Index manipulation

ALTER TABLE index {ADD|DROP} COLUMN column_name [{INTEGER|INT|BIGINT|FLOAT|BOOL|MULTI|MULTI64|JSON|STRING}]

> Change index structure

ATTACH INDEX diskindex TO RTINDEX rtindex

> move data from a regular disk index to a RT index

SHOW TABLES [LIKE pattern]

> display active indexes

RELOAD INDEX idx [FROM '/path/to/index_files']

> rotate specific index

RELOAD INDEXES

> rotate indexes

TRUNCATE RTINDEX rtindex

> truncate RealTime data

OPTIMIZE INDEX index_name

> Optimize a RealTime index

{DESC | DESCRIBE} index [LIKE pattern]

> lists index columns and their associated types.

FLUSH RAMCHUNK rtindex

> forcibly creates a new disk chunk in a RT index

FLUSH RTINDEX rtindex

> forcibly flushes RT index RAM chunk contents to disk

SHOW AGENT ['agent'|'index'] STATUS [LIKE pattern]

> Displays the statistic of remote agents or distributed index

DBA commands

SHOW STATUS [LIKE pattern]

> Display daemon performance counters

SET [GLOBAL] server_variable_name = value

> SET a global variable

SHOW THREADS [OPTION columns=width]

> List active client threads

DBA commands (cont)

SHOW [{GLOBAL | SESSION}] VARIABLES [WHERE variable_name='xxx']

> Display global variables

FLUSH HOSTNAMES

> Renew IPs associates to agent host names

FLUSH LOGS

> Initiate reopen of log files

FLUSH ATTRIBUTES

> Flushes all in-memory attribute updates

SHOW PLUGINS

> Displays all the loaded plugins and UDFs

CREATE FUNCTION udf_name RETURNS {INT | INTEGER | BIGINT | FLOAT | STRING} SONAME 'udf_lib_file'

> installs a user-defined function (UDF)

CREATE PLUGIN plugin_name TYPE 'plugin_type' SONAME 'plugin_library'

> Loads the given library (if it is not loaded yet) and loads the specified plugin from it.

DROP FUNCTION udf_name

> deinstalls a user-defined function (UDF)

DROP PLUGIN plugin_name TYPE 'plugin_type'

> Markes the specified plugin for unloading

CALL SNIPPETS

CALL SNIPPETS(data, index, query[, opt_value AS opt_name[, ...]])

CALL SUGGEST/QSUGGEST

CALL QSUGGEST(word, index [,options])

CALL SUGGEST(word, index [,options])

Options:
- limit
- max_edits
- result_stats
- delta_len
- max_matches
- reject
- result_line
- non_char

CALL PQ

CALL PQ(data, index[, opt_value AS opt_name[, ...]])

Options:
- docs_json
- docs
- verbose
-query

CALL KEYWORDS

CALL KEYWORDS(text, index [, options])

call keywords('que*', 'myindex', 1 as fold_wildcards, 1 as fold_lemmas, 1 as fold_blended, 1 as expansion_limit, 1 as stats);

```
call keywords(
'que*',
'myindex',
1 as fold_wildcards,
1 as fold_lemmas,
1 as fold_blended,
1 as expansion_limit,
1 as stats);
```

INSERT

```
INSERT INTO myindex(id,field1,field2,col1,col2)
VALUES(1,'title','content',10,100);
```

```
INSERT INTO myindex(id,field1,field2,col1,col2)
VALUES(1,'title','content',10,100),(2,'new title','next content',50,100);
```

REPLACE

```
REPLACE INTO myindex(id,field1,field2,col1,col2)
VALUES(1,'title','content',10,100);
```

UPDATE

```
INSERT UPDATE myindex SET col2=200 WHERE id=1;
```

DELETE

```
DELETE FROM myindex WHERE id=1
```

SELECT

Full-text match

```
SELECT * FROM myindex WHERE MATCH('find me')
```

Simple select

```
SELECT * FROM myindex
```

SELECT (cont)

GROUP BY

```
SELECT * FROM myindex WHERE MATCH('...') GROUP BY col1;
```

GROUP n BY

```
SELECT * FROM myindex WHERE MATCH('...') GROUP 5 BY col1;
```

WITHIN GROUP ORDER BY

```
SELECT * FROM myindex WHERE MATCH('...') GROUP by col1
WITHIN GROUP BY ORDER by col2 DESC order by col3 ASC
```

HAVING

```
SELECT * FROM myindex WHERE MATCH('...') GROUP BY col1
HAVING col2>10
```

ORDER BY

```
SELECT * FROM myindex WHERE MATCH('...') ORDER BY
WEIGHT() DESC, col1 ASC
```

LIMIT

```
SELECT * FROM myindex LIMIT 10,20
```

FACET

```
SELECT * FROM myindex FACET {expr_list} [BY {expr_list}] [ORDER
BY {expr | FACET()} {ASC | DESC}] [LIMIT [offset,] count]
```

SELECT OPTION

agent_query_timeout	max time in milliseconds to wait for remote queries to complete
boolean_simplify	enables simplifying the query to speed it up
comment	user comment that gets copied to a query log file
cutoff	max found matches threshold
field_weights	per-field user weights for ranking
global_idf	use global statistics from the global_idf file rather than local idf
idf	IDF computation flags
local_df	sum DFs over all the local parts of a distributed index

SELECT OPTION (cont)

index_weights	er-index user weights for ranking
max_matches	per-query max matches value
max_query_time	max search time threshold, msec
max_predicted_time	max predicted search time
ranker	any of proximity_bm25, bm25, none, wordcount, proximity, matchany, fieldmask, sph04, expr, or export
retry_count	distributed retries count
retry_delay	distributed retry delay, msec
reverse_scan	control the order in which full-scan query processes the row
sort_method	pq (priority queue, set by default) or kbuffer (gives faster sorting for already pre-sorted data)
rand_seed	nteger seed value for an ORDER BY RAND()
low_priority	runs the query with idle priority
expand_keywords	expand keywords with exact forms and/or stars

Query Meta and Profiling

SHOW META [LIKE pattern]

> shows additional meta-information about the latest query

SHOW WARNINGS

> retrieve the warning produced by the latest query

SHOW PLAN

> displays the execution plan of the previous SELECT statement (requires SET profiling=1)

SHOW PROFILE

> detailed execution profile of the previous SQL statement (requires SET profiling=1)

Query Meta and Profiling (cont)

START TRANSACTION | BEGIN COMMIT ROLLBACK SET AUTOCOMMIT = {0 | 1}

> start, commit or rollback transactions