



the alpha **school** system

CONFERENCE 2019

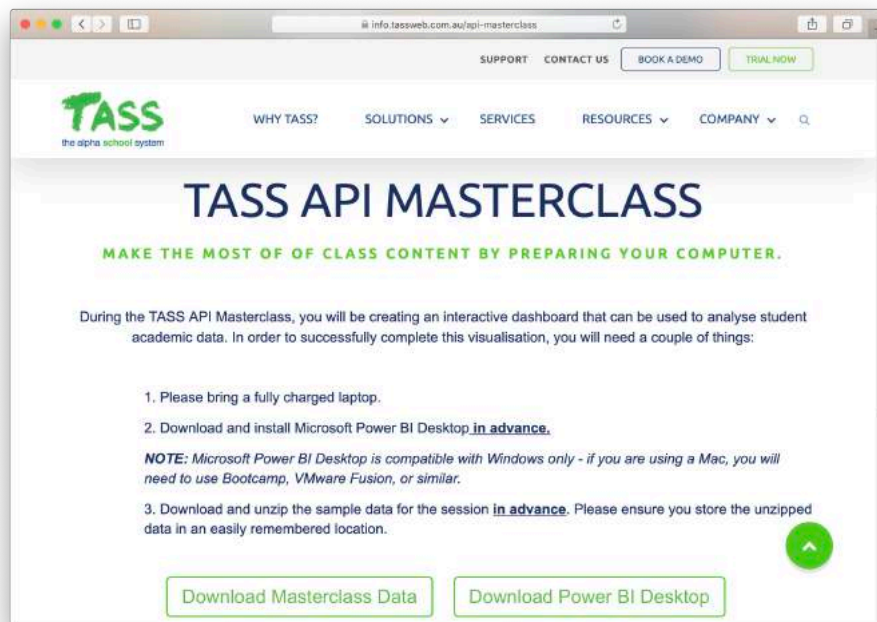
API Masterclass



Getting Started

In this session, we will develop an interactive dashboard in Power BI that will let us connect to multiple TASS APIs and analyse student overall grades.

Visit our masterclass landing page, which will have all of the links and resources you'll need to create your dashboard: <https://info.tassweb.com.au/api-masterclass>



Before proceeding, ensure that you have:

- Downloaded and installed Microsoft Power BI Desktop (Windows only, Mac users will need to use Bootcamp, VMware Fusion, etc).
- Downloaded and unzipped the sample data for the session.

Data Sources

In today's session, we will be connecting to two TASS APIs: Student Details and Student Analytics. For the purposes of the demonstration, we won't work with live APIs, but we have provided two JSON files that contain the API output from a real TASS system.

Connecting to TASS Using APIs

This section details the process to connect to a live TASS system using APIs, as demonstrated in the API Masterclass. For more details, refer to the API Introduction repository on our GitHub site.

TASS APIs are RESTful web servers and exchange data in JSON format (which is human-readable) securely over HTTPS.

Select your TASS API

TASS offers a wide range of APIs that you can use to exchange data with external systems. Refer to page 20 to see the currently available APIs, or visit the TASS GitHub Site:

<https://www.github.com/TheAlphaSchoolSystemPtyLtd>

Please note that the TASS APIs are licenced, so contact our sales team for more details.

Configure API Connection in TASS.web

Each connection must be registered in TASS.web. If this does not occur, the API connection will fail.

Tip: It is best practice to create an individual registration for every script or external system. This allows granular control for everything that connects to TASS via API.

To do this, log in to TASS.web and go to **System Admin > Utilities > API Gateway Maintenance**, then click on the TASS APIs tab, and then click on Add API Application.

TASS API Application Details			
Application Code	<input type="text" value="SA01"/>	Last Update By	<input type="text" value="sam"/> On <input type="text" value="02/05/2019"/>
Application Description	<input type="text" value="TASS API Demo"/>		
* Licence Code	<input type="text" value="API17"/>	<input type="text" value="Student Academic Analytics"/>	
Application Server	<input type="text"/>		
Token Key	<input type="text" value="tcOFvhpRTTqXlnGagsd8Dg=="/>	<input type="button" value="Generate"/>	
HTTPS Only	<input type="text" value="Yes"/>		
Enable Photos	<input type="text" value="Yes"/>		
Acknowledgement			
By enabling this API, you are acknowledging that data from the school's TASS database will be transferred to a third-party software system. Refer to the Online Help for the details of that data or contact the TASS Helpdesk.			
Would you like to enable this API?		<input checked="" type="checkbox"/>	
<input type="button" value="Cancel"/>			

The below table explains each field:

Field	Notes
Application Code	Used to uniquely identify the API connection to TASS. Note this down for later.
Description	Can be anything you like, as long as it makes sense to those administering TASS at your school.
Licence Code	Select the API you wish to use. Note: If you don't have the correct licensing for your selected API, you'll be able to complete the setup steps, but you'll see a warning triangle when you save your connection.
Application Server	Optionally used to whitelist servers that can call this API connection for increased security.
Token Key	Click Generate to generate the token encryption key that will be used to encrypt part of your API call. Note this down for later.
HTTPS Only	Will always be enabled as TASS only runs in HTTPS mode from v49.
Enable Photos	Some APIs (eg Student Details, Employee Details) can be configured to return photos, which must be enabled here.
Acknowledgement	This must be enabled to allow the API to function. You can untick this to disable the API connection if this is desired.

When you're done, click **Save**.

Note that these details can be retrieved at any time by going into the **API Gateway Maintenance** program.

Select a Method

Each TASS API offers a range of methods that allow you exchange specific sets of data with TASS, and each API call must make use of a single method. For the methods that return data from TASS, you can consider them to be like reports in TASS.web. You can find these in the GitHub repositories for each API, along with details about what data they accept and return, and how to make use of them.

As an example, the Student Analytics API offers a range of methods which allows you to return various sets of data, which include:

- Student absence records (getStudentAbsences)
- Standardised testing results (getStudentTestingData)
- Weighted academic results (getStudentWeightedResults)

Determine Parameters

Details about the parameters that are required for each API method will be defined against each method in GitHub. For the methods that return data from TASS, if we continue with the TASS.web report metaphor, you could consider them to be the report filters that allow you to decide what data to include or exclude.

The parameters are encoded in JSON format, and the pages for each method in GitHub detail all of the parameters along with the values that can be provided. Note that some are mandatory, and the API call will fail if they are not provided.

For example, if we were to use the getStudentAbsences method in the Student Analytics API, we could specify parameters to return all students in Year 9 using the mandatory **studcode** parameter and the optional **yeargrp** parameter as follows:

```
{  
  "studcode": "all",  
  "yeargrp": "9"  
}
```

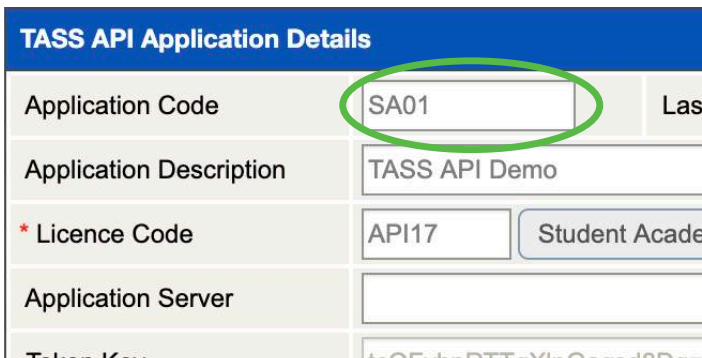
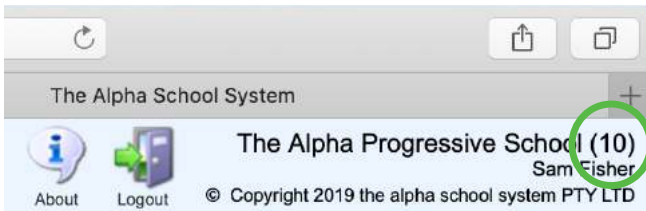
If you are unsure, each method's page on GitHub contains a set of sample parameters that you can use as a starting point.

Generate API Call

Details can now be gathered together to construct the API call. As the TASS APIs are RESTful, the calls are in the form of URLs, which we can construct using the following template:

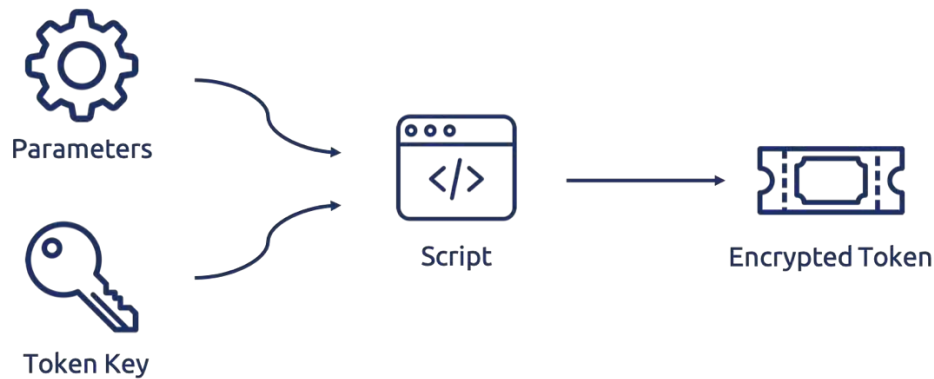
**https://{domain}/tassweb/api/?method={method}&appcode={appcode}
&company={companycode}&v=2&token={token}**

Let's take a look at each of the highlighted sections of the URL:

Field	Description	Example
{domain}	The domain for your school's TASS instance. If you aren't sure, log in to TASS and take a look in the address bar. The https:// at the start and any of the proceeding directories and files (eg /tassweb/index.cfm) are not required.	tass.yourschool.edu.au
{method}	The name of the method you selected earlier. Refer to the documentation in GitHub if you are unsure.	getStudentAbsences
{appcode}	The Application Code that you specified when you configure the API connection in TASS.web. 	SA01
{companycode}	The company code for company in TASS you want the API to interact with. In TASS, companies allow you have multiple schools or business entities as part of a single TASS system. If you aren't sure, you can find the company you are currently working in at the top right of TASS.web. 	01
{token}	The parameters, encrypted with the token key generated in TASS.web, using a script. See below for more details on this.	aaiAlvHfFfUmleg2RLythG Wcfui%2BScfZ2tQLWDrMEf 1T8BHc481ytZzQHZMgAIDC

Token Encryption

The token is generated by encrypting the parameters using the token key. The resulting encrypted token is included in your API call. The below diagram explains the process:



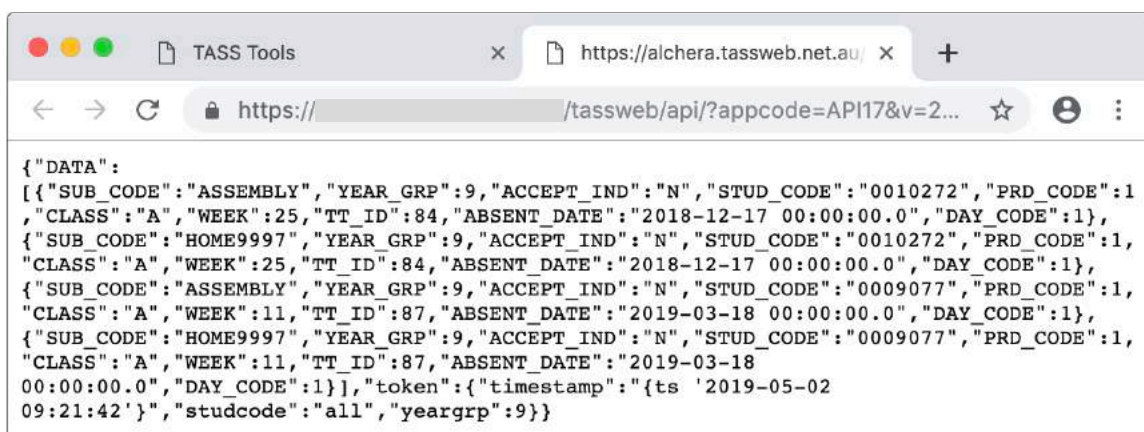
This process does require some scripting experience. There are a range of sample scripts in languages such as PowerShell, PHP, and Python that will allow you to generate your encrypted token, as well as technical details about how the encryption works if you wish to write your own script.

Note: You cannot just use the token key that you generated in TASS.web. This will not work and the single most common issue we see when customers report issues with APIs.

Test API Call

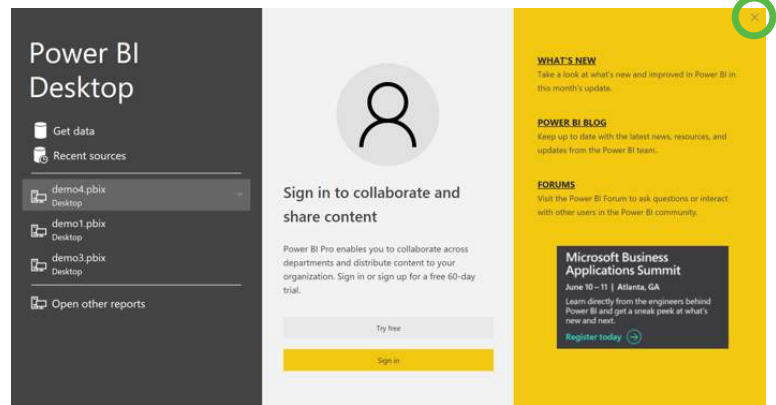
Once you have constructed your API call, test it to ensure it works as expected. As the API calls are URLs, you can easily test by pasting the URL into a web browser. The API response (in the form of JSON) will be returned. Review the data to ensure it is as expected. Should there be an issue with your API call, a JSON-encoded error message will be returned

You now have an API call that you can use with your external systems and scripts. When establishing your connection with TASS, follow the “Live TASS API” section where specified.



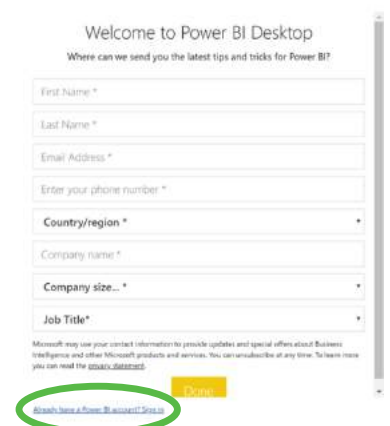
Step 1: Opening Power BI and Creating a Dashboard

When you launch Power BI, a new dashboard is created automatically. Just clear the splash screen by clicking the cross at the top right, and you are good to go.

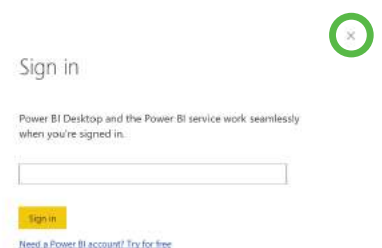


Tip: Power BI will try to make you create an account when you open it, and you'll see there's no close button. Here is a workaround:

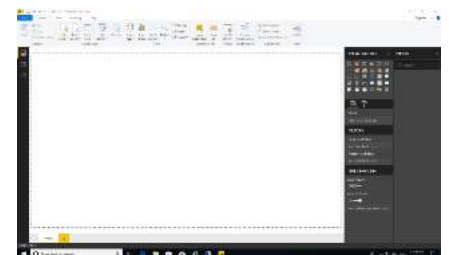
At the bottom of that account creation screen, click on **Already have a Power BI account? Sign in**



You can then close the sign-in screen by clicking on the cross at the top left.



You now have a blank dashboard that is good to go. It's now ready to bring in some data.



Step 2: Importing Data from TASS

Sample Data Sources

In our demonstration, we will work with the following data sources:

Sample Data File Name	TASS API	Method
Student Details.json	Student Details	getStudentsDetails
Student Results.json	Student Analytics	getStudentSubjectsMapResults

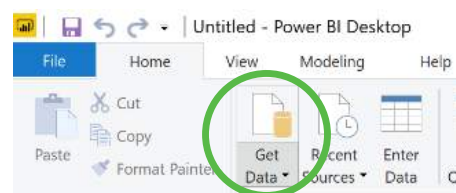
The JSON files contain the exact output of the TASS API.

You will need to work through the following steps for each of the files as they will need to be connected individually.

Set up the Data Connection

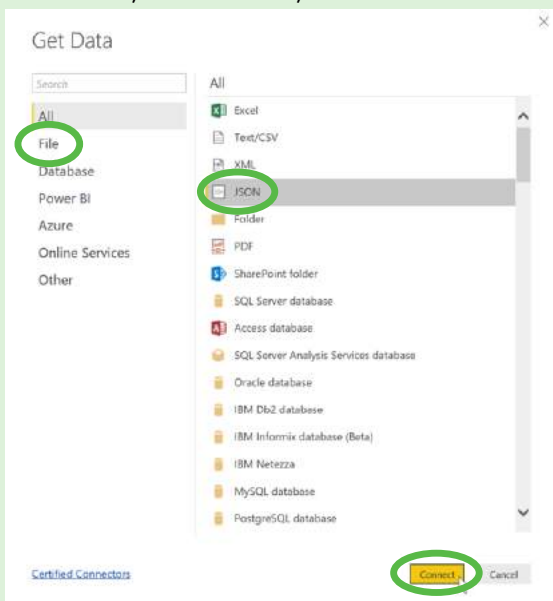
a) In the **Home** tab, click on **Get Data**.

You may then need to click on **More...**



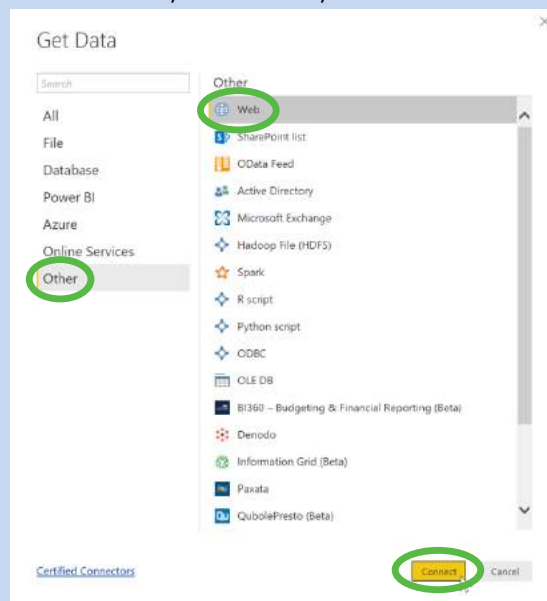
JSON Files (for today's demonstration)

b) Click **File**, click **JSON**, then click **Connect**.

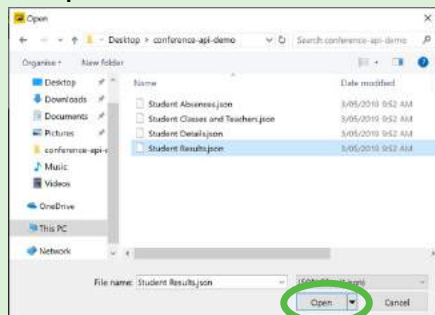


Live TASS API

b) Click **Other**, click **Web**, then click **Connect**.



- c) Navigate to and select your JSON file and click **Open**.



- c) Select **Basic**, paste in your TASS API call URL, then click **OK**.



Transforming the Data for Use in Power BI

We need to transform the data from the hierarchical structure of the JSON file to the tabular structure required by Power BI, so into a table like you would be used to in Excel. You only need to do this when you connect a data source to your dashboard for the first time.

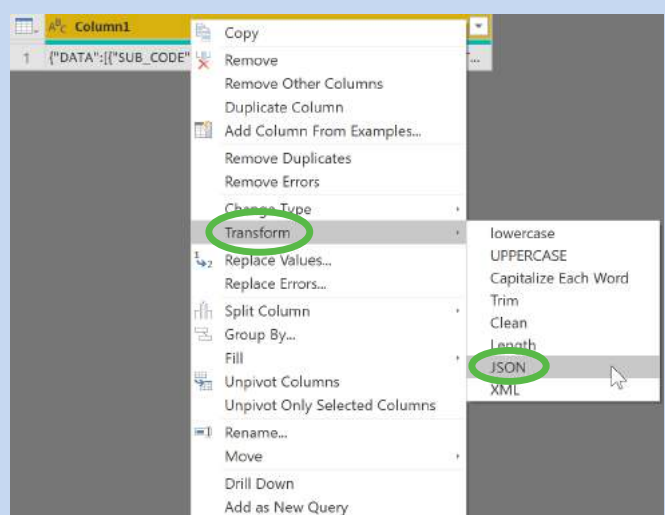
- d) The Power Query Editor window will open, allowing you to transform your data for use by Power BI. In the **Query Settings** pane on the right, in the **Properties** section, enter a meaningful name for the set of data you wish to bring in. This will help you identify it later on.

These steps are only required if using a live TASS API (not a JSON file)

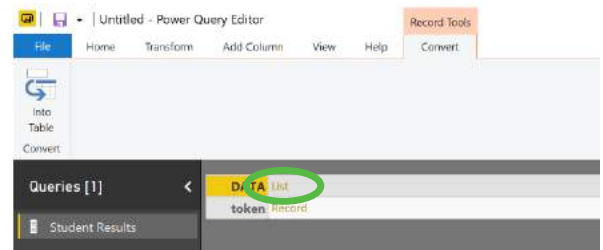
- e) Double-click on the icon with the domain name of your TASS web server in the middle of the Power BI window.

This will bring up a column with a single record, which contains all the JSON-encoded data from the API connection.

- f) To transform this, right-click on the column heading, and in the **Transform** sub-menu, click on **JSON**.
- g) You'll now see a single row containing **Record**, click on it to expand the data.

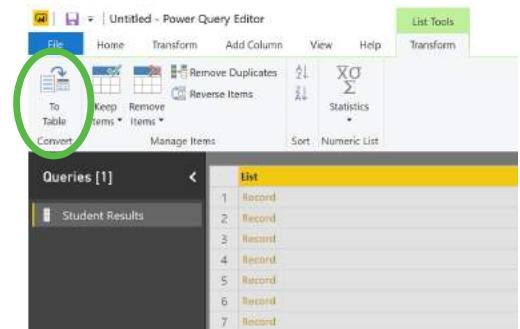


- h) Power Query will break the API response down into the actual data it has returned for us, and the token we supplied. We don't want the token, only the data, so to get this, click on **List**.



Tip: The location of the **List** option may change across data sets.

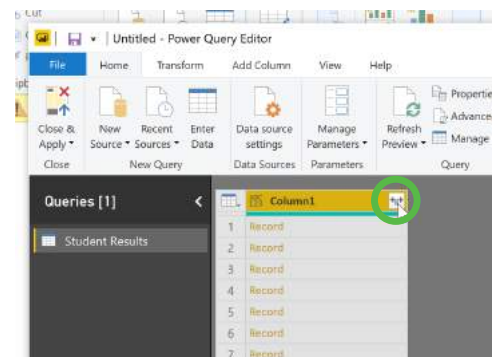
- i) You'll now see a list of rows that all contain **Record**. Click on the list heading, then click **To Table**.



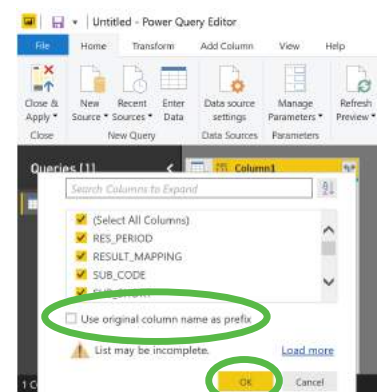
- j) Accept the defaults and click **OK**.



- k) Click the button at the top-right of the column heading to separate the records into their individual columns.



- l) Unselect **Use original column name as prefix** and click **OK**.

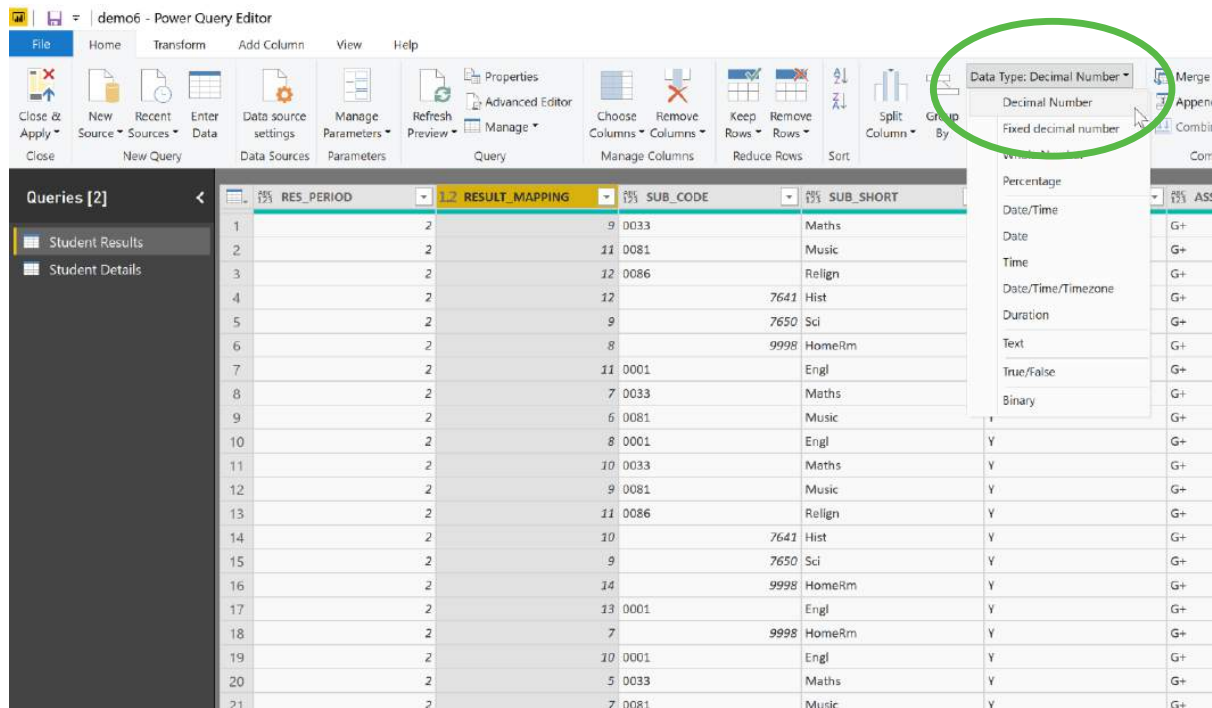


Additional Step for the Student Details File

- m) You'll then see three columns to expand, as the fields are stored in a few different groups. Expand each of these as per steps k and l.

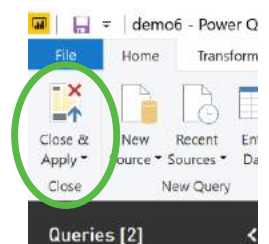
Additional Step for the Student Results File

- n) Click on the **RESULT_MAPPING** column, and in **Data Type** (which you'll find in the **Transform** section), select **Decimal Number**. This will become important later on when we sort the data.

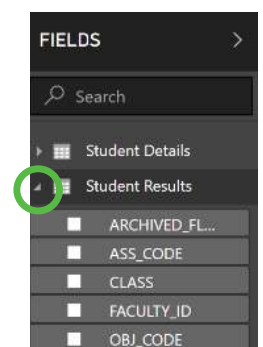


For All Files

- o) Your data is now in a tabular format, ready for Power BI to work with. To bring this data into your model, click **Close and Apply**.



- p) Power BI will apply the changes, and in the **Fields** column on the right, under the name you selected for the data source, you will see the names of each data field within. You can click on the triangles next to each to expand or collapse as required.



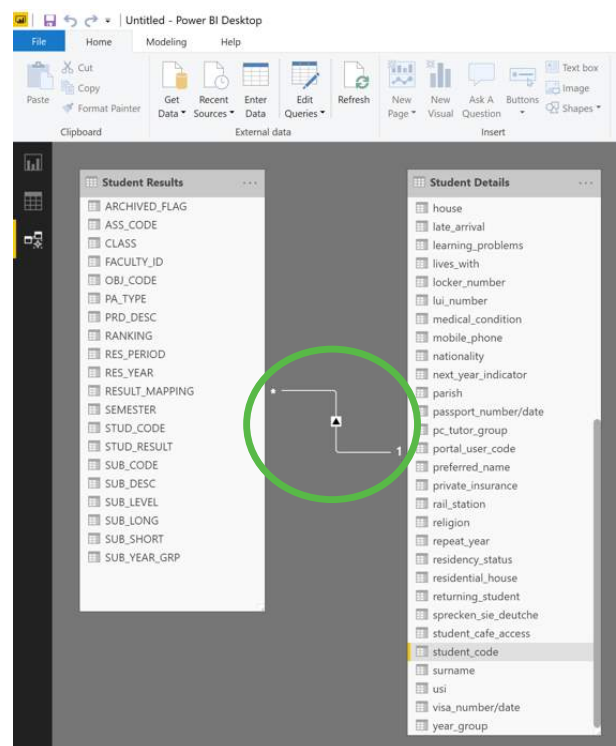
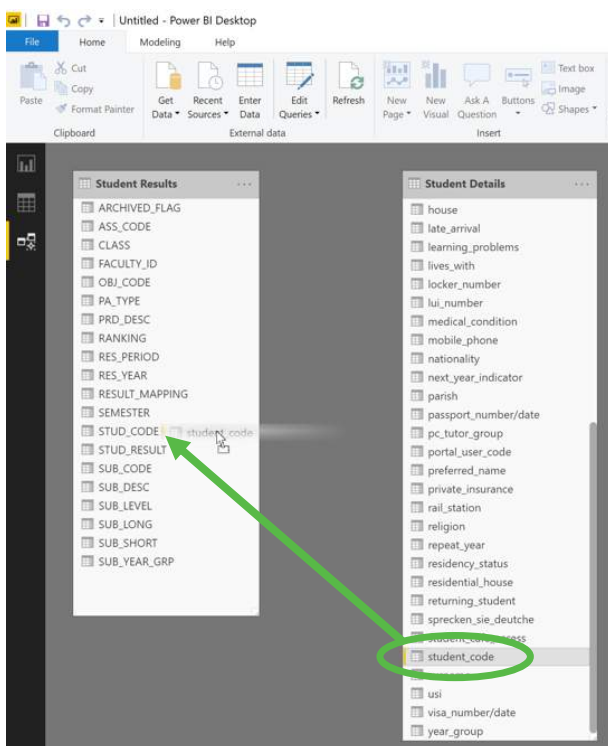
Step 3: Join Data Sources Together

Power BI allows you to bring in multiple sources of data and join them together to form a cohesive data model. To join the data sources, you need to establish a relationship between the fields that match in each data source. If you have used Microsoft Access or Microsoft Query before, this process will feel familiar.

- a) Click on the **Model** button on the left-hand side of Power BI.



- b) You'll see the tables laid out on the screen. If you can't see all of the fields in each table, click on the bottom border and drag it down until you can.
- c) In your **Student Details** table, locate the **student_code** field. Click on it, then drag it onto the **STUD_CODE** field in **Student Results**. Note that doing in in this particular order is important as it removes the need to manually specify the cardinality later on.



- d) We now need to tell Power BI that we want to be able to filter data in both directions. To do this, double-click on the relationship (the line that exists between both tables) to bring up the relationship properties.
- e) Set Cross Filter Direction to Both, then click OK.

Student Results

RES_PERIOD	RESULT_MAPPING	SUB_CODE	SUB_SHORT	ARCHIVED_FLAG	ASS_CODE	RES_YEAR
1	94	0028	SOSE	Y	N	2014
1	36	0028	SOSE	Y	N	2014
1	92	0028	SOSE	Y	N	2014

Student Details

surname	next_year_indicator	date_of_leaving	student_code	usi	portal_user_code	mobile_phone
Boulton	Advancing		0009581	000958110		
Sullivan	Advancing		0009585	000958510		
Moloney	Advancing		0009318	000931810		

Cardinality: Many to one (*:1)

☒ Make this relationship active

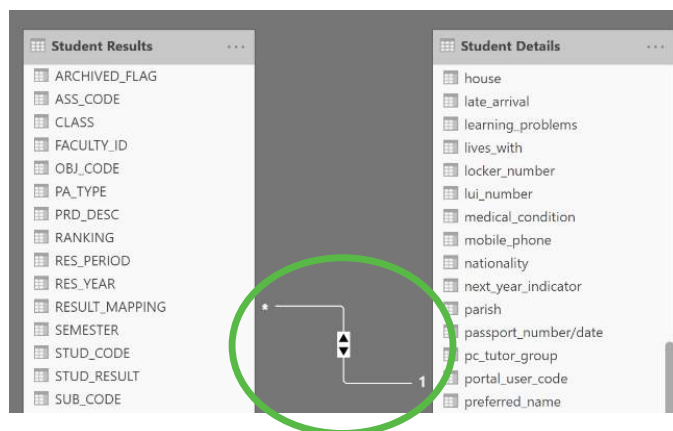
☐ Assume referential integrity

Cross filter direction: Both

OK Cancel

You'll note the relationship line updates to have arrows pointing in both directions.

Tip: Power BI only allows a single column from each table in a relationship.

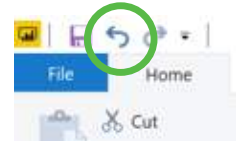


- f) You can now click on the **Report** button to return to your blank dashboard.

Building Your Interactive Student Results Dashboard

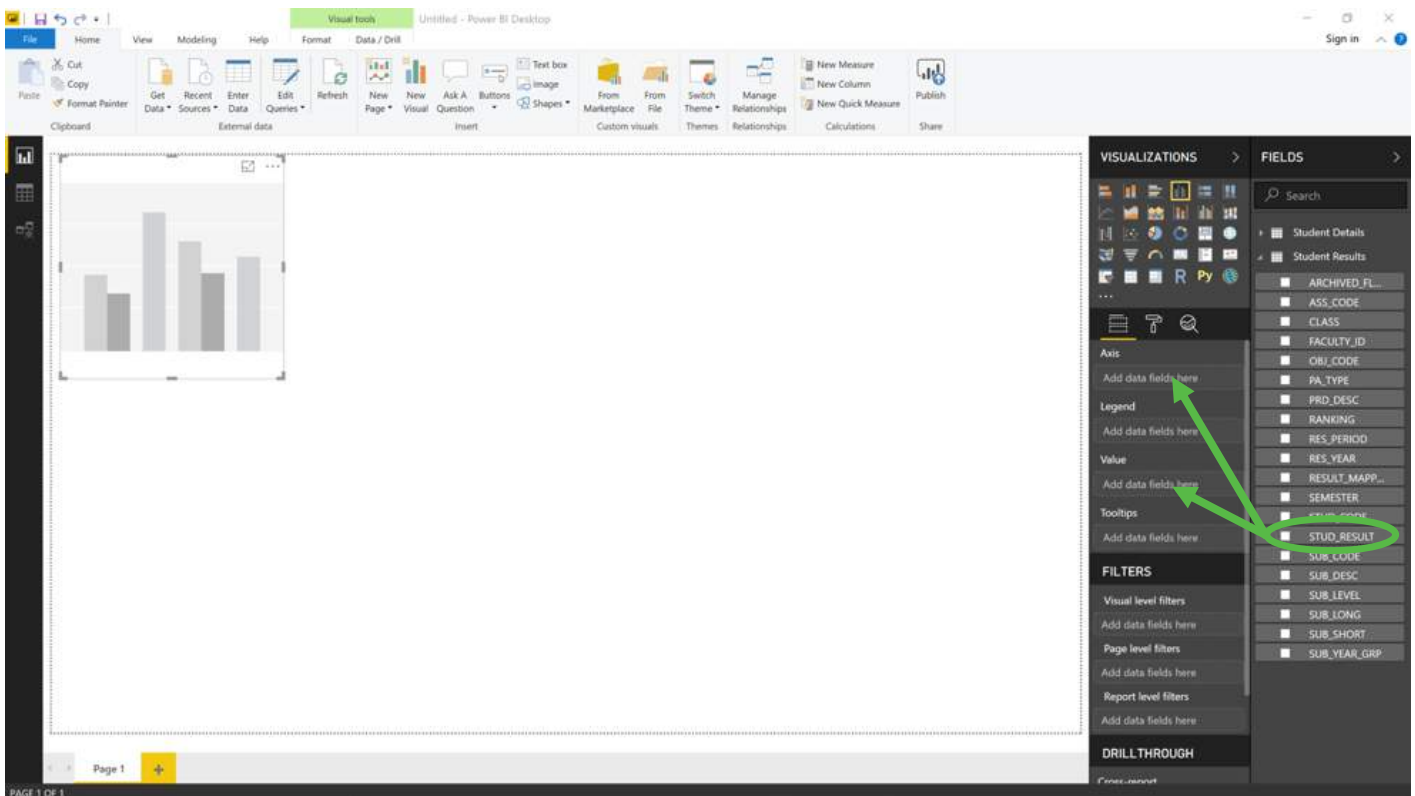
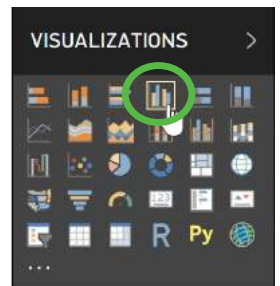
We are now ready to build the interactive dashboard.

Tip: There is an Undo button at the top of the window. You can use it at any stage if you make a mistake and need to go back.



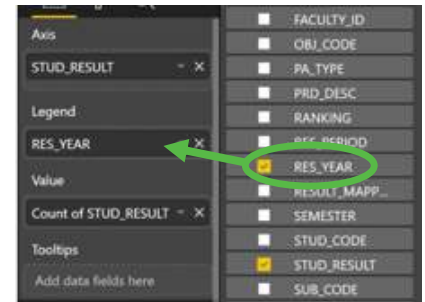
Step 4: Student Result Distribution Chart

- Click on **Clustered Column Chart**. A blank visualisation will appear on your dashboard page.
- In the **Fields** pane, under **Student Results**, locate **STUD_RESULT**. Drag it into **Axis**, and also drag it into **Value**, which you'll find in the Visualisations column.

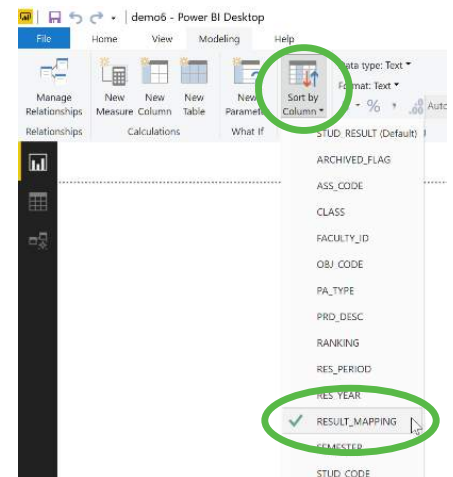


We now get a column chart showing the frequency of each result.

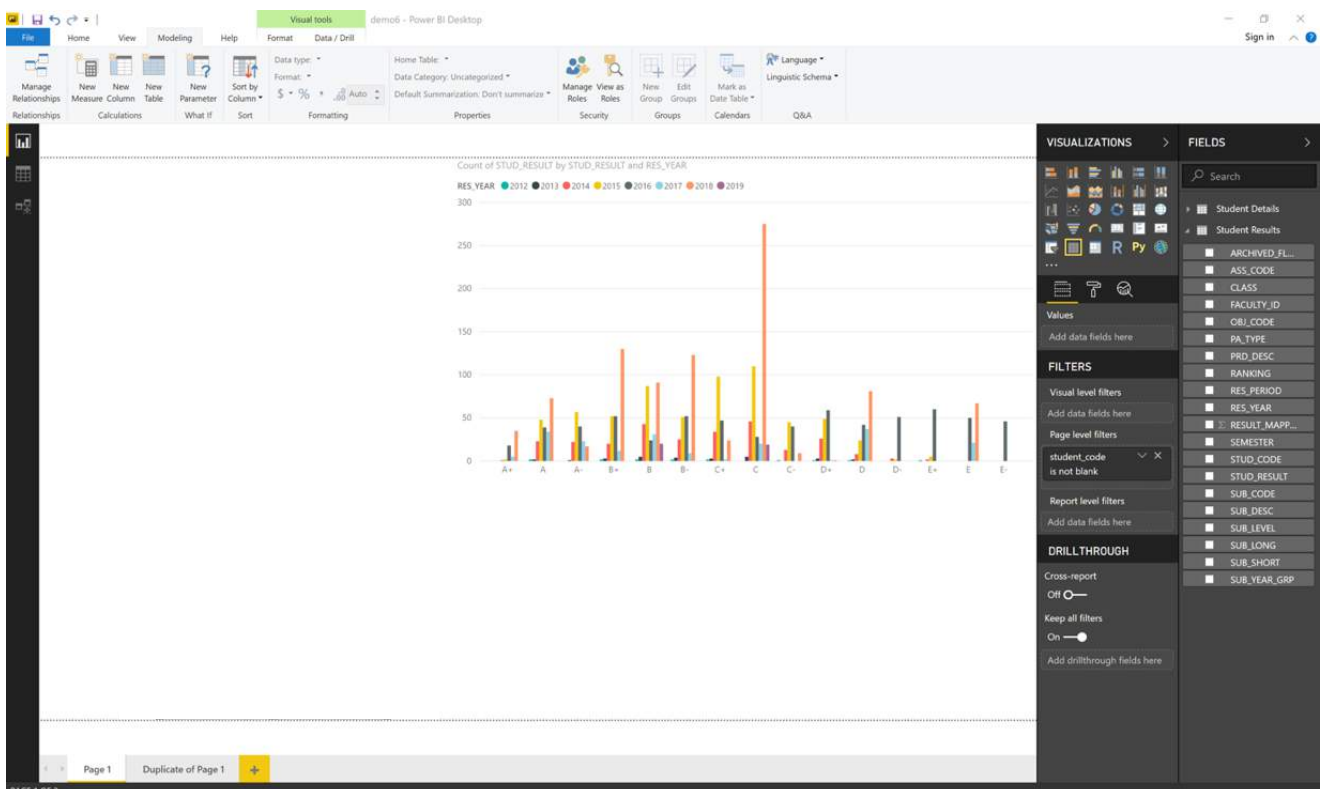
- c) Separate out each year by dragging **RES_YEAR** into the **Legend** section.
- d) The chart is quite small, so grab the handles on the sides or corners and drag them to make it bigger, then click and drag the entire visualisation to move it to the top right of the dashboard.



- e) The chart now needs to be sorted in grade order. In the **Fields** pane, click on **STUD_RESULT**.
- f) Go to the **Modelling** tab at the top of the screen, and click on **Sort by Column**. Select **RESULT_MAPPING**. You'll notice that the chart will now be sorted in the correct order.

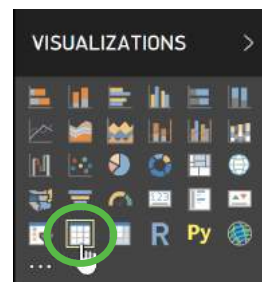


You'll now have a grade distribution chart that looks like this:



Step 5: Student Result Table

- Before attempting to add a new visualisation, click in a blank space to deselect the currently selected visualisation (if there is one).
- Click on the **Table** visualisation icon to create a table. Like before, you'll see your blank visualisation placeholder on your dashboard.
- Drag the following fields in to the **Values** section in the **Visualisations** column.



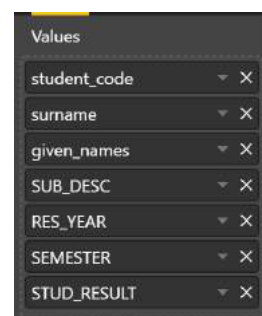
Tip: Use the search box at the top of the **Fields** pane.

From **Student Details**:

- student_code
- surname
- preferred_name

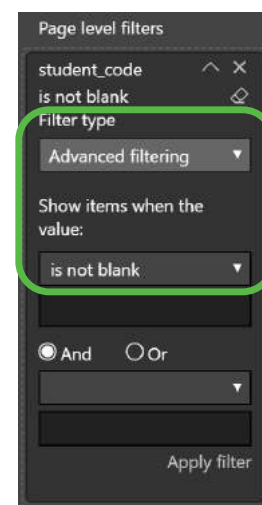
From **Student Results**:

- sub_desc
- res_year
- semester
- stud_result



Tip: The order can be changed at any time by clicking and dragging on the fields to rearrange them.

- Drag the table to the bottom right and resize it so you can see all of the columns, and adjust the width of the columns by clicking and dragging the column borders.
- Apply a page level filter to remove results without student names:
 - Under **Student Details**, drag **student_code** into the **Page Level Filters** section of the **Visualisations** pane (note that you might have to scroll down to see it).
 - Select **Advanced Filtering**
 - Under **Show items when the value**, select **is not blank**.



Note: This occurs because our files contain all results, but only current students.

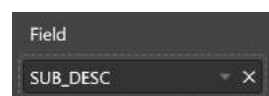
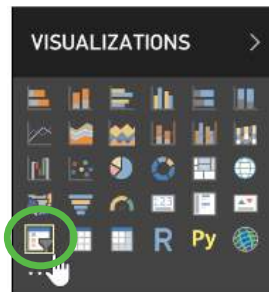
The finished table will look like this:

student_code	surname	given_names	SUB_DESC	RES_YEAR	SEMESTER	STUD_RESULT
20066	Allen	Joseph	English	2016	1	C-
20066	Allen	Joseph	English	2016	2	D+
20066	Allen	Joseph	English	2017	1	A-
20066	Allen	Joseph	English	2017	2	D
20066	Allen	Joseph	English	2018	1	B-
20066	Allen	Joseph	English	2018	2	C+
20066	Allen	Joseph	History	2016	1	D+
20066	Allen	Joseph	History	2016	2	A+
20066	Allen	Joseph	History	2017	1	B+
20066	Allen	Joseph	History	2017	2	A
20066	Allen	Joseph	History	2018	1	B
20066	Allen	Joseph	History	2018	2	D
20066	Allen	Joseph	Mathematics	2016	1	C
20066	Allen	Joseph	Mathematics	2016	2	A+
20066	Allen	Joseph	Mathematics	2017	1	B-
20066	Allen	Joseph	Mathematics	2017	2	A

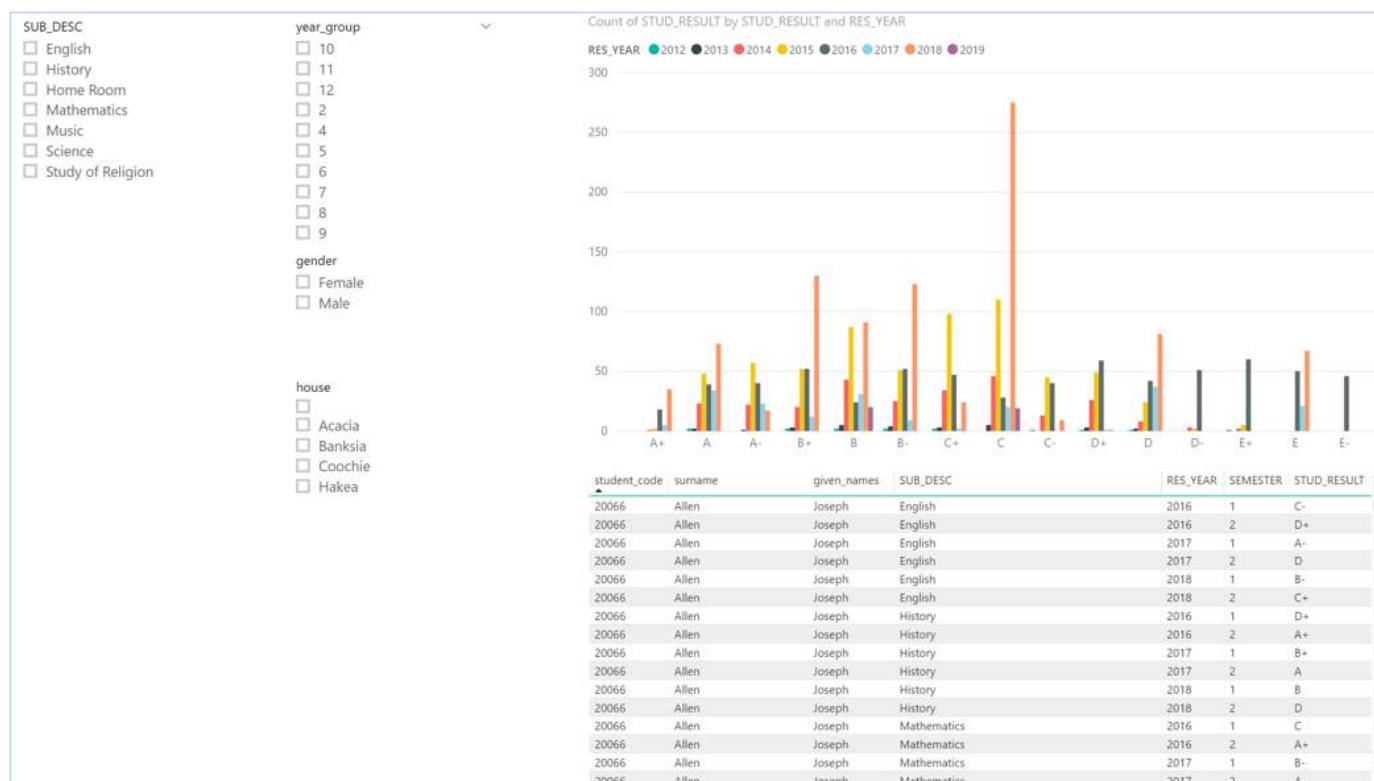
Step 6: Add Slicers to Filter Data

Now that we have some visualisations on the page, we can add some slicers to help us dig in to the data. Slicers are one of the tools in Power BI you can use to make your dashboard interactive.

- Before attempting to add a new visualisation, click in a blank space to deselect the currently selected visualisation (if there is one).
- Click the **Slicer** visualisation in the **Visualisations** pane. A blank slicer will appear on your dashboard.
- Drag the field you wish to use in your slicer into the **Field** list. For our first one, let's use **SUB_DESC** from **Student Results**.
- As with other visualisations, you can click and drag your slicer to your desired location on the dashboard, and also resize it as required.
- Repeat the same process to add filters for **year_group**, **gender**, and **house** (under **Student Details**). You now have a range of slicers you can use to further analyse your data.



Your completed dashboard will look like this:



Step 7: Explore Your Data

As you click on the various options in your slicer, you'll notice that all of your visualisations are filtered based on your selection. This allows you to explore your data in further detail. To remove your filter, simply click on your filter criteria on to de-select.

It is also possible to 'drill-down' into your data by clicking on various parts of your visualisations (eg one for the columns in your chart). This will grey out all of the unrelated data on your dashboard. To go back, just click on the data again to deselect it.

This visualisation could help you answer questions such as:

- Which subjects, houses, genders, and year levels have students that perform better than others?
- Are results improving year-on-year?
- Which students are achieving results in various grade bands?

Where Next?

Now that you have built your first dashboard in Power BI, here are some next steps:

- Bring in additional data sources (eg other TASS API endpoints including Standardised Testing, Student Absences).
- Add additional visualisations and slicers to make sense of data in different ways.
- Dress up with your dashboard with some formatting.

You can also download a copy of the completed masterclass dashboard from our landing page: <http://info.tassweb.com.au/api-masterclass-downloadable>

TASS Professional Services

Our Professional Services team can assist your school with their API integration and data analytics needs. Visit www.tassweb.com.au for more information.

TASS API Summary

TASS API	BASIC	PREMIUM	DETAILS
Accounts Payable Integration	Included	Included	Supports third-party OCR scanning and workflow automation software, for the creation of Invoices in TASS.web program Accounts Payable > Supplier Transactions > Invoices without the need for any manual data entry.
Boarding	\$\$	Included	Extracts and passes basic student and parent contact information (using Communication Rules) for students who have been flagged as a boarder within TASS.web.
Data Upload Utility	Not Available	\$\$	Mirrors the functionality available in TASS.web Data Upload Utility but via API instead of CSV upload.
Employee/HR (Extraction)	\$\$	Included	Extracts Employee Photos, HR General tab, Address tab, Next of Kin, Leave Balances and HR User Defined.
Library Management	\$\$	Included	Extracts Student Photos (if enabled in TASS.web), Basic Student information, Basic Employee information, and Parent Contact information based on a student's communication rules.
LMS Integration	Not Available	\$\$	Extracts student photos plus basic student, teacher and subject Information.
Mobile App	Not Available	\$\$	Supports third-party Mobile Apps and the following connections: <ul style="list-style-type: none"> ▪ Parent login authentication including split family access. ▪ Student eDiaries inc Timetable, LMS/Assessment Activities, Sports Fixtures, Parent Teacher Interviews, Boarders Leave, Pastoral Care, Personal Calendar, Tours and Excursions. ▪ Tours and Excursions details with SSO deep linking to Parent Lounge to action the approval and payment of the tour. ▪ School calendar and daily notices. ▪ Notifications including targeted push notifications. All based on Parent Lounge permissions.
Online Enrolments	\$\$	Included	Enables enrolment applications submitted using third-party software to be entered into TASS without manual data entry.
School Calendar / Daily Notices	Included	Included	This API allows the school to extract the details of Daily Notices and Calendar Events flagged for visibility in the Public Calendar.
Student Analytics API	Included	Included	Extracts basis subject information, student attendance and results, and statistical data for use in third-party software such as Microsoft Power BI.
Student Details (Extraction)	\$\$	Included	Extracts student photos, Student Record General and UD tabs, and Parent Contact information based on a student's communication rules.

Student Analytics API Methods

getActivityAnalyticMapping

Returns Progressive Assessment activity analytic data including activity name, objective code, assessment code, statistics type, class, activity statistic result and analytic mapping.

getActivityMinMaxResults

Returns Progressive Assessment activity minimum and maximum result data including activity name, objective code, assessment code, validation type and minimum/maximum results.

getClassLessons

Returns Timetabled Lessons including timetable ID, timetable date, day, period, subject, year group and class.

getStatisticGroups

Returns statistic groups used for standardised testing. Examples might be State Boys, National Girls

getStatisticTestingData

Returns statistic types data used for standardised testings, such as Average, Median, Quartile 1.

getStatisticTypes

Returns statistic types data for standardised testing including year group, test code, criteria code and study year.

getStudentAbsences

Returns student absence data including student code, timetable ID, absent date, week, day, period, subject code, year group, class and acceptable reason.

getStudentActivityAnalyticMapping

Returns student Progressive Assessment activity analytic mapping data including student code, activity name, objective code, assessment code, class, analytic mapping, class rank and cohort rank indicator.

getSubjectMapResults

Returns student subjects with mappings for results. This data includes student code, year group, subject code, class, result year/period, period description, semester, subject description (standard/long/short), faculty ID, objective code, assessment code, progressive assessment type, result, result mapping, ranking and archived flag.

getStudentSubjectResultsProcess

Returns student results for Progressive Assessment activities including mappings for results. This data includes student code, year group, subject code, class, result year/period, semester, objective code, assessment code, result, result mapping, ranking and percent completion.

getStudentSubjects

Returns student subject data including student code, timetable ID, subject code, subject description, year group, class and lesson count.

getStudentSubjectsTeachers

Returns student subject and the associated teacher data including student code, subject code, subject description (standard, long and short), year group, class, subject archived flag, result year/period, semester, teacher code/name and faculty ID.

getStudentTestingData

Returns student standardised testing data.

getStudentWeightedResults

Returns student weighted result data for Progressive Assessment activities including student code, year group, results year/period, semester, objective code, assessment code, result, result mapping, weighted sum, weighted count, unweighted count and archived flag indicator.

getSubjectClassMinMaxResults

Returns minimum/maximum results including results year/period, set code, subject code, objective code, assessment code, validation type, minimum/maximum result and archived flag.

getSubjectClassQuartileResults

Returns result analytics data including subject code, subject year level, results year/period, class, statistic type, objective code, assessment code, progressive assessment type and archived flag.

getTestCriteria

Returns test criteria used for standardised testing.

getTestTypes

Returns test types used for standardised testing.