information
insights
immediate

QL2 Rest API

Use the power and flexibility of HTTP for creating market collections

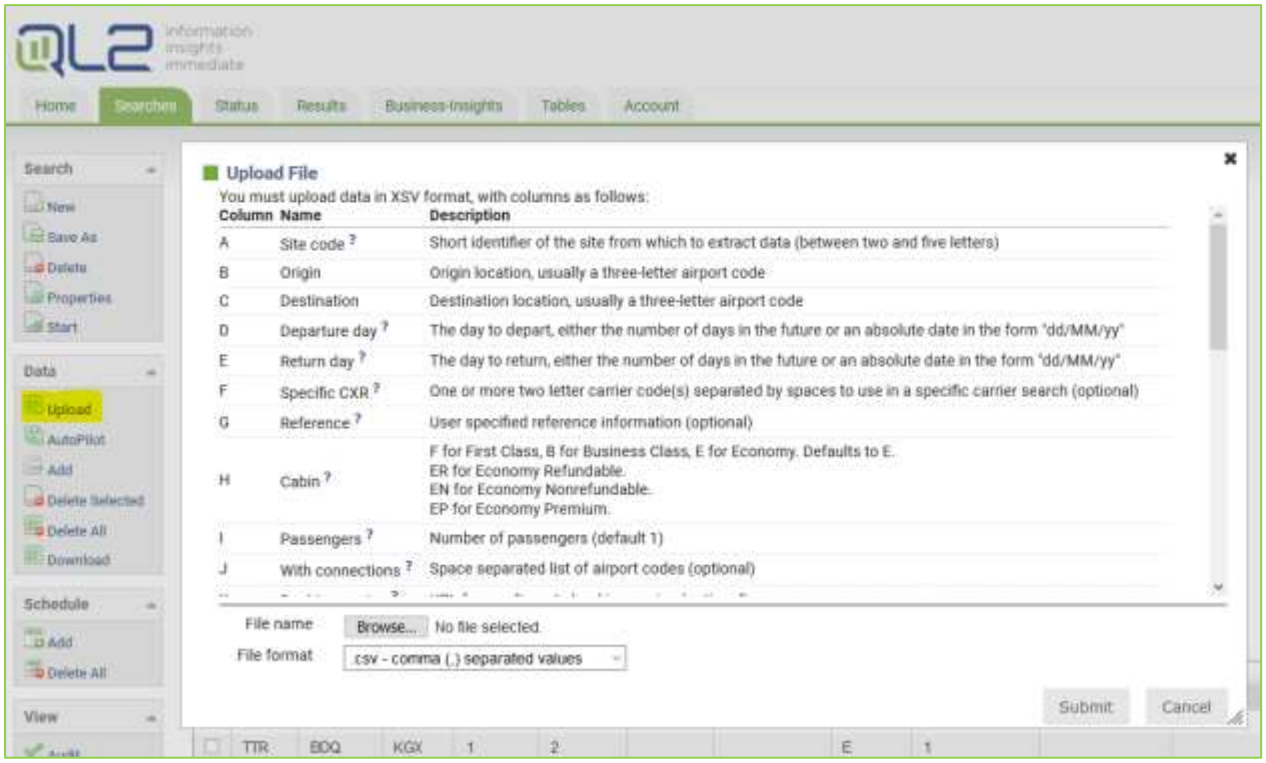
QL2 USER GUIDE

Table of contents

Getting started

QL2 Rest API offers the ability to use Hypertext Transfer Protocol for creating market collections and receiving results. The upload file format for your QL2 application can be found within a search by clicking on the “Upload” hyperlink in the QL2 Live interface.

The data should be in the same format used when uploading line items into a job.



Upload File

You must upload data in XSV format, with columns as follows:

Column	Name	Description
A	Site code ?	Short identifier of the site from which to extract data (between two and five letters)
B	Origin	Origin location, usually a three-letter airport code
C	Destination	Destination location, usually a three-letter airport code
D	Departure day ?	The day to depart, either the number of days in the future or an absolute date in the form "dd/MM/yy"
E	Return day ?	The day to return, either the number of days in the future or an absolute date in the form "dd/MM/yy"
F	Specific CXR ?	One or more two letter carrier code(s) separated by spaces to use in a specific carrier search (optional)
G	Reference ?	User specified reference information (optional)
H	Cabin ?	F for First Class, B for Business Class, E for Economy. Defaults to E. ER for Economy Refundable. EN for Economy Nonrefundable. EP for Economy Premium.
I	Passengers ?	Number of passengers (default 1)
J	With connections ?	Space separated list of airport codes (optional)

File name: No file selected.

File format: csv - comma (,) separated values

TTR	BDQ	KGX	1	2			E	1

You can initiate jobs on the Client Center by POSTing CSV data and parameters to a URL. When you follow these steps, a job is created, the CSV records are imported into it and the job starts immediately.

Use this method when you want to post the data using the same CSV syntax that you would use to upload line items into a job.

In other words, the content of the message passed in the POST section of the HTTP request must be exactly like a CSV file that you would upload to a job when creating line items.

Basic Syntax

```
https://live.ql2.com/submit?username={username}&password={pwd}&app=
{privilege}&createorreplacejob={jobname}
```

Public Parameters

- ✓ username: The name of the account.
- ✓ password: The password for the account.
- ✓ app: The short symbolic name for the application as in below table.

Application	App Name to be used in request
CARS	carrental
CRUISES	cruises
FERRY	ferry
FLIGHTS	airfare
HOTELS	hotel
RETAIL	retail
VACATIONS	vacation

- ✓ result: For most users, this is left blank or not passed at all. In that case, the job is started and the server responds immediately. However, if you want the server to wait for the job to finish, and return some specific result file, then pass the name of that file here. The Client Center will wait for up to 30 minutes for the job to run and complete. Note that the result filename must be a user-visible result file such as out.csv, so the internal "raw.csv" is not an acceptable filename. Also note that this parameter should not be used for larger jobs that will run longer than 30 minutes.
- ✓ createorreplacejob: Will search on job name. If found, act as "replacejob". If not found, act as "jobname".
- ✓ jobname: The name of the new job. This has to be a unique name that's never been used before for this account, so you should include something like a timestamp, etc.
- ✓ replacejob: This is the alternative to a unique job name. Specify a job name using this parameter to replace all line items in the existing job and then start it with the new data. The job has to exist in the account already.
- ✓ startjob: Normally you would not pass this parameter, since the default behavior is to start the job. However in some cases you might want to upload a new job and set a schedule for it to run later, so you want to prevent it from running immediately. In this case, pass **startjob=n** to prevent the job from starting.

- ✓ setschedule: Set a schedule for this job. The syntax is **yyyyMMddHHmm**. Only use this in conjunction with **startjob=n** otherwise you will start the job immediately and also set a schedule for later, which is probably not what you intended.
- ✓ stop: Pass a stop time represented as **HHmm** in your account timezone. Normally you won't pass this parameter, since you'll want the job to run until completion. However if you want to set a stop time for long-running job, you can pass it here. The hour is in 24-hour notation, so you'd represent 11:30pm as 2330. The time will be interpreted in the timezone of the user that owns the job.
- ✓ stopafter: Stops the job some amount of time after the first work unit has started. Unlike the "stop" parameter, which represents an absolute time, the "stopafter" parameter represents a duration of time. Specify the duration using HHMM syntax (e.g. 0130 for 1 hour 30 minutes after the first work unit starts). This parameter doesn't work for phantom jobs (explained below).
- ✓ priority: Pass "high" if you want to start the job in high priority. All other values (or lack of presence at all) result in the job being started in normal priority.
- ✓ timeout: Timeout period when waiting for the run to finish. Only pass this if you are waiting for a result file or run file. The timeout will cause the server to abort the run after the given duration and return an error code to you. For example if you are waiting for a result and you don't want the run to exceed 5 minutes, pass timeout=5. The number is normally represented in minutes, but you can specify seconds by adding an "s" to the end, so 30 seconds would be timeout=30s.
- ✓ phantom: Pass phantom=y if you want to create a run without any corresponding job. This is recommended if you are sending lots of real-time queries or if you don't want to clutter up an account with a bunch of one-time jobs. If you truly don't need to keep a permanent job around, this is recommended since it helps reduce unnecessary strain on the system.

Asynchronous requests

Synchronous requests do input validation in realtime, and will therefore return errors or warnings if some of the inputs are invalid. The server response will indicate errors as well as whether the job successfully started or not.

Asynchronous calls just accept the inputs and close the connection, so no validation information is returned. It's useful for:

- Larger jobs where validation may take a few minutes
- Situations where the user doesn't want to tie up their own resources holding a connection open during validation
- When the user is fairly certain they have valid inputs

For asynchronous requests, use asubmit instead of submit in the url. Result and runfile parameters are not supported.

Sample url:

https://live.ql2.com/**asubmit**?username={username}&password={pwd}&app={privilege}&jobname={jobname}

Authentication Token Method

This feature was implemented to help keep login information secure and eliminate the need to pass credentials in plaintext in the URL.

Use below URL to get your authentication token:

https://live.ql2.com/getAuthToken

Pass your username and password once to generate the token. The site will provide you with a token to replace your credentials in the URL:

Basic Syntax

```
https://live.ql2.com/submit?auth_token={token}&app={privilege}&jobname={jobname}
```

Schedule Capabilities

Monthly Schedules: Basic Syntax

```
https://live.ql2.com/submit?username={username}&password={pwd}&app={privilege}&jobname={jobname}&startjob=n&scheduletype={monthly }&setschedule={MM_ddHHmm or MM_ddHHmm or ALL_ddHHmm }
```

- ✓ scheduletype=monthly
- ✓ setschedule=Monthly Schedules: This allows user to add the monthly schedule date and time as per the user's timezone.

<u>Setschedule parameter formats:</u>

1. {MM_ddHHmm} eg : 01_140100 - On the 14th at 01:00 on these months: Jan
2. {MM_ddHHmm} eg : 010203_110100 - On the 11th at 01:00 on these months: Jan, Feb, Mar
3. {MM_ddHHmm} eg: JANFEB_081010 - Every January and February 8th at 10:10
4. {ALL_ddHHmm} eg: ALL_020200 - On the 2nd day of all months at 02:00

Weekly Schedules: Basic Syntax

https://live.ql2.com/submit?username={username}&password={pwd}&app={privilege}&jobname={jobname}&startjob=n&scheduletype={weekly}&setschedule={WEK_HHmm or WEK_HHmm or ALL_HHmm }

- ✓ scheduletype=weekly
- ✓ setschedule=Weekly Schedules : This allows user to add the weekly schedule date and time as per the user's timezone.

Setschedule parameter formats:

1. {WEK_HHmm} eg: SUN_0100-Every Sunday at 01:00 IST
2. {WEK_HHmm} eg: SUNMONTUE_0200
   - ✓ Every Sunday at 02:00 IST
   - ✓ Every Monday at 02:00 IST
   - ✓ Every Tuesday at 02:00 IST
3. {ALL_HHmm} eg:ALL_2305
   - ✓ Every Sunday at 23:05 IST
   - ✓ Every Monday at 23:05 IST
   - ✓ Every Tuesday at 23:05 IST
   - ✓ Every Wednesday at 23:05 IST
   - ✓ Every Thursday at 23:05 IST
   - ✓ Every Friday at 23:05 IST
   - ✓ Every Saturday at 23:05 IST

Biweekly Schedules: Basic Syntax

https://live.ql2.com/submit?username={username}&password={pwd}&app={privilege}&jobname={jobname}&startjob=n&scheduletype={biweekly}&setschedule={BIW_yyyyMMddHHmm or BIW_yyyyMMddHHmm or ALL_yyyyMMddHHmm }

- ✓ Scheduletype=biweekly
- ✓ Setschedule=BIWeekly Schedules: This allows user to add the bi-weekly schedule date and time as per the user's timezone.

Setschedule parameter formats:

1. {BIW_yyyyMMddHHmm} eg: MON_201902011400 :- Every other Monday at 14:00 Starting From February 1, 2019 IST
2. {BIW_yyyyMMddHHmm} eg: SUNWEDFRI_201902020500
   - ✓ Every other Sunday at 05:00 Starting From February 2, 2019 IST
   - ✓ Every other Wednesday at 05:00 Starting From February 2, 2019 IST
   - ✓ Every other Friday at 05:00 Starting From February 2, 2019 IST
3. {ALL_yyyyMMddHHmm} eg: ALL_201902040200
   - ✓ Every other Sunday at 02:00 Starting From February 4, 2019 IST
   - ✓ Every other Monday at 02:00 Starting From February 4, 2019 IST
   - ✓ Every other Tuesday at 02:00 Starting From February 4, 2019 IST
   - ✓ Every other Wednesday at 02:00 Starting From February 4, 2019 IST
   - ✓ Every other Thursday at 02:00 Starting From February 4, 2019 IST
   - ✓ Every other Friday at 02:00 Starting From February 4, 2019 IST
   - ✓ Every other Saturday at 02:00 Starting From February 4, 2019 IST

Daily Schedules: Basic Syntax

https://live.ql2.com/submit?username={username}&password={pwd}&app={privilege}&jobname={jobname}&startjob=n&scheduletype={daily}&setschedule={HH_mm or HH_mm or ALL_mm }

- ✓ Scheduletype=daily
- ✓ Setschedule= Daily Schedules: This allows user to add the daily schedule date and time as per the user's timezone.

Setschedule parameter formats:

1. {HH_mm} eg: 01_55 :- At :55 past the hour on these hours (IST): 01
2. {HH_mm} eg: 000408121620_05 :- At :05 past the hour on these hours (IST): 00, 04, 08, 12, 16, 20
3. {ALL_mm} eg: ALL_10 :- At :10 past the hour on all hours

Search Request Retrieval

You have the flexibility of receiving results of the job upon completion of the request or by retrieving them at another time via a different URL.

Realtime Results

For retrieval of results, use the result parameter (see above) to pass a filename, ensuring the filename matches one of the result files output. These result files are previously designated by QL2 for your organization/account.

https://live.ql2.com/submit?username={username}&password={pwd}&app={privilege}& jobname={jobname}

Historical Results

The retrieveIndex URL enables you to retrieve a list of result files for a search.

https://live.ql2.com/retrieveIndex?username={username}&password={password}

The result files are available as they complete. The preceding string returns an XML document with the search results from the previous ten days. The URLs for retrieving the individual result fi les are embedded in the result-file tags:

```
<?xml version=”1.0” encoding=”UTF-8”?>
<result>
<run id=”run id” job-id=”job id” job-name=”job name” time=”timestamp”>
<result-file id=”result file id” jobId=”job id” name=”filename.csv”
url=”https://live.ql2.com/retrieveFile?id={id}” />
</run>
<run id=”run id” job-id=”job id” job-name=”job name” time=”timestamp”>
<result-file id=”result file id” jobId=”job id” name=”filename.csv”
url=”https://live.ql2.com/retrieveFile?id={id}” />
</run>
<run id=”run id” job-id=”job id” job-name=”job name” time=”timestamp”>
<result-file id=”result file id” jobId=”job id” name=”filename.csv”
url=”https://live.ql2.com/retrieveFile?id={id}”/>
</run>
</result>
```

To view a file, append your username and password to the URLs given in result-file: https://live.ql2.com/retrieveFile?id={id}&username={username}&password ={password}

Optional job parameters

Each vertical has additional/optional job parameters that can also be set by appending to the url. These normally have default values for all jobs in the account, but individual parameters can be overridden through the API. Examples include:

- QUICK_QUERY=1 to set a shallow shop
- EMAIL=test@gmail.com to add/override the email recipients for job completion notices
- CHANGE_CURRENCY=USD to instruct the system to use xe.com rates to convert currency of rates in the output (may require additional script updates)

Contact your Account Manager with details if you have additional parameters you would like to control.