For this homework assignment, I actually went to the whiteboard and drew out the schemas, largely based around the queries provided in the spec sheet. I found this to be much more time-effective than the ‘run-and-gun’ philosophy I usually take. Although I spend less time programming, and therefore feel less productive, I actually end up finishing faster overall than I do when I program on the fly.

An interesting roadblock I encountered was looking at and implementing joins in a meaningful way. There were some instances where they made sense to me, and others where they plainly did not. Initially, I assumed this homework would be based almost entirely around joins; however, they often didn’t work as well as alternatives. I think this helped me understand the place of joins in SQL - fantastically useful, most of the time.

At this point, designing a functioning schema (especially when the bulk of it is provided to us) is not an issue - what proved to be more difficult for me, though, was constructing my queries in an elegant and efficient manner. I made sure I wrote and tested each query individually, populating my tables with data that could cause a query to fail if I wasn’t careful (ex: MEX has lower GDP and lower inflation, CAN has lower GDP and higher inflation; the final query on returns Canada, as it should, ensuring that I’ve got my signs in place).

When I was confused, I found it helpful to implement a query in a fashion I was familiar with - this mostly involved full joins. Once I was confident that I was actually manipulating the data in a meaningful way (getting the results almost correct), I would draw an example table resulting from the query. I crossed out what I didn’t need, circled what I did, marked duplicates, etc, and tried to modify my query in a way, such that I was getting as much as I needed, with as little fluff as possible. I found this to be a helpful exercise, especially when working with joins, where it’s easy to get way more data than you signed up for.