数据库实验报告

计 76 陈之杨 2017011377

2020.1

1 引言

DBNoC(A **D**ata**B**ase that Draws **No C**onclusions)[1]是一个简单的关系数据库管理系统，实现了基础的 SQL 语句，支持管理多个数据表与索引，并进行增删查改的操作。

2 使用方法

在 `src` 目录下，使用 `make` 命令即可完成编译。生成的可执行文件 `create`,`drop`,`use` 分别对应创建、删除、进入数据库。

3 支持的数据与语句

DBNoC 支持三种类型的数据类型：

- `int`，为 32 位整型数据。
- `float`，为双精度浮点型数据。
- `char()`，为字符串数据。由于页式文件系统单页的长度为 8KB，故字符串长度不能超过 8192。

支持的 SQL 语句类型包括：

- 创建表 `create table ... (... [not null] [primary key] [foreign key ... references ...(...)] [default ...])`

[1]这个名字化用日本京都动画的动画作品《冰果》中福部里志的经典台词：「数据库无法得出结论。」在此，用这个名字向京都动画纵火案中的逝去者，以及那些名字在历史的年表彼端化为古典的年轻人致敬。

- 删除表 `drop table ...`
- 显示表 `show`
- 添加主键 `alter table ... add primary key (...)`
- 删除主键 `alter table ... drop primary key`
- 添加外键 `alter table ... add foreign key (...) references ...(...)`
- 删除外键 `alter table ... drop foreign key ...`
- 添加列 `alter table add ... not null default ...`
- 删除列 `alter table drop ...`
- 插入数据 `insert into ...(...) values (...)`
- 删除数据 `delete from ... where ... [and ...]`
- 修改数据 `update ... set ... = ...`
- 选择数据 `select ... from ... where ... [and ...]`

4 实现细节

4.1 记录管理

这一部分使用页式文件系统对记录进行管理。数据库的每一个表对应一个文件，页中记录长度即为表所有列大小之和。为了支持存储 NULL 值，记录开头额外记录一个 bitmap 存储所有列的值是否为空。

为了充分利用页式文件资源，记录管理系统使用链表结构维护所有有空位的页。具体地，每个页记录下一个有空闲位置的页，文件头额外占用一页存储元信息（包括链表头）。每次增加/减少有空闲位置页时，更新链表信息，这样插入时可以 $O(1)$ 找到第一个有空闲位置的页。在页内，为了快速找到空闲的记录位置，在页头记录一个 bitmap 维护每个记录槽是否是空的，这样可以在 $O\left(\frac{n}{w}\right)$ 的时间找到进行一次空/非空记录的查找（n 为记录槽的数量，w 为 bitmap 字长）。

4.2 属性索引

这一部分为一个记录文件的某些属性建立基于 B+ 树的索引以加速查询。对于一个表 `table` 的属性 `col`，文件 `table.col` 存储了 `col` 属性的 B+ 树结构。事实上，联合索引也是支持的，对于联合主键，我们默认维护一个联合索引以快速判断主键是否存在重复。

4.3 系统管理

这一部分数据库系统的管理，主要包括表信息的操作。对于主键和外键的管理，建立 B+ 树索引以快速判断重复数据。对于列的增删操作，我没有想到好的实现方法，实现的方式是直接以新的记录大小建新的表文件，然后重建全部索引（因为新的文件中记录的 ID 出现了变化）。事实上，可以在建表时将记录长度预留为两倍大小，在插入属性后大小超过了上限再重建记录文件，而删除属性用懒惰删除实现，这样可以做到均摊线性的时间复杂度。不过考虑到这两个操作并不常见，不是主要的效率瓶颈，所以没有额外费力实现。

4.4 查询解析

这一部分进行增删查改命令的执行。对于单条记录插入，可直接在记录管理系统中插入，并使用主键/外键索引判断重复性。对于删除/更新/查询操作，为了快速找到符合 `where` 子句的记录，在对应属性建立索引的情况下使用索引加速查询。在理想情况下，可以做到 $O(\log N + M)$ 的时间复杂度（其中 N 为记录数量，M 为符合条件的结果数量）。

特别地，`select` 语句可以支持两个表的连接。考虑到暴力查询的复杂度是表规模平方级别的，对于大规模数据时间消耗难以忍受，因此我实现了简单的查询优化：当约束中存在 $a.attr1 = b.attr2$ 的条件时，若 a 的 $attr1$ 属性建立了索引，则枚举 b 表，用 a 表的索引加速查询；反之亦然；若两表均存在索引，则选择记录数较少的表进行枚举。经过测试，对于一个百万规模的数据集（大小约 180MB），进行两表连接查询依然可在 10s 以内得到结果。

5 实验总结

通过这次的数据库实验，我充分了解了一个数据库系统的各个组成部分与工作原理，同时也认识到两点：保持一个数据库的鲁棒性是一件非常复杂的事；提高数据库的查询效率是数据库设计的重点。总的来说，这次实验让我收获良多。虽然囿于时间限制，没能充分实现想要实现的所有特性，然管中窥豹仍窥得现代数据库的冰山一角。在此感谢各位老师和助教一学期来的付出。

最后是一点建议，希望助教们在布置大作业时能明确统一一下实验要求，尤其是避免各个说明文件之间的冲突。此外，页式文件系统中也有一些需要修补的地方，虽然不是大问题，但对于调试了半天才发现页式文件系统需要修改的同学来说，体验实属不佳。