# 1 SQL

## 二表查询

SELECT \* FROM PRODUCT;

SELECT \* FROM CATEGORY;

SELECT \* FROM PRODUCT,CATEGORY WHERE CNO=CID;

SELECT \* FROM PRODUCT P INNER JOIN CATEGORY C ON P.CNO=C.CID;

SELECT \* FROM PRODUCT P LEFT OUTER JOIN CATEGORY C ON P.CNO=C.CID;

SELECT \* FROM PRODUCT LIMIT 0,3;

SELECT \* FROM PRODUCT LIMIT 3,3;

SELECT \* FROM PRODUCT WHERE CNO=(SELECT CID FROM CATEGORY WHERE CNAME='手机数码');

SELECT P.PNAME,C.CNAME FROM PRODUCT P LEFT OUTER JOIN CATEGORY C ON P.CNO=C.CID;--左连接

SELECT PNAME,(SELECT CNAME FROM CATEGORY C WHERE P.CNO=C.CID) AS'商品分类名称' FROM PRODUCT P;

SELECT \* FROM COMPANY WHERE NAME LIKE 'KI%' ;

SELECT \* FROM COMPANY WHERE AGE=25 OR AGE=27 ;

SELECT \* FROM COMPANY WHERE AGE > (SELECT AGE FROM COMPANY WHERE SALARY > 65000);

SELECT AGE FROM COMPANY WHERE EXISTS (SELECT AGE FROM COMPANY WHERE SALARY > 65000);

SELECT \* FROM PRODUCT P LEFT OUTER JOIN CATEGORY C ON P.CNO=C.CID UNION SELECT \* FROM PRODUCT RIGHT OUTER JOIN CATEGORY ON PRODUCT.CNO=CATEGORY.CID;

SELECT \* FROM PRODUCT P JOIN CATEGORY C ON P.CNO=C.CID

SELECT \* FROM PRODUCT P LEFT JOIN CATEGORY C ON P.CNO=C.CID

## 三表查询

CREATE TABLE AB(

ABID INT PRIMARY KEY NOT NULL,

XCV CHAR(50) NOT NULL,

ZXC INT NOT NULL

);

SELECT PID,PNAME,PRICE FROM PRODUCT P JOIN CATEGORY C ON P.CNO=C.CID JOIN AB ON C.CDESC=AB.ABID ；

SELECT \* FROM PRODUCT P JOIN CATEGORY C ON P.CNO=C.CID JOIN AB ON PRICE>100 AND XCV LIKE 'S%'--|| PID = 2)--|| ABID > 10--|| XCV LIKE 'S%';

SELECT \* FROM PRODUCT P JOIN PRODUCT R ON (P.PRICE=R.CNO);--同一个表查找

SELECT P.PNAME,R.PNAME AS ‘上级’ FROM PRODUCT P LEFT JOIN PRODUCT R ON (P.PRICE=R.CNO);--同一个表查找

SELECT P.PNAME,R.PNAME AS 上级 FROM PRODUCT P LEFT JOIN PRODUCT R ON (P.PRICE=R.CNO) WHERE P.PRICE<99;--SQL99比SQL92简单，条理清晰

SELECT P.PNAME,R.PNAME AS 上级,CATEGORY.CDESC FROM PRODUCT P LEFT JOIN PRODUCT R ON (P.PRICE=R.CNO) JOIN CATEGORY ON CATEGORY.CDESC BETWEEN 10 AND 11 WHERE P.PRICE<99;--三个表查询

SELECT AB.ABID,AB.XCV,P.PNAME,R.PNAME AS 上级,CATEGORY.CDESC FROM PRODUCT P LEFT JOIN PRODUCT R ON (P.PRICE=R.CNO) JOIN CATEGORY ON CATEGORY.CDESC BETWEEN 10 AND 11 JOIN AB ON AB.ABID BETWEEN 12 AND 13 WHERE P.PRICE<99;--四个个表查询

# C++



静态变量可以起到全局变量的作用



全局变量extern

|  |  |
| --- | --- |
| [continue 语句](https://www.runoob.com/cplusplus/cpp-continue-statement.html" \o "C++ 中的 continue 语句) | 引起循环跳过主体的剩余部分，立即重新开始测试条件。 |

for( ; ; )

{ printf("This loop will run forever.\n"); }

当条件表达式不存在时，它被假设为真。您也可以设置一个初始值和增量表达式，但是一般情况下，C++ 程序员偏向于使用 for(;;) 结构来表示一个无限循环。

**注意：**您可以按 Ctrl + C 键终止一个无限循环。

我们已经在前面的章节中讲解了 **[条件运算符 ? :](https://www.runoob.com/cplusplus/cpp-conditional-operator.html)**，可以用来替代 **if...else** 语句。它的一般形式如下：

Exp1 ? Exp2 : Exp3;

其中，Exp1、Exp2 和 Exp3 是表达式。请注意，冒号的使用和位置。

? 表达式的值是由 Exp1 决定的。如果 Exp1 为真，则计算 Exp2 的值，结果即为整个 ? 表达式的值。如果 Exp1 为假，则计算 Exp3 的值，结果即为整个 ? 表达式的值。

在 C++ 中要声明一个数组，需要指定元素的类型和元素的数量，如下所示：

type arrayName [ arraySize ];

这叫做一维数组。**arraySize** 必须是一个大于零的整数常量，**type** 可以是任意有效的 C++ 数据类型。

大括号 { } 之间的值的数目不能大于我们在数组声明时在方括号 [ ] 中指定的元素数目。

如果您省略掉了数组的大小，数组的大小则为初始化时元素的个数。因此，如果：

double balance[] = {1000.0, 2.0, 3.4, 7.0, 50.0};

您将创建一个数组，它与前一个实例中所创建的数组是完全相同的。下面是一个为数组中某个元素赋值的实例：

balance[4] = 50.0;

#include <iostream>

using namespace std;

#include <iomanip>

using std::setw;

int main()

{

int n[10]; // n 是一个包含 10 个整数的数组

// 初始化数组元素

for (int i = 0; i < 10; i++)

{

n[i] = i + 100; // 设置元素 i 为 i + 100

}

cout << "Element" << setw(13) << "Value" << endl;

// 输出数组中每个元素的值

for (int j = 0; j < 10; j++)

{

cout << setw(7) << j << setw(13) << n[j] << endl;

}

return 0;

}

例如，下面的声明创建了一个三维 5 . 10 . 4 整型数组：

int threedim[5][10][4];

多维数组可以通过在括号内为每行指定值来进行初始化。下面是一个带有 3 行 4 列的数组。

int a[3][4] = {

{0, 1, 2, 3} , /\* 初始化索引号为 0 的行 \*/

{4, 5, 6, 7} , /\* 初始化索引号为 1 的行 \*/

{8, 9, 10, 11} /\* 初始化索引号为 2 的行 \*/};

内部嵌套的括号是可选的，下面的初始化与上面是等同的：

int a[3][4] = {0,1,2,3,4,5,6,7,8,9,10,11};

#include <iostream>

using namespace std;

int main()

{

// 一个带有 5 行 2 列的数组

int a[5][2] = { {0,0}, {1,2}, {2,4}, {3,6},{4,8} };

// 输出数组中每个元素的值

for (int i = 0; i < 5; i++)

for (int j = 0; j < 2; j++)

{

cout << "a[" << i << "][" << j << "]: ";

cout << a[i][j] << endl;

}

return 0;

}

数组元素可以通过数组名称加索引进行访问。元素的索引是放在方括号内，跟在数组名称的后边。例如：

double salary = balance[9];

**runoobAarray** 是一个指向 &runoobAarray[0] 的指针，即数组 runoobAarray 的第一个元素的地址。因此，下面的程序片段把 **p** 赋值为 **runoobAarray** 的第一个元素的地址：

double \*p;double runoobAarray[10];

p = runoobAarray;

数组名是指向数组中第一个元素的常量指针。

使用数组名作为常量指针是合法的，反之亦然。因此，\*(runoobAarray + 4) 是一种访问 runoobAarray[4] 数据的合法方式。

一旦您把第一个元素的地址存储在 p 中，您就可以使用 \*p、\*(p+1)、\*(p+2) 等来访问数组元素。

#include <iostream>

using namespace std;

int main()

{

// 带有 5 个元素的双精度浮点型数组

double runoobAarray[5] = { 1000.0, 2.0, 3.4, 17.0, 50.0 };

double\* p;

p = runoobAarray;

// 输出数组中每个元素的值

cout << "使用指针的数组值 " << endl;

for (int i = 0; i < 5; i++)

{

cout << "\*(p + " << i << ") : ";

cout << \*(p + i) << endl;

}

cout << "使用 runoobAarray 作为地址的数组值 " << endl;

for (int i = 0; i < 5; i++)

{

cout << "\*(runoobAarray + " << i << ") : ";

cout << \*(runoobAarray + i) << endl;

}

cout << "P地址："<<p;

return 0;

}

形式参数是一个指针：

void myFunction(int \*param){...}

### **方式 2**

形式参数是一个已定义大小的数组：

void myFunction(int param[10]){...}

### **方式 3**

形式参数是一个未定义大小的数组：

void myFunction(int param[]){...}

#include <iostream>

#include <cstdlib>

#include <ctime>

using namespace std;

// 要生成和返回随机数的函数

int\* getRandom()

{

static int r[10];

// 设置种子

srand((unsigned)time(NULL));

for (int i = 0; i < 10; ++i)

{

r[i] = rand();

cout << r[i] << endl;

}

return r;

}

// 要调用上面定义函数的主函数

int main()

{

// 一个指向整数的指针

int\* p;

p = getRandom();

for (int i = 0; i < 10; i++)

{

cout << "\*(p + " << i << ") : ";

cout << \*(p + i) << endl;

}

return 0;

}

# GIT 配置

<https://blog.csdn.net/huangqqdy/article/details/83032408>

git安装好去GitHub上注册一个账号，注册好后，点击桌面上的Git Bash快捷图标，我们要用账号进行环境配置啦啦

# 配置用户名

git config --global user.name "username" //（ "username"是自己的账户名，）

# 配置邮箱

git config --global user.email "username@email.com" //("username@email.com"注册账号时用的邮箱)

以上命令执行结束后，可用 git config --global --list 命令查看配置是否OK

生成ssh

继续刚才的操作，在命令框中输入以下命令，

ssh-keygen -t rsa

1

然后连敲三次回车键，结束后去系统盘目录下（一般在 C:\Users\你的用户名.ssh）(mac: /Users/用户/.ssh）查看是否有。ssh文件夹生成，此文件夹中以下两个文件

4. 将ssh文件夹中的公钥（ id\_rsa.pub）添加到GitHub管理平台中，在GitHub的个人账户的设置中找到如下界面

title随便起一个，将公钥（ id\_rsa.pub）文件中内容复制粘贴到key中，然后点击Ass SSH key就好啦

5. 测试一下配置是否成功，在Git Bush命令框（就是刚才配置账号和邮箱的命令框）中继续输入以下命令，回车

ssh -T git@github.com

1

要是看见下面的这句话就说明配置好啦

在命令行 输入 git init  然后回车就好了

# PYTHON3

list =["a", "b", "c"]

print (list)

print ('hello');print ('runoob');

word = 'word'

sentence = "这是一个句子。"

paragraph = """这是一个段落。

包含了多个语句"""

#!/usr/bin/python

counter = 100 # 赋值整型变量

miles = 1000.0 # 浮点型

name = "John" # 字符串

print (counter)

print (miles)

print (name)

Python有五个标准的数据类型：

* Numbers（数字）
* String（字符串）
* List（列表）
* Tuple（元组）
* Dictionary（字典）

列表

s = 'abcdef'

s[1:5]

'bcde'

#!/usr/bin/python

# -\*- coding: UTF-8 -\*-

str = 'Hello World!'

print str # 输出完整字符串

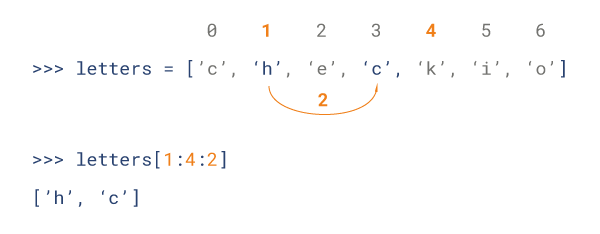
print str[0] # 输出字符串中的第一个字符

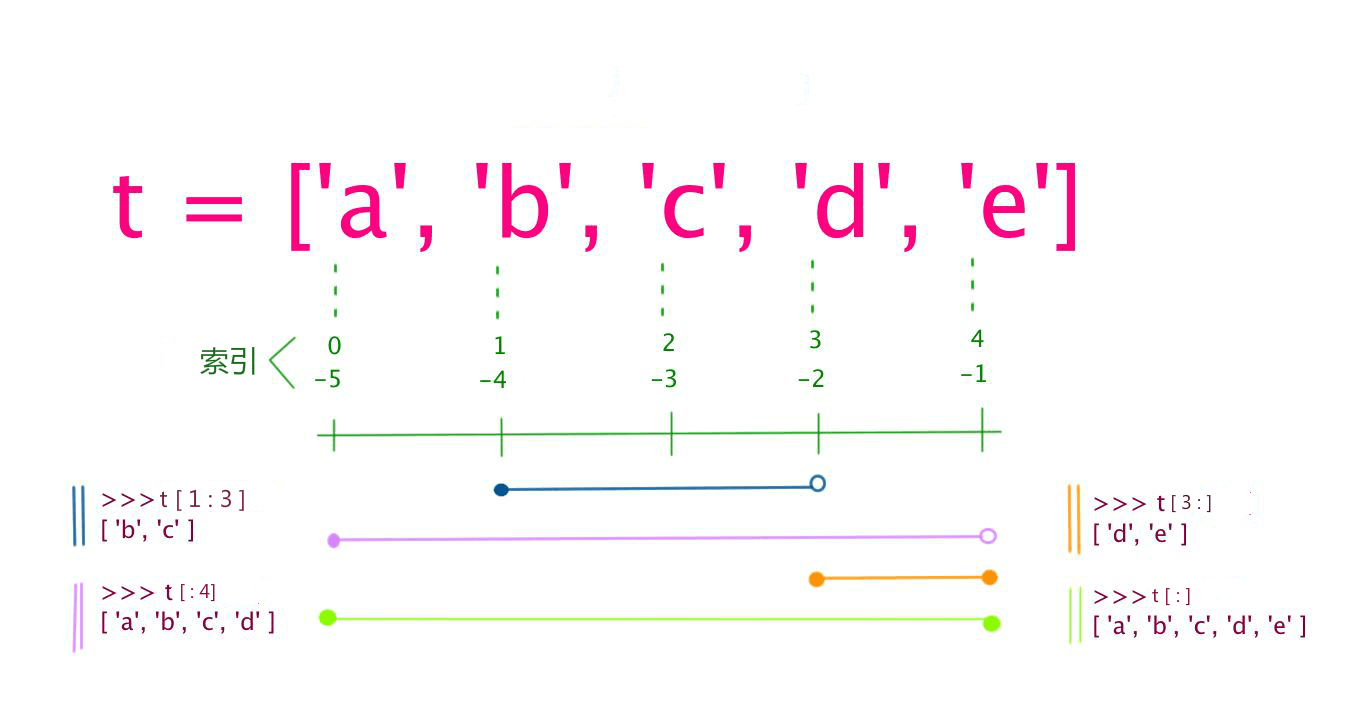
print str[2:5] # 输出字符串中第三个至第六个之间的字符串

print str[2:] # 输出从第三个字符开始的字符串

print str \* 2 # 输出字符串两次

print str + "TEST" # 输出连接的字符串





#!/usr/bin/python

# -\*- coding: UTF-8 -\*-

list = [ 'runoob', 786 , 2.23, 'john', 70.2 ]

tinylist = [123, 'john']

print list # 输出完整列表

print list[0] # 输出列表的第一个元素

print list[1:3] # 输出第二个至第三个元素

print list[2:] # 输出从第三个开始至列表末尾的所有元素

print tinylist \* 2 # 输出列表两次

print list + tinylist # 打印组合的列表

元组

#!/usr/bin/python

# -\*- coding: UTF-8 -\*-

tuple = ( 'runoob', 786 , 2.23, 'john', 70.2 )

tinytuple = (123, 'john')

print tuple # 输出完整元组

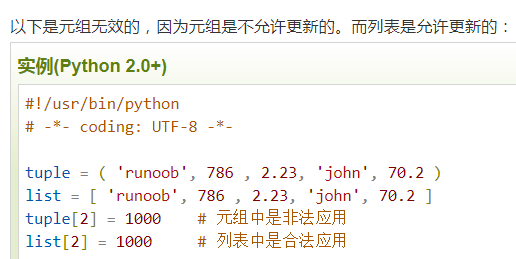
print tuple[0] # 输出元组的第一个元素

print tuple[1:3] # 输出第二个至第四个（不包含）的元素

print tuple[2:] # 输出从第三个开始至列表末尾的所有元素

print tinytuple \* 2 # 输出元组两次

print tuple + tinytuple # 打印组合的元组



字典(dictionary)是除列表以外python之中最灵活的内置数据结构类型。列表是有序的对象集合，字典是无序的对象集合。

两者之间的区别在于：字典当中的元素是通过键来存取的，而不是通过偏移存取。

字典用"{ }"标识。字典由索引(key)和它对应的值value组成。

#!/usr/bin/python

# -\*- coding: UTF-8 -\*-

dict = {}

dict['one'] = "This is one"

dict[2] = "This is two"

tinydict = {'name': 'runoob','code':6734, 'dept': 'sales'}

print dict['one'] # 输出键为'one' 的值

print dict[2] # 输出键为 2 的值

print tinydict # 输出完整的字典

print tinydict.keys() # 输出所有键

print tinydict.values() # 输出所有值

|  |  |
| --- | --- |
| [int(x [,base])](https://www.runoob.com/python/python-func-int.html" \t "https://www.runoob.com/python/_blank) | 将x转换为一个整数 |
| [long(x [,base] )](https://www.runoob.com/python/python-func-long.html" \t "https://www.runoob.com/python/_blank) | 将x转换为一个长整数 |
| [float(x)](https://www.runoob.com/python/python-func-float.html" \t "https://www.runoob.com/python/_blank) | 将x转换到一个浮点数 |
| [complex(real [,imag])](https://www.runoob.com/python/python-func-complex.html" \t "https://www.runoob.com/python/_blank) | 创建一个复数 |
| [str(x)](https://www.runoob.com/python/python-func-str.html" \t "https://www.runoob.com/python/_blank) | 将对象 x 转换为字符串 |
| [repr(x)](https://www.runoob.com/python/python-func-repr.html" \t "https://www.runoob.com/python/_blank) | 将对象 x 转换为表达式字符串 |
| [eval(str)](https://www.runoob.com/python/python-func-eval.html" \t "https://www.runoob.com/python/_blank) | 用来计算在字符串中的有效Python表达式,并返回一个对象 |
| [tuple(s)](https://www.runoob.com/python/att-tuple-tuple.html" \t "https://www.runoob.com/python/_blank) | 将序列 s 转换为一个元组 |
| [list(s)](https://www.runoob.com/python/att-list-list.html" \t "https://www.runoob.com/python/_blank) | 将序列 s 转换为一个列表 |
| [set(s)](https://www.runoob.com/python/python-func-set.html" \t "https://www.runoob.com/python/_blank) | 转换为可变集合 |
| [dict(d)](https://www.runoob.com/python/python-func-dict.html" \t "https://www.runoob.com/python/_blank) | 创建一个字典。d 必须是一个序列 (key,value)元组。 |
| [frozenset(s)](https://www.runoob.com/python/python-func-frozenset.html" \t "https://www.runoob.com/python/_blank) | 转换为不可变集合 |
| [chr(x)](https://www.runoob.com/python/python-func-chr.html" \t "https://www.runoob.com/python/_blank) | 将一个整数转换为一个字符 |
| [unichr(x)](https://www.runoob.com/python/python-func-unichr.html" \t "https://www.runoob.com/python/_blank) | 将一个整数转换为Unicode字符 |
| [ord(x)](https://www.runoob.com/python/python-func-ord.html" \t "https://www.runoob.com/python/_blank) | 将一个字符转换为它的整数值 |
| [hex(x)](https://www.runoob.com/python/python-func-hex.html" \t "https://www.runoob.com/python/_blank) | 将一个整数转换为一个十六进制字符串 |
| [oct(x)](https://www.runoob.com/python/python-func-oct.html" \t "https://www.runoob.com/python/_blank) | 将一个整数转换为一个八进制字符串 |