

|  |  |
| --- | --- |
| Starting & accessing | |
| ./arangod /path/to/my/db | start server |
| ./arangod --console --log error /path/to/my/db | start emergency console (do not use with a db which has a server attached to it!) |
| http://localhost:8529/\_admin | access admin front end in browser |
| ./arangosh | start ArangoDB shell |

|  |  |
| --- | --- |
| arangod frequently used options | |
| --log | set log level: fatal, error, warning, info, debug, trace |
| --server.http-port *host:port* | set address and port for HTTP-Clients |
| --daemon | run as daemon/background process |

|  |  |
| --- | --- |
| Database methods in arangosh | |
| db.\_create(*collection-name*, *properties*) | create collection (with properties) |
| db.\_collection(*collection-name*|*collection-id*) | get collection |
| db.\_collections() | list all collections |
| db.*collection-name* | get a collection by name |
| db.\_drop(collection-name|collection-id) | drop collection with indexes |
| db.\_truncate(*collection-name*|*collection-id)* | remove collection, keep indexes |

|  |  |
| --- | --- |
| Collection methods in arangosh | |
| collection.drop() | drop collection with indexes |
| collection.truncate() | remove documents, keep indexes |
| collection.properties() | get all document properties |
| collection.properties(properties) | change property |
| collection.figures() | get all collection figures |
| collection.load() | load collection into memory |
| collection.unload() | start to unload a collection |
| collection.rename(new-name) | rename collection to new-name |

|  |  |
| --- | --- |
| Document methods in arangosh | |
| collection.document(*document*) | get document by identifier |
| collection.save(*data*) | create new document |
| collection.replace(*document*, *data*) | replace existing document |
| collection.remove(*document*) | remove document |
| db.\_document(*document*|*document-handle*) | get document by identifier|handle |
| db.\_replace(*document*|*document-handle*,*data*) | replace existing document |
| db.\_remove(*document*) | remove document |

|  |  |
| --- | --- |
| Edges in arangosh | |
| edge-collection.save(*from*, *to*, *document*) | save new edge |
| edge-collection.edges(*vertex*) | find edges from (outbound) to (inbound) vertex |
| edge-collection.inEdges(*vertices*) | find all edges ending in (inbound) |
| edge-collection.outEdges(*vertices*) | find all edges starting from (outbound) |

|  |  |
| --- | --- |
| Queries in arangosh | |
| collection.all() | select all documents and return cursor |
| collection.byExample(*example*) | select all documents that matches the given *example* |
| collection.firstExample(*example*) | select the first document that matches the given *example* |
| collection.range(*attribute*,*left*,*right*) | select all documents with attribute >= *left* and < *right* |
| collection.count() | returns the number of living documents in the collection |
| collection.toArray() | convert the collection into an array of documents (not for production!) |

|  |  |
| --- | --- |
| Geo Queries in arangosh | |
| collection.near(*latitude*,*longitude*) | get documents near the given coordinates |
| collection.within(*latitude*,*longitude*, *distance*) | get all documents within a radious of *distance* around the given coordinates |
| collection.geo(*location*) | the next near or within operator will use the specific geo-spatial index |

|  |  |
| --- | --- |
| Pagination in arangosh | |
| query.limit(*number*) | limits a result to the first number documents |
| query.skip(*number*) | skips the first number documents |

|  |  |
| --- | --- |
| Sequential Access And Cursors in arangosh | |
| query.hasNext() | returns true if the cursor still has documents |
| query.next() | advance cursor |
| query.dispose() | free resources associated with a cursor |
| query.count() | returns number of documents in the result set |