**Dependencies**

This document defines the scheme of communication between the different classes. It might be useful in the final merging phase.

**Command**

Communicates with the user and initiates communications with SPARK through ConnectionSPARK. It also transmits all commands from the user to ExecuteCommand which handles all commands.

**ConnectionSPARK**

Connection with Spark

**ExecuteCommand**

This class handles all queries from the user. The time series can be directly created or retrieved from storage. They are referenced in a map which lists all the time series currently active (in the workspace).

**Transformations**

All transformations act on loaded data from storage. They are called by the user and handled by ExecuteCommand.

**Indexing**

Indexing is done through the RTT. The indexing scheme is controlled through the interface with the user. This means that ExecuteCommand calls the class Database Index which provides the interface to the other indexing classes. It builds the indexes on key/value pairs as a RTree. The tree is build using the nearestneighbor as metric.

**Compression**

Compression is defined by the user through 3 parameters: the degree, the minimum segment size and the mean segment error allowed between the stored and the real segment. Everything in this class is defined by the user in ExecuteCommand.

**Storage**

Storage is done through Scala and SPARK. The api of storage is exclusively called from ExecuteCommand and store data as key/value where values are stored as DBTypes. Data are stored using interval indexes.