In [1]:
```
import pyspark
from pyspark.sql import SparkSession
spark = SparkSession.builder.appName("gen_n_compare").getOrCreate()
spark
```

Out[1]: **SparkSession - in-memory**

SparkContext

Spark UI

Version	`v3.2.1`
Master	`local[*]`
AppName	`gen_n_compare`

In [2]:
```
####
from datetime import datetime
from pytz import timezone
####
from pyspark.ml.feature import VectorAssembler
from pyspark.sql.types import *
from pyspark.sql.functions import *
from pyspark.ml.feature import StringIndexer
from pyspark.ml.feature import MinMaxScaler
####

def print_now():
    #from datetime import datetime
    #from pytz import timezone

    format = "%Y-%m-%d %H:%M:%S %Z%z"
    now_utc = datetime.now(timezone('UTC'))
    SGT =  timezone('Asia/Singapore')
    now_local = now_utc.astimezone(SGT)
    print(now_local.strftime(format))


def load_csv():
    path ="./"

    # Some csv data
    pnl = spark.read.csv(path+'pnl.csv',inferSchema=True,header=True)
    return pnl

def show_stats(df,input_columns,dependent_var):
    print('just some stats on distributions')
    df.groupBy(dependent_var).count().show()
    df.groupBy(input_columns).count().show()
```

In [4]:
```
print_now()
pnl = load_csv()
pnl.show()
print('showing the relevant cases only....')
pnl.where("calc_delta_pnldiff_z = 1").select('ccy','curve','calc_exp_z',lit('<->').alias('implic'),'calc_delta_pnldiff_z').show()
show_stats(pnl,['ccy', 'curve', 'calc_exp_z'],"calc_delta_pnldiff_z")
#pnl.printSchema()
```

```
2022-07-02 18:31:09 +08+0800
+---+---+-------+-----+-------+--------+----------+----------+----------+---------+--------------------+
| id|ccy|  curve| expd|repdate|calc_ttm|calc_exp_z|     pnl_v1|     pnl_v2|delta_pnl|calc_delta_pnldiff_z|
+---+---+-------+-----+-------+--------+----------+----------+----------+---------+--------------------+
|  1|aud|usdaud1|44718|  44698|      20|         0|737.7497674|737.7497674|        0|                   0|
|  2|jpy| usdjpy|44714|  44698|      16|         0|48.18714775|44.18714775|       -4|                   1|
|  3|aud| usdaud|44710|  44698|      12|         0|989.1945424|989.1945424|        0|                   0|
|  4|aud| usdjpy|44706|  44698|       8|         0|324.4788669|324.4788669|        0|                   0|
|  5|jpy| usdjpy|44702|  44698|       4|         0|988.9285272|984.9285272|       -4|                   1|
|  6|aud| usdaud|44698|  44698|       0|         1|53.96244552|55.96244552|        2|                   1|
|  7|aud| usdaud|44694|  44698|      -4|         0| 366.246082| 366.246082|        0|                   0|
|  8|usd| usdjpy|44690|  44698|      -8|         0|955.1660852|955.1660852|        0|                   0|
|  9|aud| usdaud|44698|  44698|       0|         1|992.4625445|990.4625445|       -2|                   1|
| 10|usd| usdaud|44694|  44698|      -4|         0|929.2450594|929.2450594|        0|                   0|
+---+---+-------+-----+-------+--------+----------+----------+----------+---------+--------------------+

showing the relevant cases only....
+---+------+----------+------+--------------------+
|ccy| curve|calc_exp_z|implic|calc_delta_pnldiff_z|
+---+------+----------+------+--------------------+
|jpy|usdjpy|         0|   <->|                   1|
|jpy|usdjpy|         0|   <->|                   1|
|aud|usdaud|         1|   <->|                   1|
|aud|usdaud|         1|   <->|                   1|
+---+------+----------+------+--------------------+

just some stats on distributions
+--------------------+-----+
|calc_delta_pnldiff_z|count|
+--------------------+-----+
|                   1|    4|
|                   0|    6|
+--------------------+-----+

+---+-------+----------+-----+
|ccy|  curve|calc_exp_z|count|
+---+-------+----------+-----+
|aud| usdjpy|         0|    1|
|aud|usdaud1|         0|    1|
|usd| usdjpy|         0|    1|
|aud| usdaud|         1|    2|
|aud| usdaud|         0|    2|
|usd| usdaud|         0|    1|
|jpy| usdjpy|         0|    2|
+---+-------+----------+-----+
```

In [5]:

```
#pnl2 = pnl.withColumn("calc_delta_pnldiff_z", pnl["calc_delta_pnldiff_z"].cast(StringType())).withColumn("calc_exp_z", pnl["calc_exp_z"].c
#pnl2.printSchema()
from pyspark.sql.functions import concat

# https://stackoverflow.com/questions/51325092/pyspark-fp-growth-algorithm-raise-valueerrorparams-must-be-either-a-param
# you cannot have an array in a cell containing 0 multiple times. array items must be unique. so:
pnl2 = pnl.withColumn("ccy", concat(lit("ccy:"),col('ccy'))) \
.withColumn("curve", concat(lit('curve:'),'curve')) \
.withColumn("calc_exp_z", concat(lit('exptoday:'),'calc_exp_z')) \
.withColumn("calc_delta_pnl_nz", concat(lit('dpnlnz:'),'calc_delta_pnldiff_z'))

pnl3 = pnl2.select('calc_ttm','delta_pnl',array('ccy', 'curve', 'calc_exp_z', 'calc_delta_pnl_nz').alias("items"))
#pnl3.printSchema()
pnl3.toPandas()
```

Out[5]:

	calc_ttm	delta_pnl	items
0	20	0	[ccy:aud, curve:usdaud1, exptoday:0, dpnlnz:0]
1	16	-4	[ccy:jpy, curve:usdjpy, exptoday:0, dpnlnz:1]
2	12	0	[ccy:aud, curve:usdaud, exptoday:0, dpnlnz:0]
3	8	0	[ccy:aud, curve:usdjpy, exptoday:0, dpnlnz:0]
4	4	-4	[ccy:jpy, curve:usdjpy, exptoday:0, dpnlnz:1]
5	0	2	[ccy:aud, curve:usdaud, exptoday:1, dpnlnz:1]
6	-4	0	[ccy:aud, curve:usdaud, exptoday:0, dpnlnz:0]
7	-8	0	[ccy:usd, curve:usdjpy, exptoday:0, dpnlnz:0]
8	0	-2	[ccy:aud, curve:usdaud, exptoday:1, dpnlnz:1]
9	-4	0	[ccy:usd, curve:usdaud, exptoday:0, dpnlnz:0]

start of rule-mining

In [6]:

```
from pyspark.ml.fpm import FPGrowth
fpGrowth = FPGrowth(itemsCol="items", minSupport=0.2, minConfidence=0.1)
model = fpGrowth.fit(pnl3)
```

```
In [7]: itempopularity = model.freqItemsets
        # ... FutureWarning: Deprecated in 3.0.0. Use SparkSession.builder.getOrCreate() instead
        # ... not under my control

        itempopularity.createOrReplaceTempView("itempopularity")
        # Then Query the temp view
        print("Top 20")
        dfo = spark.sql("SELECT * FROM itempopularity ORDER BY freq desc")
        dfo.printSchema()
        dofd=dfo.select('items','freq',size(dfo.items).alias('len'),array_contains(dfo.items, lit("dpnlnz:1")).alias('isdpnlnz')) #.where(  # .coll
        dofd.limit(20).toPandas()
```

```
/usr/local/spark/python/pyspark/sql/context.py:125: FutureWarning: Deprecated in 3.0.0. Use SparkSession.builder.getOrCreate() instead.
  warnings.warn(
Top 20
root
 |-- items: array (nullable = false)
 |    |-- element: string (containsNull = true)
 |-- freq: long (nullable = false)
```

Out[7]:

	items	freq	len	isdpnlnz
0	[exptoday:0]	8	1	False
1	[dpnlnz:0]	6	1	False
2	[ccy:aud]	6	1	False
3	[dpnlnz:0, exptoday:0]	6	2	False
4	[curve:usdaud]	5	1	False
5	[ccy:aud, dpnlnz:0, exptoday:0]	4	3	False
6	[ccy:aud, exptoday:0]	4	2	False
7	[curve:usdjpy, exptoday:0]	4	2	False
8	[dpnlnz:1]	4	1	True
9	[curve:usdaud, ccy:aud]	4	2	False
10	[ccy:aud, dpnlnz:0]	4	2	False
11	[curve:usdjpy]	4	1	False
12	[curve:usdaud, dpnlnz:0, exptoday:0]	3	3	False
13	[curve:usdaud, exptoday:0]	3	2	False
14	[curve:usdaud, dpnlnz:0]	3	2	False
15	[exptoday:1]	2	1	False
16	[exptoday:1, ccy:aud]	2	2	False
17	[exptoday:1, dpnlnz:1]	2	2	True
18	[exptoday:1, dpnlnz:1, curve:usdaud]	2	3	True
19	[exptoday:1, dpnlnz:1, ccy:aud]	2	3	True

In [8]:

```
dofdx = dofd.where('len>=2 and isdpnlnz')
dofdx.toPandas()
```

Out[8]:

	items	freq	len	isdpnlnz
0	[exptoday:1, dpnlnz:1]	2	2	True
1	[exptoday:1, dpnlnz:1, ccy:aud]	2	3	True
2	[exptoday:1, dpnlnz:1, curve:usdaud]	2	3	True
3	[exptoday:1, dpnlnz:1, curve:usdaud, ccy:aud]	2	4	True
4	[curve:usdjpy, dpnlnz:1]	2	2	True
5	[curve:usdjpy, dpnlnz:1, exptoday:0]	2	3	True
6	[dpnlnz:1, ccy:aud]	2	2	True
7	[dpnlnz:1, curve:usdaud]	2	2	True
8	[dpnlnz:1, curve:usdaud, ccy:aud]	2	3	True
9	[dpnlnz:1, exptoday:0]	2	2	True
10	[ccy:jpy, curve:usdjpy, dpnlnz:1]	2	3	True
11	[ccy:jpy, curve:usdjpy, dpnlnz:1, exptoday:0]	2	4	True
12	[ccy:jpy, dpnlnz:1]	2	2	True
13	[ccy:jpy, dpnlnz:1, exptoday:0]	2	3	True

In [10]:

```
assoc = model.associationRules
assoc.createOrReplaceTempView("assoc")
# Then Query the temp view
print("Top 20")
df2a = spark.sql("SELECT * FROM assoc ORDER BY confidence desc")
df2a.limit(20).toPandas()
```

```
/usr/local/spark/python/pyspark/sql/context.py:125: FutureWarning: Deprecated in 3.0.0. Use SparkSession.builder.getOrCreate() instead.
  warnings.warn(
Top 20
```

Out[10]:

	antecedent	consequent	confidence	lift	support
0	[dpnlnz:1, ccy:aud]	[exptoday:1]	1.0	5.000000	0.2
1	[ccy:jpy, exptoday:0]	[curve:usdjpy]	1.0	2.500000	0.2
2	[dpnlnz:1, ccy:aud]	[curve:usdaud]	1.0	2.000000	0.2
3	[ccy:jpy, exptoday:0]	[dpnlnz:1]	1.0	2.500000	0.2
4	[ccy:usd, exptoday:0]	[dpnlnz:0]	1.0	1.666667	0.2
5	[exptoday:1]	[dpnlnz:1]	1.0	2.500000	0.2
6	[exptoday:1, curve:usdaud]	[dpnlnz:1]	1.0	2.500000	0.2
7	[ccy:usd]	[dpnlnz:0]	1.0	1.666667	0.2
8	[exptoday:1, curve:usdaud]	[ccy:aud]	1.0	1.666667	0.2
9	[exptoday:1]	[curve:usdaud]	1.0	2.000000	0.2
10	[ccy:usd]	[exptoday:0]	1.0	1.250000	0.2
11	[curve:usdjpy, dpnlnz:1]	[exptoday:0]	1.0	1.250000	0.2
12	[exptoday:1, curve:usdaud, ccy:aud]	[dpnlnz:1]	1.0	2.500000	0.2
13	[ccy:jpy, curve:usdjpy, exptoday:0]	[dpnlnz:1]	1.0	2.500000	0.2
14	[ccy:jpy, dpnlnz:1, exptoday:0]	[curve:usdjpy]	1.0	2.500000	0.2
15	[ccy:jpy, dpnlnz:1]	[exptoday:0]	1.0	1.250000	0.2
16	[curve:usdaud, exptoday:0]	[dpnlnz:0]	1.0	1.666667	0.3
17	[exptoday:1]	[ccy:aud]	1.0	1.666667	0.2
18	[ccy:jpy]	[curve:usdjpy]	1.0	2.500000	0.2
19	[curve:usdjpy, dpnlnz:1]	[ccy:jpy]	1.0	5.000000	0.2

In [11]:

```
df2b=df2a.select('antecedent','consequent','confidence','lift','support', \
                 size(df2a.antecedent).alias('lenA'),size(df2a.consequent).alias('lenC'), \
                 array_contains(df2a.consequent,'dpnlnz:1').alias('Cisdpnlnz'))
df2b.toPandas()
```

Out[11]:

	antecedent	consequent	confidence	lift	support	lenA	lenC	Cisdpnlnz
0	[ccy:jpy, exptoday:0]	[curve:usdjpy]	1.000000	2.500000	0.2	2	1	False
1	[ccy:jpy, exptoday:0]	[dpnlnz:1]	1.000000	2.500000	0.2	2	1	True
2	[exptoday:1]	[ccy:aud]	1.000000	1.666667	0.2	1	1	False
3	[exptoday:1]	[dpnlnz:1]	1.000000	2.500000	0.2	1	1	True
4	[exptoday:1]	[curve:usdaud]	1.000000	2.000000	0.2	1	1	False
...	...	...	...	...	...	...	...	...
89	[ccy:aud]	[exptoday:1]	0.333333	1.666667	0.2	1	1	False
90	[ccy:aud]	[dpnlnz:1]	0.333333	0.833333	0.2	1	1	True
91	[exptoday:0]	[dpnlnz:1]	0.250000	0.625000	0.2	1	1	True
92	[exptoday:0]	[ccy:usd]	0.250000	1.250000	0.2	1	1	False
93	[exptoday:0]	[ccy:jpy]	0.250000	1.250000	0.2	1	1	False

94 rows × 8 columns

In [12]:

```
dofdx2 = df2b.where('lenC==1 and Cisdpnlnz')
dofdx2.orderBy(col("lift").desc(),col("lenA").asc()).toPandas()
```

Out[12]:

	antecedent	consequent	confidence	lift	support	lenA	lenC	Cisdpnlnz
0	[exptoday:1]	[dpnlnz:1]	1.000000	2.500000	0.2	1	1	True
1	[ccy:jpy]	[dpnlnz:1]	1.000000	2.500000	0.2	1	1	True
2	[ccy:jpy, exptoday:0]	[dpnlnz:1]	1.000000	2.500000	0.2	2	1	True
3	[exptoday:1, curve:usdaud]	[dpnlnz:1]	1.000000	2.500000	0.2	2	1	True
4	[ccy:jpy, curve:usdjpy]	[dpnlnz:1]	1.000000	2.500000	0.2	2	1	True
5	[exptoday:1, ccy:aud]	[dpnlnz:1]	1.000000	2.500000	0.2	2	1	True
6	[exptoday:1, curve:usdaud, ccy:aud]	[dpnlnz:1]	1.000000	2.500000	0.2	3	1	True
7	[ccy:jpy, curve:usdjpy, exptoday:0]	[dpnlnz:1]	1.000000	2.500000	0.2	3	1	True
8	[curve:usdjpy]	[dpnlnz:1]	0.500000	1.250000	0.2	1	1	True
9	[curve:usdaud, ccy:aud]	[dpnlnz:1]	0.500000	1.250000	0.2	2	1	True
10	[curve:usdjpy, exptoday:0]	[dpnlnz:1]	0.500000	1.250000	0.2	2	1	True
11	[curve:usdaud]	[dpnlnz:1]	0.400000	1.000000	0.2	1	1	True
12	[ccy:aud]	[dpnlnz:1]	0.333333	0.833333	0.2	1	1	True
13	[exptoday:0]	[dpnlnz:1]	0.250000	0.625000	0.2	1	1	True

In [13]:
```
## .. the most recurring combination of antecedents'values for calc_delta_pnldiff_z=1 seems to be
# exptoday:1
# ccy:jpy
```

In [ ]:
```

```