腾讯 TDSQL，数据异常检测工具发布

最近，腾讯 TDSQL 团队，新入职一位超级肥胖的小企鹅，名叫小胖。小胖的工作职责是研发数据库内核引擎。

小胖第一天工作，很谦虚，在企业微信中开始了告白：

非常荣幸能够加入鹅厂，非常荣幸加入 TDSQL 数据库团队，希望能和大家一起成长和进步！（小胖刚刚在群里发完消息，突然感觉有人走了过来）

小胖你好！我是你工作的导师，我叫六鹅，欢迎加入 TDSQL 项目组，以后和我一起学习数据库中事务处理的艺术吧！

好啊好啊！我一定和导师好好学习！（好棒呀，我的导师看起来很和善，感觉很厉害的样子，以后能跟着大佬学习技术啦）

我们出去喝一杯饮料，和你聊一聊哈，看看你对事务处理这块了解到啥程度了哈。（好紧张，怎么突然觉得脑袋一片空白，哭…）

六鹅：

今天呢，我想主要和你聊一聊数据库领域中由于事务并发所造成的数据异常相关的事情。不要紧张哈，这块内容是需要我们花时间花精力深入研究的，是一个长期的课题哦。我先问你一个问题哈，你知道哪些数据异常呢？

小胖：

我想想...

脏读？不可重复读？幻读？

在我印象里好像只有这三种...

六鹅：

还有吗，再想想

小胖：

课本里面好像确实只提到了这三个，啊啊啊，我想起来了，我准备面试的时候，看到过一篇文章，里面还提到了丢失更新和脏写！（我的脑子还好没有宕机...）

六鹅：

还不错哦，那你有听说过读偏序、写偏序、读写偏序、写读偏序嘛？

小胖：

这...是什么鬼，我晕了...

六鹅：

再接再厉，一次性问到终极问题了哦，**你知道世界上有多少个数据异常嘛**？

小胖：

世界上...导师饶了我吧...小胖才疏学浅，还望导师赐教！

六鹅：

哈哈，没事没事，我们慢慢学哈，我可以先告诉你答案哦，世界上的**数据异常个数是无穷多个**！

小胖：

啊？如果有那么那么多的异常，我们该怎么学习和研究呢...

六鹅：

不用担心哈，我们之前的理论已经将无穷多个数据异常分好类啦！**数据异常的数量是无穷多个，但是类别是有限个哦！**这个，教科书里可没有写哦，这是我们 TDSQL 的科研成果哦！

TDSQL 对事务处理的技术，进行了原创性工作。在我们的理论中，将数据异常定义为：

并发事务的一个 History 中，如存在一个依据冲突可串行化技术的有向环，则称为数据异常。这个定义，可是第一个对数据异常这个整体概念下定义的哦，之前的文献讨论数据异常，都是 case by case 的方式一个个讨论每一个具体的数据异常如前面你提到的脏读等，这样的方式，缺乏全局观，不利于认识数据异常，也不利于掌握并发访问控制算法哈！

而数据异常可分为如下三类：

写异常（WAT）：环中有写写偏序。

读异常（RAT）：环中有一个或者多个写读偏序，且不包含写写偏序。

交叉异常（IAT）：除了写异常和读异常之外的异常。

更为细致的分类如下：

单元数据异常（SDA）：数据异常发生在两个事务一个变量上。

双元数据异常（DDA）：数据异常发生在两个事务两个变量上。

多元数据异常（MDA）：除了 SDA 和 DDA 以外的数据异常。

小胖：

这...让我想想...有些专有名词不太懂呀，环是个什么东东？偏序是个啥？变量又是指什么呢？

六鹅：

来这里，我在白板上给你画一画哈。我们这里环的全称是**冲突环图（带有冲突操作的有向环图）**，图中的每个顶点是一个事务，而每一条有向边就是边的两个事务顶点的偏序关系。下面我们通过一个例子来说明哈。

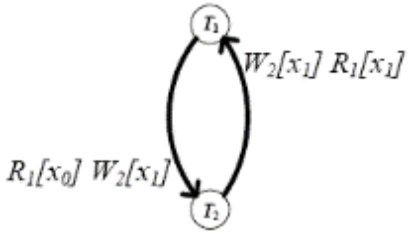

图 1 不可重复读

所图 1 所示，展示了不可重复读数据异常的冲突环图。首先给出冲突操作的定义哈，满足如下三个条件，才能称之为冲突操作：

- 必须在不同的事务中
- 操作同一个数据项
- 其中之一必须是写操作

我们来观察一下上图，从 T1 到 T2 的边是对 x 的先读后写（T1 先读，T2 后写），在不同的事务中，操作同一个数据项（我们也称之为变量），且有写操作，属于冲突操作，我们把 T1 到 T2 的冲突操作也称之为 T1 到 T2 的读写偏序。同理，我们把 T2 到 T1 的冲突操作称之为 T2 到 T1 的写读偏序。两条冲突有向边加两个事务顶点构成了环，按照我们的理

论，只要构成了冲突环图，异常则必然发生。异常发生后，我们又按照环中存在的偏序类型，对异常进行了分类，就是我们上边说的分类方法啦。

小胖：

哇哦，我好像明白一些了，感觉好厉害的样子！

亲爱的导师，我想赶紧学习一下！有没有什么学习文档呀？

六鹅：

文档当然有啦，但是还有更好的学习工具哟！

我们已经开发好了快速学习数据异常的小工具！

它拥有如下特性：

a) 支持 Linux 平台

b) 支持数据异常精准识别

c) 支持多种数据异常检测算法

d) 用户友好的交互式界面

小胖：

太棒了！可是。。linux 我不太熟怎么办。。。

六鹅：

没关系，我会给你提供全面的帮助！按照步骤执行即可哦！

立刻传送体验的最佳姿势！

来，手把手教你在 linux 平台进行源码编译，获得可执行二进制：

a) 创建并进入项目根目录

```
mkdir 3TS

cd 3TS
```

b) 拉取源代码，注意这是 TDSQL 的开源项目哦！

```
wget https://github.com/Tencent/3TS/archive/refs/heads/master.zip
```

c) 解压文件

unzip master.zip

d) 进入源代码根路径

cd 3TS-master

e) 编译二进制。由于该程序基于 C++17 开发，需要编译器支持哦

./make.sh

f) 编译成功后，可以看到名为 3TS-DAI 的二进制哦。进入小程序的运行界面

./3TS-DAI

若提示“g++: error: unrecognized command line option ‘-std=c++17’”，则可尝试下载最新版 gcc，gcc 官网：https://gcc.gnu.org/（g++8 以上的版本才会支持 C++17 哦）

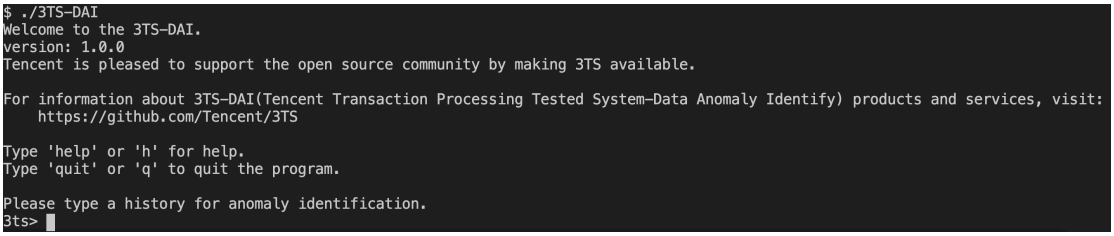

图 2 3TS-DAI 登陆界面

小胖：

历经千辛万苦，我终于进入程序啦！

六鹅：

你真棒哦！下面开始我们的体验之旅吧！

我们先来体验一下，小程序里面的命令吧~

进入系统后，输入 help 或者 h 查看帮助信息

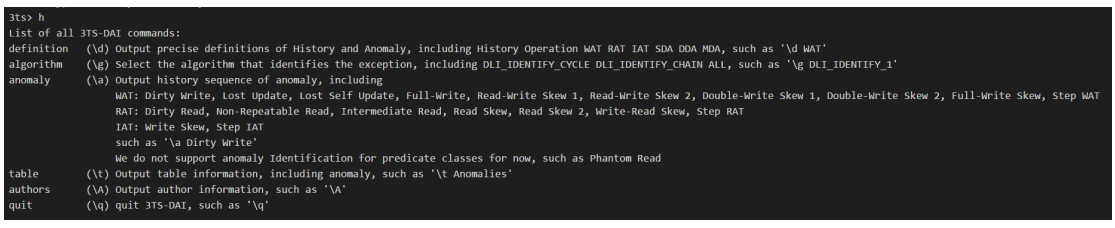

图 3 帮助信息

通过输入‘\d [definition]’ 查看相关名词定义，你可以输入\d History 看看效果哦

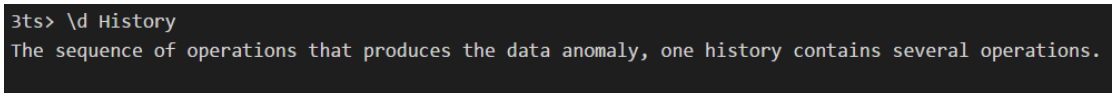

图 4 名词解释

通过输入‘\a [anomaly name]’查看数据异常信息，你可以输入\a Dirty Read 看看效果哦

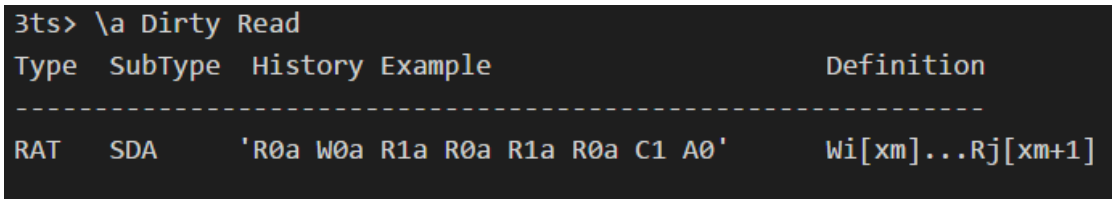

图 5 数据异常用例信息

通过输入‘\g [algorithms]’可以设置当前使用的数据异常检测算法哦，程序默认的算法为 DLI_IDENTIFY_CYCLE，你可以输入‘\g DLI_IDENTIFY_CYCLE, DLI_IDENTIFY_CHAIN’来指定多个算法进行比较哦

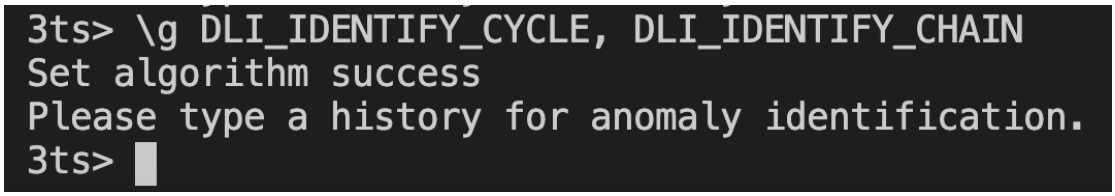

图 6 设置检测算法

下面要进入正题喽！如何进行异常识别呢？

我们的小程序主要通过输入 History 来识别数据异常哦。‘History’是产生数据异常的操作序列，一个‘History’包含多个‘Operation’。一个 Operation 包含三个字符，例如 ROx，分别为操作类型、事务编号以及数据项。操作类型包括：R：Read W: Write C: Commit A: Abort。事务编号包括：0，1，2...（必须是 10 以内的数字）。数据项包括：a，b，c...（必须是小写字母）。这个 Operation 的具体含义是事务 0 读取了 x 数据项的值。下面我们来构造 History 吧。

以丢失更新为例，需要输入的 History 长这个样子：‘R0a R1a W0a R0a W0a W1a A1 C0’。也可以通过输入\a Lost Update 获取到 History 信息哦。

我们可以按照上述理论，画出该异常的冲突环图哈：

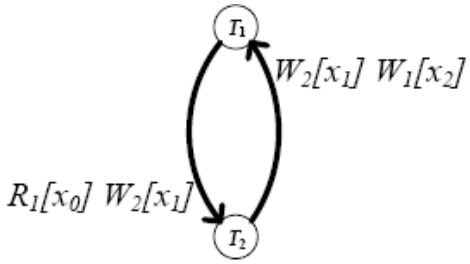

图 7 丢失更新

然后，我们只需要将示例 History 输入小程序，小程序就会返回数据异常的信息啦！

```
3ts> R0a R1a W0a R0a W0a W1a A1 C0
DLI_IDENTIFY_CYCLE's Identification Result:
Anomaly Type: WAT
Anomaly SubType: SDA
Anomaly Name: Lost Update
Anomaly Format: Ri[xm]...Wj[xm+1]...Wi[xm+2]
```

图 8 数据异常检测结果

怎么样，学会了吗，赶紧动手试一试吧！

小胖：

谢谢师傅哈！我还有一个问题：这个小程序，背后是不是有啥理论支撑哦？

六鹅：

哈哈哈，小胖，你真聪明。
鹅厂 TDSQL 出品的技术，都是有理论基础的，要做到有理有据。我们这个小程序背后，是鹅厂在事务处理技术的重要创新，里面干货多多，你快先挖掘感受下吧，下次咱们具体展开聊聊里面的核心技术哦！

	比较项	传统理论	新方法
		可串行化 / 冲突可串行化	**数据异常体系化研究**
1	发现有新数据异常	否（不同人报告不同的但个数有限的异常）	发现10+新异常，证明异常是无穷多的
2	描述了所有数据异常	否	是
3	有统一的数据异常描述方法	Case by case / 半系统化（+个例）	是（描述方法灵活，可裁剪）
4	有科学的描述数据异常的方法	无	基于数学，形式化描述
5	统一描述谓词类、非谓词类数据异常	否	是
6	可否开展量化研究	否	是（回答了数据异常的个数）
7	数据异常是否可科学分类	否 / 半系统化分类（+个例）	是（精准可量化）
8	一致性定义（ACID的C）	不明确，难理解	形式化定义，易理解
9	数据异常使用范围（区分单机、分布式系统、主从系统）	是	否
10	对隔离级别的理解	弱隔离级别可提高并发效率	破除冲突观念，灵活定义隔离级别，并提高并发算法效率
11	设计新的并发算法	Case by case	可指导新算法设计
12	算法对于回滚率的研究	回滚率高，且不能量化研究	回滚率低，且能量化研究

六鹅：

小胖，再告诉你一个秘密！我们的程序里面有彩蛋！看看你能不能找到哈