

RowGroup BloomFilter 场景介绍和性能测试

1. 测试场景

1. =, in 命中 BloomFilter 性能情况，不命中 BloomFilter 对查询的影响
2. <, >, != 等其他比较符的不命中 BloomFilter 对查询的影响
3. 特殊场景下的优势
4. 构建的性能影响

2. 总结

2.1 性能表现

1. 如果查询能命中点查（= or in），BloomFilter 过滤后的性能提升可观，随能过滤掉的 RowGroup 数据大小而定。
相比 Segment Pruning 索引文件夹级别的过滤，BloomFilter 粒度更细，定位于文件 RowGroup 组成块。
一个 Spark Task 同时只能读取一个 RowGroup 大约 128M，跳过一个 RowGroup 扫描大约节省 1s，一次查询可能跨多个 Segment 的索引，一个索引可能会有多个 RowGroup (但 Spark Task 是并行读取的)
同时可以减轻 计算资源消耗 以及 降低查询时间。
2. 当点查的内容 有较大的离散性 或 随着 时间分区有较大的相关性，能够有较好的过滤效果。
BloomFilter 适用于较高基的列（3w以上），因为在数据中会有较大的区分度
3. BLoomFilter 按需加载，如果查询没有用到，不会加载。
如果查询没有使用 BLoomFilter 列，或者查询条件不为点查（为 <,> 不等值查询等），查询没有影响。
如果查询使用 BLoomFilter，也有点查，但查询不命中，多加载一个 BLoomFilter 耗时大概 15ms，由于 Task 是并行执行的，这个影响小。

2.2 适用场景

1. 增量构建模型，查询范围较广，但列值区分度比较高，例如高基列（比如3w以上，低基列可以放心构建，parquet 实现中对于低基列，即使设置了，最终也不会生成 BloomFilter，因为已经有字典了）

2. 增量构建模型，列值和时间相关性大，查询跨读个时间分区（跨多个 Segment），此时也能做比较好的筛选。例如这种场景，查询半年内，客户 A 的购物订单，客户 A 一般只会在某些天中购物过，那么此时筛选效果就比较好
3. 全量构建模型，没有时间分区列，点查的列具有主键的性质，或者列值高基的话也适用。

3. 数据准备

3.1 相关配置

目前支持的功能：

1. 查询：**自动收集点查列**（前提是模型已经构建完成，并进行了击中模型的查询）
2. 构建：BloomFilter 支持自动按照点查频率自动筛选前几的列，也支持手动配置列
3. 查询：Metrics 结果自动收集，收集 BloomFilter 击中总次数，过滤的 RowGroup 数，过滤的数据行数

通常打开 BloomFilter 功能，只需要一个开关 `kylin.bloom.build.enabled=true`，最低可作用于模型级别

配置项	配置用途
<code>kylin.bloom.build.enabled=false</code>	<code>true</code> 代表构建 BloomFilter 列
<code>kylin.bloom.build.column-ids</code>	默认空值，自动构建点查频率高的列 也支持手动指定构建 BloomFilter 的列 ID，# 作为分隔符， <code>columnId#columnId2#columnId3</code>
<code>kylin.bloom.build.column.max-size=3</code>	自动构建多少个 BloomFilter 列（按照点查频率降序排列） 推荐调大些
<code>kylin.bloom.build.column.nvd=200000</code> (自适应 BloomFilter 做好后，废弃了)	BloomFilter 列的构建去重数预估（如果去重数超过，会降低 BloomFilter 命中概率）
<code>kylin.bloom.collect-filter.enabled=true</code>	<code>true</code> 代表开启自动收集点查列
<code>kylin.query.filter.collect-interval=1800</code>	自动收集的点查列同步至 HDFS 上供构建使用的频率，单位秒

3.2 数据集

SSB10 数据集 6000w 数据，增量构建 1992~1999 共 7 个 Segment，模型关联 SQL:

```
SSB10.P_LINEORDER t1 LEFT JOIN SSB10.CUSTOMER as t2 ON t1.LO_CUSTKEY  
= t2.C_CUSTKEY
```

先建立模型 SSB10，构建完成后查询几条 sql，带上 = or in 条件，击中模型后，内部会自动收集查询条件（本次性能测试建立的 BloomFilter 列为 LO_CUSTKEY, c_phone，V_REVENUE

3.3 环境信息

kcluster 客户端环境 10.1.2.134，查询 sparder 配置如下

- 1 `kylin.storage.columnar.spark-conf.spark.executor.memory=4g`
- 2 `kylin.storage.columnar.spark-conf.spark.executor.cores=2`
- 3 `kylin.storage.columnar.spark-conf.spark.executor.instances=5`
- 4 `kylin.storage.columnar.spark-conf.spark.driver.memory=2g`

4. 场景性能测试详情

4.1 = or `in` or `and` 组合条件命中 BloomFilter:

4.1.1 = 号查询，查询 c_name = 'Customer#000196829'

BloomFilter 能力: 50 个 rowGroup 跳过 36 个 RowGroup

扫描数据总时长: 49.3s -> 10.5s

Filter 算子总时长: 57s -> 13s

扫描数据量 (bytes): 5,164,066,453 -> 1,525,111,877

查询总时长: 5.8s -> 2.3s，由于 task 是并发执行，且测试场景资源充足，所以虽然总体 task 执行时间长，但查询时长差距不会像扫描数据总时长差距那样大(但其实使用 BloomFilter 其实也降低了不少总计算资源的消耗)

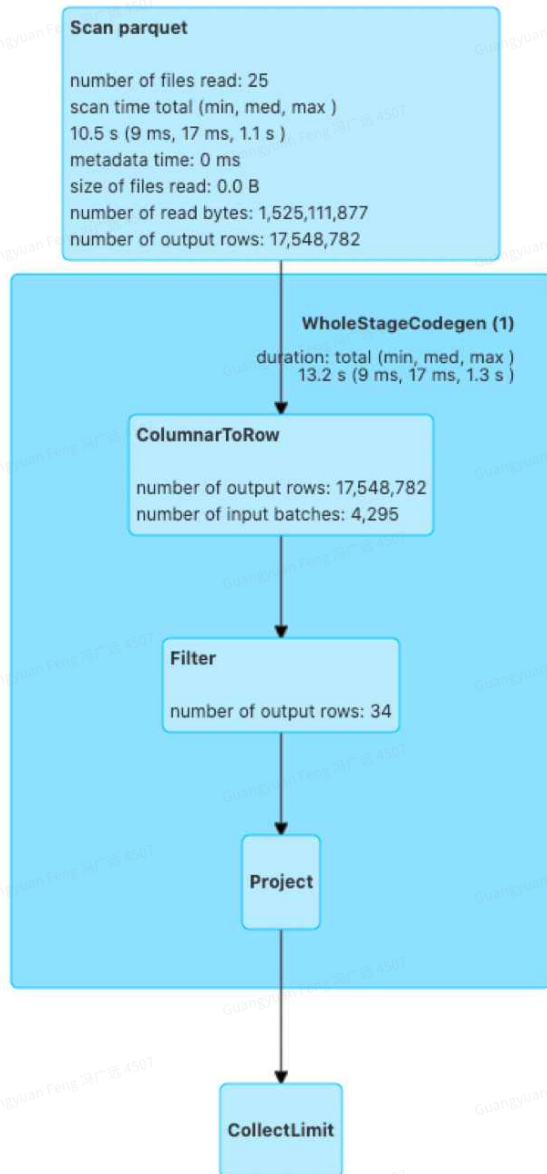
Details for Query 187

Submitted Time: 2023/03/02 18:17:14

Duration: 2 s

Succeeded Jobs: 333 334 335 336

☐ Show the Stage ID and Task ID that corresponds to the max metric



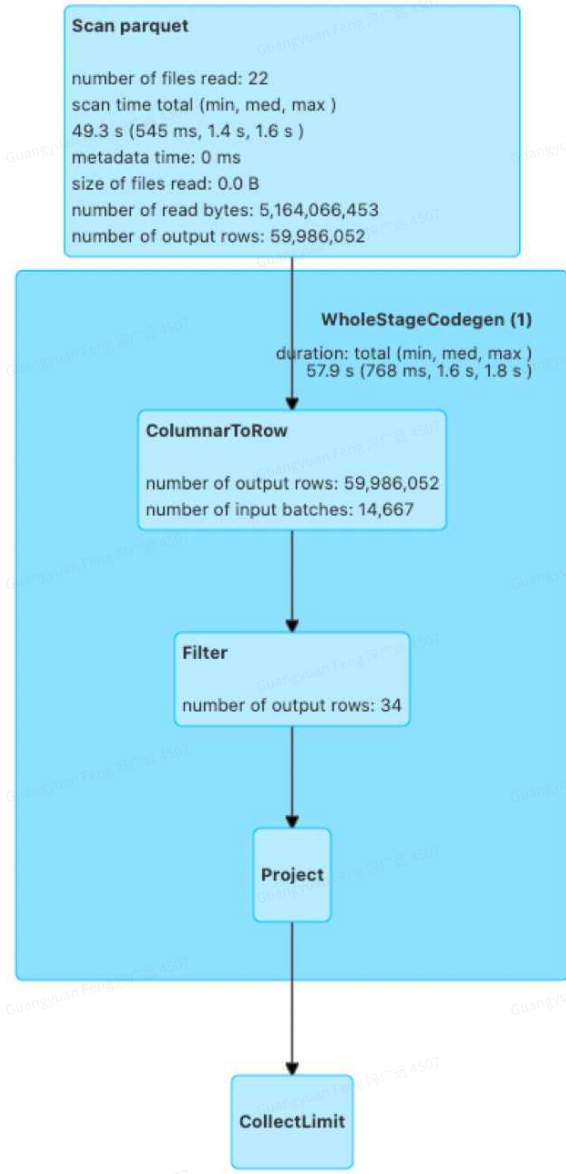
Details for Query 188

Submitted Time: 2023/03/02 18:17:52

Duration: 5 s

Succeeded Jobs: 337 338 339 340

☐ Show the Stage ID and Task ID that corresponds to the max metric



4.1.2 And 组合条件，查询 LO_CUSTKEY = 279821 and V_REVENUE = 4294052

BloomFilter 能力：测试两个 BloomFilter 列组合过滤的性能，交叉后过滤效果更好，52 个 rowGroup 跳过 45 个 RowGroup

扫描数据总时长：52 s -> 6 s

扫描数据量(bytes)：59,986,052 -> 9,957,365

查询总时长：5.5s -> 1.9s

Details for Query 41

Submitted Time: 2023/03/02 14:24:02

Duration: 2 s

Succeeded Jobs: 76 77 78 79

☐ Show the Stage ID and Task ID that corresponds to the max met

Scan parquet

number of files read: 26
scan time total (min, med, max)
6.0 s (10 ms, 20 ms, 904 ms)
metadata time: 0 ms
size of files read: 0.0 B
number of read bytes: 894,078,651
number of output rows: 9,957,365

WholeStageCodegen (1)

duration: total (min, med, max)
7.6 s (11 ms, 20 ms, 1.1 s)

ColumnarToRow

number of output rows: 9,957,365
number of input batches: 2,437

Filter

number of output rows: 1

Project

CollectLimit

[Details](#)

Details for Query 43

Submitted Time: 2023/03/02 14:25:14

Duration: 5 s

Succeeded Jobs: 84 85 86 87

☐ Show the Stage ID and Task ID that corresponds to the max metric

Scan parquet

number of files read: 22
scan time total (min, med, max)
52.7 s (530 ms, 1.4 s, 1.8 s)
metadata time: 0 ms
size of files read: 0.0 B
number of read bytes: 5,164,066,453
number of output rows: 59,986,052

WholeStageCodegen (1)

duration: total (min, med, max)
1.0 m (760 ms, 1.6 s, 1.9 s)

ColumnarToRow

number of output rows: 59,986,052
number of input batches: 14,667

Filter

number of output rows: 1

Project

CollectLimit

[Details](#)

4.1.3 In 条件查询，查询 c_phone in ('24-927-428-9519', '14-150-653-4465')

BloomFilter 能力 : 52 个 rowGroup 跳过 33 个 RowGroup

扫描数据总时长: 50 s -> 16 s

扫描数据量(bytes) : 5,164,066,453 -> 2,132,784,582

查询总时长: 5.6s -> 2.8s

4.2 查询 BloomFilter 但不命中的情况

4.2.1 = 条件，查询 LO_CUSTKEY = 7735 and C_NAME = 'Customer#000007735'

BloomFilter 能力：两个列均带 BloomFilter，且均不命中，额外读取了 50个 BloomFilter（因为这里并行 task 的原因，性能影响小）

扫描数据总时长：31.4s -> 33.1s

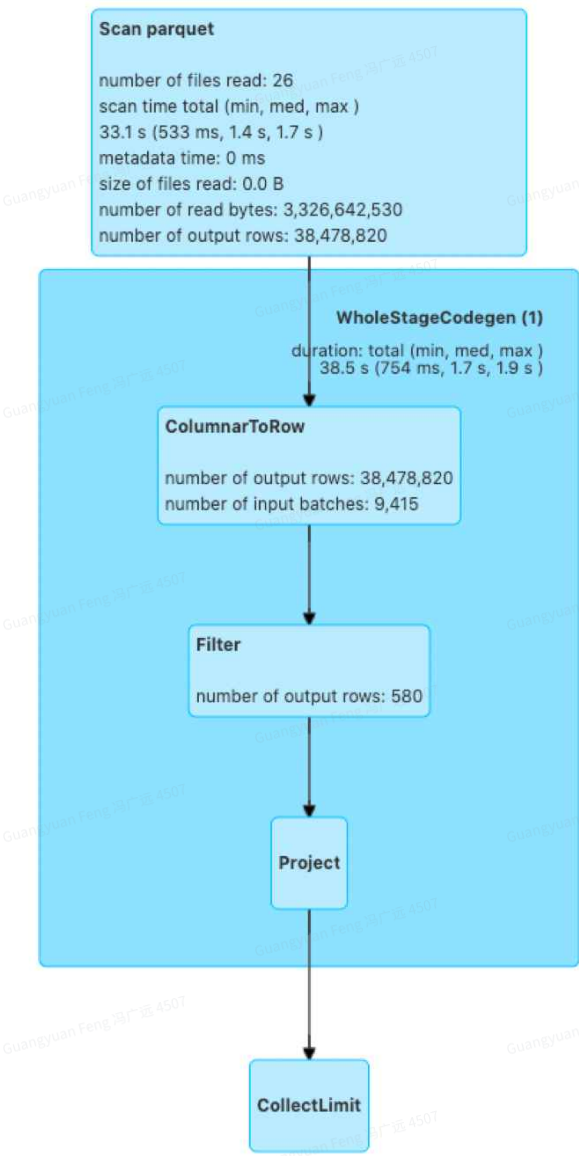
扫描数据量(bytes)：3,326,642,530 -> 3,314,779,252

查询总时长：3.82 -> 3.89s

原因总结：构建时，一个索引默认只构建 3 个列 BloomFilter（最大占 1MB），一个 RowGroup 默认为128MB，测算额外读取 1个 的BloomFilter 读取速度大约是 15 ms，默认一个 RowGroup 最多额外读取 3 个 BLoomFilter。这里额外读取 50 个 BLoomFilter，在 task 并行的情况下，查询时间影响小。

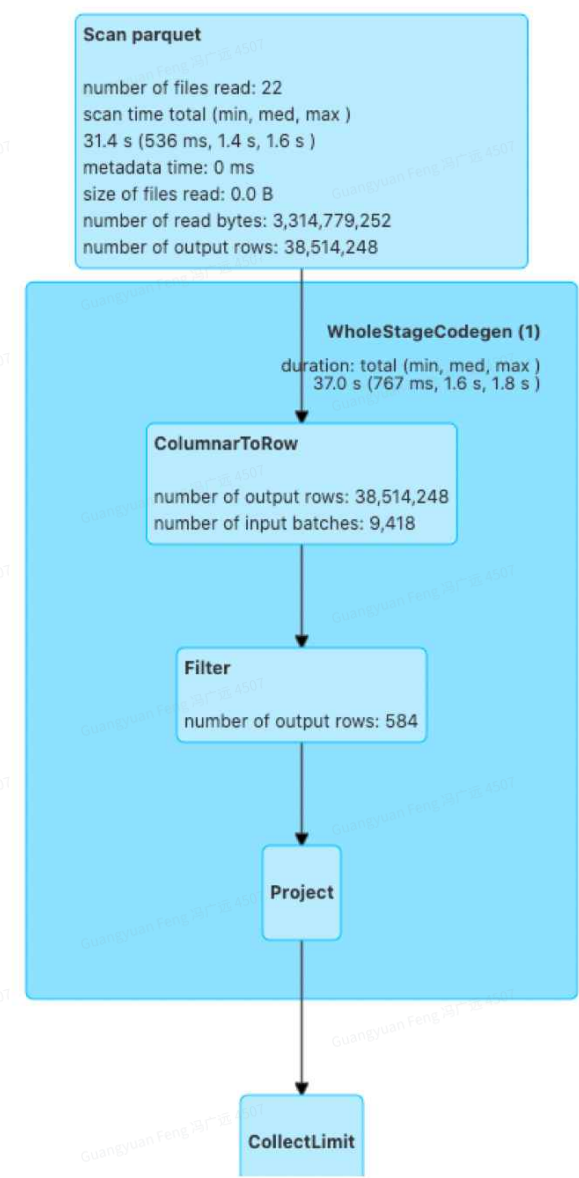
Details for Query 97

Submitted Time: 2023/03/02 15:09:51
Duration: 4 s
Succeeded Jobs: 197 198 199
☐ Show the Stage ID and Task ID that corresponds to the max metric



Details for Query 95

Submitted Time: 2023/03/02 15:06:37
Duration: 4 s
Succeeded Jobs: 191 192 193
☐ Show the Stage ID and Task ID that corresponds to the max metric



4.2.2 <、!= 等查询，查询 LO_CUSTKEY > 7735 and V_REVENUE != 6578085

速度不影响，扫描的 bytes 也基本一致（133,401,871 -> 133,371,369）

原因：

虽然构建了 BloomFilter ，但因为查询**未使用 BloomFilter**，所以不会去读取 Footer 中的 BloomFilter，所以读取的数据量基本完全一样。**Parquet 读取 BloomFilter 是按需读取。**

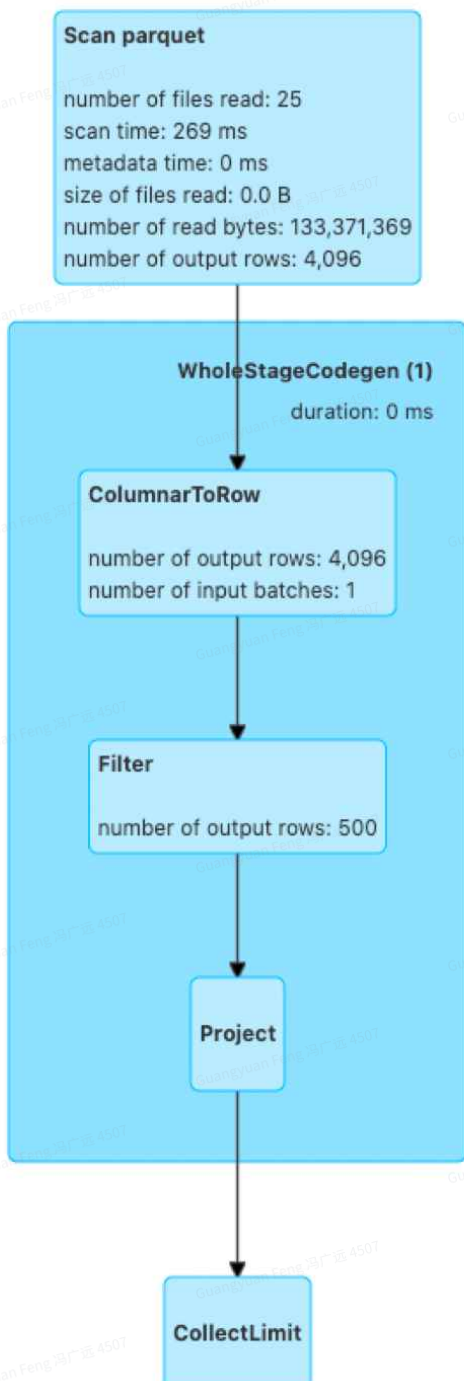
扫描文件数量不同？这是因为构建出来后，实验数据集正好在一个临界值，导致了 KE 中 repartition write 的数量不同。但是 **Spark 读取数据是以 RowGroup 为单位并发进行的, RowGroup 两者总数相同，所以这部分差别可以忽略**

Submitted Time: 2023/03/02 18:22:11

Duration: 0.4 s

Succeeded Jobs: 341

☐ Show the Stage ID and Task ID that corresponds to the max n



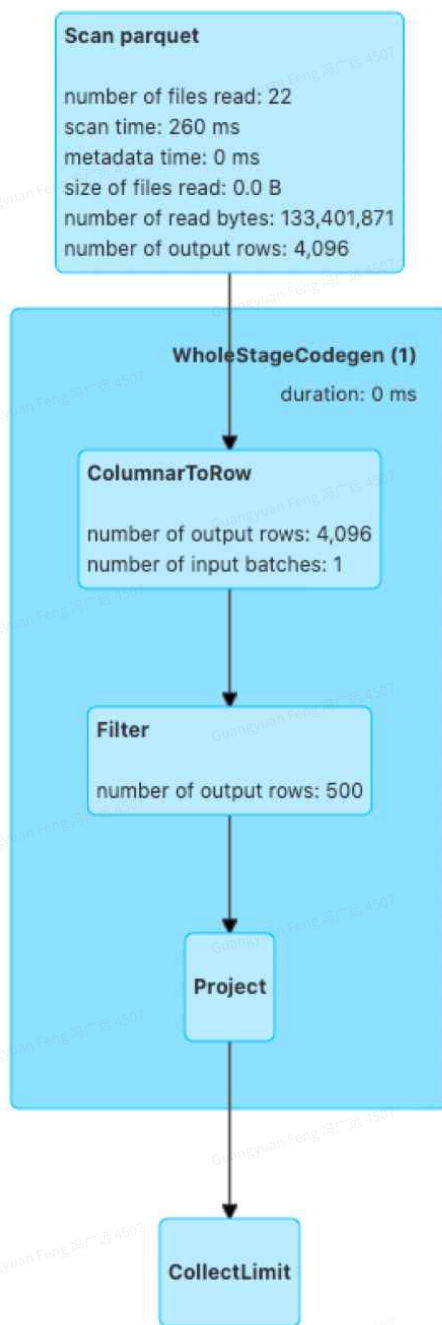
Details for Query 190

Submitted Time: 2023/03/02 18:22:21

Duration: 0.4 s

Succeeded Jobs: 342

☐ Show the Stage ID and Task ID that corresponds to the m:



4.3 特殊查询场景

4.3.1 BloomFilter 全部过滤，查询条件 LO_CUSTKEY = 274929

查询条件结果为空，BloomFilter 过滤了全部文件，此时性能最佳，查询时间为 5.3 s -> 0.4 s，后者扫描了全部数据

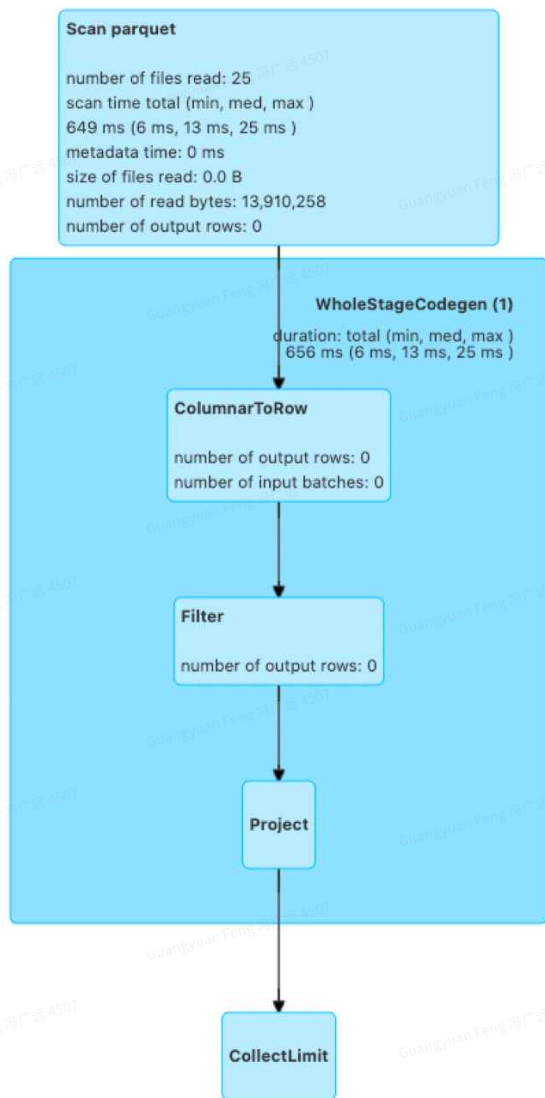
Details for Query 191

Submitted Time: 2023/03/02 18:28:07

Duration: 0.3 s

Succeeded Jobs: 343 344 345 346

☐ Show the Stage ID and Task ID that corresponds to the max metric



[Details](#)

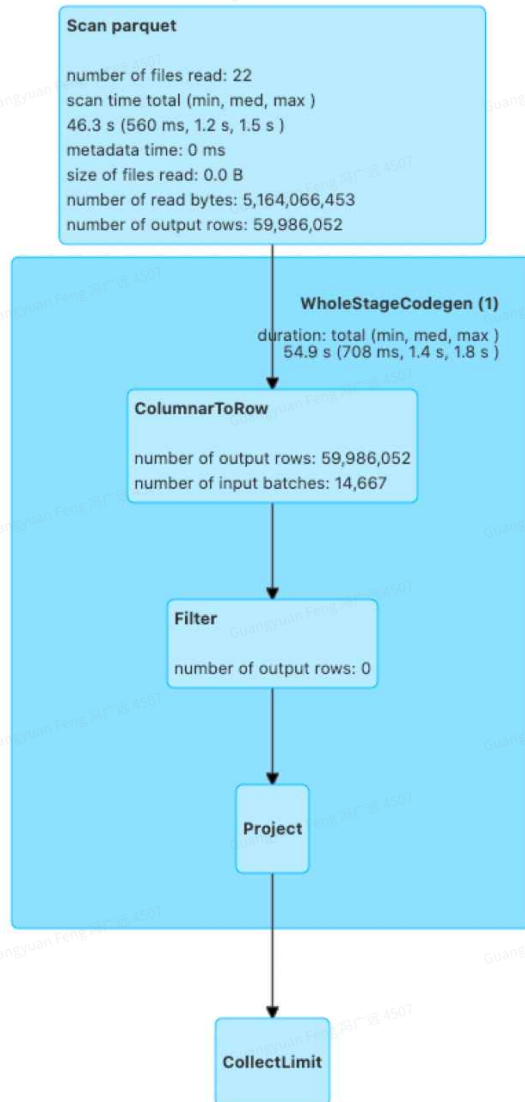
Details for Query 192

Submitted Time: 2023/03/02 18:28:12

Duration: 5 s

Succeeded Jobs: 347 348 349 350

☐ Show the Stage ID and Task ID that corresponds to the max metric



[Details](#)

4.3.2 全量构建

全量构建的性能对比类似 4.1，命中 BloomFilter 就有很好的 IO 提升