



下载APP



16 | 游标：对于数据集中的记录，该怎么逐条处理？

2021-04-15 朱晓峰

MySQL 必知必会

[进入课程 >](#)**讲述：朱晓峰**

时长 11:20 大小 10.39M



你好，我是朱晓峰。今天，我来和你聊一聊游标。

咱们前面学习的 MySQL 数据操作语句，都是针对结果集合的。也就是说，每次处理的对象都是一个数据集合。如果需要逐一处理结果集中的记录，就会非常困难。

虽然我们也可以通过筛选条件 WHERE 和 HAVING，或者是限定返回记录的关键字 LIMIT 返回一条记录，但是，却无法在结果集中像指针一样，向前定位一条记录、向后定位一条记录，或者是随意定位到某一条记录，并对记录的数据进行处理。



这个时候，就可以用到游标。所谓的游标，也就是能够对结果集中的每一条记录进行定位，并对指向的记录中的数据进行操作的数据结构。

这么说可能有点抽象，我举一个生活中的例子，你一看就明白了。比如，你想去某个城市旅游，现在需要订酒店。你打开预订酒店的 App，设置好价格区间后进行搜索，得到了一个酒店列表。接下来，你可能要逐条查看列表中每个酒店的客户评价，最后选择一个口碑不错的酒店。这个逐条搜索并对选中的数据进行操作的过程，就相当于游标对数据记录进行操作的过程。

今天我就来给你讲一讲游标的使用方法，同时还会通过一个案例串讲，帮助你更好地使用游标，让你能够轻松地处理数据集中的记录。

游标的使用步骤

游标只能在存储程序内使用，存储程序包括存储过程和存储函数。关于存储过程，我们上节课刚刚学过，这里我简单介绍一下存储函数。创建存储函数的语法是：

```
1 CREATE FUNCTION 函数名称 (参数) RETURNS 数据类型 程序体
```

[复制代码](#)

存储函数与存储过程很像，但有几个不同点：

1. 存储函数必须返回一个值或者数据表，存储过程可以不返回。
2. 存储过程可以通过 CALL 语句调用，存储函数不可以。
3. 存储函数可以放在查询语句中使用，存储过程不行。
4. 存储过程的功能更加强大，包括能够执行对表的操作（比如创建表，删除表等）和事务操作，这些功能是存储函数不具备的。

这节课，我们主要学习下游标在存储过程中的使用方法，因为游标在存储过程中更常用。游标在存储函数中的使用方法和在存储过程中的使用方法是相同的。

在使用游标的时候，主要有 4 个步骤。

第一步，定义游标。语法结构如下：

```
1
```

[复制代码](#)

```
DECLARE 游标名 CURSOR FOR 查询语句
```

这里就是声明一个游标，它可以操作的数据集是“查询语句”返回的结果集。

第二步，打开游标。语法结构如下：

```
1 OPEN 游标名称;
```

[复制代码](#)

打开游标之后，系统会为游标准备好查询的结果集，为后面游标的逐条读取结果集中的记录做准备。

第三步，从游标的数据结果集中读取数据。语法结构是这样的：

```
1 FETCH 游标名 INTO 变量列表;
```

[复制代码](#)

这里的意思是通过游标，把当前游标指向的结果集中那一条记录的数据，赋值给列表中的变量。

需要注意的是，**游标的查询结果集中的字段数，必须跟 INTO 后面的变量数一致**，否则，在存储过程执行的时候，MySQL 会提示错误。

第四步，关闭游标。语法结构如下：

```
1 CLOSE 游标名;
```

[复制代码](#)

用完游标之后，你一定要记住及时关闭游标。因为游标会占用系统资源，如果不及时关闭，游标会一直保持到存储过程结束，影响系统运行的效率。而关闭游标的操作，会释放游标占用的系统资源。

知道了基本步骤，下面我就结合超市项目的实际案例，带你实战一下。

案例串讲

在超市项目的进货模块中，有一项功能是对进货单数据进行验收。其实就是在对进货单的数据确认无误后，对进货单的数据进行处理，包括增加进货商品的库存，并修改商品的平均进价。下面我用实际数据来演示一下这个操作流程。

这里我们要用到进货单头表（demo.importheadl）、进货单明细表（demo.importdetails）、库存表（demo.inventory）和商品信息表（demo.goodsmaster）。

进货单头表：

Listnumber (进货单编号)	Supplierid (供货商编号)	Stockid (仓库编号)	Operatorid (操作员编号)	Totalquantity (进货数量)	Totalvalue (进货金额)	Recordingdate (进货日期)
1234	1	1	1	8	171	2020-12-12

进货单明细表：

Listnumber (进货单号)	Itemnumber (商品编号)	Quantity (进货数量)	Importprice (进货价格)	Importvalue (进货金额)
1234	1	5	33	165
1234	2	3	2	6

库存表：

Stocknumber (仓库编号)	Itemnumber (商品编号)	Invquantity (库存数量)
1	1	10
1	2	20

商品信息表：

Itemnumber (商品编号)	Barcode (商品条码)	Goodsname (商品名称)	Specification (商品规格)	Unit (单位)	Salesprice (售价)	Avgimportprice (平均进价)
1	0001	书	16开	本	89	30
2	0002	笔	NULL	支	5	3

要验收进货单，我们就需要对每一个进货商品进行两个操作：


- 1. 在现有库存数量的基础上，加上本次进货的数量；
- 2. 根据本次进货的价格、数量，现有商品的平均进价和库存，计算新的平均进价：（本次进货价格 * 本次进货数量 + 现有商品平均进价 * 现有商品库存） / （本次进货数量 + 现有库存数量）。

针对这个操作，如果只用我们在 [第 4 讲](#) 里学习的 SQL 语句，完成起来就比较困难。

因为我们需要通过应用程序来控制操作流程，做成一个循环操作，每次只查询一种商品的数据记录并进行处理，一直到把进货单中的数据全部处理完。这样一来，应用必须发送很多的 SQL 指令到服务器，跟服务器的交互多，不仅代码复杂，而且也不够安全。

这个时候，如果使用游标，就很容易了。因为所有的操作都可以在服务器端完成，应用程序只需要发送一个命令调用存储过程就可以了。现在，我们就来看看如何用游标来解决问题。

我用代码创建了一个存储过程 `demo.mytest ()` 。当然，你也完全可以在 Workbench 中创建存储过程，非常简单，我就不多说了。创建存储过程的代码如下：

 复制代码

```
1 mysql> DELIMITER //
```

```
2 mysql> CREATE PROCEDURE demo.mytest(mylistnumber INT)
```

```
3 -> BEGIN
```

```
4 -> DECLARE mystockid INT;
```

```
5 -> DECLARE myitemnumber INT;
```

```
6 -> DECLARE myquantity DECIMAL(10,3);
```

```
7 -> DECLARE myprice DECIMAL(10,2);
```

```
8 -> DECLARE done INT DEFAULT FALSE; -- 用来控制循环结束
```

```
9 -> DECLARE cursor_importdata CURSOR FOR -- 定义游标
```

```
10 -> SELECT b.stockid,a.itemnumber,a.quantity,a.importprice
```

```
11 -> FROM demo.importdetails AS a
```

```
12 -> JOIN demo.importthead AS b
```

```
13 -> ON (a.listnumber=b.listnumber)
```

```
14 -> WHERE a.listnumber = mylistnumber;
```

```
15 -> DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE; -- 条件处理语句
```

```
16 ->
```

```
17 -> OPEN cursor_importdata; -- 打开游标
```

```
18 -> FETCH cursor_importdata INTO mystockid,myitemnumber,myquantity,myprice; --
```

```
19 -> REPEAT
```

```
20 -> -- 更新进价
```

```
21 -> UPDATE demo.goodsmaster AS a,demo.inventory AS b
```

```
22 -> SET a.avgimportprice = (a.avgimportprice*b.invquantity+myprice*myquantity)/
```

```
23 -> WHERE a.itemnumber=b.itemnumber AND b.stockid=mystockid AND a.itemnumber=my
```

```
24 -> -- 更新库存
```

```
25 -> UPDATE demo.inventory
```

```
26 -> SET invquantity = invquantity + myquantity
```

```
27 -> WHERE stockid = mystockid AND itemnumber=myitemnumber;
```

```
28 -> -- 获取下一条记录
```

```
29 -> FETCH cursor_importdata INTO mystockid,myitemnumber,myquantity,myprice;
```

```
30 -> UNTIL done END REPEAT;
```

```
31 -> CLOSE cursor_importdata;
```

```
32 -> END
```

```
33 -> //
```

```
34 Query OK, 0 rows affected (0.02 sec)
```

```
35 -> DELIMITER ;
```

这段代码比较长，核心操作有 6 步，我来给你详细解释下。

1. 把 MySQL 的分隔符改成 “//” 。
2. 开始程序体之后，我定义了 4 个变量，分别是 `mystockid`、`myitemnumber`、`myquantity` 和 `myprice`，这几个变量的作用是，存储游标中读取的仓库编号、商品编

号、进货数量和进货价格数据。

3. 定义游标。这里我指定了游标的名称，以及游标可以处理的数据集（mylistnumber 指定的进货单的全部进货商品明细数据）。
4. 定义条件处理语句 “DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;”。
5. 打开游标，读入第一条记录，然后开始执行数据操作。
6. 关闭游标，结束程序。

可以看到，在这个存储过程中，我用到了条件处理语句，它的作用是告诉系统，在存储程序遇到问题的时候，应该如何处理。

条件处理语句

条件处理语句的语法结构：

```
1 DECLARE 处理方式 HANDLER FOR 问题 操作;
```

 复制代码

下面我结合刚刚的 “DECLARE CONTINUE HANDLER FOR NOT FOUND SET done = TRUE;”，来解释一下条件处理语句是如何工作的。

1. **语法结构中的“问题”是指 SQL 操作中遇到了什么问题。**比如这里的问题是 “NOT FOUND”，意思就是游标走到结果集的最后，没有记录了。也就是说，数据集中的所有记录都已经处理完了。
2. 执行的操作是 “SET done=TRUE”，done 是我定义的用来标识数据集中的数据是否已经处理完成的一个标记。done=TRUE，意思是数据处理完成了。
3. **处理方式有 2 种选择，分别是 “CONTINUE” 和 “EXIT”**，表示遇到问题，执行了语法结构中的 “操作” 之后，是选择继续运行程序，还是选择退出，结束程序。

所以，这个条件处理语句的意思就是：当游标读到结果集的最后，没有记录了，设置操作完成标识为真，然后继续运行程序。

在存储过程的第 5 步，为了逐一处理每一条记录，我还使用了流程控制语句。

解决复杂问题不可能通过一个 SQL 语句完成，我们需要执行多个 SQL 操作。流程控制语句的作用就是控制存储过程中 SQL 语句的执行顺序，是我们完成复杂操作必不可少的一部分。下面我就给你具体讲解一下。

流程控制语句

MySQL 的流程控制语句也只能用于存储程序。主要有 3 类。

1. 跳转语句：ITERATE 和 LEAVE 语句。
2. 循环语句：LOOP、WHILE 和 REPEAT 语句。
3. 条件判断语句：IF 语句和 CASE 语句。

接下来我依次讲解一下跳转语句、循环语句和条件判断语句。


跳转语句

1. ITERATE 语句：只能用在循环语句内，表示重新开始循环。
2. LEAVE 语句：可以用在循环语句内，或者以 BEGIN 和 END 包裹起来的程序体内，表示跳出循环或者跳出程序体的操作。

循环语句


LOOP 语句的语法结构是：

```
1  标签: LOOP
2  操作
3  END LOOP 标签;
```

 复制代码

关于这个语句，需要注意的是，LOOP 循环不能自己结束，需要用跳转语句 ITERATE 或者 LEAVE 来进行控制。


WHILE 语句的语法结构：

 复制代码

```
1 WHILE 条件 DO
2 操作
3 END WHILE;
```

WHILE 循环通过判断条件是否为真来决定是否继续执行循环中的操作，你要注意一点，**WHILE 循环是先判断条件，再执行循环体中的操作。**

REPEAT 语句的语法结构：

 复制代码


```
1 REPEAT
2 操作
3 UNTIL 条件 END REPEAT;
```

REPEAT 循环也是通过判断条件是否为真来决定是否继续执行循环内的操作的，与 WHILE 不同的是，**REPEAT 循环是先执行操作，后判断条件。**

最后我来讲讲条件判断语句：IF 语句和 CASE 语句。

条件判断语句

IF 语句的语法结构是：

 复制代码

```
1 IF 表达式1 THEN 操作1
2 [ELSEIF 表达式2 THEN 操作2].....
3 [ELSE 操作N]
4 END IF
```

这里“[]”中的内容是可选的。**IF 语句的特点是，不同的表达式对应不同的操作。**

CASE 语句的语法结构是：

 复制代码

```
1 CASE 表达式
```

```
2 WHEN 值1 THEN 操作1
3 [WHEN 值2 THEN 操作2].....
4 [ELSE 操作N]
5 END CASE;
```

这里“[]”中的内容是可选的。CASE 语句的特点是，表达式不同的值对应不同的操作。

到这里，我们处理进货单验收的存储过程就创建好了。现在，让我们来运行一下这个存储过程，看看能不能得到我们想要的结果：

 复制代码

```
1 mysql> CALL demo.mytest(1234);           -- 调用存储过程，验收单号是1234的进货单
2 Query OK, 0 rows affected (11.68 sec)     -- 执行成功了
3
4 mysql> select * from demo.inventory;      -- 查看库存，已经改过来了
5 +-----+-----+-----+
6 | stockid | itemnumber | invquantity |
7 +-----+-----+-----+
8 | 1 | 1 | 15.000 |
9 | 1 | 2 | 23.000 |
10 +-----+-----+-----+
11 2 rows in set (0.00 sec)
12
13 mysql> select * from demo.goodsmaster;    -- 查看商品信息表，平均进价也改过来了
14 +-----+-----+-----+-----+-----+-----+-----+
15 | itemnumber | barcode | goodsname | specification | unit | salesprice | avgim
16 +-----+-----+-----+-----+-----+-----+-----+
17 | 1 | 0001 | 书 | 16开 | 本 | 89.00 | 31.00 |
18 | 2 | 0002 | 笔 | NULL | 包 | 5.00 | 2.87 |
19 +-----+-----+-----+-----+-----+-----+-----+
20 2 rows in set (0.00 sec)
```

很显然，库存和平均价格都被正确地计算出来了。

最后，有个小问题要提醒你注意：**如果一个操作要用到另外一个操作的结果，那我们一定不能搞错操作的顺序。**比如，在刚刚的例子中，我是先计算平均价格，后消减库存数量，这就是因为，计算平均价格的时候会用到库存数量，如果先消减库存数量，平均价格的计算就会不准。

总结

这节课，我们学习了游标的使用方法，包括在存储过程中使用游标的 4 个步骤，分别是定义游标、打开游标、读取游标数据和关闭游标。除此之外，我还介绍了经常与游标结合使用的流程控制语句，包括循环语句 LOOP、WHILE 和 REPEAT；条件判断语句 IF 和 CASE；还有跳转语句 LEAVE 和 ITERATE。

游标是 MySQL 的一个重要的功能，为逐条读取结果集中的数据，提供了完美的解决方案。跟在应用层面实现相同的功能相比，游标可以在存储程序中使用，效率高，程序也更加简洁。但是游标会消耗系统资源，所以我建议你养成用完之后就关闭的习惯，这样才能提高系统的整体效率。

思考题

假设我有一个数据表 demo.test，具体信息如下所示：

Id (编号, INT, 主键)	Myquant (INT)
1	100
2	101
3	102
4	103

你能自己写一个简单的存储过程，用游标来逐一处理一个数据表中的数据吗？

要求：编号为偶数的记录， $myquant = myquant + 1$ ；编号是奇数的记录， $myquant = myquant + 2$ 。

欢迎在留言区写下你的思考和答案，我们一起交流讨论。如果你觉得今天的内容对你有所帮助，欢迎你把它分享给你的朋友或同事，我们下节课见。

提建议

更多课程推荐

深入浅出计算机组成原理

带你掌握计算机体系全貌

徐文浩

bothub 创始人



涨价倒计时 🕒

今日订阅 **¥89**，5月12日涨价至 **¥199**

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。

上一篇 15 | 存储过程：如何提高程序的性能和安全性？

下一篇 17 | 触发器：如何让数据修改自动触发关联操作，确保数据一致性？

精选留言 (3)

写留言



lesserror

2021-04-17

游标这块儿知识比较模糊，以后如果要使用游标，看这一讲基本够了，老师讲的很细致。以下是简单的总结：

1. 游标只能在存储程序内使用，存储程序包括存储过程和存储函数。
2. 游标为逐条读取结果集中的数据，提供了完美的解决方案。因为不需要后端语言写循...

展开 ∨



Harry 
2021-04-16

游标 适用于需要对集合中的行进行单独处理的场景。

展开 ▾



Harry 
2021-04-15

工作中已经在使用 function 来创建表、修改字段及返回自增 id 等。

本节课，需要重点理解进货单的验收逻辑，才能很好的掌握游标、流程控制、条件处理等内容。 ...

展开 ▾

