# Discovering Spatial Frequent patterns in Big Data Using SpatialECLAT Algorithm

In this tutorial, we will discuss two approaches to find Spatial Frequent patterns in big data using SpatialECLAT algorithm.

1. **Basic approach:** Here, we present the steps to discover Spatial Frequent patterns using a single minimum support value
2. **Advanced approach:** Here, we generalize the basic approach by presenting the steps to discover Spatial Frequent patterns using multiple minimum support values.

---

## Basic approach: Executing SpatialECLAT on a single dataset at a particular minimum support value

### Step 1: Import the SpatialECLAT algorithm

In [1]:
```
from PAMI.frequentSpatialPattern.basic import SpatialECLAT  as alg
```

### Step 2: Specify the following input parameters

In [2]:
```
inputFile = 'transactional_T10I4D100K.csv'

minimumSupportCount=100  #Users can also specify this constraint between 0 to 1.
neighborFile='T10_utility_neighbour.txt'
seperator='¥t'
```

### Step 3: Execute the SpatialECLAT algorithm

In [3]:
```
obj = alg.SpatialECLAT(iFile=inputFile, minSup=minimumSupportCount,nFile=neighborFi
obj.startMine()             #Start the mining process
```

```
Spatial Frequent patterns were generated successfully using SpatialECLAT algorithm
```

### Step 4: Storing the generated patterns

#### Step 4.1: Storing the generated patterns in a file

In [4]:
```
obj.savePatterns(outFile='frequentPatternsMinSupCount1000.txt')
```

#### Step 4.2. Storing the generated patterns in a data frame

In [5]:
```
frequentPatternsDF= obj.getPatternsAsDataFrame()
```

### Step 5: Getting the statistics

#### Step 5.1: Total number of discovered patterns

In [6]:
```
print('Total No of patterns: ' + str(len(frequentPatternsDF)))
```

```
Total No of patterns: 4300
```

### Step 5.2: Runtime consumed by the mining algorithm

In [7]:
```
print('Runtime: ' + str(obj.getRuntime()))
```

```
Runtime: 38.03338432312012
```

In [8]:
```
##### Step 5.3: Total Memory consumed by the mining algorithm
```

In [9]:
```
print('Memory (RSS): ' + str(obj.getMemoryRSS()))
print('Memory (USS): ' + str(obj.getMemoryUSS()))
```

```
Memory (RSS): 242855936
Memory (USS): 204525568
```