Query Types

Qserv recognizes two distinct types of user queries: *synchronous* (also called *interactive*, or *low-volume queries*) and *asynchronous* (also called *long-running,* or *high-volume queries*).

Synchronous queries are like typical RDBMS queries. An example use case: user starts a mysql client, types "SELECT * FROM Object WHERE objectId = 1234" and receives the answer. Synchronous queries are expecting to be quick / short lived: a typical synchronous query is expected to be answered in under 10 seconds. Since they are taking short time and involve a small number of servers, a failure is relatively unlikely, and the cost of rerunning a failed query is low, so Qserv will not be taking any special actions to attempt to recover such query, and user should simply rerun the query.

Asynchronous queries on the under hand, unlike synchronous queries, will involve touching a large data set. A typical asynchronous query might scan entire Object Catalog (billions of rows), Source of ForceSource catalogs (trillions of rows) or join Object table with *Source table. Such queries will run for a long time, ranging from ~1 hour (Object scan) to ~12 hours (*Source scans or joins). In Qserv, when user submits asynchronous query, Qserv will immediately return a unique queryId. User can then use the queryId to manage the query.

When submitting a query, user will be able to send a hint with a query to indicate what query type it is. The exact API is TBD, we expect to have something like "queryType = sync", "queryType = "async". Related story:

DM-3478 - Jira project doesn't exist or you don't have permission to view it.

In cases when it is not obvious what is the right query type for a given query is, user will be able to check the type through Qserv prior to executing the query, similarly to how "EXPLAIN <query>" works. Related story:

DM-3477 - Jira project doesn't exist or you don't have permission to view it.

	SQL	RESTful
submit sync query	SELECT * FROM Object	GET db/v0/L2/DR1/query/sql=SELECT+*+FROM+Object
submit async query	to do, see DM-3480 - Jira project doesn't exist or you don't have permission to view it.	POST db/v0/L2/DR1/query/sql=SELECT+*+FROM+Object
retrieve type of the query	to do, see DM-3477 - Jira project doesn't exist or you don't have permission to view it.	to do, see DM-3484 - Jira project doesn't exist or you don't have permission to view it.
request status of the query	to do, see DM-3480 - Jira project doesn't exist or you don't have permission to view it.	GET /db/v0/query/<queryId>/status

retrieve schema of the results	to do, see DM-3480 - Jira project doesn't exist or you don't have permission to view it.	GET /db/v0/query/<queryId>/schema
retrieve results	to do, see DM-3480 - Jira project doesn't exist or you don't have permission to view it.	GET /db/v0/query/<queryId>/results
retrieve partial results while the query is running	to do, see DM-3480 - Jira project doesn't exist or you don't have permission to view it.	to do, see DM-3479 - Jira project doesn't exist or you don't have permission to view it.
kill the query	KILL <queryId>	to do, see DM-3479 - Jira project doesn't exist or you don't have permission to view it.

The API document shows the RESTful API.

Results from async queries will be stored in user workspace.

It is important to note that Qserv will attempt to transparently recover from failures without disrupting asynchronous queries. For example, if some worker nodes or even the front-end server user started the query through fail, the query should survive and user should be able to retrieve the query status and results through alternative front-end server.