



Users working together

Connect. Share. Learn.

Designing Effective Aggregations in Analysis Services 2008

Chris Webb
Crossjoin Consulting Limited
chris@crossjoin.co.uk



Agenda

- What are aggregations and why should I build them?
- How Analysis Services answers MDX queries
- The importance of good dimension design
- The Aggregation Design Wizard
- Usage-based optimisation
- Designing aggregations manually
- Influence of cube design on aggregation usage



European PASS Conference 2009

Why build aggregations?

- Query performance is the top priority for all OLAP solutions
- Aggregations are the single most important feature in Analysis Services regarding query performance
 - Though not always useful, as we'll see...
- Every cube with a fact table of more than a few million rows should benefit from aggregations



European PASS Conference 2009

What is an aggregation?

- It's a copy of the data in your fact table, pre-aggregated to a certain level
 - Created when the cube is processed
 - Stored on disk
- Think of it as being similar to the results of a GROUP BY query in SQL
- It makes queries fast because it means SSAS does not have to aggregate as much data at query time



European PASS Conference 2009

Visualising aggregations

- You can actually see an aggregation if you:
 - Create a cube using ROLAP storage
 - Build aggregations
 - Process the cube
 - Look in your relational data source at the indexed view or table created



European PASS Conference 2009

The price you pay

- Aggregations are created at processing time
- Therefore building more aggregations means processing takes longer
- Also increases disk space used by the cube
- But SSAS can build aggregations very quickly
- And with the 'right' aggregation design, relatively little extra processing time or disk space is needed



European PASS Conference 2009

Can I pre-aggregate everything?

- **NO!**
- Leads to the problem of database explosion, where the aggregations become bigger than the original data
- Processing time would be way too long
- There is no need: SSAS can reuse aggregations
- In fact, too many aggregations can be bad for query performance



European PASS Conference 2009

Database explosion

10	20
30	40



10	20	30
30	40	70
40	60	100

5 extra values needed to hold all possible aggregations on a table that had only 4 values in it originally!

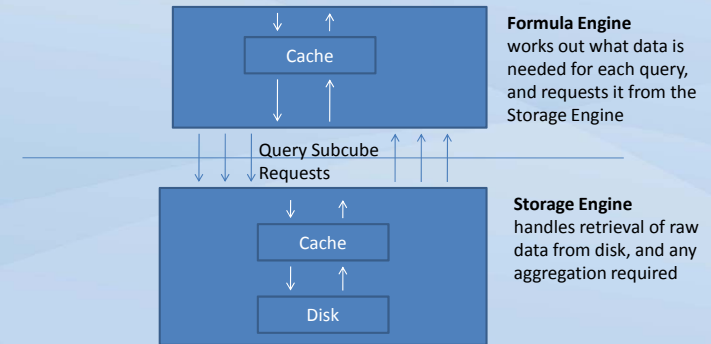


European PASS Conference 2009

How Analysis Services answers queries

MDX Query In

Cellset Out



European PASS Conference 2009

When aggregations are used

- Aggregations are only useful when the Storage Engine has to fetch data from disk
- Aggregations will not be used if the data is in the Storage Engine cache
- Aggregations may not be useful if the cause of query performance problems lies in the Formula Engine



European PASS Conference 2009

Profiler

- You can use SQL Profiler to see what is happening internally when a query is run
- The following events are useful:
 - **Query Begin/End** – the start and end of the query
 - **Progress Report Begin/End** – for all reads from partitions or aggregations
 - **Get Data From Aggregation** – appears every time an aggregation is used
 - **Query Subcube** or **Query Subcube Verbose** – show details of the requests made to the Storage Engine by the Formula Engine



European PASS Conference 2009

Aggregation designs

- Each measure group can have 0 or more **aggregation design** objects associated
 - They are what gets created when you run either of the Aggregation Design wizards
 - They detail which aggregations should be built
 - Remember to assign designs to partitions!
- Each partition in a measure group can be associated with 0 or 1 aggregation designs
- Aggregations are built on a per-partition basis



European PASS Conference 2009

Aggregation design methodology

1. Ensure your dimension design is correct
2. Set all properties that affect aggregation design
3. Run the Aggregation Design Wizard to build some initial aggregations
4. Perform Usage-Based Optimisation for at least a few weeks, and repeat regularly throughout the cube's lifetime
5. Design aggregations manually for individual queries when necessary



European PASS Conference 2009

Dimension design

- Dimension design has a big impact on the effectiveness of aggregations
- Your dimension designs should be stable before you think about designing aggregations
- Three important things:
 - Delete any attributes that won't ever be used
 - Check your attribute relationships to ensure they are set optimally
 - Build any natural user hierarchies you think might be needed



European PASS Conference 2009

Attribute relationships

- Attribute relationships allow SSAS to derive values at higher granularities from aggregations built at lower granularities
- The more 'bushy' the attribute relationships on your dimension, the more effective aggregations will be
- Flat attribute relationships mean aggregation design is much harder



European PASS Conference 2009

Properties to set

- Before starting aggregation design, you should set the following properties:
 - **EstimatedRows** property for partitions
 - **EstimatedCount** property for dimensions
 - **AggregationUsage** property for attributes of cube dimensions
- Setting these correctly will ensure the aggregation design wizards will do the best job possible



AggregationUsage

- The **AggregationUsage** property has the following possible values:
 - **Full** – meaning every aggregation will include this attribute
 - **None** – meaning no aggregation will include this attribute
 - **Unrestricted** – meaning an aggregation *may* include this attribute
 - **Default** – means the same as Unrestricted for key attributes, or attributes used in natural user hierarchies



The Aggregation Design Wizard

- The Aggregation Design Wizard is a good way to create a 'first draft' of your aggregation design
- Do not expect it to build every aggregation you'll ever need...
- ...it may not even produce any useful aggregations at all!
- Increased cube complexity from AS2005 onwards means it has a much harder job



The Aggregation Design Wizard

- The first few steps in the wizard ask you to set properties we've already discussed
- However, the **Partition Count** property can only be set in the wizard
- It specifies the number of members on an attribute that are likely to have data in any given partition
- For example, if you partition by month, a partition will only have data for one month



Set Aggregation Options step

- This is where the aggregations get designed!
- Warning: may take minutes, even hours to run
- Four options for working out which aggregations should be built:
 - **Estimated Storage Reaches** – build enough aggregations to fill a certain amount of disk
 - **Performance Gain Reaches** – build aggregations to get x% of all possible performance gain from aggregations.
 - **I Click Stop** – carry on until you get bored of waiting
 - **Do Not Design Aggregations**



Set Aggregation Options step

- Best strategy to use is:
 - Choose **I Click Stop**, then see if the design process finishes quickly and if so, how many aggregations are built and what size
 - If you have a reasonable size and number of aggregations (build no more than 50 aggregations here), click **Finish**
 - Otherwise, click **Stop** and **Reset** and choose **Performance Gain Reaches** and set it to 30%
 - If only a few, small aggregations are designed, increase by 10% and continue until you are happy



Usage-Based Optimisation

- Usage-Based Optimisation involves logging Query Subcube requests and using that information to influence aggregation design
- To set up logging:
 - Open SQL Management Studio
 - Right-click on your instance and select Properties
 - Set up a connection to a SQL Server database using the **Log\Query Log** properties
 - The **QueryLogSampling** property sets the % of requests to log – be careful, as setting this too high might result in your log table growing very large very quickly



Usage-Based Optimisation

- You should log for at least a few weeks to get a good spread of queries
- Remember: any changes to your dimensions will invalidate the contents of the log
- Next, you can run the Usage-Based Optimisation wizard
- This is essentially the same as the Aggregation Design wizard, but with log data taken into account (you can also filter the data in the log)



Manual aggregation design

- You will find that the wizards do not build the aggregations needed for specific queries
 - Running the wizard is a bit hit-and-miss
 - The wizards may never build certain aggregations, for example they may think they are too large
- In this case you will need to design aggregations manually
- BIDS 2008 allows you to do this on the Aggregations tab of the cube editor
- BIDS Helper (<http://www.codeplex.com/bidshelper>) has better functionality for this



European PASS Conference 2009

Manual aggregation design

- To design aggregations manually:
 - Clear the cache
 - Start a Profiler trace
 - Run your query
 - Look for Query Subcube events that take more than 500ms
 - Build aggregations at the same granularity as the Query Subcube events
 - Save, deploy then run a ProcessIndex



European PASS Conference 2009

Redundant attributes

- Often more than one attribute from the same dimension will be used in a subcube request
- If those attributes are connected via attribute relationships, only include the lowest one in the aggregation
 - Eg If Year, Quarter and Month are selected, just use Month
- This will ensure that the aggregation can be used by the widest range of queries



European PASS Conference 2009

Influence of cube design

- There is a long list of features you might use in your cube design that affect aggregation design and usage
- Mostly they result in queries executing at lower levels of granularity than you'd expect
- This means aggregations are less likely to be hit, and less useful



European PASS Conference 2009

Many-to-many relationships

- Aggregations will never be used on an intermediate measure group in a many-to-many relationship, if the measure group is only used as an intermediate measure group
- M2M queries are resolved at the granularity of the key attributes of all dimensions common to the main measure group and the intermediate measure group
- Therefore aggregations are less likely to be hit on the main measure group



European PASS Conference 2009

Semi-additive measures

- Queries involving semi-additive measures are always resolved at the granularity attribute of the Time dimension
- Therefore any aggregations must include the granularity attribute of the Time dimension if they're to be used



European PASS Conference 2009

Partitions

- It's pointless to build aggregations above the granularity of the attribute you're using to slice your partitions
 - Eg, if you partition by Month, don't build aggregations above Month level
 - Nothing stops you doing this though!
- Do this and you get no performance benefit and limit the queries the aggregations can be used for



European PASS Conference 2009

Parent-child hierarchies

- You cannot build aggregations that include parent-child hierarchies
- That also means that the levels that appear 'within' the parent-child structure cannot be used in an aggregation
- You should build aggregations on the Key Attribute of your dimension instead
 - Very important if you have a DefaultMember or if IsAggregatable=False



European PASS Conference 2009

MDX Calculations

- Calculated members and MDX Script assignments are very likely to request data for different granularities from the one you're querying
- Don't assume – check what's happening in Profiler!



Other things to watch for

- **Measure expressions** cause queries to be evaluated at the common granularity of the two measure groups involved
- **Unary operators other than +** and **Custom Rollups** will also force query granularity down
- Do not include attributes from **non-materialised reference dimensions** in aggregations as this may result in incorrect values being returned



Questions?



Thank you!

