Prashant Garmella **LAB 4 Report**
pg1910

Part 1 and Part 2

Query	**File**	**Trials**	**Min**	**Median**	**Max**
csv_avg_income	people_small.csv	25	1.031991	1.491647	4.406990
csv_avg_income	people_medium.csv	25	2.070965	2.207761	2.966112
csv_avg_income	people_large.csv	25	33.393495	34.143776	37.815225
csv_max_income	people_small.csv	25	1.193147	1.686682	4.680297
csv_max_income	people_medium.csv	25	1.701714	1.985315	2.392672
csv_max_income	people_large.csv	25	29.656693	30.420860	35.714299
csv_anna	people_small.csv	25	0.059173	0.067386	0.175025
csv_anna	people_medium.csv	25	0.604238	0.618322	0.648367
csv_anna	people_large.csv	25	28.664482	31.056611	32.730573
pq_avg_income	people_small.parquet	25	1.167195	1.477652	1.860673
pq_avg_income	people_medium. parquet	25	2.229306	3.996998	4.651561
pq_avg_income	people_large. parquet	25	5.587183	5.938744	6.400656
pq_max_income	people_small. parquet	25	1.381715	1.818207	2.696031
pq_max_income	people_medium. parquet	25	0.917246	1.166410	1.475466
pq_max_income	people_large. parquet	25	4.597737	5.182013	6.379729
pq_anna	people_small. parquet	25	0.052965	0.056406	0.525218
pq_anna	people_medium. parquet	25	1.076286	1.089115	1.156715
pq_anna	people_large. parquet	25	4.602650	4.915585	5.842460

Part 2: The comparision between times for CSV and parquet files:

1. For small files, for all the queries, the CSV and the parquet results on almost same benchmark times.

2. For medium files, the parquet files did little better for max_income and anna queries and the csv files did better for the avg_income query. Overall, I could not observe a significant performance improvement with medium parquet files.
3. For large files, it was a completely different ball game. Wherein the performance improvement in benchmark times for all the queries for parquet files was huge, on an average going down from 30 seconds to 5 seconds. Almost an improvement of 600%.

Part 3

Query	**File**	**Trials**	**Min**	**Median**	**Max**
Sorting by columns					
pq_avg_income	people_small_sorted.parquet	25	2.47744	2.710182	5.735567
pq_avg_income	people_medium_sorted. parquet	25	2.623950	3.044242	4.577368
pq_avg_income	people_large_sorted. parquet	25	4.998908	5.457127	6.528325
Setting replication factor as 2					
pq_avg_income	people_small.parquet	25	1.288203	1.459219	1.746665
pq_avg_income	people_medium. parquet	25	0.860674	1.128406	1.959744
pq_avg_income	people_large. parquet	25	4.230856	4.934718	7.503671
Repartitioning set as 10					
pq_avg_income	people_small_repar.parquet	25	1.613537	2.984154	3.848125
pq_avg_income	people_medium_repar. parquet	25	0.883748	1.074595	1.408225
pq_avg_income	people_large_repar. parquet	25	3.591282	3.842066	4.564424
Sorting by columns					
pq_max_income	people_small_sorted.parquet	25	1.977269	2.123284	2.455748
pq_max_income	people_medium_sorted. parquet	25	1.307645	1.496937	1.915234
pq_max_income	people_large_sorted. parquet	25	3.093833	4.182736	5.303842
Setting replication factor as 2					
pq_max_income	people_small.parquet	25	1.156720	1.407076	1.671529
pq_max_income	people_medium. parquet	25	0.932965	1.078947	1.415204
pq_max_income	people_large. parquet	25	5.184244	5.970955	9.358389
Repartitioning set as 10					
pq_max_income	people_small_repar.parquet	25	1.685918	2.913784	3.708182
pq_max_income	people_medium_repar. parquet	25	0.908721	1.135798	1.441898
pq_max_income	people_large_repar. parquet	25	4.708747	5.105352	6.199985

Sorting by columns					
pq_anna	people_small_ sorted.parquet	25	0.057989	0.059443	0.723423
pq_anna	people_medium_sorted. parquet	25	1.034433	1.069384	1.290939
pq_anna	people_large_sorted. parquet	25	0.515447	0.545430	1.430427
Setting replication factor as 2					
pq_anna	people_small.parquet	25	0.053332	0.056716	0.096144
pq_anna	people_medium. parquet	25	0.087161	0.102286	0.178723
pq_anna	people_large. parquet	25	2.591359	4.186236	6.0434503
Repartitioning set as 10					
pq_anna	people_small_repar.parquet	25	0.073418	0.082913	0.706327
pq_anna	people_medium_repar. parquet	25	0.096635	0.110339	0.187193
pq_anna	people_large_repar. parquet	25	2.209195	2.320198	2.847126

Configuration used for the getting better results:

1. Replication factor was reduced from a default of 3 to 2. (Increasing replication would increase time)
2. Repartioning was increased from a default of 1 for small and medium files and a default of 4 for large files to 10.
3. Sorting by columns was performed on all the columns, first_name, last_name, income and zipcode.

Compared to the results in Part 2,

pq_avg_income:

1. small files: Setting replication factor as 2 gave the best result (1.459219) followed closely by original parquet file (1.477652)
2. medium files: Repartitioning set as 10 gave the best result (1.074595) whereas the original parquet file result was (3.996998)
3. large files: Repartitioning set as 10 gave the best result (3.842066) whereas the original parquet file result was (5.938744)

pq_max_income:

1. small files: Setting replication factor as 2 gave the best result (1.407076) whereas the original parquet file result was (1.818207). Slight improvement but not that significant.

2. medium files: Setting replication factor as 2 gave the best result (1.078947) whereas the original parquet file result was (1.166410). Slight improvement but not that significant.

3. large files: Sorting by columns gave the best result (4.182736) whereas the original parquet file result was (5.182013).

pq_anna:

1. small files: The original parquet file result was the best (0.056406) whereas setting replication factor as 2 gave the next best result (0.056716)

2. medium files: Setting replication factor as 2 gave the best result (0.102286) whereas the original parquet file result was (1.089115)

3. large files: Sorting by columns gave the best result (0.545430) whereas the original parquet file result was (4.915585)

Observation comparing original parquet files and optimization techniques:

1. Setting replication factor as 2 worked best with small files as well as medium or majority of queries
2. Sorting by columns worked best with large files for majority of queries
3. Increasing repartitioning to 10 only gave best results for medium and large files run on pq_avg_income query

Comparing all the optimization technique results in the table for Part 3

pq_avg_income:

1. small files: Setting replication factor as 2 gave the best results (1.459219) and the worst result was for reaptitioning by 10 (3.842066)

2. medium files: Repartitioning set as 10 gave the best results (1.074595) followed closely by setting replication factor as 2 (1.128406) and the worst result was for sorting by columns (3.044242)

3. large files: Repartitioning set as 10 gave the best results (3.842066) and the worst result was for sorting by columns (5.457127)

pq_max_income:

1. small files: Setting replication factor as 2 gave the best results (1.407076) and the worst result was for repartitioning set as 10 (2.913784)
2. medium files: Setting replication factor as 2 gave the best results (1.078947) followed closely by repartitioning set as 10 (1.135798) and the worst result was for sorting by columns (1.496937)
3. large files: Sorting by columns gave the best results (4.182736) and the worst result was for setting replication factor as 2 (5.970955)

pq_anna:

1. small files: Performance was great for all the optimization techniques but setting replication factor as 2 gave the best results (0.056716)
2. medium files: Setting replication factor as 2 gave the best results (0.102286) followed closely by repartitioning set as 10 (0.110339) and the worst result was for sorting by columns (1.069384)
3. large files: Sorting by columns gave the best results (0.545430) and the worst result was for setting replication factor as 2 (4.186236)

Observation comparing among all the optimization techniques:

1. Setting replication factor as 2 gave the best results for small files and medium files
2. Sorting by columns worked best with large files for majority of queries
3. Increasing repartitioning to 10 only gave best results for medium and large files run on pq_avg_income query

Overall when I used the below setting all together, i.e.,

1. First, sorting the large file and storing it as a parquet file
2. Second, decreasing the replication factor to 2 and increasing the repartitioning to 10

gave me the best possible results:

Repartitioning set as 10, replication factor set as 2 and benchmarks run on large sorted and repartioned files					
pq_avg_income	people_large_repar_rep. parquet	25	3.086998	3.478473	4.909566
pq_max_income	people_large_repar_rep. parquet	25	3.184307	3.369428	5.280875
pq_anna	people_large_repar_rep. parquet	25	0.425662	0.395398	0.608480