Mysql DATETIME 毫秒坑

jianzhangg 2025-01-16 600 阅读4分钟

今天写代码突发一个诡异的 bug，代码逻辑大概如下。

```java
// 1. 新增退款单记录
boolean save = shopOrderRefundService.save(shopOrderRefundAdd);

// 2. 调用京东退款
MiniappRefundResponse response = jdOrderOpenApiService.miniappRefund(...);
if (response.isSuccess()) {
    shopOrderRefundAdd.setStatus(ShopOrderRefundStatusEnum.SYNC_FAIL.getDatabaseCode());
    boolean update = shopOrderRefundService.updateByIdAndStatus(shopOrderRefundAdd);
    //返回失败
    ...
}

// 3. 更新退款单状态调用成功
shopOrderRefundAdd.setStatus(ShopOrderRefundStatusEnum.JD1001.getDatabaseCode());
boolean update = shopOrderRefundService.updateByIdAndStatus(shopOrderRefundAdd);
...
```

先生成退款单入库，再调京东接口，根据接口返回值再修改退款单状态。

问题是第三步修改的时候，有时成功有时失败。

本地跑了下，跟事物没关系。

```java
Creating a new SqlSession
SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@af21da9] was not registered for syn
JDBC Connection [org.apache.shardingsphere.driver.jdbc.core.connection.ShardingSphereConnection@546
Closing non transactional SqlSession [org.apache.ibatis.session.defaults.DefaultSqlSession@af21da9]
```

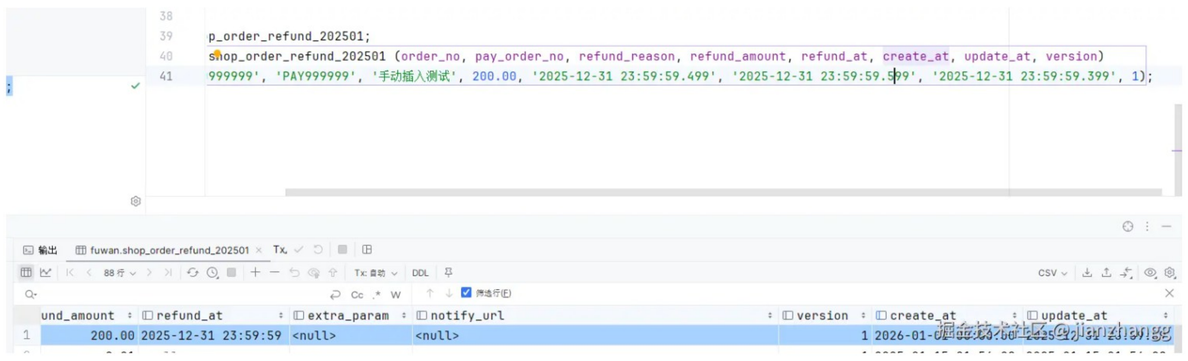
拉同事过来一起看，发现修改的代码有问题。

```java
public boolean updateByIdAndStatus(ShopOrderRefund shopOrderRefund, Integer status) {
    DateTime date = DateUtil.date();
    return update(shopOrderRefund,new LambdaUpdateWrapper<ShopOrderRefund>()
            .eq(ShopOrderRefund::getOrderNo,shopOrderRefund.getOrderNo())
            .eq(ShopOrderRefund::getStatus,status)
            .between(ShopOrderRefund::getCreateAt, DateUtil.offsetMonth(date, -2), date)
    );
}
```

copy 代码的时候忘把时间范围去掉，用单号查就没问题了。

但就算有时间范围，创建的时候肯定在修改前，为什么还是查不到呢?

又跑了几遍发现时间插入的问题。



注意看时间的毫秒，如果传入的SQL带毫秒，MySQL在入库的时候自动四舍五入了，这导致本来是 07.599 秒的数据变成了 08 秒。

但也不对，后面就算是 07.699，如果转成 08 也能查到。

我把更新 SQL 的查询部分单独拎出来看。

假设数据库里有只有这条数据。 order_no 是主键，create_at 有索引，create_at 是 datetime 类型，不带毫秒。

order_no	create_at
SR20250115215607789061	2025-01-15 21:56:08

jianzhangg LV.5
java @福禄
作者榜No.1 优秀作者
101 文章 221k 阅读 589 粉丝
关注 私信

相关推荐
Elasticsearch：Jira 连接器教程第二部...
38阅读 · 1点赞
kafka 的原理和应用
139阅读 · 0点赞
RocketMQ 安装使用
78阅读 · 1点赞
为什么 C++ 中需要内存分配器，而不能...
85阅读 · 0点赞
Spring MVC 数据绑定机制详解：@Mod...
120阅读 · 4点赞

精选内容
SkyWalking 10.1.0 实战：从零构建全链...
程序员小严 · 179阅读 · 3点赞
MySQL深度剖析-InnoDB索引与B+树
Anarkh_Lee · 64阅读 · 0点赞
【本地是好的，线上环境报错系列】代...
皮皮的江山 · 49阅读 · 0点赞
【DeepSeek版】JeecgBoot低代码 3.7....
JEECG低代码平台 · 97阅读 · 1点赞
AutoMQ 如何实现没有写性能劣化的极...
AutoMQ · 33阅读 · 0点赞

```sql
select order_no, create_at
from shop_order_refund_202501
where
    order_no = 'SR202501152156077890 61'
  and
    create_at BETWEEN '2025-01-15 21:56:07.974' AND '2025-01-15 21:56:07.974';
```

```sql
select order_no, create_at
from shop_order_refund_202501
where
    create_at BETWEEN '2025-01-15 21:56:07.274' AND '2025-01-15 21:56:07.974';
```

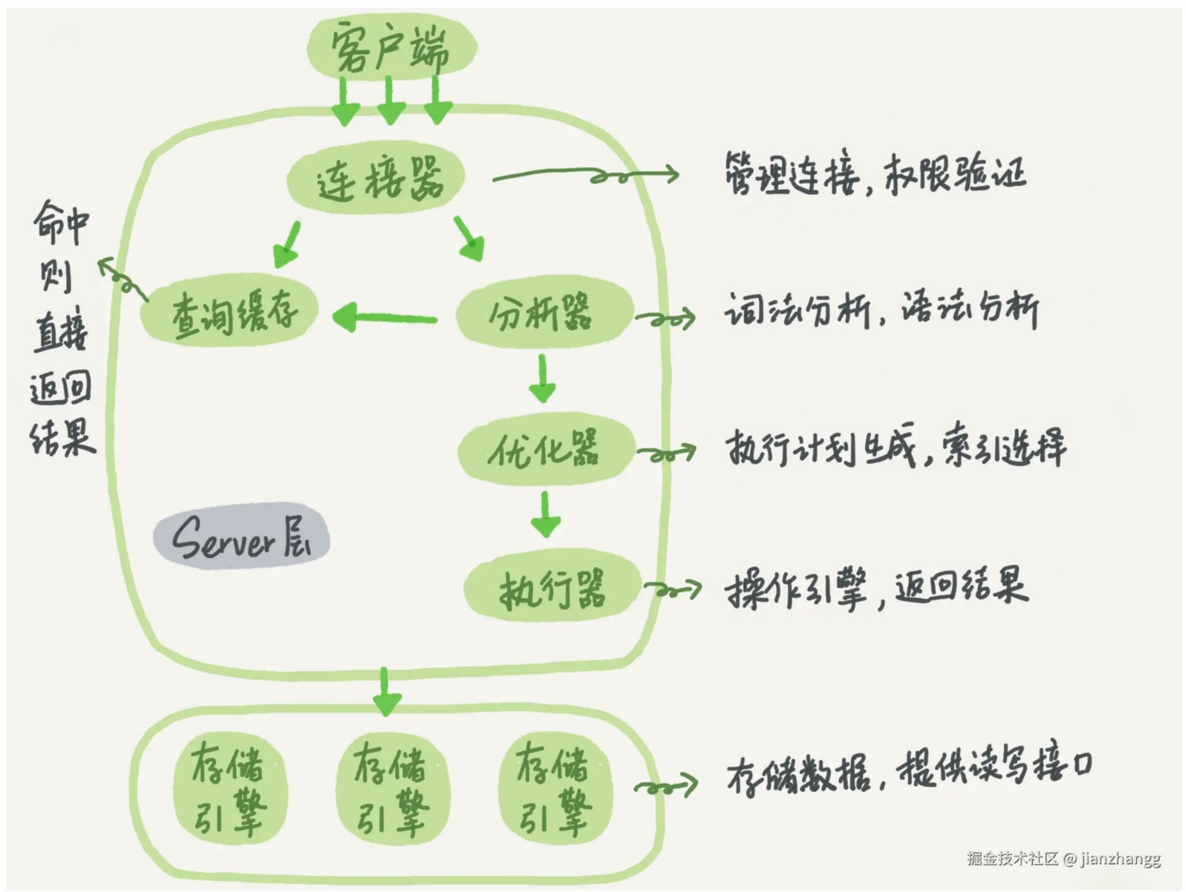
```sql
select order_no, create_at
from shop_order_refund_202501
where
    create_at BETWEEN '2025-01-15 21:56:07.974' AND '2025-01-15 21:56:07.974';
```

```sql
select order_no, create_at
from shop_order_refund_202501
where create_at = '2025-01-15 21:56:07.974' ;
```

```sql
select order_no, create_at
from shop_order_refund_202501
where
    order_no = 'SR20250115215607789061';
```

猜猜哪些 SQL 能查到数据?

答案是前两个查不到，后三个查得到。



这是查看 MySQL Server 层的 Trace 的 SQL。

```java
SET optimizer_trace = "enabled=on";
select order_no, create_at
from shop_order_refund_202501
where
    order_no = 'SR20250115215607789061'
  and
    create_at BETWEEN '2025-01-15 21:56:07.974' AND '2025-01-15 21:56:07.974';
SELECT *
FROM INFORMATION_SCHEMA.OPTIMIZER_TRACE;
SET optimizer_trace = "enabled=off";
SHOW SESSION VARIABLES LIKE 'optimizer_trace%';
SHOW GLOBAL VARIABLES LIKE 'optimizer_trace%';
```

Trace 很长我不贴代码了。通过 Trace 可以看到 MySQL 分析器、优化器、执行器操作逻辑 。

这里面关于时间的坑很多，一一说。

第一个为什么查不到?

优化器先通过 order_no 查询到这条数据，再在优化器中直接比较 "condition_value": false，因为 查出来 create_at 是 2025-01-15 21:56:08 != 2025-01-15 21:56:07.974。

优化器能识别毫秒，插入时候的四舍五入是执行器入库时候转的。优化器和执行器在必要的时候都会四舍五入，但在这种直接比较的场景没有转。

第二个为什么查不到?

优化器将这个范围查询执行做成了 "'2025-01-15 21:56:07' < create_at < '2025-01-15 21:56:08'"，自然就查不到了，<= 才查得到。

为了验证我试了各种范围 SQL，发现虽然优化器做了四舍五入，但在范围查询的时候，< 和 <=，> 和 >=，也根据毫秒做了区分。

第三个、第四个为什么查得到？ 分析器和优化器把第三个 SQL 优化成了第四个 SQL，由于是 = 查询，优化器和执行器都做了四舍五入成了 08 秒，所以查得到。

第五个直接走 id 查询自然查的出来。

以上是我的探索过程，很早前听过 MySQL DATETIME 有坑，我这就是个真实的案例了。

其实吧应该算自己对最底层了解的不够深刻，分析器、优化器、执行器的代码必然是非常复杂的，都是在踩坑中学习。

所以好一点的处理方式，要么换成时间戳，要么带毫秒，要么用字符串，根据业务选择吧。

方法论

这是 MySQL 隐式转换的问题，除了时间戳，还有其他场景。

1. 数字和字符串之间，不能解析为数字的字符串，会转成 0，包括查询、聚合函数场景，比较会转成。

mysql 代码解读 复制代码

```mysql
select *
from table where int_column = 'abc';

# 等效于
select *
from table where int_column = 0;
```

mysql 代码解读 复制代码

```mysql
select *
from table where varchar_column = 0;
# 会把所有不能转成数字的和能转成数字 0 的结果查出来。
```

2. boolean 的 true、false 和 int 1、0 互相转换。

今日份新技能 get，加油，共勉。

标签： MySQL 数据库 后端

评论 4

登录 / 注册 即可发布评论!

最热 | 最新

小白还菜 小白 @菜鸡互啄
排查时候巨诡异 时好时坏
1月前 点赞 评论 ···

小白还菜 小白 @菜鸡互啄
遇到过
1月前 点赞 评论 ···

用户175551021568
学到了，但是我想问下，“优化器能识别毫秒，插入时候的四舍五入是执行器入库时候转的。”这个逻辑您是如何发现的
1月前 点赞 1 ···

jianzhangg 作者 : trace 日志显示通过 timestamp 转的
1月前 2 回复 ···

为你推荐

一个诡异的MySQL查询超时问题，居然隐藏着存在了两年的BUG
CoderW | 3年前 | 2.1k | 8 | 2 | MySQL

MySQL 亿级数据分页的优化
编程学习网 | 3年前 | 908 | 3 | 评论 | MySQL

【惊天BUG】select xxx from 表 where value = 0；查询结果让大师兄都傻眼了
JavaDog程序狗 | 5月前 | 365 | 4 | 6 | MySQL | SQL | 数据库

聊聊当业务数据时间和预期的不一样，可以从哪些方向排查
linyb极客之路 | 3年前 | 292 | 点赞 | 评论 | MySQL | 后端 | 数据库

准线上事故之MySQL优化器索引选错
转转技术团队 | 11月前 | 2.0k | 11 | 评论 | MySQL | 后端

尊嘟假嘟？三行代码提升接口性能600倍
2YSP | 1年前 | 5.6k | 18 | 27 | 后端 | 性能优化 | MySQL

MySQL字段截断原理和源码分析
ZoneTials | 2年前 | 512 | 1 | 评论 | MySQL

thinkphp6里duplicate函数的奇葩实现
翟码农 | 7月前 | 176 | 1 | 评论 | ThinkP...

线上数据库死锁了！震惊
小红帽的大灰狼 | 8月前 | 1.3k | 5 | 1 | 后端

如何剔掉 sql 语句中的尾巴，我用 C# 苦思了五种办法
一线码农聊技术 | 4年前 | 256 | 2 | 评论 | C#

一个 MySQL 隐式转换的坑，差点把服务器整崩溃了
古时的风筝 | 2年前 | 6.9k | 42 | 14 | MySQL | 数据库 | 后端

一个烂分页，踩了三个坑！
why技术 | 1年前 | 23k | 194 | 59 | 后端 | Java | MySQL

不懂Mysql排序的特性，加班到12点，认了认了！
程序新视界 | 3年前 | 1.9k | 17 | 4 | MySQL

【番外：一个NestJS的后端从0到1】TypeORM+Mysql奇奇怪怪
蛋炒饭不加冰 | 8月前 | 132 | 点赞 | 评论 | NestJS | 后端

Mysql优化案例一：低效的SQL
Yideng | 1年前 | 5.0k | 9 | 3 | 后端