Project I-Task2

Tianchi YU
319877 - Database System Course
EPFL

March 30, 2020

Notes

- All sql tests offered by the project are passed by the scala.
- We will discuss about different execution models and data layouts
- Execution models: volcano(tuple-at-a-time),operator-at-a-time(column-at-a-time with early materialization),block-at-a-time(vector-at-a-time) and late-operator-at-a-time(late materialization)
- Data layouts: NSM(row store),DSM(column sotre),PAX(pax store)
- All sql queries are the same for different models or data layouts
- The measurements were conducted on Windows 10 , with i7-7700HQ CPU 2.8 GHz Intel Core and 8GB memory.

So we are going to see the execution time for differnt gourps of queries in the program (sql tests) . . .

I. Warmup - Queries

Only One SQL Query

```
select
    l_returnflag,
    l_linestatus,
    sum(l_quantity) as sum_qty,
    sum(l_extendedprice) as sum_base_price,
    sum(l_extendedprice * (1 - l_discount)) as sum_disc_price,
    sum(l_extendedprice * (1 - l_discount) * (1 + l_tax)) as sum_charge,
    avg(l_quantity) as avg_qty,
    avg(l_extendedprice) as avg_price,
    avg(l_discount) as avg_disc,
    count(*) as count_order
from
    tpch0_001_lineitem
where
    l_shipdate <= date ’1998-12-01’ - interval ’90’ day
group by
    l_returnflag,
    l_linestatus
order by
    l_returnflag,
    l_linestatus
;
```

i. Result

Table 1: Execution time of Warmup-Queries over different execution model-data layout

Execution Model	Data Layout	Execution Time
Volcano	NSM	2 s 33 ms
	DSM	566 ms
	PAX	611 ms
Operator-at-a-time	NSM	816 ms
	DSM	443 ms
	PAX	468 ms
Block-at-a-time	NSM	813 ms
	DSM	643 ms
	PAX	944 ms
Late-operator-at-a-time	NSM	1 s 13 ms
	DSM	975 ms
	PAX	1 s 203 ms

ii. Observation

As we can see, this is a basic test for all models. The NSM of Volcano is the slowest one, and the DSM is obviously the best, compared with NSM and PAX, the operator-at-time performs better than Volcano and Block-at-a-time since all data are fit into memory and thus it does not has next() function calls overhead.
The late-operator-at-a-time should be faster than else, however in this case, it's not the ideal case.

II. Micro - Queries

In this part, we have 33 micro-tests, which contain the basic queries of 'scan', 'filter', 'projection', 'aggregate', 'join'.

i. Result

Table 3: Execution time of Micro-Queries over different execution model-data layout

Execution Model	Data Layout	Execution Time
Volcano	NSM	910 ms
	DSM	327 ms
	PAX	341 ms
Operator-at-a-time	NSM	647 ms
	DSM	325 ms
	PAX	294 ms
Block-at-a-time	NSM	537 ms
	DSM	331 ms
	PAX	275 ms
Late-operator-at-a-time	NSM	514 ms
	DSM	282 ms
	PAX	257 ms

ii. Observation

Because all the queries are simple ones, so their execution time could explain the basic principles behinds the scenerio, like: the operator-at-a-time is always better than volcano; since the size of the data is small, so the block-at-a-time preforms as good as operator-at-a-time; and DSM should be the best data layout while PAX sometimes is faster than DSM because of the small number of minipages; we can see the late materialization for the operator-at-a-time shows its good performance in this case.

III. Complex - Queries

In this part, we have 10 complex-tests, which are mixed with all queries of 'scan', 'filter', 'projection', 'aggregate', 'join' and 'sort'.

i. Result

Table 5: Execution time of Warmup-Queries over different execution model-data layout

Execution Model	**Data Layout**	**Execution Time**
Volcano	NSM	1 s 903 ms
	DSM	1 s 222 ms
	PAX	1 s 147 ms
Operator-at-a-time	NSM	1 s 303 ms
	DSM	779 ms
	PAX	951 ms
Block-at-a-time	NSM	3 s 298 ms
	DSM	2 s 887 ms
	PAX	2 s 695 ms
Late-operator-at-a-time	NSM	8 s 939 ms
	DSM	11 s 83 ms
	PAX	13 s 156 ms

ii. Observation

The regular of operator-at-a-time and volcano are similary with the warmup query, while the block-at-a-time is much slower than the Volcano.
The late-operator-at-a-time is still a innormal situation because of the problem of the program. So I will talk about it later in the conclusion part.

IV. In Total

`Execution time for all queries`

```
Data Layout  volcano  operator-at-a-time   block-at-a-time  late-operator-at-a-time
---------------------------------------------------------------------------------
   NSM        4s 864ms      2s 766ms            4s 648ms            10s 466ms
   DSM        2s 115ms      1s 547ms            3s 861ms            12s 340ms
   PAX        2s  99ms      1s 713ms            3s 914ms            14s 616ms
```

V. Conclusion

v.i NSM vs DSM vs PAX

NSM means Row data layout, DSM means Columnar data layout and PAX means Partition Attributes Across data layout.
As we know, for every scenario, DSM is the faster, and PAX is slower than DSM.PAX uses a mini page method in the page, and cuts records into different mini pages. In volcano and block-at-a-time PAX has an additional overhead for materialization of the minipages. Every next() call from parent requires PAX to form Tuple from minipages. There are two situations for PAX to make execution slower : decreasing page size(too small) or increasing page size(too big) , since both lead to increase in search time for select operator (too many pages or too many tuples in one page). In the ideal situation, if the page size is 1, PAX will basically become NSMwith slower performance.
So, It can be seen that the format of PAX is actually a compromise between NSM and DSM. When scanning according to a certain column, we can conveniently scan sequentially in the mini page and make full use of the cache. And when we need to access multiple attributes to get the final tuple, we just need to read related data between mini pages in the same page.

v.ii Operator-at-a-time vs Block-at-a-time

Basicly, the performance of operator-at-a-time is better than block-at-a-time. One thing important for block-at-a-time is the vector size. If we increase the vector size, the performance of block-at-a-time/vector-at-a-time execution model will approach operator-at-a-time/column-at-a-time execution model, obviously.

v.iii Early-Operator vs Late-Operator

As we know, the performance of late-materialization should be better than the early-materialization. When I compared the different cases in the complex queries of ececution model of late-operator-at-a-time, not all of them are slower than operator-at-a-time. There are several cases innormal because of my programming way.