Lab 4

1. 代码逻辑

1.1 exercise 1

exercise1 要求我们实现一个“基于桶的计算直方图的方法”来记录表统计信息以用于选择性估计，需要我们实现 IntHistogram.java。

——IntHistogram 函数实现了直方图的初始化。首先，将最小值和最大值赋给类的成员变量，并根据桶数和范围计算实际使用的桶数，确保桶数不会超过区间范围。然后，为每个桶创建一个 HistogramBar 对象，并确定每个桶的左边界和右边界，其中左边界通过最小值和当前桶的索引计算，右边界则由下一个桶的左边界决定，最后一个桶的右边界为最大值。

——estimateSelectivity 函数实现了根据给定的谓词操作符和值，估计查询在直方图中的选择性的功能首先判断值是否在直方图的范围内，如果不在，则根据操作符直接返回 0 或 1 的选择性。如果值在范围内，通过计算值所在的桶索引，依据不同的操作计算选择性。具体实现包括累加相关桶的计数，考虑值在桶中的相对位置，并最终返回选择性结果。

1.2 exercise 2

exercise 2 要求我们实现 TableStats 类，以计算表的元组和页数，并使用直方图估计谓词选择性，具体包括实现构造函数、选择性估计、扫描成本估计和表基数估计的方法。

——TableStats 函数通过获取指定表的 DbFile 并扫描其元组来计算统计信息实现了 TableStats 类的初始化。首先，初始化表 ID、IO 成本和元组描述符，并创建哈希表来存储每个字段的最大值、最小值以及相应的直方图。接着，使用迭代器扫描表元组，更新每个整型字段的最大值和最小值，并为每个字段创建 IntHistogram 对象。然后，重新扫描表元组，填充每个字段的直方图，并计数表中的总元组数。

——estimateSelectivity 函数实现了估算指定字段与给定谓词选择性的功能。首先，检查字段是否存在于直方图映射中，如果存在，则根据字段类型调用相应的直方图对象来估算选择性。

1.3 exercise 3

exercise 3 要求我们实现 JoinOptimizer.java，估算连接操作的选择性和成本。

——estimateTableJoinCardinality 函数实现了估算两个表在特定连接条件下的连输出的元组数量。根据连接操作符（等于、不等于或其他）和两个表的主键情况，选择不同的计算逻辑：如果是等于操作符且两个表都是主键，则返回较小的基数；如果只有一个表是主键，则返回另一个表的基数；如果都不是主键，则返回较大的基数。对于不等于操作符，根据主键情况计算所有可能组合减去相应的最小或最大基数。对于其他操作符，估算连接基数为两个表基数乘积的 30%。

1.4 exercise 4

exercise 4 要求我们实现 JoinOptimizer 类中的 orderJoins 函数，以确定连接操作的顺序，并返回一个向量表示左深树计划中连接操作的顺序，并确保相邻连接共享至少一个字段，同时输出连接顺序的表示用于信息目的。

——orderJoins 函数实现了如何确定连接操作的最佳顺序。首先，通过枚举所有可能的子集，并对每个子集计算其连接成本和基数，选择成本最低的连接计划，并将其存储在全局计划缓存中。最终，返回全局计划缓存中存储的最佳连接顺序，以确保生成一个左深树计划。

2. 实验心得

该实验花费了我 5 天时间完成。其中，我花费了两天时间认真学习了查询计划成本估计的方法论，收获颇多。