Data Warehouse Optimization Report

Tymon Bielski & Michał Balcerowicz

Aim of lab work

The aim of the task is to show issues concerning various physical cube models and aggregation design.

Preliminary Assumptions

Size of the database (Data Warehouse): 26256.00 MB

Number of rows in monitoring fact table (our main): 30196006

Testing environment: Measurements were taken on X-com "G4MER" gaming pc equipped with a 13th Generation Intec Core i7-13700F, 32GB of RAM and 930GB of Internal SSD Storage (WD Blue SN570) (120GB free) running Windows 11. To evaluate the processing time of a cube, we used SQL Server Management Studio 20 with extension SQL Server Profiler 20. During the measurements, the active applications on the pc were 2 instances of SSMS (analysis and database services), a web browser with opened instructions (notably using 4GB of ram), Visual Studio 2023, Discord and Spotify (both using negligible system resources - >400mb ram, >0.2% CPU utilisation).

Theoretical Assumptions

	MOLAP	HOLAP	ROLAP
Querying time	Short	Moderate (short with well designed aggregations)	Long
Processing time	Long	Moderate (if no aggregations are designed, it will be short)	Short
Total size	Big (size of the measure group is much smaller if no aggregations are designed for them)	Moderate	Small

Testing

The testing consisted of: query execution times for different models, with and without defined aggregations and cube processing times with the same testing settings.

Used queries

Total Ski lift use number for the January of 2022 & 2023

```
SELECT
    {[Measures].[Number of total usages]} ON COLUMNS,
    {
        ([Date].[Hierarchy].[Year].&[2023].&[January]),
        ([Date].[Hierarchy].[Year].&[2022].&[January])
    } ON ROWS
FROM
    [Warehouse]
```

Difference in total use number of different ski lift types in 2022 & 2023

```
WITH
MEMBER [Measures].[Usage Difference] AS
    ([Measures].[Number of total usages], [Date].[Year].[2022]) -
    ([Measures].[Number of total usages], [Date].[Year].[2023])
SELECT
    { [Measures].[Usage Difference] } ON COLUMNS,
    {
        [Ski Lift].[Lift Type].[Lift Type].ALLMEMBERS
    } ON ROWS
FROM [Warehouse]
```

Average time spent of a ski lift by age group in 2019

```
SELECT
    {[Measures].[Average weighted ride time]} ON COLUMNS,
    NON EMPTY{([Customers].[Age Range].[Age Range].ALLMEMBERS)} ON ROWS
FROM
    [Warehouse]
WHERE ([Date].[Hierarchy].[Year].&[2019])
```

To reach believable and optimal results I took 10 samples for each Test. The results are presented in the table below.

Cube processing	Molap		Rolap		Holap
	No aggregation	aggregation	No aggregation	aggregation	
1	39326	42990	293	291	295
2	37340	43905	271	280	299
3	37742	44400	275	277	267
4	37366	43641	284	262	272
5	37527	45008	277	253	276
6	39826	42818	283	312	324
7	38566	42867	233	289	264

8	38318	44144	252	240	257
9	37243	43156	278	266	309
10	37880	44161	259	265	284
Query 1					
1	128	14	766	561	732
2	106	2	596	531	352
3	102	2	785	996	562
4	103	3	1012	700	814
5	105	3	1354	584	573
6	105	2	957	1087	1152
7	107	2	610	582	727
8	106	2	639	1029	531
9	107	2	465	501	574
10	105	2	800	704	564
Query 2					
1	149	2	653	580	636
2	132	2	368	633	555
3	138	2	587	547	602
4	139	2	499	481	555
5	134	1	576	573	460
6	134	5	631	552	678
7	134	2	631	545	592
8	139	2	565	587	576
9	136	2	509	732	530
10	133	1	584	662	576
Query 3					
1	117	2	644	708	577
2	111	2	546	549	529
3	103	2	594	515	527
4	107	2	551	577	650
5	103	2	534	557	554
6	104	2	538	574	796
7	103	2	498	435	564
8	102	2	574	532	621
9	108	6	611	457	554
10	111	2	506	522	457

Aggregation settings

The aggregation settings for the cube were set as follows:

- Year and Month to full
- Customer age range to full
- Lift number and lift manufacturer to full

- All FKs and BKs to default
- Rest to none

In summary, all attributes used in queries were set to full, unused, but required for correct working of the cube to default and the rest to none. The performance gain reached 49% and resulted in extra 308mb of partition.

Cache clearing

During the measurements the cache was cleared after every query execution to prevent query caching from influencing the results. The code used for doing so was the same presented on enauczanie:

```
<ClearCache xmlns=http://schemas.microsoft.com/analysisservices/2003/engine>
   <Object>
      <DatabaseID>your analytical database ID</DatabaseID>
   </Object>
</ClearCache>
```

Outliers

With the collected data I decided to not discard any due to being an outlier. All collected data was used for calculating averages used in the Discussion segment of this report. There were a few values that caught my attention (like 1354 or 465 for Rolap query 1) but no data points were larger than double the average, nor smaller than half of the average. This fact solidified my decision not to classify any outliers.

Discussion

Average Cube Processing Time

Molap	Rolap	Holap
38113	270.5	284.7

As expected, the cube processing time is by far the largest for Molap, taking nearly 150 times longer than Rolap or Holap. The Holap time is also longer than Rolap, though their difference is quite small, and I was personally expecting a bigger difference. Overall this mostly confirms the assumptions from the table earlier in this report.

Average Query execution time

Molap	Rolap	Holap
107.4	798.4	658.1
136.8	560.3	576
106.9	559.6	582.9

Again, as expected, queries execute a lot faster for Molap than Rolap (nearly 8 times faster in some cases). As Molap stores all the data in the cube and does not need to execute additional sql queries to the database this result was to be expected and agrees with the assumptions from the beginning of this report. What was not expected is that Holap was faster from Rolap in only 1 of 3 queries. This result is most surprising to me and The reasons for it remain unknown to me.

The influence of aggregation

	Molap	
	No aggregation	Aggregation
Cube processing	38113	43709
Query 1	107.4	3.4
Query 2	136.8	2.1
Query 3	106.9	2.4

When I first saw that a query executed in 2 milliseconds I though I forgot to clear the cache, but I cleared it again, double and triple checked, and the results still came out the same. The aggregation I did was very methodical and heavily optimised, so in the end, it makes sense that the resulting time reduction was so large. As expected from theory, the cube processing took over 5 seconds longer for Molap with aggregation. The lower query execution time was also an expected outcome.

	Rolap	
	No aggregation	Aggregation
Cube processing	270.5	273.5
Query 1	798.4	727.5
Query 2	560.3	559.2
Query 3	559.6	542.6

Again, as expected, Aggregation has increased cube processing time and decreased query execution time. However, due to not all data being held in the Rolap, the difference is much smaller in both cases than in Molap. The cube processed only 3 milliseconds longer on average with aggregations and the time difference in query execution speed is negligible.

When it comes to aggregation, it makes little difference to Rolap models (nearly unnoticeable without a profiler), but seems immensely useful for Molaps, which execute queries frequently.