CBS Orchestration Tool Description

The CBS orchestration tool is uniquely built for the CBS\_Low\_Income rule application. It creates the initial entity state from 3 SQL database tables, calls the rule application, then maps the final entity state back to 2 SQL tables.

The appSettings.json file parameterizes most of the variables to limit hard coding as much as possible. Since the database structure is flat, it must be mapped to a JSON object by mapping each SQL column value to a strongly typed object defined in a C# class in the model directory. There are three model files. The OrchestrationToolConfig model is for the app settings configuration, the REX\_Model is for the rule execution objects, and the CBSModel is to define the low-income entity state.

Once the app settings are loaded into the main portion of the orchestration tool, the database connection is made, and the output tables are cleared if the app setting "DeleteOutputTables" is set to true. Otherwise, the output would be added on top of existing data in the two output tables.

The SQL statement is built from the tables set in the app settings and executed to map the needed data to create the initial entity state. Inside of that SQL statement are two variables, rowStart and rowEnd, that determines the range of households that are pulled in on each run based on the household number. By default, this starts at 0 and ends at 500,000 meaning that households with numbers 0 to 500,000 are pulled in on the first run. The next run is incremented by 500,000 until the entire database has been queried and executed.

Once the data has been pulled from the SQL database and mapped to the strongly typed object defined in the model class, it is then serialized into JSON as required by irServer.

For optimal performance 3 warm-up batches are made first to warm up irServer. This is a standard practice that should always be done and only the first household object is executed.

After warm up the set number of batches are queued and then sent to irServer to execute each batch of JSON. If successful, the response is then mapped back into JSON to pull the final entity state. That final entity state is then bulk uploaded back into the SQL database to the defined output tables.

As the orchestration tool runs it will keep a count of the time each current process runs as well as provide the final time and the number of households that were executed. Below is an example of that output:

Text

Description automatically generated