Qlik Sense Performance Cheat Sheet

The primary purpose of this cheat sheet is to provide a series of good practices to help ensure that Qlik applications are developed as efficiently, productively and with as low risk to the infrastructure environment as possible. This does not cover a range of other good practices such as user experience, data visualisation and other more advanced scripting and expression capabilities.

Performance good practice in the load script and data model

- Stage source data in QVD files and then load from the QVD as this will avoid strain on the source systems and possibly network bandwidth as well as be a lot quicker, safer and more productive
- Break out different data source load process into different script sections and use an Exit Script section, which can be easily moved to test each of your load processes separately
- If possible, develop with a meaningful subset of data using Where clauses and/or Exists clauses in the load process to ensure you maintain relevant key matches
- Avoid trying to create overly large applications covering multiple use cases, it is far more efficient to create several smaller applications each covering a discrete user journey
- Remove synthetic keys and where possible and circular references
- Remove (or comment out a better practice) all unused fields from the load
- Remove or simplify time stamps (for example you don't need 1/100th of a second so you could use the ceil function to round up to the nearest minute) or highly unique system fields
- Use Limited Load in debug mode to test your logic of the script before running a full reload or use the First function to limit the load
- Use Autonumber to replace text string based key fields with more efficient integers
- Remove, join or concatenate unnecessary snow flaked tables
- Avoid using nested if statements – alternatives are mapping tables in the load script and pick (match functions and Set Analysis with flag fields in the User Interface
- Consider the use of incremental loads for large data sets that need to be regularly updated, this will reduce the load on the source system and speed up the overall load process

Performance good practice in the user interface

- Avoid the use of Calculated Dimensions in the user interface and be careful with the AGGR function, which can be very resource intensive
- Complex calculations (and calculated dimensions) should be done as part of the load process rather than in the front end expression editor
- For objects in the User Interface that are particularly resource intensive, consider the use of calculation conditions where the user selects a subset of data to analyse first
- Use Search Exclude and then explicitly add in the fields using Search Include in the load script that you want users to be able to search, this will improve user experience and search performance
- Avoid expressions that calculate using fields from different tables as this can be very resource intensive over a large number of rows. Look to collapse the fields into a single table in the data model where possible

Related resources:
For many good practices and more information then Qlik Continuous Classroom is a great resource
Performance and Optimization Best Practices in QlikView & Qlik Sense
QlikView Best Practice Guidelines (a lot still relevant to Qlik Sense!)
Part of the Qlik Diagnostic Toolkit, this application checklist covers much of the above and more
For an in depth optimisation framework and advice see the Qlik Application Performance Optimization Strategies post

bca@qlik.com
2018