

ClinicalAI: Instruction Manual

How to build the clinicalai knowledge graph, and query it with natural language using a large language model

Prerana S K | ClinicalAI | 19-07-2024

# **Brief Overview of the Project**

90% of clinical experiments fail, due to insufficient background work and research. This can be improved with the help of better pre-planning and more informed decision making. ClinicalAI is a project that aims to bring together the logical deduction capabilities and structure provided by a KG (Knowledge Graph) with the natural language interpretation capabilities of an LLM. This is a step-by-step instruction manual that goes through all the steps (from scratch) of completing the set-up for this project. In the following sections, we first see how to collect and clean data from <https://www.clinicaltrials.gov/>. Then we see how a schema is built on Neo4j’s AuraDB, and finally how to integrate it with GPT-4.

\*\*\* All code can be found at <https://github.com/Narpear/HeCiX>

## **Section-A (collecting and cleaning the dataset)**

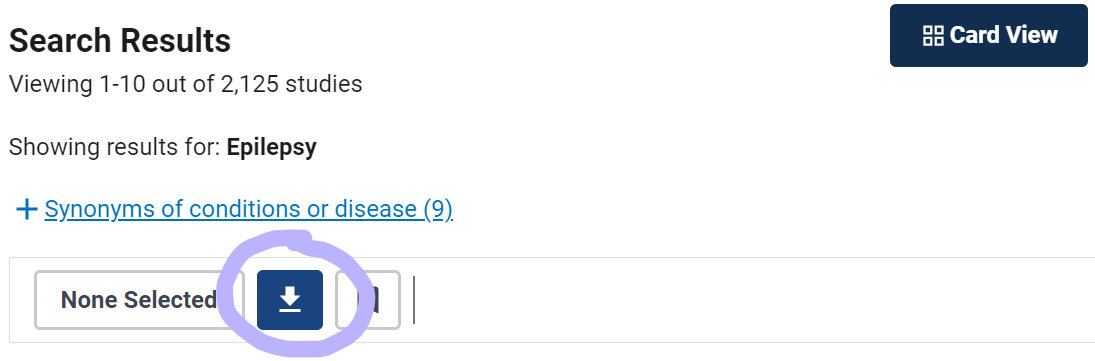
### **Collecting the Dataset**

The first step of our project involves collecting our data and cleaning it. For this proof-of-concept project, we acquire data from <https://www.clinicaltrials.gov/> of six diseases, 200 records each. The six diseases we use are vitiligo, melanoma, atopic dermatitis, alopenia areata, vitiligo, and epilepsy. (On reverse engineering, we realized that these have common nodes such as principal investigator, conditions, etc. and therefore picked these six diseases.)

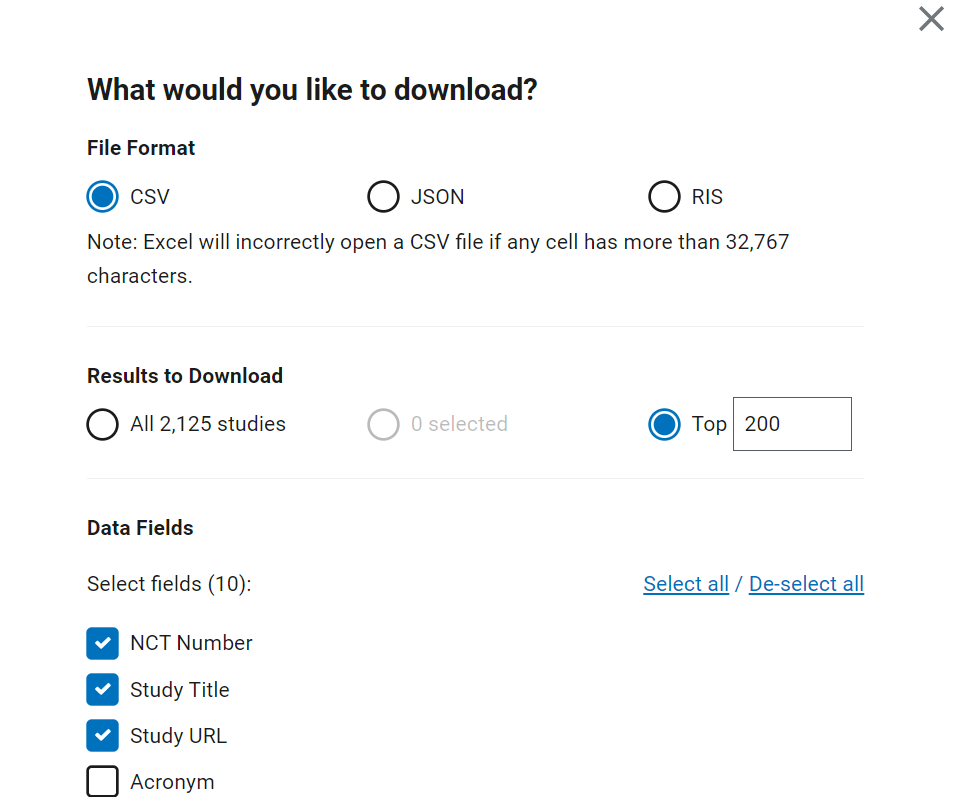
To do this, we visit <https://www.clinicaltrials.gov/> and type in the name of the disease of interest. In our case, this is epilepsy. The other fields can be left blank.



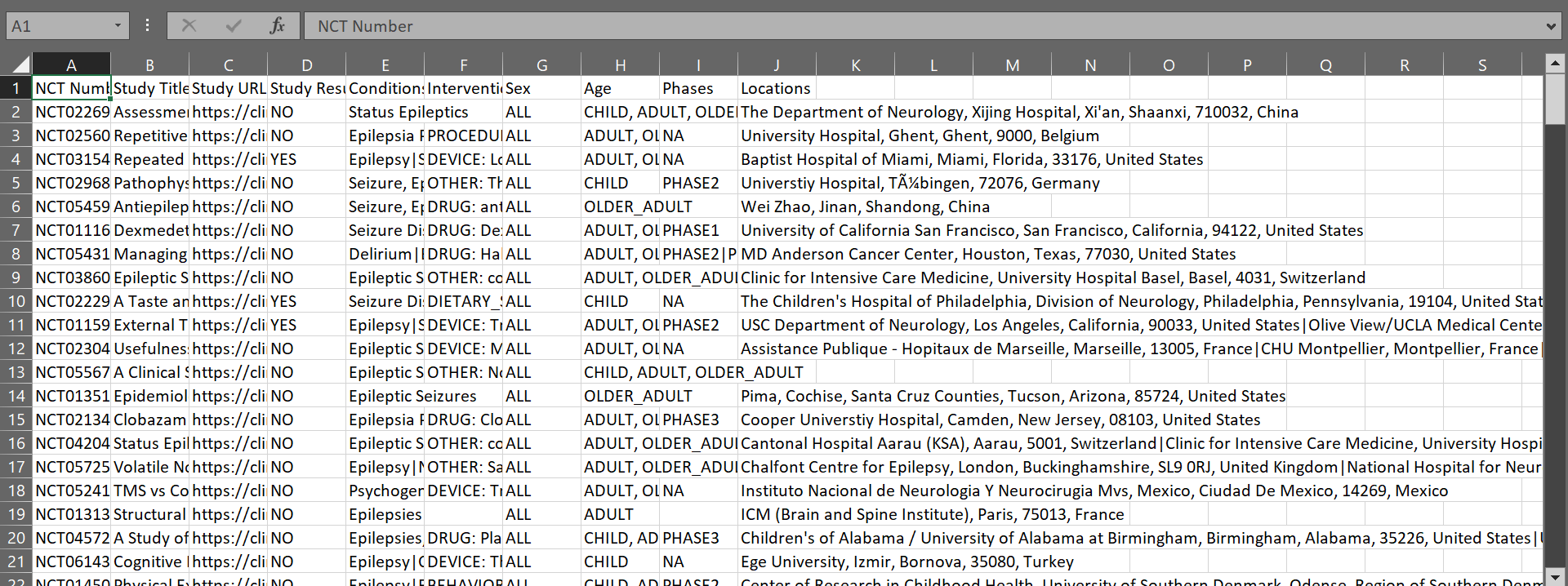
We then click on the download button on the page.



Then, we choose the selected fields and download our data. We chose to download the data in .csv format, and decided to download 200 records for the purpose of our study. The fields selected are NCT Number, Study Title, Study URL, Study Results, Conditions, Interventions, Sex, Age, Phases, and Locations.



The downloaded .csv file should look something like this



Since this does not include information about Principal Investigators, our next step is to scrape this data from the webpage dedicated to each clinical trial.

### **Scraping Principal-Investigators from Individual Webpages**

To scrape this data from webpages of each clinical trial, we write some basic python code. This code can be found at <https://github.com/Narpear/HeCiX/tree/main/clinical_trials>.

There are four python files here that help with scraping data we need.

* download\_html.py (<https://github.com/Narpear/HeCiX/blob/main/clinical_trials/download_html.py>)

This function downloads and saves the formatted HTML content of a specific clinical trial page from clinicaltrials.gov using its NCT number.

* get\_div.py (<https://github.com/Narpear/HeCiX/blob/main/clinical_trials/get_div.py>)

This function extracts and returns the principal investigator names from HTML pages locally saved by downloag\_html.py, identified by its NCT number.

* delete\_html.py (<https://github.com/Narpear/HeCiX/blob/main/clinical_trials/delete_html.py>)

This function deletes a specific HTML file named after an NCT number if it exists in the current directory, or prints a message if the file does not exist.

* append\_pis.py (<https://github.com/Narpear/HeCiX/blob/main/clinical_trials/append_pis.py>)

This script processes a CSV file of clinical trials, downloads each trial's webpage, extracts principal investigator names, updates a DataFrame with this information, deletes the downloaded HTML, and saves the results to a new CSV file.

The file titled append\_pis.py is the main file that is to be run, and the others are basically helper functions. The paths used within the code should be changed, to match the locations of the files on other systems.