**第24章：使用游标**

1. **游标的定义**

游标（CURSOR）是一个存储在MYSQL服务器上的数据库查询，它不是一条SELECT语句，而是被该语句检索出来的一个结果集。在存储了游标之后，可以在检索出来的行中前进或后退一行或多行，应用程序可以根据需要滚动或浏览其中的数据。

1. **游标的使用**
2. **使用范围：**

**MYSQL游标只能用于存储过程或函数。**

1. **使用步骤：**
2. **声明定义（DECLARE）：定义要使用的SELECT语句，并没有检索数据。**

**declare 游标名 cursor for select\_statement**

1. **打开游标（OPEN）：此过程把数据才实际检索出来。**

**open 游标名**

1. **取出数据（FETCH）：根据需要检索各行。**

**fetch 游标名 into var1,var2[,...] --将取到的一行赋值给多个变量**

1. **关闭游标（CLOSE）：使用结束，必须关闭，可以再次打开。**

**close 游标名**

1. **应用举例：**
2. **创建表：**

**CREATE TABLE IF NOT EXISTS `store` (**

**`id` int(11) NOT NULL AUTO\_INCREMENT,**

**`name` varchar(20) NOT NULL,**

**`count` int(11) NOT NULL DEFAULT '1',**

**PRIMARY KEY (`id`)**

**) ENGINE=InnoDB DEFAULT CHARSET=latin1 AUTO\_INCREMENT=7;**

**2、插入记录：**

**INSERT INTO `store` (`id`, `name`, `count`) VALUES**

**(1, 'android', 15),**

**(2, 'iphone', 14),**

**(3, 'iphone', 20),**

**(4, 'android', 5),**

**(5, 'android', 13),**

**(6, 'iphone', 13);**

**3、创建存储过程与游标**

**--在windows系统中写存储过程时，如果需要使用declare声明变量，需要添加这个关键字，否则会报错。**

**delimiter //**

**drop procedure if exists StatisticStore;**

**CREATE PROCEDURE StatisticStore()**

**BEGIN**

**--创建接收游标数据的变量**

**declare c int;**

**declare n varchar(20);**

**--创建总数变量**

**declare total int default 0;**

**--创建结束标志变量**

**declare done int default false;**

**--创建游标**

**declare cur cursor for select name,count from store where name = 'iphone';**

**--指定游标循环结束时的返回值**

**declare continue HANDLER for not found set done = true;**

**--设置初始值**

**set total = 0;**

**--打开游标**

**open cur;**

**--开始循环游标里的数据**

**read\_loop:loop**

**--根据游标当前指向的一条数据**

**fetch cur into n,c;**

**--判断游标的循环是否结束**

**if done then**

**leave read\_loop; --跳出游标循环**

**end if;**

**--获取一条数据时，将count值进行累加操作，这里可以做任意你想做的操作，**

**set total = total + c;**

**--结束游标循环**

**end loop;**

**--关闭游标**

**close cur;**

**--输出结果**

**select total;**

**END;**

**4、调用存储过程**

**call StatisticStore();**

**存储过程处理完成后，游标就消失，因为它局限于存储过程。如果不明确关闭游标，MYSQL将会在到达END语句时自动关闭它。（隐含关闭）**

**5、说明：**

**fetch是获取游标当前指向的数据行，并将指针指向下一行（不重复读取同一行），当游标已经指向最后一行时继续执行会造成游标溢出。**

**使用loop循环游标时，他本身是不会监控是否到最后一条数据了。在MySql中，造成游标溢出时会引发mysql预定义的NOT FOUND错误，所以在上面使用的代码指定了当引发not found错误时定义一个continue 的事件，指定这个事件发生时修改done变量的值。**

**DECLARE语句的次序：用DECLARE语句定义的局部变量必须在定义任意游标或句柄之前定义，而句柄必须在游标之后定义。否则报错！**

1. **样例库举例：**

1、创建游标

CREATE PROCEDURE processorders()

BEGIN

DECLARE ordernumbers CURSOR

FOR

SELECT order\_num FROM orders;

END;

2、打开和关闭游标

OPEN ordernumbers;

CLOSE ordernumbers;

3、使用游标数据

CREATE PROCEDURE processorders()

BEGIN

-- Declare local variables

DECLARE o INT;

-- Declare the cursor

DECLARE ordernumbers CURSOR

FOR

SELECT order\_num FROM orders;

-- Open the cursor

OPEN ordernumbers;

-- Get order number

FETCH ordernumbers INTO o;

SELECT O;

-- Close the cursor

CLOSE ordernumbers;

END;

4、循环检索数据

CREATE PROCEDURE processorders()

BEGIN

-- Declare local variables

DECLARE done BOOLEAN DEFAULT 0;

DECLARE o INT;

-- Declare the cursor

DECLARE ordernumbers CURSOR

FOR

SELECT order\_num FROM orders;

-- Declare continue handler

**DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET done=1;**

-- Open the cursor

OPEN ordernumbers;

-- Loop through all rows

REPEAT

-- Get order number

FETCH ordernumbers INTO o;

SELECT o;

-- End of loop

UNTIL done END REPEAT;

-- Close the cursor

CLOSE ordernumbers;

END;

5、循环处理数据

CREATE PROCEDURE processorders()

BEGIN

-- Declare local variables

DECLARE done BOOLEAN DEFAULT 0;

DECLARE o INT;

DECLARE t DECIMAL(8,2);

-- Declare the cursor

DECLARE ordernumbers CURSOR

FOR

SELECT order\_num FROM orders;

-- Declare continue handler,**SQLSTATE '02000' 是一个“未找到”条件**

**DECLARE CONTINUE HANDLER FOR SQLSTATE '02000' SET done=1;**

-- Create a table to store the results

CREATE TABLE IF NOT EXISTS ordertotals

(order\_num INT, total DECIMAL(8,2));

-- Open the cursor

OPEN ordernumbers;

-- Loop through all rows

REPEAT

-- Get order number

FETCH ordernumbers INTO o;

-- Get the total for this order

CALL ordertotal(o, 1, t);

-- Insert order and total into ordertotals

INSERT INTO ordertotals(order\_num, total)

VALUES(o, t);

-- End of loop

UNTIL done END REPEAT;

-- Close the cursor

CLOSE ordernumbers;

END;

CALL processorders();

**SELECT DISTINCTROW \***

**FROM ordertotals order by order\_num;**

附关系图：

