数据库管理系统测试报告

1. 引言

本报告旨在对数据库管理系统——达梦数据库8.0（简称 DM8 ）进行测试分析，通过使用 3TS Coo 一致性检查工具，我们将对这款数据库在不同隔离级别下的行为进行测试，揭示它在处理并发事务时的优缺点。

在引言部分，我们将简要介绍数据库管理系统的基本概念和功能特点，随后详细阐述 DM8 的特性。接着，我们将介绍 3TS Coo 工具的功能及其在一致性检查中的应用，以及我们所做测试进行的修改。报告的后续部分将涵盖环境配置、测试方法及结果分析，最后总结测试结果。

2. 背景知识

2.1 数据库管理系统概述

数据库管理系统（DBMS）是用于创建、管理和操作数据库的软件。它们提供了一系列功能和特点，主要包括：

- **数据存储与管理**：提供有效的数据存储机制，让用户可以方便地插入、更新、删除和查询数据。
- **数据安全**：通过用户权限管理和加密等措施，保证数据的安全性。
- **数据完整性**：通过主键、外键、唯一约束等机制，保证数据的一致性和完整性。
- **事务管理**：支持原子性、一致性、隔离性和持久性（ACID）原则，确保事务处理的可靠性。
- **并发控制**：支持多个用户同时访问和操作数据库，实现并发控制。
- **备份与恢复**：提供数据库备份和恢复功能，以保护数据免受灾难性损失。

2.2 选择的数据库系统的特点

DM8

DM8是达梦公司在总结DM系列产品研发与应用经验的基础上，坚持开放创新、简洁实用的理念，推出的新一代自研数据库。DM8吸收借鉴当前先进新技术思想与主流数据库产品的优点，融合了分布式、弹性计算与云计算的优势，对灵活性、易用性、可靠性、高安全性等方面进行了大规模改进，多样化架构充分满足不同场景需求，支持超大规模并发事务处理和事务-分析混合型业务处理，动态分配计算资源，实现更精细化的资源利用、更低成本的投入。一个数据库，满足用户多种需求，让用户能更加专注于业务发展。

2.3 3TS Coo 一致性检查工具

3TS Coo 是一个用于事务数据库的一致性检查工具。其特点包括：

- **准确性**：能够识别所有类型的异常。

- **用户友好性**：基于 SQL 的测试，易于使用。
- **经济高效性**：能够在几分钟内完成一次检查。

3TS Coo 通过检测数据库在不同隔离级别下的行为，识别异常与一致性问题。结果行为分为两类：

1. **异常（Anomaly）**：数据库无法识别数据异常，导致数据不一致。
2. **一致性（Consistency）**：数据库通过具有可序列化结果的异常测试用例（无 POP 周期），确保数据保持一致性。

3TS Coo 支持以下隔离级别：

- **可序列化（SER）**
- **可重复读取（RR）**
- **已提交读取（RC）**
- **未提交读取（RU）**
- **快照隔离（SI）**

3. 环境配置

3.1 硬件环境

- 操作系统: macOS Monterey 12.7.1
- CPU: Apple M1
- 内存: 16 GB RAM
- 存储: 1T SSD

3.2 软件环境

- Docker
- 数据库管理系统:
  - DM8
- 虚拟机:
  - Ubuntu 20.04 x86

3.3 安装步骤

步骤1：安装编译环境

```
sudo apt install build-essential cmake
```

步骤2：安装 unixODBC

在终端中执行以下命令安装 unixODBC：

```
sudo apt update
sudo apt install unixodbc unixodbc-dev
odbcinst -j   # 检验安装是否成功
```

步骤3：下载 DM8 Docker 镜像并安装

访问达梦数据库官网，在下载中心下载 DM8 Docker 镜像。按照DM 数据库 Docker 安装教程的指导，完成安装并启动 DM8。

步骤4：安装 DM8 ODBC 驱动

DM Docker 安装成功后，拷贝 `/opt/dmdbms/bin/libdodbc.so` ODBC 驱动到本地环境目录。

```
[DM8 ODBC DRIVER]
Description = ODBC DRIVER FOR DM8
Driver = your_driver_path
```

步骤5：配置 ODBC 数据源

编辑 `/etc/odbc.ini` 文件，添加以下内容：

```
[DM8]
Description = DM ODBC DSN
Driver = DM8 ODBC DRIVER
SERVER = localhost
UID = SYSDBA
PWD = SYSDBA001
TCP_PORT = 5236
```

步骤6：测试连接

使用 `isql` 命令测试连接：

```
root@766abb807578:~# isql -v DM8
+---------------------------------------+
| Connected!                            |
|                                       |
| sql-statement                         |
| help [tablename]                      |
| quit                                  |
|                                       |
+---------------------------------------+
SQL>
```

4. 运行方法

4.1 修改源代码

根据：

- DM8 不支持 BEGIN SQL，在用户连接开启会话或事务提交或回滚之后的执行第一条增删改 SQL 时，会自动开启新事务。
- DM8 在执行 DDL 语句时会把之前的事务提交，并开启一个新事务来执行，会导致隔离级别设置失效。
- DM8 的隔离级别设置是在单个事务内有效，需要在每个事务开始时执行 `SET TRANSACTION ISOLATION LEVEL <serializable>` 来设置使隔离级别设置生效。
- DM8 默认情况下，事务冲突发生回滚仅回滚单条 SQL，事务回滚需要用户输入 "Rollback"。
- DM8 默认情况下，事务间发生冲突，在阻塞状态下的事务会保持阻塞，需要对导致阻塞的事务传入 "Rollback" 命令进行回滚，否则 `3TS Coo` 程序无法进行下去。

修改 `sqltest.cc` 、 `sql_cntl.cc` 等文件来支持 DM8 数据库的测试。

4.2 编译源代码

在 `3TS/src/dbtest` 目录下执行以下命令编译 3TS Coo：

```
cmake -S ./
make
```

4.3 执行测试

静态测试

包含 33 个测试用例，其测试文件在 p 文件夹下，不需要生成测试文件。源文件包括：common.cc、case_cntl.cc、sql_cntl.cc、sqltest.cc 以及对应的头文件，生成的可执行文件名为 3ts_dbtest。

动态测试

需要通过 python 脚本文件动态生成测试文件。源文件包括:common.cc、case_cntl_v2.cc、sql_cntl_v2.cc、sqltest_v2.cc，生成的可执行文件名为 3ts_dbtest_v2。

动态测试中测试样例由 python 脚本文件生成，包含 random_do_list.py、 mda_detect.py、mda_generate.py 文件。其中 random_do_list.py 用于随机生成测试文件列表；mda_generate.py 用于生成指定格式测试样例；mda_detect.py: 可能用于检测测试结果是否出现异常现象。

测试脚本

`auto_test.sh` 是一个数据库测试工具的调用脚本。它根据传入的数据库类型和隔离级别参数，执行相应的数据库测试。添加需要测试的数据库和执行的测试类型，并填写正确的数据库连接信息（例如：**-user** 和 **-passwd**）。

`auto_test_all.sh` 是一个用于自动化测试不同数据库隔离级别的脚本。它通过调用 `auto_test.sh`，对多种数据库类型和隔离级别进行测试。我们可以注释掉不需要执行的测试命令，然后执行 `auto_test_all.sh` 脚本。

```
./auto_test_all.sh
```

5. 测试 DM8

5.1 测试结果

执行 `3TS Coo` 程序对 DM8 数据库进行静态和动态测试，并对产生的数据库文件夹下的 `result_summary` 文件夹内各个隔离级别的结果通过附件 `3TS_summary.py` 脚本进行处理，得到 `DM8_static.xlsx`、`DM8_dynamic.xlsx` 等表格文件。

5.2 测试结果分析

对 `DM8_static.xlsx` 表格文件进行分析可得：

Read Uncommitted：

- 允许大多数异常发生，这符合预期，因为这是最低的隔离级别。

Read Committed：

- `Read Uncommitted` 中发生异常的测试用例一部分显示为 "Pass"，符合预期。

Repeatable Read：

- 发生异常的用例进一步减少，大多数用显示 "Pass" 或者 "Rollback"，保证了数据的一致性。

Serializable：

- 与 `Repeatable Read` 一致。

对 `DM8_dynamic.xlsx` 表格文件进行分析可得，发现大多数测试发生了回滚，而有一部分 finish。

5.3 总结

DM8 在 `Read Uncommitted`、`Read Committed` 隔离级别与 MySQL相似，但是 `Repeatable Read` 隔离级别有着更好的表现，`Serializable` 隔离级别下，当一个串行化事务试图更新或删除数据时，如果这些数据在此事务开始后被其他事务修改并提交时，DM8 数据库将报“串行化事务被打断”错误，而对于读事务则不会报错，允许了一部分异常的发生。

6. 参考文献

1. 3TS: Coo 一致性检查 - GitHub
2. 3TS 文档 - Axing Guchen
3. Docker安装 | 达梦技术文档 - DM
4. 管理事务 | 达梦技术文档 - DM