讲述MySQL中的预编译语句并从MySQL的Connector/J源码出发讲述其在Java语言中相关使用。

一、预编译语句是什么

一条sql在DB接收到最终执行完毕返回可以分为下面三个过程：

1. 词法和语义解析
2. 优化sql语句，指定执行计划
3. 执行并返回结果

把这种普通语句称作Immediate Statement。

但是很多情况，一条sql语句可能会反复执行，或者每次执行的时候只有个别的值不同（比如query的where子句值不同，update的set子句值不同,insert的values值不同）。

如果每次都需要经过上面的词法语义解析、语句优化、制定执行计划等，则效率就明显不行了。

预编译语句就是将这类语句中的值用占位符替代，可以视为将sql语句模板化或者说参数化，一般称这类语句叫Prepared Statements或者Parameterized Statements。预编译语句的优势在于归纳为：一次编译、多次运行，省去了解析优化等过程；此外预编译语句能防止sql注入。当然就优化来说，很多时候最优的执行计划不是光靠知道sql语句的模板就能决定了，往往就是需要通过具体值来预估出成本代价。

二、MySQL的预编译功能

MySQL老版本（4.1之前）是不支持服务端预编译的，但基于目前业界生产环境普遍的情况，基本可以认为MySQL支持服务端预编译。

MySQL中预编译语句的使用。首先有一张测试表user，结构如下：



#### 1、编译

通过PREPARE stmt\_name FROM preparable\_stm的语法来预编译一条sql语句。

prepare ins from 'insert into user select ?,?,?,?';

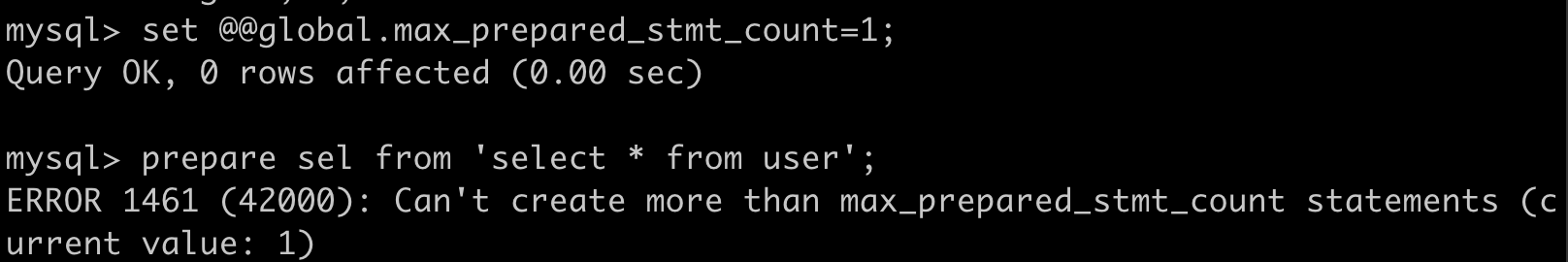
#### 2、执行

通EXECUTE stmt\_name [using @var\_name [, @var\_name] ...]的语法来执行编译语句。

set @a=6,@b='lucy',@c=23,@d='lucy@163.com';

execute ins using @a,@b,@c,@d; 执行

MySQL中的预编译语句作用域是session级，但我们可以通过max\_prepared\_stmt\_count变量来控制全局最大的存储的预编译语句。



#### 3、释放

使用{DEALLOCATE | DROP} PREPARE stmt\_name的语法进行释放。

deallocate prepare ins；

三、通过MySQL驱动进行预编译

以MySQL Java驱动Connector/J为例来介绍通过MySQL驱动进行预编译。

#### 1、客户端预编译

首先，简要提一下JDBC中java.sql.PreparedStatement是java.sql.Statement的子接口,它主要提供了无参数执行方法如executeQuery和executeUpdate等，以及大量形如set{Type}(int, {Type})形式的方法用于设置参数。



在Connector/J中，java.sql.connection的底层实现类为com.mysql.jdbc.JDBC4Connection，它的类层次结构如下图所示：



<https://www.cnblogs.com/micrari/p/7112781.html>