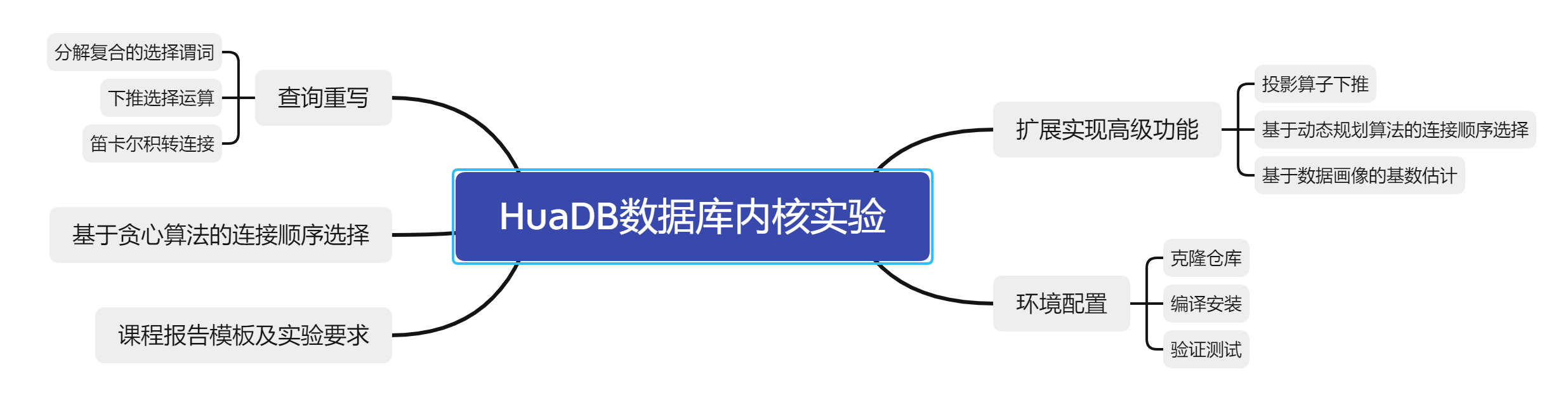
实验十：

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优化的实验，意图通过查询重写和连接顺序选择来更好地理解数据库系统如何选择查询代价较低的执行计划。

查询优化是数据库系统提供高性能查询的关键组件。这一模块的相关研究自数据库系统起源以来一直广受学界和业界的关注。具体实现上，查询优化相关的算法种类丰富，覆盖范围囊括了查询重写、物化视图、连接顺序选择、索引选择、算子选择、代价估计等诸多方面。实验框架选择了其中比较经典且重要的部分作为本地的实验内容：查询重写和连接顺序选择。

查询重写按照查询等价转化规则修改查询计划树结构，本次实验涉及其中实现相对简单的算子下推规则，通过完成算子的下推可以优化扫描算子的执行和基数估计。连接顺序选择的难点有二：第一是连接算子执行代价的估计，不同表不同列之间一般不满足独立分布假设，显著提高连接代价估计的难度。第二是连接顺序选择算法的复杂度，在连接数量较多时，讨论所有的连接顺序组合将面临指数级的复杂度，对于系统负担过高。本次实验的基础功能要求实现选择和连接算子的下推功能，并降低了连接顺序选择的实现难度，仅要求在满足课程简化假设条件下实现的连接顺序选择。高级功能要求在此基础上添加投影算子下推功能，并使用基于数据画像优化基数估计过程，从而优化连接顺序选择功能。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。

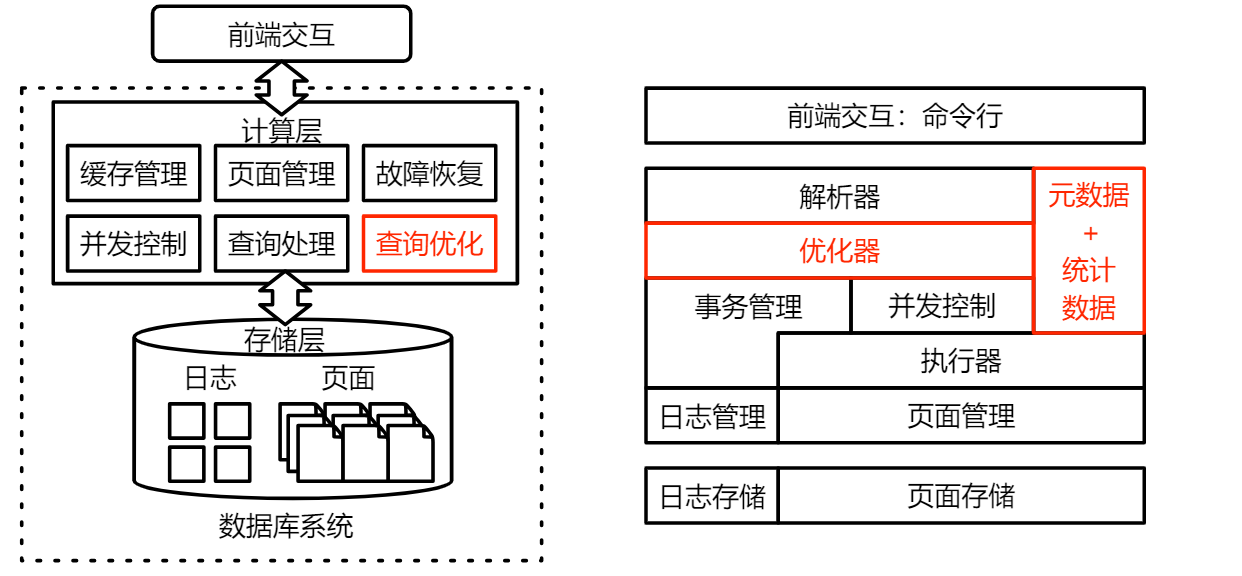


图1 数据库系统整体层次划分图

* + 1. 实验目标

**本次实验要求完成如下基础功能：**

* 选择和连接算子下推：补全优化器类中选择和连接算子下推的相关函数，实现选择算子下推。
* 连接顺序选择：基于统计信息实现使用贪心算法的连接顺序选择。

**（扩展）在基础功能之上，实验框架支持完成以下高级功能：**

* 投影算子下推：补全优化器类中投影算子下推的相关函数，实现投影算子下推。需要维护查询数的下推后变化的数据列信息。
* 连接顺序选择优化：实现基于动态规划算法的连接顺序选择，优化连接顺序选择的查找空间。
* 基于数据画像的基数估计：使用数据画像方法实现更准确的基数估计，优化连接顺序选择的准确性。
  1. 实现查询优化功能概述

与上次实验类似，本次实验同样具有较为明确的目标，所有修改均在 optimizer.cpp 和 optimizer.h 中完成，你需要在优化器中操作查询计划树节点，得到更优的查询计划。

完成本实验时，你需要操作查询计划树节点，可能会生成新的节点，熟悉 Operator 的成员变量及构造方式将会对完成本次实验有较大的帮助，建议在开始实验前阅读 operators 文件夹下的代码，了解不同 Operator 的构造方式。

* + 1. 查询重写

本任务中，你将实现课程讲述的查询重写算法，涉及 SplitPredicates, PushDownFilter, PushDownSeqScan 和 PushDownJoin 四个函数。

由于谓词的形式多种多样，为简化代码工作量，你只需通过本次实验测例中出现的情况即可，具体而言：

复合谓词均由单一谓词的合取 (and) 组成，不包含析取 (or)。

单一谓词仅包括普通谓词和连接谓词两类。

普通谓词均为 <列> <比较操作符> <常数> 的形式，即列出现在谓词左侧，常数出现在谓词右侧，如 id < 1, a.id = 3 等。其中 <列> 和 <常数> 均为普通值，不会出现其他形式的表达式（如 a.id + 1 = 3 \* 2 这样的复杂形式）。

每个表最多只有一个对应的普通谓词。

连接谓词均为列与列的等值连接。

多表连接均为左深树。

**步骤 1：分解复合的选择谓词**

我们首先来完成 SplitPredicates，该函数将复合的选择谓词分解为多个选择节点。你需要遍历查询计划树，当遍历到选择（Filter）节点时，判断其是否由多个合取谓词组成，若满足条件，则将其分解为多个选择节点。

**步骤 2：下推选择运算**

在 PushDown 函数中，优化器会再次遍历查询计划树，并根据查询计划树节点调用对应的优化函数，本步骤中你需要完成PushDownFilter和PushDownSeqScan 函数，完成后将通过10-filter-pushdown.test测例。

PushDownFilter 函数负责处理 Filter 节点，你需要判断谓词是普通谓词还是连接谓词。

PushDownSeqScan 函数负责处理查询计划树底层的扫描节点，该函数需要查找当前计划树的普通谓词是否使用了扫描节点中包含的列，若存在这样的普通谓词，则将相应的谓词下推到该扫描节点上方。

本次实验的测试使用了 explain 命令，该命令会输出查询计划树，测试程序会比对你优化后的查询计划树与期望结果是否相同。

**步骤 3：笛卡尔积转连接**

你需要补充 PushDownJoin 的函数实现部分，完成后将通过20-join-pushdown.test测例。

PushDownSeqScan 函数处理连接节点，对于查询计划树中的笛卡尔积节点，即 join\_condition 为 true 的节点，需要判断当前查询计划树是否存在连接谓词可以用于该节点，如果有则修改该节点的 join\_condition，从而实现笛卡尔积到连接的转换。

* + 1. 基于贪心算法的连接顺序选择

在本任务中，你将使用贪心算法进行多表连接的连接顺序选择，你需要补充 optimizer.cpp 中的 ReorderJoin 函数，完成后将通过最后一个测例 30-join-order.test。

贪心算法的实现流程在课程中已有详细的描述，文档中不再赘述，你可以参考课程 PPT 来熟悉连接顺序选择问题的贪心算法。

本任务的测例也与课程 PPT 中的例子相同，唯一的区别是在表 r3 中多插入了一条数据，这是为了避免贪心算法中将 r3 选为第一个表，从而产生不同的运行结果。你的算法运行流程应与课程 PPT 中的步骤相同，你可以对照课程 PPT 进行调试。

* + 1. 扩展高级功能

**（1）任务 1：**投影算子下推

【实验描述】

基础功能中完成了选择和连接算子的下推，高级功能中要求在此基础上添加投影算子的下推功能。

【实现思路】

步骤 0：理解投影算子下推相较于选择算子的区别，思考如何在算子下推后维护数据列信息。

步骤 1：修改 Optimizer 类，添加用于辅助 PushProjection 函数的相关私有成员函数和成员变量。

步骤 2：补全 Optimizer 类的 PushProjection 函数，注意并非部分连接运算后需要添加投影运算。

**（2）任务 2：基于动态规划算法的连接顺序选择**

【实验描述】

基础功能中的连接顺序选择使用的是贪心算法，高级功能中要求在此基础上添加基于动态规划的连接顺序选择算法。

【实现思路】

在 ReorderJoin 函数中基于连接种类选择使用的算法类型，并在Optimizer 中添加基于动态规划的连接顺序选择算法的成员函数。

**（3）任务 3：基于数据画像的基数估计**

【实验描述】

基础功能中连接顺序选择过程基于了独立性简化假设进行技术估计，高级功能中要求使用数据画像来提高distinct value 估计的准确性（基于Count-Min或HyperLogLog算法）。

【实现思路】

修改数据库 Analyze 过程，在分析过程中使用Count-Min或HyperLogLog算法估计distinct value。

1.2.4 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：

|  |
| --- |
| * [optimizer](https://github.com/thu-db/huadb/blob/main/src/optimizer)：优化器相关类   + [optimizer](https://github.com/thu-db/huadb/blob/main/src/optimizer/optimizer.h)：优化器类，需要补全算子下推功能以及连接算子重新排序的功能。 |

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：

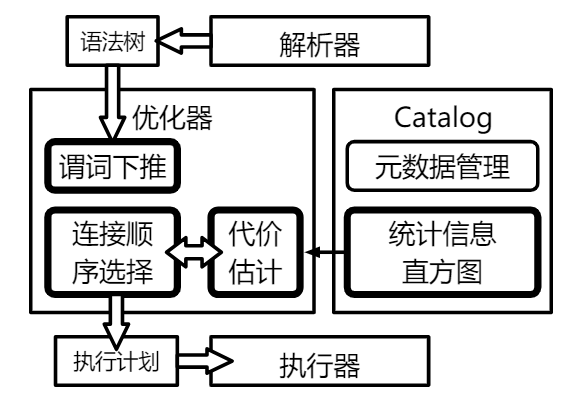


图2 相关功能模块的抽象示意图

* 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** | | | |

# 实验十：查询优化

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5：查询优化](https://thu-db.github.io/huadb-doc/lab5/1-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。