## Mysql实践

### 一、基础知识

1、外键

外键是某个表的一列，也是另一个表的主键。它定义了表与表之间的关系。一般在多的一方建立外键（1对N），也有利于数据完整性。

2、完全限定列名

一个点分割的表名和列名

引入的列具有二义性时，必须使用完全限定名，否则会报错

3、笛卡尔积

连接两张表A,B而不附带任何条件。返回的记录行数为countA \* countB

4、内连接的两种写法

1. select vend\_name,prod\_name,prod\_price from vendors,products

where vendors.vend\_id = products.vend\_id;

2. select vend\_name,prod\_name,prod\_price from vendors inner join products on vendors.vend\_id = products.vend\_id;

5、多表关联的语法

6、多表连接性能问题

MySQL在运行时关联指定的每个表以处理联结。这种处理可能是非常耗费资源的，因此应该仔细，不要联结不必要的表。联结的表越多，性能下降越厉害

7、子查询

子查询的读取从内到外，写的时候将子查询看做是一个或两个固定值。

8、多做实验

对于一个查询功能，肯定存在多种sql写法，多尝试，找到最优方法。

9、别名

1.列别名

可返回客户端

2.表别名

缩短sql语句，允许在单条select语句中多次使用相同的表

比如自连接

10、自连接

表A关联表A进行查询

例子：DTNTR商品是有问题的，查询该商品对应供应商下所有的商品

select p1.prod\_id, p1.prod\_name from products p1 inner join products p2 on p1.vend\_id = p2.vend\_id and p2.prod\_id='DTNTR'

11、外连接

1.左连接 left join

A left join B：A表所有数据以及符合关联条件的B表数据

2.右连接 right join

A right join B: B表所有数据以及符合关联条件的A表数据

12、多表关联中使用聚集函数

单表中group by时,查找的列只能是分组的列和聚集函数。

多表关联后group by,查找的列可以使任意列

13、组合查询

使用组合查询的情形：

1. 在单个查询中从不同的表返回类似结构的数据

2. 对单个表执行多个查询，按单个查询返回数据

Union使用的规则：

1.连接select语句

2.每个查询必须包含相同的列，表达式或聚集函数

3.order by子句只能出现在最后一个select语句中

14、union 和 union all

1.union会消除重复的行

2.union all会返回所有的行

### 二、存储过程

含义：存储过程简单来说，就是为以后的使用而保存

的一条或多条MySQL语句的集合。可将其视为批文件

特点：简单、安全、高性能

例1：获取统计商品信息

创建存储过程：

OUT：存储过程的输出参数

IN： 存储过程的输入参数

INOUT: 存储过程输入输出参数

INTO：后面接变量名表示查询出来的结果存储到该变量中

|  |
| --- |
| create PROCEDURE collect(OUT avgprice DECIMAL(8,2),  OUT maxprice DECIMAL(8,2), OUT minprice DECIMAL(8,2)  )  BEGIN  select avg(prod\_price) from products into avgprice;  select max(prod\_price) from products into maxprice;  select min(prod\_price) from products into minprice;  END; |

调用存储过程：

必须先调用存储过程，才能查询到变量的值

|  |
| --- |
| call collect(@avgprice,@maxprice,@minprice);  select @avgprice,@maxprice,@minprice; |

删除存储过程：

|  |
| --- |
| drop PROCEDURE if EXISTS collect; |

例2：根据订单号获取统计的金额

|  |
| --- |
| create PROCEDURE totalamount(IN ordernum int, OUT total DECIMAL(8,2))  BEGIN  select sum(order\_item\*item\_price) from orderitems where order\_num=ordernum INTO total;  END;  call totalamount(20005, @total);  SELECT @total; |

存储过程的进阶：

当存储过程中包含业务规则和智能处理时，这才是它的优势处。