In [48]:
```
#prints stats for the numeric columns (int64)
mySample_cleaned1.select_dtypes(['int64', 'float64']).describe().T
```

Out[48]:

	count	mean	std	min	25%	50%
ExternalRiskEstimate	944.0	71.738347	11.051234	-9.0	65.0	72.0
MSinceOldestTradeOpen	944.0	195.483051	105.323487	-8.0	130.0	181.0
MSinceMostRecentTradeOpen	944.0	9.846398	15.241942	0.0	3.0	6.0
AverageMInFile	944.0	79.834746	35.913650	6.0	58.0	76.0
NumSatisfactoryTrades	944.0	21.611229	12.049576	1.0	13.0	20.0
PercentTradesNeverDelq	944.0	92.345339	11.425976	33.0	89.0	97.0
MaxDelq2PublicRecLast12M	944.0	5.733051	1.696709	0.0	5.0	6.0
MaxDelqEver	944.0	6.358051	1.893500	2.0	6.0	6.0
NumTotalTrades	944.0	23.024364	13.231940	0.0	14.0	21.0
NumTradesOpeninLast12M	944.0	1.954449	1.814003	0.0	1.0	2.0
PercentInstallTrades	944.0	33.490466	17.939391	0.0	21.0	32.0
NetFractionRevolvingBurden	944.0	33.582627	28.957598	-8.0	6.0	29.0
NetFractionInstallBurden	944.0	42.559322	42.095265	-8.0	-8.0	53.0
NumRevolvingTradesWBalance	944.0	3.880297	3.367469	-8.0	2.0	3.0
NumBank2NatlTradesWHighUtilization	944.0	0.599576	2.626773	-8.0	0.0	1.0
PercentTradesWBalance	944.0	65.698093	22.417161	-8.0	50.0	67.0

In [49]:
```
#average value for each numeric feature only
mySample_cleaned1.mean()
```

Out[49]:
```
ExternalRiskEstimate                   71.738347
MSinceOldestTradeOpen                 195.483051
MSinceMostRecentTradeOpen               9.846398
AverageMInFile                         79.834746
NumSatisfactoryTrades                  21.611229
PercentTradesNeverDelq                 92.345339
MaxDelq2PublicRecLast12M                5.733051
MaxDelqEver                             6.358051
NumTotalTrades                         23.024364
NumTradesOpeninLast12M                  1.954449
PercentInstallTrades                   33.490466
NetFractionRevolvingBurden             33.582627
NetFractionInstallBurden               42.559322
NumRevolvingTradesWBalance              3.880297
NumBank2NatlTradesWHighUtilization      0.599576
PercentTradesWBalance                  65.698093
dtype: float64
```

In [50]:
```
#standard deviation from the average for each numeric feature only
mySample_cleaned1.std()
```

Out[50]:
```
ExternalRiskEstimate                   11.051234
MSinceOldestTradeOpen                 105.323487
MSinceMostRecentTradeOpen              15.241942
AverageMInFile                         35.913650
NumSatisfactoryTrades                  12.049576
PercentTradesNeverDelq                 11.425976
MaxDelq2PublicRecLast12M                1.696709
MaxDelqEver                             1.893500
NumTotalTrades                         13.231940
NumTradesOpeninLast12M                  1.814003
PercentInstallTrades                   17.939391
NetFractionRevolvingBurden             28.957598
NetFractionInstallBurden               42.095265
NumRevolvingTradesWBalance              3.367469
NumBank2NatlTradesWHighUtilization      2.626773
PercentTradesWBalance                  22.417161
dtype: float64
```

In [51]:
```
#mid-way value for each numeric feature only
mySample_cleaned1.median()
```

Out[51]:
```
ExternalRiskEstimate                   72.0
MSinceOldestTradeOpen                 181.0
MSinceMostRecentTradeOpen               6.0
AverageMInFile                         76.0
NumSatisfactoryTrades                  20.0
PercentTradesNeverDelq                 97.0
MaxDelq2PublicRecLast12M                6.0
MaxDelqEver                             6.0
NumTotalTrades                         21.0
NumTradesOpeninLast12M                  2.0
PercentInstallTrades                   32.0
NetFractionRevolvingBurden             29.0
NetFractionInstallBurden               53.0
NumRevolvingTradesWBalance              3.0
NumBank2NatlTradesWHighUtilization      1.0
PercentTradesWBalance                  67.0
dtype: float64
```

In [52]:
```
#minimum value for each numeric feature only
mySample_cleaned1.min()
```

Out[52]:
```
RiskPerformance                              Bad
ExternalRiskEstimate                          -9
MSinceOldestTradeOpen                         -8
MSinceMostRecentTradeOpen                      0
AverageMInFile                                 6
NumSatisfactoryTrades                          1
PercentTradesNeverDelq                        33
MSinceMostRecentDelq                     2years+
MaxDelq2PublicRecLast12M                       0
MaxDelqEver                                    2
NumTotalTrades                                 0
NumTradesOpeninLast12M                         0
PercentInstallTrades                           0
MSinceMostRecentInqexcl7days         6-12months
NumInqLast6M                            OneOrTwo
NumInqLast6Mexcl7days                   OneOrTwo
NetFractionRevolvingBurden                    -8
NetFractionInstallBurden                      -8
NumRevolvingTradesWBalance                    -8
NumInstallTradesWBalance                       1
NumBank2NatlTradesWHighUtilization            -8
PercentTradesWBalance                         -8
dtype: object
```

In [53]:
```
#maximum value for each numeric feature only
mySample_cleaned1.max()
```

Out[53]:
```
RiskPerformance                          Good
ExternalRiskEstimate                       94
MSinceOldestTradeOpen                     589
MSinceMostRecentTradeOpen                 184
AverageMInFile                            273
NumSatisfactoryTrades                      74
PercentTradesNeverDelq                    100
MSinceMostRecentDelq                  Unknown
MaxDelq2PublicRecLast12M                    9
MaxDelqEver                                 8
NumTotalTrades                             77
NumTradesOpeninLast12M                     10
PercentInstallTrades                      100
MSinceMostRecentInqexcl7days          Unknown
NumInqLast6M                          unknown
NumInqLast6Mexcl7days                 unknown
NetFractionRevolvingBurden                115
NetFractionInstallBurden                  196
NumRevolvingTradesWBalance                 25
NumInstallTradesWBalance              unknown
NumBank2NatlTradesWHighUtilization         15
PercentTradesWBalance                     100
dtype: object
```

In [54]:
```
print("Feature       [Unique Values]")
for column in numeric_columns:
    print(column, [str(len(mySample_cleaned1[column].unique()))])
```

```
Feature       [Unique Values]
ExternalRiskEstimate ['48']
MSinceOldestTradeOpen ['347']
MSinceMostRecentTradeOpen ['59']
AverageMInFile ['165']
NumSatisfactoryTrades ['63']
PercentTradesNeverDelq ['48']
MaxDelq2PublicRecLast12M ['9']
MaxDelqEver ['7']
NumTotalTrades ['67']
NumTradesOpeninLast12M ['11']
PercentInstallTrades ['76']
NetFractionRevolvingBurden ['105']
NetFractionInstallBurden ['107']
NumRevolvingTradesWBalance ['21']
NumBank2NatlTradesWHighUtilization ['15']
PercentTradesWBalance ['80']
```

Prepare a table with descriptive statistics for all the categorical features

In [55]:
```
#keep only categorical features
categorical_columns = mySample_cleaned1.select_dtypes(['category']).columns
```

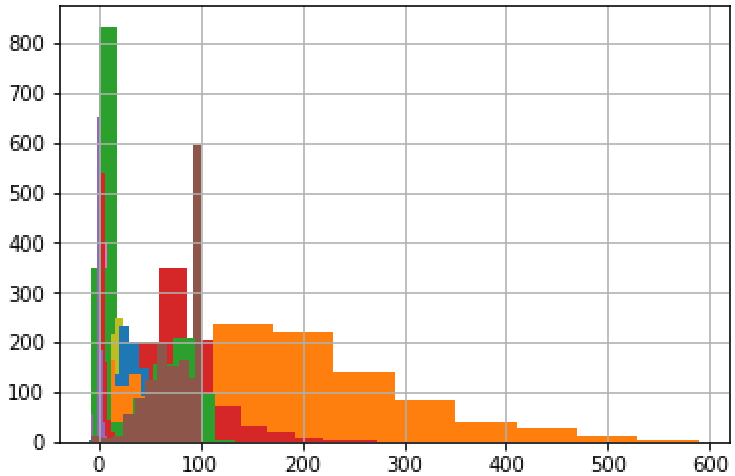
In [56]:
```
#prints stats for the categorical columns
mySample_cleaned1.select_dtypes(['category']).describe().T
```

Out[56]:

	count	unique	top	freq
NumTrades60Ever2DerogPubRec	307	4	Never	183
NumTrades90Ever2DerogPubRec	216	4	Never	145
MSinceMostRecentDelq	944	7	Unknown	460
MSinceMostRecentInqexcl7days	944	6	Never	431
NumInqLast6M	944	6	unknown	367
NumInqLast6Mexcl7days	944	6	unknown	374
NumInstallTradesWBalance	944	7	1	277

Plot histograms for all the continuous features

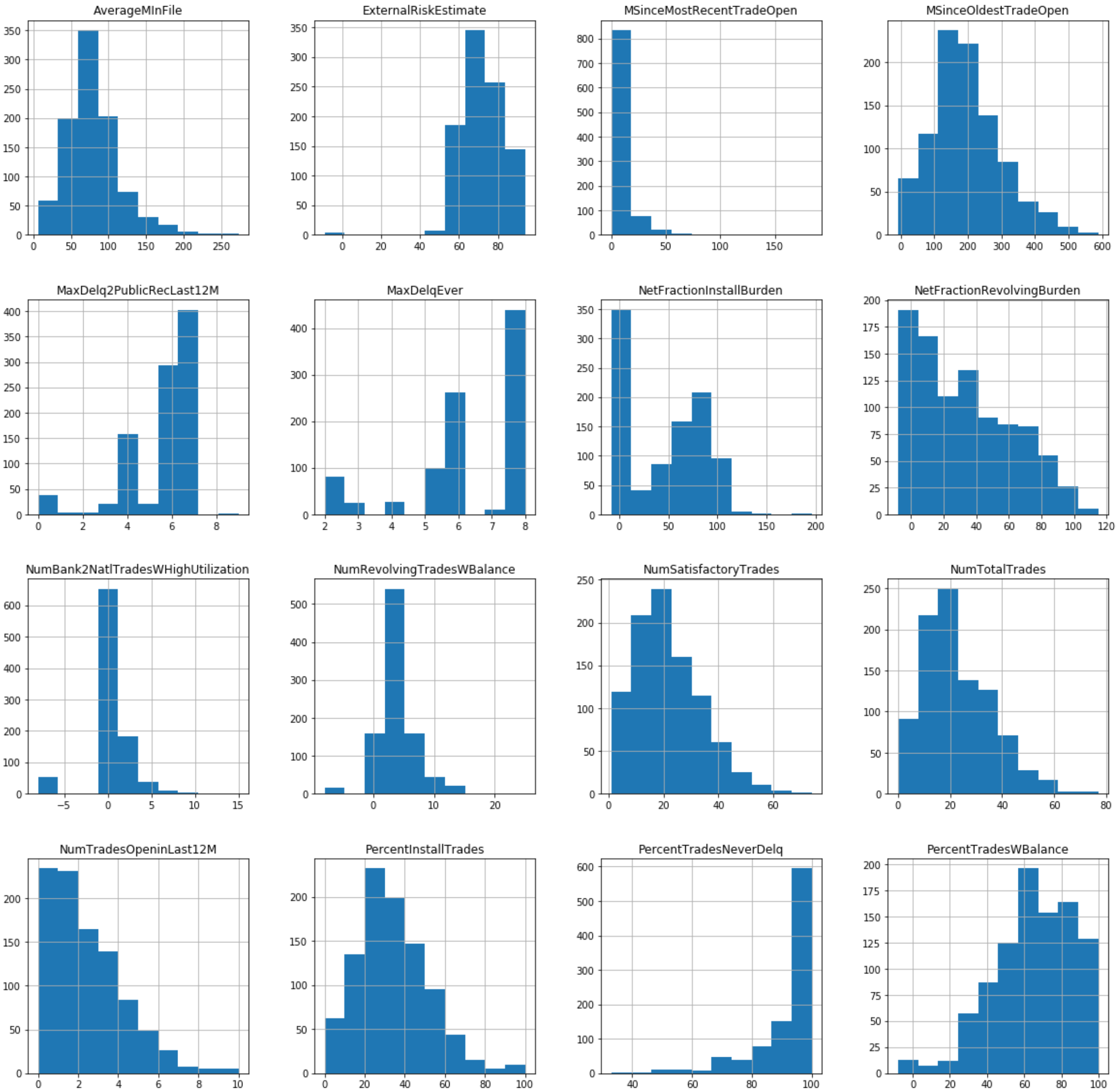
In [57]:
```
for column in numeric_columns:
    if mySample_cleaned1[column].dtype == 'int64' or mySample_cleaned1[column]
.dtype == 'float64':
        mySample_cleaned1[column].hist()
```



A bit too literal and crowded I think

In [58]:
```
# For visualisation/plotting
import matplotlib.pyplot as plt
%matplotlib inline

#Plots all numeric features at same time
plt.figure()
mySample_cleaned1.hist(figsize=(20, 20))
```

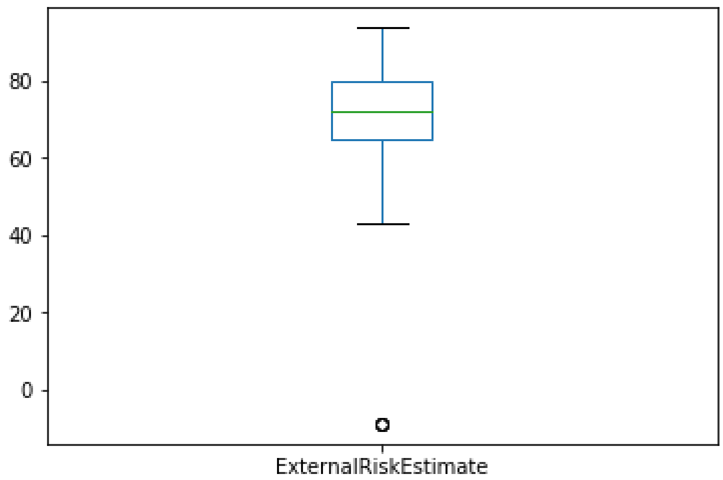
```
Out[58]: array([[<matplotlib.axes._subplots.AxesSubplot object at 0x00000189A19E9A20>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A196F160>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A08A33C8>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A1915630
         >],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x00000189A04CD898>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A0463B00>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A0467D68>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A1A93FD0
         >],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x00000189A1A9C080>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A07887F0>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A0666198>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A0779710
         >],
                [<matplotlib.axes._subplots.AxesSubplot object at 0x00000189A07D1470>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A06C8668>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A057D358>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x00000189A041DF98
         >]],
               dtype=object)

<Figure size 432x288 with 0 Axes>
```



Plot box plots for all the continuous features

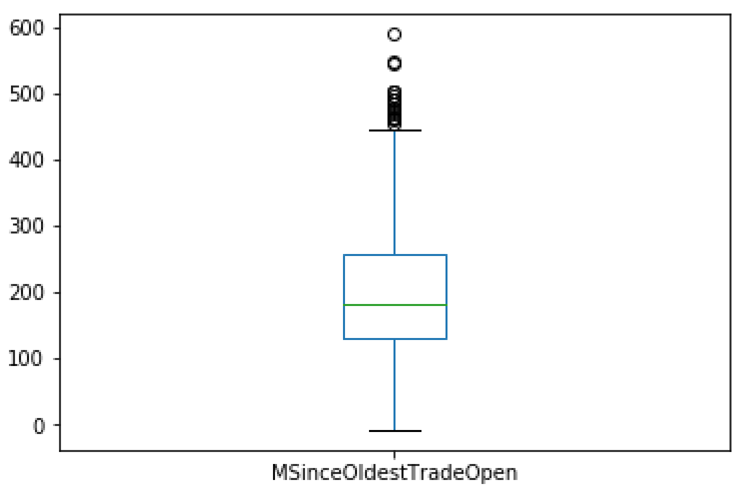
In [59]:
```
mySample_cleaned1['ExternalRiskEstimate'].plot(kind='box')
```

Out[59]: <matplotlib.axes._subplots.AxesSubplot at 0x189a231ee48>



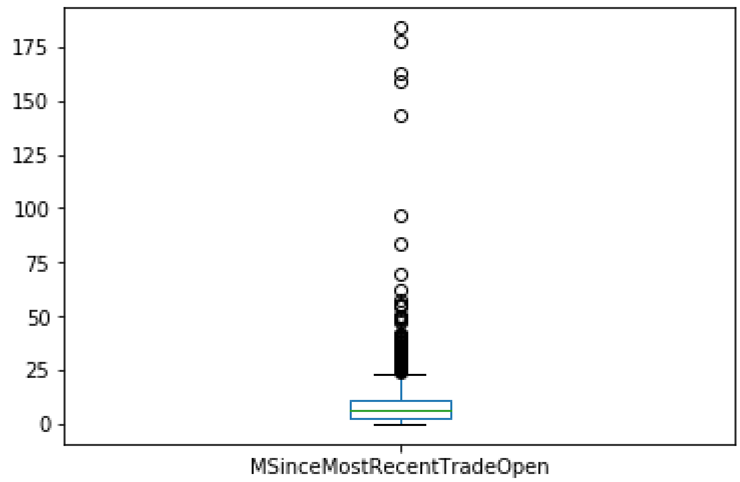
In [60]:
```
mySample_cleaned1['MSinceOldestTradeOpen'].plot(kind='box')
```

Out[60]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1b04668>



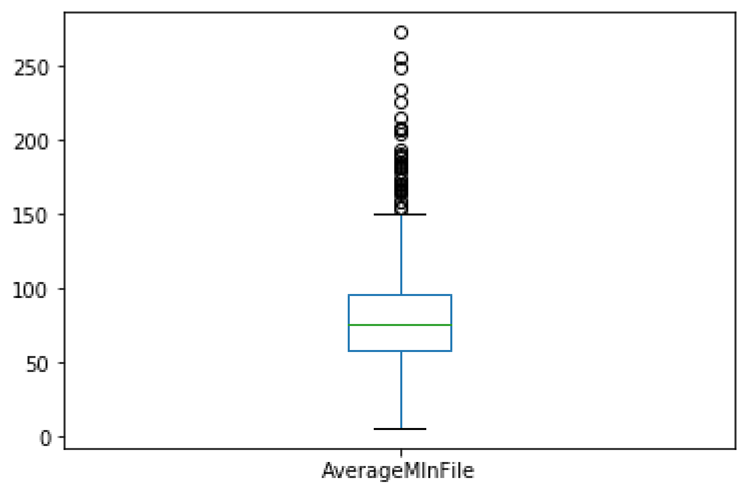
In [61]:
```
mySample_cleaned1['MSinceMostRecentTradeOpen'].plot(kind='box')
```

Out[61]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1b618d0>



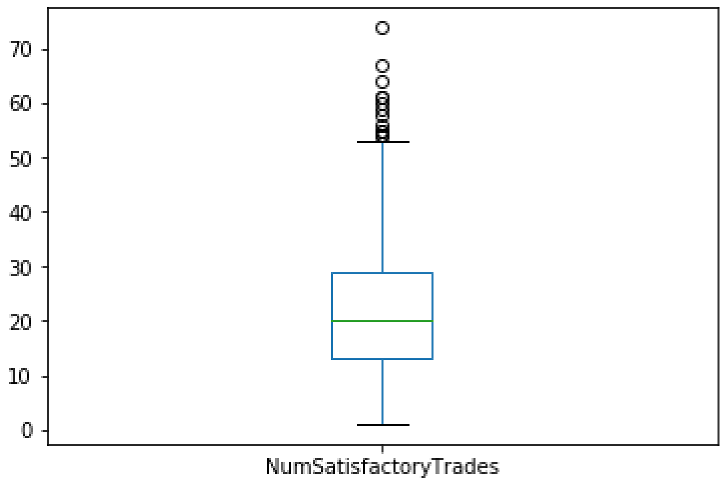
In [62]:
```
mySample_cleaned1['AverageMInFile'].plot(kind='box')
```

Out[62]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1bbdf60>



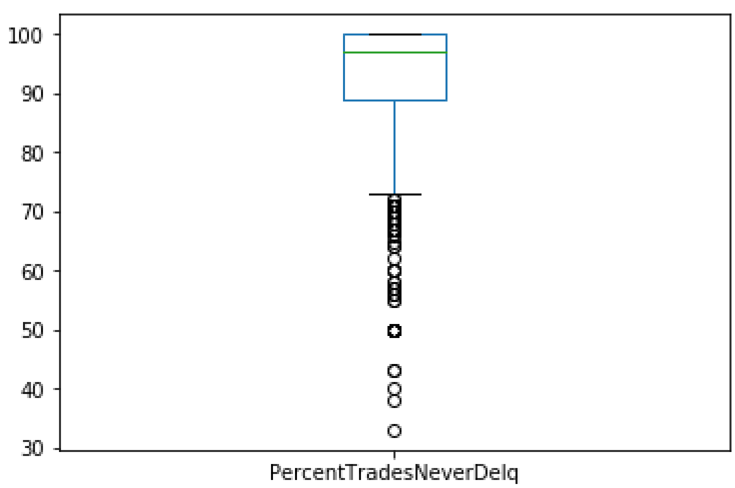
In [63]:
```
mySample_cleaned1['NumSatisfactoryTrades'].plot(kind='box')
```

Out[63]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1c11f28>



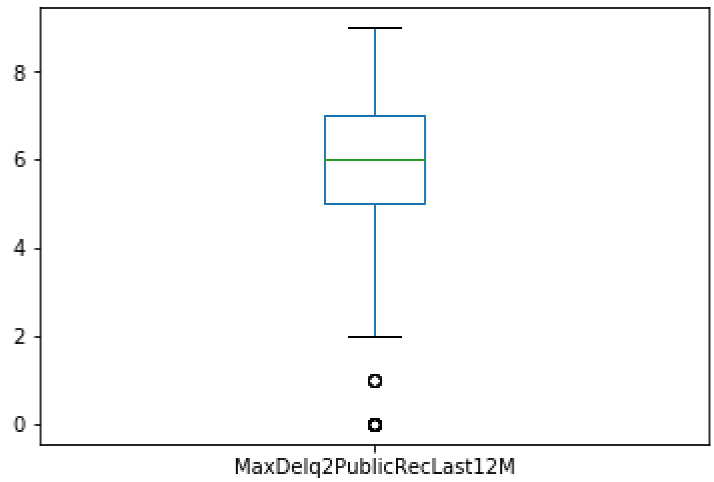
In [64]:
```
mySample_cleaned1['PercentTradesNeverDelq'].plot(kind='box')
```

Out[64]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1c73cf8>



In [65]:
```
mySample_cleaned1['MaxDelq2PublicRecLast12M'].plot(kind='box')
```

Out[65]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1ccc320>

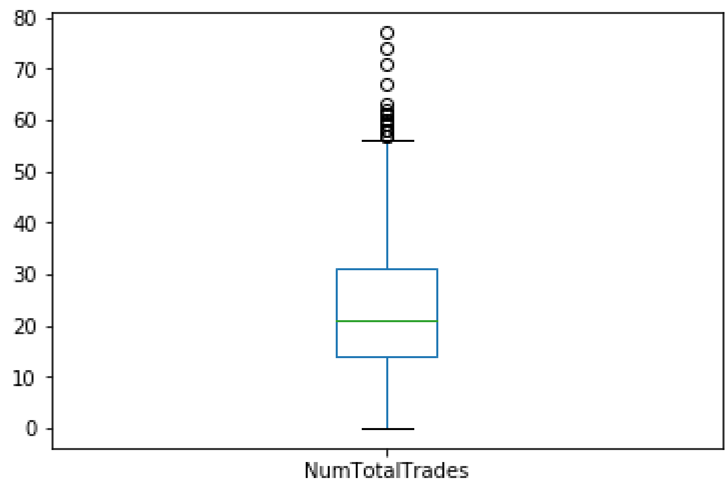


In [66]:
```
mySample_cleaned1['MaxDelqEver'].plot(kind='box')
```

Out[66]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1d1e7f0>

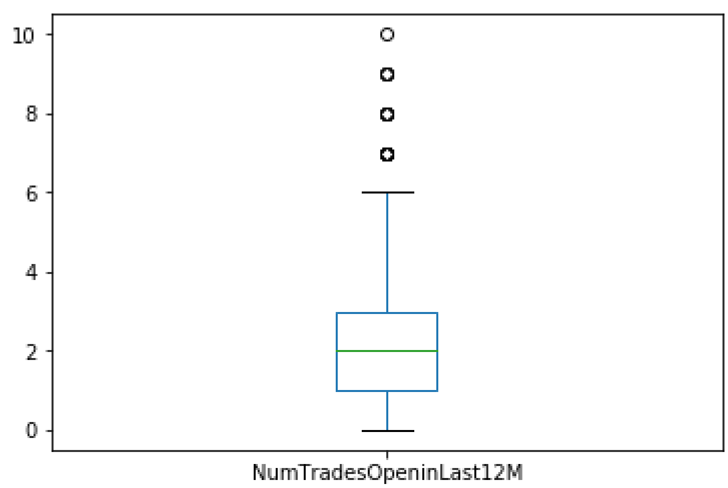
In [67]:
```
mySample_cleaned1['NumTotalTrades'].plot(kind='box')
```

Out[67]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1d79390>



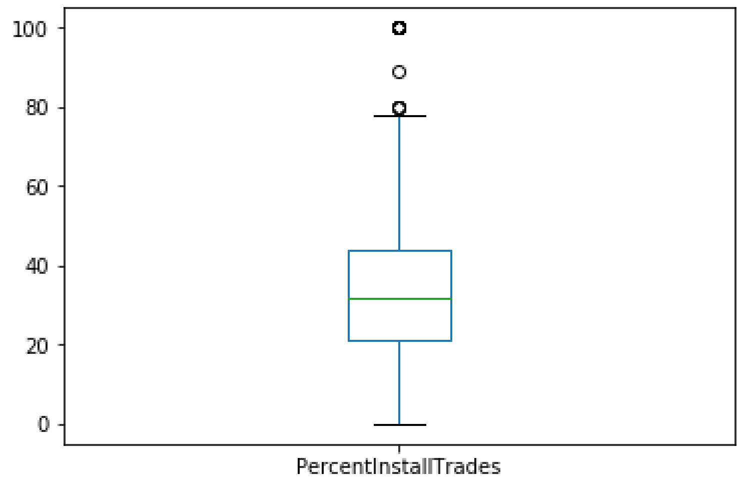
In [68]:
```
mySample_cleaned1['NumTradesOpeninLast12M'].plot(kind='box')
```

Out[68]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1de2470>



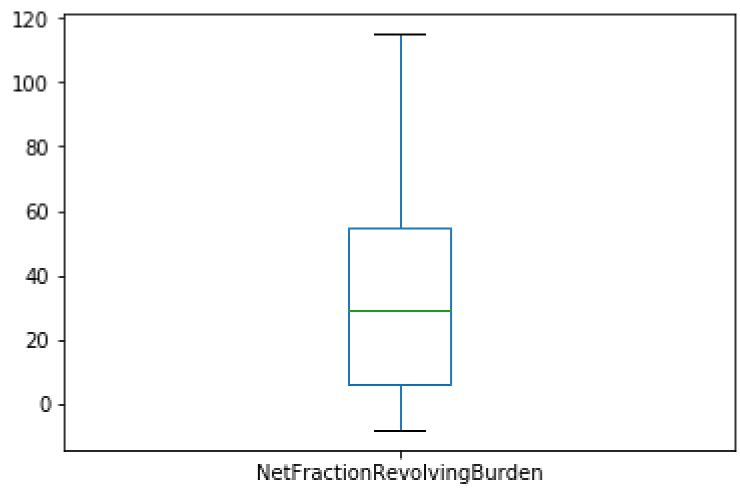
In [69]:
```
mySample_cleaned1['PercentInstallTrades'].plot(kind='box')
```

Out[69]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1e24358>



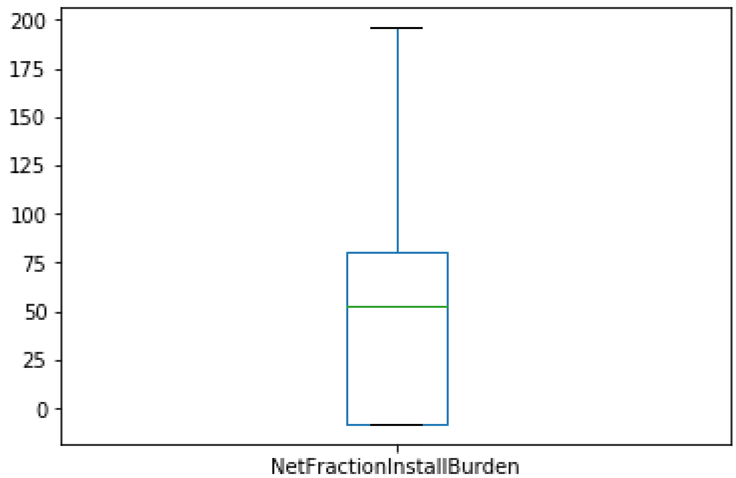
In [70]:
```
mySample_cleaned1['NetFractionRevolvingBurden'].plot(kind='box')
```

Out[70]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1e89da0>



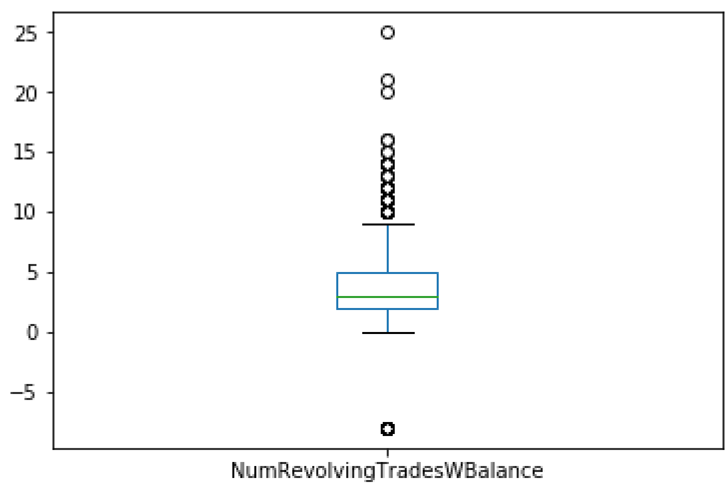
In [71]:
```
mySample_cleaned1['NetFractionInstallBurden'].plot(kind='box')
```

Out[71]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1ee8be0>



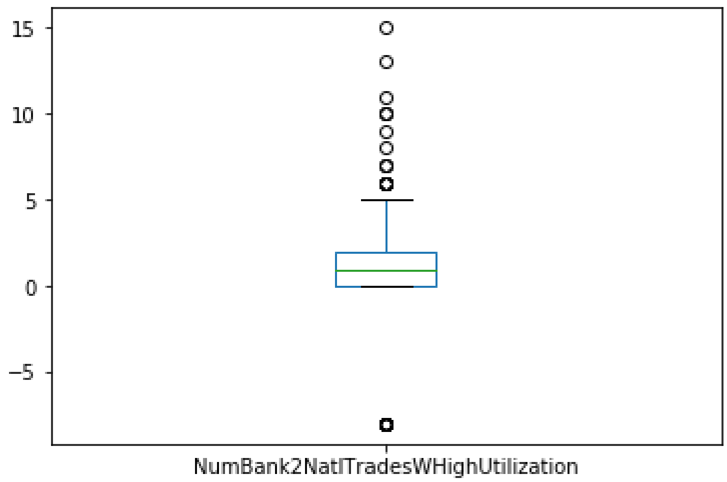
In [72]:
```
mySample_cleaned1['NumRevolvingTradesWBalance'].plot(kind='box')
```

Out[72]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1f43550>



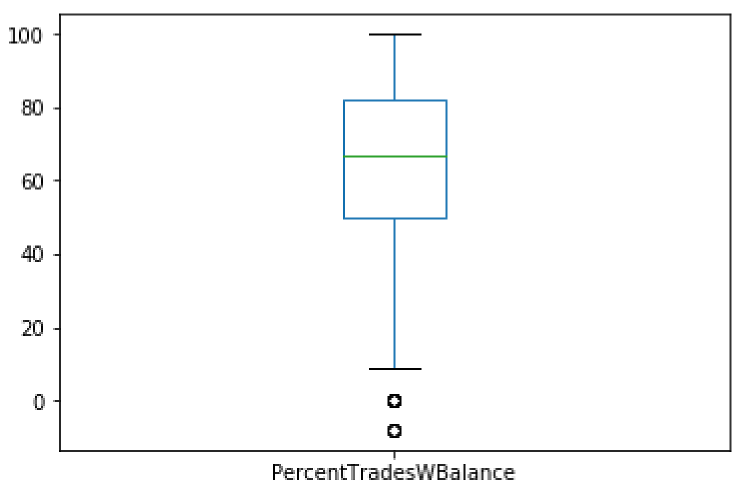
In [73]:
```
mySample_cleaned1['NumBank2NatlTradesWHighUtilization'].plot(kind='box')
```

Out[73]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1f9ea20>



In [74]:
```
mySample_cleaned1['PercentTradesWBalance'].plot(kind='box')
```

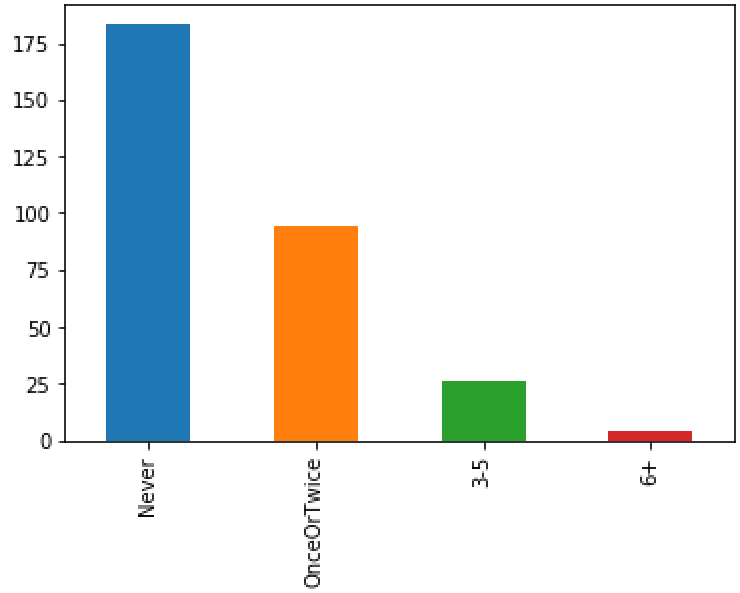
Out[74]: <matplotlib.axes._subplots.AxesSubplot at 0x189a1ff3f60>



Plot bar plots for all the categorical features

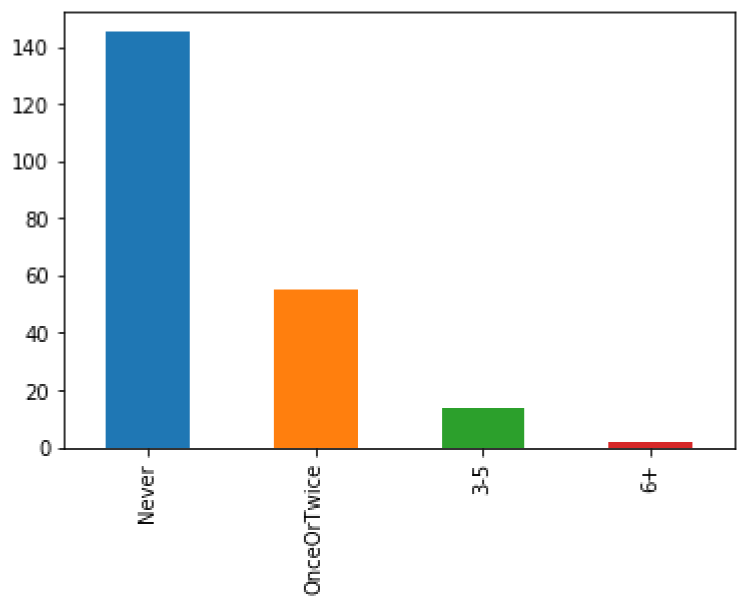
In [75]:
```
mySample_cleaned1['NumTrades60Ever2DerogPubRec'].value_counts().plot(kind='ba
r')
```

Out[75]: <matplotlib.axes._subplots.AxesSubplot at 0x189a2042978>



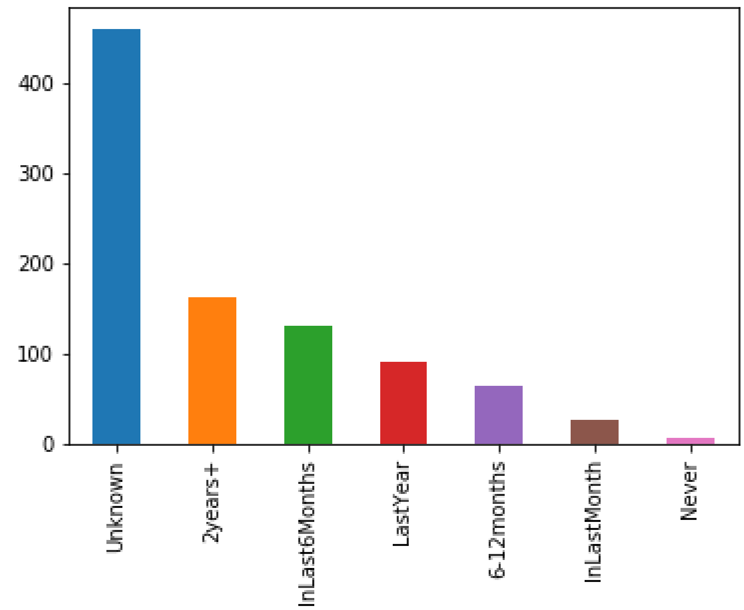
In [76]:
```
mySample_cleaned1['NumTrades90Ever2DerogPubRec'].value_counts().plot(kind='ba
r')
```

Out[76]: <matplotlib.axes._subplots.AxesSubplot at 0x189a20b5438>



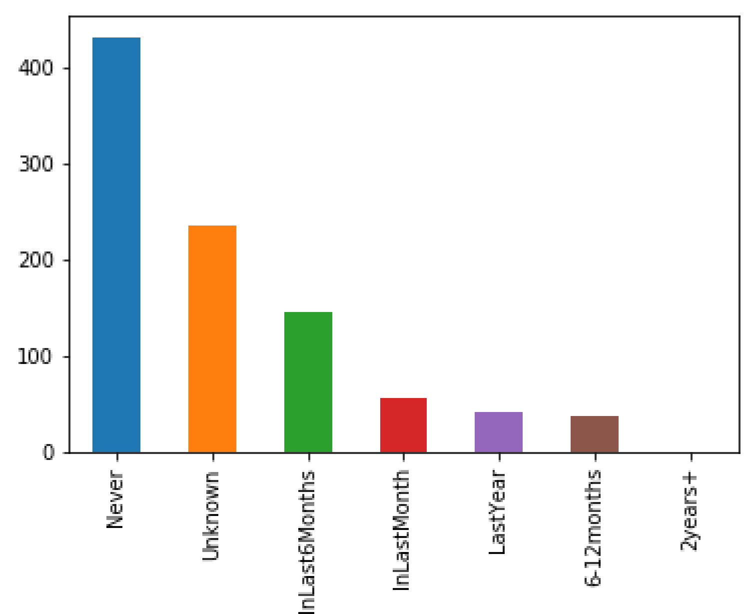
In [77]:
```
mySample_cleaned1['MSinceMostRecentDelq'].value_counts().plot(kind='bar')
```

Out[77]: <matplotlib.axes._subplots.AxesSubplot at 0x189a2117ac8>



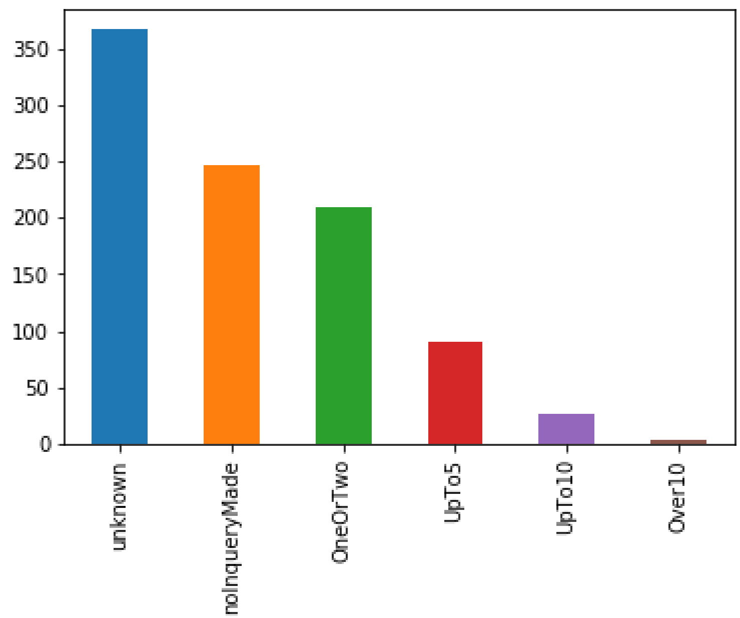
In [78]:
```
mySample_cleaned1['MSinceMostRecentInqexcl7days'].value_counts().plot(kind='ba
r')
```

Out[78]: <matplotlib.axes._subplots.AxesSubplot at 0x189a217e240>



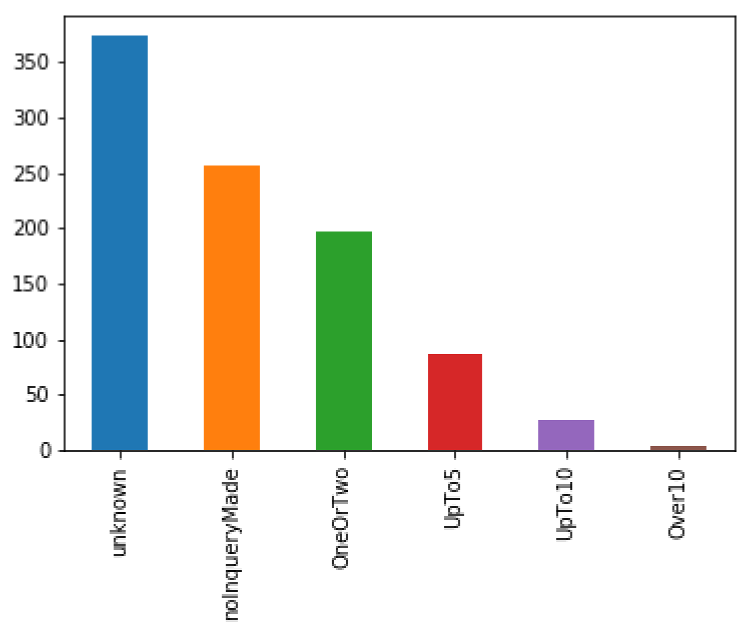
In [79]:
```
mySample_cleaned1['NumInqLast6M'].value_counts().plot(kind='bar')
```

Out[79]: <matplotlib.axes._subplots.AxesSubplot at 0x189a21e7940>



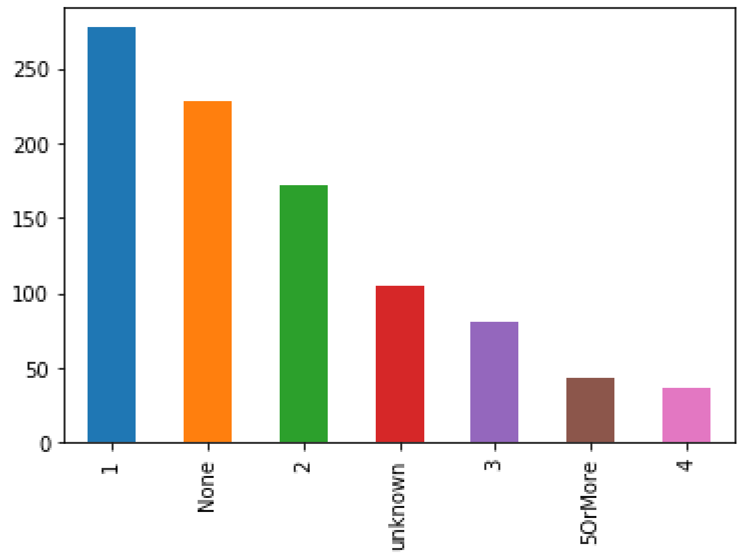
In [80]:
```
mySample_cleaned1['NumInqLast6Mexcl7days'].value_counts().plot(kind='bar')
```

Out[80]: <matplotlib.axes._subplots.AxesSubplot at 0x189a223f828>



In [81]:
```
mySample_cleaned1['NumInstallTradesWBalance'].value_counts().plot(kind='bar')
```

Out[81]: <matplotlib.axes._subplots.AxesSubplot at 0x189a22ad5f8>



Discuss your initial findings

In [82]:
```
mySample_cleaned1.select_dtypes(['int64', 'float64']).describe().T
```

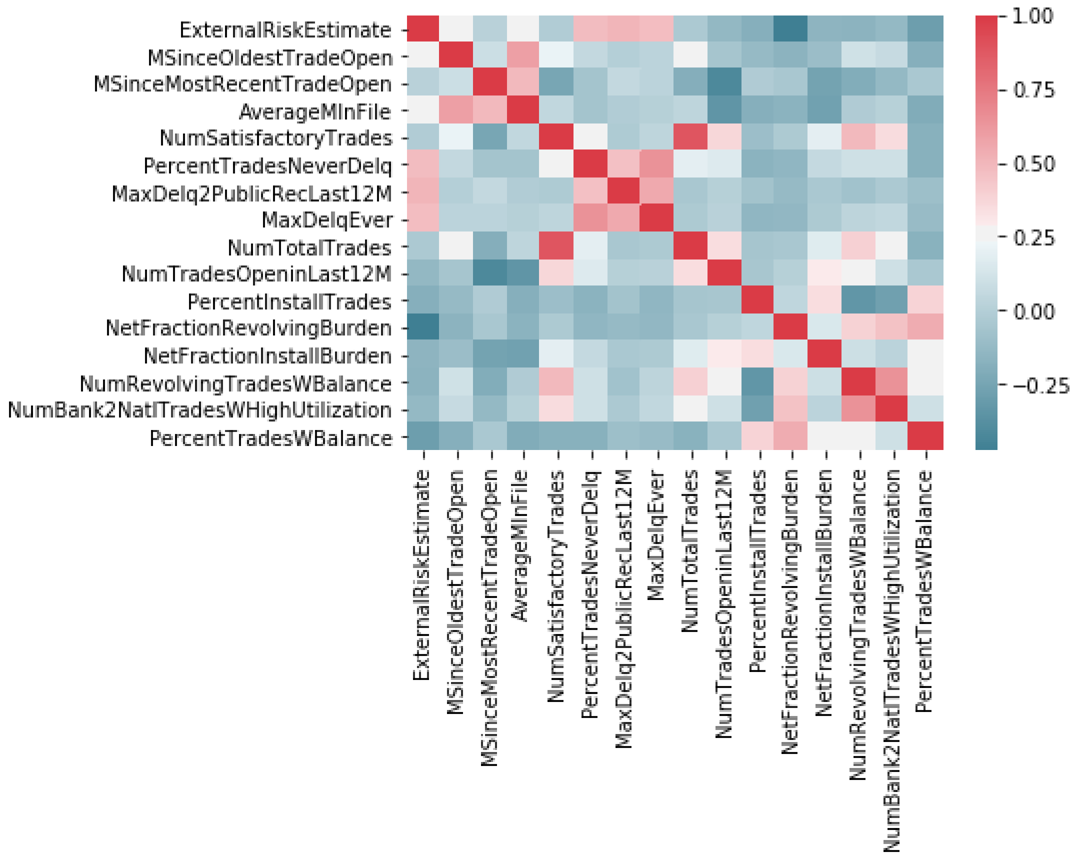
Out[82]:

	count	mean	std	min	25%	50%
ExternalRiskEstimate	944.0	71.738347	11.051234	-9.0	65.0	72.0
MSinceOldestTradeOpen	944.0	195.483051	105.323487	-8.0	130.0	181.0
MSinceMostRecentTradeOpen	944.0	9.846398	15.241942	0.0	3.0	6.0
AverageMInFile	944.0	79.834746	35.913650	6.0	58.0	76.0
NumSatisfactoryTrades	944.0	21.611229	12.049576	1.0	13.0	20.0
PercentTradesNeverDelq	944.0	92.345339	11.425976	33.0	89.0	97.0
MaxDelq2PublicRecLast12M	944.0	5.733051	1.696709	0.0	5.0	6.0
MaxDelqEver	944.0	6.358051	1.893500	2.0	6.0	6.0
NumTotalTrades	944.0	23.024364	13.231940	0.0	14.0	21.0
NumTradesOpeninLast12M	944.0	1.954449	1.814003	0.0	1.0	2.0
PercentInstallTrades	944.0	33.490466	17.939391	0.0	21.0	32.0
NetFractionRevolvingBurden	944.0	33.582627	28.957598	-8.0	6.0	29.0
NetFractionInstallBurden	944.0	42.559322	42.095265	-8.0	-8.0	53.0
NumRevolvingTradesWBalance	944.0	3.880297	3.367469	-8.0	2.0	3.0
NumBank2NatlTradesWHighUtilization	944.0	0.599576	2.626773	-8.0	0.0	1.0
PercentTradesWBalance	944.0	65.698093	22.417161	-8.0	50.0	67.0

Here we can see all 944 rows out of 1000 rows (minus removed all -9 rows) in the dataset represented for the continuous features. We can actually see some minimum values now, or -8 or -9's in instances where individual values are missing/unknown. We can also see the large percentage of missing data in NetFractionInstallBurden as it is the only feature to also have -8 after the first quartile

In [83]:
```
import seaborn as sb
corr = mySample_cleaned1.loc[:,mySample_cleaned1.dtypes == 'int64'].corr()
sb.heatmap(corr, xticklabels=corr.columns, yticklabels=corr.columns, cmap=sb.d
iverging_palette(220, 10, as_cmap=True))
```

Out[83]: <matplotlib.axes._subplots.AxesSubplot at 0x189a450a8d0>



As NetFractionInstallBurden shows little correlation with most features, while averages could be put in place, it's better to remove the feature as the information entropy (amount of unknown information/derivable information from this feature) is very low, and replacing over a 1/3rd of the data in the feature could accidentally synthesize or overemphasize a known to be statistically insignificant trend, thus removal is the better option in this instance. The other features are low enough that replacing them missing values with averages should be beneficial.

Looking at the 75% versus max for

MSinceOldestTradeOpen (258.00, 589.0),

MSinceMostRecentTradeOpen (11.00, 184.0),

AverageMInFile (96.00, 273.0),

NumSatisfactoryTrades (29.00, 74.0),

NumTotalTrades (31.00, 77.0),

NumTradesOpeninLast12M (3.00, 10.0),

PercentInstallTrades (44.00, 100.0),

NetFractionRevolvingBurden (55.00, 115.0),

NetFractionInstallBurden (80.25, 196.0),

NumRevolvingTradesWBalance (5.00, 25.0),

NumBank2NatlTradesWHighUtilization (2.00, 15.0),

we can see a large gap in the values, indicating quite a lot of outliers. If these outliers are lowered to just above the upper bound as shown through box plots, the relevancy of any potential trends may be more obvious.

PercentTradesNeverDelq has this issue in reverse, but with the lowest value representing 0 (as in never delinquent) there will be value in keeping this for any modelling as there is correlated pattern shown in the heatmap between those who have a percentage of trades that were delinquent, and those that had none so non change will occur here.

In [84]:
```
mySample_cleaned1.select_dtypes(['category']).describe().T
```

Out[84]:

	count	unique	top	freq
NumTrades60Ever2DerogPubRec	307	4	Never	183
NumTrades90Ever2DerogPubRec	216	4	Never	145
MSinceMostRecentDelq	944	7	Unknown	460
MSinceMostRecentInqexcl7days	944	6	Never	431
NumInqLast6M	944	6	unknown	367
NumInqLast6Mexcl7days	944	6	unknown	374
NumInstallTradesWBalance	944	7	1	277

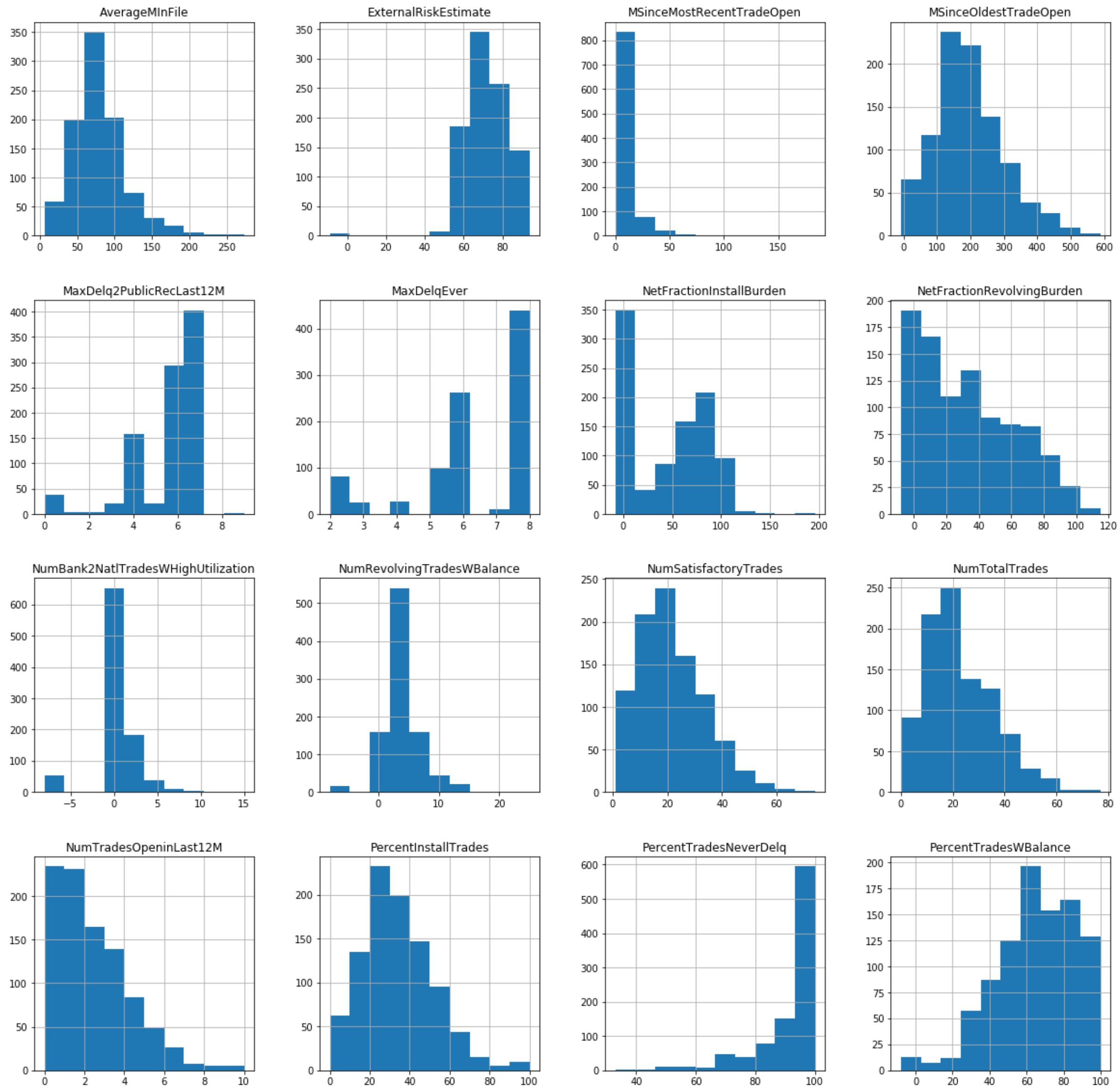
Categorically speaking, NumInqLast6M and NumInqLast6Mexcl7days are needlessly similar as they represent almost the same data, with no meaning to skipping the 7 day period. Therefore only the feature *including* the 7 days should be kept to avoid duplicates. Arguably, this also applies to the NumTradesXEver/DerogPubRec as it was noted on initial inspection when choosing categorical or continuous, that there would be quite a lot of overlap. For this reason, either could be removed (both showed the same trend with 'good' customers vs 'bad' customers). Therefore, 90 days should be removed over 60 days as 60 days encompasses more records.

The top frequency in many instances was 'never', implying most customers with loans, overdrafts or credit cards have not acquired any serious delinquencies of note, while also having at least one active item.

It will be interesting to see the most frequent options for those with unknown values as the most frequent response after the data quality is improved

Histograms for Continuous Features

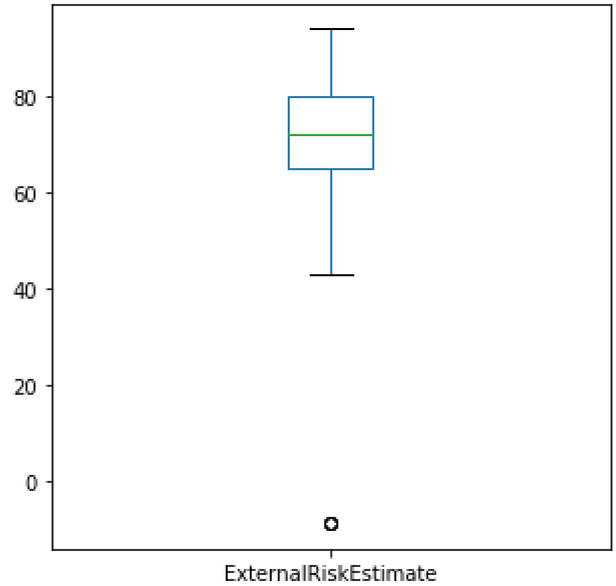
In [85]:
```
mySample_cleaned1.hist(figsize=(20, 20))

#save histograms as image
plt.savefig('CreditRisk-18206383-DataQualityReport-NumericFeatures-Histograms.png')
```

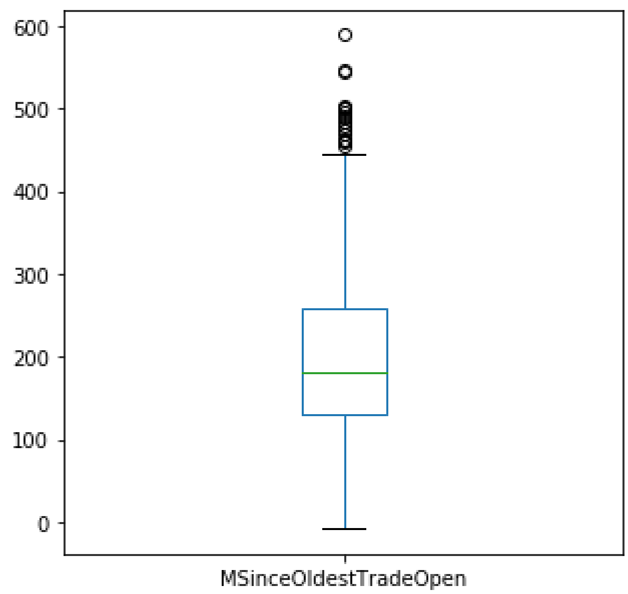


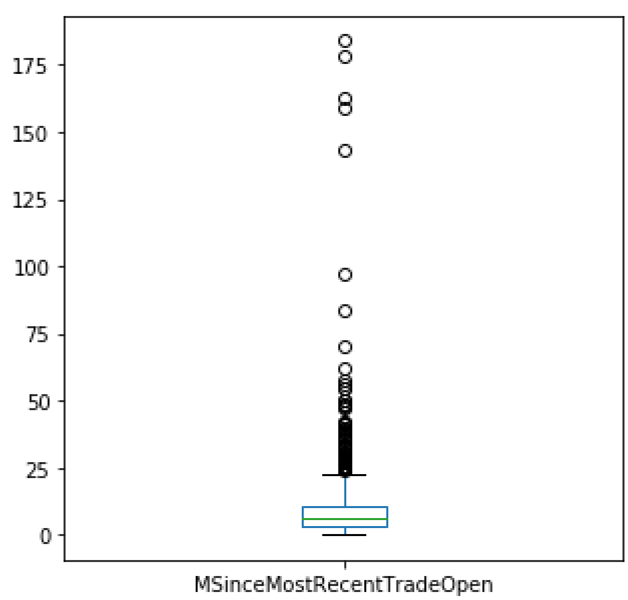
The average of monthsSinceOldestTradeOpened is distributed normally, with most customers being clients of the bank for ~200 months (over 16 years!). The majority of customers are close to if not at 100% of their trades never being delinquent (PercentTradesNeverDelq), and most customers also appear to have recently opened one or more trades (MSinceMostRecentTradeOpen/NumTradesOpeninLast12M) explaining the high amount of trades with a balance left to pay (PercentTradesWBalance).

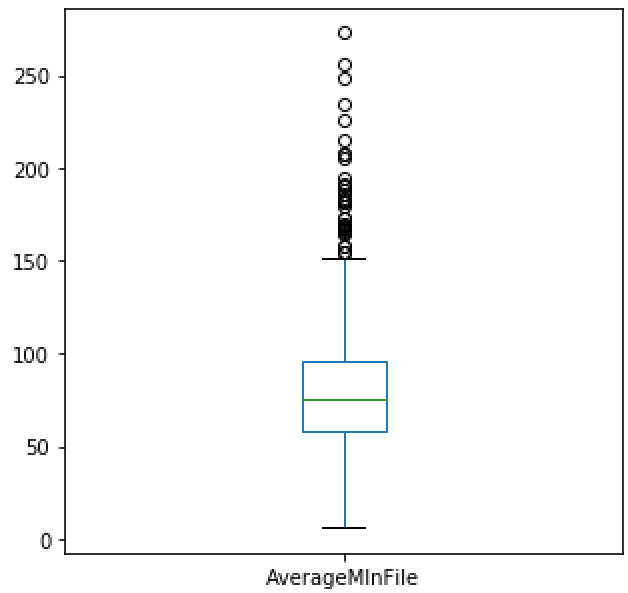
Box Plots for Continuous Features

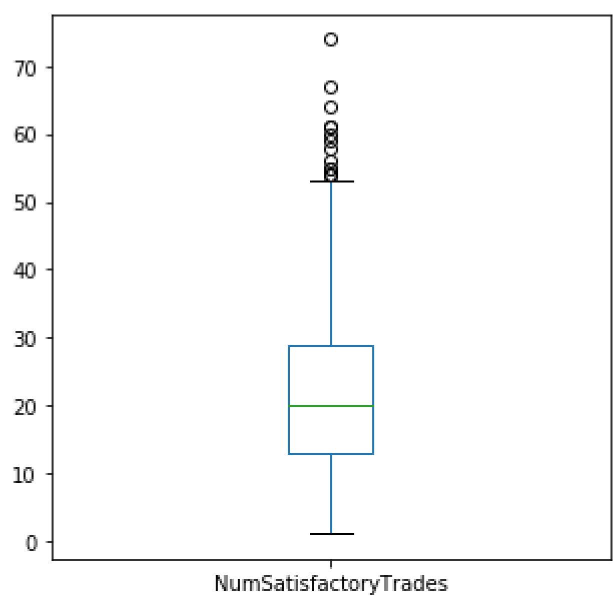
In [86]:
```
for column in numeric_columns:
    mySample_cleaned1[column].plot(kind='box', figsize=(5,5))
    plt.show()


#save boxplots as image
plt.savefig('CreditRisk-18206383-DataQualityReport-NumericFeatures-Boxplots.pn
g')
```

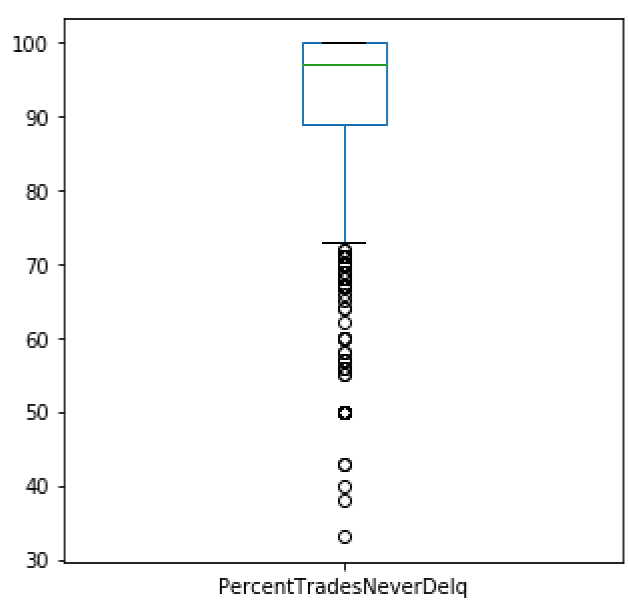

80
60
40
20
0
ExternalRiskEstimate

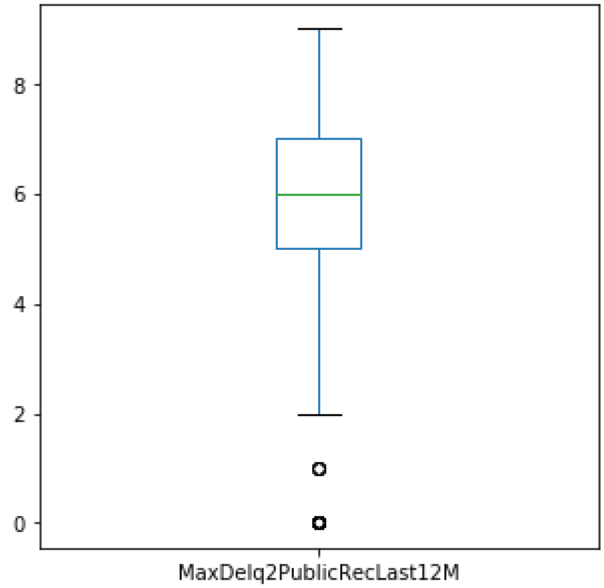

600
500
400
300
200
100
0
MSinceOldestTradeOpen

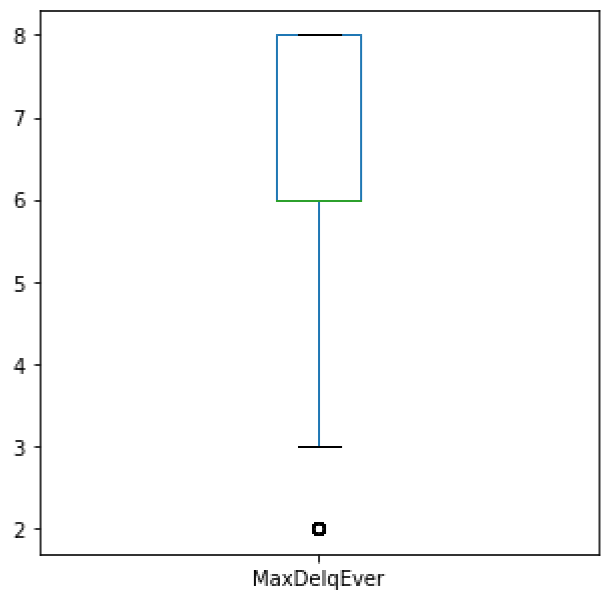

175
150
125
100
75
50
25
0
MSinceMostRecentTradeOpen

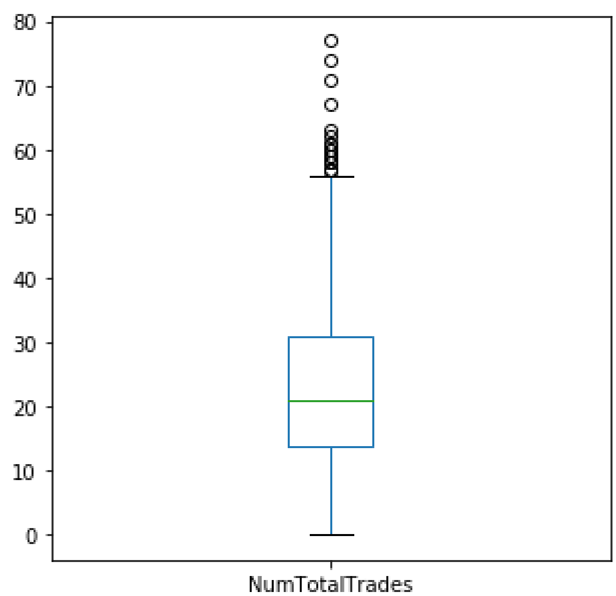

250
200
150
100
50
0
AverageMInFile

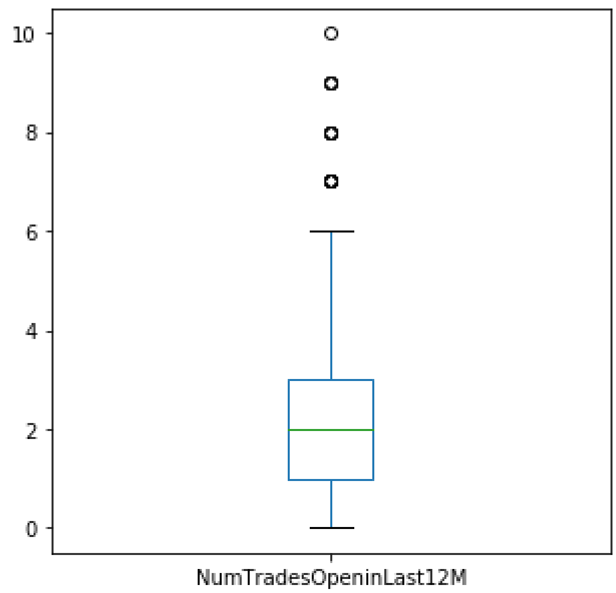

70
60
50
40
30
20
10
0
NumSatisfactoryTrades

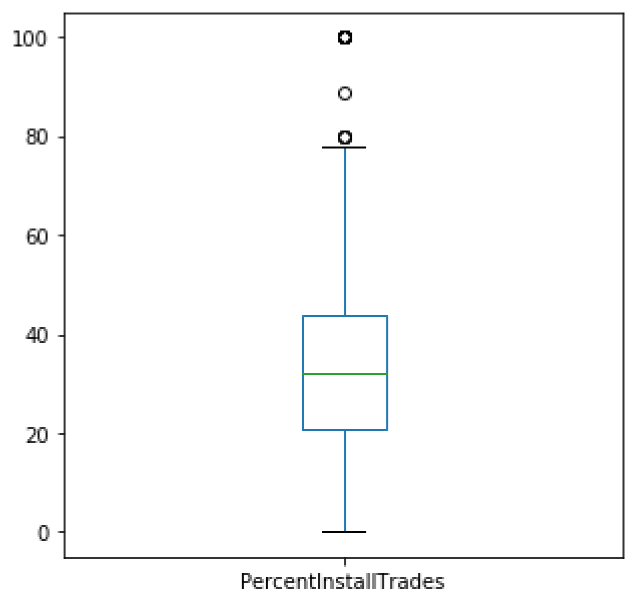

100
90
80
70
60
50
40
30
PercentTradesNeverDelq

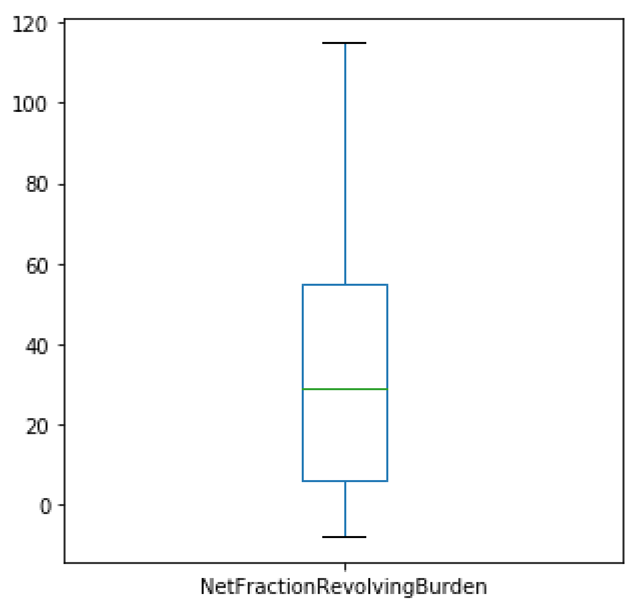

0
2
4
6
8
MaxDelq2PublicRecLast12M

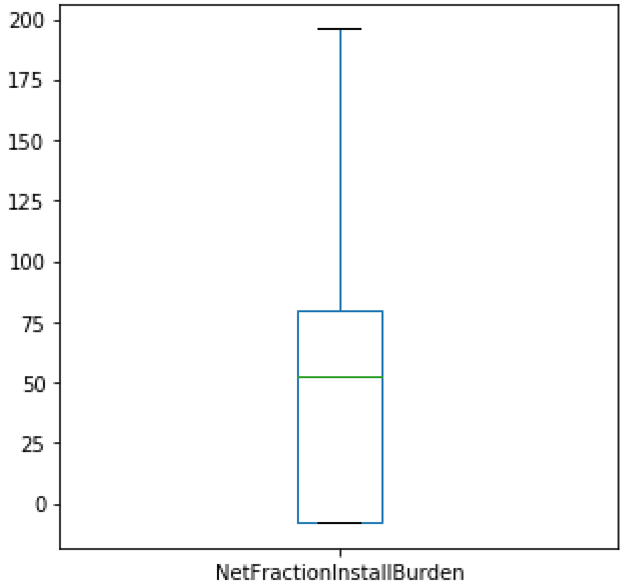

2
3
4
5
6
7
8
MaxDelqEver

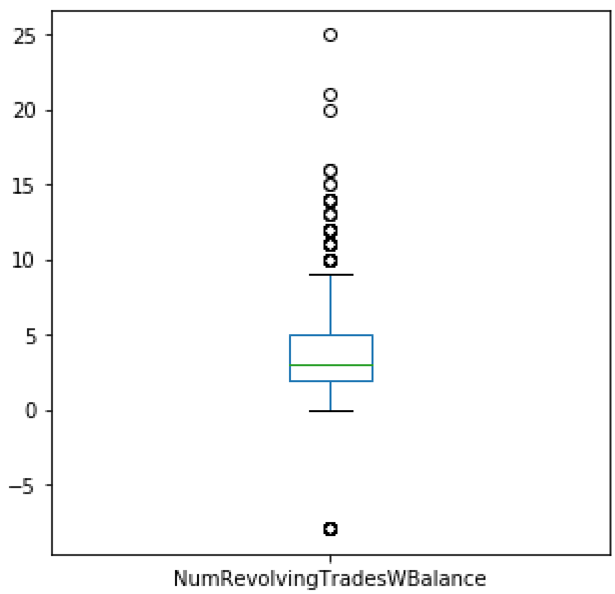

0
10
20
30
40
50
60
70
80
NumTotalTrades

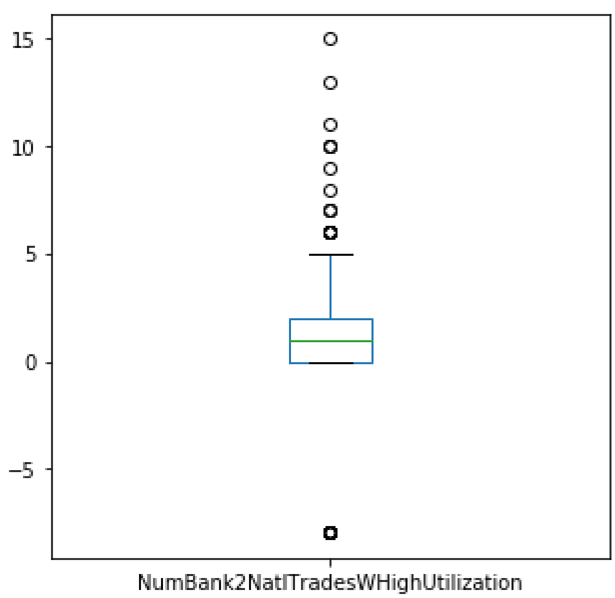

10
8
6
4
2
0
NumTradesOpeninLast12M

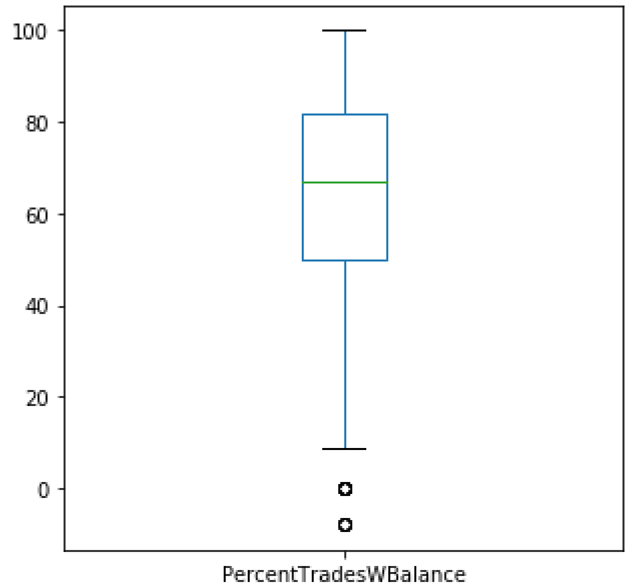

100
80
60
40
20
0
PercentInstallTrades


120
100
80
60
40
20
0
NetFractionRevolvingBurden


200
175
150
125
100
75
50
25
0
NetFractionInstallBurden


25
20
15
10
5
0
−5
NumRevolvingTradesWBalance


15
10
5
0
−5
NumBank2NatlTradesWHighUtilization



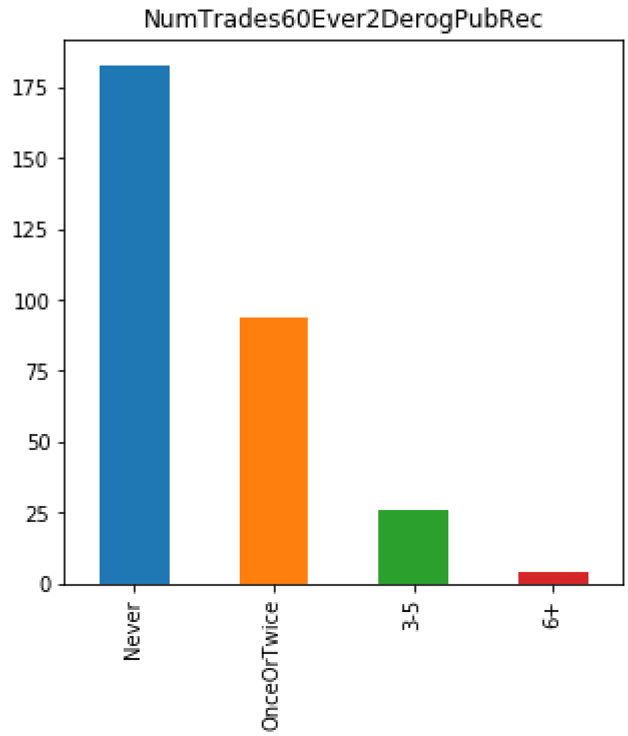

```
<Figure size 432x288 with 0 Axes>
```

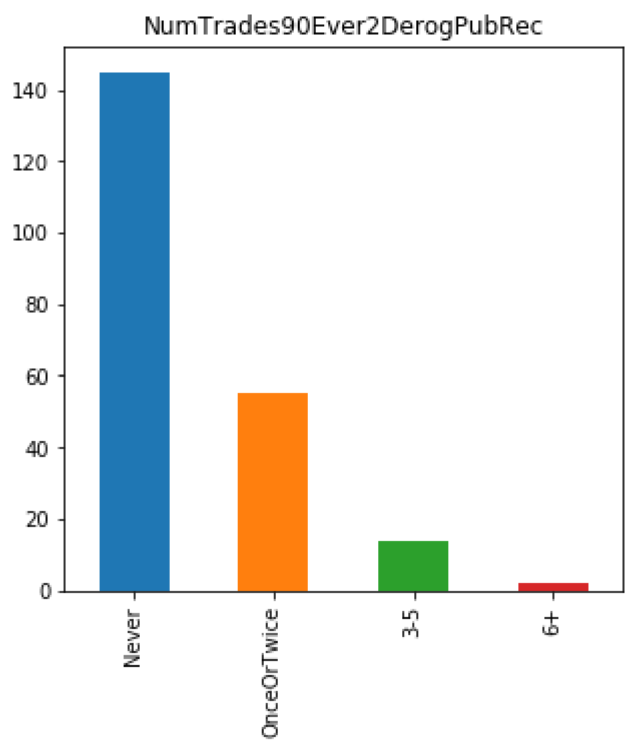
Many of the boxplotted features show outliers. Most of the outliers are larger than the max cut off point (upper bound), and in some cases such as MSinceMostRecentTradeOpen make the plot difficult to read. For these instances, it is suggesed that imputation to remove the unknown data be combined with an upper bound to represent the data without losing the pattern due to scattered outliers.

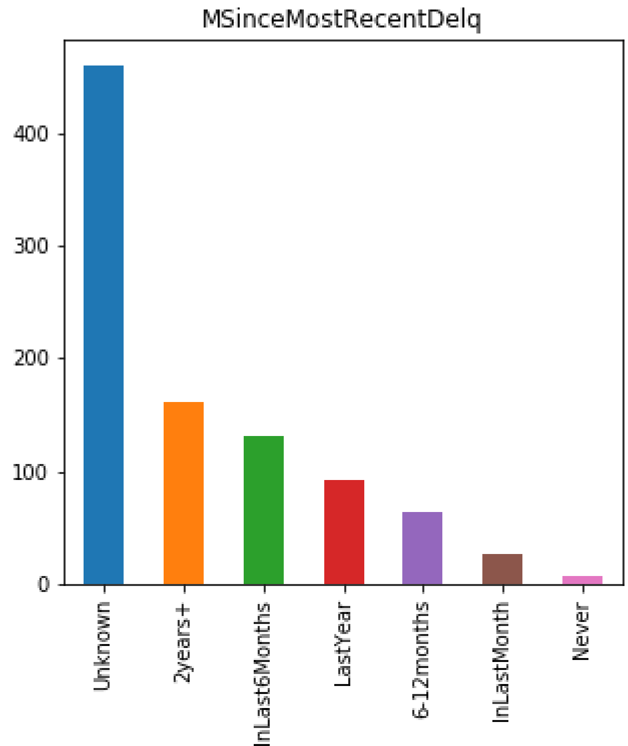
The only instance of a significant amount of outliers below the lower bound is PercentTradesNeverDelq. However the fact that many people have not been delinquent is represented across many features, and is an importannt trend and as such will neither be bounded nor removed.

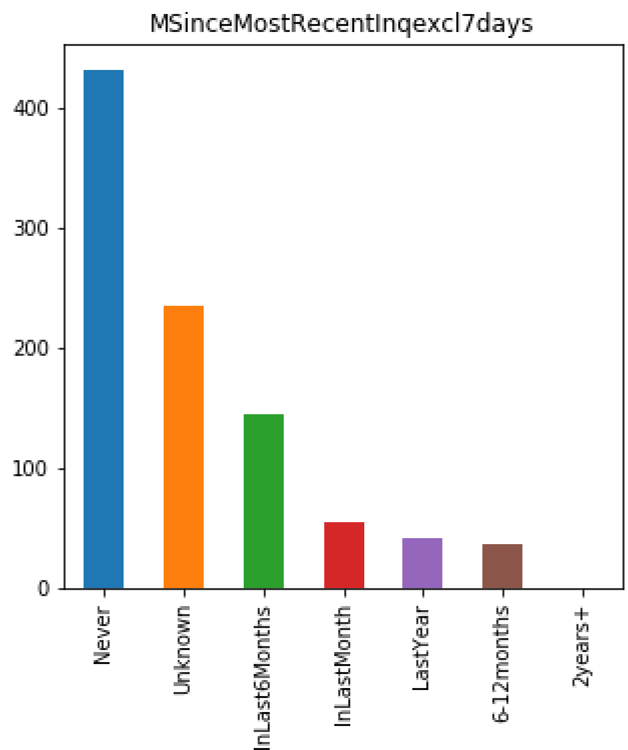
Bar Plots for Categorical Features

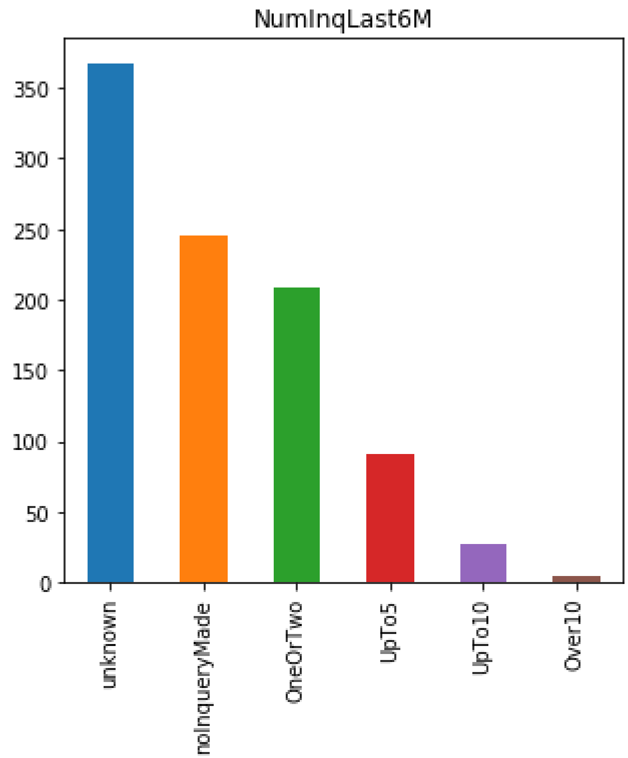
In [87]:
```
for column in categorical_columns:
    mySample_cleaned1[column].value_counts().plot(kind='bar', title=column, fi
gsize=(5,5))
    plt.show()


#save barplots as image
plt.savefig('CreditRisk-18206383-DataQualityReport-NumericFeatures-Barplots.pn
g')
```

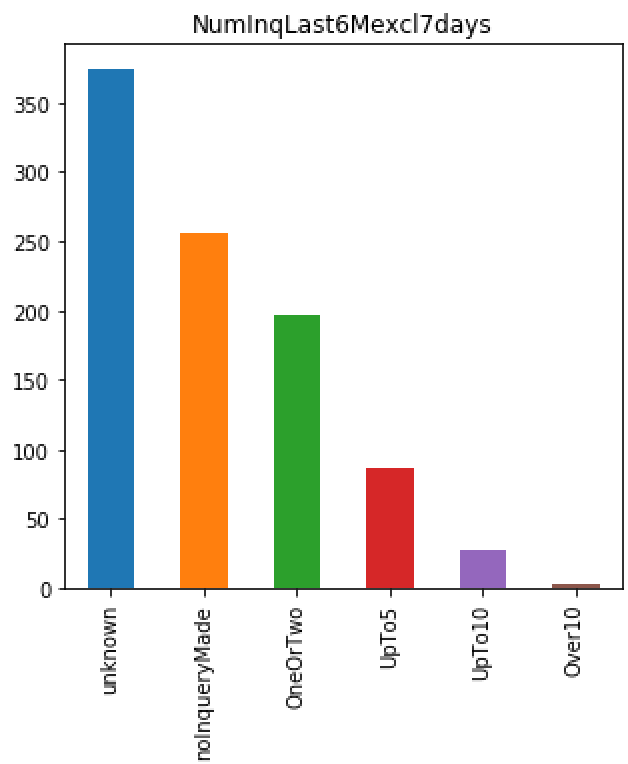

NumTrades60Ever2DerogPubRec
175
150
125
100
75
50
25
0
Never
OnceOrTwice
3-5
6+

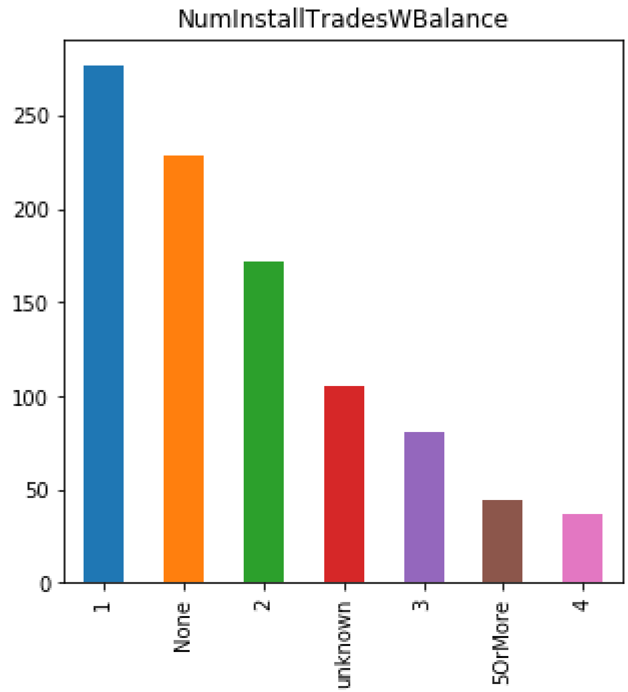

NumTrades90Ever2DerogPubRec
140
120
100
80
60
40
20
0
Never
OnceOrTwice
3-5
6+


MSinceMostRecentDelq
400
300
200
100
0
Unknown
2years+
InLast6Months
LastYear
6-12months
InLastMonth
Never


MSinceMostRecentInqexcl7days
400
300
200
100
0
Never
Unknown
InLast6Months
InLastMonth
LastYear
6-12months
2years+


NumInqLast6M
350
300
250
200
150
100
50
0
unknown
noInqueryMade
OneOrTwo
UpTo5
UpTo10
Over10


NumInqLast6Mexcl7days
350
300
250
200
150
100
50
0
unknown
noInqueryMade
OneOrTwo
UpTo5
UpTo10
Over10



<Figure size 432x288 with 0 Axes>

It should be noted to read with caution, as the plots appear order by frequency, than by x-axis value. All of the plotted features were changed from continuous features, and so the bins are relatively suited to the needs without cardinality issues. There is however an issue with multiple features effectively providing the same data, resulting in redundant information in the data. For this reason, the 2 sets of feature pairs below will see one feature removed each:

NumTrades60Ever2DerogPubRec: Do nothing | NumTrades90Ever2DerogPubRec: Delete

NumInqLast6M: Do nothing | NumInqLast6Mexcl7days: Delete

It was explained above during analysis of the categorical data table why the features to delete were chosen. NetFractionInstallBurden was also indicated to be deleted, as almost all of the data represented is either 0 or missing (34.5%), with no high correlation to any other useful features and thus is being removed rather than risking data falsification.

Save the initial discussion of your findings into a single data quality report PDF file

The PDF report should focus on the key issues identified in the data and discuss potential strategies to handle them. Simple listing of tables and plots without discussion and justification will not receive full marks.