:MM:SS
NOW()	返回当前日期和时间，包含年、月、日、时、分、秒，格式为 YYYY-MM-DD HH:MM:SS
UNIX_TIMESTAMP(date)	将 date 转化为 UNIX 时间戳
FROM_UNIXTIME(timestamp)	将 UNIX 时间戳转化为日期时间，格式为 YYYY-MM-DD HH:MM:SS
UTC_DATE()	返回 UTC 日期
UTC_TIME()	返回 UTC 时间
YEAR(date)	返回日期所在的年份，取值返回为 1970~2069
MONTH(date)	返回日期对应的月份，取值返回为 1~12
MONTHNAME(date)	返回日期所在月份的英文名称
DAY(date)	只返回日期

续表

函数	功能
DAYNAME(date)	返回日期对应星期的英文名称
DAYOFWEEK(date)	返回日期对应的一周中的索引值。1 表示星期日，2 表示星期一
WEEKDAY(date)	返回日期对应的一周中的索引值，1 表示星期日，2 表示星期一
WEEK(date)	返回给定日期是一年中的第几周
WEEKOFYEAR(date)	返回日期位于一年中的第几周
DAYOFYEAR(date)	返回日期是一年中的第几天
DAYOFMONTH(date)	返回日期位于所在月份的第几天
QUARTER(date) 函数	返回日期对应的季度，取值范围为 1~4
HOUR(time)	返回指定时间的小时
MINUTE(time)	返回指定时间的分钟，取值范围为 0~59
TIME_TO_SEC(time)	将 time 转化为秒，并返回结果值
SEC_TO_TIME(seconds)	将 seconds 描述转化为包含小时、分钟和秒的时间
ADDTIME(time1, time2)	返回 time1 加上 time2 的时间
SUBTIME(time1, time2)	返回 time1 减去 time2 后的时间
DATEDIFF(date1, date2)	计算两个日期之间相差的天数
FROM_DAYS(N)	返回从 0000 年 1 月 1 日起，*N* 天以后的日期
LAST_DAY(date)	返回 date 所在月份的最后一天的日期
MAKEDATE(year, n)	返回给定年份与所在年份中的天数返回一个日期
MAKETIME(hour, minute, second)	将给定的小时、分钟和秒组合成时间并返回
PERIOD_ADD(time, n)	返回 time 加上 n 后的时间
TO_DAYS(date)	返回日期 date 距离 0000 年 1 月 1 日的天数

实例 6-12 使用不同形式显示日期和时间。

```
mysql> SELECT CURDATE(),CURTIME(), NOW();
+-----------------+-----------------+---------------------------
-+
| CURDATE() | CURTIME( ) | NOW()               |
+-----------------+-----------------+---------------------------
--+
| 2022-02-04  |  20:36:05  | 2022-02-04 20:36:05 |
+-----------------+-----------------+---------------------------
--+
```

```
1 row in set (0.00 sec)
```

从运行结果可以看出，展示了 3 种不同形式的日期。

6.2.4 流程处理函数


流程处理函数

流程处理函数用于通过条件的值修改程序运行的流程。流程处理函数可以根据不同的条件，执行不同的流程，在SQL语句中实现不同的条件选择。系统流程处理函数如表 6.6 所示。

表 6.6　流程处理函数

函数	功能
IF(value, value1,value2)	如果value的值为TRUE，则返回value1，否则返回value2
IFNULL(value1, value2)	如果value1不为NULL，则返回value1，否则返回value2
CASE WHEN value1 THEN result1［WHEN value2 THEN result2...］ELSE default END	如果WHEN后面的某个value值为TRUE，则返回对应THEN后面的结果值；如果所有WHEN后面的value值都为FALSE，则返回ELSE后面的结果值
CASE expr WHEN value1 THEN result1［WHEN value2 THEN result2...］ELSE default END	如果expr的值与某个WHEN后面的值相等，则返回对应THEN后面的结果；如果expr的值与所有WHEN后面的值都不相等，则返回ELSE后面的结果

实例 6-13　根据不同的条件执行不同的流程。

```
mysql> SELECT IF(1>2,'no','yes') AS IF1,IFNULL('a','b') AS
IF2,CASE WHEN 1 > 0 THEN 'yes' WHEN 1 <= 0 THEN 'no' ELSE
'unknown' END AS CASE1,CASE 1 WHEN 0 THEN 0 WHEN 1 THEN 1 ELSE
-1 END AS CASE2;
+-----+-----+-----------+-----------+
| IF1 | IF2 | CASE1     | CASE2     |
+-----+-----+-----------+-----------+
| yes | a   | yes       |     1     |
+-----+-----+-----------+-----------+
1 row in set (0.00 sec)
```

从运行结果可以看出，根据不同的条件会执行不同的流程。

6.2.5 获取MySQL信息函数


获取MySQL信息函数

通过MySQL内置的获取MySQL信息的函数，可以帮助开发人员轻松地获取MySQL的相关系统信息，从而帮助开发人员对MySQL进行维护或升级等工作。获取MySQL信息的函数如表 6.7 所示。

表 6.7　获取MySQL信息函数

函数	功能
VERSION()	返回当前MySQL的版本号
CONNECTION_ID()	返回当前MySQL服务器的连接数
DATABASE()	返回MySQL命令行当前所在的数据库
USER()	返回当前连接MySQL的用户名，返回结果格式为“主机名@用户名”
LAST_INSERT_ID()	返回自增列最新的值
CHARSET(value)	查看MySQL使用的字符集
COLLATION(value)	返回字符串value的排序方式

实例 6-14　使用函数获取MySQL的信息。

```
mysql> SELECT VERSION(),CONNECTION_ID(),DATABASE(),USER(),CHARSE
T('ABC');
+--------------------+------------------------+----------------+
--------------------+------------------------+
| VERSION()          | CONNECTION_ID()        | DATABASE()     |
USER()               | CHARSET('ABC')         |
+--------------------+------------------------+----------------+
--------------------+------------------------+
|   8.0.27           |     13                 |   shopping     |
  root@localhost     |      utf8mb4           |
+--------------------+------------------------+----------------+
--------------------+------------------------+
1 row in set (0.00 sec)
```

从运行结果可以看出，使用系统函数可以获取到MySQL的版本、服务器连接数等多种信息。


加锁与解锁函数

6.2.6 加锁与解锁函数

通过加锁与解锁可以更好地实现对数据的保护。MySQL提供的对数据进行加锁和释放锁的函数如表 6.8 所示。

表 6.8　加锁和解锁函数

函数	功能
GET_LOCK(value, timeout)	使用字符串value给定的名字获取锁，持续timeout秒。如果成功获取锁，则返回1；如果获取锁超时，则返回0；如果发生错误，则返回NULL。使用GET_LOCK(value, timeout)函数获取的锁，当执行RELEASE_LOCK(value)或断开数据库连接（包括正常断开和非正常断开），锁都会被解除

续表

函数	功能
RELEASE_LOCK(value)	将以 value 命名的锁解除。如果解除成功，则返回 1；如果线程还没有创建锁，则返回 0；如果以 value 命名的锁不存在，则返回 NULL
IS_FREE_LOCK(value)	判断以 value 命名的锁是否可以被使用。如果可以被使用，则返回 1；如果不能使用，也就是说正在被使用，则返回 0；如果发生错误，则返回 NULL
IS_USED_LOCK(value)	判断以 value 命名的锁是否正在被使用，如果正在被使用，则返回使用该锁的数据库连接 ID，否则返回 NULL

实例 6-15 使用加锁与解锁函数对指定字段进行操作。

```
mysql> SELECT GET_LOCK('mysql',1000);
+----------------------------------+
| GET_LOCK('mysql',1000)           |
+----------------------------------+
|                      1           |
+----------------------------------+
1 row in set (0.10 sec)
mysql> SELECT RELEASE_LOCK('mysql');
+----------------------------------+
| RELEASE_LOCK('mysql')            |
+----------------------------------+
|                    1             |
+----------------------------------+
1 row in set (0.00 sec)

mysql> SELECT IS_FREE_LOCK('mysql');
+----------------------------------+
| IS_FREE_LOCK('mysql') |
+----------------------------------+
|                    1             |
+----------------------------------+
1 row in set (0.00 sec)
mysql> SELECT IS_USED_LOCK('mysql');
+----------------------------------+
| IS_USED_LOCK('mysql')            |
+----------------------------------+
|                  NULL            |
+----------------------------------+
1 row in set (0.00 sec)
```

从运行结果可以看出，通过加锁和解锁函数可以实现对指定字段的加锁和解锁，并且还可以判断指定字段是否被加锁，是否被使用。

为author数据表添加作者住址字段和信息

任务 6-2　为author数据表添加作者住址字段和信息

【任务描述】

（1）为数据表author添加address字段。

（2）为address字段添加对应的内容。

【任务实施】

1. 为author数据表添加address字段

添加字段使用ALTER TABLE语句，效果如下所示。

```
mysql> ALTER TABLE author ADD COLUMN address varchar(30) DEFAULT
NULL;
Query OK, 0 rows affected (0.21 sec)
Records: 0  Duplicates: 0  Warnings: 0
```

2. 为address字段添加数据

添加的地址值包括北京、上海、深圳3个地方。使用IN比较运算符添加id值为1、3、5的作者的地址值为北京，效果如下所示。

```
mysql> UPDATE author SET address = '北京' WHERE id IN(1,3,5);
Query OK, 3 rows affected (0.01 sec)
Rows matched: 3  Changed: 3  Warnings: 0
```

使用求余运算符添加id值为2、4的作者的地址值为上海，效果如下所示。

```
mysql> UPDATE author SET address = '上海' WHERE id%2=0;
Query OK, 3 rows affected (0.10 sec)
Rows matched: 3  Changed: 3  Warnings: 0
```

使用等于运算符添加id值为7的作者的地址值为深圳，效果如下所示。

```
mysql> UPDATE author SET address = '深圳' WHERE id=7;
Query OK, 1 row affected (0.01 sec)
Rows matched: 1  Changed: 1  Warnings: 0
```

3. 查看数据表当前数据

使用SELECT语句查看添加数据后的author表，效果如下所示。

```
mysql> SELECT * FROM author;
+----+---------------+-------------------+--------+-----------+
| id |  name         |  tel              | age    | address   |
+----+---------------+-------------------+--------+-----------+
|  1 | 周志强        | 12332365911       |   32   |  北京     |
|  2 | 章耐魂        | 12332365461       |   33   |  上海     |
```

```
|  3 | 何恩承          | 12323563432        |   25 | 北京     |
|  4 | 王小美          | 12325622357        |   55 | 上海     |
|  5 | 张无奈          | 12349832321        |   27 | 北京     |
|  6 | 黄华清          | 12250367654        |   38 | 上海     |
|  7 | 李国良          | 12152566591        |   40 | 深圳     |
+----+---------------+--------------------+------+----------+
7 rows in set (0.00 sec)
```

从运行结果可以看出，使用不同的运算符成功地为address字段添加了对应的数据。

思政园地

打破国外垄断 国产数据库技术再突破


知识拓展（视频）


知识拓展（内容）

“根深才能叶茂，根技术理应高质量发展，做大做强。”在华为全球智慧金融峰会2023上，华为常务董事、华为云CEO张平安表示，华为公司将坚定战略投入数据库，并呼吁更多的伙伴和国产数据库厂商强强联合，共同抓住行业数字化转型和自主创新的巨大市场机会，共同建设繁荣开放的数据库产业生态。

当日，华为发布了新一代分布式数据库GaussDB，该数据库打破了国外对中国数据库市场的长期垄断。张平安表示，该技术已在华为内部IT系统和银行、保险、证券、能源等多个行业核心业务系统得到应用。

数字化和智能化转型是经济发展的新动力。其中软件技术在其中起到基础地位。但如果将整个软件产业体系比喻成一棵参天大树，数据库则被称为“根技术”，在其之上衍生和支撑着几乎全部的软件生态，进而支撑起整个数字中国各行各业核心业务系统的正常运转。

在2013年之前，中国大多数公司主要使用美国企业的数据库工具，如甲骨文、MySQL、微软等，其他国产数据库也受到外国相关技术的支撑。GaussDB的出现可谓打破了国外对中国数据库市场的长期垄断。

以金融业为例，其一直处于数字化转型的第一梯队。但在转型的过程中，最大挑战来自业务核心系统的升级。

据了解，针对传统集中式架构在处理性能、资源扩展、业务上线等方面的能力瓶颈，GaussDB已可实现平滑迁移，支撑业务高效创新与可持续。

在华为与中国邮政储蓄银行的合作中，邮储银行向华为开放了6.5亿用户的银行分布式新核心系统建设机会，其系统中就使用了GaussDB。目前，该系统已全面投产上线，可实现日均20亿笔交易、峰值6.7万笔/秒的能力，新系统效率平均提升40%。

（资料来源：王硕，《打破国外垄断 国产数据库技术再突破》，人民政协报，2023-06-15，节选）

感悟

华为的这一创举，无疑将激励更多的中国企业投身于技术创新的浪潮中，推动国产数据库技术的蓬勃发展。随着越来越多的“华为”式的企业涌现，中国在全球科技竞争中的

地位将更加稳固，中华民族的伟大复兴也将更加指日可待。随着GaussDB的广泛应用，我们有理由相信，中国将在未来的科技革命中扮演更加重要的角色，为全球科技发展贡献更多的中国智慧和中国方案。

本章习题

一、填空题

1. 等号运算符用于判断等号两边的数据是否相等，如果相等返回_____，不相等则返回_____。

2. 系统函数ASCII(S)可以返回字符串S中的第一个字符的_____。

二、选择题

1. 可以判断某个值是否与列表中的任意一个值相等的运算符是(　　)。

A. BETWEEN AND　　B. GREATEST

C. IS NOT NULL　　D. IN

2. 获取当前年月日时分秒的系统函数是(　　)。

A. CURDATE(　　)　　B. CURTIME(　　)

C. NOW(　　)　　D. DAY(date)

三、判断题

1. IF(value, value1,value2)流程处理函数中，如果value的值为FALSE，会返回value1 的值。(　　)

2. 安全等于运算符(<=>)和等号运算符(=)的功能完全相同。(　　)

四、操作题

为book数据表添加price字段，并使用比较运算符为price字段添加价格，图书价格如表6.9所示。

表6.9　图书价格

书名	价格/元	书名	价格/元
五国争霸	30	中途中的世界	50
大罗大陆探险记	50	胜败人生	80
你是谁	80	世界只有你	30
六国争霸	30	触摸世界	100