Partitioned User-Defined Aggregates

When you create or edit a user-defined aggregate (UDA), you can specify a dimensional attribute to use as a partitioning key. If you do so, instances of the aggregate will be partitioned, with one partition created for each key value. Partitioning an aggregate can result in faster query performance.

- When to use partitioning
- Procedure
- Result
- More information

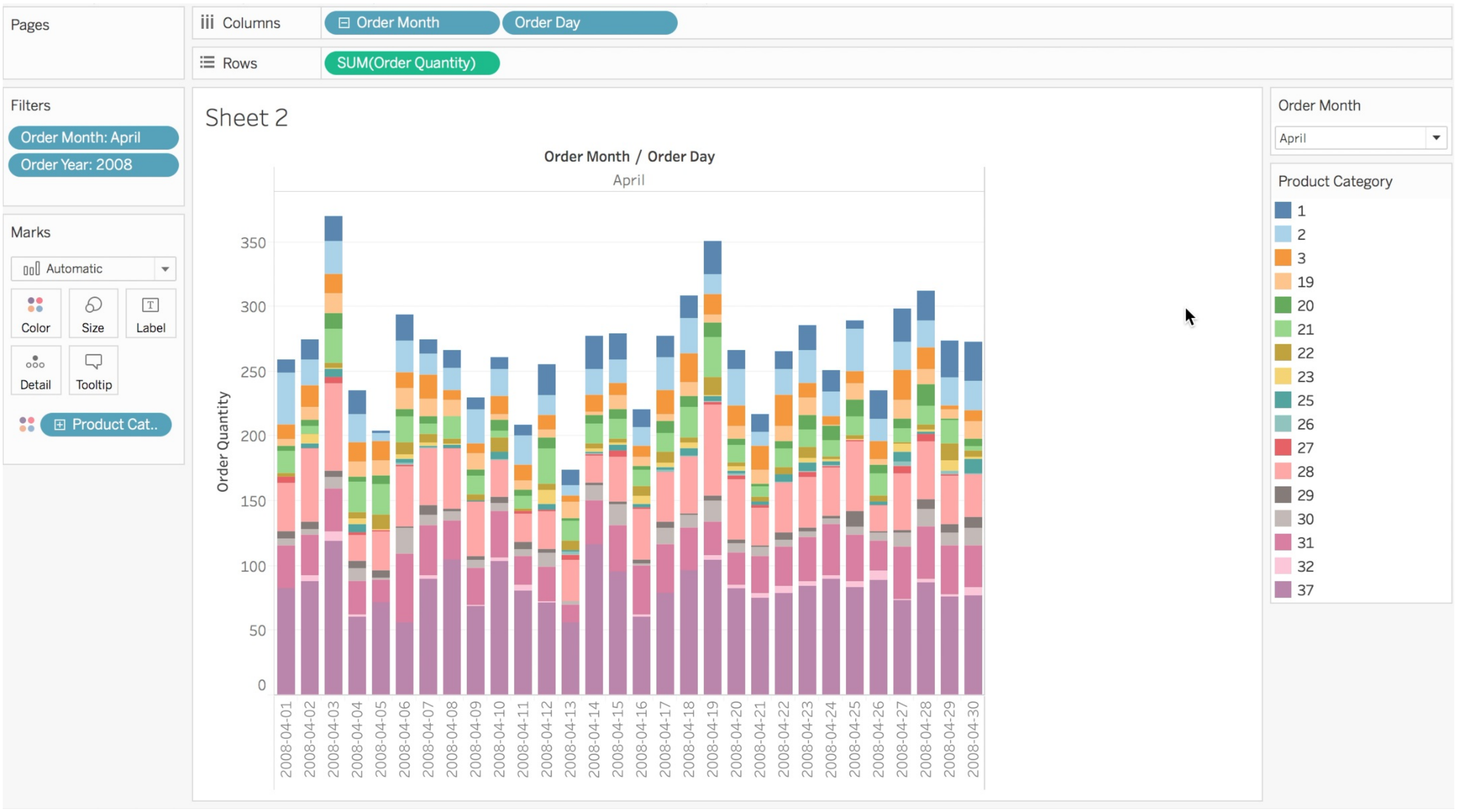
Be aware of the following:

- AtScale supports aggregate partitioning with Google BigQuery, Snowflake, and Databricks data warehouses; partitioning with InterSystems IRIS is not currently supported.
- For Google BigQuery, aggregate tables can be partitioned by Integer, Date, and DateTime columns.

When To Use Partitioning

For example, suppose that you look at the SUM of Order Quantity for all product categories per Order Day, Order Month, and Order Year. In Tableau, you might have a visualization such as this one.

Figure 1. Tableau workbook visualization



Because you need to issue the query for this visualization and similar queries often, looking at the data for particular months and years, you define an aggregate that includes these dimensions and measures:

Dimensions

Order Day, Order Month, Order Year, Product Line, Product Category, Product Name

Measures

Average Sales per Order, Order Quantity, Sales Amount

When you refresh this visualization, filtering in this case on Order Month = "April" and Order Year = "2008", the query uses the aggregate, but you notice that the refresh takes longer than you would like. Given the very large size of your fact dataset, which could contain records from thousands of days and potentially thousands of orders per day, instances of the aggregate that you defined could also be very large. The query issued by AtScale requires a full scan on the aggregate to find the records that meet the filtering criteria.

To try improving the speed of the query, you can edit the aggregate's SML to select a column (in this case, Order Month) to use as a partitioning key. When the AtScale engine builds instances of the aggregate, a partition is created for each value in the key. So, in your aggregate, there would be one partition per month. After you redeploy the catalog that contains the model and you refresh the data visualization in Tableau, the query runs against only the partition for the key value that you filter on, which in the current state of your visualization is "April". Because the partition contains only a subset of the data that is in the aggregate as a whole, the query should run faster. Of course, you'll notice the improvement more when you partition very large tables than when you partition relatively smaller ones.

Partitioning user-defined aggregates is a good idea when the combination of dimensions in an aggregate would lead to aggregate instances with very high cardinalities. If you decide to partition a UDA, ensure that queries against the aggregate always use a WHERE clause that filters only on the partitioning key.

Procedure

1. Make sure the following engine settings are set to **True** (default value). To access these settings, open the main menu and select **Settings**, then click **Engine** in the **Settings** panel.
    - `TABLES.CREATE.PARTITIONS.ENABLED`
    - `AGGREGATES.CREATE.PARTITION.USERDEFINEDAGGREGATE.ENABLED`
2. Proceed with the steps described in Defining Aggregates Yourself.
3. When editing the `aggregates` property in the model's SML, be sure to include the `partition` property (within the `attributes` property), and to specify whether the corresponding partition should be defined on the key column, name column, or both. For more information, see Models.

Performance Best Practice: MDX tools like Excel specify filters using attribute key bindings, while SQL and DAX-based tools like Tableau use attribute name bindings. You should be aware of your user community's tool usage so you can adopt a partitioning strategy that maximizes their query performance. If a model services a mix of BI tools, AtScale recommends partitioning by both name and key columns, or by using the same physical column for the AtScale key and name attribute bindings.

Result

Another engine configuration setting (`TABLE.CREATE.PARTITIONS.MAXIMUMESTIMATEDNUMBEROFPARTITIONS`) specifies the maximum number of partitions to allow for a user-defined aggregate. The AtScale engine refers to this setting after you deploy a catalog in which a partitioned UDA is defined on a model. The engine estimates the cardinality of the instances that would be built from the aggregate definition. If the estimate is at or below the configured maximum allowed number of partitions, the engine will build the UDA. However, if the estimate is above the configured maximum, the engine will not build the UDA and will issue an error message to say that the build of the UDA failed.

If the engine fails to build one of your partitioned UDAs, there are three different actions that you can try for a successful build:

- Use a partitioning key that has a cardinality that is lower than the configured maximum allowed number of partitions.
- Increase the maximum allowed number of partitions. (However, AtScale recommends that you set the maximum no higher than 1000.)
- Edit the user-defined aggregate by removing the partitioning key.

Instances of partitioned UDAs can be rebuilt either fully or incrementally. If an attribute that is used as a partitioning key is updated, the AtScale engine performs a full rebuild of the aggregate, creating a new instance and new partitions, whether or not you have specified to use incremental builds.

To monitor partitioned UDAs, you can use the **Aggregates** page in the Design Center, as you would for all other aggregate types. There are no special considerations that you need to be aware of when monitoring. To make troubleshooting easier, however, the **Aggregates** page displays the partitioning key for each partitioned user-defined aggregate.

More Information

Engine Settings for User-Defined Aggregates Only