UNIVERSITY of York
Information Services

Essential Spreadsheets

Exercises

IT

www.york.ac.uk/it-services/training

Spreadsheets – Exercises

Exercise files

The files for these exercises are available in both Excel and Google Sheets format, except where a feature is unique to a particular application.

To access the Google Sheets versions, visit the support site, Understanding Spreadsheets

Excel versions are available on the Teaching(T:) drive on all classroom and managed PCs:

Teaching(T:)\IT Training\Essential Spreadsheets

Last updated: June 2016

Exercises – Introducing Spreadsheets

Workspace 1

Open the file **Workspace 1** and make sure you are viewing Sheet1, which shows the beginnings of an attempt to keep a record of project spending.

1. Rename Sheet1 'Year 14-15' and set a red colour on the tab.
2. Increase the width of column B so text in rows 3-12 fits OK.
3. Reduce the height of row 7 to match the other rows.
4. Adjust the width of columns D-I so they are all equal and the dates in row 12 are clearly visible.
5. Select the data in cells D20:I26 and move it to the range D3:I9.
6. Column C is empty, so remove it completely.
7. Use autofill to complete the months in row 2 from October (col C) to March (col H).
8. Insert a row between **Subsistence** and **Printing**, and the label *'Hospitality'* in col B.
9. Enter these hospitality values for Oct-March: 50, 0, 25, 35.5, 0, 65
10. Using autofill, complete the item refs in column A, rows 3-9, with items numbered 1, 2, 3, 4 etc down to 8.
11. In row 2 turn on text-wrap and adjust row height, if necessary, so 'six-month subtotal' wraps to two lines in column H.
12. There is a redundant copy of the expenditure values on Sheet2 - delete this sheet completely.
13. Sheet3 is data from a previous year - rename Sheet3 as 'Year 13-14'.

Playing with Autofill

Open the file **Auto fill - values**. The examples tab contains several different 'starters' for lists.

Select the shaded (green) cell(s) in a column, and then drag from the bottom right corner of the selected area to see how AutoFill completes the list.

Notes:

1. If you provide the first two or more items in a regular series, autofill can continue the pattern, either forwards or backwards:
    a. days (cols B, C)
    b. months (cols E, F)
    c. years (cols G, H)
    d. dates (cols I, J)
    e. numbers (cols K, L, M)
2. Other single cells or patterns are repeated (see cols A, D & N)

Exercises T1 - Calculating with Spreadsheets

Constructing formulas

Open the file **T1 - Formulae**

1. Switch to the **Sandwiches** tab.
    a. In cell C18, create a formula to calculate the cost of the sandwich (white bread with cheese) shown in B18.
    b. Use autofill to replicate this formula across the row to calculate the calories, protein, carbs and fat content of the sandwich in row 18.
    c. Repeat these steps to fill out all the costs and nutritional information for the other sandwiches in cells C19:G22.
    d. In row 24 use the **Sum** function to calculate the cost and nutritional information of all the sandwiches for the week.

2. Switch to tab **Orders**.
    a. Adjust the width of column A so the item descriptions fit OK.
    b. In cell D4 use a formula to calculate the cost, based on the price in B4 and quantity in C4.
    c. Autofill this formula down to row 10.
    d. In column F, calculate the total price by subtracting the discount from the subtotal.
    Note: The discount is pre-calculated using conditional functions.

3. Switch to the **EnergyCosts** tab.
    This sheet contains the amounts spent on Electricity and Gas each month over 1 year.
    a. In cell B10, calculate the cost of electricity in January. This will be the number of units used (B8) multiplied by the electricity unit cost (P7).
    Hint: In order to be able to copy this formula for other months, you will need to use an *absolute reference* for P7.
    b. In cell B11 calculate the monthly charge for January, based on the daily charge in P3 and the number of days in January (B2).
    Hint: In order to be able to copy this formula for other months, you will need to use an *absolute reference* for one of these.
    c. In cell B12 calculate the cost of electricity for January by adding the unit charge and monthly charge.
    d. Check these three formulae, particularly as regards the use of absolute references, and copy them to calculate costs for all the remaining months.
    e. Likewise, in cells B19, B20 & B21, calculate the cost of gas for January, using the units cost in cell P8 and the daily charge in cell P4. Make sure you use absolute references where appropriate.
    f. In cell B23 calculate the cost of both electricity and gas for January, and in cell B24 the cost per day. Copy these for all months.

g. In cell P12 and P13 use **Sum** functions to calculate the total cost of electricity and gas for the year, and calculate the total energy cost for the year in P14.
h. Calculate the total number of days in P16, and in cell P17 calculate the mean (average) daily cost for energy.
i. There is a lot of inconsistency in the display of values. Format all cells in columns B-M that show costs to 2 decimal places. Display the total costs (P12:P14) as currency and the mean daily cost (P17) as currency to 3 decimal places.

4. Switch to the **Mileage** tab.
   This is used, in conjunction with the **MileageData** tab, to record car trips for work purposes so a mileage allowance can be calculated. **MileageData** includes the charge per mile and also needs to include totals for the number of miles and cost.
   a. In cell E2 of the **Mileage** sheet, enter a formula to calculate the claim cost of the distance shown in D2 - this must use the *Cost per mile* figure in B2 on the **MileageData** sheet and should convert the figure to £.
      Hint: you will need to use an absolute reference to ensure the formula will replicate correctly down the column.
   b. Replicate this formula down the column.
   c. Switch to the **MileageData** tab and in cell B4 use a suitable function to total the number of miles recorded in column D of the **Mileage** sheet. You need to ensure this will still be correct when more values are added to the **Mileage** sheet.
   d. In cell B5 of the **MileageData** sheet use a suitable function to total the costs in column E of the **Mileage** sheet. You need to ensure this will still be correct when more values are added to the **Mileage** sheet.
   e. Both sheets have poor number formatting. Ensure that values are displayed to an appropriate number of decimal places, allowing for the fact that mileage is recorded to the nearest 0.1 miles.

Using functions

Open the file **Functions 1**

1. Switch to the **Numbers** tab.
   This is a simple data set, with space to calculate some summary data using common functions, including **count, sum, average, max** and **min**.
   a. In cells I2:L2 use functions to calculate the total, average highest and lowest for group A.
   b. Replicate these formulae down the columns to obtain results for the other groups.
   c. Complete cells C10:C14 by inserting appropriate functions to calculate figures for the whole data set - note that there are two possible solutions for most of these.

2. Switch to the tab **Students**.
   This sheet shows the Numeracy, Literacy and Science assessment marks for a group of primary school pupils. The tests were out of 110, 125 and 95 as shown in row 2, but they need to be shown as percentages.
   a. In cell G5, enter a formula to convert the *Numeracy* mark in D5 to a percentage - divide the numeracy mark by the value in D2 and format as a percentage. Use absolute references as appropriate to ensure it will replicate down the column.
   b. Likewise, convert the marks in E5 and F5 to percentages in H5 and I5 respectively.
   c. Copy these three formulae down the columns to generate values for the remaining students.
   d. In column J use the **average** function to calculate the average %mark for each student (average of columns G, H & I). Likewise, use suitable functions in columns K and L to show the highest and lowest %marks for each pupil.

3. Switch to the tab **Student Summary.**
   This sheet calculates some summary figures for the data on the **Students** sheet.
   a. In cell B6 enter a function that will **Count** the number of pupils listed in column A of the **Students** sheet.
   b. The collection cells B2:D4 is intended to calculate the average, highest and lowest percentages in the three assessments. In cell B2 enter a function to calculate the average of the *Numeracy* results on the **Students** sheet, column G.
   c. Add the appropriate functions to find the highest and lowest *Numeracy* results, and similarly find the average, highest and lowest for *Literacy* and *Science*.

Exercises T2 – Creative functions

There's quite a lot of exercises in this section, but if you don't need to work much with dates and times you could skip some of these and return to them another time.

Knowing how to use 'Names' and 'Conditionals' is really useful, though, so make sure you don't skip this!

Names and Conditionals

Student Info

Open the file **Student Info**

The **Students** tab is a list of the students taking the *Culinary Appreciation* course, each of whom also makes a donation towards food for a homeless charity (so we can practice using the SUMIF function).
You are going to process this data to find out 'useful' information.

1. We need to know if a student has passed, so **Students** column H is prepared for this. The pass mark is set on the **Admin** tab, cell B2. To make it easier to use this value, make this cell a named range called *passMark*.
2. Back on the **Students** sheet:
    a. In cell H2 enter an IF function that will display "Pass" if the mark in column F is greater than or equal to the value in *passMark*, or "Fail" if it isn't.
    b. Copy this function down the column - it should replicate correctly.
    c. Try changing the value in ***passMark*** (on Admin) to see the change in **Students** column H.
3. To make it easier to work with the columns of data on the **Students** tab, make them into named ranges as follows:

    All of column D – *module*
    All of column E – *year*
    All of column F – *mark*
    All of column G – *donation*
    All of column H – *result*

    Now we'll use these to process the **Students** data - switch to the **Admin** sheet for this.
4. In cell B6:
    a. Use a COUNTIF function to find how many students are taking the *Pies* module. Use the named range you defined for the module column and although you could 'hard code' the word Pies into the function for the criterion, it's better to reference it from **Admin** cell A6.
    b. Copy this function down for the other 3 modules - it should replicate OK.
5. Complete the cells for *Donations total, Average marks* and *Number of passes* using appropriate conditional functions. There are some subtle hints in row 11, but note that *Number of passes* needs two criteria - you're looking for the correct module (pies etc) and also for the word Pass in the Result column - that's why it needs the plural version *countif**s***.
6. In **Admin** F6:F9 calculate the *Pass rates* (a simple division, formatted as a percentage).

7. Try changing the *Pass mark* again (**Admin** B2) to see the effect this has on these calculated results.
8. Now the hardest bit: The group of cells **Admin** B17-E20 are very similar to the first set, except this time the year group shown in B14 needs to be used as an additional criterion so only results for this year are shown - changing the year should change the results. This means that all the functions need to use their 'plural' versions. Include the pass rate on the end as before.

Holidays

Open the file **Hols**

Switch to the **Hols** tab. This sheet is used in conjunction with **HolsAdmin**, to calculate the cost of a group of people staying for several nights at a campsite.

The arrival dates will be entered in cells D1 and D2 of this sheet, and details of the party will be entered from row 9 downwards (some sample data is already entered). This, together with values for discounts for concessions and larger groups, will be used to calculate the cost.

In order to make formulae easier to work with, cells containing key values in will be configured as **named ranges**: these names can then be used in formulae without having to worry about using dollar notation for absolute references.

1. Create **named ranges** for the key data cells, as shown below:

sheet tab	cell	name	purpose
HolsAdmin	B1	dailyRate	standard daily charge per person
	B3	vatRate	VAT rate to be applied to final bill
	E1	concessionsDiscount	discount rate for concessions
	H1	partyLimit	number in party at which discount is applied
	H2	partyDiscount	discount rate for larger parties
Hols	D3	days	number of days to be charged
	D4	partySize	total number of people in party

   Note: the names are also shown as notes on the appropriate cells.

2. An easy calculation: On the **Hols** tab, cell D3, work out the number of days by subtracting the arrival date from the departure date - ensure this is formatted as a number.
3. In cell D4 count how many people are in the party (use the list of names in column B) - you'll need to use COUNTA (not COUNT) as you are counting text not numbers. To allow for adding or subtracting people from the list, count the whole column but subtract 1 to take into account the label at the top ('Name').

4. Still on the **Hols** tab, cells in column E, starting with E9, need to calculate the daily rate for each person, using the standard daily rate ('s' in column C) or applying a discount if they qualify as a concession (shown by a 'c' in column C). This will require an **IF** function.

   To test if the person qualifies for a concession, as the testable condition in E9 we need to use:

   C9 = "c"

   If a person qualifies as a concession, using named cells, the calculation is:

   dailyRate - (dailyRate * concessionDiscount)

   If a person does not qualify then the value is simply the dailyRate.

   a. Construct the IF function in E9 using the 'hints' above. Make sure you use named cells where appropriate so the formula will replicate down the column correctly.
   b. Copy the formula down the column and check it is correctly identifying concessions.

      Note: You may notice that this formula still generates a daily cost for rows where there is no person entered - this could be rectified by using a more complex formula, but for now simply copy it only as far as the last name.

5. Now two easier bits:
   a. In cell H1 calculate the total daily cost by adding the values in column E (for maximum flexibility total the whole column).
   b. In cell H2 calculate the cost of the visit for the number of days in the visit - remember the number of days is one of the named cells.
6. In cell H3 a larger party discount needs to be calculated **IF** the number of people in the group is equal to or greater than the party limit on **HolsAdmin**. Use named cells again wherever you can. There's a subtle hint above as to which function you may need…
7. Finally, complete the calculations on the **Hols** tab in cells H4 to H6:
   a. H4 - subtract the party discount from the charge for visit.
   b. H5 - calculate VAT for the amount in H4 (remember vatRate is a named cell).
   c. H6 - add the VAT to the Cost of the visit to find the amount payable.
8. Make some changes to the composition of the party to check values are changing as they should, in particular to ensure the larger party discount is only applied when it should be.

   Also make some changes to key admin values such as the size of a 'larger' party and the concessions discount rate.

Expenses

Open the **Expenses** file

Details of expense claims are entered on the **Expenses** tab. Summary information is calculated on the **Summary** tab, which also includes the value at which an expense must be checked with the manager.

1. To make later formulae simpler, on the **Summary** tab name cell B2 *checkLimit*.
2. Switch to the **Expenses** sheet:
    a. In cell E2 construct an **IF** function that will display "Check with manager" if the amount in D2 is greater than the check limit set in the newly named cell; otherwise it should display "ok".
    b. Copy the formula down the column and check it has replicated correctly.
    c. Also try changing the value for the check limit (on **Summary**) - messages should change as appropriately.
3. Two easy ones - switch to the **Summary** tab:
    a. In cell B4 enter a function to count the total number of entries in **Expenses** column A.
    b. In cell B5 enter a function to total the amounts in **Expenses** column D.
4. Now the harder ones:
    a. Cells B9 to B12 need to count the number of claims made by each person - to do this you'll need to use the COUNTIF function in each of the cells. Ensure they will remain correct if more items are added to the list.
    Hint: You'll be counting in **Expenses** column B, and you should be able to reference the person's name on the **Summary** sheet for the criterion.
    b. Cells C9 to C12 need to show the total of expenses for each person - you will need to use the SUMIF function for this. Ensure it will remain correct if more items are added to the list.
    Hint: the values for totalling are in **Expenses** column D, but the values for testing are in column B. As with COUNTIF you should also be able to reference the person's name on the **Summary** sheet.
5. If all has gone well, trying making some changes:
    a. Change the check limit on the **Summary** sheet to a low value (eg £10) and check the effect this has on the **Expenses** sheet.
    b. On the **Expenses** tab, add some more expenses (using the same staff names in column B).

Dates and Times

Always bear in mind the basics:

- dates are stored as a single whole number, and the number increases by one each day
- times are stored as a single decimal number, less than 1
- when working with times it is important to distinguish between 'time of day' and 'duration'
- the display on the sheet is entirely the result of formatting these numbers
- as dates and times are stored as numbers, you can perform calculation with them

Open the file **Dates and times**

Dates

Switch to the **play with dates** tab. Cells B2 and B3 contain entered dates, and columns C-G in these two rows simply replicate these two dates, but formatting can be used to display these differently. Below is space to try some date-related functions.

1. Configure the dates in rows 2 & 3 as indicated in the header row:

format	example
medium date	30 Jun 15
long date	30 June 2015
day	Tuesday
custom	Tuesday, 30 June 2015
date value	42,185

   You may need to use custom formatting, using combinations of:

   d, dd, ddd, dddd (days)

   m, mm, mmm (months)

   yy, yyyy (years)

2. In cell B4 find the number of days between the two entered dates by simple subtraction - you may need to reformat the result as a number.

   Try the same calculation in cell G4 - you should get the same result.

3. Another date has been entered in cell A8.
   a. In cells C8 - C11 try the functions indicated, with A8 as the argument, which should return individual components of the supplied date:

day(A8)	day of the month
weekday(A8)	day of the week, where Sunday = 1
month(A8)	month number
year(A8)	4-digit year

b. In cells D13 - D16 try some further functions that calculate another date from the one supplied. All except the first require 2 arguments: the date (A8) and a number of days or months to add/subtract):

simple addition/subtraction	adds/subtracts days
edate(A8,C14)	adds/subtracts months
eomonth(A8,C15)	adds/subtracts months but returns the last day of that month
workday(A8,C16)	adds/subtracts days, but omits weekends

Times

Switch to the **play time** tab. Cells B2 - F3 have some start and end times entered for some days of the week.

1. In cell B5 use a simple subtraction to find the difference between the start and end times for Monday, and replicate this for the other days. Check the result is formatted correctly.
2. In cell G5 insert a SUM function to total the times. This is where the distinction between time of day and time duration becomes apparent, as if G5 is formatted as time rather than duration, it will not sure the correct answer (you may wish to try the two possibilities). All the cells B5 - G5 should be formatted as duration, so make sure you do this.
3. A duration displays time appropriately, but cannot easily be used in calculations, for example to multiply the number of hours by an hourly cost. In cell H5 enter a formula to multiply the duration in G5 by 24 to obtain the duration as decimal hours. You may need to think about why this works!
4. Time of day can be displayed using 24hr or 12hr notation, and can be set not to display leading zeros on hours. Cell A9 contains an entered time value and B9 - E9 replicated this. Apply formatting as indicated - you may need to use custom formatting (examples shown here):

HH:mm:ss	hours, minutes and seconds (24)
HH:mm	hours & minutes (24)
H:mm	hours & minutes (no leading zeros, 24)
H:mm AM/PM	hours & minutes (no leading zeros, 12)

5. Cell A13 also contains an entered time. In C14 - C16 to extract the hour, minute and second components from the time, using the functions indicated:

hour(A13)	extract the hour (24hr clock)
minute(A13)	extract the minute
second(A13)	extract the seconds

Date-times

As dates are stored as whole numbers, and times as a decimal less than 1, it follows that you can potentially store both a date and a time using just one value, using formatting to generate the appropriate display. It even allows you to perform calculations involving both dates and times.

Switch to the **play date-time** tab.

1. In cell D2 enter a simple formula to add together the date value (B2) and the time value (C2). Do the same in D3 for the date and time on this row.
2. Make sure D2 and D3 are formatted to display both date and time, and you should see the combined values in these cells. For comparison, the corresponding date and time values are shown in cells E2:G3.
3. In D4 enter a simple subtraction to find the difference between the two date-times in D2 and D3, but make sure the result is formatted as a time duration. This gives the time between the two date-times, even when it spans several days.
4. In D5 convert this duration into a decimal time in hours (multiply by 24 - have you figured out why this works yet?).
5. In cell A9 enter the function, **=Now()**. This displays the current date and time, based on the computer's internal calendar and clock. As it is a function, it doesn't 'tick' automatically, but whenever the sheet is recalculated (which happens every time you make a change) it will update. Cell D9 is set equal to A9, but has been formatted differently. As the value is a date-time, you can apply formatting to display any part of it you wish, from just the seconds to the entire date and time. Experiment with this using custom formatting.

Timesheet example

Open the file **Timesheet**

Switch to the tab **TimeSheet**. This is intended to keep a record of the time spent on a job, for which an hourly rate is paid; a 30min lunch break must be allowed for, if 5 or more hours are worked, and for very small jobs there is a minimum charge of £50 (see F1, J1 and J2). The date and start/end times are entered in column A, C and D.

1. First create any named ranges you think you will need, particularly for single cells that would otherwise mean needing to use dollar notation in formulae. Add more as you go along if needed.
2. Edit the data area of the sheet (ie from row 4) as follows:
    a. Column B: Set equal to the date in column A, but format to display the day of the week.
    b. Columns C & D: Format to display 24hr time without seconds.
    c. Column E: Insert a simple subtraction to calculate the time worked, and format this appropriately.

d. Column F: Use an IF function to apply the break length if the number of hours worked exceeds the maximum continuous working time permitted - otherwise, this should be zero.
e. Column G: Insert a simple subtraction so the break time is subtracted from the hours worked.

3. Additional calculations:
    a. In cell J4 find the total chargeable hours (the total of column G).
    b. The calculated value in cell J5 should be the charge for 'J4' hours, charged at 'J2' per hour. A simple multiplication, however, will give a time duration, not a figure that can be displayed as currency - try it and you'll see the problem.
    To get round this:
    c. In cell L4 convert the total hours (J4) to a decimal value (multiply by 24 - figured out yet why this works?).
    d. Calculate the time charge in cell J5 from this decimal value.
    e. Finally, in cell J6 pop in an IF function so if the time charge is less than the minimum charge, it will be set to the value of the minimum charge.
4. If this was 'for real', you would now want to test it to ensure the conditionals work correctly:
    a. Try clearing the contents of the data entry cells (those bordered in blue) in rows 5, 6 and 7, leaving the formulae intact.
    b. Change the End time in cell D4 to 10:00 - this should give a total below the minimum charge, checking the conditional in J6.

Note: This is, of course, a very simply attempt at a time sheet, mainly designed to help you understand how to work with times. In practice you would almost certainly want to put the 'processing' and key data (hourly rate etc) on another tab. The approach used here also has the disadvantage that each 'job' would need a separate file, or one file with an awful lot of similar sheets.

An alternative solution would be to record all jobs on one sheet, using a unique identifier for each job, and interrogate this data to extract the values for each separate job. If using Google sheets, the data could even be entered using a Form, removing the need to access the spreadsheet to record daily figures.

Exercises T3 – Working with data sets

Validation

In order to reduce errors you set validation tests on entered data. This cannot show if something is correct, but it can show if something is 'wrong' or 'needs attention' by virtue of not meeting specified criteria.

Open **Expenses v2**, another version of the expenses exercise used earlier, which we'll use to illustrate a few different types of validation. The **config** tab is going to include some data used in the validation.

In each case we'll apply the validation to the whole column, to allow for extra data, but you may wish to remove it from the top row of headers.

1. **Date** column - as this is for expenses, the entered date cannot be later than today. Set the validation to allow only dates, and specify they must be earlier than, or equal to, today using **=NOW()** for 'today'. Reject the input if the date is invalid, and write a suitable error message. Try entering a date later than today to check if it works.
2. **Staff** column - To make sure the name is always spelt the same you can provide a drop-down list.
    a. The list of staff is on the config sheet, at the top of column B. You could make this column a named range[*].
    b. Once the list is made, return to the **Expenses** tab and add a drop-down list validation to this column, using the names list on config as the source of the list. Configure it so values not in the list may also be entered.
3. **Category** column - this could be another drop-down list, but as you wouldn't want staff inventing extra categories, it should be configured so only values in the list are allowed. Start by creating the list of valid categories on the config sheet (column D).
4. **Cost** column - there is already an IF function in col F to check the value entered in col E, but the amount can also be validated in the column itself. The value above which entries need checking is defined in **config** F1. Use this value when setting up validation which will alert the user if the amount is above the limit, but will still accept the entry.

[*] When using Google sheets, you can auto-generate the list from **Expenses** col B using the UNIQUE function. This means another name can be added on the live sheet and will automatically also appear in the dropdown. To do this, in **config** B1 you would enter:

```
=unique(Expenses!B2:B)
```

Sorting and Filtering

Open the file **Sort and Filter**

(**Note:** the bracketed figures at the end of each task indicate which records should be at the top and bottom of the list if the sort is correct, and for Q4 onwards may also indicate the number of records that should be obtained)

1. The **StudentList** worksheet lists marks for module exams. To help with this exercise, students also donate a sum of money to a homeless charity. The list of students is currently sorted in **Student Number** order. Try the following sorts:
    a. Sort by **Final Mark**, highest mark first. (10014,10002)
    b. Re-sort by amount donated, lowest amount first. (10029,10034)
    c. Re-sort the list alphabetically by **Surname** and **Forename** (so that those with the same surname are sorted by forename). (10009,10029)
    d. Return the list to its original sort order, by **Student Number**. (10001,10036)
2. Filter to display the following – *clear the filter after each one*:
    a. Students who are studying **Chocolate** as their module subject (8)
    b. 3rd year students who have scored a mark of **60** or more (5)
    c. 2nd and 3rd year students who have donated less than **£10** (13)
    d. People with surnames starting with *Sm* (eg Smith, Smythe, Smithers etc) (3)
3. Use a combination of sort and filter to generate a list of Year 1 students with final marks in descending order. (12, 10003, 10002)
4. Produce a list of all students taking the Chocolate module, sorted alphabetically by surname and forename. (8, 10014,10029)
5. Use sorting and filtering to produce a list of students in second and third years who donated at least £5, sorted with the highest donation at the top (20, 10034, 10035).
6. Generate a list of year 2 and 3 students (in full alphabetical order) where the Final Mark is ***not*** in the range 40-70 inclusive. (8, 10014, 10029)

Note: You will see that cell K1 counts the number of records in the list. You can't do this with a simple COUNT function as this does not take into account filtering. To allow for filtering, use the SUBTOTAL function instead. The operation is determined by the first argument, where:

1 is AVERAGE

2 is COUNT

3 is COUNTA

4 is MAX

5 is MIN

9 is SUM

Collaborative Sort and Filter

The methods in the exercises above would change the view of data for all concurrent users. Google sheets includes extra features and functions to enable more than one person to work on the same data set at the same time.

This exercise can only be done using Google sheets as the feature is not available in Excel.

To get the most of the first part of this exercise, find a friend, share your copy of **Sort and Filter**, and work together in this file. If you don't have any friends, open the file in another browser window so you have it open twice. You can then pretend you have a friend.

1. Work on the **StudentList** tab. First, one of you should apply a simple sort or filter, as above, so you can see how this affects the view for both of you. Not very helpful - remove any filters.
2. Now both of you should create new *Filter Views:*
    a. Person 1 - Show all the first years taking the **Cake** module, with marks sorted in order, highest first. Save the filter view as *1st years - Cake*.
    b. Person 2 - Show all students taking the **Pies** module, in alphabetical order. Save the filter view as *All years - Pies*.
3. Both of you now switch out of the filter view and check the new views have been saved in **Data > Filter views…**
4. Check it's working correctly:
    a. Person 1 - View person 2's filter view.
    b. Person 2 - Check the source data list is unaffected.
    c. Person 1 - exit filter view.
    d. Now swap roles and repeat.

This second option uses functions to do the sort and filter on another sheet, leaving the source data intact. You don't need a friend for this part.

5. Switch to the **Sort** tab:
   In cell A2 (leaving space for a header row), enter the **SORT** function so as to list the data sorted by exam mark, highest first. You will find that if the data range argument includes the header row, this will be shown in the sorted list, so the range will have to start at row 2.
6. Try some further sorts using the sort function:
    a. Sorted by surname and forename, in alphabetical order.
    b. Sorted by module, then by years.
7. Switch to the **Filter** tab:
   In cell A2, enter the **FILTER** function so only students in year 3 are shown. The header row should not appear as it will never contain the correct criteria, however you will find that columns containing both text and numbers behave oddly when numerical criteria are used.
8. Try some further filters using the filter function:
    a. Show only students in year 1 who have passed.
    b. Show only students taking the Stews module who have a mark less than 40.

Sub-totals

This exercise can only be done using Excel as the feature is not included in Google Sheets.

If a list of data is sorted appropriately, the Subtotal feature can be used to insert additional rows containing totals, averages etc for grouped values.

Open **T3 - Subtotals.xlsx**

1. We'd like to find the total donated by each year group:
    a. Sort the list by **Year**.
    b. Use the subtotal feature to find the total donated by each year group.
    c. We'd also like to know the average mark for each year group - add this without removing the donation total.
2. We've also been asked to provide averages for each module.
    a. Remove the existing subtotals and re-sort the list by **Module**, with the students in each module in alphabetical order (if you can't figure that out, at least sort the list by Module).
    b. Add subtotals to show the average mark for each module.

Exercises T4 – Processing data sets

Lookups

Open the file **Lookup Functions**

This spreadsheet contains 2 tabs: **Modules** contains a list of course modules, with two columns containing coded data; the **Data** tab includes tables to interpret the codes used for department and college. We'll use *Lookup* functions on **Modules** to decode the dept and college IDs.

1. On the **Data** tab, create a named range **departments** for the department list, using columns A:B (naming entire columns makes it easier to add new departments).
2. Likewise, create a named range for the list of **colleges** in columns D:E of the **Data** tab.
3. Switch to the **Modules** tab, and insert two extra columns for the department and colleges names - it would make sense to insert these next to the relevant coded columns, so that the Department Name is col D and the College is col F. Enter suitable headers for these in row 1.
4. In row 2 of the new *Department name* column, enter a VLOOKUP function that will look up the Department name (in the *departments* named range) using the ID in column C.
5. Replicate this down the column.
6. Likewise, in row 2 of the new *College name* column, use VLOOKUP to decode the college ID to a college name, and copy this down the column.
7. Alcuin college has decided to change its name to *Albert* college. On the **Data** tab, cell E2, change ***Alcuin*** to ***Albert*** and note how this affects all instances on the **Modules** sheet. You can change it back if it bothers you.

Open the file **Students500**, which lists results and other data for 500 students on the **Students** tab.

8. Switch to the **Lookup** tab. The idea here is that details for an individual student can be found by simply entering their ID in B2[*].
   Switch back to **Students** and make the whole student data set into a named range, **studentList**, and to make replication of the lookups more straightforward also name cell B2 from the **Lookup** sheet.
9. Using these named ranges, enter a lookup into cell **Lookup** D2 that will display the correct forname.
10. If you've used named ranges you should now be able to copy this **Lookup** into the other cells (D3:D6 and B5), needing only to change the column reference in the function to make it display the appropriate value.
11. On the **Students** tab, column M will display the degree awarded based on the Result in col L.
    A table on the **config** tab defines the boundaries for the degree awarded. Start by defining this table as a named range.
12. In cell **Students** M2, enter a range lookup that will show the degree awarded.
13. Edit the file as necessary so that cell **Lookup** B6 will also display the correct degree awarded.

[*] Using this method is quicker than looking down a long list, and the alternative of using the 'Find' feature will require looking across several columns, increasing the possibility of reading data from the wrong row. It also allows you to extract just the data you need, and the sheet could be designed for printing if necessary. And it shows how a lookup works...

Queries

The QUERY function is available in Google Sheets, but not in Excel. It is part of a collection of features and functions that make Google Sheets better suited to collaborative working.

You can only do this in a Google Spreadsheet - the function is not available in Excel.

The QUERY function allows us to:

- Extract selected columns, rather than the whole set
- Define criteria to filter for values and ranges of values
- Define a sort order independent of the original data
- View a subset of the data without affecting the original data source

Use the Google Sheet version of the file **Students500**. In this exercise we'll construct query functions to extract data from the student list on the **Students** sheet.

This exercise needs a named range for the student list, so if you did not create one in the **Lookups** exercise above, do this now, making the whole student data set (on **Students**) into a named range called **studentList**.

1. Switch to the **Query** sheet, and in cell A4 enter this query function:
   =query(studentList,"select B,C,G,L",1)
   This should generate a full list of students, but showing only the 4 defined columns.
2. The result of the query has a live link to the underlying data. Try these:
   a. On the **Query** tab, attempt to edit someone's forename; this should generate an error in A4 as you are overwriting cells populated by the query.
   b. On the **Students** tab, edit someone's forename; when you view the **Query** tab again it should have changed to match the edit.
3. Edit the query to include the degree class as a fifth column:
   =query(studentList,"select B,C,G,L,**M**",1)
4. Edit the query function again to add a criterion, showing only those students in year 3:
   =query(studentList,"select B,C,G,L,M **where I = 3**",1)
   Note that a column can be used for criteria, even though it is not displayed.
5. Edit the function again to use a text-based criterion - this needs single quotes around criteria values (in this case the word 'Fail'):
   =query(studentList,"select B,C,G,L,M **where M = 'Fail'** ",1)
6. Construct a query that will show only 3rd years, but sorted by surname:
   =query(studentList,"select B,C,G,L,M **where I=3 Order By C Asc**",1)
7. … and also sorted by forename:
   =query(studentList,"select B,C,G,L,M where I=3 **Order By C Asc, B Asc**",1)
8. Construct a query to combine the two criteria and the two-column sorting:
   =query(studentList,"select B,C,G,L,M **where I=3 And M='Fail' Order By C Asc**",1)

Extension: Criteria external to the function

Criteria do not have to be 'hard-coded' into the function; instead, values in other cells can be referenced. The syntax is different for numbers and text, as text must be enclosed in single quotes, but numbers must not.

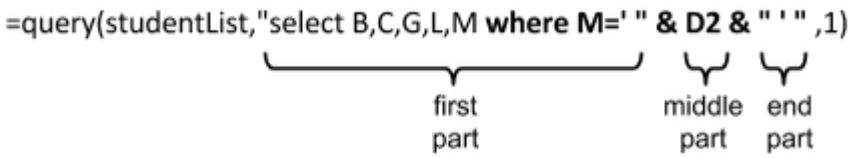
1. ***Numbers***. In **Query** C1 enter the label *Year* and in cell D1 enter *1* as the year.
2. Enter a query that shows the same 5 columns for year 1 students, but references D1:

   =query(studentList,"select B,C,G,L,M **where I=" & D1** ,1)

   Note that as the criteria value does not need enclosing in quotes we close the query text and concatenate the cell reference using ampersand.
3. Try changing the year in D1 - the result of the query function should also change.
4. ***Text***. In **Query** C2 enter the label *Degree* and in cell D2 enter *Fail* as the degree class.
5. Enter a query that shows the same 5 columns for all students with Fail as the degree class:

   =query(studentList,"select B,C,G,L,M where M=' " & D2 & " ' " ,1)

   As the criterion needs to be enclosed in single quotes, which must be inside double quotes, the query text is constructed by concatenating three parts together, the middle one being the referenced cell.



6. Try changing the degree in D2 from *Fail* to *U-2nd*.
7. This query function will combine the year and degree class queries (it has 5 sections):

   =query(studentList,"select B,C,G,L,M w**here I=" & D1 & " And M=' " & D2 & " ' "** ,1)

Pivot Tables

Open the file **NewPivot**. The **data** tab is a list of events undertaken by students to raise money for charity, with most students undertaking 3 activities. Note that each activity is recorded on a separate row, duplicating some data - this is to be expected in 'pivotable' data sets.

Create pivot tables to:

1. Display the total sum raised by each **Activity** (as row labels), grouped by **Year** (as column headers).
    a. Add a filter for the whole pivot table so the totals can be shown for specific **Colleges**.
    b. Configure to display totals for years but not activities.
2. Display the total sum raised by each **Activity**, grouped by **Colleges**, with the Activities and Colleges in alphabetical order.
    a. Add a filter for the whole pivot table so the totals can be shown for specific **year** groups.
3. Display a count of how many students took part in each **Activity** (row labels), grouped by **colleges**, and with a filter so figures for **year** groups can be viewed.
4. Show a small table of the total sum raised by each **activity**, which can be filtered by **year** group.
5. Display the sum raised by each student (row labels), including **Student ID**, **Forename** and **Surname**, grouped by **Activity** (column labels), with the option of filtering to display data for the different **colleges**.
   Ensure that subtotals are not shown for ID and Forename.
6. Show the average sum raised by each college, grouped by years.
   Ensure the averages are displayed to a sensible number of decimal places.