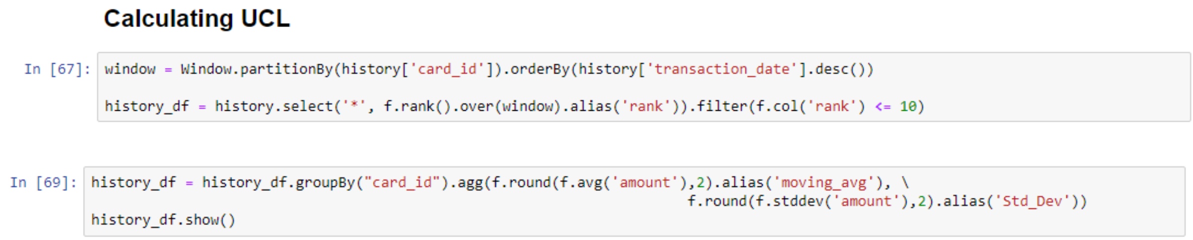
Loading the Lookup Table

To load the relevant data in the Lookup Table Script to calculate the moving average, standard deviation of the last 10 transactions for each card_id for the data present in Hadoop and NoSQL database:

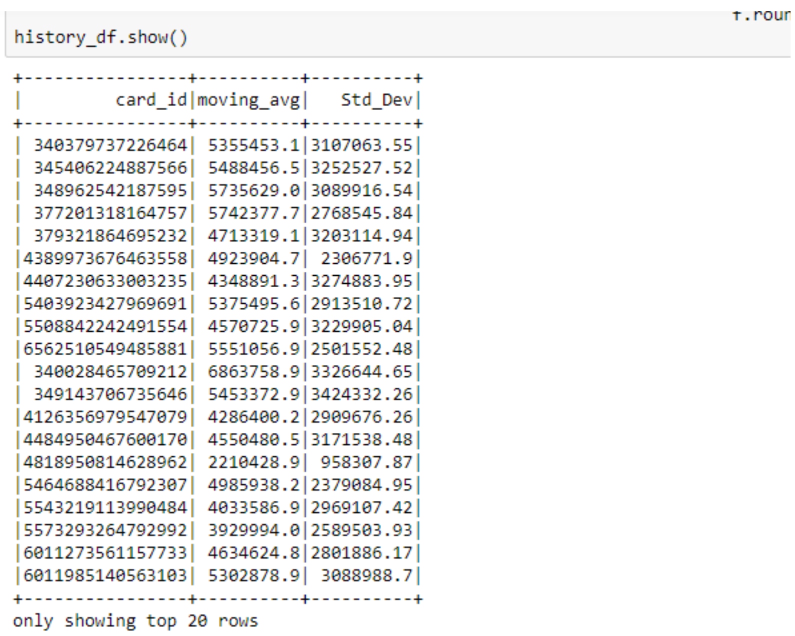
To calculate moving average & Std Deviation of last 10 transactions based on card_id. Created a window over existing df, grouping df on card_id so that all same card_id's collate and then order by transaction-date. This gives out transactions of each card_id in asc order. Now, we give a rank to each of those identified rows and only select ranked rows upto 10. Which means we are selecting only last 10 transactions. import sql f library in pyspark which imports all SQL functions to pyspark. Use f.avg what gives moving average of top 10 rows, f.stdev on amount field to calculate standard deviation of those top 10 rows. Print them to a separate column inside DF. While we execute, these newly calculated columns and data looks like:



Calculating UCL

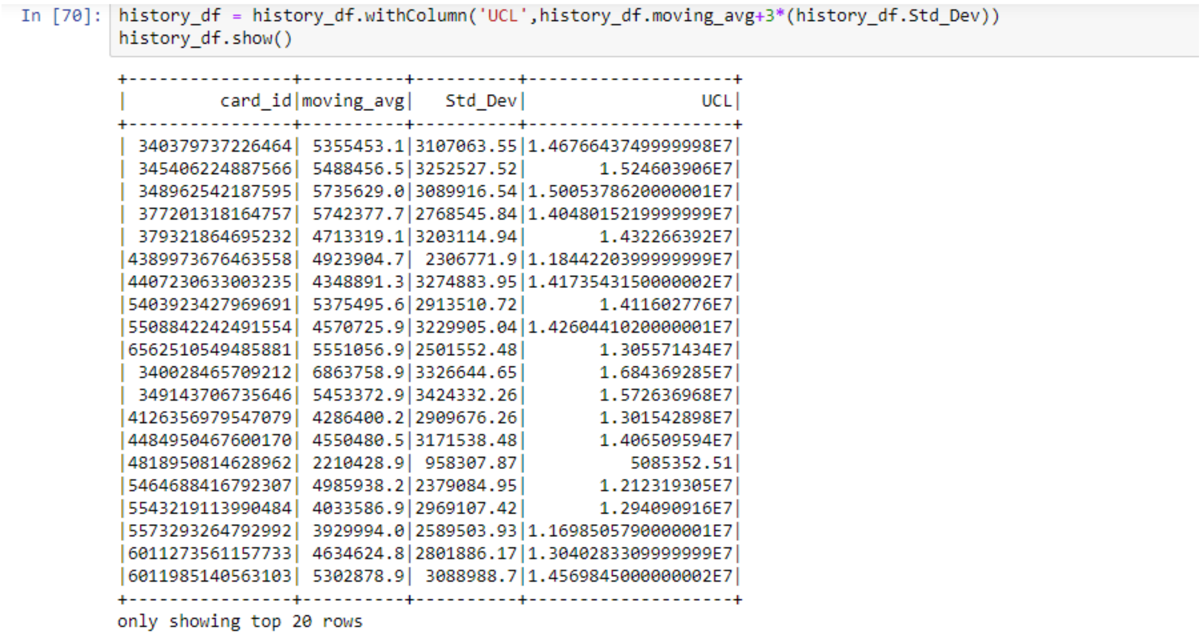
```
In [67]: window = Window.partitionBy(history['card_id']).orderBy(history['transaction_date'].desc())

         history_df = history.select('*', f.rank().over(window).alias('rank')).filter(f.col('rank') <= 10)
```

```
In [69]: history_df = history_df.groupBy("card_id").agg(f.round(f.avg('amount'),2).alias('moving_avg'), \
                                                         f.round(f.stddev('amount'),2).alias('Std_Dev'))
         history_df.show()
```



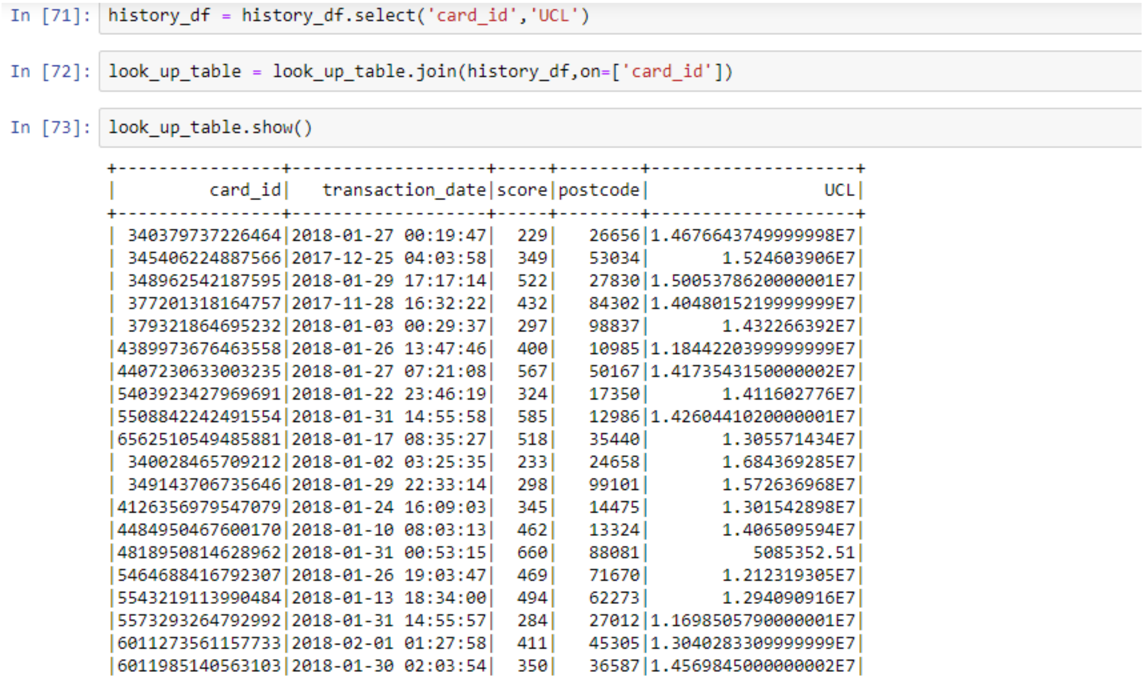
```
                                                        f.rour
history_df.show()

+----------------+----------+----------+
|         card_id|moving_avg|   Std_Dev|
+----------------+----------+----------+
| 340379737226464| 5355453.1|3107063.55|
| 345406224887566| 5488456.5|3252527.52|
| 348962542187595| 5735629.0|3089916.54|
| 377201318164757| 5742377.7|2768545.84|
| 379321864695232| 4713319.1|3203114.94|
|4389973676463558| 4923904.7| 2306771.9|
|4407230633003235| 4348891.3|3274883.95|
|5403923427969691| 5375495.6|2913510.72|
|5508842242491554| 4570725.9|3229905.04|
|6562510549485881| 5551056.9|2501552.48|
| 340028465709212| 6863758.9|3326644.65|
| 349143706735646| 5453372.9|3424332.26|
|4126356979547079| 4286400.2|2909676.26|
|4484950467600170| 4550480.5|3171538.48|
|4818950814628962| 2210428.9| 958307.87|
|5464688416792307| 4985938.2|2379084.95|
|5543219113990484| 4033586.9|2969107.42|
|5573293264792992| 3929994.0|2589503.93|
|6011273561157733| 4634624.8|2801886.17|
|6011985140563103| 5302878.9| 3088988.7|
+----------------+----------+----------+
only showing top 20 rows
```

Calculating UCL from moving average & standard deviation: Use the given formula in upgrad submission to calculate UCL from standard deviation & moving average. UCL = moving average + 3 *(standard deviation)



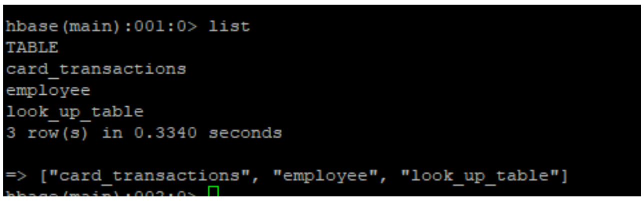
```
In [70]: history_df = history_df.withColumn('UCL',history_df.moving_avg+3*(history_df.Std_Dev))
         history_df.show()

         +----------------+----------+----------+--------------------+
         |         card_id|moving_avg|   Std_Dev|                 UCL|
         +----------------+----------+----------+--------------------+
         | 340379737226464| 5355453.1|3107063.55|1.4676643749999998E7|
         | 345406224887566| 5488456.5|3252527.52|       1.524603906E7|
         | 348962542187595| 5735629.0|3089916.54|1.5005378620000001E7|
         | 377201318164757| 5742377.7|2768545.84|1.4048015219999999E7|
         | 379321864695232| 4713319.1|3203114.94|       1.432266392E7|
         |4389973676463558| 4923904.7| 2306771.9|1.1844220399999999E7|
         |4407230633003235| 4348891.3|3274883.95|1.4173543150000002E7|
         |5403923427969691| 5375495.6|2913510.72|       1.411602776E7|
         |5508842242491554| 4570725.9|3229905.04|1.4260441020000001E7|
         |6562510549485881| 5551056.9|2501552.48|       1.305571434E7|
         | 340028465709212| 6863758.9|3326644.65|       1.684369285E7|
         | 349143706735646| 5453372.9|3424332.26|       1.572636968E7|
         |4126356979547079| 4286400.2|2909676.26|       1.301542898E7|
         |4484950467600170| 4550480.5|3171538.48|       1.406509594E7|
         |4818950814628962| 2210428.9| 958307.87|          5085352.51|
         |5464688416792307| 4985938.2|2379084.95|       1.212319305E7|
         |5543219113990484| 4033586.9|2969107.42|       1.294090916E7|
         |5573293264792992| 3929994.0|2589503.93|1.1698505790000001E7|
         |6011273561157733| 4634624.8|2801886.17|1.3040283309999999E7|
         |6011985140563103| 5302878.9| 3088988.7|1.4569845000000002E7|
         +----------------+----------+----------+--------------------+
         only showing top 20 rows
```

Join previous dataframe to this dataframe which has UCL calculated to reproduce a new dataframe with all data required to have for look up table.

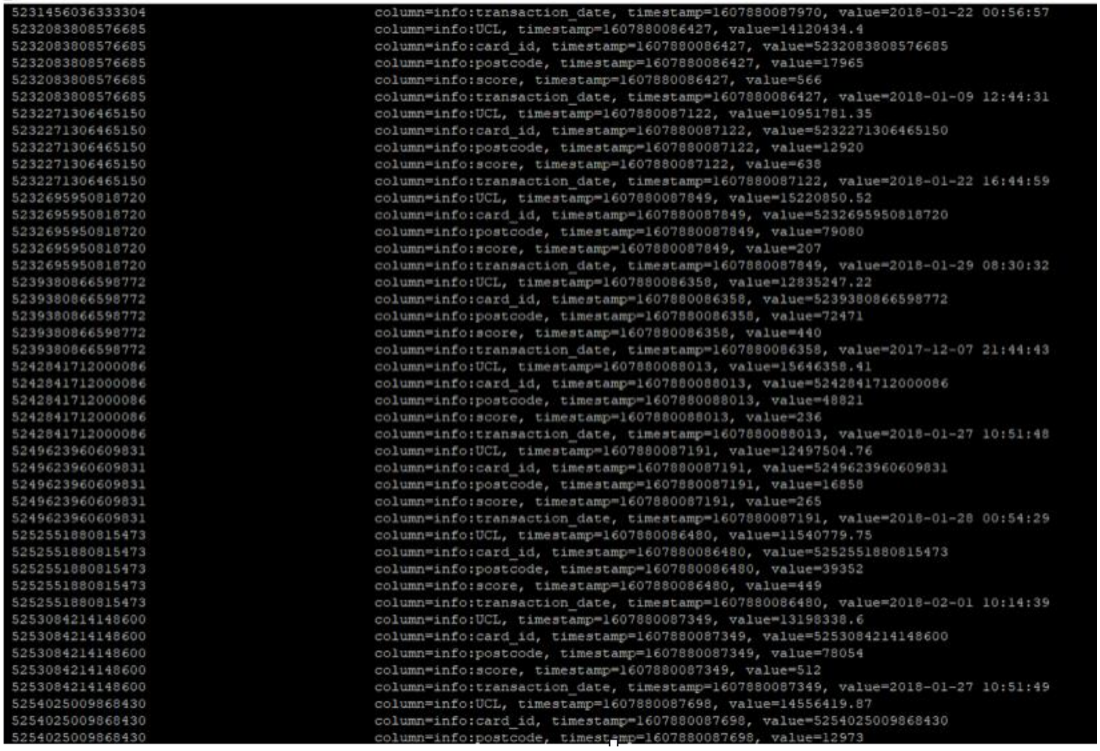


```
In [71]: history_df = history_df.select('card_id','UCL')

In [72]: look_up_table = look_up_table.join(history_df,on=['card_id'])

In [73]: look_up_table.show()

         +----------------+-------------------+-----+--------+--------------------+
         |         card_id|   transaction_date|score|postcode|                 UCL|
         +----------------+-------------------+-----+--------+--------------------+
         | 340379737226464|2018-01-27 00:19:47|  229|   26656|1.4676643749999998E7|
         | 345406224887566|2017-12-25 04:03:58|  349|   53034|       1.524603906E7|
         | 348962542187595|2018-01-29 17:17:14|  522|   27830|1.5005378620000001E7|
         | 377201318164757|2017-11-28 16:32:22|  432|   84302|1.4048015219999999E7|
         | 379321864695232|2018-01-03 00:29:37|  297|   98837|       1.432266392E7|
         |4389973676463558|2018-01-26 13:47:46|  400|   10985|1.1844220399999999E7|
         |4407230633003235|2018-01-27 07:21:08|  567|   50167|1.4173543150000002E7|
         |5403923427969691|2018-01-22 23:46:19|  324|   17350|       1.411602776E7|
         |5508842242491554|2018-01-31 14:55:58|  585|   12986|1.4260441020000001E7|
         |6562510549485881|2018-01-17 08:35:27|  518|   35440|       1.305571434E7|
         | 340028465709212|2018-01-02 03:25:35|  233|   24658|       1.684369285E7|
         | 349143706735646|2018-01-29 22:33:14|  298|   99101|       1.572636968E7|
         |4126356979547079|2018-01-24 16:09:03|  345|   14475|       1.301542898E7|
         |4484950467600170|2018-01-10 08:03:13|  462|   13324|       1.406509594E7|
         |4818950814628962|2018-01-31 00:53:15|  660|   88081|          5085352.51|
         |5464688416792307|2018-01-26 19:03:47|  469|   71670|       1.212319305E7|
         |5543219113990484|2018-01-13 18:34:00|  494|   62273|       1.294090916E7|
         |5573293264792992|2018-01-31 14:55:57|  284|   27012|1.1698505790000001E7|
         |6011273561157733|2018-02-01 01:27:58|  411|   45305|1.3040283309999999E7|
         |6011985140563103|2018-01-30 02:03:54|  350|   36587|1.4569845000000002E7|
```

<Command to see the table created and it’s content>



```
hbase(main):001:0> list
TABLE
card_transactions
employee
look_up_table
3 row(s) in 0.3340 seconds

=> ["card_transactions", "employee", "look_up_table"]
hbase(main):002:0>
```



```
 5231456036333304                          column=info:transaction_date, timestamp=1607880087970, value=2018-01-22 00:56:57
 5232083808576685                          column=info:UCL, timestamp=1607880086427, value=14120434.4
 5232083808576685                          column=info:card_id, timestamp=1607880086427, value=5232083808576685
 5232083808576685                          column=info:postcode, timestamp=1607880086427, value=17965
 5232083808576685                          column=info:score, timestamp=1607880086427, value=566
 5232083808576685                          column=info:transaction_date, timestamp=1607880086427, value=2018-01-09 12:44:31
 5232271306465150                          column=info:UCL, timestamp=1607880087122, value=10951781.35
 5232271306465150                          column=info:card_id, timestamp=1607880087122, value=5232271306465150
 5232271306465150                          column=info:postcode, timestamp=1607880087122, value=12920
 5232271306465150                          column=info:score, timestamp=1607880087122, value=638
 5232271306465150                          column=info:transaction_date, timestamp=1607880087122, value=2018-01-22 16:44:59
 5232695950818720                          column=info:UCL, timestamp=1607880087849, value=15220850.52
 5232695950818720                          column=info:card_id, timestamp=1607880087849, value=5232695950818720
 5232695950818720                          column=info:postcode, timestamp=1607880087849, value=79080
 5232695950818720                          column=info:score, timestamp=1607880087849, value=207
 5232695950818720                          column=info:transaction_date, timestamp=1607880087849, value=2018-01-29 08:30:32
 5239380866598772                          column=info:UCL, timestamp=1607880086358, value=12835247.22
 5239380866598772                          column=info:card_id, timestamp=1607880086358, value=5239380866598772
 5239380866598772                          column=info:postcode, timestamp=1607880086358, value=72471
 5239380866598772                          column=info:score, timestamp=1607880086358, value=440
 5239380866598772                          column=info:transaction_date, timestamp=1607880086358, value=2017-12-07 21:44:43
 5242841712000086                          column=info:UCL, timestamp=1607880088013, value=15646358.41
 5242841712000086                          column=info:card_id, timestamp=1607880088013, value=5242841712000086
 5242841712000086                          column=info:postcode, timestamp=1607880088013, value=48821
 5242841712000086                          column=info:score, timestamp=1607880088013, value=236
 5242841712000086                          column=info:transaction_date, timestamp=1607880088013, value=2018-01-27 10:51:48
 5249623960609831                          column=info:UCL, timestamp=1607880087191, value=12497504.76
 5249623960609831                          column=info:card_id, timestamp=1607880087191, value=5249623960609831
 5249623960609831                          column=info:postcode, timestamp=1607880087191, value=16858
 5249623960609831                          column=info:score, timestamp=1607880087191, value=265
 5249623960609831                          column=info:transaction_date, timestamp=1607880087191, value=2018-01-28 00:54:29
 5252551880815473                          column=info:UCL, timestamp=1607880086480, value=11540779.75
 5252551880815473                          column=info:card_id, timestamp=1607880086480, value=5252551880815473
 5252551880815473                          column=info:postcode, timestamp=1607880086480, value=39352
 5252551880815473                          column=info:score, timestamp=1607880086480, value=449
 5252551880815473                          column=info:transaction_date, timestamp=1607880086480, value=2018-02-01 10:14:39
 5253084214148600                          column=info:UCL, timestamp=1607880087349, value=13198338.6
 5253084214148600                          column=info:card_id, timestamp=1607880087349, value=5253084214148600
 5253084214148600                          column=info:postcode, timestamp=1607880087349, value=78054
 5253084214148600                          column=info:score, timestamp=1607880087349, value=512
 5253084214148600                          column=info:transaction_date, timestamp=1607880087349, value=2018-01-27 10:51:49
 5254025009868430                          column=info:UCL, timestamp=1607880087698, value=14556419.87
 5254025009868430                          column=info:card_id, timestamp=1607880087698, value=5254025009868430
 5254025009868430                          column=info:postcode, timestamp=1607880087698, value=12973
```