MQ结构

[1. 前言 1](#_Toc1909)

[2. 概览 2](#_Toc3961)

[3. 文件存储 2](#_Toc31617)

[3.1. 文件服务 2](#_Toc25503)

[3.2. 存储模型 2](#_Toc15699)

[3.3. 缓存服务 3](#_Toc30206)

[4. 表级服务 3](#_Toc18139)

[5. SQL解析 3](#_Toc16147)

[6. WAL 4](#_Toc17324)

[7. 事务 4](#_Toc9487)

[8. 执行计划 4](#_Toc20113)

[9. 结束语 4](#_Toc22400)

[10. 附件 4](#_Toc8451)

# 前言

2017年下半年，因工作原因利用业余时间对数据库学习研究的较多。偶然在github上找到了一个Java版的数据库（教学用途），代码简洁清晰，仔细阅读了其代码后，将其核心技术点记录下来。

# 概览

此数据库的核心模块包括SQL解析，表管理，事务管理，WAL和存储服务。



# 文件存储

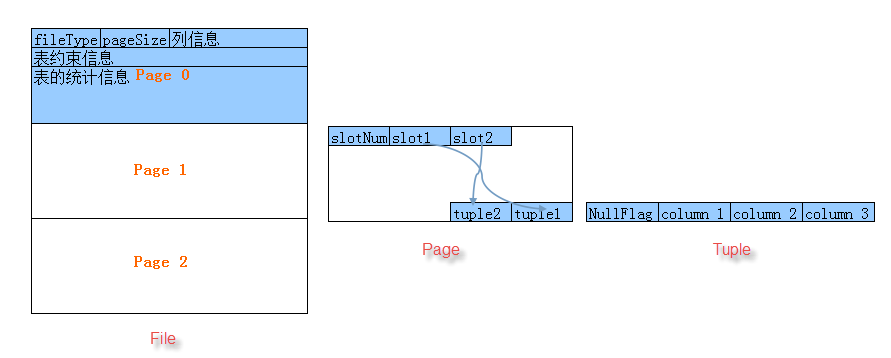
## 文件服务

提供DBFile的创建，加载，保存，删除。DBFile类包含File属性和RandomAccessFile属性，上层服务通过DBFile的RandomAccessFile来操作文件，包含数据的读取，写入和同步到磁盘。

## 存储模型

一个DBFile对象对应一个文件，DBFile由DBPage组成，每页大小8kb。

使用HeapFile来保存数据。一张表对应一个DBFile，首页(page 0)用来记录文件类型，页大小，表的结构信息和统计信息。从page 1开始的页都用于存储表数据。Page页的前2Byte用于存储slot总数，每个slot存放对应的tuple在本页内的偏移量。Slot从前往后顺序存储，tuple从页尾开始往前堆积。Tuple中存储了表中的一条记录的值。如下图中的tuple存储了该记录column1,2,3列的值。每列在NullFlag中用一个bit表示，若某列为Null则将此bit置为1，后面就不用再为此列开辟空间了。



## 缓存服务

通过缓存服务减少磁盘IO次数。缓存服务提供API将脏页同步到磁盘，保证处在session期内的page不会被移除到缓存。

缓存服务采用LRU算法(Least recently used，最近最少使用)进行page的管理。通过将LinkedHashMap的accessOrder属性置为true来实现LRU。

# 表级服务

表服务提供tuple的增删改查，通过调用存储服务实现此部分的功能。

# SQL解析

SQL解析模块负责解析符合SQL语法的输入，形成各种Command对象。Command调用相关服务实现其对应功能。

SQL解析采用ANTLR框架实现。ANTLR(Another Tool for Language Recognition)可以根据预先定义好的语法词法规则，对当前的输入生成对应的语法树。

# 事务

WAL(Write Ahead Log)， 因日志文件是顺序写的，WAL 可以显著减少磁盘写的次数。每次在修改数据页后，需要记录WAL。

# 执行计划

# 结束语

文章的结束语部分，一般是对文章内容或技术的总结，或其它启发性思考等。

# 附件

<https://github.com/williamxww/columndb-demo>