PBS API V3 - File Mapping Generation Process

File mapping Version 3.0

API Version 3

Please Note:

The Department does not endorse the PBS Text files (PBS TXT Extracts) currently released as they are no longer fit for purpose. The Department encourages Software vendors to include the many updated PBS policy and legislative concepts that are now available.

This document is to assist users to recreate the PBS Text files (PBS TXT Extracts) from the data available via the PBS API Version 3. It may also be useful for non-text file users to help identify data relationships.

These SQL queries are for reference purposes only to help identify relationships between the text files and the API data and will not be maintained in the future.

More information about the API can be found on the [PBS Software Developers website](https://data.pbs.gov.au/).

Last Modified 12 November 2024

Department of Health and Aged Care

© Commonwealth of Australia

Contents

[1. Introduction 5](#_Toc182332550)

[1.1. PBS Data Distribution 5](#_Toc182332551)

[1.2. PBS artifact - text files have become an over-simplification of the PBS 5](#_Toc182332552)

[1.3. Objective 6](#_Toc182332553)

[1.4. Steps 6](#_Toc182332554)

[1.5. Supply Only Indicator Status 6](#_Toc182332555)

[2. Steps to Create Database from API Calls 7](#_Toc182332556)

[3. Creating a Database Tables 7](#_Toc182332557)

[4. Invoke API and download data 11](#_Toc182332558)

[4.1. URL to view PBS API Catalogue. 11](#_Toc182332559)

[4.2. Steps to download API data 11](#_Toc182332560)

[4.3. Steps to Load API data into database and verify. 11](#_Toc182332561)

[5. AMT 13](#_Toc182332562)

[5.1. Steps to download and import amt.txt data and store results 13](#_Toc182332563)

[5.2. 2013 Text Field Descriptions 15](#_Toc182332564)

[5.3. Sample ATM Data 15](#_Toc182332565)

[6. ATC 16](#_Toc182332566)

[6.1. Steps to download and import atc.txt data store results 16](#_Toc182332567)

[6.2. 2013 Text Field Descriptions 16](#_Toc182332568)

[6.3. Sample ATC data 16](#_Toc182332569)

[7. Authorities Restrictions 17](#_Toc182332570)

[7.1. Steps to download and import CautionExtract.txt data and store results 17](#_Toc182332571)

[7.2. 2013 Text Field Descriptions 17](#_Toc182332572)

[7.3. Sample Caution data 18](#_Toc182332573)

[7.4. Steps to download and import NoteExtract.txt and store results 18](#_Toc182332574)

[7.5. 2013 Text Field Descriptions 19](#_Toc182332575)

[7.6. Sample Note data 19](#_Toc182332576)

[7.7. Steps to download and import RestictionExtract.txt and store results 19](#_Toc182332577)

[7.8. 2013 Text Field Descriptions 21](#_Toc182332578)

[7.9. Sample Restriction Data 21](#_Toc182332579)

[7.10. Steps to download and import streamlined.txt and store results 22](#_Toc182332580)

[7.11. 2013 Text Field Descriptions 22](#_Toc182332581)

[7.12. Sample Streamlined Authority Items data 23](#_Toc182332582)

[8. Continued Dispensing (CD) 24](#_Toc182332583)

[8.1. Steps to download and import cd.txt data and store results 24](#_Toc182332584)

[8.2. 2013 Text Field Descriptions 25](#_Toc182332585)

[8.3. Sample CD data 25](#_Toc182332586)

[9. Drug data 26](#_Toc182332587)

[9.1. Steps to download and import drug.txt and store results 26](#_Toc182332588)

[9.2. 2013 Text Field Descriptions 27](#_Toc182332589)

[9.3. Sample Drug data 27](#_Toc182332590)

[10. Fees 28](#_Toc182332591)

[10.1. Steps to download and import Patient Contribution Fees (fees.txt) and store results 28](#_Toc182332592)

[10.2. 2013 Text Field Descriptions 28](#_Toc182332593)

[10.3. Sample Patient Contribution Fees 29](#_Toc182332594)

[10.4. Steps to download and import Container Fees (fees.txt) and store results 29](#_Toc182332595)

[10.5. 2013 Text Field Descriptions 33](#_Toc182332596)

[10.6. Sample Patient Contribution Fees 33](#_Toc182332597)

[10.7. Sample Dispensing Fees 33](#_Toc182332598)

[10.8. Sample Markup Fees 33](#_Toc182332599)

[11. Manufacturer data 34](#_Toc182332600)

[11.1. Steps to download and import mnfr.txt and store results 34](#_Toc182332601)

[11.2. 2013 Text Field Descriptions 34](#_Toc182332602)

[11.3. Sample Manufacturer Text data 35](#_Toc182332603)

[12. Medication Chart 36](#_Toc182332604)

[12.1. Steps to download and import Medication Chart Electronic data (med-chart-electronic.txt) and store results 36](#_Toc182332605)

[12.2. 2013 Text Field Descriptions 37](#_Toc182332606)

[12.3. Sample Medication Chart Electronic 37](#_Toc182332607)

[12.4. Steps to download and import Medication Chart Paper data (med-chart-paper.txt) and store results 38](#_Toc182332608)

[12.5. 2013 Text Field Descriptions 39](#_Toc182332609)

[12.6. Sample Medication Chart Electronic 39](#_Toc182332610)

[13. PBS Item Table 40](#_Toc182332611)

[13.1. Steps to download and import Pharcacy\_PBS\_Item\_Table.txt and store results 40](#_Toc182332612)

[13.2. 2013 Text Field Descriptions 42](#_Toc182332613)

[13.3. Sample PBS Item Table 42](#_Toc182332614)

[14. Prescriber Type 43](#_Toc182332615)

[14.1. Steps to download and import Prescriber\_type.txt and store results 43](#_Toc182332616)

[14.2. 2013 Text Field Descriptions 44](#_Toc182332617)

[14.3. Sample Prescriber Type 44](#_Toc182332618)

[15. Safety Net 45](#_Toc182332619)

[15.1. Steps to download and import sn20dr.txt and store results 45](#_Toc182332620)

[15.2. 2013 Text Field Descriptions 46](#_Toc182332621)

[15.3. Sample safety Net data 46](#_Toc182332622)

[16. Link Extract 47](#_Toc182332623)

[16.1. Steps to download and import linkextract.txt and store results 47](#_Toc182332624)

[16.2. 2013 Text Field Descriptions 47](#_Toc182332625)

[16.3. Sample Link Extract data 48](#_Toc182332626)

[17. Control Text 48](#_Toc182332627)

[18. Dispensing Incentive Items (DI) 48](#_Toc182332628)

[19. Appendix 49](#_Toc182332629)

[19.1. Links 49](#_Toc182332630)

[19.2. Get Schedule Code 49](#_Toc182332631)

[19.3. Verhoeff Checksum Algorithm 51](#_Toc182332632)

# Introduction

## PBS Data Distribution

PBS data will soon only be available via the PBS API.

The formatted versions of the PBS Schedule in PDF, including the Summary of Changes will continue to be available to the public via <https://www.pbs.gov.au/browse/publications> and under embargo to approved software vendors.

The following PBS outputs will soon be discontinued.

* PBS XML file, referred to as XML V3;
* PBS XML file based on the pre-September 2017 format, referred to as XML V2 or the “down-converted XML”; and
* PBS Text files, also referred to as the PBS TXT Extracts, which were derived from the XML V2.0 using a set of XSL stylesheets.

## PBS artifact - text files have become an over-simplification of the PBS

Please Note: The Department does not endorse the current text files as they are no longer fit for purpose. The Department encourages Software vendors to include the many updated PBS policy and legislative concepts that are now available.

A key reason for initiating this project was that the text files have become an over-simplification of the PBS. The text files were designed in the 1990s, when the PBS was much simpler and smaller than it is today and have not been updated since 2013. Many PBS policy and legislative concepts are therefore only partly represented in the text files (or not at all). The purpose of the API is to expose this complexity, as an accurate representation of the PBS data.  
  
The API has been put together so all different end users can obtain only the data they need (e.g. Doctors do not need pricing). Changes since the Text files stopped being updated may mean that fields may need to be combined or used together to obtain the current PBS schedule.

For example;

1. Continued Dispensing, now 2 distinct fields.
2. Supply only is part of the data and not a separate set of text files.
3. Notes indicator, Text files = N for Note, whereas the API is a yes/no field. This means N is used to mean different things.

## Objective

The goal of this document is to assist users in creating PBS text files by using API calls. This involves several key steps: setting up database tables, downloading data via API calls, importing this data into the tables, running SQL queries, and finally exporting the data to CSV files.

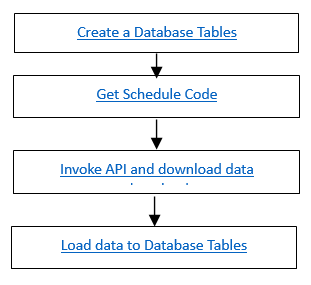
## Steps

1. **Creating Database Tables:**
   * **Purpose:** Set up new tables in your database to store the data you will download.
   * **Action:** Follow the instructions in the “[Creating a Data Base tables](#_Creating_a_Database)” section to define the structure of these tables (e.g., columns, data types).
2. **Downloading Data via API Calls:**
   * **Purpose:** Retrieve the necessary data from an external source.
   * **Action:** Use API calls to download the data. This typically involves sending a request to a web service and receiving data in a format like JSON.
3. **Importing Data into Tables:**
   * **Purpose:** Store the downloaded data in your newly created database tables.
   * **Action:** Convert the data from the API response into a format that can be inserted into your database (e.g., parsing JSON to extract relevant fields) and then insert this data into the tables.
4. **Executing SQL Queries:**
   * **Purpose:** Process the data to prepare it for export.
   * **Action:** Run a set of SQL queries on your database to manipulate and organize the data as needed.
5. **Exporting Data to CSV:**
   * **Purpose:** Create the final PBS text files.
   * **Action:** Export the processed data from your database tables to CSV files. These CSV files will be used to produce the PBS text files.

## Supply Only Indicator Status

The queries in the document can be changed to find a brand for a specific item code that has a supply-only status. In the Item table, the SUPPLY\_ONLY\_INDICATOR flag can be set to ‘Y’ or ‘N’ based on the information needed.

# Steps to Create Database from API Calls



# Creating a Database Tables

|  |  |
| --- | --- |
| **Create a Database Tables** | 1. **Access Your Database:**   * Open your preferred database management tool (e.g., MySQL Workbench, pgAdmin, or SQL Server Management Studio). * DB Schema name can include schema-name and version.   2**. Create a New Tables:**   * If you haven’t already, created a new tables in your database use below script to create tables.   CREATE TABLE <<dbschema>>.[ATC](  ATC\_CODE [NVARCHAR](20),  ATC\_DESCRIPTION [NVARCHAR](255),  ATC\_LEVEL [SMALLINT],  ATC\_PARENT\_CODE [NVARCHAR](10),  SCHEDULE\_CODE [NVARCHAR](20)  )  CREATE TABLE <<dbschema>>.[PRESCRIBER](  [PBS\_CODE] [nvarchar](50),  [PRESCRIBER\_CODE] [nvarchar](10),  [SCHEDULE\_CODE] [float],  [PRESCRIBER\_TYPE] [nvarchar](100)  )  CREATE TABLE <<dbschema>>.[RSTRCTN\_PRSCRBNG\_TXT](  [SCHEDULE\_CODE] [FLOAT] NULL,  [RES\_CODE] [NVARCHAR](83) NULL,  [PRESCRIBING\_TEXT\_ID] [FLOAT] NULL,  [PT\_POSITION] [INT] NULL  )    CREATE TABLE <<dbschema>>.[PRESCRIBING\_TXT](  [SCHEDULE\_CODE] [FLOAT] NULL,  [PRESCRIBING\_TXT\_ID] [FLOAT] NULL,  [PRESCRIBING\_TYPE] [NVARCHAR](100) NULL,  [PRESCRIBING\_TXT] [NVARCHAR](MAX) NULL,  [PRSCRBG\_TXT\_HTML] [NVARCHAR](MAX) NULL,  [COMPLEX\_AUTHORITY\_RQRD\_IND] [NVARCHAR](1) NULL,  [ASSESSMENT\_TYPE\_CODE] [NVARCHAR](20) NULL,  [APPLY\_TO\_INCREASE\_MQ\_FLAG] [NVARCHAR](1) NULL,  [APPLY\_TO\_INCREASE\_NR\_FLAG] [NVARCHAR](1) NULL  )    CREATE TABLE <<dbschema>>.[ORGANISATION](  [ORGANISATION\_ID] [FLOAT],  [SCHEDULE\_CODE] [FLOAT],  [NAME] [NVARCHAR](4000),  [ABN] [NVARCHAR](100),  [STREET\_ADDRESS] [NVARCHAR](MAX),  [CITY] [NVARCHAR](4000),  [STATE] [NVARCHAR](100),  [POSTCODE] [NVARCHAR](30),  [TELEPHONE\_NUMBER] [NVARCHAR](100),  [FACSIMILE\_NUMBER] [NVARCHAR](100)  )  CREATE TABLE <<dbschema>>.[ITEM\_DISPENSING\_RULE\_RLTD](  [SCHEDULE\_CODE] [float] ,  [LI\_ITEM\_ID] [nvarchar](214) ,  [DISPENSING\_RULE\_MNEM] [nvarchar](50) ,  [DISPENSING\_RULE\_REFERENCE] [nvarchar](100) ,  [BRAND\_PREMIUM] [decimal](12, 2) ,  [DISPENSE\_FEE\_TYPE\_CODE] [nvarchar](2) ,  [DANGEROUS\_DRUG\_FEE\_CODE] [nvarchar](2) ,  [THERAPEUTIC\_GROUP\_PREMIUM] [decimal](12, 2) ,  [CMNWLTH\_PRICE\_TO\_PHARMACIST] [decimal](12, 2) ,  [MAN\_PRICE\_TO\_PHARMACIST] [decimal](12, 2) ,  [MAN\_DISPNSD\_PRICE\_MAX\_QTY] [float] ,  [MAX\_RECORD\_VAL\_FOR\_SAFETY\_NET] [float] ,  [CMNWLTH\_DSP\_PRICE\_MAX\_QTY] [decimal](12, 2) ,  [TGM\_PRICE\_PHRMCST] [decimal](12, 2) ,  [TGM\_DISP\_PRICE\_MAX\_QTY] [decimal](12, 2) ,  [SPECIAL\_PATIENT\_CONTRIBUTION] [decimal](12, 2)  )  CREATE TABLE <<dbschema>>.[EXTEMPORANEOUS\_PREPARATION](  [SCHEDULE\_CODE] [float] ,  [PREPARATION] [nvarchar](500) ,  [MAXIMUM\_QUANTITY] [float] ,  [SFP\_PBS\_CODE] [nvarchar](50) ,  [PBS\_CODE] [nvarchar](50) ,  [SFP\_DRUG\_NAME] [nvarchar](200) ,  [SFP\_REFERENCE] [nvarchar](50) ,  [CONTAINER\_FEE] [float] ,  [DISPENSING\_FEE\_MAX\_QUANTITY] [float] ,  [SAFETY\_NET\_PRICE] [float] ,  [MAXIMUM\_PATIENT\_CHARGE] [float] ,  [MAXIMUM\_QUANTITY\_UNIT] [nvarchar](50)  )  CREATE TABLE <<dbschema>>.[ITEM](  [SCHEDULE\_CODE] [float] ,  [LI\_ITEM\_ID] [nvarchar](214) ,  [DRUG\_NAME] [nvarchar](4000) ,  [LI\_DRUG\_NAME] [nvarchar](2000) ,  [LI\_FORM] [nvarchar](500) ,  [SCHEDULE\_FORM] [nvarchar](4000) ,  [BRAND\_NAME] [nvarchar](4000) ,  [PROGRAM\_CODE] [nvarchar](10) ,  [PBS\_CODE] [nvarchar](50) ,  [BENEFIT\_TYPE\_CODE] [nvarchar](1) ,  [CAUTION\_INDICATOR] [nvarchar](1) ,  [NOTE\_INDICATOR] [nvarchar](1) ,  [MANNER\_OF\_ADMINISTRATION] [nvarchar](60) ,  [MAXIMUM\_PRESCRIBABLE\_PACK] [float] ,  [MAXIMUM\_QUANTITY\_UNITS] [float] ,  [NUMBER\_OF\_REPEATS] [float] ,  [ORGANISATION\_ID] [float] ,  [MANUFACTURER\_CODE] [nvarchar](2) ,  [PACK\_SIZE] [float] ,  [PRICING\_QUANTITY] [float] ,  [PACK\_NOT\_TO\_BE\_BROKEN\_IND] [nvarchar](1) ,  [CLAIMED\_PRICE] [float] ,  [DETERMINED\_PRICE] [float] ,  [DETERMINED\_QTY] [nvarchar](1) ,  [SAFETY\_NET\_RESUPPLY\_RULE\_DAYS] [float] ,  [SAFETY\_NET\_RESUP\_RULE\_CNT\_IND] [nvarchar](1) ,  [EXTEMPORANEOUS\_INDICATOR] [nvarchar](1) ,  [EXTEMPORANEOUS\_STANDARD] [nvarchar](50) ,  [DOCTORS\_BAG\_GROUP\_ID] [float] ,  [SECTION100\_ONLY\_INDICATOR] [nvarchar](1) ,  [DOCTORS\_BAG\_ONLY\_INDICATOR] [nvarchar](1) ,  [BRAND\_SUBSTITUTION\_GROUP\_ID] [float] ,  [BRAND\_SUBSTITUTION\_GROUP\_CODE] [nvarchar](1) ,  [CONTINUED\_DISPENSING\_EMERGENCY] [nvarchar](1) ,  [CONTINUED\_DISPENSING\_FLAG] [nvarchar](1) ,  [SUPPLY\_ONLY\_INDICATOR] [nvarchar](1) ,  [SUPPLY\_ONLY\_DATE] [nvarchar](10) ,  [NON\_EFFECTIVE\_DATE] [nvarchar](10) ,  [WEIGHTED\_AVG\_DISCLOSED\_PRICE] [float] ,  [ORIGINATOR\_BRAND\_INDICATOR] [nvarchar](1) ,  [INNOVATOR\_INDICATOR] [nvarchar](1) ,  [PAPER\_MED\_CHART\_ELIGIBLE\_IND] [nvarchar](1) ,  [ELECT\_MED\_CHART\_ELIGIBLE\_IND] [nvarchar](1) ,  [HSPTL\_MED\_CHART\_ELIGIBLE\_IND] [nvarchar](1) ,  [PAPER\_MED\_CHART\_DURATION] [float] ,  [ELECT\_MED\_CHART\_DURATION] [float] ,  [HSPTL\_CHART\_ACUTE\_DURATION] [float] ,  [HSPTL\_CHART\_SUB\_ACUTE\_DURATION] [float] ,  [HSPTL\_CHART\_CHRONIC\_DURATION] [float] ,  [PACK\_CONTENT] [decimal](12, 0) ,  [VIAL\_CONTENT] [decimal](12, 0) ,  [INFUSIBLE\_INDICATOR] [nvarchar](1) ,  [UNIT\_OF\_MEASURE] [nvarchar](50) ,  [MAXIMUM\_AMOUNT] [decimal](12, 0) ,  [FORMULARY] [nvarchar](100) ,  [WATER\_ADDED\_IND] [nvarchar](1) ,  [SECTION\_19A\_EXPIRY\_DATE] [nvarchar](10) ,  [CONTAINER\_FEE\_TYPE] [nvarchar](50) ,  [POLICY\_APPLIED\_BIO\_SIM\_UP\_FLAG] [nvarchar](1) ,  [POLICY\_APPLIED\_IMDQ60\_FLAG] [nvarchar](1) ,  [POLICY\_APPLIED\_IMDQ60\_BASE\_FLAG] [nvarchar](1) ,  [POLICY\_APPLIED\_INDIG\_PHAR\_FLAG] [nvarchar](1) ,  [THERAPEUTIC\_EXEMPTION\_INDICATOR] [nvarchar](1) ,  [PREMIUM\_EXEMPTION\_GROUP\_ID] [float] ,  [DOCTORS\_BAG\_GROUP\_TITLE] [nvarchar](100) ,  [THERAPEUTIC\_GROUP\_ID] [float] ,  [THERAPEUTIC\_GROUP\_TITLE] [nvarchar](100) ,  [ADVANCED\_NOTICE\_DATE] [nvarchar](10) ,  [SUPPLY\_ONLY\_END\_DATE] [nvarchar](10) ,  [FIRST\_LISTED\_DATE] [nvarchar](10) ,  [LEGAL\_UNAR\_IND] [nvarchar](1) ,  [LEGAL\_CAR\_IND] [nvarchar](1)  )  CREATE TABLE <<dbschema>>.[ITEM\_ATC\_RLT](  [ATC\_CODE] [nvarchar](10) ,  [SCHEDULE\_CODE] [float] ,  [PBS\_CODE] [nvarchar](50) ,  [ATC\_PRIORITY\_PCT] [float]  )  CREATE TABLE <<dbschema>>.[COPAYMENT](  [SCHEDULE\_CODE] [FLOAT] NULL,  [GENERAL] [FLOAT] NULL,  [CONCESSIONAL] [FLOAT] NULL,  [SAFETY\_NET\_GENERAL] [FLOAT] NULL,  [SAFETY\_NET\_CONCESSIONAL] [FLOAT] NULL,  [SAFETY\_NET\_CARD\_ISSUE] [FLOAT] NULL,  [INCREASED\_DISCOUNT\_LIMIT] [DECIMAL](12, 2) NULL,  [SAFETY\_NET\_CTG\_CONTRIBUTION] [DECIMAL](12, 2) NULL  ) |

# Invoke API and download data

## URL to view PBS API Catalogue.

|  |  |
| --- | --- |
| **API Catalogue** Link to provide key metadata and documentation for all the API’s | [APIs: Details - Department of Health and Aged Care API Catalogue](https://data-api-portal.health.gov.au/api-details#api=pbs-prod-api-public-v3&operation=get-api-v3-amt-items) |

## Steps to download API data

|  |  |
| --- | --- |
| **Download API Endpoint Data**  **Prerequisite**: Obtaining the Schedule Code  Before retrieving caution information, follow these steps:   1. Make a call to the API to obtain the schedule code. 2. Refer to the appendix for the specific URL related to schedules. | 1. **Open Postman:**   * + If you haven’t already, download and install Postman on your machine.   + Launch Postman.   2. **Create a New Request:**   * Click the “+” icon in the workbench to open a new tab. * In the request URL field, enter the following endpoint:   [https://data-api.health.gov.au/pbs/api/v3/<<Operation>>? [?schedule\_code=<<nnnn>>][&limit][&page]](https://data-api.health.gov.au/pbs/api/v3/%3c%3cOperation%3e%3e?%20%5b?schedule_code=%3c%3cnnnn%3e%3e%5d%5b&limit%5d%5b&page%5d)  Note :-  <<nnnn>> can be obtained from making schedules call.  <<Operation>> can be items,fees..etc full list can be found here in  [API Catalogue](#_URL_to_view)  3. **Set Headers:**   * + Add the following headers to your request:   + Accept: text/csv   + Subscription-Key: <use your given subscription key>   4. **Send the Request:**   * + Click the “Send” button to execute the request.   + Postman will display the response from the API.   5. **Save the Response:**  Save the comma separated values (csv) response data into a file named operations-texts.csv. |

## Steps to Load API data into database and verify.

|  |  |
| --- | --- |
| **Load data** | 1. **Access Your Database:**   * Open your preferred database management tool (e.g., MySQL Workbench, pgAdmin, or SQL Server Management Studio).   2. **Import Data:**   * Use the appropriate SQL command to load data from the CSV file into the "operation-texts.csv” table. For example:   SQL  LOAD DATA INFILE 'path/to/operation-texts.csv'  INTO TABLE <<Table>>  FIELDS TERMINATED BY ',' ENCLOSED BY '"'  LINES TERMINATED BY '\n'  IGNORE 1 LINES; -- Skip the header row if present  3 **Verify Data**  SELECT \* FROM <<Table>>;  4. **Adjust as Needed:**   * Depending on your specific requirements, you may need to further manipulate or transform the data within the "prescribing” table. schedule\_code [nvarchar](20)   ) |
| **Store Results** | 1. **Execute Query:**   * Run the above query in your database management tool to retrieve the data.   2. **Export Data:**   * + Export the query results to your chosen format:   + For CSV: Save the results as a CSV file.   + For Excel: Export to an Excel spreadsheet.   + For JSON: Format the data as a JSON file.   **3. Verify Export:**  Open the exported file to ensure that the data is accurate and properly formatted. |

# AMT

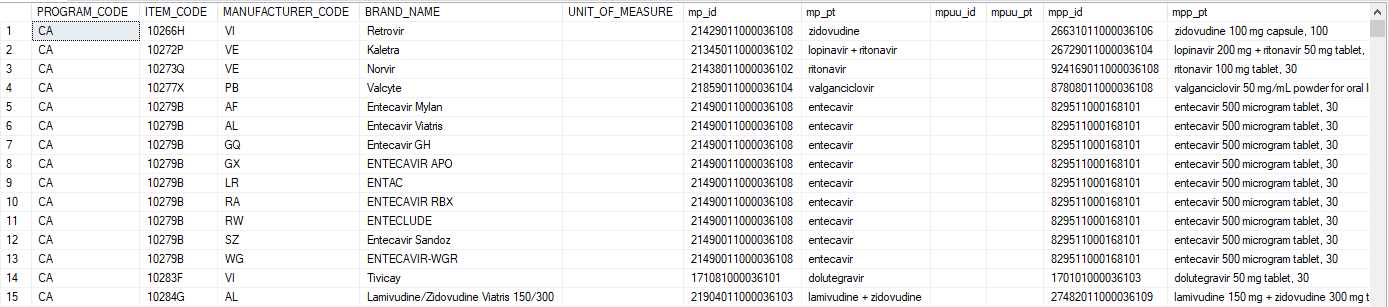
## Steps to download and import amt.txt data and store results

|  |  |
| --- | --- |
| **Testing results** | Status :- **In Progress**  **Current text file will be available until data is available.**   1. We have identified an issue in the following text file columns:   tpp\_pt – Content does not exist in the API for Non-AMT.  cemp-tpp (Proportional Published Ex-Manufacturer Price)– Content does not exist in the API.  tpuu\_id – Content does not exist.  2. PFDI is not in API data. From 1 July 2020, the Premium-free Dispensing Incentive (PFDI) was discontinued, with funding reallocated to other 7CPA components including the significant increases to the dispensing fee, AHI and dangerous drug fee. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /items | Retrieve information about items | | /item-amt | Retrieve information about items amt text relationship | | /item-dispensing-rule-relationships | Retrieve information about item dispensing rule relationship | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract AMT Data** | WITH main AS (  SELECT DISTINCT  ITEM.PROGRAM\_CODE,  ITEM.DRUG\_NAME,  ITEM.SCHEDULE\_FORM,  ITEM.PBS\_CODE AS item\_code,  ITEM.MANUFACTURER\_CODE,  ITEM.BRAND\_NAME,  CASE WHEN ITEM.UNIT\_OF\_MEASURE IS NOT NULL THEN concat(ITEM.PACK\_SIZE,' ',ITEM.UNIT\_OF\_MEASURE)  ELSE ITEM.UNIT\_OF\_MEASURE END AS UNIT\_OF\_MEASURE,  ITEM.VIAL\_CONTENT AS VIAL\_CONTENT,  ITEM.MAXIMUM\_PRESCRIBABLE\_PACK AS mq\_pack,  CASE  WHEN ITEM.MAXIMUM\_QUANTITY\_UNITS IS NULL THEN ITEM.MAXIMUM\_AMOUNT  ELSE ITEM.MAXIMUM\_QUANTITY\_UNITS END AS mq\_uu ,  ITEM.SCHEDULE\_CODE,  ITEM.LI\_ITEM\_ID,  CASE  WHEN ITEM.PROGRAM\_CODE IN('IP','PL','IN') THEN ITEM.DETERMINED\_PRICE  WHEN ITEM.PROGRAM\_CODE IN('R1','PQ','GE','DB') THEN ITEM.CLAIMED\_PRICE  ELSE  ITEM\_DISP\_RLTD.CMNWLTH\_PRICE\_TO\_PHARMACIST END AS cemp\_tpp,  CASE  WHEN ITEM.PROGRAM\_CODE IN('IN','IP','TY') THEN ROUND(ITEM\_DISP\_RLTD.CMNWLTH\_PRICE\_TO\_PHARMACIST,0)  WHEN ITEM.PROGRAM\_CODE IN('TZ') THEN ITEM.CLAIMED\_PRICE  ELSE 0 END AS cemp\_tpuu,  CASE  WHEN ITEM.PROGRAM\_CODE IN('IP','GE') THEN ITEM.DETERMINED\_PRICE  WHEN ITEM.PROGRAM\_CODE IN('R1','PQ','PL','IN','DB','TZ') THEN ITEM.CLAIMED\_PRICE  WHEN ITEM.PROGRAM\_CODE IN('TY') THEN ITEM\_DISP\_RLTD.CMNWLTH\_PRICE\_TO\_PHARMACIST  ELSE ITEM\_DISP\_RLTD.MAN\_PRICE\_TO\_PHARMACIST END AS memp\_tpp,  CASE  WHEN ITEM.PROGRAM\_CODE IN('IP','IN','TY') THEN CAST(ITEM\_DISP\_RLTD.CMNWLTH\_PRICE\_TO\_PHARMACIST AS VARCHAR)  WHEN ITEM.PROGRAM\_CODE IN('TZ') THEN ITEM.CLAIMED\_PRICE  ELSE '' END AS memp\_tpuu FROM  <<dbschema>>.ITEM AS ITEM  INNER JOIN  <<dbschema>>.[ITEM\_DISPENSING\_RULE\_RLTD] AS ITEM\_DISP\_RLTD  ON  ITEM.LI\_ITEM\_ID = ITEM\_DISP\_RLTD.LI\_ITEM\_ID  AND ITEM.SCHEDULE\_CODE = ITEM\_DISP\_RLTD.SCHEDULE\_CODE  INNER JOIN  <<dbschema>>.[ITEM\_AMT] AS ITEM\_AMT  ON  ITEM.LI\_ITEM\_ID = ITEM\_AMT.LI\_ITEM\_ID  AND ITEM.SCHEDULE\_CODE = ITEM\_AMT.SCHEDULE\_CODE  WHERE  ITEM.SUPPLY\_ONLY\_INDICATOR = 'N' AND  ITEM.MANUFACTURER\_CODE NOT IN('JB','ZP')  AND ITEM.SCHEDULE\_CODE = (  SELECT SCHEDULE\_CODE  FROM <<dbschema>>.[SCHEDULE]  WHERE EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED'  )  ),  combined AS (  SELECT DISTINCT  main.\*,  ISNULL(ITEM\_AMT\_MP.amt\_code, concat(ITEM\_AMT\_MP.PBS\_CONCEPT\_ID,100014410))AS mp\_id,  ISNULL(MAIN.DRUG\_NAME,ITEM\_AMT\_MP.PREFERRED\_TERM) AS mp\_pt,  ISNULL(ITEM\_AMT\_MPP.amt\_code, concat(ITEM\_AMT\_MPP.PBS\_CONCEPT\_ID,100014410))AS mpp\_id,  ISNULL(ITEM\_AMT\_MPP.PREFERRED\_TERM,MAIN.SCHEDULE\_FORM) AS mpp\_pt,  ISNULL(ITEM\_AMT\_TPP.amt\_code, concat(ITEM\_AMT\_TPP.PBS\_CONCEPT\_ID,100014410))AS tpp\_id,  ISNULL(ITEM\_AMT\_TPP.PREFERRED\_TERM,REPLACE(MAIN.SCHEDULE\_FORM,MAIN.DRUG\_NAME,MAIN.BRAND\_NAME))AS tpp\_pt  FROM  main  LEFT JOIN  <<dbschema>>.ITEM\_AMT AS ITEM\_AMT\_MP  ON  main.LI\_ITEM\_ID = ITEM\_AMT\_MP.LI\_ITEM\_ID  AND main.SCHEDULE\_CODE = ITEM\_AMT\_MP.SCHEDULE\_CODE  AND ITEM\_AMT\_MP.CONCEPT\_TYPE\_CODE = 'MP'  LEFT JOIN  <<dbschema>>.ITEM\_AMT AS ITEM\_AMT\_MPP  ON  main.LI\_ITEM\_ID = ITEM\_AMT\_MPP.LI\_ITEM\_ID  AND main.SCHEDULE\_CODE = ITEM\_AMT\_MPP.SCHEDULE\_CODE  AND ITEM\_AMT\_MPP.CONCEPT\_TYPE\_CODE = 'MPP'  LEFT JOIN  <<dbschema>>.ITEM\_AMT AS ITEM\_AMT\_TPP  ON  main.LI\_ITEM\_ID = ITEM\_AMT\_TPP.LI\_ITEM\_ID  AND main.SCHEDULE\_CODE = ITEM\_AMT\_TPP.SCHEDULE\_CODE  AND ITEM\_AMT\_TPP.CONCEPT\_TYPE\_CODE = 'TPP'  )  SELECT DISTINCT  combined.PROGRAM\_CODE, combined.ITEM\_CODE,combined.MANUFACTURER\_CODE,combined.BRAND\_NAME,  ISNULL(TRY\_CONVERT(varchar, combined.UNIT\_OF\_MEASURE), '') AS UNIT\_OF\_MEASURE,  combined.mp\_id,combined.mp\_pt,    CASE WHEN MPUU.PBS\_CONCEPT\_ID IS NULL THEN ''  ELSE ISNULL(TRY\_CONVERT(varchar, ISNULL(MPUU.AMT\_CODE, concat(MPUU.PBS\_CONCEPT\_ID, 100014410))), '') END AS mpuu\_id,  ISNULL(TRY\_CONVERT(varchar, MPUU.PREFERRED\_TERM),'') AS mpuu\_pt,  combined.mpp\_id,combined.mpp\_pt,  combined.tpp\_id,combined.tpp\_pt,  ISNULL(TRY\_CONVERT(varchar, combined.VIAL\_CONTENT), '') AS VIAL\_CONTENT  ,ISNULL(TRY\_CONVERT(varchar,combined.MQ\_PACK), '') AS MQ\_PACK ,combined.MQ\_UU,  combined.CEMP\_TPP,combined.CEMP\_TPUU,  combined.MEMP\_TPP,  CASE WHEN combined.memp\_tpuu = 0 THEN ''  ELSE ISNULL(TRY\_CONVERT(varchar, combined.memp\_tpuu),'') END as memp\_tpuu  FROM  combined  LEFT JOIN (  SELECT  ITEM\_AMT\_MPUU.LI\_ITEM\_ID,  ITEM\_AMT\_MPUU.SCHEDULE\_CODE,  ITEM\_AMT\_MPUU.AMT\_CODE,  ITEM\_AMT\_MPUU.PBS\_CONCEPT\_ID,  ITEM\_AMT\_MPUU.PREFERRED\_TERM,  ITEM\_AMT\_MPUU.CONCEPT\_TYPE\_CODE,  ROW\_NUMBER() OVER (PARTITION BY SCHEDULE\_CODE,LI\_ITEM\_ID,CONCEPT\_TYPE\_CODE ORDER BY (ISNULL(AMT\_CODE, concat(PBS\_CONCEPT\_ID,100014410)))) AS RowNum  FROM  <<dbschema>>.ITEM\_AMT AS ITEM\_AMT\_MPUU  WHERE ITEM\_AMT\_MPUU.CONCEPT\_TYPE\_CODE = 'MPUU'  ) MPUU ON MPUU.LI\_ITEM\_ID = combined.LI\_ITEM\_ID  AND MPUU.SCHEDULE\_CODE= combined.SCHEDULE\_CODE  AND combined.PROGRAM\_CODE IN ('IN', 'IP', 'TZ', 'TY')  And RowNum = 1 |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample ATM Data



# ATC

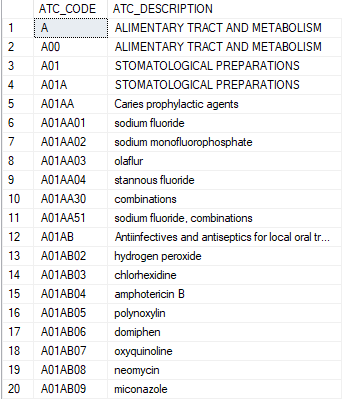
## Steps to download and import atc.txt data store results

|  |  |
| --- | --- |
| **Testing Results** | Text files and API data extracted show a difference of extra white space in middle/end of text in the sentences. This is the same as the PBS data. Space or return shows as a space. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /atc-codes | Retrieve information about atc codes | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract ATC Data** | **Specific date**  SELECT DISTINCT  ATC\_CODE,  ATC\_DESCRIPTION  FROM  <<dbschema>>.ATC  WHERE  SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.SCHEDULE  WHERE  EFFECTIVE\_DATE = ' <<effective-date>> '  AND PUBLICATION\_STATUS = 'PUBLISHED')  ORDER BY  ATC\_CODE; |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample ATC data



# Authorities Restrictions

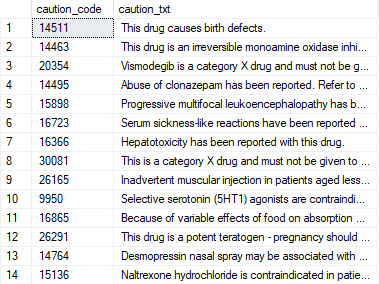
## Steps to download and import CautionExtract.txt data and store results

|  |  |
| --- | --- |
| **Testing results** | Text files and API data extracted show a difference of extra white space in middle/end of text in the sentences. This is the same as the PBS data. Space or return shows as a space. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /prescribers | Retrieve information about prescribers | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Caution Data** | **Note:-** Querying Prescribing\_text table returns Caution data.  **Specific date**  SELECT  prescribing\_txt\_id as caution\_code,  prescribing\_txt as caution\_txt  FROM  <<dbschema>>.[ prescribing\_txt]  WHERE  PRESCRIBING\_TYPE = 'CAUTION'  AND SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.[SCHEDULE  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' )  **Latest Schedule date**  SELECT  prescribing\_txt\_id as caution\_code,  prescribing\_txt as caution\_txt  FROM  <<dbschema>>.[ prescribing\_txt]  WHERE  PRESCRIBING\_TYPE = 'CAUTION'  AND SCHEDULE\_CODE =  (  SELECT  top(1) SCHEDULE\_CODE  FROM  <<dbschema>>.[SCHEDULE  ORDER BY  EFFECTIVE\_DATE desc,  REVISION\_NUMBER desc ) |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample Caution data



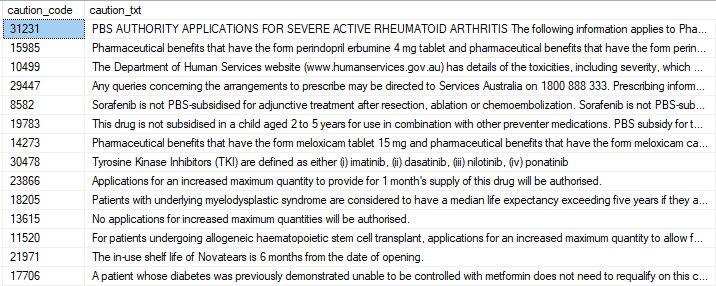
## Steps to download and import NoteExtract.txt and store results

|  |  |
| --- | --- |
| **Testing Results** | Text files and API data extracted show a difference of extra white space in middle/end of text in the sentences. This is the same as the PBS data. Space or return shows as a space. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /prescribers | Retrieve information about prescribers | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Notes Data** | **Note:-** Querying Prescribing\_text table returns Notes data.  **Specific date**  SELECT  prescribing\_txt\_id as caution\_code,  prescribing\_txt as caution\_txt  FROM  <<dbschema>>.[ prescribing\_txt]  WHERE  PRESCRIBING\_TYPE = 'ADMINISTRATIVE\_ADVICE'  AND SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.[SCHEDULE  WHERE  EFFECTIVE\_DATE = ' <<effective-date>> '  AND PUBLICATION\_STATUS = 'PUBLISHED' )  **Latest Schedule date**  SELECT  prescribing\_txt\_id as caution\_code,  prescribing\_txt as caution\_txt  FROM  <<dbschema>>.[ prescribing\_txt]  WHERE  PRESCRIBING\_TYPE = 'ADMINISTRATIVE\_ADVICE'  AND SCHEDULE\_CODE =  (  SELECT  top(1) SCHEDULE\_CODE  FROM  SCHEDULE  ORDER BY  EFFECTIVE\_DATE desc,  REVISION\_NUMBER desc ) |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample Note data



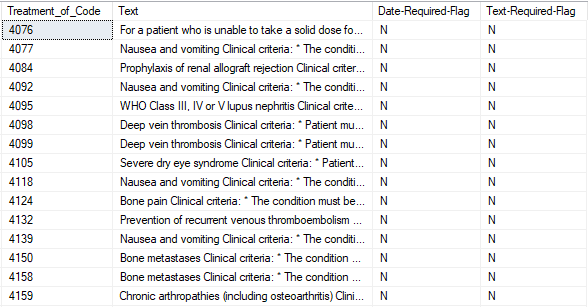
## Steps to download and import RestictionExtract.txt and store results

|  |  |
| --- | --- |
| **Testing results** | There are two restriction text files, Fixed and Delimited, but only one example is provided.  The main difference noted is that the misc-flag column exists in the text file but not in the API result-set. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /restrictions | Retrieve information about restrictions. | | /restriction-prescribing-text-relationships | Retrieve information about restriction prescribing relationships | | /prescribing-texts | Retrieve information about prescribing texts. | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Caution Data** | **Specific date**  with Restrictions\_Text as  (  SELECT DISTINCT  Restrictions.[TREATMENT\_OF\_CODE] AS [Treatment\_of\_Code],  [PRESCRIBING\_TXT\_ID] ,  CASE  WHEN  [PRESCRIBING\_TX].PRESCRIBING\_TXT not like '%.'  and PRESCRIBING\_TYPE = 'CRITERIA'  THEN  [PRESCRIBING\_TX].PRESCRIBING\_TXT + '.'  ELSE  [PRESCRIBING\_TX].PRESCRIBING\_TXT  END AS TEXT ,  'N' AS [Date-Required-Flag] ,  'N' AS [Text-Required-Flag] ,  PT\_POSITION ,  PRESCRIBING\_TYPE ,  CRITERIA\_TYPE ,  TREATMENT\_PHASE,  CASE  WHEN  criteria\_type is not null  then  ROW\_NUMBER() OVER  (  PARTITION BY restrictions.schedule\_code,  treatment\_of\_code ,  CRITERIA\_TYPE  order by pt\_position  )  ELSE  NULL  END as "c\_pos"  FROM  <<dbschema>>.[RESTRICTION\_TEXT] Restrictions  INNER JOIN  <<dbschema>>.[RSTRCTN\_PRSCRBNG\_TXT\_RLTD] RSTRCTN\_PRSCRBNG  ON  Restrictions.RES\_CODE = RSTRCTN\_PRSCRBNG.RES\_CODE  AND Restrictions.SCHEDULE\_CODE = RSTRCTN\_PRSCRBNG.SCHEDULE\_CODE  INNER JOIN  <<dbschema>>.[PRESCRIBING\_TXT] AS [PRESCRIBING\_TX]  ON  PRESCRIBING\_TX.PRESCRIBING\_TXT\_ID = RSTRCTN\_PRSCRBNG.PRESCRIBING\_TEXT\_ID  AND Restrictions.SCHEDULE\_CODE = [PRESCRIBING\_TX].SCHEDULE\_CODE  LEFT JOIN  <<dbschema>>.[CRITERIA] as Criteria  on  Criteria.CRITERIA\_PRESCRIBING\_TXT\_ID = [PRESCRIBING\_TX].PRESCRIBING\_TXT\_ID  and Criteria.SCHEDULE\_CODE = Restrictions.SCHEDULE\_CODE  WHERE  [Restrictions].SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.SCHEDULE  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' )  and PRESCRIBING\_TYPE not in ( 'ADMINISTRATIVE\_ADVICE',  'CAUTION')  ) ,  m as  (  select  CASE  WHEN  c\_pos = 1  THEN  replace(res\_txt.TEXT, 'criteria:', 'criteria: \*')  WHEN  c\_pos > 1  THEN  replace(res\_txt.TEXT, lower(criteria\_type) +' criteria: ', ', AND \* ')  ELSE  res\_txt.TEXT  END as modified\_txt ,  res\_txt.\*  from  Restrictions\_Text res\_txt )  SELECT  res\_txt.Treatment\_of\_Code,  case when TREATMENT\_PHASE is null then  REPLACE(STRING\_AGG(res\_txt.modified\_txt, ' ') WITHIN GROUP (ORDER BY res\_txt.PT\_POSITION), '. ,', ',')  else  REPLACE(REPLACE(STRING\_AGG(res\_txt.modified\_txt, ' ') WITHIN GROUP (ORDER BY res\_txt.PT\_POSITION), '. ,', ','), 'Clinical criteria:', CONCAT('Treatment Phase: ', TREATMENT\_PHASE , ' Clinical criteria:'))  end AS Text,  res\_txt.[Date-Required-Flag],  res\_txt.[Text-Required-Flag]  FROM  m res\_txt  GROUP BY  res\_txt.Treatment\_of\_Code,  res\_txt.[Date-Required-Flag],  res\_txt.[Text-Required-Flag],  res\_txt.TREATMENT\_PHASE |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample Restriction Data



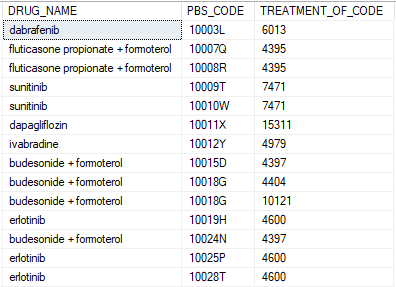
## Steps to download and import streamlined.txt and store results

|  |  |
| --- | --- |
| **Testing results** | Text files and API data extracted are matching with no issues found. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /items | Retrieve information about Items | | /item-restriction-relationships | Retrieve information about Prescriber | | /restrictions | Retrieve information about restrictions. | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Streamlined Authority Items Data** | **Specific date**  SELECT DISTINCT  [Items].DRUG\_NAME,  [Items].PBS\_CODE ,  [Restrictions].TREATMENT\_OF\_CODE  FROM  (<<DBSCHEMA>>.[ITEM] Items INNER JOIN <<DBSCHEMA>>.[ITEM\_RESTRICTION\_RLTD][Item-restriction-relationships] ON [Items].PBS\_CODE = [Item-restriction-relationships].PBS\_CODE  AND [Items].SCHEDULE\_CODE = [Item-restriction-relationships].SCHEDULE\_CODE)  INNER JOIN  <<DBSCHEMA>>.[RESTRICTION\_TEXT] Restrictions  ON  [Item-restriction-relationships].RES\_CODE = [Restrictions].RES\_CODE  AND [Item-restriction-relationships].SCHEDULE\_CODE = [Restrictions].SCHEDULE\_CODE  WHERE  (  (  (  [Restrictions].TREATMENT\_OF\_CODE) Is Not Null)  AND (  (  [Items].BENEFIT\_TYPE\_CODE)='S'  AND [Items].SUPPLY\_ONLY\_INDICATOR = 'N')  AND [Items].SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<DBSCHEMA>>.[SCHEDULE]  WHERE  EFFECTIVE\_DATE = ' <<effective-date>> '  AND PUBLICATION\_STATUS = 'PUBLISHED'))  and TREATMENT\_OF\_CODE <> 9999  order by  PBS\_CODE |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample Streamlined Authority Items data



# Continued Dispensing (CD)

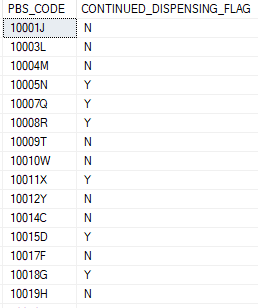
## Steps to download and import cd.txt data and store results

|  |  |
| --- | --- |
| **Testing results** | Text files and API data extracted are matching with no issues found. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /items | Retrieve information about atc codes | | /extemporaneous-preparations | Retrieve information about Extemporaneous Preparations. | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Continued Dispensing (CD) Data** | **Specific date**  SELECT  T.\*  FROM  (  SELECT DISTINCT  ISNULL(REPLICATE('0',5-LEN(PBS\_CODE)) + PBS\_CODE, PBS\_CODE) AS "item-code-padded" ,  ISNULL(  CASE  WHEN  CONTINUED\_DISPENSING\_EMERGENCY = 'Y'  THEN  'Y'  ELSE  CONTINUED\_DISPENSING\_FLAG  END, 'N') AS "cont-disp"  FROM  <<dbschema>>.[ITEM] I  WHERE  SUPPLY\_ONLY\_INDICATOR='N'  AND (  PROGRAM\_CODE <> 'EP'  OR PBS\_CODE IN  (  SELECT  EP.PBS\_CODE  FROM  <<dbschema>>.EXTEMPORANEOUS\_PREPARATION EP  WHERE  EP.SCHEDULE\_CODE = I.SCHEDULE\_CODE ) )  AND SCHEDULE\_CODE =  (  SELECT  S.SCHEDULE\_CODE  FROM  <<DB-SCHEMA>>.SCHEDULE S  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' )    UNION ALL    SELECT  \*  FROM  (VALUES ('0001A', 'N'), ('V99F', 'N')) AS T("item-code-padded","cont-disp") ) T  ORDER BY  (  CASE  WHEN  "item-code-padded" LIKE '[A-Z]%'  THEN  0  ELSE  1  END),  "item-code-padded" ASC |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample CD data



# Drug data

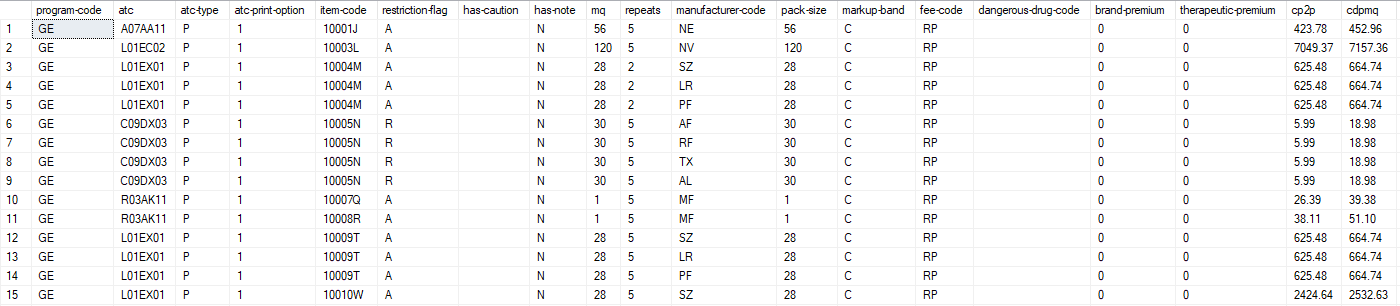
## Steps to download and import drug.txt and store results

|  |  |
| --- | --- |
| **Testing results** | Health is reviewing the business rules to identify the correct values for the differences identified.   * Text file truncated drug name have caused issues with prescribers and dispensers in the past. Fields like mp-pt have an 80-character length limit, but the API is displaying the full content without truncation. * mp-pt differences for RPBS - Investigation underway * Manufacture price to Pharmacy – Investigation underway API is NULL for infusibles and Extemp. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /items | Retrieve information about Items | | /item-atc-relationships | Retrieve information about Item ATC Relationships Rule Relationships | | /items-dispensing-rule-relationships | Retrieve information about Items Dispensing Rule Relationships | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Drug Data** | SELECT [Items].PROGRAM\_CODE as "program-code",  [Item\_atc\_relationships].ATC\_CODE as "atc",  (SELECT CASE WHEN rn = 1 then 'P' else 'S' END  FROM(  SELECT [ATC\_CODE]  , ROW\_NUMBER() OVER(PARTITION BY [SCHEDULE\_CODE], [PBS\_CODE] ORDER BY [ATC\_PRIORITY\_PCT] DESC, [ATC\_CODE] ASC) rn  FROM [MDSDDSPresentDev].<<dbschema>>.[ITEM\_ATC\_RLTD] iar  where iar.SCHEDULE\_CODE = items.schedule\_code  and iar.pbs\_code = items.pbs\_code  ) t  where t.atc\_code = [Item\_atc\_relationships].ATC\_CODE  ) AS "atc-type",  1 as "atc-print-option",  [Items].PBS\_CODE as "item-code",  IIf([Items].[BENEFIT\_TYPE\_CODE]='S','A',[Items].[BENEFIT\_TYPE\_CODE]) AS "restriction-flag",  CASE WHEN [Items].CAUTION\_INDICATOR = 'Y' then 'C' else '' end as "has-caution",  CASE WHEN [Items].NOTE\_INDICATOR ='Y' then 'N' else '' end AS "has-note",  CASE WHEN [Items].MAXIMUM\_QUANTITY\_UNITS IS NULL THEN  [Items].MAXIMUM\_AMOUNT  ELSE  [Items].MAXIMUM\_QUANTITY\_UNITS END AS "mq",  [Items].NUMBER\_OF\_REPEATS AS "repeats",  [Items].MANUFACTURER\_CODE AS "manufacturer-code",  [Items].PACK\_SIZE AS "pack-size",  CASE WHEN PROGRAM\_CODE IN ('CA','GH','HS','IF','IN','MF','TY') then 'D'  WHEN DISPENSING\_RULE\_MNEM = 's90-cp' THEN 'C'  WHEN DISPENSING\_RULE\_MNEM like '%public%' then 'A'  when DISPENSING\_RULE\_MNEM = 'ds-assoc' then 'E'  else null  END AS "markup-band",  [Item\_dispensing\_rule\_relationships].DISPENSE\_FEE\_TYPE\_CODE as "fee-code",  COALESCE(CONVERT(VARCHAR(2), [Item\_dispensing\_rule\_relationships].DANGEROUS\_DRUG\_FEE\_CODE), '') AS "dangerous-drug-code",  [Item\_dispensing\_rule\_relationships].BRAND\_PREMIUM as "brand-premium",  case when [items].PBS\_CODE ='1888J' or [items].PBS\_CODE ='3310F' or [items].PBS\_CODE ='8298R' then  [Item\_dispensing\_rule\_relationships].SPECIAL\_PATIENT\_CONTRIBUTION  else  [Item\_dispensing\_rule\_relationships].THERAPEUTIC\_GROUP\_PREMIUM end as "therapeutic-premium",  [Item\_dispensing\_rule\_relationships].CMNWLTH\_PRICE\_TO\_PHARMACIST as cp2p,  [Item\_dispensing\_rule\_relationships].CMNWLTH\_DSP\_PRICE\_MAX\_QTY as cdpmq,  [Item\_dispensing\_rule\_relationships].TGM\_PRICE\_PHRMCST as lp2p,  [Item\_dispensing\_rule\_relationships].TGM\_DISP\_PRICE\_MAX\_QTY as ldpmq,  IIf([Item\_dispensing\_rule\_relationships].[BRAND\_PREMIUM]='0.00','0.00',[Item\_dispensing\_rule\_relationships].[MAN\_PRICE\_TO\_PHARMACIST]) AS mp2p,  IIf([Item\_dispensing\_rule\_relationships].[BRAND\_PREMIUM]='0.00','0.00',[Item\_dispensing\_rule\_relationships].[MAN\_DISPNSD\_PRICE\_MAX\_QTY]) AS mdpmq,  [Item\_dispensing\_rule\_relationships].MAX\_RECORD\_VAL\_FOR\_SAFETY\_NET as mrvsn,  COALESCE(CONVERT(varchar(1),[Items].BRAND\_SUBSTITUTION\_GROUP\_CODE), '') as bioequivalence,  [Items].BRAND\_NAME as "brand-name",  [Items].DRUG\_NAME as "mp-pt",  CONCAT([Items].BRAND\_NAME,' ',[Items].VIAL\_CONTENT,' ',[Items].UNIT\_OF\_MEASURE,' ',[Items].LI\_FORM) as "tpuu-or-mpp-pt"  FROM <<dbschema>>.[ITEM] as [Items]  left JOIN <<dbschema>>.[ITEM\_ATC\_RLTD] as Item\_atc\_relationships  ON [Items].PBS\_CODE = [Item\_atc\_relationships].PBS\_CODE  and [Items].SCHEDULE\_CODE = [Item\_atc\_relationships].SCHEDULE\_CODE  LEFT JOIN <<dbschema>>.[ITEM\_DISPENSING\_RULE\_RLTD] as [Item\_dispensing\_rule\_relationships]  ON [Items].LI\_ITEM\_ID = [Item\_dispensing\_rule\_relationships].LI\_ITEM\_ID  AND [Items].SCHEDULE\_CODE = [Item\_dispensing\_rule\_relationships].SCHEDULE\_CODE  AND DISPENSING\_RULE\_MNEM = (SELECT TOP (1) idr2.[DISPENSING\_RULE\_MNEM]  FROM [MDSDDSPresentDev].<<dbschema>>.[ITEM\_DISPENSING\_RULE\_RLTD] idr2  WHERE idr2.SCHEDULE\_CODE = [Item\_dispensing\_rule\_relationships].SCHEDULE\_CODE and idr2.LI\_ITEM\_ID = [Item\_dispensing\_rule\_relationships].LI\_ITEM\_ID  order by CASE  WHEN idr2.[DISPENSING\_RULE\_MNEM] like '%cp%' THEN 1  WHEN idr2.[DISPENSING\_RULE\_MNEM] like '%private%' THEN 2  WHEN idr2.[DISPENSING\_RULE\_MNEM] like '%public%' THEN 3  ELSE 4  END asc)  WHERE [Items].SUPPLY\_ONLY\_INDICATOR='N'  and [Items].PROGRAM\_CODE <> 'EP'  and [Items].SCHEDULE\_CODE = (  SELECT SCHEDULE\_CODE  FROM <<dbschema>>.[SCHEDULE]  WHERE EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED'  )  order by items.PBS\_CODE |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample Drug data



# Fees

## Steps to download and import Patient Contribution Fees (fees.txt) and store results

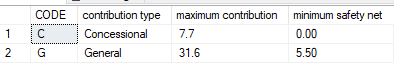
|  |  |
| --- | --- |
| **Testing results** | Status :-**In Progress**  **Current text file will be available until data is available.**  Current values are hard-coded in the query, and the results can vary based on changes in the database values. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /copayments | Retrieve information about copayments. | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Caution Data** | **Specific date**  SELECT  'C' AS CODE ,  'Concessional' AS 'contribution type' ,  CONCESSIONAL AS 'maximum contribution',  '0.00' AS 'minimum safety net'  FROM  <<DBSCHEMA>>.[COPAYMENT]  WHERE  SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<DBSCHEMA>>.[SCHEDULE  WHERE  EFFECTIVE\_DATE = ' <<effective-date>> '  AND PUBLICATION\_STATUS = 'PUBLISHED' )  union  SELECT  'G' AS CODE ,  'General' AS 'contribution type' ,  GENERAL AS 'maximum contribution',  '5.50' AS 'minimum safety net'  FROM  <<DBSCHEMA>>.[COPAYMENT]  WHERE  SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<DBSCHEMA>>.[SCHEDULE  WHERE  EFFECTIVE\_DATE = ' <<effective-date>> '  AND PUBLICATION\_STATUS = 'PUBLISHED' ) |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample Patient Contribution Fees

Note: Field values are hard-coded here and are subject to change based on fee value changes.



## Steps to download and import Container Fees (fees.txt) and store results

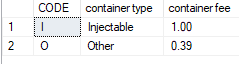
|  |  |
| --- | --- |
| **Testing results** | Status :-**In Progress**  **Current text file will be available until data is available.** |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /containers | Retrieve information about containers. | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Caution Data** | **Specific date**  SELECT  'I' AS CODE ,  'Injectable' AS 'container type',  AVERAGE\_ROUNDED\_UNIT\_PRICE AS 'container fee'  FROM  <<DBSCHEMA>>.CONTAINER  WHERE  SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<DBSCHEMA>>.[SCHEDULE  WHERE  EFFECTIVE\_DATE = ' <<effective-date>> '  AND PUBLICATION\_STATUS = 'PUBLISHED' )  and CONTAINER\_TYPE ='injectable'  union  SELECT  'O' AS CODE ,  'Other' AS 'container type',  AVERAGE\_ROUNDED\_UNIT\_PRICE AS 'container fee'  FROM  <<DBSCHEMA>>.CONTAINER  WHERE  SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<DBSCHEMA>>.[SCHEDULE  WHERE  EFFECTIVE\_DATE = ' <<effective-date>> '  AND PUBLICATION\_STATUS = 'PUBLISHED' )  and CONTAINER\_TYPE ='other' |
| **Sample SQL Statements to DISPENSING FEES** | SELECT  'DD' AS CODE ,  'Dangerous Drug' AS 'fee description',  DISPENSING\_FEE\_DANGEROUS\_DRUG AS 'dispensing fee' ,  '0.00' AS 'additional fee'  FROM  <<dbschema>>.FEE  WHERE  SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.SCHEDULE  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' )  and PROGRAM\_CODE ='IF'  union  SELECT  'EP' AS CODE ,  'Extemp Prepared' AS 'fee description',  '10.41' AS 'dispensing fee' ,  '1.80' AS 'additional fee'  FROM  <<dbschema>>.FEE  WHERE  SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.SCHEDULE  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' )  and PROGRAM\_CODE ='IF'  union  SELECT  'EW' AS CODE ,  'Water added' AS 'fee description',  DISPENSING\_FEE\_WATER\_ADDED AS 'dispensing fee' ,  '1.80' AS 'additional fee'  FROM  <<dbschema>>.FEE  WHERE  SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.SCHEDULE  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' )  and PROGRAM\_CODE ='IF'  union  SELECT  'NF' AS CODE ,  'No Fee' AS 'fee description',  '0.00' AS 'dispensing fee' ,  '0.00' AS 'additional fee'  FROM  <<dbschema>>.FEE  WHERE  SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.SCHEDULE  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' )  and PROGRAM\_CODE ='IF'  union  SELECT  'RP' AS CODE ,  'Ready Prepared' AS 'fee description',  DISPENSING\_FEE\_READY\_PREPARED AS 'dispensing fee' ,  SAFETY\_NET\_RECORDING\_FEE\_RP AS 'additional fee'  FROM  <<dbschema>>.FEE  WHERE  SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.SCHEDULE  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' )  and PROGRAM\_CODE ='IF' |
| **MARKUP** | with main\_a as (  SELECT DISTINCT  LIMIT,  CASE  WHEN FIXED = 0 THEN VARIABLE  WHEN VARIABLE = 0 THEN FIXED  END AS MARKUP,  MARKUP\_BAND\_CODE,  (  SELECT STRING\_AGG([PROGRAM\_CODE], ',') WITHIN GROUP (ORDER BY [PROGRAM\_CODE])  FROM (  SELECT DISTINCT [PROGRAM\_CODE]  FROM<<DB-SCHEMA>>.[MARKUP\_BAND\_V\_3]  WHERE  [DISPENSING\_RULE\_MNEM] like '%s94-public-s100%'  AND SCHEDULE\_CODE = (SELECT SCHEDULE\_CODE FROM PRESENT\_PBS\_SCHDL\_EMBARGO.SCHEDULE\_V\_3 WHERE EFFECTIVE\_DATE = '<<effective-date>>' AND PUBLICATION\_STATUS = 'PUBLISHED')  AND MARKUP\_BAND\_CODE = 'A'  ) AS distinct\_programs  ) AS PROGRAM\_CODES  FROM  <<DB-SCHEMA>>.[MARKUP\_BAND\_V\_3]  WHERE  [DISPENSING\_RULE\_MNEM] LIKE '%s94-public-s100%'  AND SCHEDULE\_CODE = (SELECT SCHEDULE\_CODE FROM PRESENT\_PBS\_SCHDL\_EMBARGO.SCHEDULE\_V\_3 WHERE EFFECTIVE\_DATE = '<<effective-date>>' AND PUBLICATION\_STATUS = 'PUBLISHED')  AND MARKUP\_BAND\_CODE = 'A'  ) ,  main\_d as (  SELECT DISTINCT  LIMIT,  CASE  WHEN FIXED = 0 THEN VARIABLE  WHEN VARIABLE = 0 THEN FIXED  END AS MARKUP,  MARKUP\_BAND\_CODE,  (  SELECT STRING\_AGG([PROGRAM\_CODE], ',') WITHIN GROUP (ORDER BY [PROGRAM\_CODE])  FROM (  SELECT DISTINCT [PROGRAM\_CODE]  FROM<<DB-SCHEMA>>.[MARKUP\_BAND\_V\_3]  WHERE  [DISPENSING\_RULE\_MNEM] like '%s94-private-s100%'  AND SCHEDULE\_CODE = (SELECT SCHEDULE\_CODE FROM PRESENT\_PBS\_SCHDL\_EMBARGO.SCHEDULE\_V\_3 WHERE EFFECTIVE\_DATE = '<<effective-date>>' AND PUBLICATION\_STATUS = 'PUBLISHED')  AND MARKUP\_BAND\_CODE = 'D'  ) AS distinct\_programs  ) AS PROGRAM\_CODES  FROM  <<DB-SCHEMA>>.[MARKUP\_BAND\_V\_3]  WHERE  [DISPENSING\_RULE\_MNEM] LIKE '%s94-private-s100%'  AND SCHEDULE\_CODE = (SELECT SCHEDULE\_CODE FROM PRESENT\_PBS\_SCHDL\_EMBARGO.SCHEDULE\_V\_3 WHERE EFFECTIVE\_DATE = '<<effective-date>>' AND PUBLICATION\_STATUS = 'PUBLISHED')  AND MARKUP\_BAND\_CODE = 'D'  ), main\_c as (  SELECT DISTINCT  LIMIT,  CASE  WHEN FIXED > VARIABLE THEN FIXED  WHEN VARIABLE > FIXED THEN VARIABLE  END AS MARKUP,  MARKUP\_BAND\_CODE,  (  SELECT STRING\_AGG([PROGRAM\_CODE], ',') WITHIN GROUP (ORDER BY [PROGRAM\_CODE])  FROM (  SELECT DISTINCT [PROGRAM\_CODE]  FROM<<DB-SCHEMA>>.[MARKUP\_BAND\_V\_3]  WHERE  [DISPENSING\_RULE\_MNEM] like '%s90-cp%'  AND SCHEDULE\_CODE = (SELECT SCHEDULE\_CODE FROM PRESENT\_PBS\_SCHDL\_EMBARGO.SCHEDULE\_V\_3 WHERE EFFECTIVE\_DATE = '<<effective-date>>' AND PUBLICATION\_STATUS = 'PUBLISHED')  AND MARKUP\_BAND\_CODE = 'C'  ) AS distinct\_programs  ) AS PROGRAM\_CODES  FROM  <<DB-SCHEMA>>.[MARKUP\_BAND\_V\_3]  WHERE  [DISPENSING\_RULE\_MNEM] LIKE '%s90-cp%'  AND SCHEDULE\_CODE = (SELECT SCHEDULE\_CODE FROM PRESENT\_PBS\_SCHDL\_EMBARGO.SCHEDULE\_V\_3 WHERE EFFECTIVE\_DATE = '<<effective-date>>' AND PUBLICATION\_STATUS = 'PUBLISHED')  AND MARKUP\_BAND\_CODE = 'C'  ), result as (  select  concat(LIMIT , ' to ' , COALESCE(LEAD(LIMIT - 0.01) OVER (ORDER BY LIMIT), '9999.99')) as RANGE,  LIMIT,  MARKUP,  MARKUP\_BAND\_CODE,  PROGRAM\_CODES  from main\_a  union  select  concat(LIMIT , ' to ' , COALESCE(LEAD(LIMIT - 0.01) OVER (ORDER BY LIMIT), '9999.99')) as RANGE,  LIMIT,  MARKUP,  MARKUP\_BAND\_CODE,  PROGRAM\_CODES  from main\_c  union  select  concat(LIMIT , ' to ' , COALESCE(LEAD(LIMIT - 0.01) OVER (ORDER BY LIMIT), '9999.99')) as RANGE,  LIMIT,  MARKUP,  MARKUP\_BAND\_CODE,  PROGRAM\_CODES  from main\_d )  select MARKUP\_BAND\_CODE AS 'code',  PROGRAM\_CODES AS 'drug types',  RANGE,  MARKUP  from result ORDER BY MARKUP\_BAND\_CODE, LIMIT,PROGRAM\_CODES |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

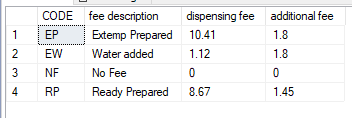
## Sample Patient Contribution Fees

Note: Field values are hard-coded here and are subject to change based on fee value changes.



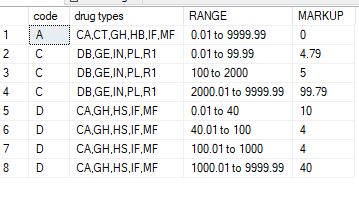
## Sample Dispensing Fees

Note: Field values are derived here and are subject to change based on fee value changes.



## Sample Markup Fees

Note: Field values are derived here and are subject to change based on fee value changes.



# Manufacturer data

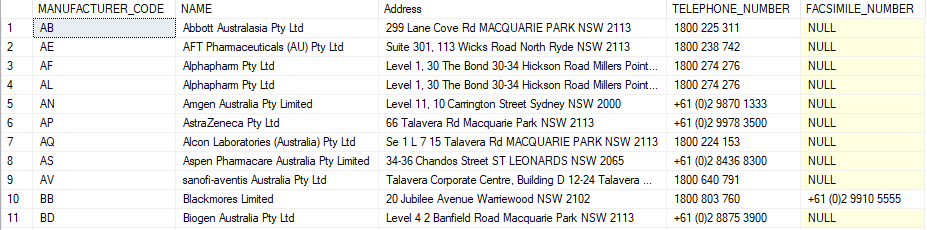
## Steps to download and import mnfr.txt and store results

|  |  |
| --- | --- |
| **Testing results** | Text files and API data extracted show a difference in mnfr-telephone column. The API data extracted includes the country code This is the same as the PBS data. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /organisations | Retrieve information about Organisations | | /items | Retrieve information about Items | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Manufacturer Data** | **Specific date**  SELECT DISTINCT  [Items].MANUFACTURER\_CODE ,  [Organisations].NAME ,  [Organisations].[STREET\_ADDRESS] + ' ' + [Organisations].[CITY] + ' ' + [Organisations].[STATE] + ' ' + [Organisations].[POSTCODE] AS Address,  CASE  WHEN  [Organisations].TELEPHONE\_NUMBER LIKE '+61%'  THEN  REPLACE([Organisations].TELEPHONE\_NUMBER, '+61 (0)', '0')  ELSE  [Organisations].TELEPHONE\_NUMBER  END AS TELEPHONE\_NUMBER,  CASE  WHEN  [Organisations].FACSIMILE\_NUMBER LIKE '+61%'  THEN  REPLACE([Organisations].FACSIMILE\_NUMBER, '+61 (0)', '0')  ELSE  [Organisations].FACSIMILE\_NUMBER  END AS FACSIMILE\_NUMBER  FROM  <<effective-date>>.[ITEM\_V\_3] as [Items]  INNER JOIN  <<effective-date>>.ORGANISATION\_V\_3 as [Organisations]  ON  [Items].ORGANISATION\_ID = [Organisations].ORGANISATION\_ID  AND [Items].SCHEDULE\_CODE = [Organisations].SCHEDULE\_CODE  AND [Items].SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<effective-date>>.SCHEDULE\_V\_3  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED')  ORDER BY  [Items].MANUFACTURER\_CODE |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample Manufacturer Text data



# Medication Chart

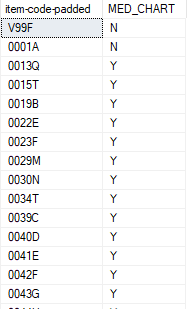
## Steps to download and import Medication Chart Electronic data (med-chart-electronic.txt) and store results

|  |  |
| --- | --- |
| **Testing Result** | Text files and API data extracted are matching with no issues found. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    * This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    * To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /extemporaneous-ingredient | Retrieve information about extemporaneous ingredient | | /items | Retrieve information about items | | /extemporaneous-tariff | Retrieve information about extemporaneous tariff | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Medication Chart Electronic Data** | **Specific date**  select  \*  from  (  SELECT distinct  ISNULL(REPLICATE('0',5-LEN(PBS\_CODE)) + PBS\_CODE, PBS\_CODE) AS "item-code-padded",  IIf(ELECT\_MED\_CHART\_ELIGIBLE\_IND='Y','N','Y') AS MED\_CHART  FROM  <<DBSCHEMA>>.[ITEM] i  WHERE  SUPPLY\_ONLY\_INDICATOR='N'  AND SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  SCHEDULE  WHERE  EFFECTIVE\_DATE = ' <<effective-date>> '  AND PUBLICATION\_STATUS = 'PUBLISHED' )  and pbs\_code not in  (  select  dt.pbs\_code  from  <<DBSCHEMA>>.[EXTEMPORANEOUS\_DRUG\_TARIFF] dt  where  dt.SCHEDULE\_CODE = i.SCHEDULE\_CODE    union all    select  ei.pbs\_code  from  <<DBSCHEMA>>.[EXTEMPORANEOUS\_INGREDIENT] ei  where  ei.SCHEDULE\_CODE = i.SCHEDULE\_CODE )    UNION ALL    SELECT  \*  from  (VALUES ('V99F', 'N'), ('0001A', 'N')) as t("item-code-padded","cont-disp") ) t  ORDER BY  (  CASE  WHEN  "item-code-padded" like '[a-z]%'  THEN  0  ELSE  1  END),  "item-code-padded" asc |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample Medication Chart Electronic



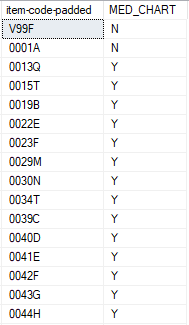
## Steps to download and import Medication Chart Paper data (med-chart-paper.txt) and store results

|  |  |
| --- | --- |
| **Testing Result** | Text files and API data extracted are matching with no issues found. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    * This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    * To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /extemporaneous-ingredient | Retrieve information about extemporaneous ingredient | | /items | Retrieve information about items | | /extemporaneous-tariff | Retrieve information about extemporaneous tariff | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Medication Chart Paper Data** | **Specific date**  select  \*  from  (  SELECT distinct  ISNULL(REPLICATE('0',5-LEN(PBS\_CODE)) + PBS\_CODE, PBS\_CODE) AS "item-code-padded",  IIf([PAPER\_MED\_CHART\_ELIGIBLE\_IND]='Y','N','Y') AS MED\_CHART  FROM  <<DBSCHEMA>>.[ITEM] i  WHERE  SUPPLY\_ONLY\_INDICATOR='N'  AND SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  SCHEDULE  WHERE  EFFECTIVE\_DATE = ' <<effective-date>> '  AND PUBLICATION\_STATUS = 'PUBLISHED' )  and pbs\_code not in  (  select  dt.pbs\_code  from  <<DBSCHEMA>>.[EXTEMPORANEOUS\_DRUG\_TARIFF] dt  where  dt.SCHEDULE\_CODE = i.SCHEDULE\_CODE    union all    select  ei.pbs\_code  from  <<DBSCHEMA>>.[EXTEMPORANEOUS\_INGREDIENT] ei  where  ei.SCHEDULE\_CODE = i.SCHEDULE\_CODE )  UNION ALL  SELECT  \*  from  (VALUES ('V99F', 'N'), ('0001A', 'N')) as t("item-code-padded","cont-disp") ) t  ORDER BY  (  CASE  WHEN  "item-code-padded" like '[a-z]%'  THEN  0  ELSE  1  END),  "item-code-padded" asc |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample Medication Chart Electronic



# PBS Item Table

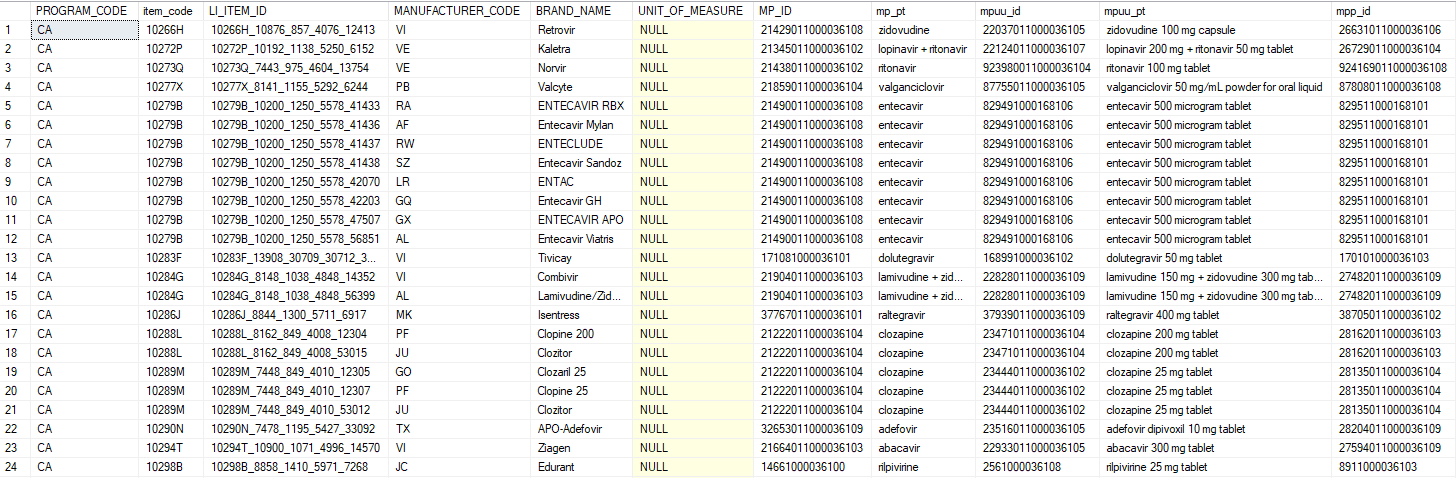
## Steps to download and import Pharcacy\_PBS\_Item\_Table.txt and store results

|  |  |
| --- | --- |
| **Testing results** | Text files and API data extracted are matching with no issues found.  Difference in count is those records from Pharmacy PBS Item Table Txt file which contains end-date. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    1. This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    1. To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /items | Retrieve information about items | | /item-prescribing-text-relationships | Retrieve information about items prescribing text relationship | | /prescribing-texts | Retrieve information about prescribing texts | | /item-restriction-relationships | Retrieve information about item restriction relationship | | /restriction-prescribing-text-relationships | Retrieve information about restriction prescribing text relationship | | /restrictions | Retrieve information about restrictions | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract PBS Items** | **Specific date**  with main as  (  SELECT distinct  [Items].PBS\_CODE AS PBS\_CODE ,  items.BENEFIT\_TYPE\_CODE as BENEFIT\_TYPE\_CODE,  pt.PRESCRIBING\_TXT\_ID ,  pt.PRESCRIBING\_TYPE ,  items.SCHEDULE\_CODE  FROM  <<dbschema>>.[ITEM\_V\_3] AS [Items]  left JOIN  <<dbschema>>.ITEM\_PRESCRIBING\_TXT\_RLTD\_V\_3 AS ipr  ON  [Items].PBS\_CODE = ipr.PBS\_CODE  and [Items].SCHEDULE\_CODE = ipr.SCHEDULE\_CODE  left join  <<dbschema>>.[PRESCRIBING\_TXT\_V\_3] pt  on  pt.SCHEDULE\_CODE = items.SCHEDULE\_CODE  and pt.PRESCRIBING\_TXT\_ID = ipr.PRESCRIBING\_TXT\_ID  where  items.program\_code != 'EP'  and items.SCHEDULE\_CODE = =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.[SCHEDULE\_V\_3]  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' )  and items.SUPPLY\_ONLY\_INDICATOR = 'N'  union all  SELECT distinct  [Items].PBS\_CODE AS PBS\_CODE ,  items.BENEFIT\_TYPE\_CODE as BENEFIT\_TYPE\_CODE,  pt.PRESCRIBING\_TXT\_ID ,  pt.PRESCRIBING\_TYPE ,  items.SCHEDULE\_CODE  FROM  <<dbschema>>.[ITEM\_V\_3] AS [Items]  left join  <<dbschema>>.[ITEM\_RESTRICTION\_RLTD\_V\_3] irr  ON  irr.schedule\_code = items.schedule\_code  and irr.PBS\_CODE = items.PBS\_CODE  left join  <<dbschema>>.[RSTRCTN\_PRSCRBNG\_TXT\_RLTD\_V\_3] rptr  on  rptr.SCHEDULE\_CODE = items.SCHEDULE\_CODE  and rptr.RES\_CODE = irr.RES\_CODE  left join  <<dbschema>>.[PRESCRIBING\_TXT\_V\_3] pt  on  pt.SCHEDULE\_CODE = items.SCHEDULE\_CODE  and pt.PRESCRIBING\_TXT\_ID = rptr.PRESCRIBING\_TEXT\_ID  where  pt.PRESCRIBING\_TYPE in ('ADMINISTRATIVE\_ADVICE',  'CAUTION')  and items.program\_code != 'EP'  and items.SUPPLY\_ONLY\_INDICATOR = 'N'  and pt.SCHEDULE\_CODE = =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.[SCHEDULE\_V\_3]  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' ) ) ,  m\_agg as  (  select  pbs\_code ,  CASE  WHEN BENEFIT\_TYPE\_CODE='S' THEN 'A'  ELSE  BENEFIT\_TYPE\_CODE  END AS pbs\_code,  SCHEDULE\_CODE ,  PRESCRIBING\_TYPE ,  STRING\_AGG(PRESCRIBING\_TXT\_ID, ', ') as PRESCRIBING\_TXT\_ID  from  main  group by  PBS\_CODE ,  BENEFIT\_TYPE\_CODE,  SCHEDULE\_CODE ,  PRESCRIBING\_TYPE )  select  pbs\_code ,  BENEFIT\_TYPE\_CODE ,  SCHEDULE\_CODE ,  ADMINISTRATIVE\_ADVICE,  CAUTION  from  m\_agg pivot ( max(PRESCRIBING\_TXT\_ID)  for PRESCRIBING\_TYPE in (ADMINISTRATIVE\_ADVICE, CAUTION) ) piv |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample PBS Item Table



# Prescriber Type

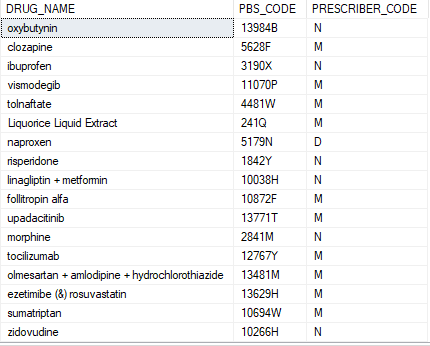
## Steps to download and import Prescriber\_type.txt and store results

|  |  |
| --- | --- |
| **Testing results** | Text files and API data extracted show a difference of drug name-text in a few cases.    This is the same as the API PBS data. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code)**:**    * This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    * To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /items | Retrieve information about Items | | /prescribers | Retrieve information about Prescriber | | /extemporaneous-preparations | Retrieve information about Extemporaneous Preparations | | /standard-formula-preparations | Retrieve information about formula preparations | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Extemporaneous Prescriber Type Data** | **Specific date**  WITH MAIN AS  (  SELECT DISTINCT  [ITEMS].PBS\_CODE ,  [ITEMS].DRUG\_NAME ,  PRESCRIBER.PRESCRIBER\_CODE,  [ITEMS].SCHEDULE\_CODE  FROM  <<dbschema>>.[ITEM\_V\_3] AS ITEMS  INNER JOIN  <<dbschema>>.[PRESCRIBER\_V\_3] AS PRESCRIBER  ON  PRESCRIBER.PBS\_CODE = ITEMS.PBS\_CODE  AND [ITEMS].SCHEDULE\_CODE = [ITEMS].SCHEDULE\_CODE  WHERE  [ITEMS].SUPPLY\_ONLY\_INDICATOR = 'N'  AND [ITEMS].SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.[SCHEDULE\_V\_3]  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' ) )  SELECT  M.PBS\_CODE,  CASE  WHEN  EP.PREPARATION IS NOT NULL  THEN  CONCAT(COALESCE(EP.PREPARATION, ''), ', ', COALESCE(EP.MAXIMUM\_QUANTITY, ''), COALESCE(EP.[MAXIMUM\_QUANTITY\_UNIT], ''))  WHEN  SP.SFP\_DRUG\_NAME IS NOT NULL  THEN  CONCAT(COALESCE(SP.SFP\_DRUG\_NAME, ''), ', ', COALESCE(SP.MAXIMUM\_QUANTITY, ''), COALESCE(SP.[MAXIMUM\_QUANTITY\_UNIT], ''))  ELSE  M.DRUG\_NAME  END AS DRUG\_NAME,  M.PRESCRIBER\_CODE  FROM  MAIN M  LEFT JOIN  <<dbschema>>.[EXTEMPORANEOUS\_PREPARATION\_V\_3] EP  ON  M.PBS\_CODE = EP.PBS\_CODE  AND M.SCHEDULE\_CODE = EP.SCHEDULE\_CODE  LEFT JOIN  <<DBSCHEMA>>.[STANDARD\_FORMULA\_PREPARATION\_V\_3] SP  ON  M.PBS\_CODE = SP.PBS\_CODE  AND M.SCHEDULE\_CODE = SP.SCHEDULE\_CODE  ORDER BY  PBS\_CODE ,  DRUG\_NAME,  PRESCRIBER\_CODE |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample Prescriber Type



# Safety Net

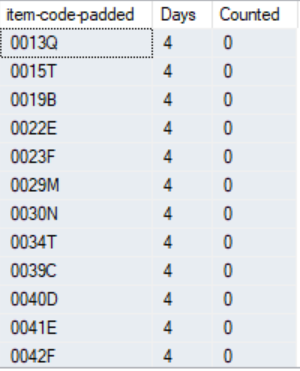
## Steps to download and import sn20dr.txt and store results

|  |  |
| --- | --- |
| **Testing results** | Text files and API data extracted are matching with no issues found. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    * This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    * To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /items | Retrieve information about Items | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Safety Net Data** | **Specific date**  SELECT  t.\*  FROM  (  SELECT DISTINCT  ISNULL(REPLICATE('0', 5 - LEN(PBS\_CODE)) + PBS\_CODE, PBS\_CODE) AS "item-code-padded",  CASE  WHEN  SAFETY\_NET\_RESUPPLY\_RULE\_DAYS IS NULL  THEN  '04'  ELSE  FORMAT(SAFETY\_NET\_RESUPPLY\_RULE\_DAYS, '00')  END AS "Days",  SAFETY\_NET\_RESUP\_RULE\_CNT\_IND AS "Counted"  FROM  <<dbschema>>.[ITEM] i  WHERE  SUPPLY\_ONLY\_INDICATOR = 'N'  AND SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.SCHEDULE  WHERE  EFFECTIVE\_DATE = ' <<effective-date>> '  AND PUBLICATION\_STATUS = 'PUBLISHED' )  AND PBS\_CODE NOT IN  (  SELECT  dt.pbs\_code  FROM  <<dbschema>>.[EXTEMPORANEOUS\_DRUG\_TARIFF] dt  WHERE  dt.SCHEDULE\_CODE = i.SCHEDULE\_CODE    UNION ALL    SELECT  ei.pbs\_code  FROM  <<dbschema>>.[EXTEMPORANEOUS\_INGREDIENT] ei  WHERE  ei.SCHEDULE\_CODE = i.SCHEDULE\_CODE )    UNION ALL    SELECT  "item-code-padded",  "Days" ,  "Counted"  FROM  (VALUES ('V99F', '04', 'N'), ('0001A', '04', 'N')) AS t("item-code-padded", "Days", "Counted")) t  order by  "item-code-padded" |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample safety Net data



# Link Extract

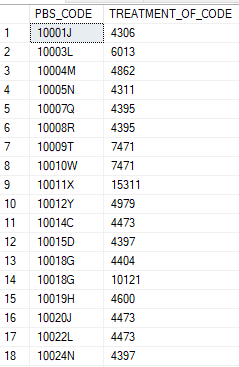
## Steps to download and import linkextract.txt and store results

|  |  |
| --- | --- |
| **Testing results** | Text files and API data extracted are matching with no issues found where data is available.  The following fields will not be displayed in the API response:   * increase-code – not recruitable. * start-date – restriction start date will be added soon. * end-date – when it stops being in the data is when it ends. |
| **Download API Endpoint Data** | 1. [**Get Schedule Code**](#_Get_Schedule_Code):    * This refers to obtaining information related to a specific schedule code. 2. [**Download API Data**](#_Steps_to_download_1):    * To download data from the API, you can use the following endpoints along with their respective operations:  |  |  | | --- | --- | | **Operations** | **Description** | | /schedules | Retrieve information about schedules | | /items | Retrieve information about Items | |
| **Load data** | Import data into table :- [Load API data into database](#_Steps_to_Load) |
| **Sample SQL Statements to Extract Safety Net Data** | **Specific date**  select distinct  ITEM.PBS\_CODE ,  REST\_TEXT.TREATMENT\_OF\_CODE,  ITEM.SUPPLY\_ONLY\_DATE  FROM  <<DBSCHEMA>>.ITEM\_V\_3 AS ITEM  left JOIN  <<DBSCHEMA>>.[ITEM\_RESTRICTION\_RLTD\_V\_3] ITEM\_REST\_RLTD  ON  ITEM.SCHEDULE\_CODE = ITEM\_REST\_RLTD.SCHEDULE\_CODE  AND ITEM.PBS\_CODE = ITEM\_REST\_RLTD.PBS\_CODE  left JOIN  <<DBSCHEMA>>.[RESTRICTION\_TEXT\_V\_3] REST\_TEXT  ON  ITEM.SCHEDULE\_CODE = REST\_TEXT.SCHEDULE\_CODE  AND ITEM\_REST\_RLTD.RES\_CODE = REST\_TEXT.RES\_CODE  WHERE  ITEM.SCHEDULE\_CODE =  (  SELECT  SCHEDULE\_CODE  FROM  <<dbschema>>.SCHEDULE  WHERE  EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' )  AND REST\_TEXT.TREATMENT\_OF\_CODE <> 9999 )  SELECT  PBS\_CODE,  TREATMENT\_OF\_CODE  FROM  MAIN  WHERE  SUPPLY\_ONLY\_DATE >= '<<effective-date>>'  OR SUPPLY\_ONLY\_DATE IS NULL  ORDER BY  PBS\_CODE |

## 2013 Text Field Descriptions

<https://data.pbs.gov.au/text-extracts.html>

## Sample Link Extract data



# Control Text

The control text (control.txt) is only for extracting text. If a publication date is needed, it can be obtained through the schedule endpoint. The control text has been removed.

# Dispensing Incentive Items (DI)

From 1 July 2020, the Premium-free Dispensing Incentive (PFDI) was discontinued.

# Appendix

## Links

|  |  |
| --- | --- |
| **API Developer Portal** | [Home - Department of Health and Aged Care API Catalogue](https://data-api-portal.health.gov.au/) |
| **Postman Download** | [Download Postman | Get Started for Free](https://www.postman.com/downloads/) |
| **Schedule Code** | <https://data-api.health.gov.au/pbs/api/v3/schedules> |
| **Caution** | [https://data-api.health.gov.au/pbs/api/v3/prescribing-texts[?schedule\_code][&limit][&page]](https://data-api.health.gov.au/pbs/api/v3/prescribing-texts%5b?schedule_code%5d%5b&limit%5d%5b&page%5d) |
| **Notes** | [https://data-api.health.gov.au/pbs/api/v3/prescribing-texts[?schedule\_code][&limit][&page]](https://data-api.health.gov.au/pbs/api/v3/prescribing-texts%5b?schedule_code%5d%5b&limit%5d%5b&page%5d) |
| **ATC** | [https://data-api.health.gov.au/pbs/api/v3/atc-codes[?schedule\_code][&limit][&page]](https://data-api.health.gov.au/pbs/api/v3/atc-codes%5b?schedule_code%5d%5b&limit%5d%5b&page%5d) |
| **Items** | [https://data-api.health.gov.au/pbs/api/v3/items[?schedule\_code]](https://data-api.health.gov.au/pbs/api/v3/items%5b?schedule_code%5d)  [[&pbs\_code][&program\_code][&manufacturer\_code][&limit][&page]](https://data-api.health.gov.au/pbs/api/v3/items%5b?schedule_code%5d) |
| **Extemporaneous Preparation** | [https://data-api.health.gov.au/pbs/api/v3/extemporaneous-preparations[?schedule\_code][&pbs\_code][&limit][&page]](https://data-api.health.gov.au/pbs/api/v3/extemporaneous-preparations%5b?schedule_code%5d%5b&pbs_code%5d%5b&limit%5d%5b&page%5d) |
| **Public** | [https://data-api-portal.health.gov.au/api-details#api=pbs-prod-api-public-v3](https://data-api-portal.health.gov.au/api-details%23api=pbs-prod-api-public-v3) |

## Get Schedule Code

|  |  |
| --- | --- |
| **Download API Endpoint Data** | 1. **Open Postman:**   * + If you haven’t already, download and install Postman on your machine.   + Launch Postman.   2. **Create a New Request:**   * Click the “+” icon in the workbench to open a new tab. * In the request URL field, enter the following endpoint:   <https://data-api.health.gov.au/pbs/api/v3/schedules>  3. **Set Headers:**   * + Add the following headers to your request:   + Subscription-Key: <use your given subscription key>   4. **Send the Request:**   * + Click the “Send” button to execute the request.   + Postman will display the response from the API.   + Collect Schedule Code from the Response.   + Or save this into DB |



Sample Schedule Code

|  |  |
| --- | --- |
| **For Download Schedule data and Save into Schedule table.** [**Refer**](#_Invoke_API_and) | CREATE TABLE <<dbschema>>.[SCHEDULE](  [SCHEDULE\_CODE] [FLOAT],  [EFFECTIVE\_DATE] [NVARCHAR](10),  [EFFECTIVE\_MONTH] [NVARCHAR](9),  [EFFECTIVE\_YEAR] [INT],  [EMBARGO\_INDICATOR] [NVARCHAR](1),  [LATEST\_SCHEDULE\_INDICATOR] [NVARCHAR](1),  [PUBLICATION\_STATUS] [NVARCHAR](20),  [REVISION\_NUMBER] [FLOAT],  [START\_TSP] [DATETIME2](0)  ) |
| **To Get Latest Schedule Code** | SELECT top(1) SCHEDULE\_CODE  FROM SCHEDULE  ORDER BY EFFECTIVE\_DATE DESC,  REVISION\_NUMBER DESC |
| **To Get Specific Schedule Code** | SELECT SCHEDULE\_CODE  FROM SCHEDULE  WHERE EFFECTIVE\_DATE = '<<effective-date>>'  AND PUBLICATION\_STATUS = 'PUBLISHED' |

## Verhoeff Checksum Algorithm

To generate a Verhoeff checksum in SQL Server, you can create a User-Defined Function (UDF). The Verhoeff algorithm involves several steps, including digit manipulation using lookup tables for multiplication, permutation, and inversion.

**Reference Link** :- <https://confluence.ihtsdotools.org/display/DOCRELFMT/6.4.2+Check-digit+Computation>

**Used in AMT Query**

CREATE FUNCTION dbo.VerhoeffChecksum (@number NVARCHAR(MAX))

RETURNS CHAR(1)

AS

BEGIN

DECLARE @d TABLE (d0 INT, d1 INT, d2 INT, d3 INT, d4 INT, d5 INT, d6 INT, d7 INT, d8 INT, d9 INT);

DECLARE @p TABLE (p0 INT, p1 INT, p2 INT, p3 INT, p4 INT, p5 INT, p6 INT, p7 INT, p8 INT, p9 INT);

DECLARE @inv TABLE (inv0 INT);

-- Populate the multiplication table

INSERT INTO @d VALUES (0, 1, 2, 3, 4, 5, 6, 7, 8, 9);

INSERT INTO @d VALUES (1, 2, 3, 4, 0, 6, 7, 8, 9, 5);

INSERT INTO @d VALUES (2, 3, 4, 0, 1, 7, 8, 9, 5, 6);

INSERT INTO @d VALUES (3, 4, 0, 1, 2, 8, 9, 5, 6, 7);

INSERT INTO @d VALUES (4, 0, 1, 2, 3, 9, 5, 6, 7, 8);

INSERT INTO @d VALUES (5, 9, 8, 7, 6, 0, 4, 3, 2, 1);

INSERT INTO @d VALUES (6, 5, 9, 8, 7, 1, 0, 4, 3, 2);

INSERT INTO @d VALUES (7, 6, 5, 9, 8, 2, 1, 0, 4, 3);

INSERT INTO @d VALUES (8, 7, 6, 5, 9, 3, 2, 1, 0, 4);

INSERT INTO @d VALUES (9, 8, 7, 6, 5, 4, 3, 2, 1, 0);

-- Populate the permutation table

INSERT INTO @p VALUES (0, 1, 2, 3, 4, 5, 6, 7, 8, 9);

INSERT INTO @p VALUES (1, 5, 7, 6, 2, 8, 3, 0, 9, 4);

INSERT INTO @p VALUES (5, 8, 0, 3, 7, 9, 6, 1, 4, 2);

INSERT INTO @p VALUES (8, 9, 1, 6, 0, 4, 3, 5, 2, 7);

INSERT INTO @p VALUES (9, 4, 5, 3, 1, 2, 6, 8, 7, 0);

INSERT INTO @p VALUES (4, 2, 8, 6, 5, 7, 3, 9, 0, 1);

INSERT INTO @p VALUES (2, 7, 9, 3, 8, 0, 6, 4, 1, 5);

INSERT INTO @p VALUES (7, 0, 4, 6, 9, 1, 3, 2, 5, 8);

-- Populate the inverse table

INSERT INTO @inv VALUES (0);

INSERT INTO @inv VALUES (4);

INSERT INTO @inv VALUES (3);

INSERT INTO @inv VALUES (2);

INSERT INTO @inv VALUES (1);

INSERT INTO @inv VALUES (5);

INSERT INTO @inv VALUES (6);

INSERT INTO @inv VALUES (7);

INSERT INTO @inv VALUES (8);

INSERT INTO @inv VALUES (9);

DECLARE @c INT = 0;

DECLARE @i INT = 0;

DECLARE @len INT = LEN(@number);

DECLARE @digit INT;

WHILE @i < @len

BEGIN

SET @digit = CONVERT(INT, SUBSTRING(@number, @len - @i, 1));

SELECT @c = d.d0

FROM @d d

JOIN @p p ON p.p0 = @c

WHERE p.p1 = @i % 8 AND d.d1 = @digit;

SET @i = @i + 1;

END

SELECT @c = inv.inv0

FROM @inv inv

WHERE inv.inv0 = @c;

RETURN CONVERT(CHAR(1), @c);

END;

**Sample to invoke above User Defined Function.**

DECLARE @x BIGINT;

-- Calculate the sum and store it in the variable

SET @x = 57213 + 100014410;

-- Get the Verhoeff checksum for the calculated sum

SELECT dbo.VerhoeffChecksum(CAST(@x AS NVARCHAR(MAX))) AS Checksum;

**Note**: We provide these IDs instead of SNOMED information because these concepts are not recognized as official SNOMED/AMT concepts. They are only PBS concepts and have no relationship to AMT. Also, from the item table these fields can be used to derive the preferred term/text part of the codes when the amt concept part is null:

mp\_id can be derived from Item Drug name.

mpp\_id can be derived from Item Schedule Form.

tpp\_id can be derived from Item Brand name.