FIT5196 – Data Wrangling

Assignment Two

Michael McKay
Student ID: 32270208

Submitted:
14-Jun-2021

Part 1:

Data source for this assignment was a csv file called 'dataset1_with_error.csv'. The first step was to import it into Jupyter notebook as a dataframe using the pandas module. Shape of the datashape was 55169 rows by 11 columns. First 10 rows appeared as below:

	Id	Title	Location	Company	ContractType	ContractTime	Category	Salary	OpenDate	CloseDate	Source
0	12612628	Engineering Systems Analyst	Dorking	Gregory Martin International	NaN	permanent	Engineering Jobs	25000	20130708T120000	20130906T120000	cv-library.co.uk
1	12612830	Stress Engineer Glasgow	Glasgow	Gregory Martin International	NaN	permanent	Engineering Jobs	30000	20120130T000000	20120330T000000	cv-library.co.uk
2	12612844	Modelling and simulation analyst	Hampshire	Gregory Martin International	NaN	permanent	Engineering Jobs	30000	20121221T150000	20130120T150000	cv-library.co.uk
3	12613049	Engineering Systems Analyst / Mathematical Mod...	Surrey	Gregory Martin International	NaN	permanent	Engineering Jobs	27500	20131208T150000	20140206T150000	cv-library.co.uk
4	12613647	Pioneer, Miser Engineering Systems Analyst	Surrey	Gregory Martin International	NaN	permanent	Engineering Jobs	25000	20130302T120000	20130501T120000	cv-library.co.uk
5	19047429	Trainee Mortgage Advisor East Midlands	East Midlands	Brite Recruitment	NaN	permanent	Accounting & Finance Jobs	21000	20130103T000000	20130403T000000	cv-library.co.uk
6	20199757	PROJECT ENGINEER, PHARMACEUTICAL	Witney	MatchBox Recruiting Ltd	NaN	permanent	Healthcare & Nursing Jobs	37500	20120412T150000	20120611T150000	cv-library.co.uk
7	20797143	Chef de Partie Award Winning Restaurant Exce...	Derby	Chef Results	-	-	Hospitality & Catering Jobs	16000	20130328T120000	20130527T120000	caterer.com
8	22579462	Quality Engineer	Gateshead	Asset Appointments	NaN	permanent	Engineering Jobs	22000	20131222T150000	20140220T150000	cv-library.co.uk
9	22933091	Chef de Partie Award Winning Dining Live In ...	UK	Chef Results	-	-	Hospitality & Catering Jobs	18000	20131219T000000	20140102T000000	caterer.com

Next, the column information was summarised using the info command. Details are below:

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 55169 entries, 0 to 55168
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype 
---  ------        --------------  ----- 
 0   Id            55169 non-null  int64 
 1   Title         55169 non-null  object
 2   Location      55169 non-null  object
 3   Company       51320 non-null  object
 4   ContractType  33493 non-null  object
 5   ContractTime  47047 non-null  object
 6   Category      55169 non-null  object
 7   Salary        53584 non-null  object
 8   OpenDate      55169 non-null  object
 9   CloseDate     55169 non-null  object
 10  Source        55169 non-null  object
dtypes: int64(1), object(10)
memory usage: 4.6+ MB
None
```

Dataframe consists of 11 columns. First column is the ID, which appears to be a unique identifier for each entry in the dataframe. There are 55169 entries in this column, which matches the total number of rows in the dataframe, so we can assume there are no null values. It was also determined that there are 55169 unique entries in this field, so we can also assume no duplicates.

Next we will investigate the other columns. All other columns are currently objects, so we can view a summary using the command csv_data.describe(include=['O']). The summary appears as below.

	Title	Location	Company	ContractType	ContractTime	Category	Salary	OpenDate	CloseDate	Source
count	55169	55169	51320	33493	47047	55169	53584	55169	55169	55169
unique	55166	489	9064	4	4	8	3757	2194	2418	106
top	Senior Financial Advisor	UK			permanent	IT Jobs	35000	20120415T150000	20131208T000000	totaljobs.com
freq	2	8397	1133	14902	33637	14344	1865	45	45	10102

The title column has 55166 unique values out of 55169 total values. As the total number of values matches the total number of rows, we can assume there are no null values in this array. We would be expected many unique values though, as the title field is unique to each job been advertised and the way it manually entered by the person creating the job advertisement. Different people would have different ways of phrasing the job title. The fact that there are three values not unique mean we could have a couple of duplicate entries in this table. We should check what the non-unique values are. We can do this using the command csv_data.duplicated(['Table']).

```
csv_data.duplicated(['Title'])
```

```
0        False
1        False
2        False
3        False
4        False
         ...
55164    False
55165     True
55166     True
55167    False
55168     True
Length: 55169, dtype: bool
```

This shows that rows 55165, 55166 and 55168 are the values with non-unique values. The six entries appear as below:

	Id	Title	Location	Company	ContractType	ContractTime	Category	Salary	OpenDate	CloseDate	Source
47471	71808610	Pensions Administrators (Temporary/Contract)	UK	Abenefit2u	-	contract	Accounting & Finance Jobs	24000	20130801T150000	20130831T150000	professionalpensionsjobs.com
55165	72705205	Pensions Administrators (Temporary/Contract)	UK	Abenefit2u	NaN	contract	Accounting & Finance Jobs	24000	20130801T150000	20130831T150000	cv-library.co.uk
8023	67290277	Quality Assurance Environmental Manager Nottin...	Nottingham	Stephen James Consulting	NaN	permanent	Healthcare & Nursing Jobs	35000.0	20120110T150000	20120409T150000	tntjobs.co.uk
55168	72705244	Quality Assurance Environmental Manager Nottin...	Nottingham	Stephen James Consulting	NaN	permanent	Healthcare & Nursing Jobs	35000.0	20120110T150000	20120409T150000	tntjobs.co.uk
34675	70086310	Senior Financial Advisor	London	Fram Executive Search	-	permanent	Accounting & Finance Jobs	40000	20130126T000000	20130225T000000	ifaonlinejobs.co.uk
55166	72705221	Senior Financial Advisor	London	Fram Executive Search.	-	permanent	Accounting & Finance Jobs	40000	20130126T000000	20130225T000000	ifaonlinejobs.co.uk

They appear to be the same job listed twice. The Quality Assurance Environmental Manager and the Senior Financial Advisor positions are duplicate values from within the same data source, but the Pensions Administrators position is the same job listed from two different sources. I will now drop these duplicate values from the dataFrame using the drop_duplicates command. DataFrame now appears as below:

```
(55166, 11)
```

	Title	Location	Company	ContractType	ContractTime	Category	Salary
count	55166	55166	51317	33492	47044	55166	53581
unique	55166	489	9063	4	4	8	3757
top	Underwriter Mechanical Breakdown Extended War...	UK			permanent	IT Jobs	35000
freq	1	8396	1133	14902	33635	14344	1865

We now have 3 less rows in the table, and every value in the Title column is now unique.

Next, we will look at the Location column. This column now has 55166 values, so we can assume there are no null values. It has 489 unique values, which we'd expect as there should be many entries per location. Let's do a value count of the location field now:

```
UK                  8397
London              7046
South East London   2961
The City            1184
Central London       889
                    ...
Manchaster             1
BRISTOL                1
Livepool               1
Leads                  1
Cembridge              1
Name: Location, Length: 489, dtype: int64
```

We have quite a few fields where the count is 1. It looks like the location names have been spelt incorrectly for these entries. We can fix these by doing a high frequency to low frequency comparison, as the entries spelt correctly should appear many times, and the entries spelt incorrectly only appear once or twice. First, we will do some cleaning of the data to make it easier for the high frequency to low frequency comparison to work. We'll convert all strings to upper case, remove any special characters except space and full stop and then remove any multiple spaces. After that we'll run the match_highfreq_To_lowfreq function with a match factor of 0.8 and with 1 entry been considered a low frequency entry:

```
Number of lowfreq_data 4
Number of highfreq_data 480
CEMBRIDGE : CAMBRIDGE 0.8888888888888888
LEADS : LEEDS 0.8
LIVEPOOL : LIVERPOOL 0.9411764705882353
MANCHASTER : MANCHESTER 0.9
MANCHASTER : LANCASTER 0.8421052631578947
```

BRISTOL is no longer appearing as a low frequency item. This must have been fixed when we converted all our entries to upper case. Cembridge, Leads, Livepool and Manchaster all look like spelling mistakes. If we were to accept these matches the function would replace Manchaster with Lancaster, which doesn't look right. We can get around this by running the match_highfreq_To_lowfreq function with a factor of 0.9, updating and then running again with a factor of 0.8.

After this we will convert the location names back to capitalized values using the 'capitalizedLocation' function, and then do a value_counts to see if we've removed all the low frequency items.

```
UK                  8397
London              7048
South East London   2961
The City            1184
Central London       889
                    ...
St. Neots            10
South Brent          10
City                 10
North Finchley        7
Oxfords               2
Name: temp_Location, Length: 480, dtype: int64
```

We still have one low frequency item 'Oxfords'. This is a location name which has been falsely pluralised. We can fix this by replacing all instances of Oxfords with Oxford in the file. Also, there are 1184 entries of 'The City' and 10 of 'City'. These location names are the same and should be normalised as well. In this case 'the' is a stop word which adds little value to the location name. We can modify out 'normaliseLocation' to remove the word 'the' and then run it again. Now our value counts of suburbs appear as below.

```
UK                  8396
London              7047
South East London   2961
City                1194
Central London       889
                    ...
Gatwick              11
St. Neots            10
Thorpe St. Andrew    10
South Brent          10
North Finchley        7
Name: temp_Location, Length: 478, dtype: int64
```

Now all location names have been normalised we can move onto the company name. First, we will run a normaliseCompany function which will conver to upper case, remove any special characters, change all forms of limited to ltd and change 'AND' to '&'. Then do a match high frequency to low frequency with a match factor of 0.95.

The list of matches shown in the jupter notebook below step 18 shows all entries which appear once but are like another entry which appears more often. There are a lot of false positives in this list unfortunately, and the highest false positive appears with a match factor of 0.975. We shouldn't accept these changes as falsely changing correct data to something incorrect is much worse than having a handful of entries out of 8540 companies which are slightly different.

We can see there are some common spelling mistakes in the above data though, and correct them using the 'spellFixCompany' function, and then run the match function again with a match factor of 0.98, where we know we won't get any false positives.

```
Number of lowfreq_data 6576
Number of highfreq_data 1942
ANDERSON WRIGHT CONSULTING  : ANDERSON WRIGHT CONSULTING 0.9811320754716981
PENRHYN WILLIAMS RECRUITMENT  : PENRHYN WILLIAMS RECRUITMENT 0.9824561403508771
KENNEDY PEARCE CONSULTING  : KENNEDY PEARCE CONSULTING 0.9803921568627451
RANDSTAD FINANCIAL & PROFESSIONAL  : RANDSTAD FINANCIAL & PROFESSIONAL 0.9850746268656716
AMBRIDGE HARRIS ASSOCIATES  : AMBRIDGE HARRIS ASSOCIATES 0.9811320754716981
HW ASSOCIATES RECRUITMENT  : HWA ASSOCIATES RECRUITMENT  0.9811320754716981
XCEDE RECRUITMENT SOLUTIONS  : XCEDE RECRUITMENT SOLUTIONS 0.9818181818181818
MERIDIAN BUSINESS SUPPORT  : MERIDIAN BUSINESS SUPPORT 0.9803921568627451
MERROW LANGUAGE RECRUITMENT  : MERROW LANGUAGE RECRUITMENT 0.9818181818181818
IMPACT CREATIVE RECRUITMENT  : IMPACT CREATIVE RECRUITMENT 0.9818181818181818
BERKELEY SCOTT CONTRACT CATERING FACILITIES MANAGEMENT : BERKELEY SCOTT CONTRACT CATERING & FACILITIES MANAGEMENT 0
.9818181818181818
REDOAK RECRUITMENT SOLUTIONS  : REDOAK RECRUITMENT SOLUTIONS 0.9824561403508771
REED BUSINESS INFORMATION  : REED BUSINESS INFORMATION 0.9803921568627451
BIG RED RECRUITMENT MIDLANDS  : BIG RED RECRUITMENT MIDLANDS 0.9824561403508771
MORTIMER BELL INTERNATIONAL  : MORTIMER BELL INTERNATIONAL 0.9818181818181818
RECRUITMENT REVOLUTION.COM  : RECRUITMENTREVOLUTION.COM  0.9811320754716981
CEEMA TECHNOLOGY RECRUITMENT : CEEMA TECHNOLOGY RECRUITMENT  0.9824561403508771
ELECTUS RECRUITMENT SOLUTIONS  : ELECTUS RECRUITMENT SOLUTIONS 0.9830508474576272
ANGLO TECHNICAL RECRUITMENT  : ANGLO TECHNICAL RECRUITMENT 0.9818181818181818
DATASOURCE COMPUTER EMPLOYMENT : DATASOURCE COMPUTER EMPLOYMENT  0.9836065573770492
EURO PROJECTS RECRUITMENT  : EURO PROJECTS RECRUITMENT 0.9803921568627451
BEEBY ANDERSON RECRUITMENT : BEEBY ANDERSON RECRUITMENT  0.9811320754716981
INTERNAL AUDIT CONNECTIONS : INTERNAL AUDIT CONNECTIONS  0.9811320754716981
FUTURE ENGINEERING RECRUITMENT : FUTURE ENGINEERING RECRUITMENT  0.9836065573770492
PEARSON WHIFFIN RECRUITMENT : PEARSON WHIFFIN RECRUITMENT  0.9818181818181818
TASTE HOSPITALITY RECRUITMENT : TASTE HOSPITALITY RECRUITMENT  0.9830508474576272
GRAVITAS RECRUITMENT GROUP : GRAVITAS RECRUITMENT GROUP  0.9811320754716981
REGIONAL RECRUITMENT SERVICES  : REGIONAL RECRUITMENT SERVICES 0.9830508474576272
```

Of this list of entries, the only change I don't want to accept is for HW Associates, as a google search has determined this is a distinct company from HWA associates. We can remove then entry from the match_data set, then update the csv_data['temp_Company'] field. Lastly, we will capitilise the first letter of each word to neaten up the appearance of the company names.

The number of unique company names has now dropped from 9064 to 8516. But I was also been cautious when updating to ensure no company names were falsely changed.

The next column to look at is the ContractType and the ContractTime columns.

```
             14902
full_time    12303
-             4719
part_time     1568
Name: ContractType, dtype: int64
```

```
permanent    33635
-             6249
contract      6088
              1072
Name: ContractTime, dtype: int64
```

Null values are either represented as a '-' or as ''. To make things consistent, we'll change everything to a ''. But there are a lot of null values for both columns. We may be able to get information regarding the contract type and contract times from the Title column. We can do this by using regular expressions to see if the terms full time, part time, fixed term and permanent appear in the title. The process is shown in the functions extractContractType and extractContractTime.

We the combine the extracted Contract time and Contract type with the original contract time and contract type but we give priority to the new extracted columns. The process is shown in the function 'combineContractColumn'. I've assumed that what is in the title is correct, as it displays how the job is been advertised to whoever is viewing the ad. The original contract time, type is a field used for filtering the ad and a mistake could easily be made here when the ad is been put online.

Next, we look at the Category field using the value_counts command.

```
IT Jobs                          14344
Healthcare & Nursing Jobs         8808
Engineering Jobs                  8210
Accounting & Finance Jobs         7136
Sales Jobs                        5349
Hospitality & Catering Jobs       4788
Teaching Jobs                     3779
PR, Advertising & Marketing Jobs  2752
Name: Category, dtype: int64
```

This looks fine, there don't seem to be any double ups on categories.

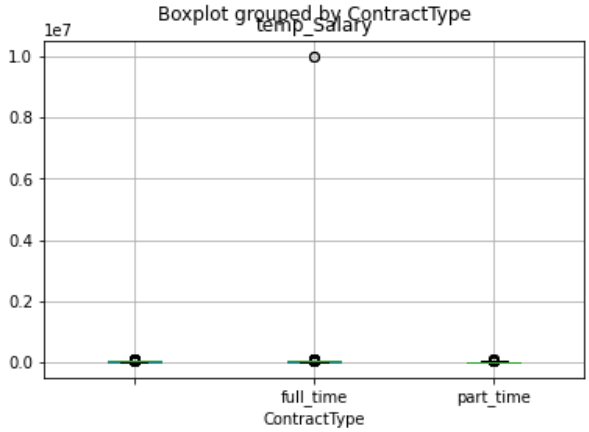
Next is the Salary column. The salary column current exists as an object column. Which means it couldn't be converted to a float or integer column because there were non-numeric characters present. First, we'll use a regular expression to find all non-numeric characters present in the Salary column.

```
# Check for non-numeric characters in the salary column.

csv_data_Sal_list = list(csv_data['Salary'])

nonNumList = []

for entry in csv_data_Sal_list:

    if type(entry) == str:
        if (re.search(r"[\D]+", entry)):
            match = re.search(r"[\D]+", entry)
            nonNumList.append(match.group())

print(set(nonNumList))
```

```
{'K', ' per annum', ' - ', ' to ', ' per Annum', ' To ', '/year', '-', '/Year', '.', ' pa'}
```

So it looks like the issue here is that Salaries are present in all sorts of units. We will need a function that will convert all of these to annual salaries. This has been written with the function normalizeSalaries. It assumes that all hourly paying jobs work 40 hours a week and 52 weeks a year. It will also average the two values given if Salary is given as a range.

We should further investigate the Salary column to look for outliers. We can do this using a boxplot of salary grouped by ContractType.



We can see there is something throwing our boxplot off. There are some entries where the salary is 10,000,000 pounds per year. This seems unusually high. We can look these entries up.

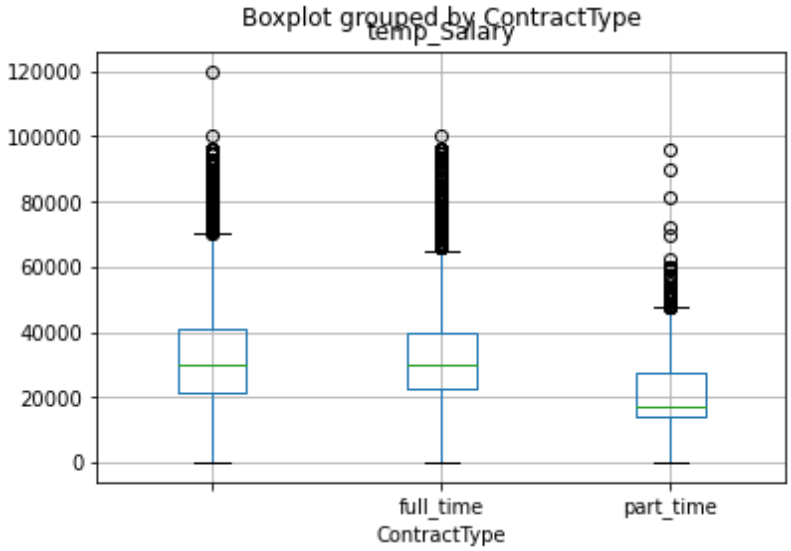
```
csv_data[csv_data['temp_Salary'] > 1000000]
```

Out[38]:

	Id	Title	Category	Salary	OpenDate	CloseDate	Source	Location	Company	ContractTime	ContractTyp
26074	60181124	Office Technical ...trator	Engineering Jobs	10000000	20130325T120000	20130424T120000	workthing.com	Cheshire	Group Cytek	permanent	full_tim
48506	71869302	New Business Development Exec/Manager	Teaching Jobs	10000000	20131122T120000	20140220T120000	jobs.guardian.co.uk	Central London	Periscope Uk Ltd	permanent	full_tim

click to scroll output; double click to hide

These look like mistakes. We should try plotting without these entries included. Box plot appears as below:

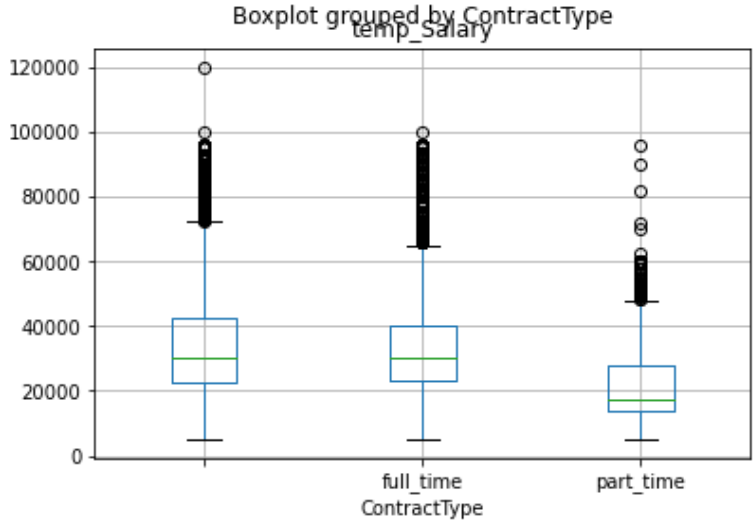


This looks better, but there appear to be some jobs where Salary is 0 pounds per year. This shouldn't be. We can do a search for these positions.

	Id	Title	Category	Salary	OpenDate	CloseDate	Source	Location	Company	ContractTime	Contract
1354	55408278	Software Engineer, C++, MFC, STL ****k East ...	IT Jobs	0	20130824T150000	20131122T150000	planetrecruit.com	West Sussex	Spectrum It Recruitment	permanent	full_
1379	55408791	Senior IT Project Manager Group IS Development	IT Jobs	0	20120103T150000	20120202T150000	planetrecruit.com	Cambridgeshire	Jobg8	permanent	full_
1405	55409302	Cisco Channel Sales Manager Wireless, Voice, ...	IT Jobs	0	20130205T000000	20130406T000000	planetrecruit.com	Surrey	Palm It Services	permanent	full_
1411	55409391	Software Developer C, ASPNet **** to 6 Mth C...	IT Jobs	0	20121208T000000	20130308T000000	planetrecruit.com	South Lanarkshire	Abrecco	contract	full_
1413	55409436	Junior SQL Developer	IT Jobs	0	20130126T120000	20130426T120000	planetrecruit.com	Surrey	Jobg8	permanent	full_
...	...	...	...	...	...	...	...	...	...	...	
54603	72675839	Sales Manager / Engineering / West Midlands	Sales Jobs	0	20120515T000000	20120714T000000	totaljobs.com	UK	Hiredonline	permanent	
54606	72675881	NPI Manufacturing Engineer / Manufacturing Pro...	Engineering Jobs	0	20131224T120000	20140123T120000	totaljobs.com	Chichester	Trs Consulting	permanent	
55137	72697286	Training Delivery Instructor	Engineering Jobs	0	20120202T000000	20120303T000000	justengineers.net	UK	Minstrell Recruitment Ltd	permanent	
55139	72697334	Experienced Tree Surgeon	Engineering Jobs	0	20120726T000000	20120924T000000	justengineers.net	Poole	R S S Ltd	permanent	
55143	72702355	Senior Operations Manager PPI	Accounting & Finance Jobs	0	20120510T150000	20120709T150000	hays.co.uk	Middlesbrough		contract	

773 rows × 12 columns

These aren't volunteer jobs. We can tell by looking at the title. We should remove the Salary entries which are 0 pounds and 10,000,000 pounds as these are outliers and try plotting our box plot again.



This looks better. The data makes sense. We'd expect the full time positions to be making more then the part time positions, and we'd also expect the average for the unknown positions to be somewhere in between.

Now we can delete the Salary column and rename the temp_Salary to Salary. The last column to check now is the source column.

```
totaljobs.com          10102
cv-library.co.uk        7840
jobsite.co.uk           3630
cwjobs.co.uk            3132
staffnurse.com          2778
                        ...
grb.uk.com                 5
scotsman.com               4
jobs.gponline.com          3
cvjobstore.com             1
thegraduate.co.uk          1
Name: Source, Length: 106, dtype: int64
```

This seems fine. There are a couple of entries with low frequency, but there don't appear to be any mistakes in how they're written. Now we can reorganise all the columns and move onto Part 2.

Part 2:

Firstly we need to open the new dataset to integrate and inspect.

	Opening	Closing	Job Title	Organisation	Location	Category	Salary per month	Fraction	Contract Type
0	2013-10-06 00:00:00	2013-12-05 00:00:00	Aviation loans administration	cer Financial	London	Finance and Accounting	2800	NaN	Contract
1	2012-10-03 12:00:00	2012-11-02 12:00:00	Payroll Analyst City upto **** , ****	LMA Recruitment Ltd	London	Finance and Accounting	2917	NaN	Permanent
2	2012-01-01 00:00:00	2012-01-31 00:00:00	Investment Team Assistant for leading Private ...	Austin Andrew Ltd	London	Finance and Accounting	3750	NaN	Permanent
3	2012-10-14 00:00:00	2012-11-13 00:00:00	SWAPS COLLATERAL CONTROL OFFICER	Brian Durham Recruitment Services Limited	City	Finance and Accounting	3333	NaN	Permanent
4	2012-11-17 12:00:00	2013-01-16 12:00:00	Loans Administration Temp	cer Financial	London	Finance and Accounting	3280	NaN	Contract
5	2012-11-01 12:00:00	2012-12-31 12:00:00	Oversight and Compliance Manager Global Custo...	Citifocus Limited	The City	Finance and Accounting	3750	NaN	Permanent
6	2012-03-22 00:00:00	2012-04-21 00:00:00	ALM Actuary	Reed Insurance	South East England	Finance and Accounting	6250	NaN	Permanent
7	2013-03-09 12:00:00	2013-05-08 12:00:00	SUPPORT ANALYST Front Office	Newside Consulting Limited	London	Finance and Accounting	4667	NaN	Permanent
8	2013-11-26 12:00:00	2013-12-26 12:00:00	Forex Sales Manager	LMA Recruitment Ltd	London	Finance and Accounting	4583	NaN	Permanent
9	2013-06-21 15:00:00	2013-09-19 15:00:00	Management Accountant Commodity Trading	LMA Recruitment Ltd	London	Finance and Accounting	4792	NaN	Permanent

We can see the opening and closing date columns are in a different format. The Salary is listed as per month instead of as per year. Category's may not be consistent, and a lot of the other columns may not contain consistent entries.

We can start renaming columns so that they match what was in the previous file. Job Title was changed to Title and Organisation was changed to Company. Salary per month was converted to Salary per year into a new column called 'Salary'. Contract Type was renamed to ContractTime and Fraction to ContractType.

Next the dates in OpenDate and CloseDate were reformatted to match what they are like in the previous task. They are now formatted as YYYYMMDDTHHMMSS.

A new column created for Id, which is currently blank. A new column was created for source, which is set to csv_dataset2. Columns were then reordered to match what's in the previous dataset.

Datatypes for ID and Salary were changed to float to match the previous dataset. Then we loaded the first dataset and checked what the highest ID number was from this set. It's 72705240. To be safe, we'll set the ID for the second dataset as an incremental number starting from 80000000.

The two datasets were then combined into one dataframe. We'll then look at the Categories to see if there is any double ups.

```
# Check what the category names are, and how they compare between datasets.
csv_dataset['Category'].unique()
```

```
array(['Engineering Jobs', 'Accounting & Finance Jobs',
       'Healthcare & Nursing Jobs', 'Hospitality & Catering Jobs',
       'IT Jobs', 'Sales Jobs', 'Teaching Jobs',
       'PR, Advertising & Marketing Jobs', 'Finance and Accounting',
       'PR, Advertising and Marketing', 'Information Technology'],
      dtype=object)
```

We do have some double up on categories. The function updateCategories will be used to remove redundant categories by renaming them to how they're named in the first dataset.

Next, we'll check the Contract type and time to see if there are also redundant categories.

```
# Check what the Contract Types are, and how they compare between datasets.
csv_dataset['ContractType'].unique()
```

```
array([nan, 'full_time', 'part_time', 'Part Time'], dtype=object)
```

Part time is listed as part_time and as Part time. The function updateContractType is used to rename all part time positions to part_time.

```
# Check what the Contract Times are, and how they compare between datasets.
csv_dataset['ContractTime'].unique()
```

```
array(['permanent', nan, 'contract', 'Contract', 'Permanent'],
      dtype=object)
```

There are similar issues with the ContractTime column. Contract and Permanent are written in two different ways. The function updateContractTime will rename them so they're all written in the same way as the dataset from Part 1.

Now finally we will check the dataset.info for the output dataFrame.

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 55500 entries, 0 to 333
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Id            55500 non-null  int64
 1   Title         55500 non-null  object
 2   Location      55500 non-null  object
 3   Company       50111 non-null  object
 4   ContractType  13955 non-null  object
 5   ContractTime  40081 non-null  object
 6   Category      55500 non-null  object
 7   Salary        52767 non-null  float64
 8   OpenDate      55500 non-null  object
 9   CloseDate     55500 non-null  object
 10  Source        55500 non-null  object
dtypes: float64(1), int64(1), object(9)
memory usage: 5.1+ MB
```

All columns required by the brief are present in the correct order and with the correct data type assigned to it.

Finally we will check for duplicates within the final dataset. The command .duplicated() is used to create a dataFrame dup_csv_data containing all duplicate data sorted by Title.

	Id	Title	Location	Company	ContractType	ContractTime	Category	Salary	OpenDate	CloseDate	
22757	68840315	Accounts Payable / Compliance / Administrator	London	Prime Personnel Services Ltd	NaN	permanent	Accounting & Finance Jobs	28500.0	20130106T000000	20130205T000000	jobs.catererandhotelk
155	80000155	Accounts Payable / Compliance / Administrator	London	Prime Personnel Services Ltd	NaN	permanent	Accounting & Finance Jobs	28500.0	20130106T000000	20130205T000000	cs
12469	68218723	Assistant Accountant Financial Services Experi...	London	Maldon Partners Ltd	NaN	permanent	Accounting & Finance Jobs	27500.0	20131105T000000	20140104T000000	cw
89	80000089	Assistant Accountant Financial Services Experi	London	Maldon Partners Ltd	NaN	permanent	Accounting & Finance Jobs	27504.0	20131105T000000	20140104T000000	cs

Values are very similar. The only difference is salary is slightly different for both entries. We'll keep the first value for these instances, as the salaries in the original first values are rounded to less significant figures and look more plausible.

Dataframe csv_dataset is now ready to export.