

为什么DBA怒吼：MySQL小数必须用decimal？float/double是隐藏的财务刺客！

原创 DB哥 DB哥 2025年08月06日 07:00 安徽

程序员小张自信满满地指着屏幕："看！MySQL 8.1的0.1+0.2=0.3！" DBA老王冷笑一声，默默创建了一张表.....

一、MySQL 8.1的"善意谎言"：常量计算的陷阱

1. 迷惑行为大赏

```
1  -- MySQL 8.1的"贴心" 优化
2  SELECT (0.1 + 0.2);           -- 显示0.3
3  SELECT (0.1 + 0.2) = 0.3;    -- 返回1 (true)
```

但这是假象！MySQL对常量表达式进行了优化处理，实际存储时：

```
1  -- 创建表暴露真相
2  CREATE TABLE dbbro_number_trap (
3    id INT PRIMARY KEY,
4    float_num FLOAT,
5    double_num DOUBLE
6  ) ENGINE=InnoDB;
7
8  INSERT INTO dbbro_number_trap VALUES
9  (1, 0.1, 0.1),
10 (2, 0.2, 0.2);
11
12 -- 震撼！真面目出现了！
13 SELECT SUM(float_num), SUM(double_num)
14 FROM dbbro_number_trap;
```

执行结果：

```
1  +-----+-----+
2  | SUM(float_num) | SUM(double_num) |
3  +-----+-----+
4  |      0.3000001 | 0.30000000000000004 |
5  +-----+-----+
```

二、解剖float/double的"诈骗"原理（附送财务毁灭指南）

1. IEEE 754的二进制骗局

- float（32位）：1位符号位 + 8位指数位 + 23位尾数位
- double（64位）：1位符号位 + 11位指数位 + 52位尾数位

十进制小数转二进制时的致命问题：

```
1  0.1 (10) = 0.00011001100110011... (2) ← 无限循环！
```

当用有限位存储时：

- float的23位尾数 → 精度约7位十进制
- double的52位尾数 → 精度约15位十进制

2. 财务毁灭现场直播

```
1  -- 创建死亡账单表
2  CREATE TABLE dbbro_financial_suicide (
3    id INT AUTO_INCREMENT PRIMARY KEY,
4    user_id VARCHAR(20) NOT NULL,
5    -- 致命选择：用double记账
6    balance DOUBLE(16,8) NOT NULL,
7    -- 救世主：decimal
8    safe_balance DECIMAL(16,8) NOT NULL
9  ) ENGINE=InnoDB DEFAULT CHARSET=utf8mb4;
10
11 -- 模拟用户交易
12 INSERT INTO dbbro_financial_suicide (user_id, balance, safe_balance)
13 VALUES
14 ('VIP_666', 100.00, 100.00),
15 ('VIP_777', 0.00000001, 0.00000001);
16
17 -- 执行死亡操作：重复扣款
18 UPDATE dbbro_financial_suicide
19 SET balance = balance - 0.00000001,
20     safe_balance = safe_balance - 0.00000001
21 WHERE user_id = 'VIP_777';
22
23 -- 查询结果：见证灾难
24 SELECT * FROM dbbro_financial_suicide
25 WHERE user_id = 'VIP_777';
```

输出结果：

1					
2	id	user_id	balance	safe_balance	
3					
4	2	VIP_777	-0.000000000000000000027105	0.00000000	
5					

三、decimal的圣光：为什么它是财务系统的救世主？🔥

1. 存储原理揭秘

decimal(M,D) 采用打包存储的BCD码（Binary-Coded Decimal）：

- 每4位二进制表示1位十进制（0-9）
- 9位数字打包成4字节存储
- 小数点位置固定

例如 decimal(16,8)：

存储值：12345678.87654321
实际存储：整数部分12345678 + 小数部分87654321

2. 精确计算实战

```
1  -- 创建安全交易表
2  CREATE TABLE dbbro_financial_savior (
3    id BIGINT AUTO_INCREMENT PRIMARY KEY,
4    tx_id CHAR(36) NOT NULL,
5    amount DECIMAL(36,18) NOT NULL -- 支持加密货币高精度
6  ) ENGINE=InnoDB;
7
8  -- 插入比特币交易
9  INSERT INTO dbbro_financial_savior (tx_id, amount)
10 VALUES
11 ('tx_01', 0.123456789012345678),
12 ('tx_02', 0.000000000000000001);
13
14 -- 完美聚合计算
15 SELECT SUM(amount) FROM dbbro_financial_savior;
```

输出结果：

```
1  +-----+
2  | SUM(amount) |
3  +-----+
4  | 0.123456789012345679 |
5  +-----+
```

四、深度技术对决：decimal vs float/double

特性	decimal	float/double
存储方式	打包BCD码	IEEE 754二进制浮点
精度保证	精确到指定位数	近似值
误差累积	无	累加运算指数级放大
适用场景	金融/财务/交易系统	科学计算/物理仿真
CPU计算成本	较高（需十进制转换）	极低（硬件直接支持）
空间占用	可变（每9位数字需4字节）	float 4字节/double 8字节
范围限制	最大65位数字	float ±3.4E38, double ±1.7E308
比较操作	精确比较	需范围比较

🔥 DBA血泪法则：

- 1. 钱、税、利率、汇率 → 必须用decimal
- 2. 用户积分、游戏金币 → 必须用decimal
- 3. 科学计算、地理坐标 → 可用float/double
- 4. 用float/double处理金钱 → 准备坐牢

谢谢大家的关注、点赞、分享！
如有疑问，可以留言，DB哥看到后会及时回复，也可以加DB哥微信交流



-- 加入「DB哥数据库帮」

DB哥微信：dbelder

🎁 DB哥数据库帮专属福利
▸ 授人以渔

关注DB哥微信公众号「DB哥」**免费学**DBA级MySQL视频课程【149课时】




DB哥


10年数据库救火队老炮 | 用实战教你少熬三年夜。遇到数据库别慌，DB哥专治数据库各种...
207篇原创内容


公众号


▸ 技术辅助

- 1

 **10**年数据库救火队老炮 | 用实战教你少熬三年夜
- 2

 亲手调优**3000**+故障库 | 企业级数据库架构
- 3

 库崩了？锁死了？**SQL**慢如🐢？CPU**100**%
- 4

 别慌，DB哥专治数据库各种“不调”！

▸ 背锅侠租赁

临时工小张随时待命：

```
1 UPDATE salary SET bonus =0;-- 小张干的！
```

帮规：

- 1

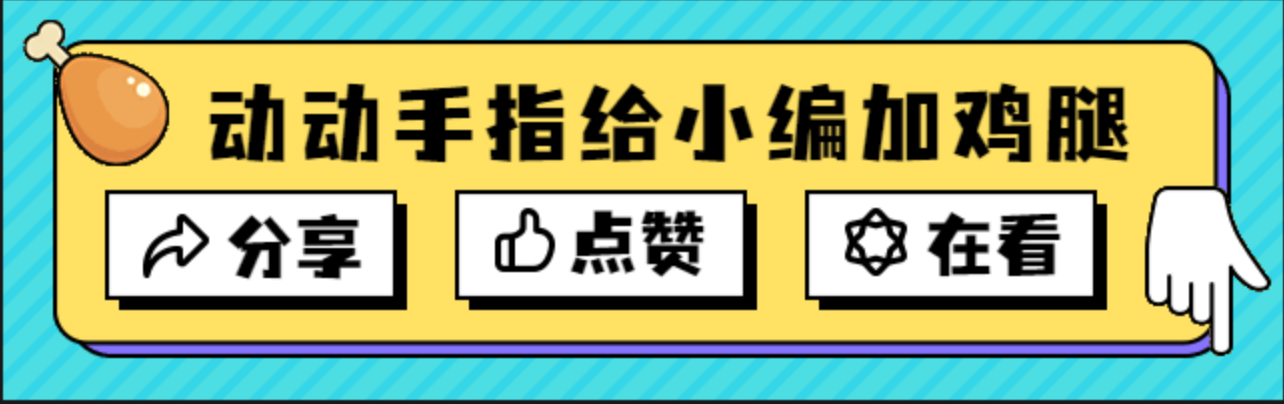
1.不准在生产环境执行**UPDATE**不带**WHERE**，否则罚用触控板代替鼠标一周
- 2

2.删库后不跑路，否则罚用Windows XP装 MySQL**5.0**（不兼容也要装）
- 3

3.必须用 **JOIN** 代替子查询，否则罚直播用子查询实现复杂报表（不许用**JOIN**！）
- 4

4.生产环境执行DDL必须测试，否则罚胸口碎大石（罪名：惊动监控告警）
- 5

5.不用**SELECT *** 横扫全表，否则罚罚抄《索引优化十诫》**100**遍（用毛笔写**SQL**语句）



DB哥

 喜欢作者

DB哥讲数据库 · 目录 ≡

< 上一篇

为什么DBA要求MySQL表索引不能超过5个

下一篇 >

MySQL大字段拆分：拯救你的数据库性能

