# Extract – Transform – Load

Project Report

## Introduction

The National Disability Insurance Scheme provides supports to over half a million individuals with disabilities. The scheme has an annual budget of over 15 billion dollars and services are provided by almost ten thousand service providers across Australia.

## The problem to be solved…

The NDIS website provides extensive data. Some of it has been presented in an easy-to-use format on the ‘Explore Data’ page. However, the data presented on this page is a snapshot only and doesn’t contain the archived data of previous snapshots which is stored elsewhere on the NDIS website as CSVs. From a business intelligence perspective access to this longitudinal data is essential for analysing trends and making predictions.

The second issue is that this data is disconnected. It would be useful to be able to combine data from, for example, the ‘Participants’ table and the ‘Plan utilisation’ table.

## Project Aim

The aim for this project was to build a relational database in SQL of all four datasets presented on the ‘Explore Data’ page, using all available longitudinal data. It quickly became apparent that the scope of this aim was outside the timeframe available for this project.

The revised aim was to build a database of two of the datasets (‘Participants’ and 'Utilisation of Plan Budgets'). On account of this change, no ERD has been presented as part of this report, as it is unnecessary with the data model consisting of only two tables and a shared key.

Note that it was part of the project design for the final database to contain only WA data.

## Extraction, Transformation and Loading

The next sections detail the ETL process used for this project. This includes the text from the ‘walkthrough’ section of the project ReadMe, and a discussion of specific decisions made and challenges faced at each step of the ETL process.

From the project ReadMe:

To replicate the ETL process follow the following steps:

1. Clone this repository on your computer.

2. Download data (CSVs) from NDIS website.

- Go to https://data.ndis.gov.au/explore-data.

- In the 'Participants' section go to the States/Territories filter and select WA only.

- Select 'Download Data as CSV'.

- Repeat this process in the 'Utilisation of Plan Budgets' section.

- Go to https://data.ndis.gov.au/data-downloads.

- In the 'Participant numbers and average plan budgets' section download the available CSVs back to and including September 2020.

- In the 'Budget data downloads' section download the available CSVs back to and including September 2020.

- Move all downloaded CSVs into the 'resources' folder.

3. Run the ETL.ipynb notebook. Note the following:

- The CSVs downloaded from the 'Explore Data' page in step 2 are datestamped. The python script accounts for this. However, if you download the CSV on a different day to when you run the script you will need to manually enter the date.

- This notebook will remain current up until the end of December 2022. After this additional 'longitudinal' csvs will be added to the 'Data-downloads' page and the python script will need to be manually adjusted. The sections requiring adjustment are noted in the script.

- This notebook will generate two CSVs into the 'output' folder, which will be used in the next step. If there any issues in this process (including access to the NDIS website), examples of these CSVs have been added to the archive folder so that the remaining steps can be completed.

4. Create the ndis\_db in PostgresSQL.

- In PostgresSQL create a database 'ndis\_db'.

- In this database run the script included in ndis\_db\_tables.sql as a query.

- Use the import function in PostgresSQL to import participant\_data.csv into the participants table and plan\_utilisation\_data.csv into the utilisation table.

5. Query the tables as required.

- Note that the two tables can be joined on their key column.

## Extraction

The data sources were derived from two different sections of the NDIS website. As detailed above, the extraction involved manually downloading a large of CSVs.

There were different challenges with extracting data from each section. The CSVs generated from the ‘Explore Data’ webpage were datestamped as standard. To address this, the python code uses a datetime function. Though it is noted, that this is not a foolproof solution and this may create an error in the execution of the code (if for example, the code was run on a different day to when the CSV was downloaded).

The challenge with the longitudinal data was that there was no consistency with NDIS’ naming convention. As evident in the python code, the many CSVs were entered manually for this section of the extraction.

A note on the decision to only go back as far as September 2020 in the extraction process. The state-based disability service that proceeded the NDIS in WA was ended in June 2020. From this period and earlier, there was a ‘hybrid’ disability structure in WA. September 2020 represents the first time the datasets would be capturing a ‘NDIS-only’ disability sector in WA.

## Transformation

The following data transformations were required:

* *Basic cleaning.*  The longitudinal data had its date report column in inconsistent formats. A single format was selected and this was fixed.
* *Value replacement.* The participant count had a value of ‘<11’ used to maintain client confidentiality in districts with low participant counts. These values were replaced by ‘5’ as an approximation. This enabled this column to be defined as an integer data type (see below), without significantly distorting the statistical value of this data.
* *Consistency of column names.* Column names were inconsistent both across the longitudinal data and between the participant and utilisation datasets. These were renamed.
* *Longitudinal data appended into a single dataframe.*  This was a key transformation that was possible once column names had been made consistent. There were two steps to this transformation. Firstly, the longitudinal data (from the ‘Data downloads’ page) were appended. Then the current data was appended to this dataframe. For this second step, the current data required its own ‘report date’ column.
* *Unnecessary columns removed.* Some of the NDIS data columns appeared to have been collected in an ad hoc manner (ie many null values, inconsistencies, etc). It was decided to drop these columns at this stage.
* *Data types set.* Both dataframes required some small cleaning of punctuation, then the data types for each column could be set. Note that the report date was set as a string, as it was not anticipated to be used in future date calculations.
* *Key created.* With the view to later being able to join the two dataframes a unique key was created, this is detailed below.
* *Column order set.* This was done for consistency and to enable easy import into SQL.

A unique key was created using a technique where a unique composite concatenate was created for each row in both dataframes, and these were assigned to an index which became the Key value. This Key was created in such a way that future datestamped CSVs would add to the end of the index.

## Loading

A decision was made not to use SQLALchemy to automatically load these CSVs to a SQL database. In a professional environment this would be the preferred option.

As detailed in the walkthrough above, with only two tables it was not difficult manually creating the database in PostgresSQL. The repo includes the query used to create the SQL tables, including the data types and other constraints. As the columns and data types align with the dataframes created in the Transform step, it was straight-forward to import these CSVs into the database.

## Future analysis and final thoughts

This project hasn’t presented any data analysis. However, the ndis\_db database that has been created is business-ready for analysis.

Here are some examples of the types of analysis that is readily achievable with the data now that it has undergone an ETL process.

* Is the utilisation of plan funding improving? Are there geographical areas, or age groups, or disability types where the improvement is more evident? Likewise, in any of these categories has plan funding utilisation regressed? This type of inquiry was not possible when the longitudinal data was separate.
* Is there any correlation between the number of participants with a specific disability condition, and the utilisation of plan funding? This type of inquiry was not possible when participant data was separate from utilisation data.

As a final thought, the tools developed as part of this project could be used to expand the ndis\_db to contain further sections of the data that exists on their website. With additional ETL processes this database could become a useful resource for disability service organisations seeking to gain business intelligence from the available NDIS’ data.