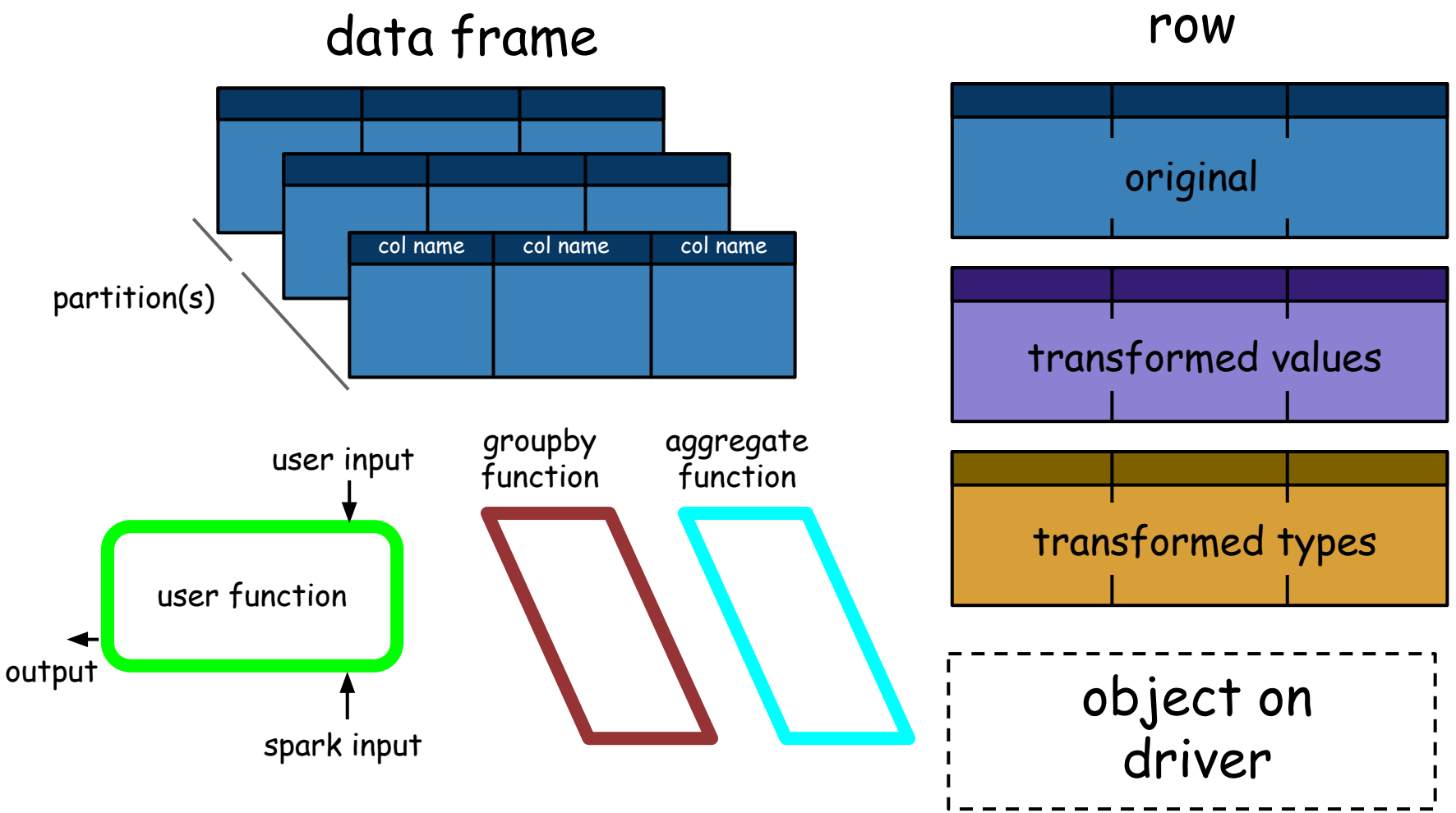
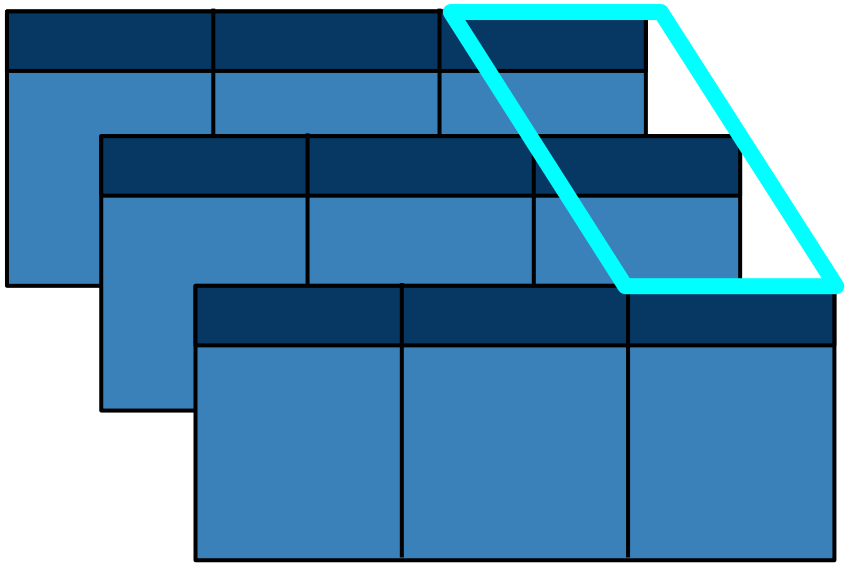
pyspark-pictures data frames

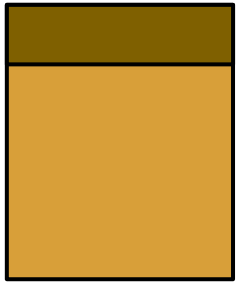
Learn the pyspark API through pictures and simple examples

https://github.com/jkthompson/pyspark-pictures

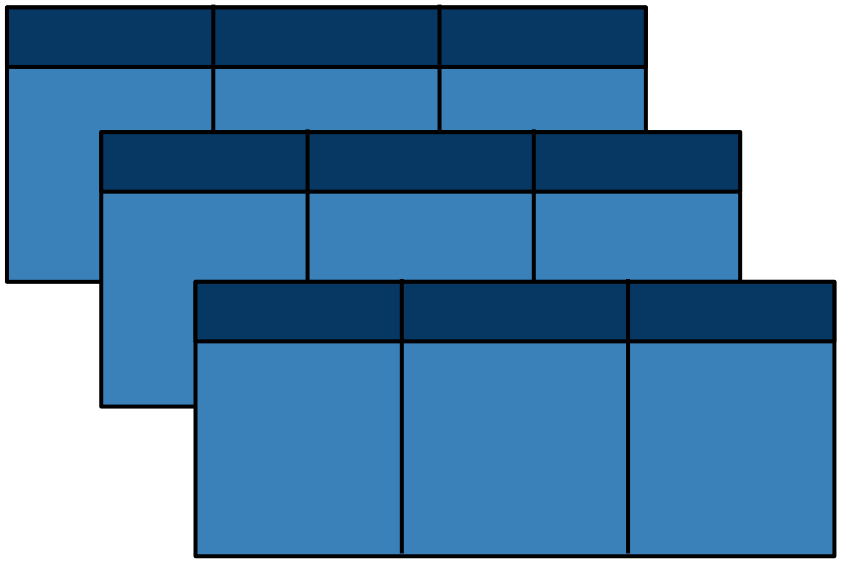

data frame
col name
col name
col name
partition(s)
row
original
transformed values
transformed types
user input
user function
output
spark input
groupby function
aggregate function
object on driver

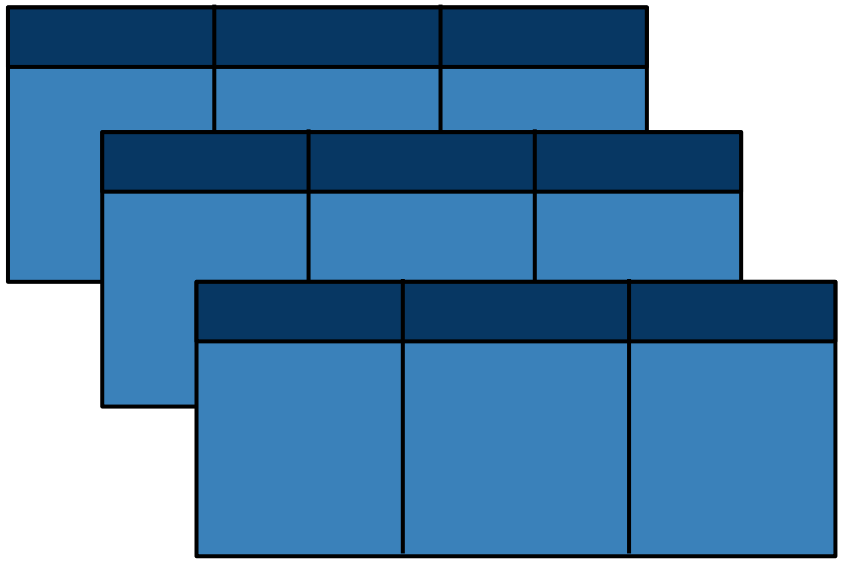
agg



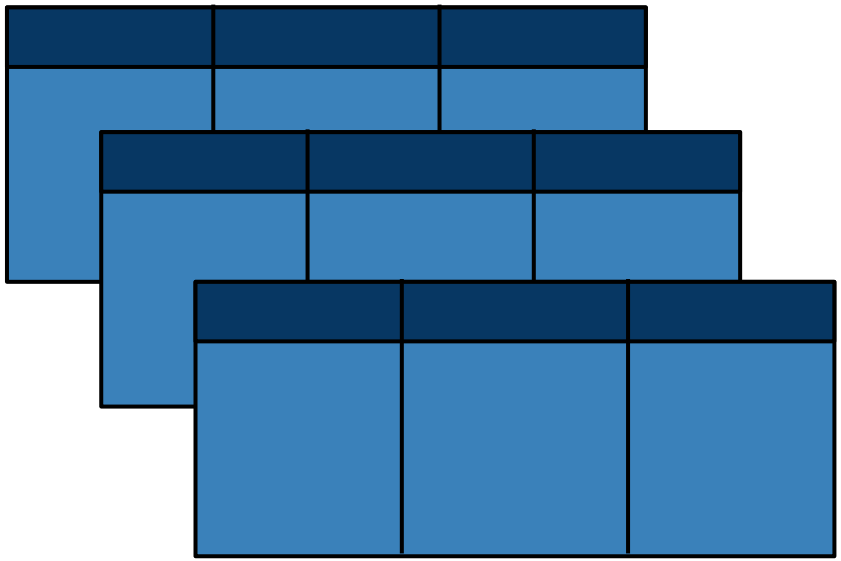


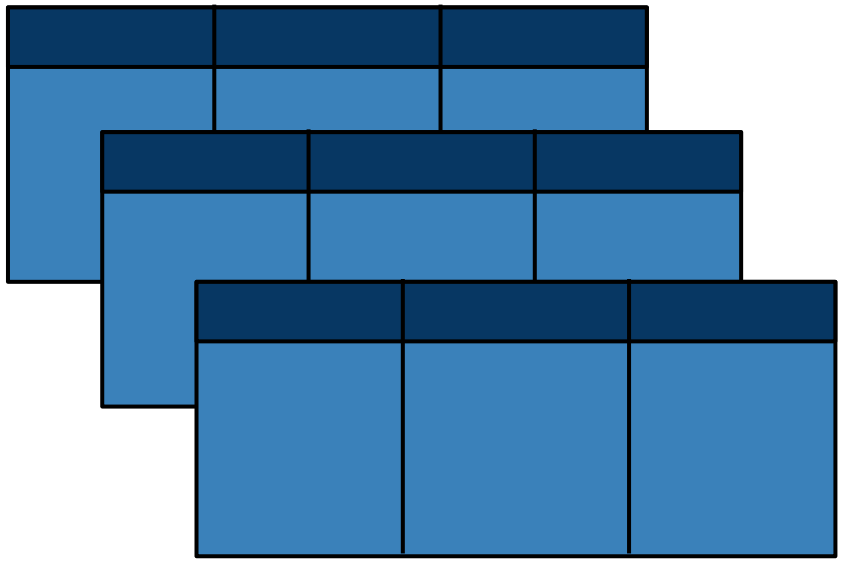
alias





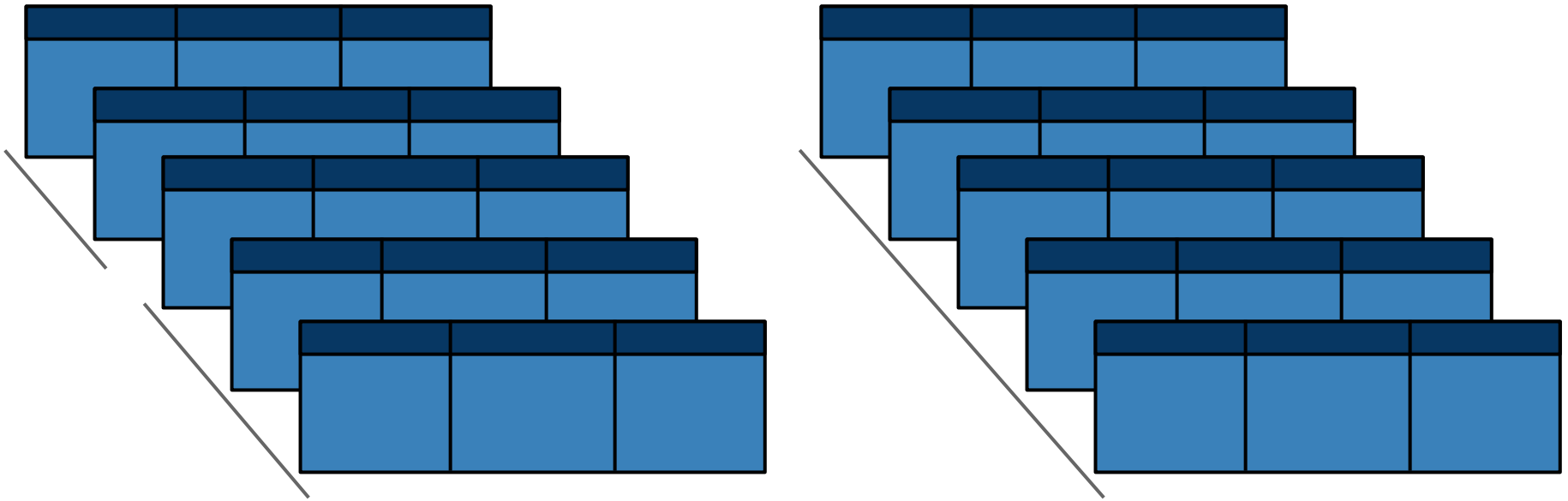
cache



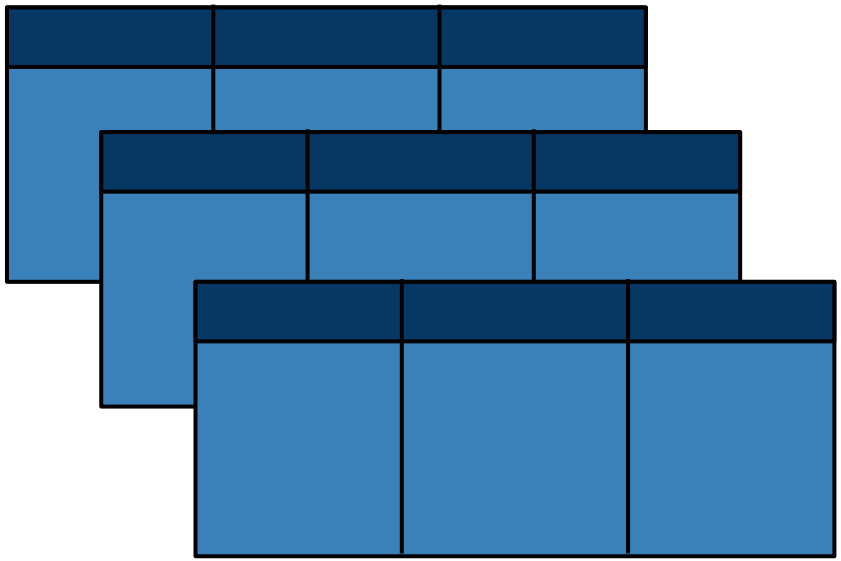


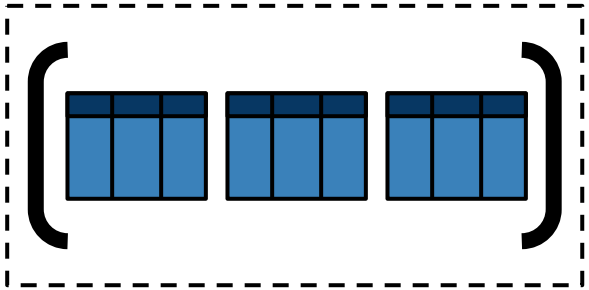
coalesce

numPartitions = 1

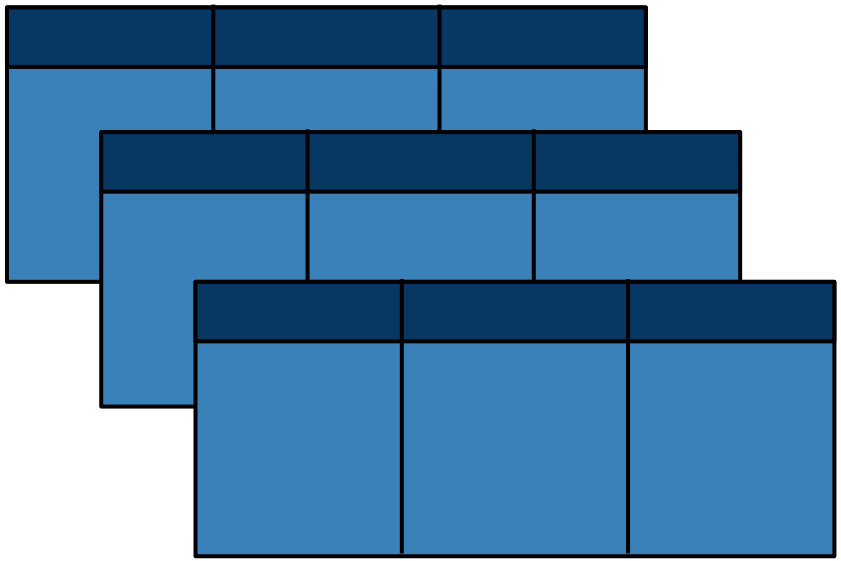


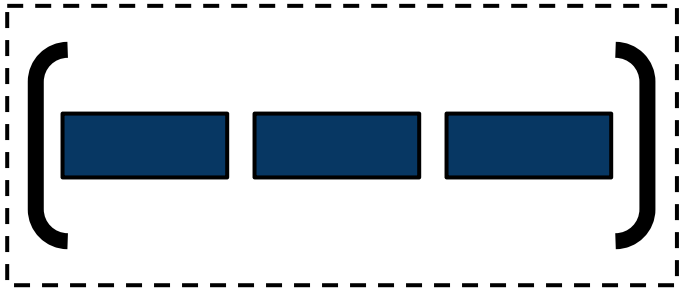
collect



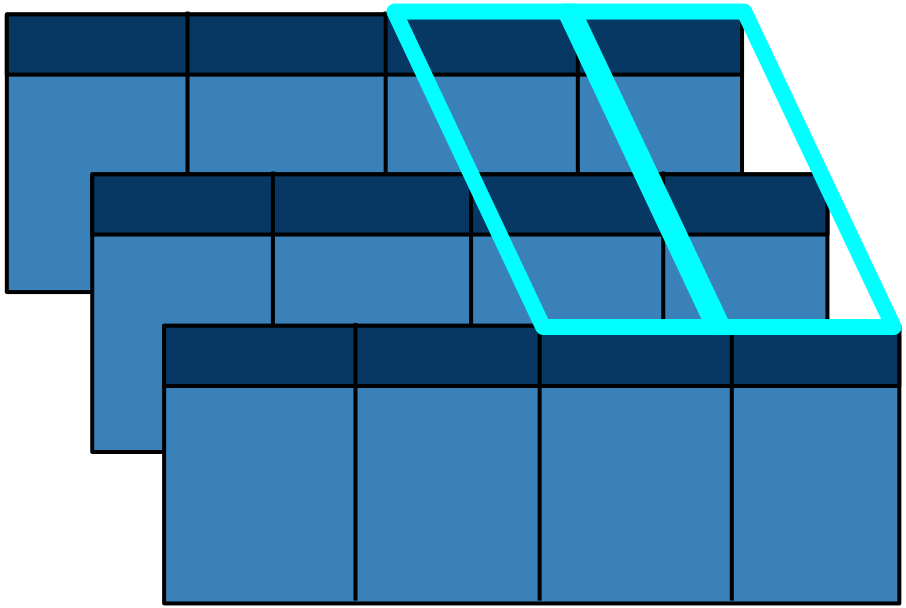


columns





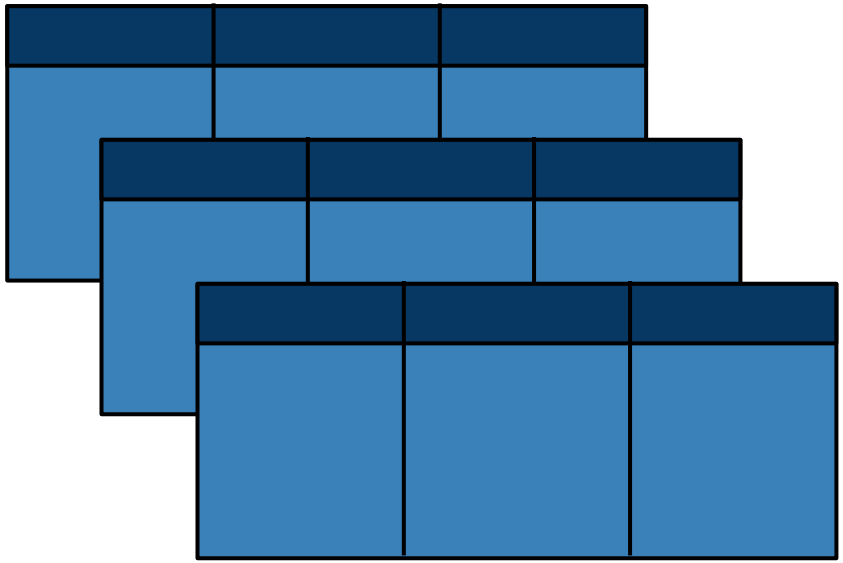
corr



Pearson's r

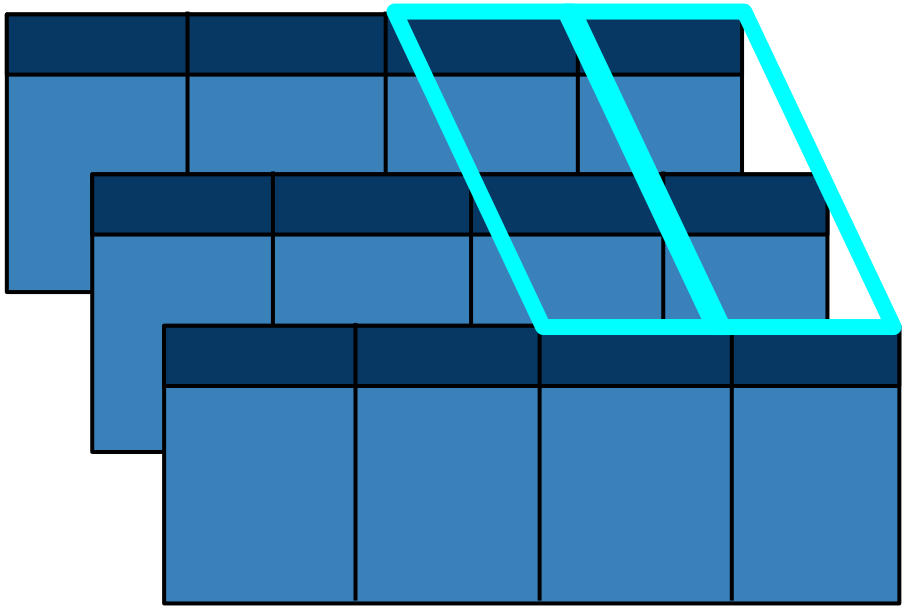
$$r = \frac{\sum_i (A_i - \bar{A})(C_i - \bar{C})}{\sqrt{\sum_i (A_i - \bar{A})^2} \sqrt{\sum_i (C_i - \bar{C})^2}}$$

count



3

cov

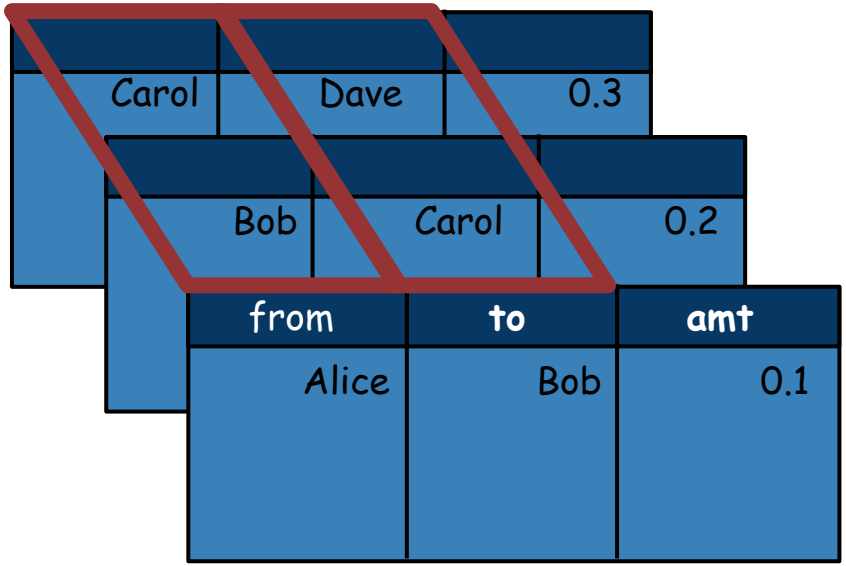


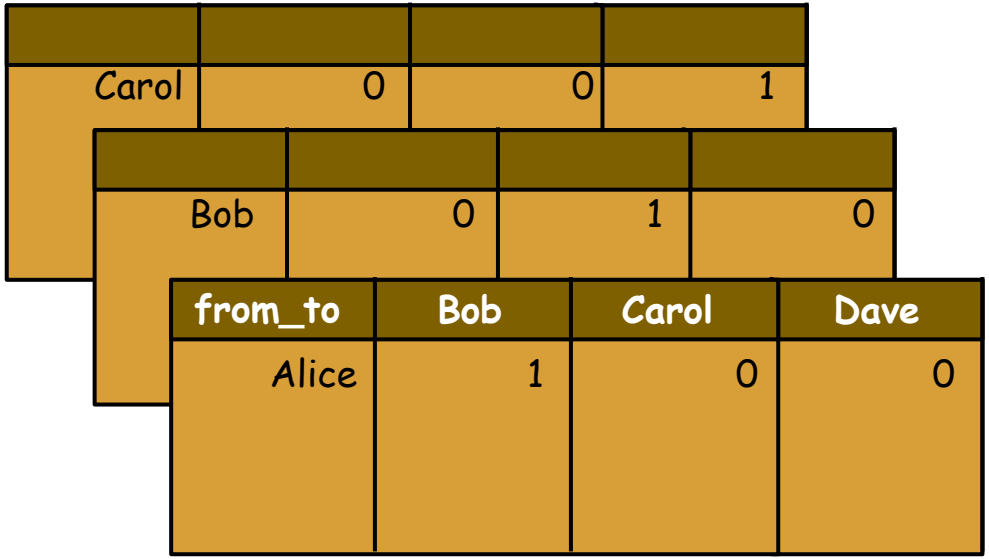
Sample Covariance

$$\frac{1}{N-1}\sum_i (A_i - \bar{A})(C_i - \bar{C})$$

crosstab

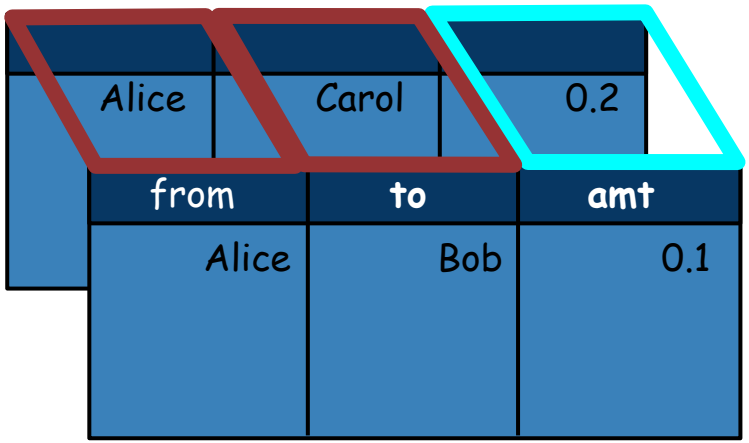
col1 = 'from' col2 = 'to'

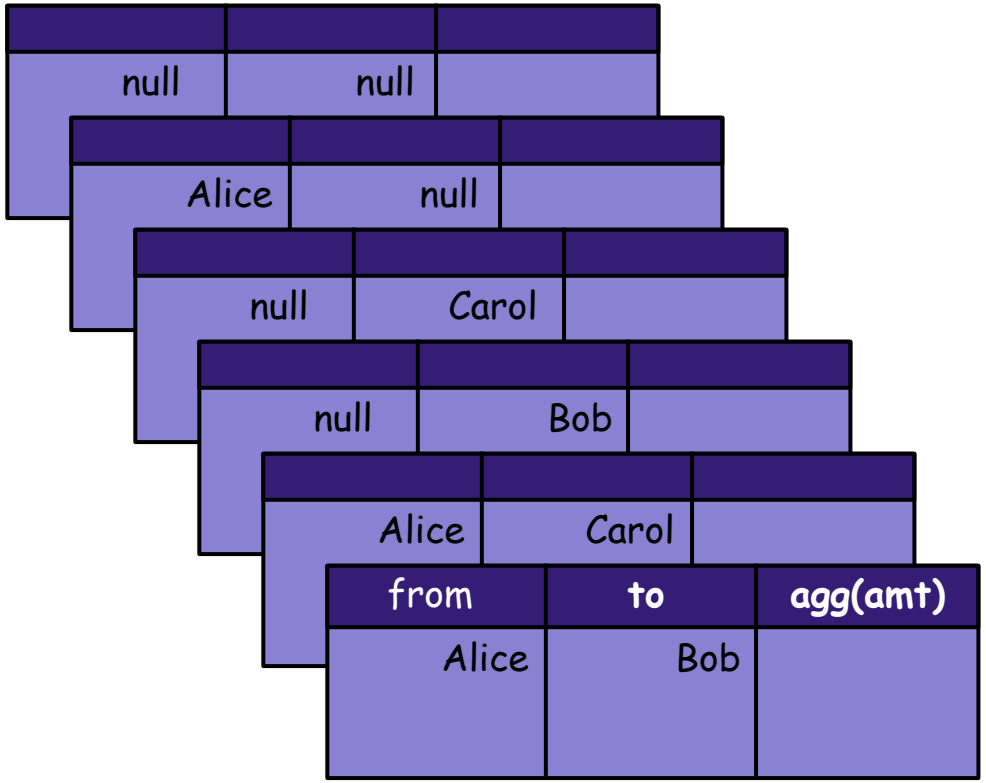




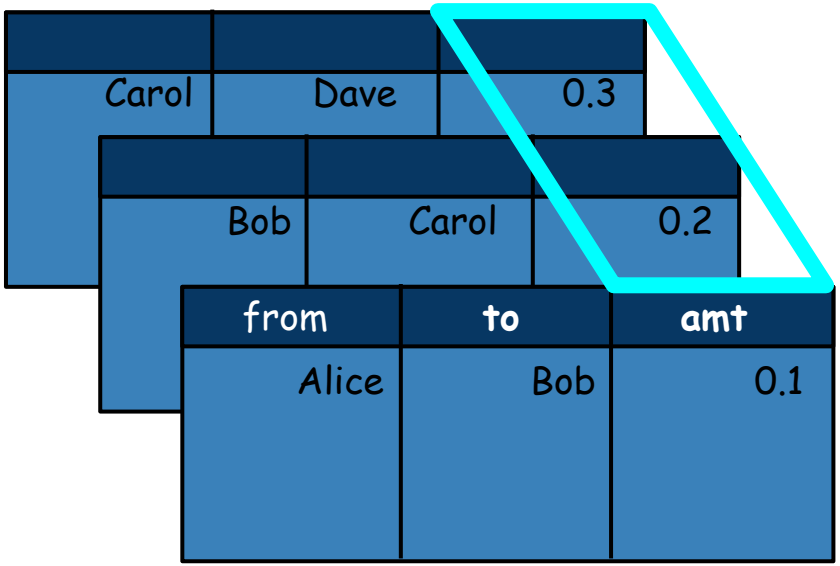
cube

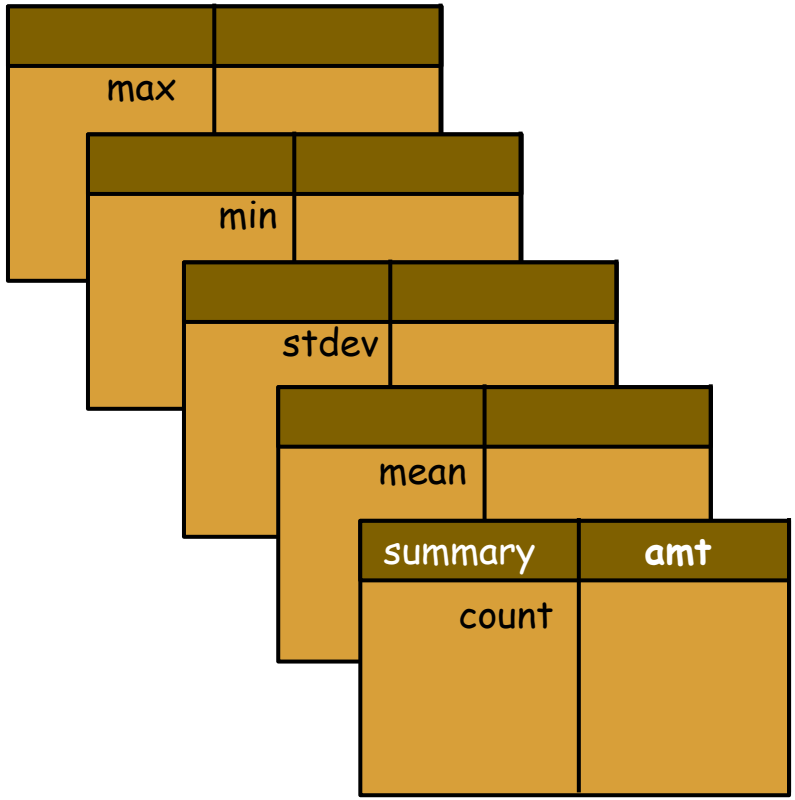
*cols = ‘from’, ‘to’



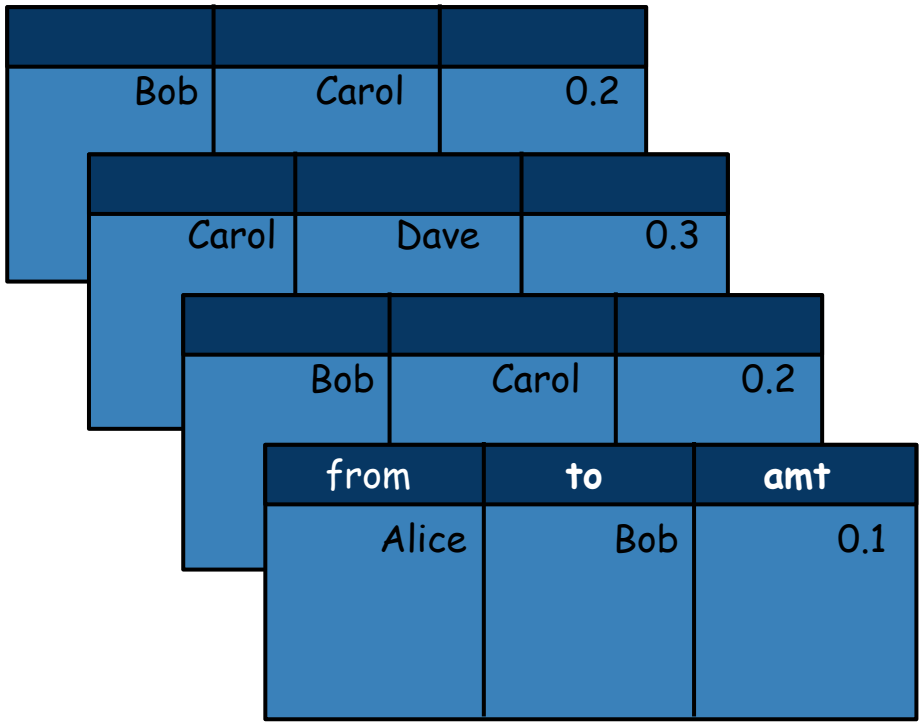


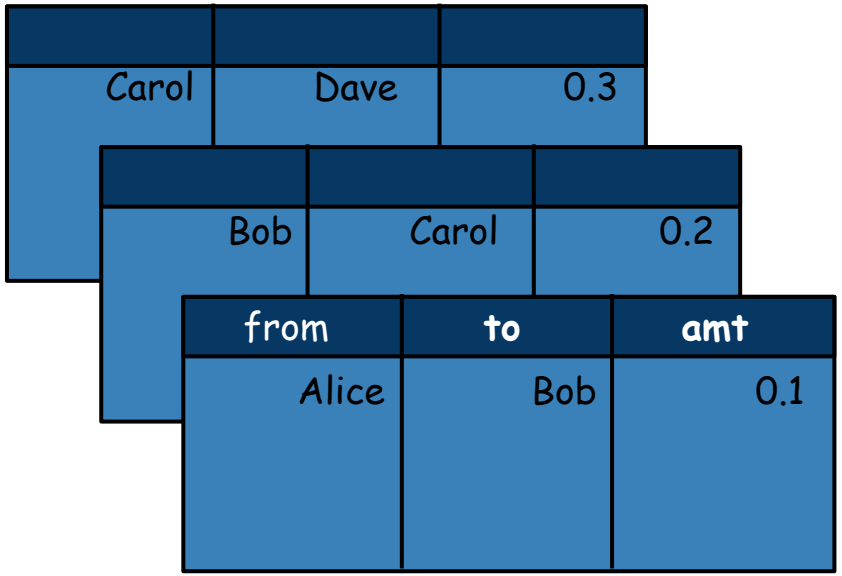
describe


Carol
Dave
0.3
Bob
Carol
0.2
from
to
amt
Alice
Bob
0.1


max
min
stdev
mean
summary
amt
count

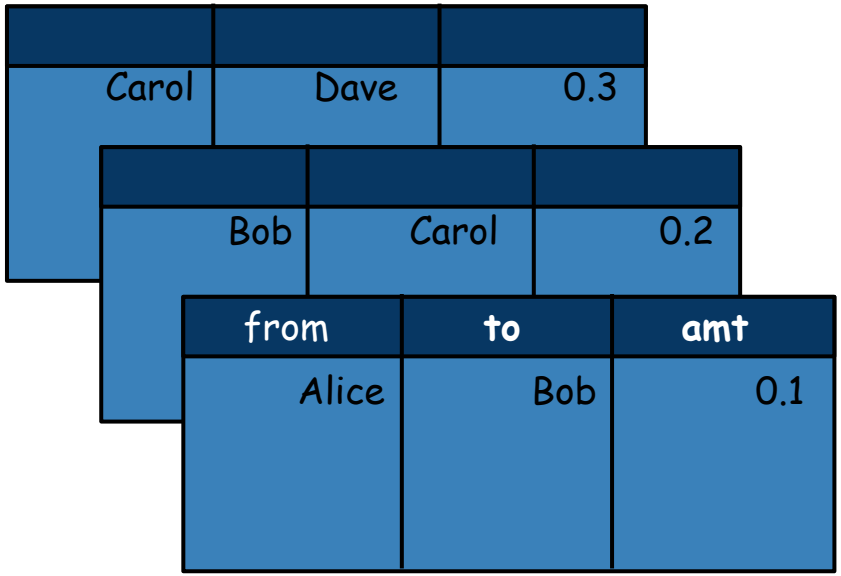
distinct

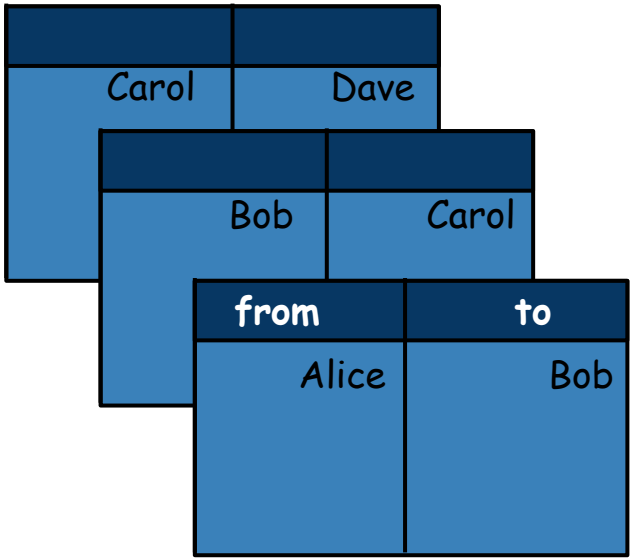

Bob Carol 0.2
Carol Dave 0.3
Bob Carol 0.2
from to amt
Alice Bob 0.1


Carol Dave 0.3
Bob Carol 0.2
from to amt
Alice Bob 0.1

drop

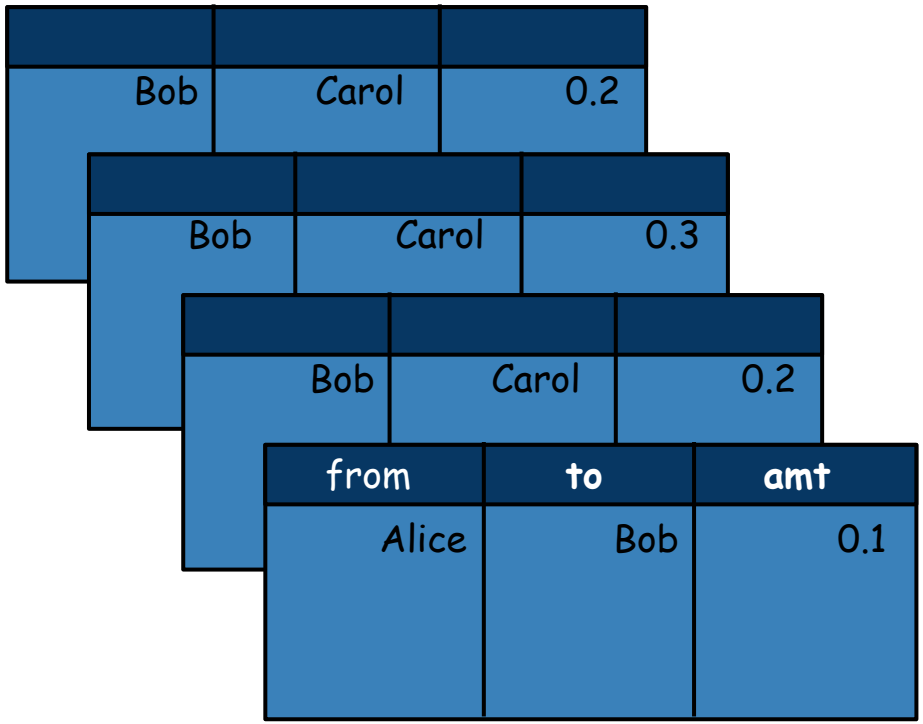
col = 'amt'

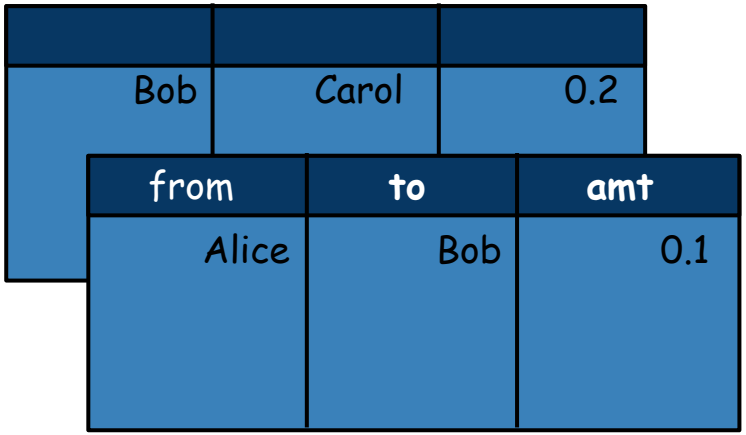

Carol
Dave
0.3
Bob
Carol
0.2
from
to
amt
Alice
Bob
0.1


Carol
Dave
Bob
Carol
from
to
Alice
Bob

dropDuplicates

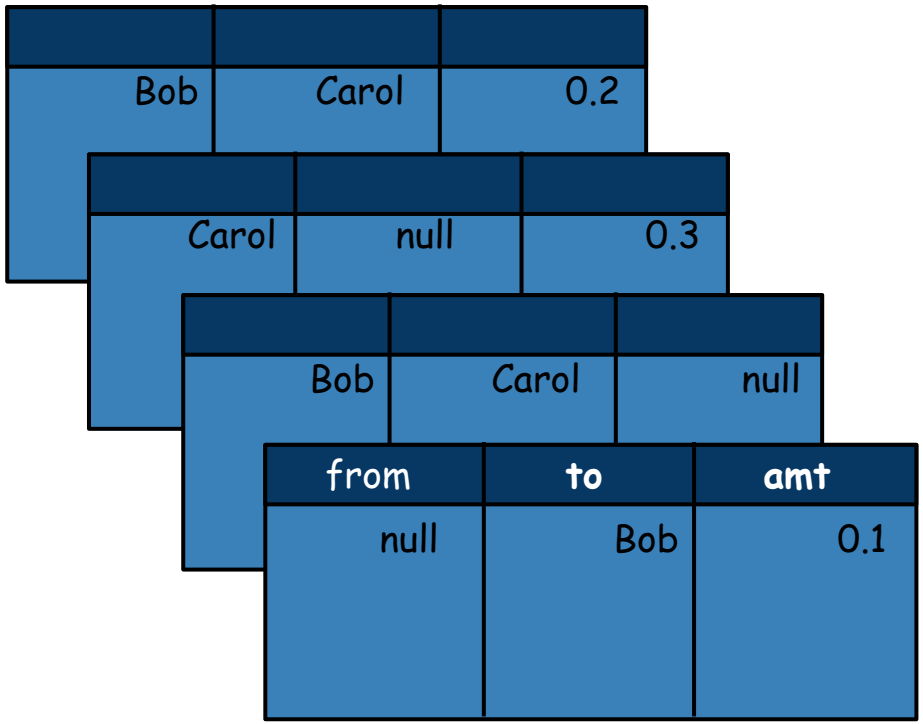
subset = ['from','to']

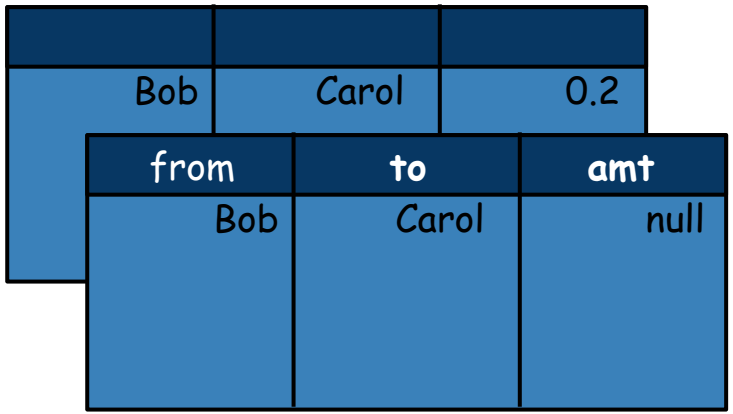




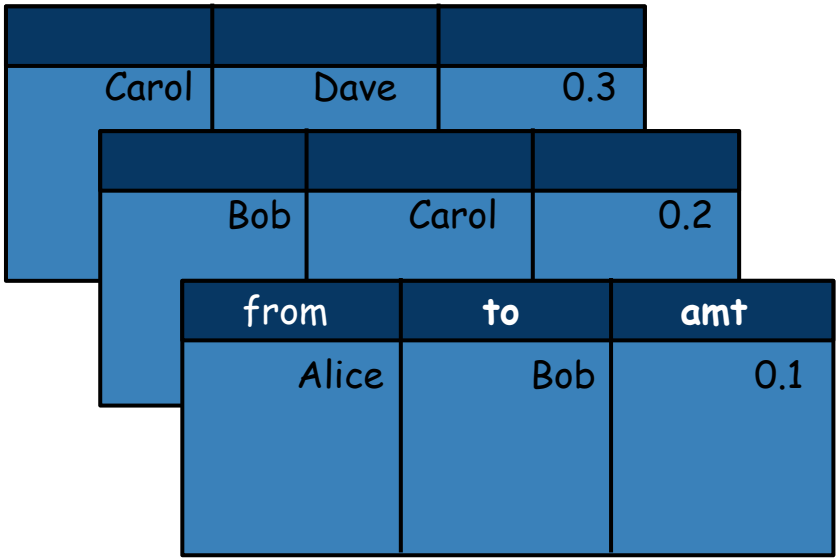
dropna

how = 'any' subset = ['from', 'to']





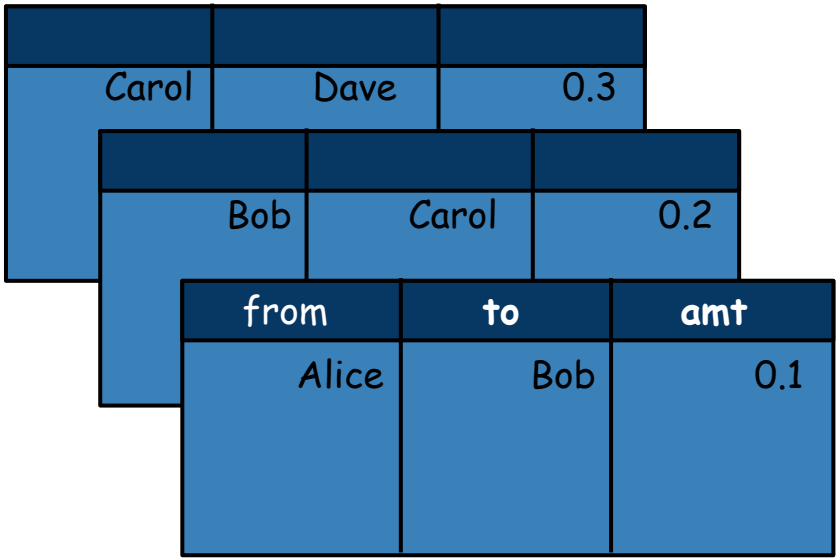
dtypes



[('from','string'), ('to', 'string'), ('amt', 'double')]

explain

extended = True




```
== Parsed Logical Plan ==
...
== Analyzed Logical Plan ==
...
== Optimized Logical Plan ==
...
== Physical Plan ==
...
== RDD ==
```

fillna

value = 'unknown' subset = ['from', 'to']

Carol	null	0.3

Bob	Carol	nan

from	to	amt
null	Bob	0.1

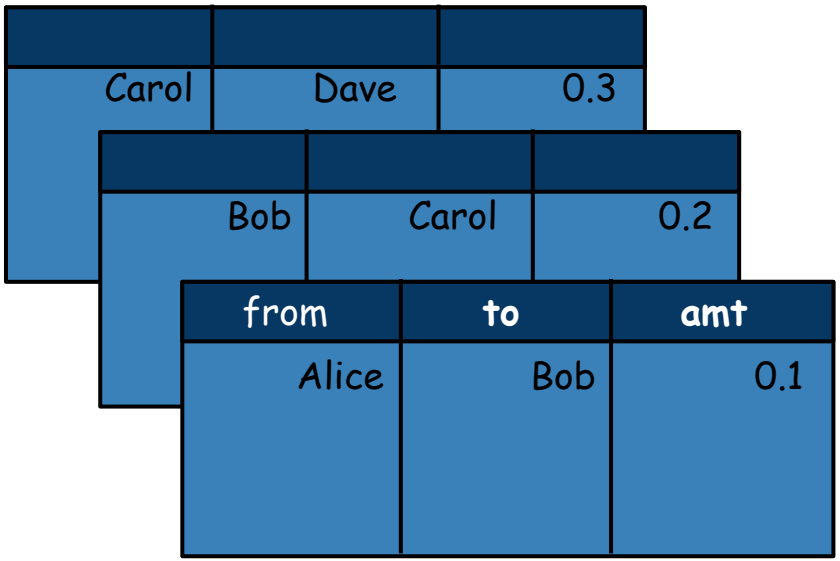
Carol	unknown	0.3

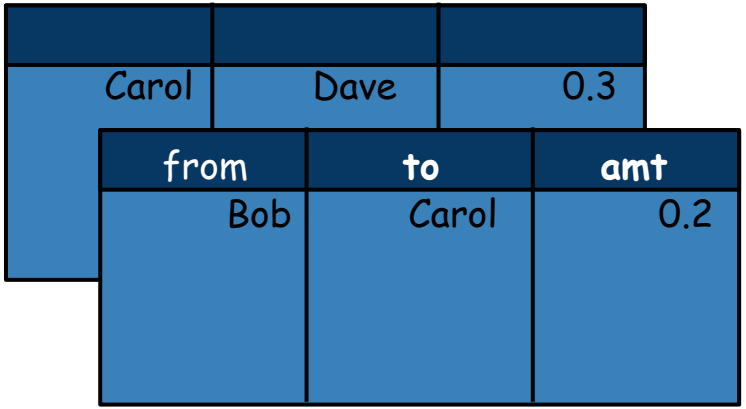
Bob	Carol	nan

from	to	amt
unknown	Bob	0.1

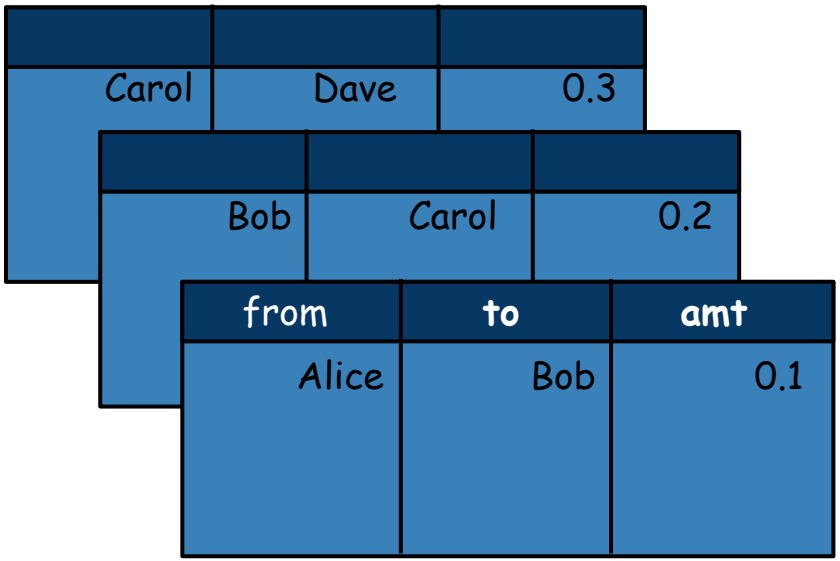
filter

condition = "amt > 0.1"



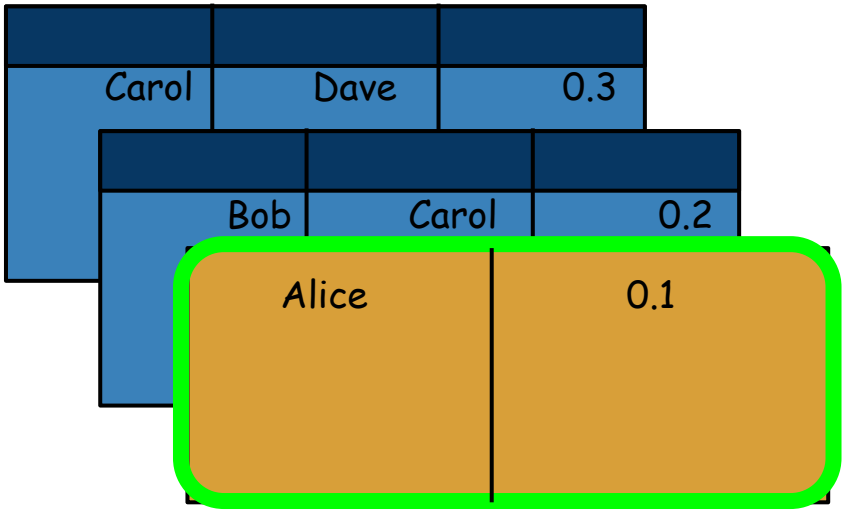


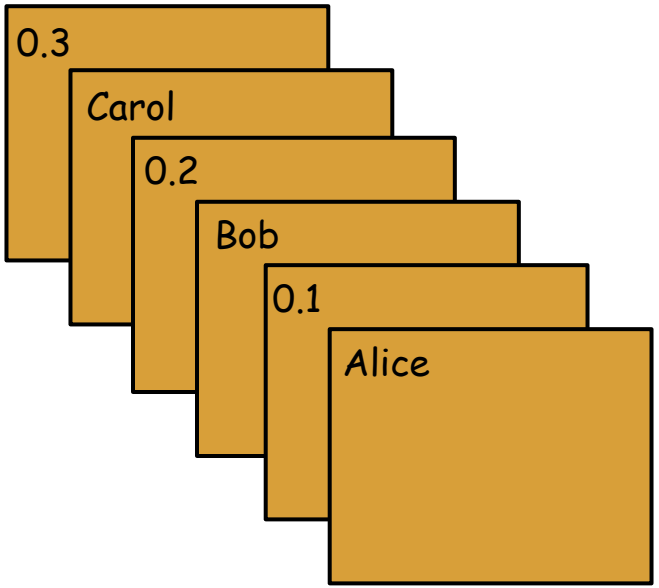
first



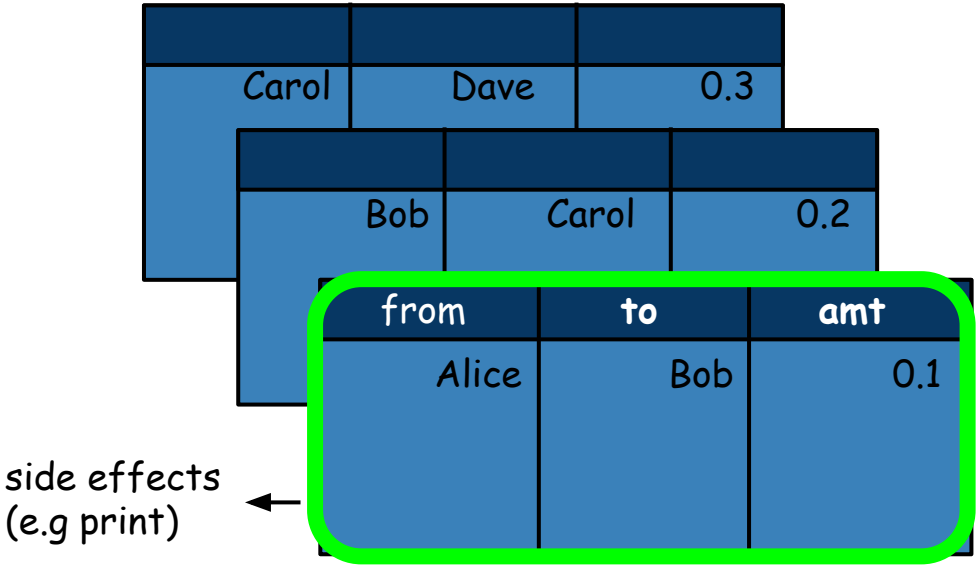
Row(from='Alice', to='Bob', amt=0.1)

flatMap



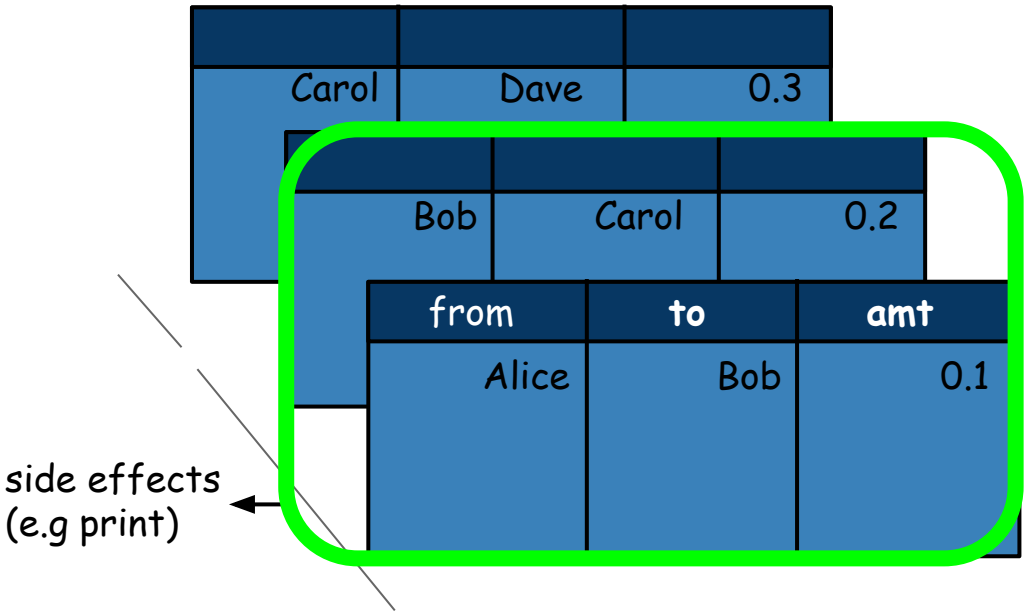


foreach



*no return value, original DataFrame unchanged

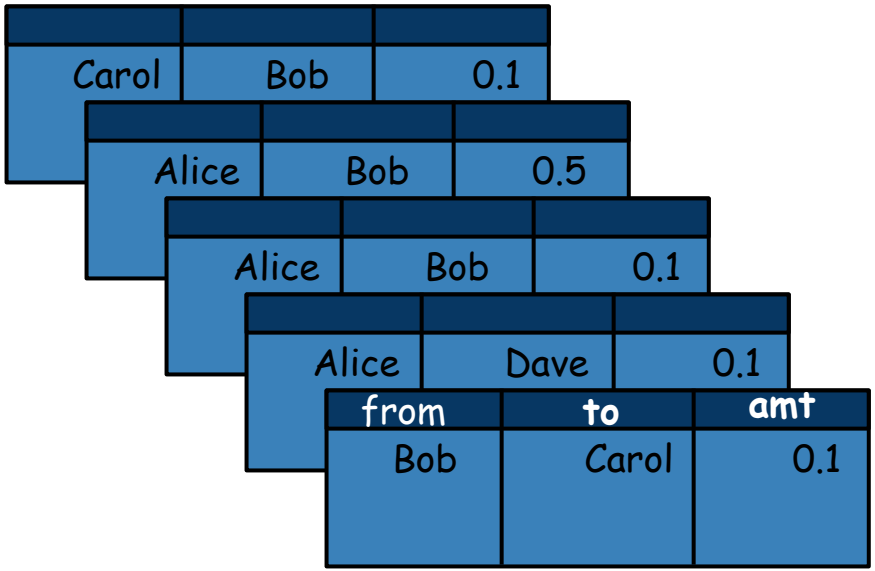
foreachPartition



*no return value, original DataFrame unchanged

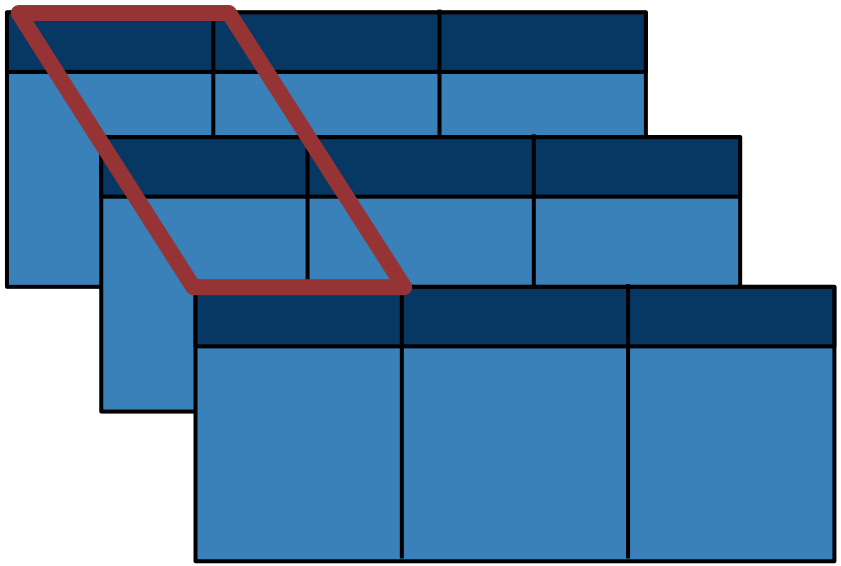
freqItems

cols = ['from', 'amt'] support = 0.8



from_freqItems	amt_freqItems
[Alice]	[0.1]

groupBy (groupby)

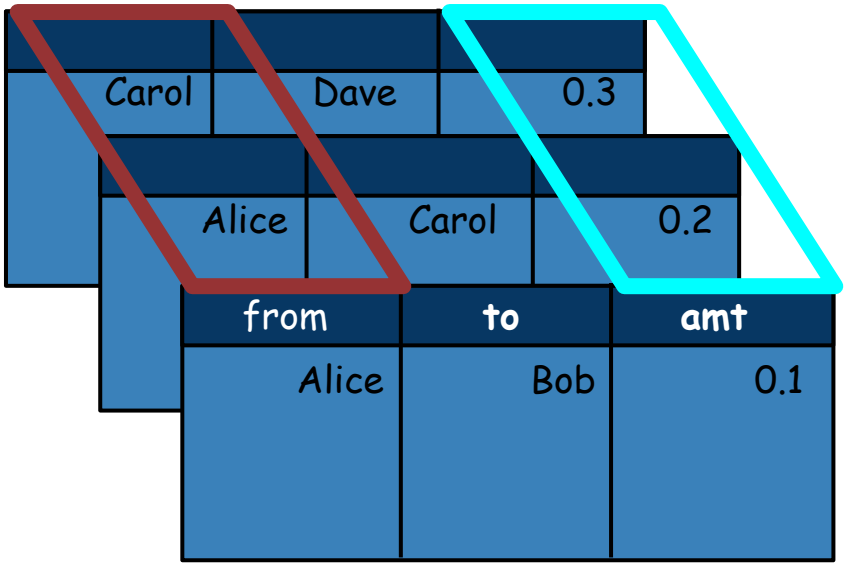


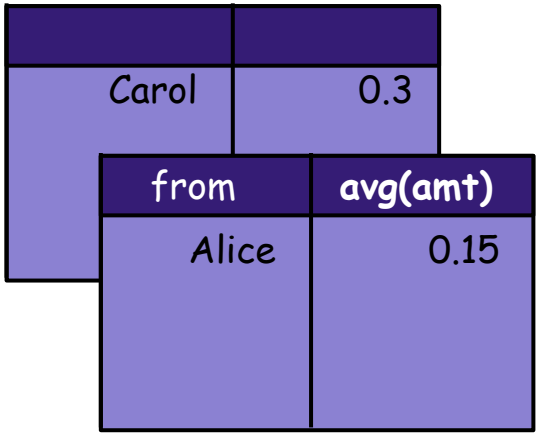
GroupedData Object

with methods: agg, avg, count, max, mean, min, pivot, sum

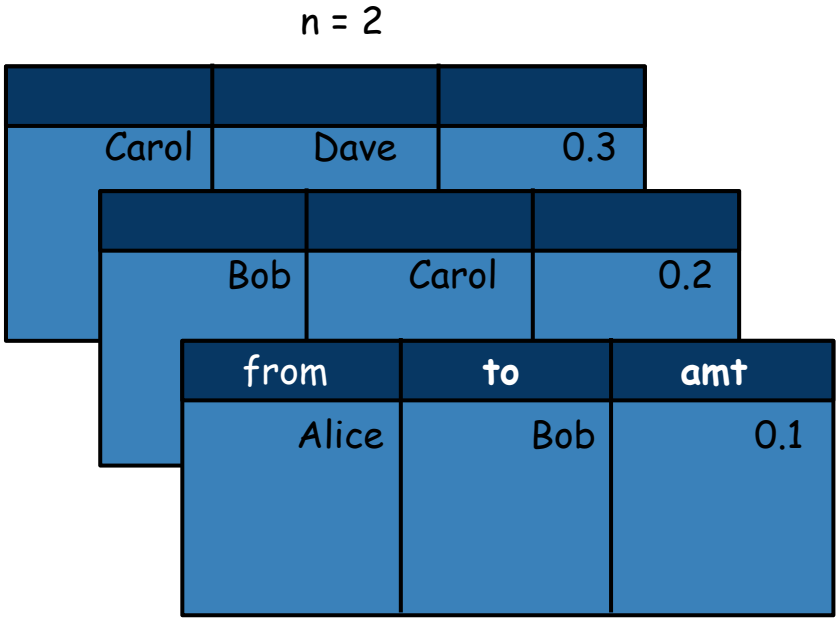
groupBy(col1).avg(col2)

col1 = 'from' col2 = 'amt'



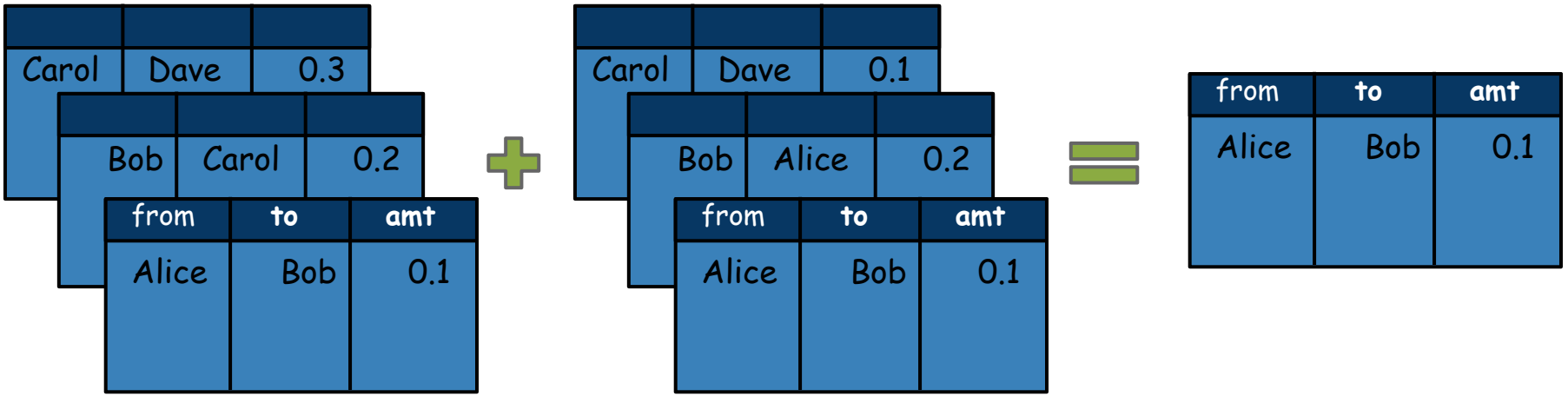


head

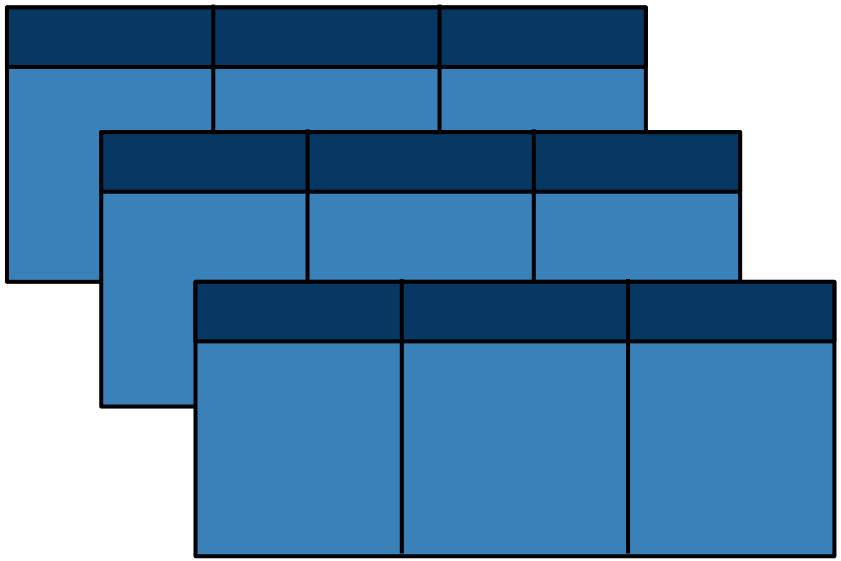


[Row(from=u'Alice', to=u'Bob', amt=0.1),
Row(from=u'Bob', to=u'Carol', amt=0.2)]

intersect


Carol
Dave
0.3
Bob
Carol
0.2
from
to
amt
Alice
Bob
0.1
Carol
Dave
0.1
Bob
Alice
0.2
from
to
amt
Alice
Bob
0.1
from
to
amt
Alice
Bob
0.1

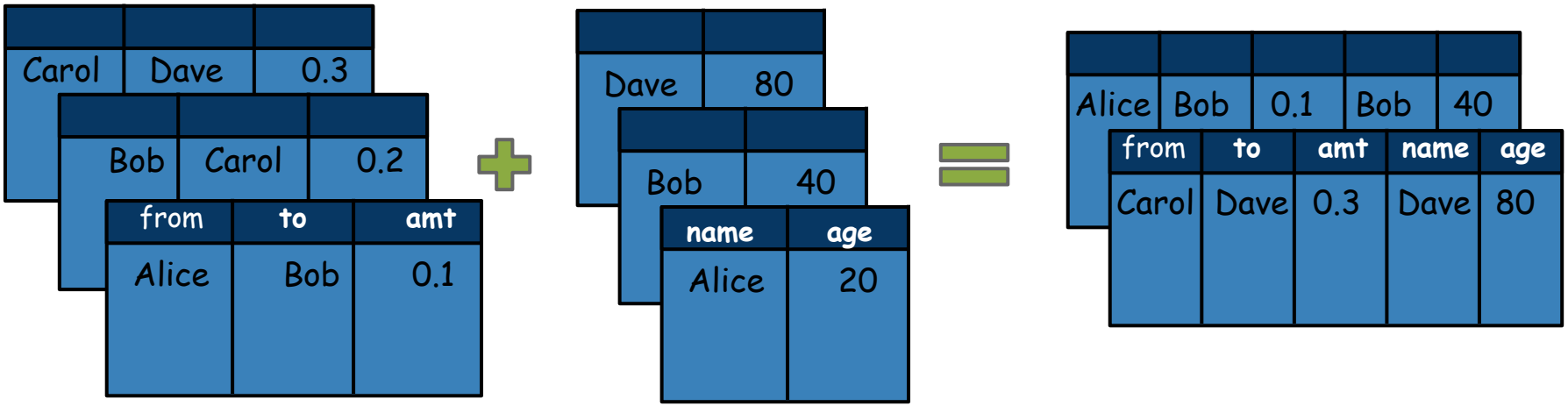
isLocal



False

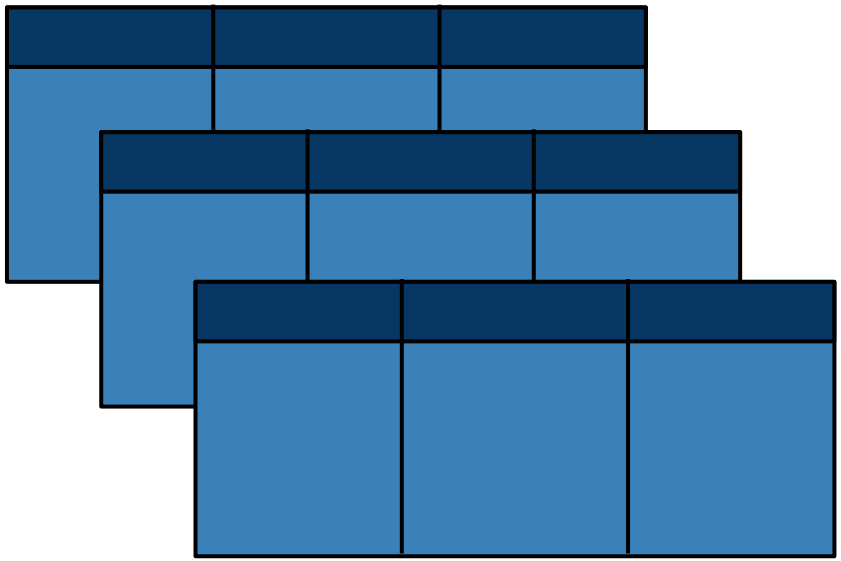
join

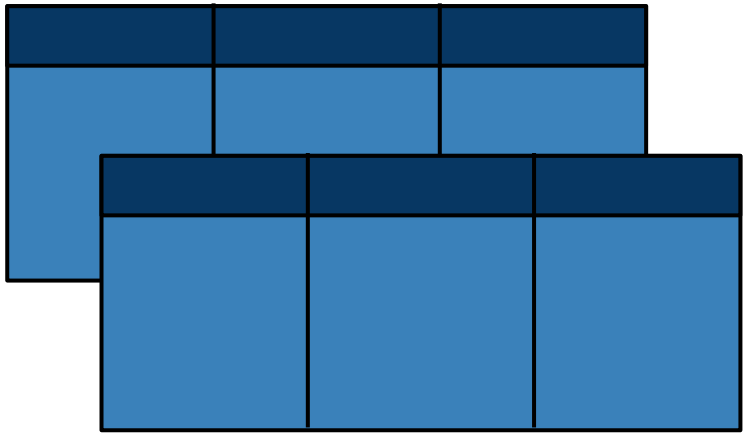
joinExprs = x.to==y.name joinType = 'inner'



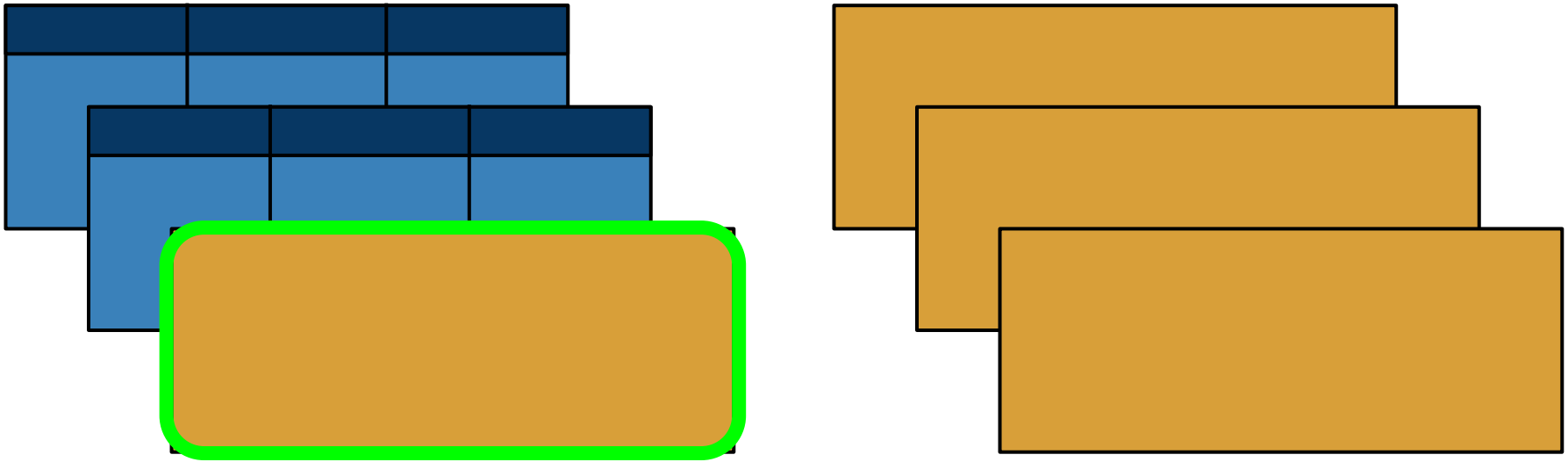
limit

num = 2

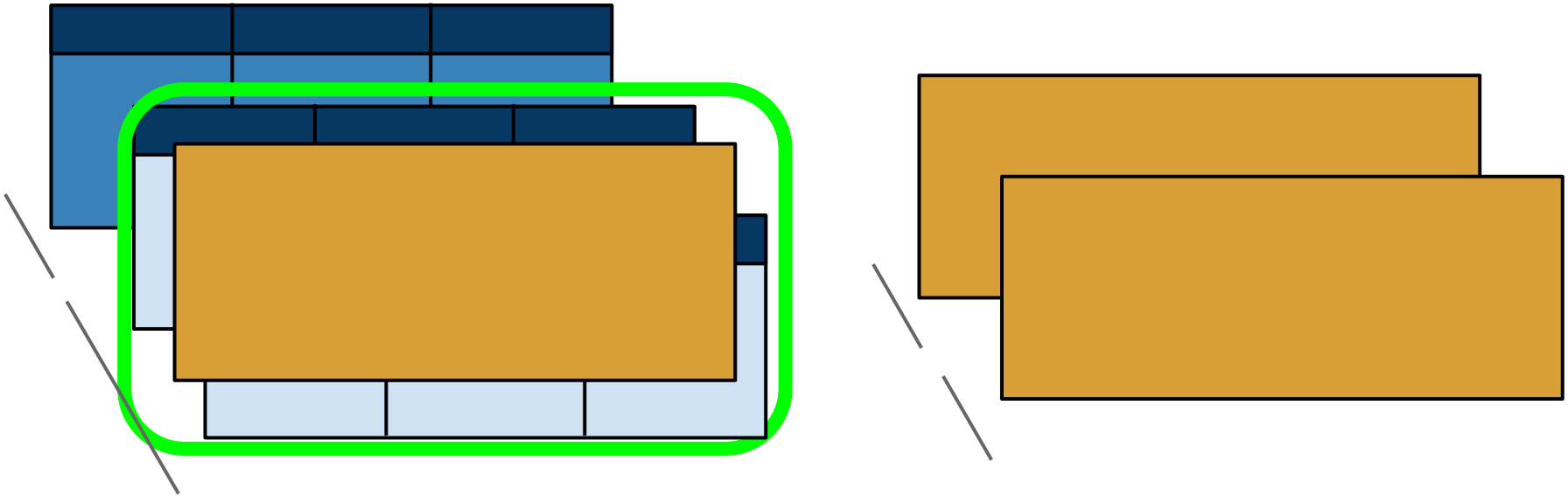




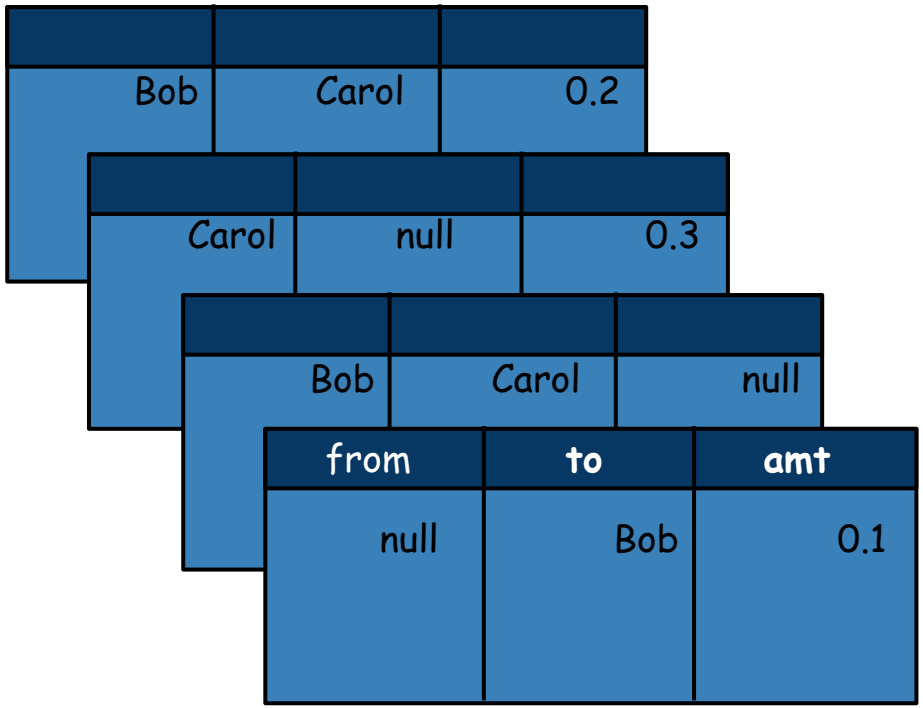
map



mapPartitions



na

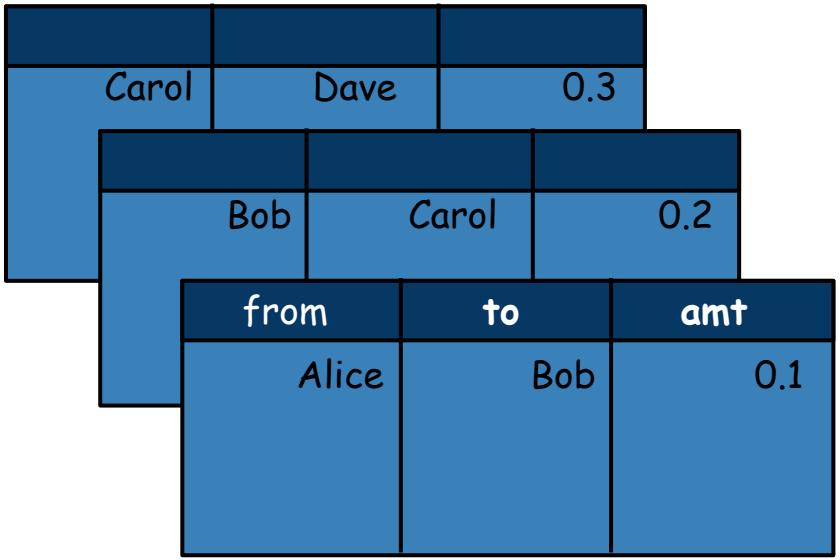


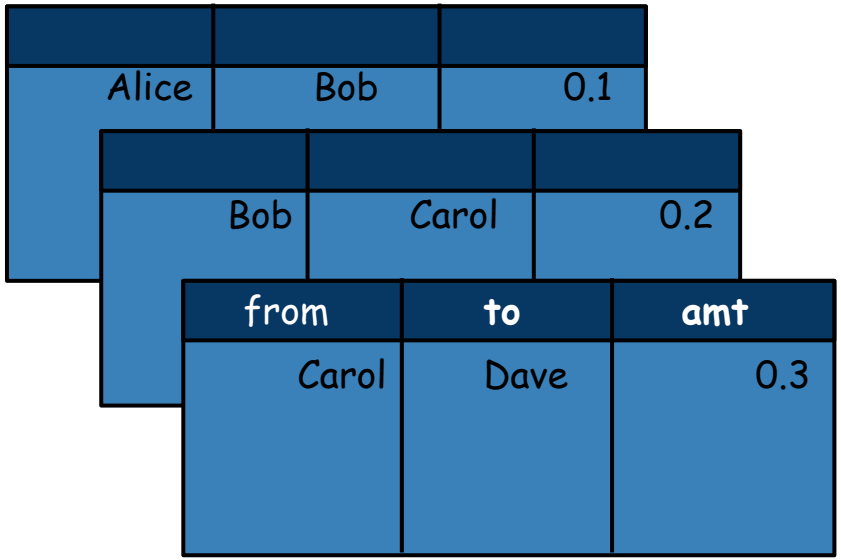
DataFrameNaFunctions Object

with methods: drop, fill, replace

orderBy

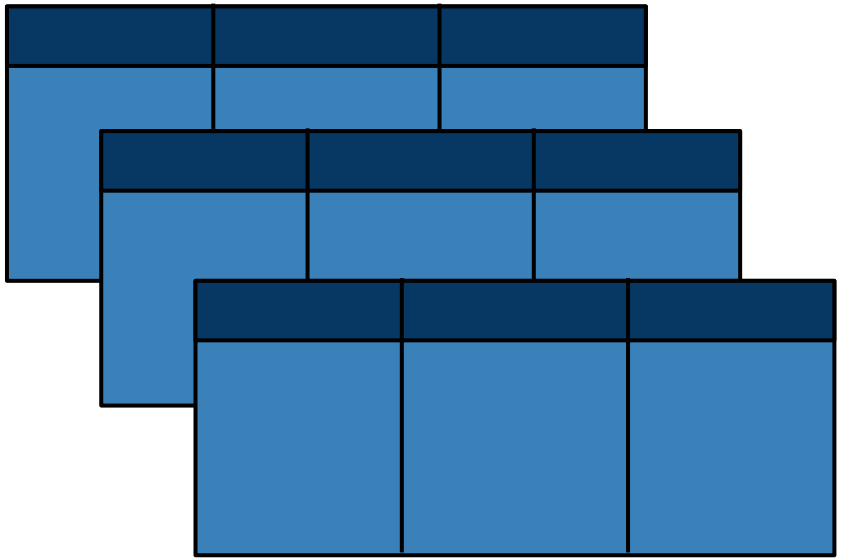
cols = ['from'], ascending = [False]

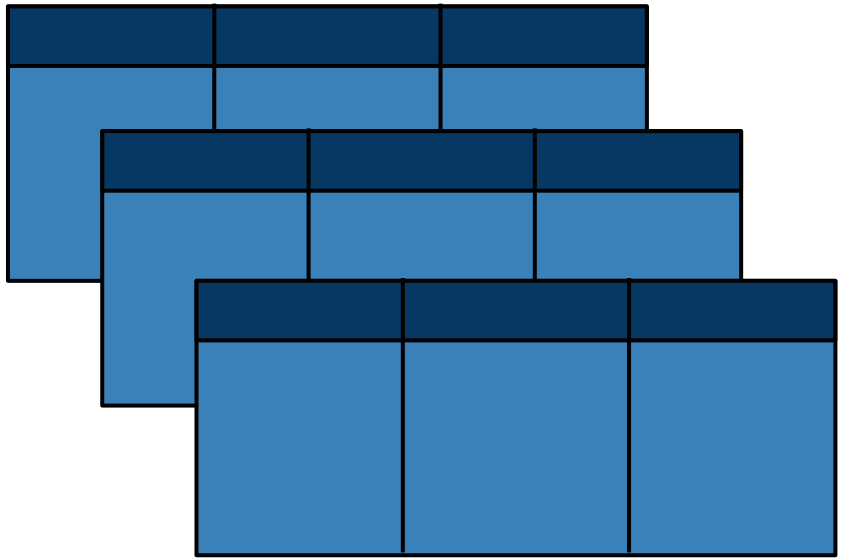




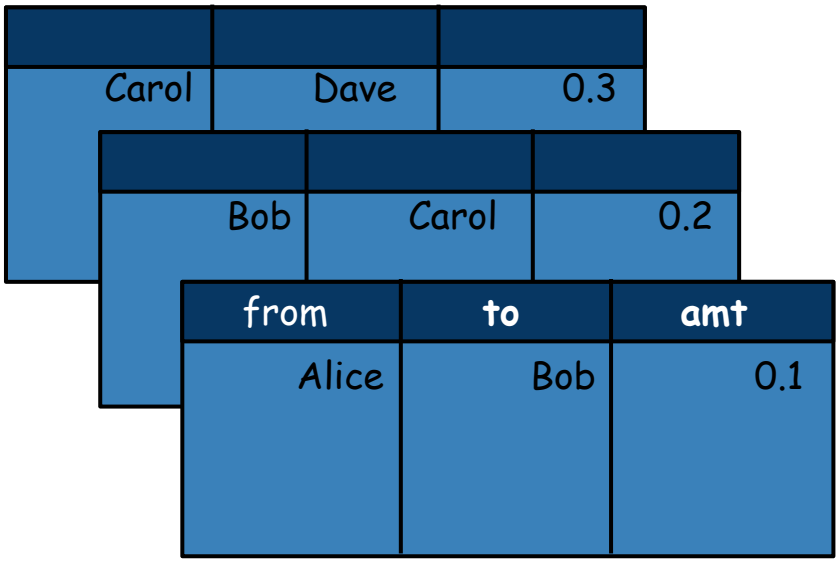
persist

strorageLevel =
StorageLevel(MEMORY_ONLY_SER)





printSchema

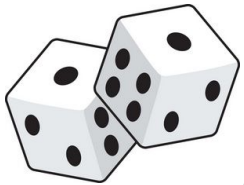


stdout

```
root
 |-- from: string (nullable = true)
 |-- to: string (nullable = true)
 |-- amt: double (nullable = true)
```

randomSplit

weights = [0.5,0.5]



Carol	Dave	0.3

Bob	Carol	0.2

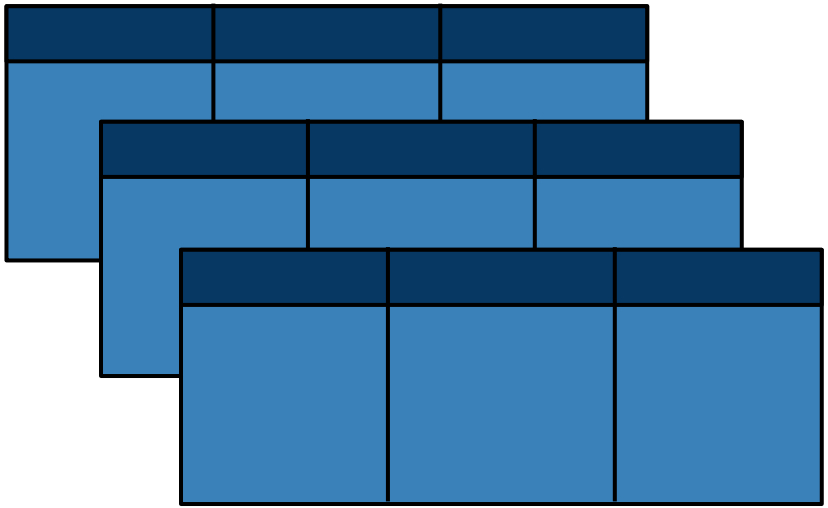
from	to	amt
Alice	Bob	0.1

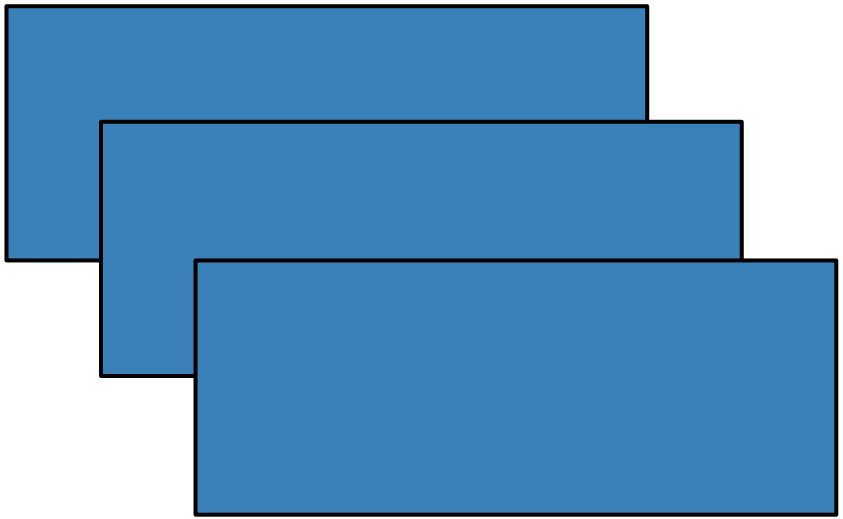
Carol	Dave	0.3

from	to	amt
Bob	Carol	0.2

from	to	amt
Alice	Bob	0.1

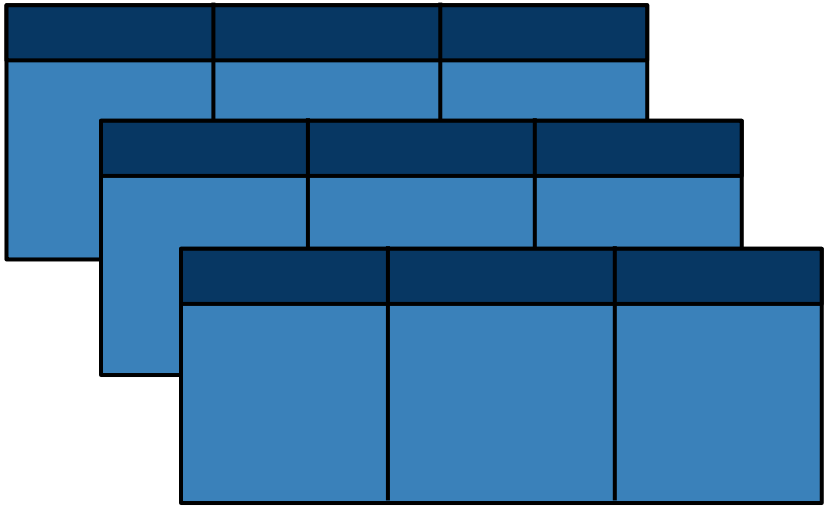
rdd



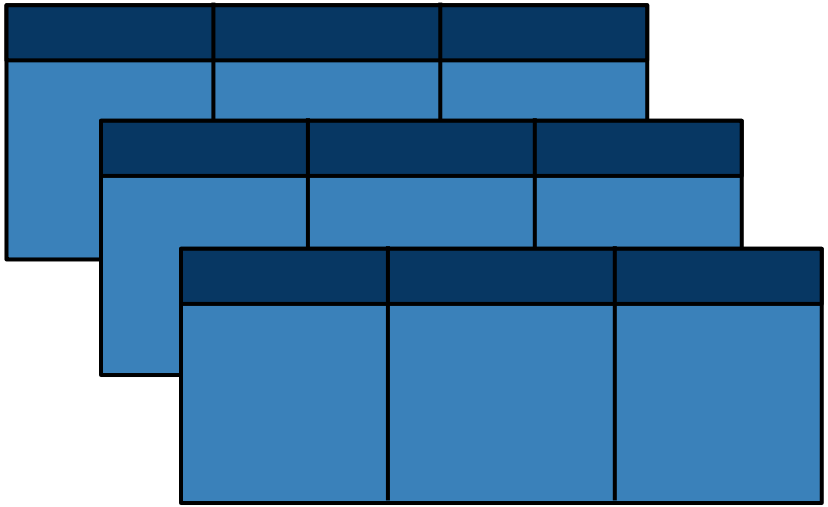


registerTempTable

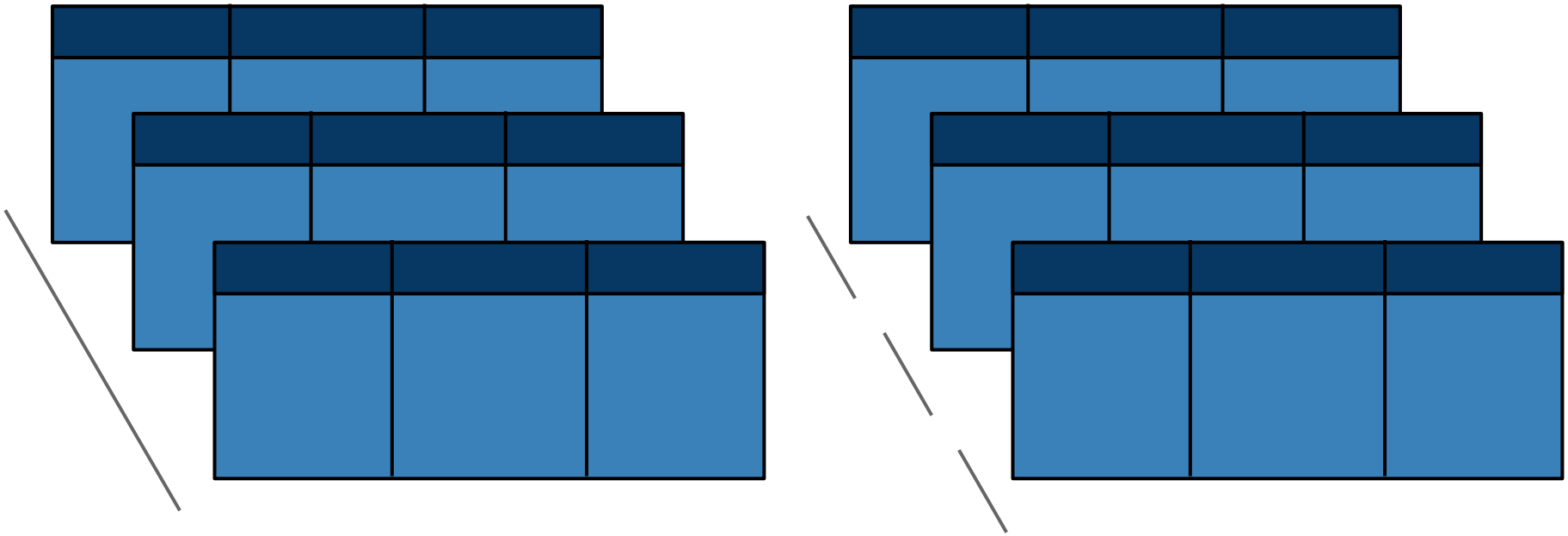
name = "TRANSACTIONS"



TRANSACTIONS

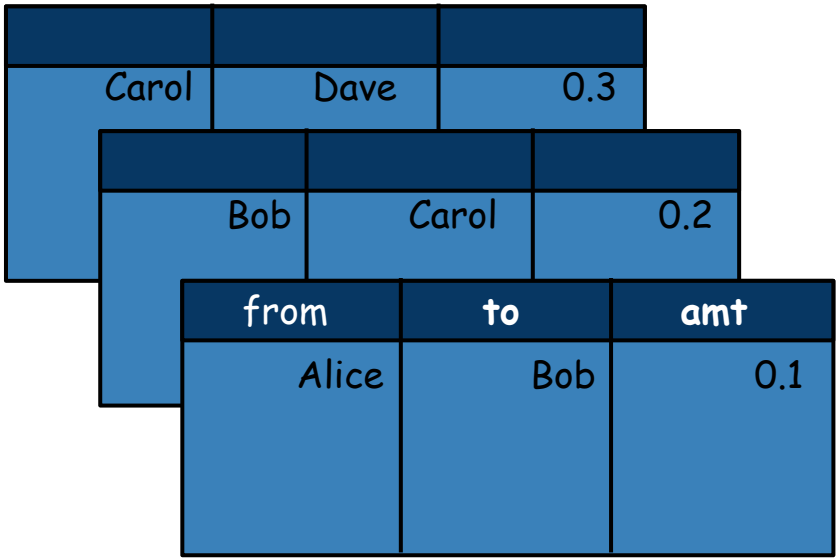


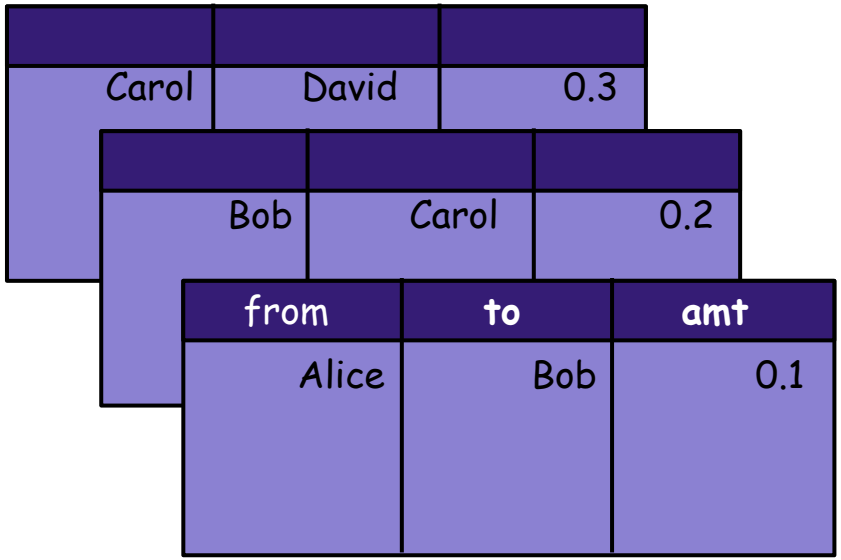
repartition



replace

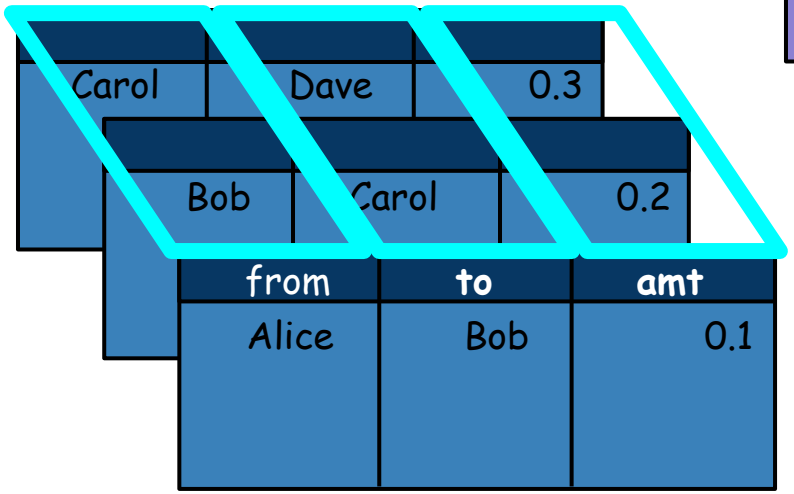
to_replace = 'Dave' value = 'David'

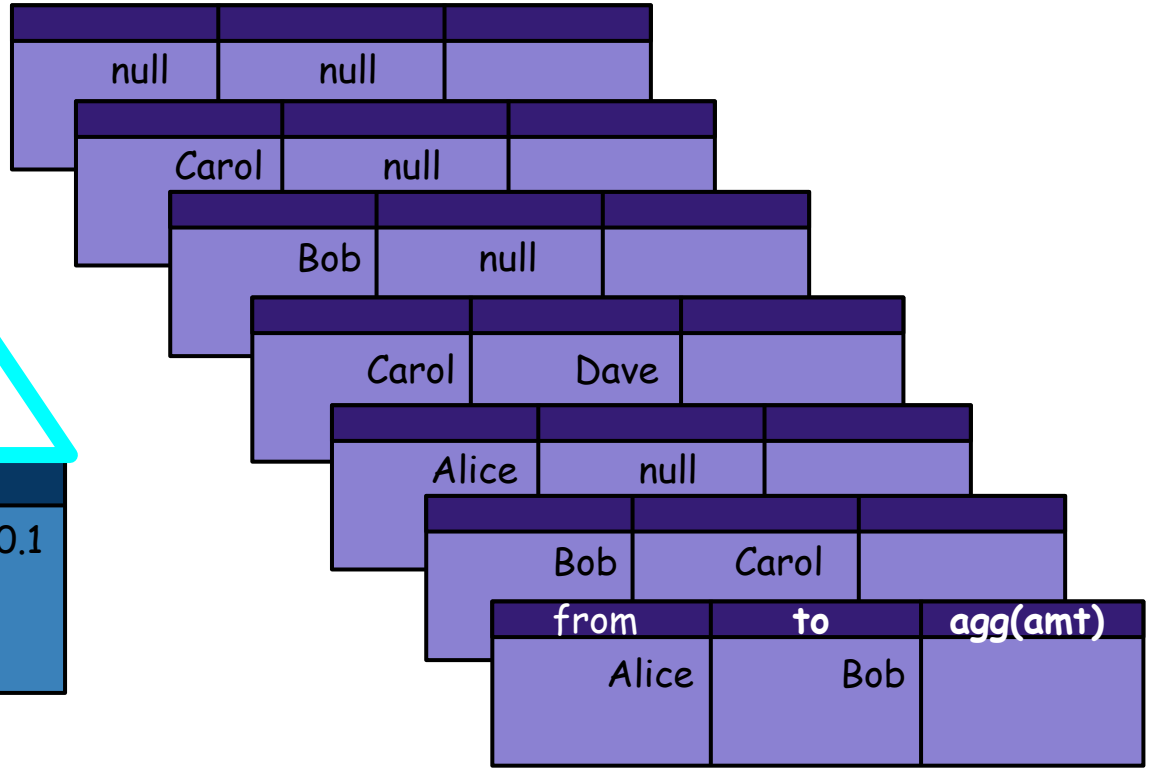




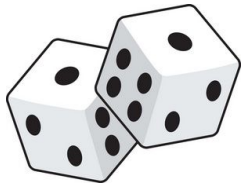
rollup

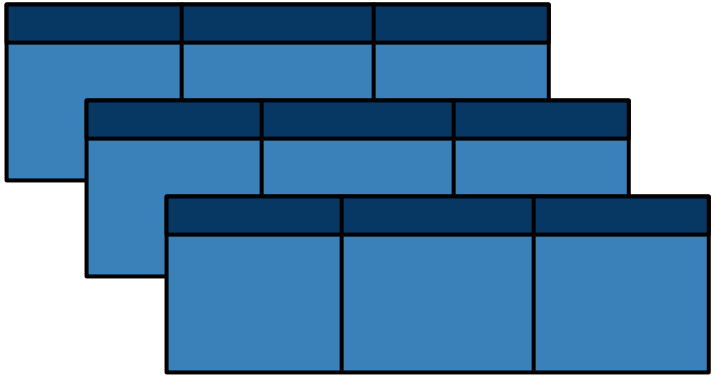
cols = ['from','to']

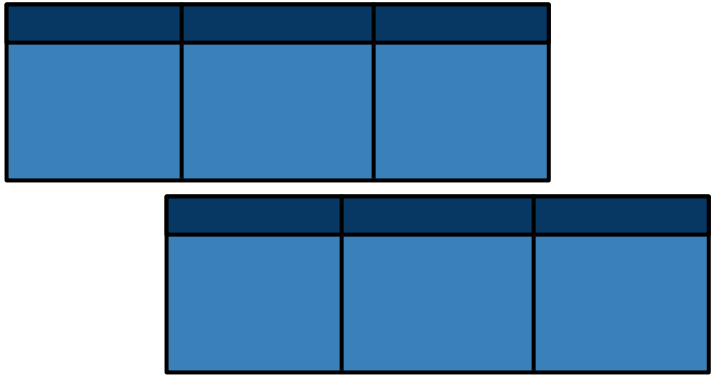




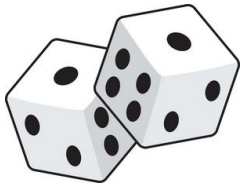
sample







sampleBy



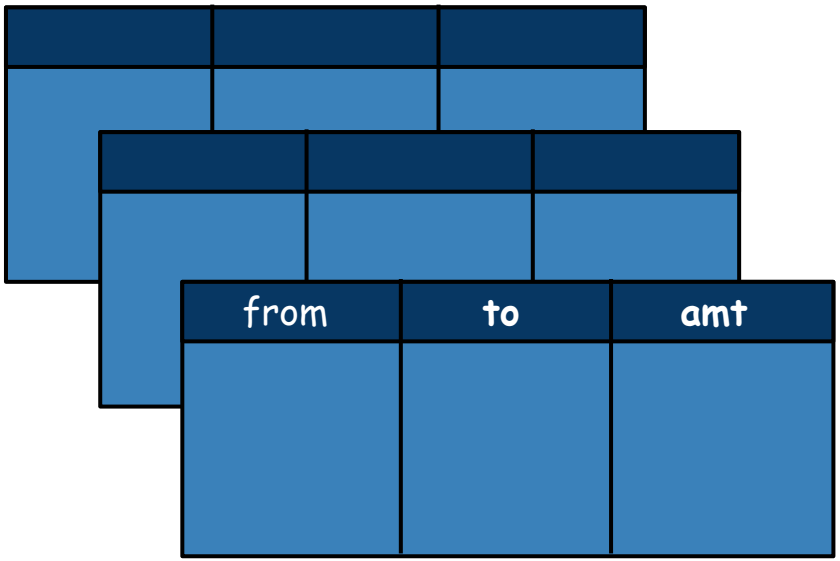
Bob	Carol	0.6
Bob	Bob	0.5
Alice	Dave	0.4
Alice	Alice	0.3
Alice	Carol	0.2

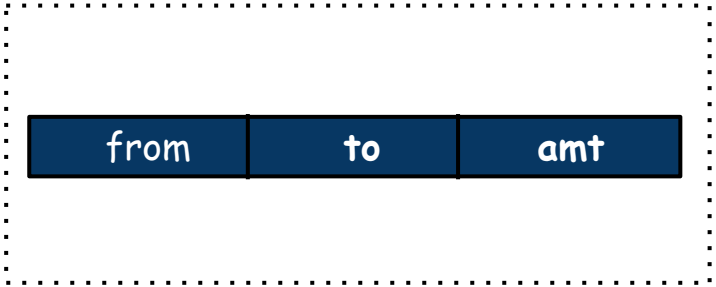
from	to	amt
Alice	Bob	0.1

Bob	Carol	0.6
Bob	Bob	0.5

from	to	amt
Alice	Bob	0.3

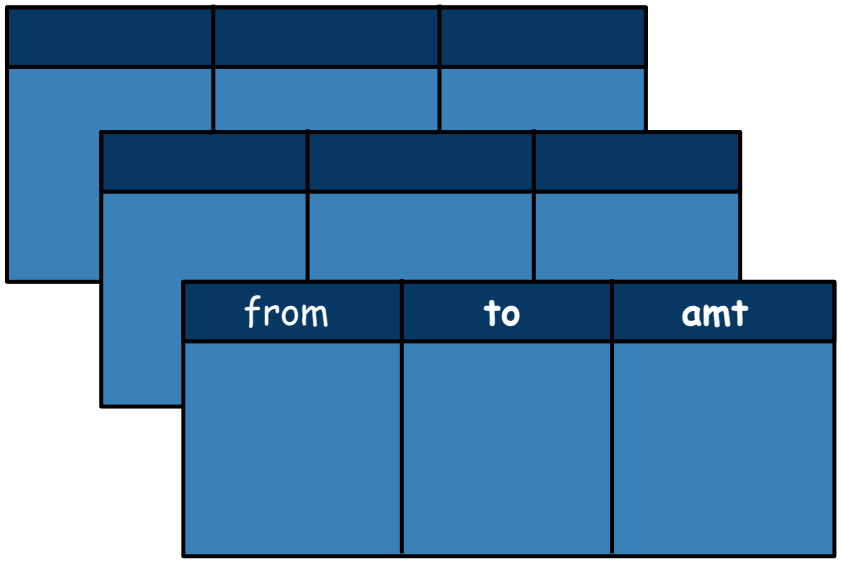
schema

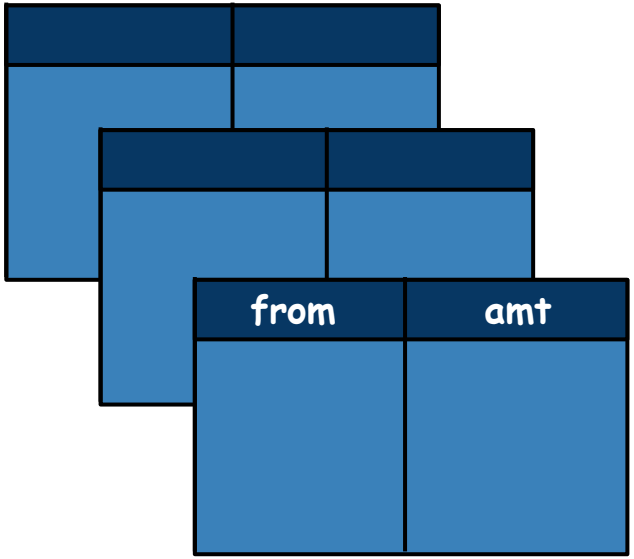




select

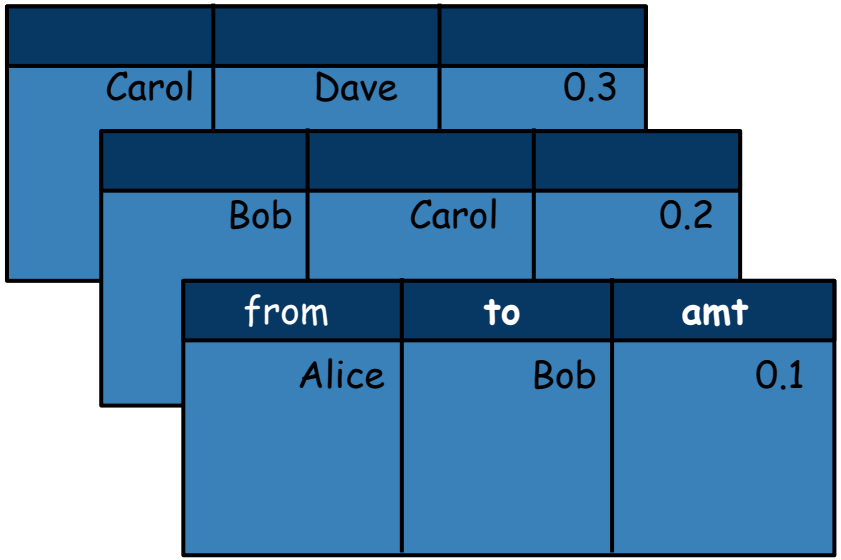
cols = ['from', 'amt']

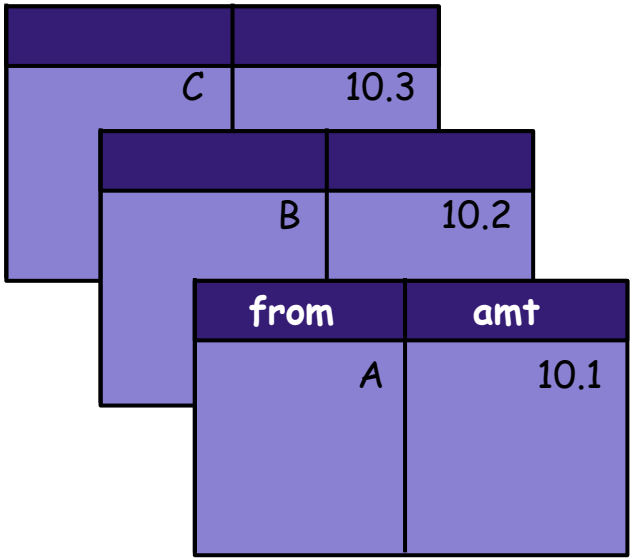




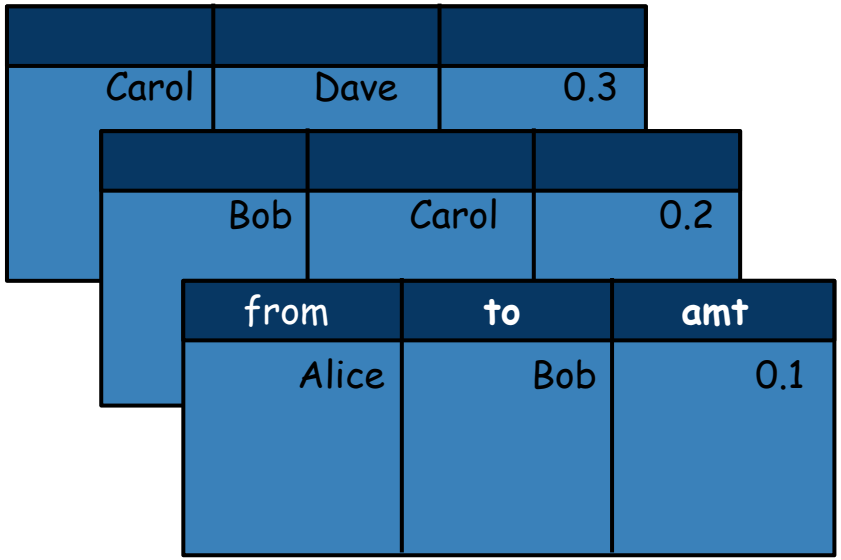
selectExpr

expr = ["substr(from,1,1)", "amt + 10"]





show

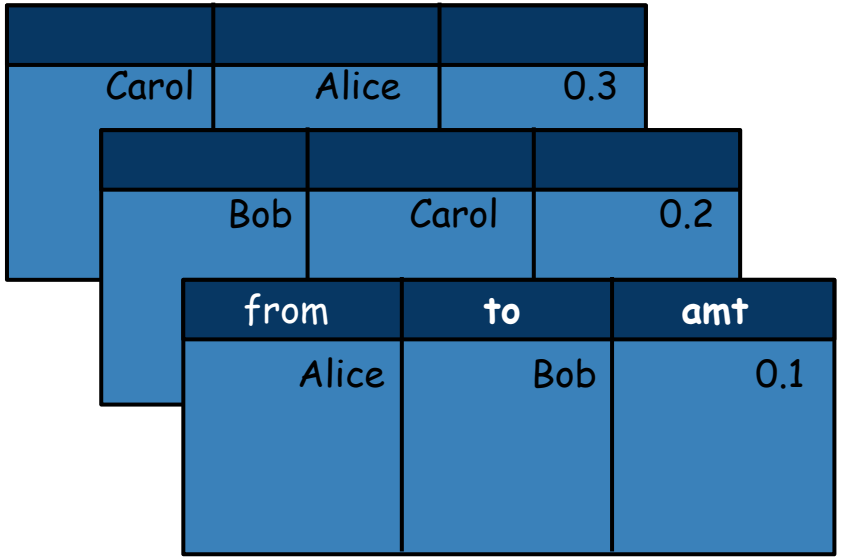


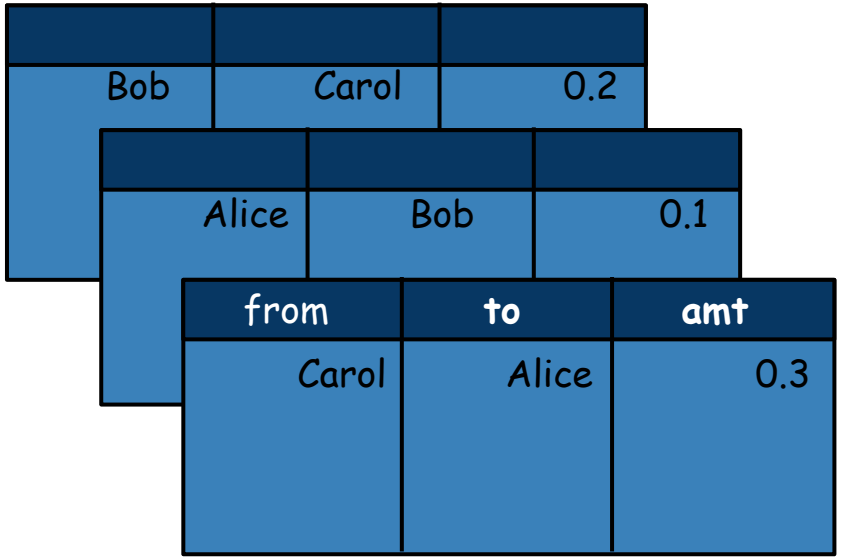
stdout

```
+-----+-----+---+
| from|   to|amt|
+-----+-----+---+
|Alice|  Bob|0.1|
|  Bob|Carol|0.2|
|Carol| Dave|0.3|
+-----+-----+---+
```

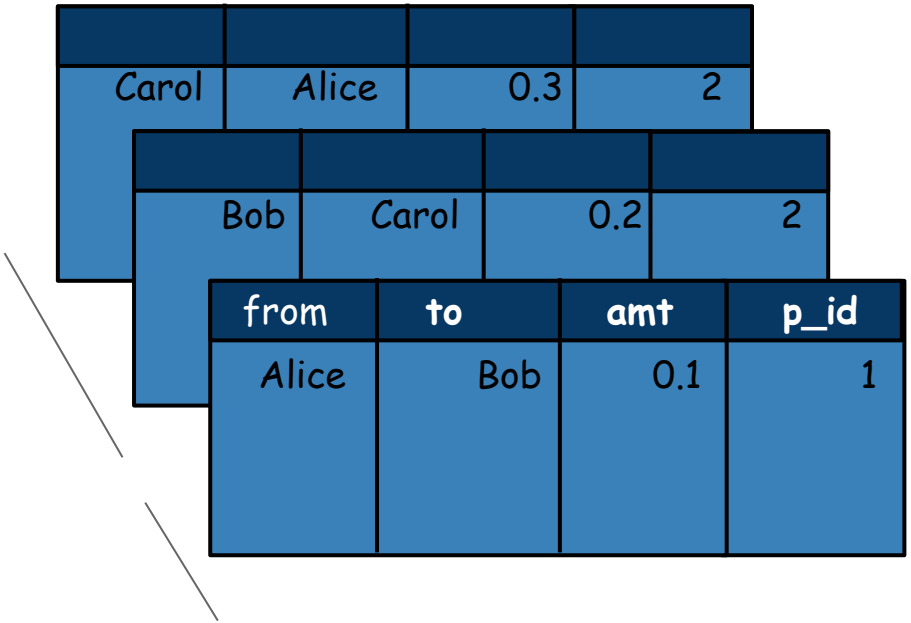
sort

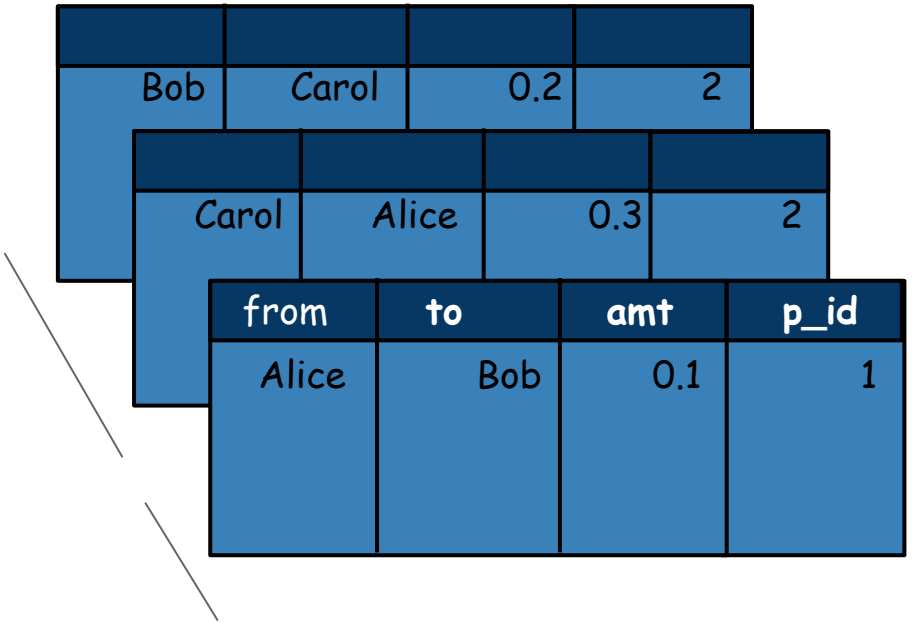
cols = ['to']


Carol Alice 0.3
Bob Carol 0.2
from to amt
Alice Bob 0.1

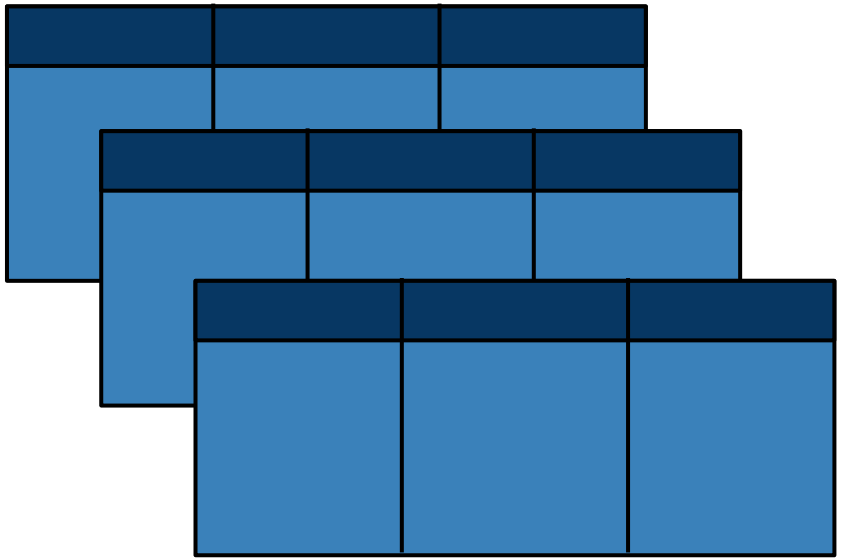

Bob Carol 0.2
Alice Bob 0.1
from to amt
Carol Alice 0.3

sortWithinPartitions





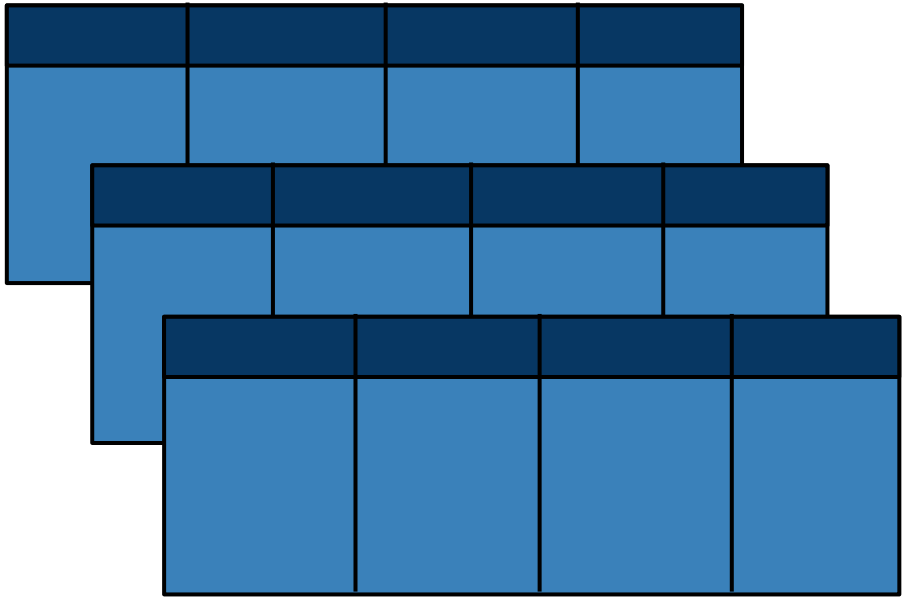
stat



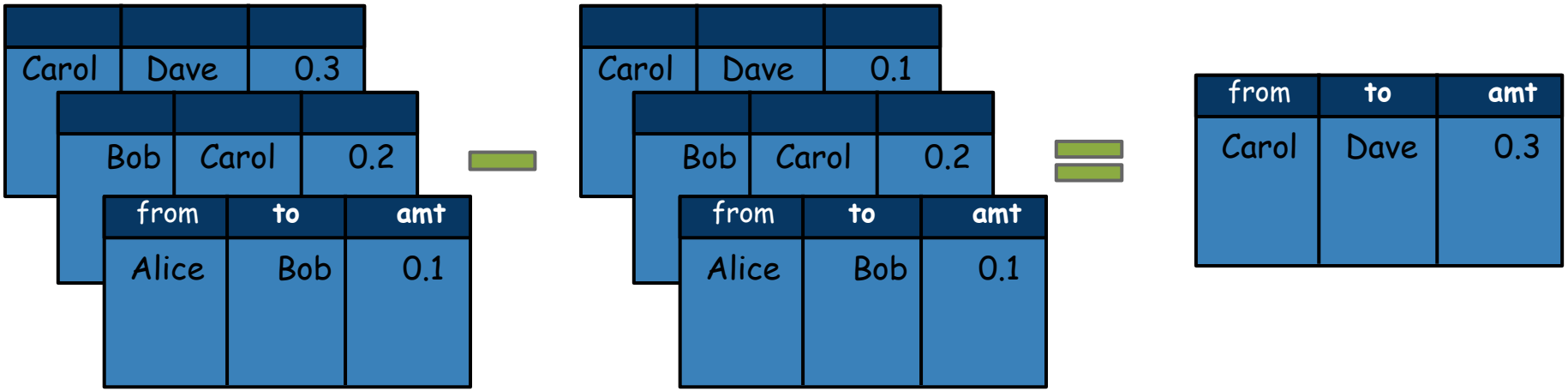
DataFrameStatFunctions Object

with methods: corr, cov, corsstab, freqItems, sampleBy

stat

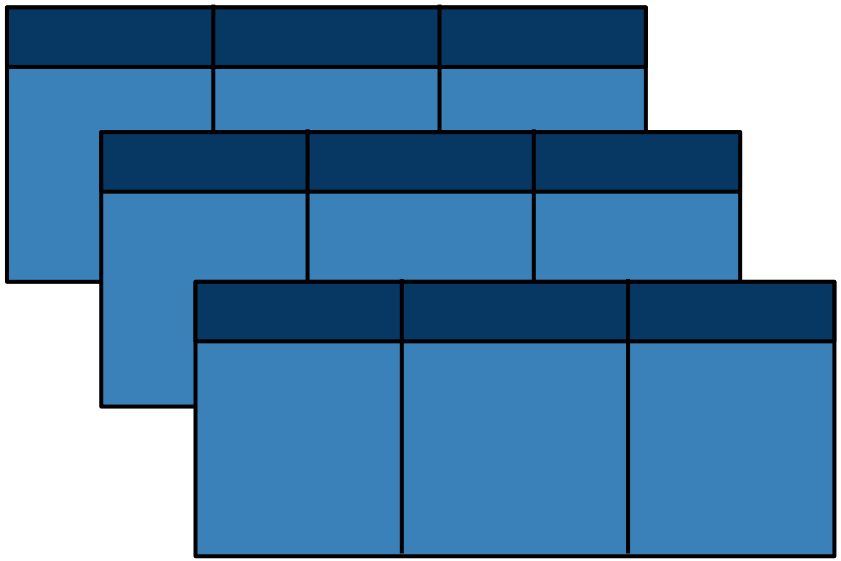


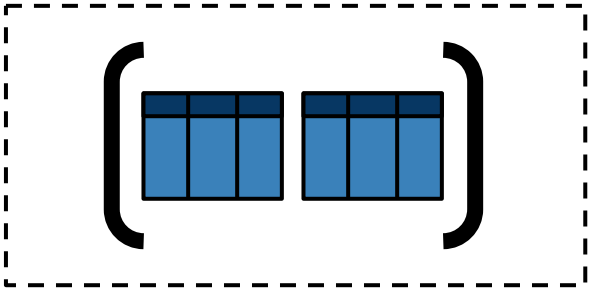
subtract


Carol
Dave
0.3
Bob
Carol
0.2
from
to
amt
Alice
Bob
0.1
Carol
Dave
0.1
Bob
Carol
0.2
from
to
amt
Alice
Bob
0.1
from
to
amt
Carol
Dave
0.3

take

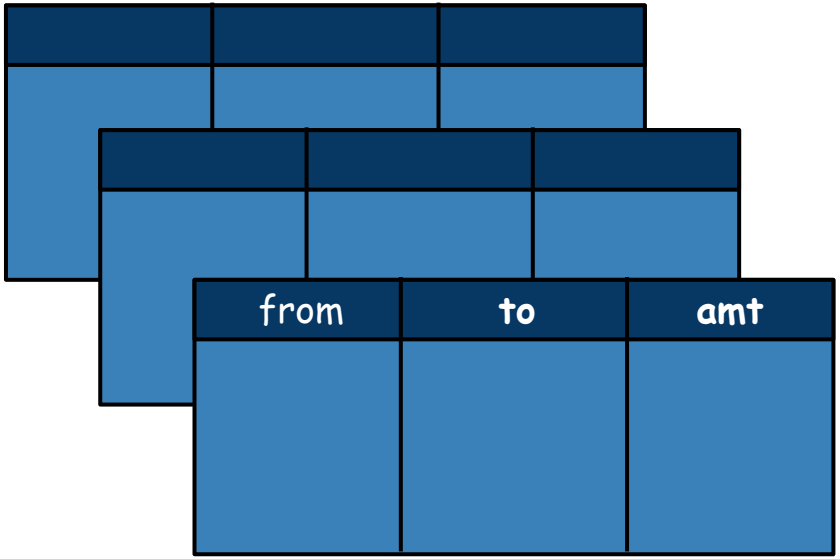
num = 2

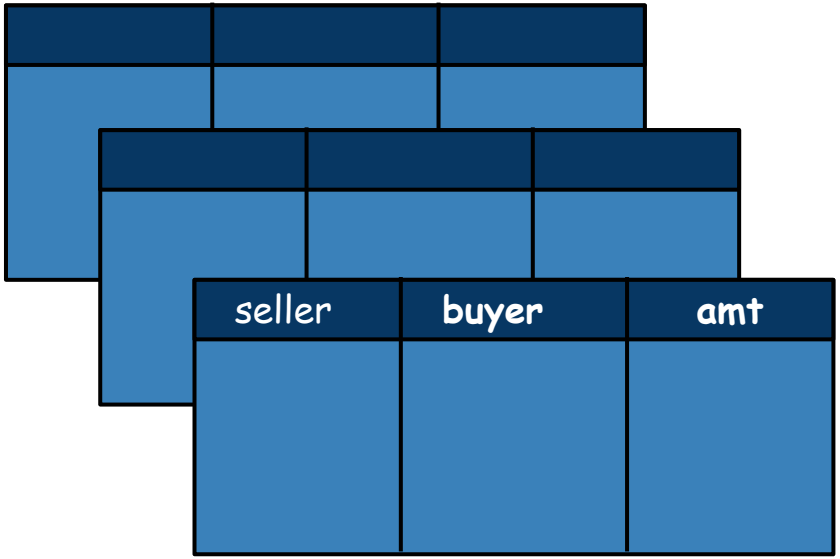




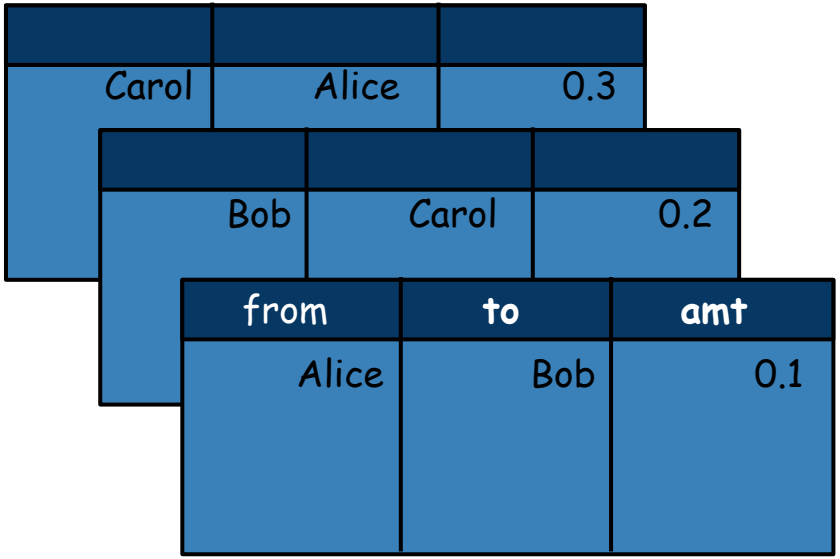
toDF

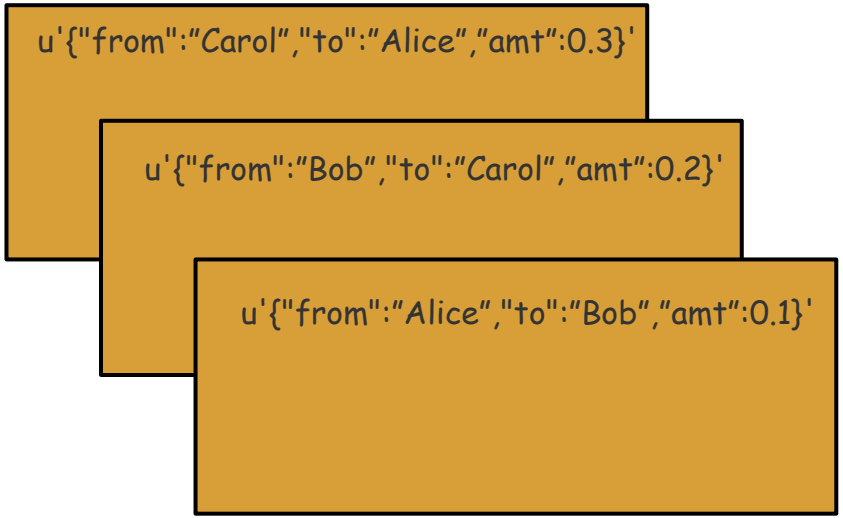
cols = ["seller", "buyer"]



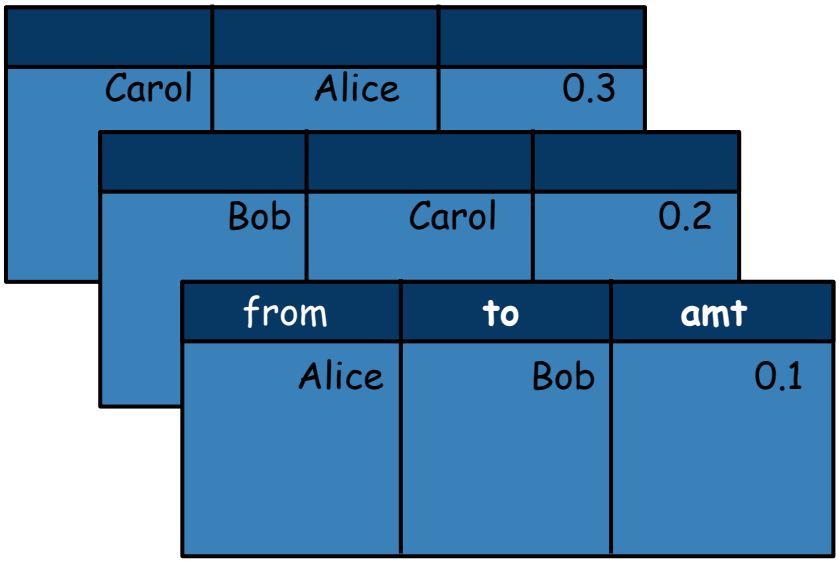


toJSON


Carol
Alice
0.3
Bob
Carol
0.2
from
to
amt
Alice
Bob
0.1

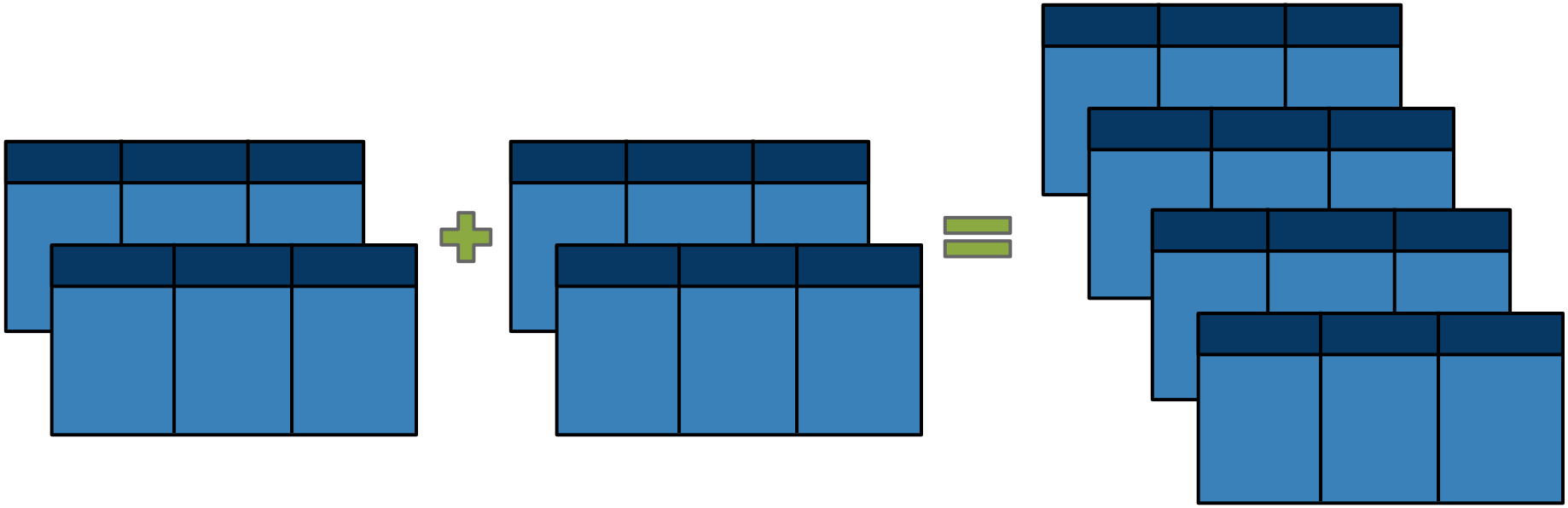

u'{"from":"Carol","to":"Alice","amt":0.3}'
u'{"from":"Bob","to":"Carol","amt":0.2}'
u'{"from":"Alice","to":"Bob","amt":0.1}'

toPandas

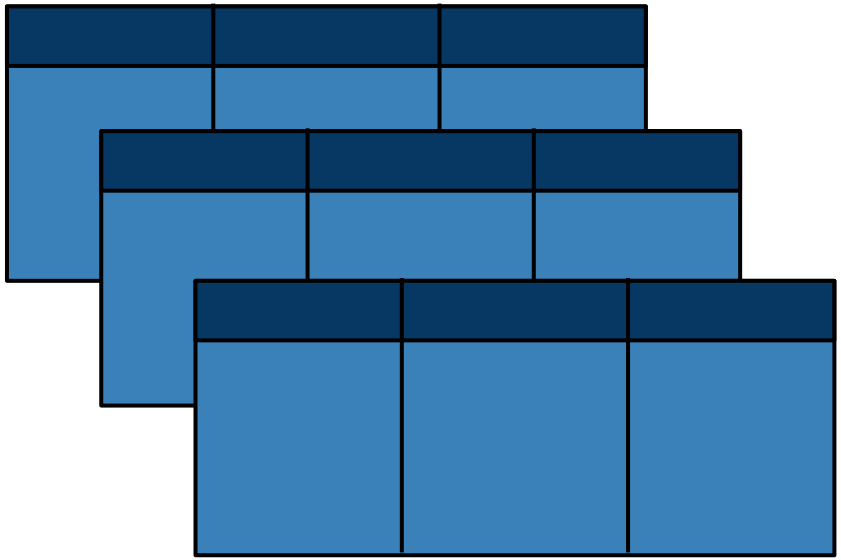



```
   from to  amt
0  Alice Bob 0.1
1  Bob Carol 0.2
2  Carol Alice 0.3
```

unionAll

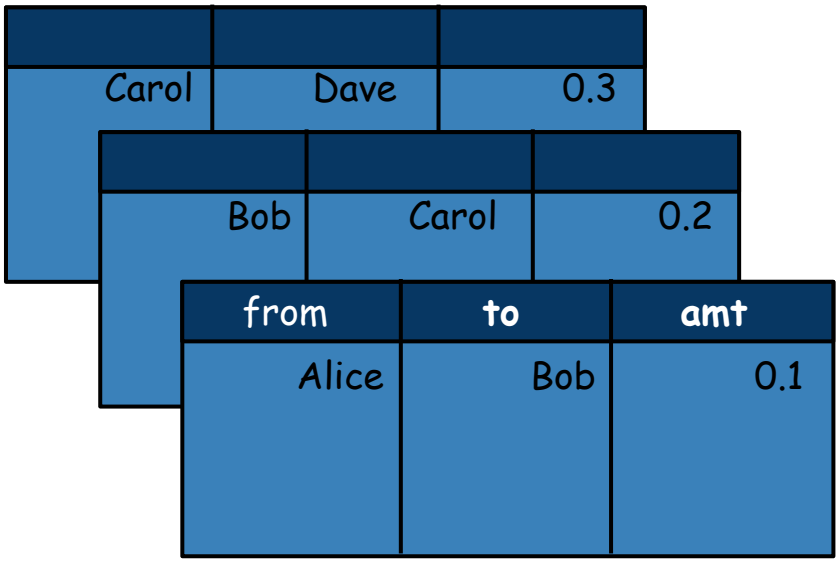


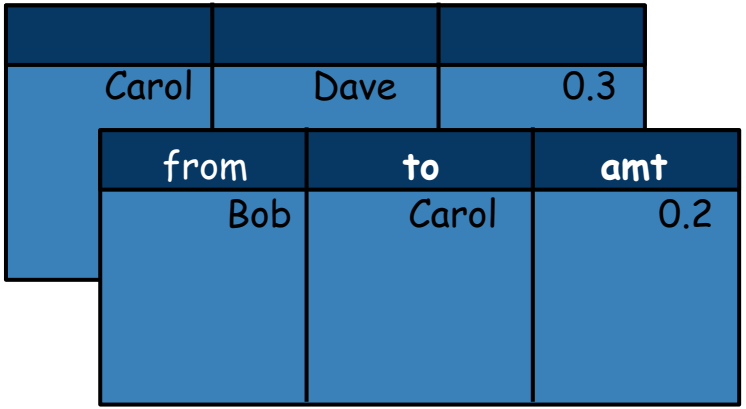
unpersist



where (filter)

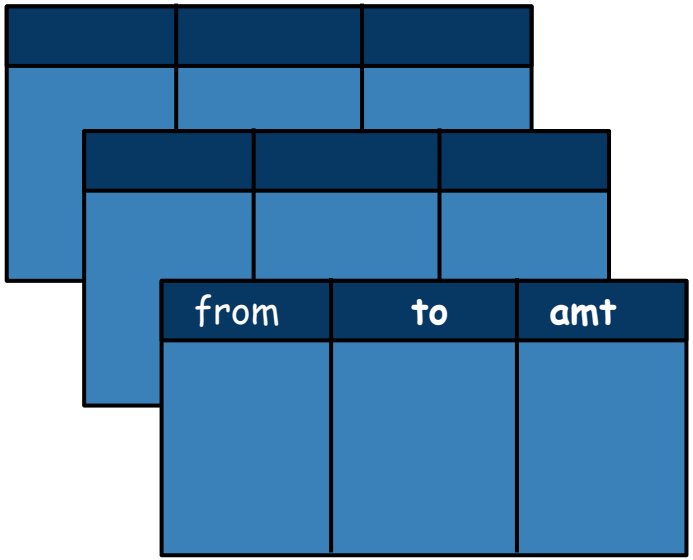
condition = "amt > 0.1"

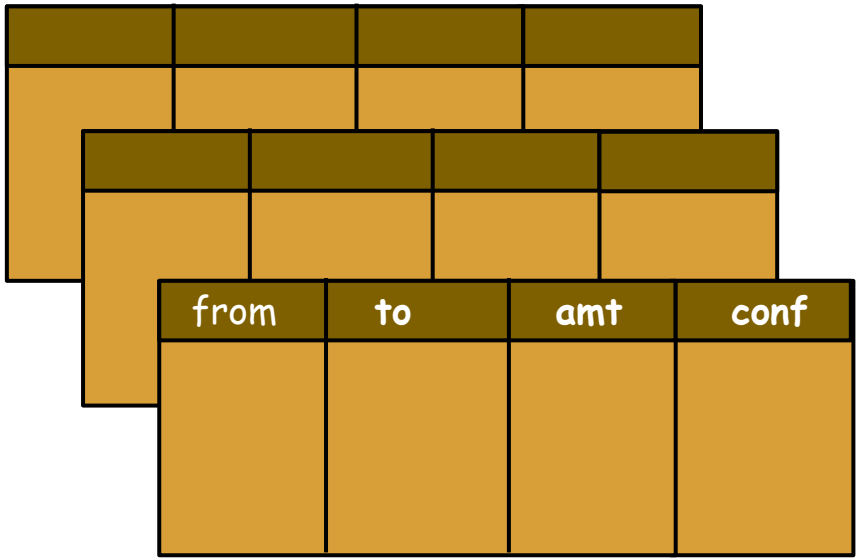




withColumn

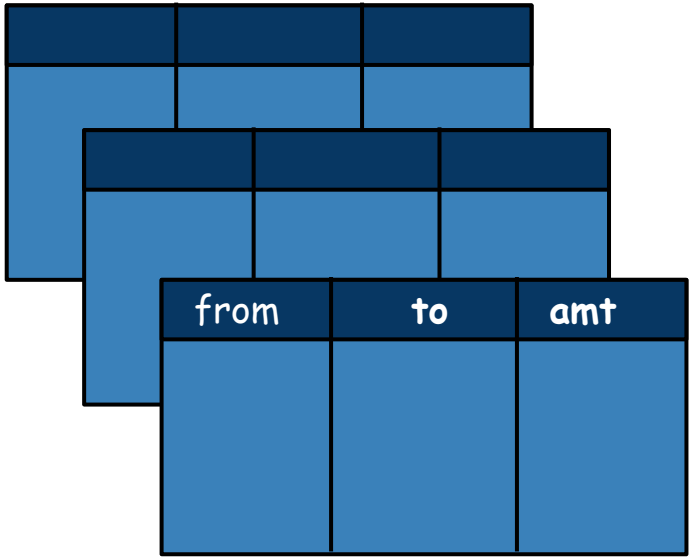
colName = 'conf'

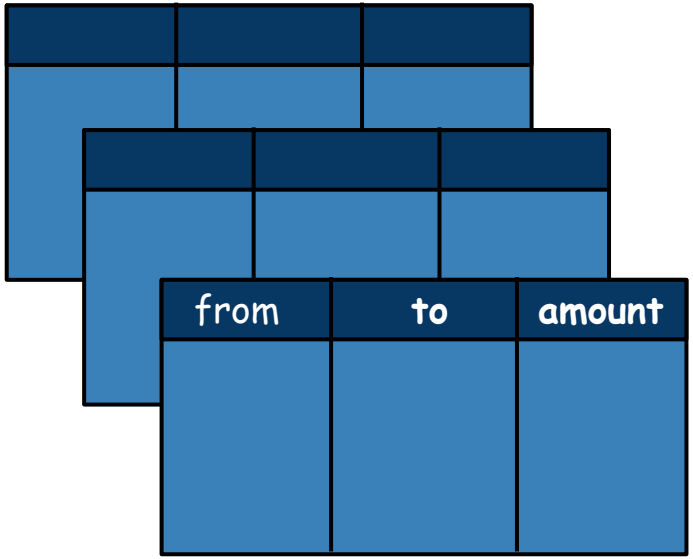




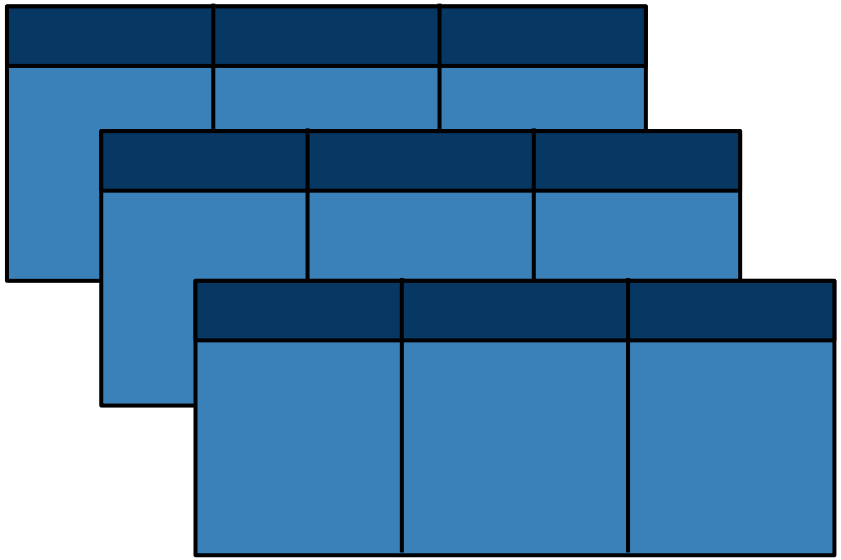
withColumnRenamed

existing = 'amt' col = 'amount'





write



DataFrameWriter Object

with methods: format, insertInto, jdbc, json, mode, option, options, orc, parquet, partitionBy, save, saveAsTable, text