**How to ingest data From Liveperson With Python**

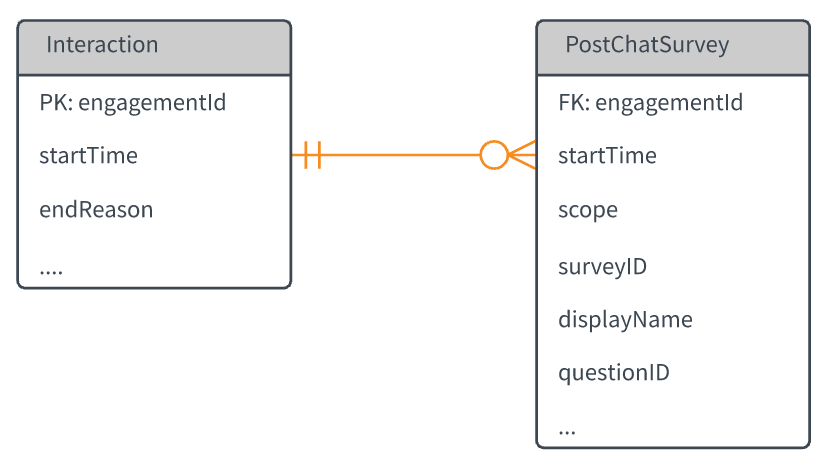
[Engagment History API](https://developers.liveperson.com/data-engagement-history-overview.html) let you grab livechat interaction data from Liveperson. It is based on the REST architecture and uses OAuth1.0. You first need to [retrieve API Keys](https://developers.liveperson.com/guides-gettingstarted.html). In this example, I am using the requests and requests\_oauthlib modules to make API calls from Python. Liveperson offers a good code examples and [this](https://github.com/LivePersonInc/developers-community/tree/master/assets/Code%20examples/APIs/Engagement%20History%20API) is a good place to start.

The resulting JSON file include all the [attributes](https://developers.liveperson.com/data-engagement-history-appendix.html) associated with live chat activities (like agent stats, customer feedback, transcripts, customer’s behaviour on the website and so on). To convert it into structured tables, you need to create a data model. CSV files can be inserted into a database.

**Data Model**

In the info node, you can find ‘engagementId’. All the tables can be tied with this. I also think it is a good idea to bring ‘startTime’ from the info node into all the tables. Depending on how your Liveperson is configured, the number of tables would be different. But, you always need a base table which contains unique engagement id per row so that you can create one to many relationships.

In this example, I choose interaction and post chat survey tables for simplification. The interaction table has a primary key engagement\_id. Post chat survey typically contains multiple questions. This means one interaction can have many post chat survey records. We will add the engagement\_id as the foreign key to the interaction table to handle this one-to-many relationship. By using the same method, you can create a data model for all the attributes.



**Gocha!**

* The start and end date parameters are in epochtime millisecond. The function convert\_to\_epochtime converts date in string into epochtime.
* Some of the timestamp fields has epochtime. I created a function to convert epochtime back to timestamp.
* API limit is 100 records. We need to increment offset parameter in while loop until we get all the records.
* Some field does not exist all the time, which requires you to handle the key error with try-except (for example, ‘mcs’ in the interaction node).
* It is better to use pipe instead of comma because some values contain comma.

Let’s put them together. Here it comes!

**Code**

**[cc lang="python" tab\_size="4" lines="-1" theme="GeSHi" line\_numbers="false"]**

**[/cc]**