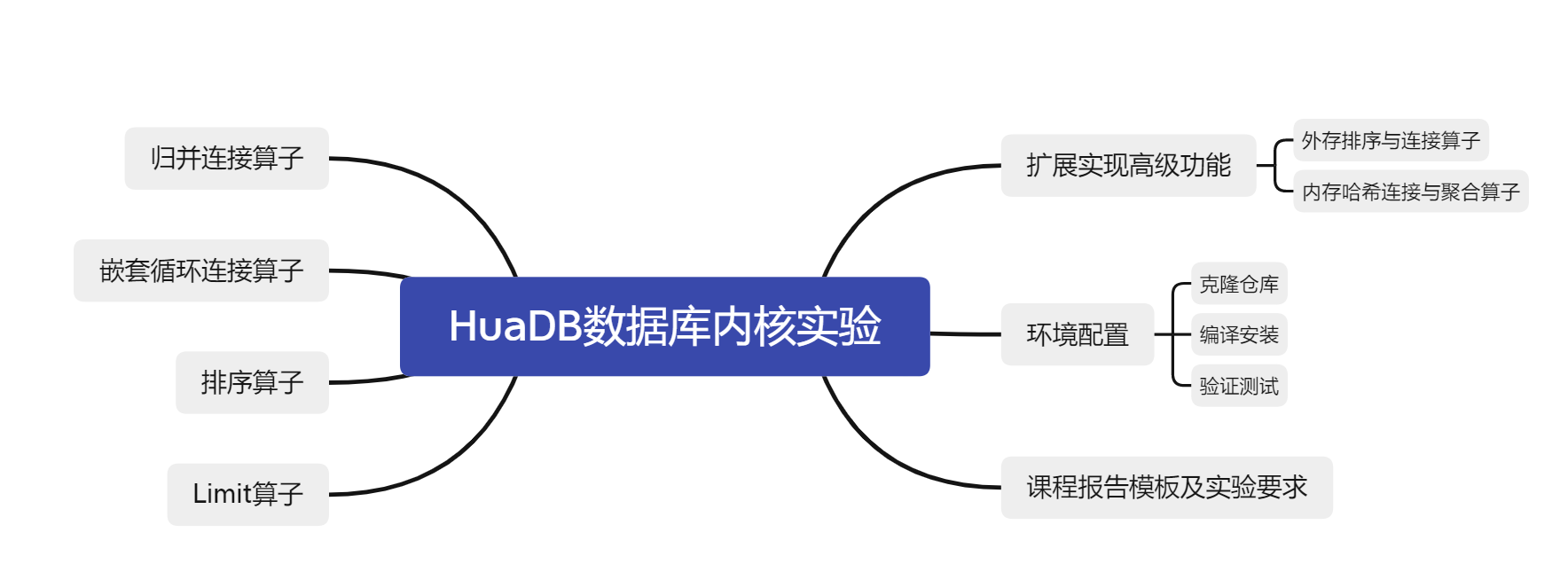
实验九：

HuaDB数据库内核实验



本实验概览图

# HuaDB数据库内核实验手册

* 1. 实验介绍
     1. 关于本实验

本次实验为数据库系统查询处理的实验，意图通过实现基于火山模型的执行器引擎来更好地理解数据库系统如何按照查询算子逐级处理查询请求。

执行器是数据库系统将磁盘存储的页面数据处理为符合查询所需结果的模块。火山模型是一种经典的拉取式执行器模型，通过抽象算子的执行逻辑，上层算子逐层从下层算子拉取数据，算子内部完成处理后响应上层的拉取请求。通过这种方式可以在抽象结构上构建一种结构上非常简洁的执行器模型，便于在执行引擎上实现查询优化和异常处理。

算子的实现是本次实验的难点。火山模型基于策略模式可以设计出不同的算子结构，并结合工厂模式可以快速创建不同类型的算子。系统根据查询计划树逐层生成算子，并调用算子的拉取函数完成查询的实际执行。部分算子在内部处理过程中需要存储中间结果来加速算子执行，例如连接算子以及聚合算子等，这种情况一般会使用内存作为中间结果的缓存。但是，部分算子在执行过程中将产生巨大存储容量的中间结果，不能够简单地存储到内存中（例如大表间的连接运算）。这类算子需要引入磁盘作为外部存储，一般称为外存算子，其中包括外存连接和外存排序算子。外存算子的执行效率对于磁盘 IO 的开销敏感，在实现上相较于内存算子更为复杂。本次实验要求补全几个常见的算子：连接、排序、Limit、聚合。考虑到不同算子本身实现难度的差异，基础功能中要求完成内存中的 NestedLoopJoin、MergeJoin 两类连接算子、排序算子以及 Limit 算子，高级功能在此基础上要求中引入内存资源的限制，并设计相应运算的外存算子，同时也可以选择实现难度较高的哈希连接算子以及聚合算子。图1为数据库系统整体层次划分图，左侧为功能示意图，右侧为层次架构图。

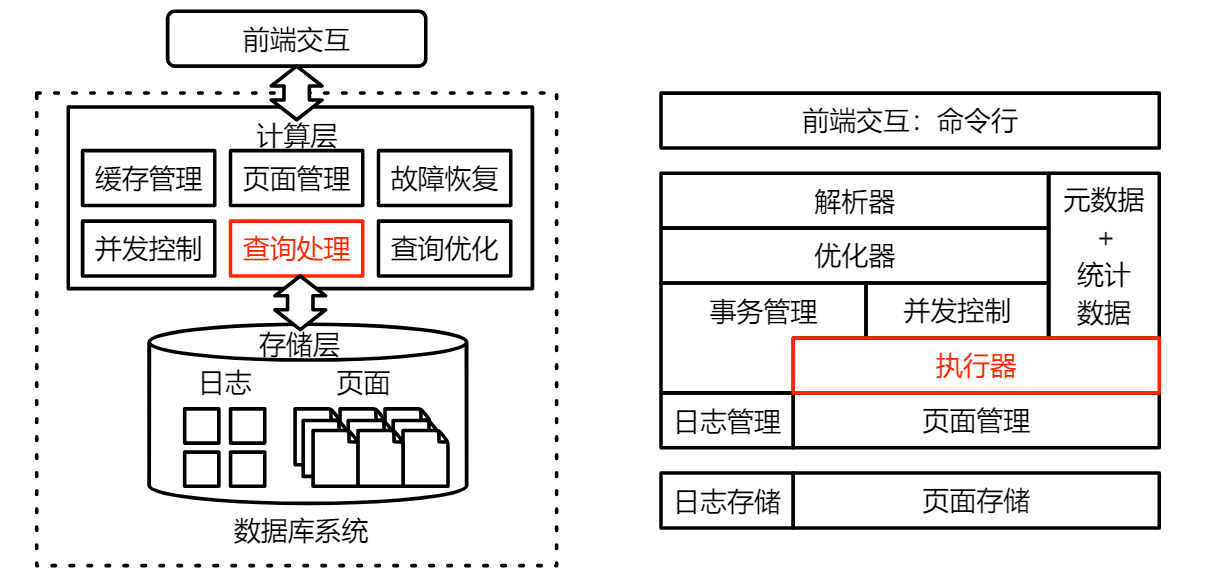


图1 数据库系统整体层次划分图

* + 1. 实验目标

**本次实验要求完成如下基础功能：**

* 实现 Limit 算子。
* 实现内存排序算子。
* 实现内存连接算子：实现嵌套循环连接和归并连接两类连接算子。

**（扩展）在基础功能之上，实验框架支持完成以下高级功能：**

* 外存排序和连接算子：在内存排序和连接算子的基础上添加内存限制，使用外存储存中间结果，并设计内外存数据交互机制加速外存算法处理。
* 内存哈希连接算子和聚合算子：实现内存存储的哈希表，并基于哈希表实现内存哈希连接算子以及聚合算子，中间结果利用哈希表保存。
  1. 实现查询处理功能概述

本次实验的目标比较明确，每次任务实现一个算子的 Next 函数，正确实现后便可通过该算子对应的测试。因此本次实验不再给出详细的实验步骤，你只需按照实验任务说明完成相应的算子即可。

基础功能中所有算子仅要求在内存中实现即可，如果你对外存算子的实现感兴趣，可以参考本次实验的高级功能。

* + 1. Limit算子

Limit 算子是本次实验最基础的一个任务，算子本身的实现非常简单，希望你通过这个任务熟悉实验框架中火山模型的运作流程。

你需要补充 limit\_executor.cpp 中的 Next 函数，实现过程中你可能会用到对应的 LimitOperator 中的成员变量。你也可以为 LimitExecutor 增加新的成员变量来存储信息。

完成后你将通过10-limit.test测例。需要注意的是，为确保 limit 结果的唯一性，在该测例中使用了 order by 语句，而排序算子的实现是任务 2 的内容。但是该测例中所有排序均与插入顺序一致，因此只要你在 lab 1 中的实现可以保证读取顺序与插入顺序保持一致，便可以通过本测例。如果你的实现不能保证这一点，你可以在完成任务2后再来进行本测例的测试。

* + 1. 排序算子

你需要补充 orderby\_executor.cpp 中的 Next 函数，与 Limit 算子类似，你可能会使用到 OrderByOperator 中的成员变量，从中获取排序的列以及排序的类型（升序还是降序），实现过程中可以使用 STL 的 sort 函数。

正确实现后将通过20-sort.test测例。

* + 1. 嵌套循环连接算子

你需要补充 nested\_loop\_join\_executor.cpp 中的 Next 函数，从 NestedLoopJoinOperator 中获取连接条件，本次实验中仅要求实现内连接即可。

正确实现后将通过30-nested-loop-join.test测例。

* + 1. 归并连接算子

你需要补充 merge\_join\_executor.cpp 中的Next函数，从 MergeJoinOperator 中获取连接条件，本次实验中仅要求实现内连接即可，注意可能需要对join两侧的表存在重复值的情况特殊处理。

在查询计划生成时，已对归并连接的两个子节点添加了OrderBy节点进行排序操作，你在实现归并连接时无需再进行排序。

正确实现后将通过40-merge-join.test测例。

* + 1. 扩展高级功能

**（1）任务 1：外存排序与连接算子**

【实验描述】

基础功能中完成了内存中的排序和连接算子的实现，高级功能中要求在两类算子上添加内存使用限制，并设计基于外存的排序和连接算子以及内外存数据交互机制。

【实现思路】

步骤 0：阅读教材相关算法的伪代码，理解内存算法和外存算法的实现差异以及查询代价差异。

步骤 1：拓展 BufferPool 功能，添加对于查询算子使用 BufferPool 的支持。

步骤 2：修改查询算子的 Next 函数实现，并基于步骤 1 种拓展的 BufferPool 接口，通过缓存池实现内外存数据交互。

**（2）任务 2：基于版本链的垃圾回收**

【实验描述】

基础功能中完成了几种难度适中的算子的实现，高级功能中要求在此基础完成哈希表相关功能，并基于此完成哈希连接算子以及聚合算子。

【实现思路】

步骤 1：设计哈希表，实现记录字段的哈希功能。

步骤 2：基于哈希表和哈希函数实现内存上的哈希连接算子以及聚合算子。

1.2.6 代码结构

实验框架主要包含以下几个模块，加粗部分为实验中涉及到的，需要你来补充的模块：

* binder: 语义解析模块
* catalog: 系统表模块
* common: 工具模块，包含字符串处理函数、异常相关类等
* database: 数据库引擎
* **executors: 查询执行模块**
* **log: 日志模块**
* operators: 查询计划树节点
* **optimizer: 优化器**
* planner: 查询计划生成模块
* **storage: 存储模块**
* **table: 表相关类及函数**
* **transaction: 事务模块**
  + 1. 相关代码模块

本次实验涉及到代码中如下的功能模块：

|  |
| --- |
| * [excutors](https://github.com/thu-db/huadb/blob/main/src/executors)：执行器相关类   + [executor](https://github.com/thu-db/huadb/blob/main/src/executors/executor.h)：执行器算子的抽象模板，用于衍生出各类实际算子。   + [limit\_executor](https://github.com/thu-db/huadb/blob/main/src/executors/limit_executor.h)：Limit 算子，用于限制输出数量。   + [orderby\_executor](https://github.com/thu-db/huadb/blob/main/src/executors/orderby_executor)：排序算子，用于实现内存或外存的记录排序算法。   + [nested\_loop\_join\_executor](https://github.com/thu-db/huadb/blob/main/src/executors/nested_loop_join_executor.h)：Nested Loop 连接算子，用于实现最基础的连接算法。   + [merge\_join\_executor](https://github.com/thu-db/huadb/blob/main/src/executors/merge_join_executor.h)：Merge 连接算子，要求底层算子节点已经完成排序，一种快速的连接算法。   + [hash\_join\_executor](https://github.com/thu-db/huadb/blob/main/src/executors/hash_join_executor.h)：（高级功能）哈希连接算子，需要补充哈希组件后实现，一种快速高效的连接算法。   + [aggregate\_executor](https://github.com/thu-db/huadb/blob/main/src/executors/aggregate_executor.h)：（高级功能）聚合算子，依赖于哈希表结构，用于实现各类聚合算子的运算。 |

基础功能需要补充约 200 行代码，本次实验依赖于实验 1，请确保完成以上任务再开始本次实验。

相关功能模块的抽象示意图如图2：

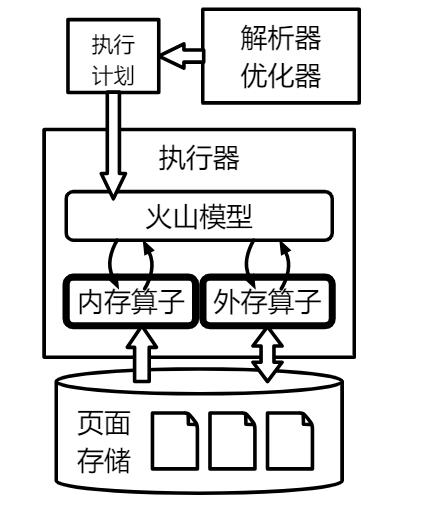


图2 相关功能模块的抽象示意图

* 1. 前置步骤

本实验均在HuaDB上进行，由C++语言编写，支持基础的页面存储、故障恢复、缓存管理、查询处理等功能。该实验提供HuaDB基础实验框架，在实验3中，需要在框架中填充相应的函数实现代码，使之通过该次实验的所有测例。实验前，需要先执行环境配置、克隆仓库、编译及测试等步骤。

### 环境配置

HuaDB 使用了 C++17 标准，开始实验前，请确保你的开发环境支持 C++17 标准。

目前 HuaDB 仅支持 macOS 和 Linux 操作系统，使用 Windows 的同学建议使用虚拟机进行实验。

HuaDB 代码下载与提交需要使用[git](https://git-scm.com/)工具，代码编译需要使用[CMake](https://cmake.org/)及[Make](https://www.gnu.org/software/make/)工具，且需要安装[gcc](https://gcc.gnu.org/)或[clang](https://clang.llvm.org/)编译器。此外，代码调试中可能会用到调试器[gdb](https://www.sourceware.org/gdb/)或[lldb](https://lldb.llvm.org/)。开始实验前，请确保你的开发环境安装了这些工具并可以正常使用，如没有，请根据你使用的操作系统选择对应的命令进行环境配置。

Ubuntu

|  |
| --- |
| sudo apt install git g++ make cmake gdb |

### 克隆仓库

从GitHub克隆代码，命令如下：

|  |
| --- |
| git clone https://github.com/thu-db/huadb.git |

克隆成功后，进入huadb目录下：

|  |
| --- |
| cd huadb |

### 编译安装

修改环境变量，命令如下：

|  |
| --- |
| vim CMakeLists.txt  set(CMAKE\_CXX\_FLAGS "-pthread")  CMAKE\_BUILD\_PARALLEL\_LEVEL=$(nproc) make |

通过以上指令可以编译出 debug 版本的程序，为便于调试，不建议编译 release 版本。

编译生成的文件位于 build/debug 目录，实验过程中只需要关注build/debug/bin 目录中的可执行程序即可，具体包括：

1. shell: 数据库程序，运行后可以与数据库进行交互。
2. sqllogictest: 测试程序，用于批量测试。

### 验证测试

运行如下命令来验证你编译出的数据库程序和测试程序可以正常运行：

|  |
| --- |
| make lab0 |

如果产生如下输出，表示程序正常运行。

|  |
| --- |
| Test: 1/3  lab0/10-basic.test PASS  lab0/20-expression.test PASS  lab0/30-set\_and\_show.test PASS  lab0/40-error.test PASS  Test: 2/3  lab0/10-basic.test PASS  lab0/20-expression.test PASS  lab0/30-set\_and\_show.test PASS  lab0/40-error.test PASS  Test: 3/3  lab0/10-basic.test PASS  lab0/20-expression.test PASS  lab0/30-set\_and\_show.test PASS  lab0/40-error.test PASS |

如果程序报错，可以根据报错信息对照[测试说明](https://thu-db.github.io/huadb-doc/overview)进行排查。

此外，你可以通过如下命令进入数据库交互界面：

|  |
| --- |
| make shell |

你可以在交互界面中运行一些基础的 DDL 命令，如：

|  |
| --- |
| Welcome to HuaDB. Type "\?" or "\h" for help  huadb> \?  \? or \h show help message  \c [database\_name] change database  \d show tables  \d [table\_name] describe table  \l show databases  \q quit  huadb> \l  +---------------+  | database\_name |  +---------------+  | tmp |  +---------------+  (1 row)  huadb> create table test(id int, info varchar(10));  CREATE TABLE  huadb> \d  +------------+  | table\_name |  +------------+  | test |  +------------+  (1 row)  huadb> \d test  +------+---------+------+  | name | type | size |  +------+---------+------+  | id | int | 4 |  | info | varchar | 10 |  +------+---------+------+  (2 rows)  huadb> drop table test;  DROP TABLE  huadb> \d  +------------+  | table\_name |  +------------+  (0 rows)  huadb> \q |

# 课程报告模板

**数据库课程实验报告**

**(2023-2024学年第一学期)**

**提交日期：年 月 日 本科生姓名：**

|  |  |  |  |
| --- | --- | --- | --- |
| **学 号** |  | **学 院** |  |
| **课程名称** |  | **任课老师** |  |
| **教师评语：** | | | |
| **成绩评定： 分 任课教师签名： 年 月 日** | | | |

# 实验九：查询处理

参考https://thu-db.github.io/huadb-doc/的实验四部分

https://gitee.com/thu-db/huadb

# 一、实验要求

1、完成HuaDB数据库内核[实验4：查询处理](https://thu-db.github.io/huadb-doc/lab4/1-intro/)的基础功能部分，按照功能点描述基础功能的实现思路（如果实现思路与文档一致，简略描述即可，如果实现思路与文档不同，可以重点描述）。如果有新增的成员变量与成员函数，简要概括其作用，以及在实现过程中遇到的难点。

2、在实验过程中你遇到什么困难和问题，你怎么解决的？（没有则可以不写）

3、扩展：[尝试实现高级功能](https://thu-db.github.io/huadb-doc/lab2/3-advanced/)，若无法实现，说说你的遇到的问题。如果在本次实验中完成了高级功能，应将此部分作为报告的重点内容，详细阐述高级功能的设计思路与实现方法。可以按照如下流程进行介绍：

(1)相关原理与示意图：总结高级功能对应的知识点，并绘制示意图。

(2)新增类的描述：介绍在基础框架之上添加的新的类的功能，以及和已有一些类的关联关系。

(3)新增成员函数的描述：介绍新增或已有类上增加的重要成员函数，以及这些函数的功能描述和调用关系。

(4)高级功能效果展示：为高级功能添加测试用例，说明新增的高级功能的正确性或优化效果。