main

August 28, 2023

0.1 Import event log

[1]:
```
import pm4py
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import warnings
%matplotlib inline
warnings.filterwarnings('ignore')
```

[2]:
```
domestic_path = 'data/DomesticDeclarations.xes'
international_path = 'data/InternationalDeclarations.xes'

log= pm4py.read_xes(domestic_path);
```

```
parsing log, completed traces ::   0%|          | 0/10500 [00:00<?, ?it/s]
```

0.2 Statistical Analysis of Event Data

[3]:
```
log.head(10)
```

[3]:
```
                       id  org:resource  \
0          st_step 86794_0  STAFF MEMBER
1          st_step 86793_0  STAFF MEMBER
2  dd_declaration 86791_19        SYSTEM
3  dd_declaration 86791_20        SYSTEM
4          st_step 86798_0  STAFF MEMBER
5          st_step 86799_0  STAFF MEMBER
6          st_step 86797_0  STAFF MEMBER
7  dd_declaration 86795_19        SYSTEM
8  dd_declaration 86795_20        SYSTEM
9          st_step 86804_0  STAFF MEMBER

                             concept:name            time:timestamp  \
0         Declaration SUBMITTED by EMPLOYEE 2017-01-09 08:49:50+00:00
1  Declaration FINAL_APPROVED by SUPERVISOR 2017-01-09 10:27:48+00:00
2                           Request Payment 2017-01-10 08:34:44+00:00
```

```
3                          Payment Handled 2017-01-12 16:31:22+00:00
4           Declaration SUBMITTED by EMPLOYEE 2017-01-09 09:26:14+00:00
5        Declaration APPROVED by PRE_APPROVER 2017-02-22 09:29:21+00:00
6  Declaration FINAL_APPROVED by SUPERVISOR 2017-02-23 07:14:45+00:00
7                          Request Payment 2017-03-06 13:07:25+00:00
8                          Payment Handled 2017-03-13 16:30:59+00:00
9           Declaration SUBMITTED by EMPLOYEE 2017-01-09 10:13:33+00:00

      org:role            case:id  case:concept:name case:BudgetNumber  \
0     EMPLOYEE  declaration 86791  declaration 86791      budget 86566
1   SUPERVISOR  declaration 86791  declaration 86791      budget 86566
2    UNDEFINED  declaration 86791  declaration 86791      budget 86566
3    UNDEFINED  declaration 86791  declaration 86791      budget 86566
4     EMPLOYEE  declaration 86795  declaration 86795      budget 86566
5  PRE_APPROVER  declaration 86795  declaration 86795      budget 86566
6   SUPERVISOR  declaration 86795  declaration 86795      budget 86566
7    UNDEFINED  declaration 86795  declaration 86795      budget 86566
8    UNDEFINED  declaration 86795  declaration 86795      budget 86566
9     EMPLOYEE  declaration 86800  declaration 86800      budget 86566

     case:DeclarationNumber  case:Amount
0  declaration number 86792    26.851205
1  declaration number 86792    26.851205
2  declaration number 86792    26.851205
3  declaration number 86792    26.851205
4  declaration number 86796   182.464172
5  declaration number 86796   182.464172
6  declaration number 86796   182.464172
7  declaration number 86796   182.464172
8  declaration number 86796   182.464172
9  declaration number 86801   320.646137
```

[4]:
```
# show rows where case:id not equal case:concept:name
log[log['case:id'] != log['case:concept:name']]
```

[4]:
```
Empty DataFrame
Columns: [id, org:resource, concept:name, time:timestamp, org:role, case:id,
case:concept:name, case:BudgetNumber, case:DeclarationNumber, case:Amount]
Index: []
```

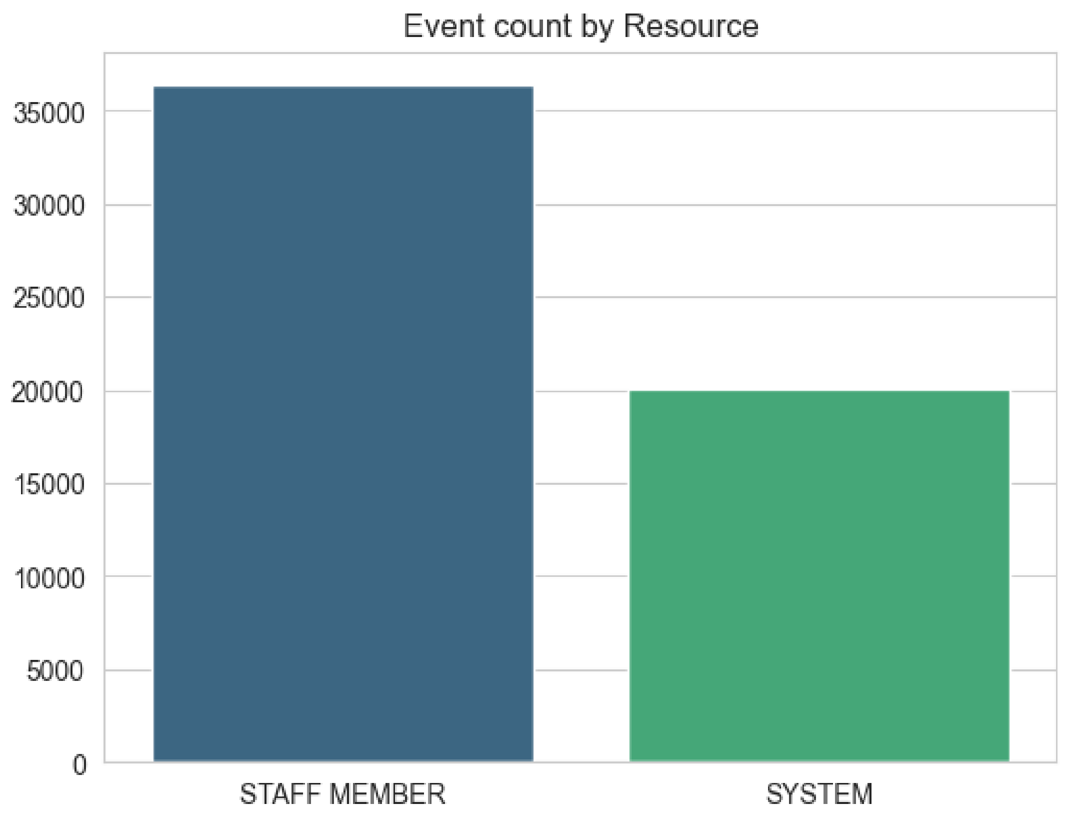
It looks like case_id and case_concept_name columns are the same.

[5]:
```
# to improve readability we trim the word 'Declaration' out of concept:name␣
↪column, if it exists
log['concept:name'] = log['concept:name'].str.replace('Declaration ', '')
```

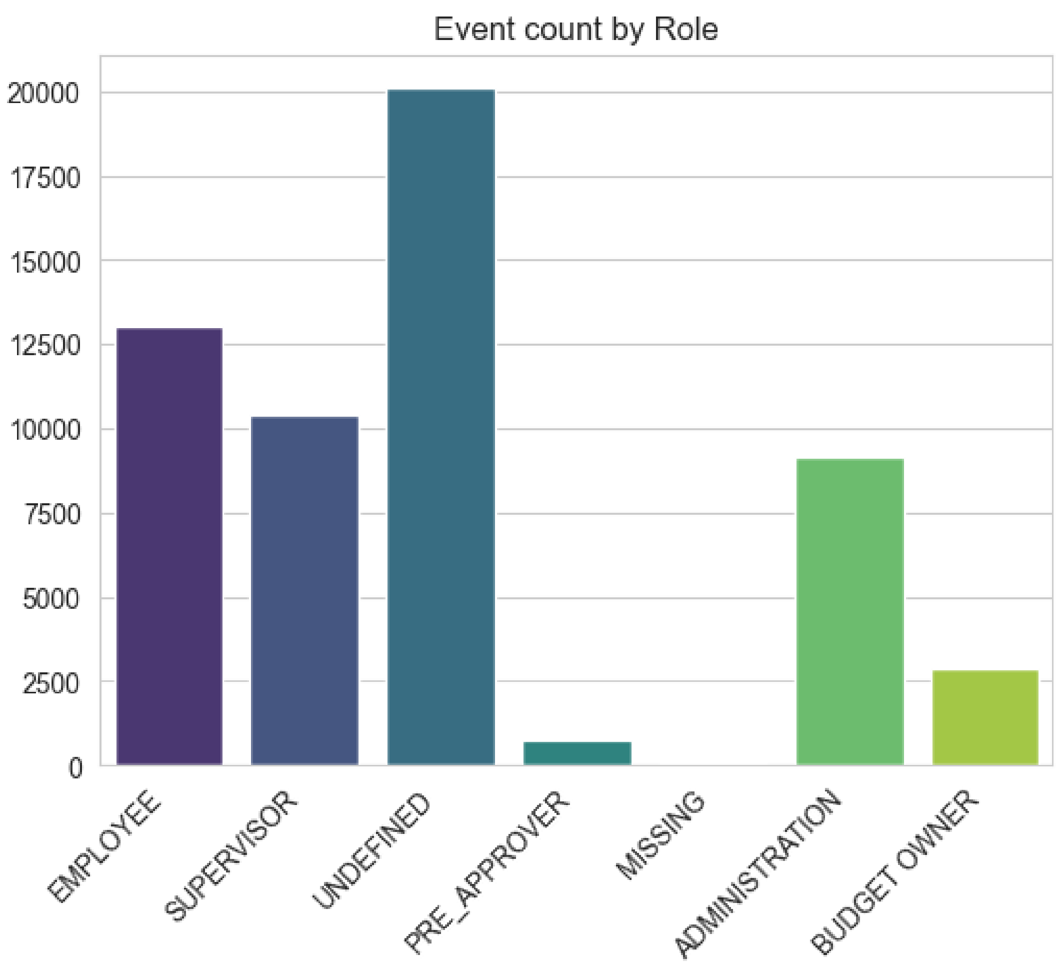
```
[6]: # pick random case
     case_ids = log['case:id'].unique()
     random_case = log[log['case:id'] == np.random.choice(case_ids)]
     random_case = random_case.sort_values(by='time:timestamp')
     random_case
```

```
[6]:                                 id  org:resource                  concept:name  \
     25225          st_step 123557_0  STAFF MEMBER         SUBMITTED by EMPLOYEE
     25226          st_step 123556_0  STAFF MEMBER    APPROVED by ADMINISTRATION
     25227          st_step 123558_0  STAFF MEMBER      APPROVED by BUDGET OWNER
     25228          st_step 123559_0  STAFF MEMBER  FINAL_APPROVED by SUPERVISOR
     25229  dd_declaration 123554_19        SYSTEM               Request Payment
     25230  dd_declaration 123554_20        SYSTEM               Payment Handled

                       time:timestamp        org:role             case:id  \
     25225 2018-05-02 08:18:14+00:00        EMPLOYEE  declaration 123554
     25226 2018-05-02 08:20:33+00:00  ADMINISTRATION  declaration 123554
     25227 2018-05-02 08:22:33+00:00    BUDGET OWNER  declaration 123554
     25228 2018-05-03 09:03:47+00:00      SUPERVISOR  declaration 123554
     25229 2018-05-12 12:12:37+00:00       UNDEFINED  declaration 123554
     25230 2018-05-17 15:31:32+00:00       UNDEFINED  declaration 123554

             case:concept:name case:BudgetNumber     case:DeclarationNumber  \
     25225  declaration 123554      budget 86566  declaration number 123555
     25226  declaration 123554      budget 86566  declaration number 123555
     25227  declaration 123554      budget 86566  declaration number 123555
     25228  declaration 123554      budget 86566  declaration number 123555
     25229  declaration 123554      budget 86566  declaration number 123555
     25230  declaration 123554      budget 86566  declaration number 123555

            case:Amount
     25225    42.322488
     25226    42.322488
     25227    42.322488
     25228    42.322488
     25229    42.322488
     25230    42.322488
```

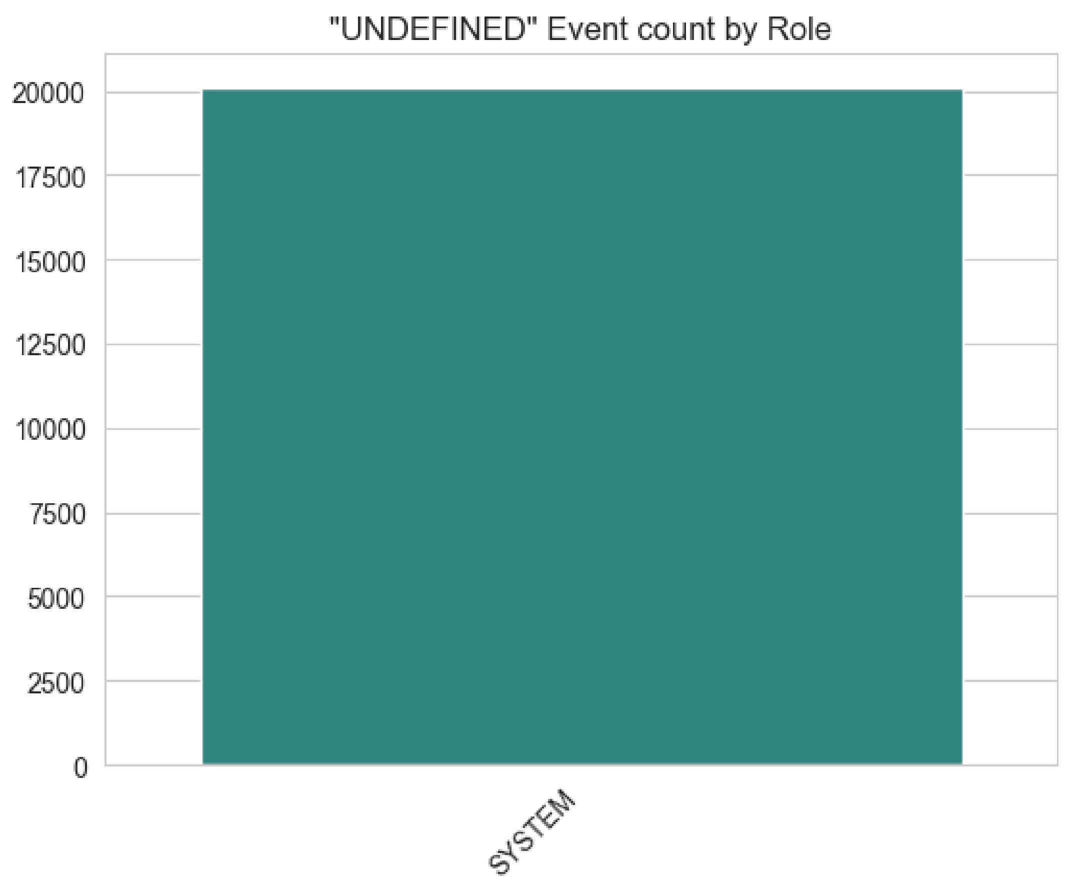
```
[7]: resources = log['org:resource'].unique()
     sns.countplot(x='org:resource', data=log, palette='viridis').set(title='Event␣
       ↪count by Resource', xlabel='', ylabel='');
```



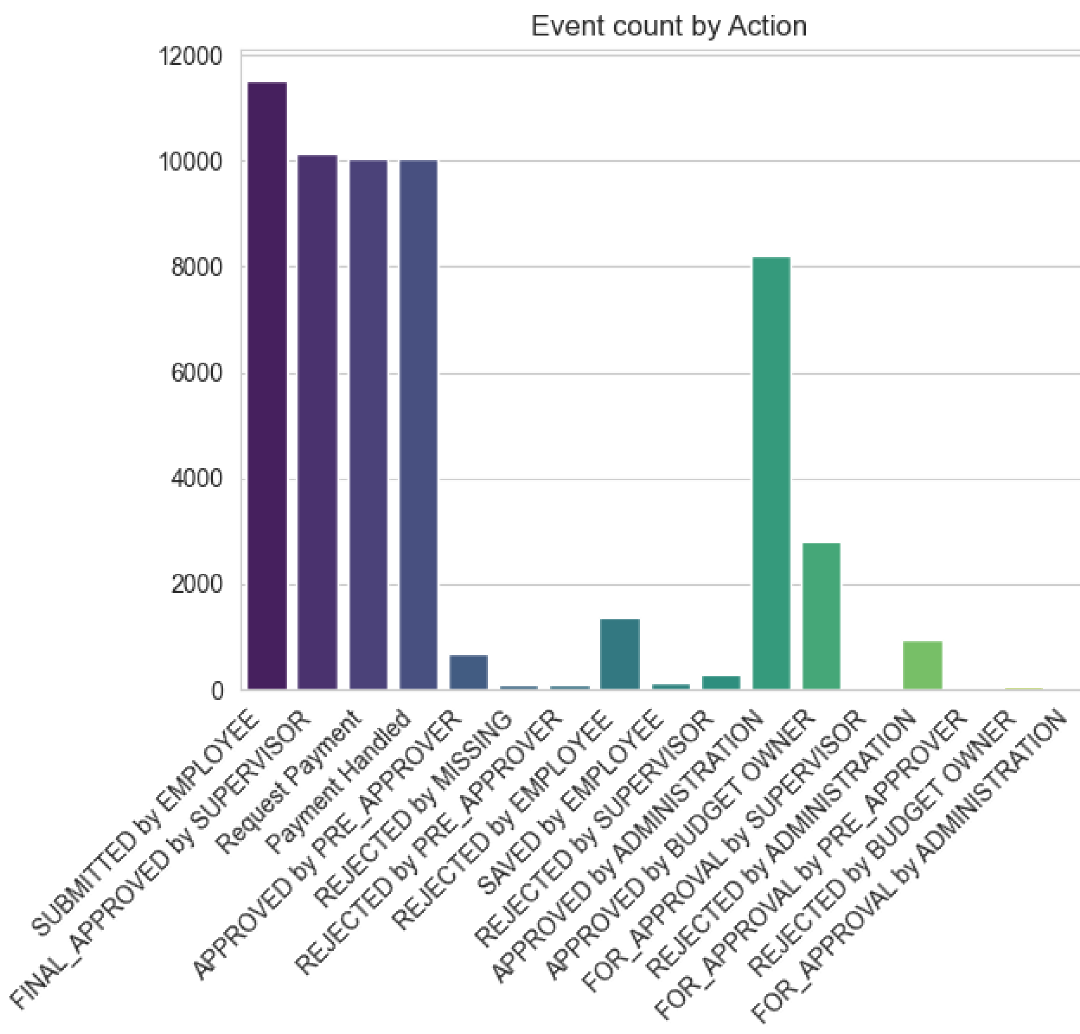
```
[8]: roles = log['org:role'].unique()
     sns.countplot(x='org:role', data=log, palette='viridis').set(title='Event count␣
      ↪by Role', xlabel='', ylabel='')
     plt.xticks(rotation=45, ha='right');
```



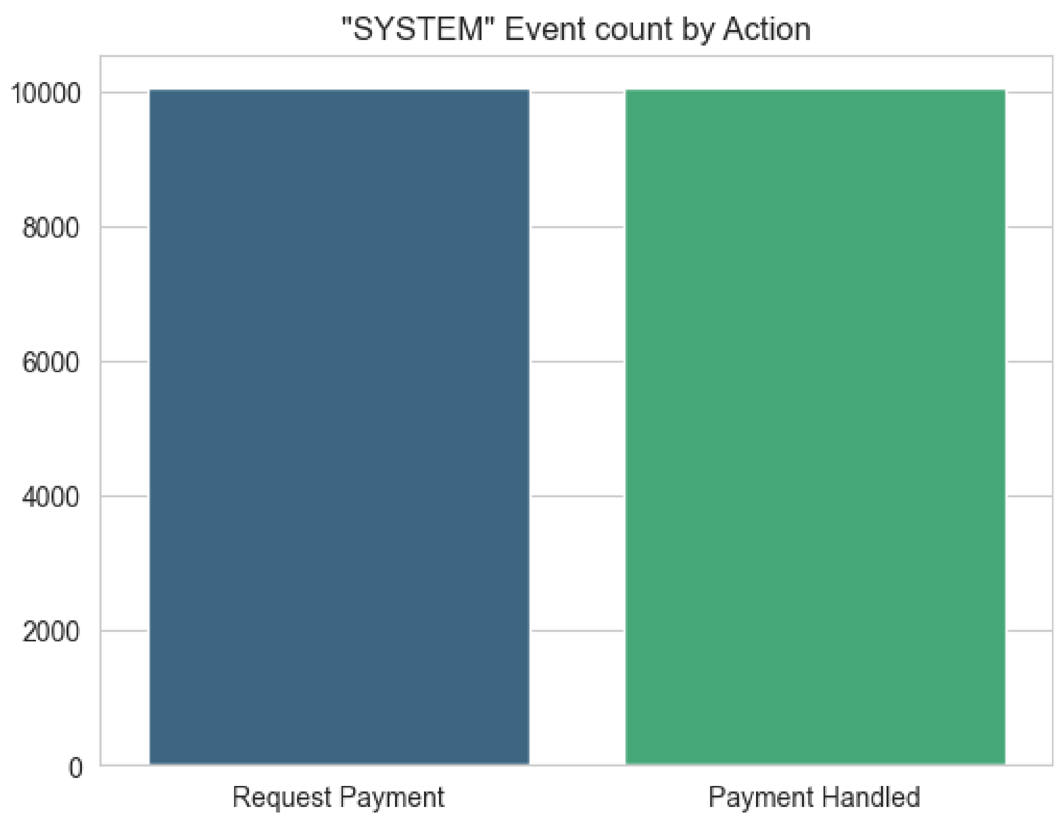
[9]:
```
log_i = log[log['org:role'] == 'UNDEFINED']
sns.countplot(x='org:resource', data=log_i, palette='viridis').
  ↪set(title='"UNDEFINED" Event count by Role', xlabel='', ylabel='')
plt.xticks(rotation=45, ha='right');
```



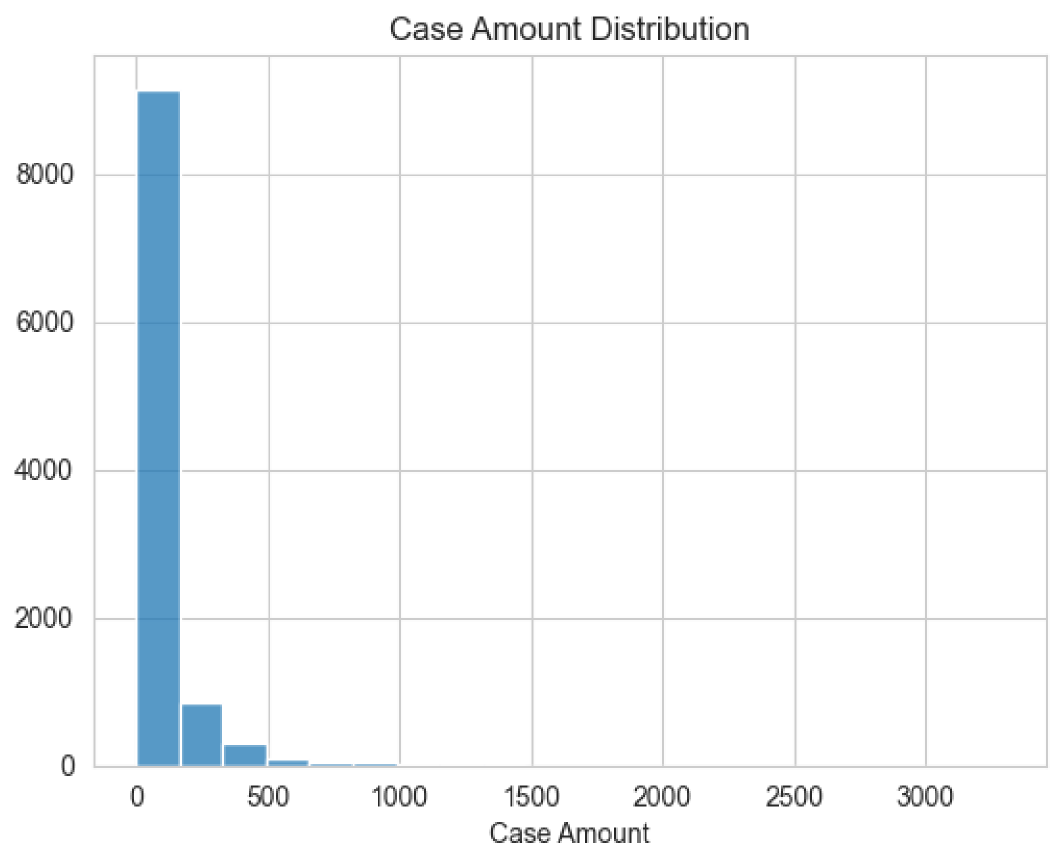
```
[10]: actions = log['concept:name'].unique()
      sns.countplot(x='concept:name', data=log, palette='viridis').set(title='Event␣
       ↪count by Action', xlabel='', ylabel='')
      plt.xticks(rotation=45, ha='right');
```



```
[11]: # System Events
      log_s = log[log['org:resource'] == 'SYSTEM']
      sns.countplot(x='concept:name', data=log_s, palette='viridis').
       ↪set(title='"SYSTEM" Event count by Action', xlabel='', ylabel='');
```



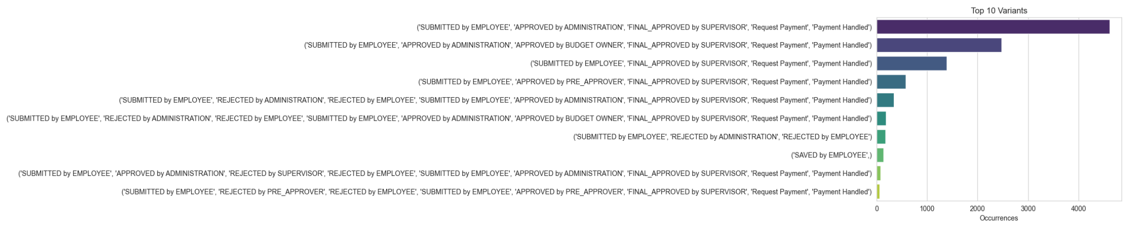
[12]:
```
# case amount distribution for distinct case:id
distinct_case_amounts = log.groupby('case:id')['case:Amount'].max()
sns.histplot(distinct_case_amounts, kde=False, bins=20).set(title='Case Amount␣
↪Distribution', xlabel='Case Amount', ylabel='');
```



0.3 Process Discovery

Having mined the model we may vizualize it as a Process Tree or Petri Net.

[13]:
```
variants_dict = pm4py.get_variants(log)

variants_arr = []
idx = 1
for variant, n in variants_dict.items():
    variant_in_dict = {}
    variant_in_dict['variant_number'] = idx
    variant_in_dict['variant_count'] = n
    variant_in_dict['variant_trace'] = variant

    variants_arr.append(variant_in_dict)

    idx += 1


variants_df = pd.DataFrame(variants_arr)
variants_df = variants_df.sort_values(by='variant_count', ascending=False)

sns.barplot(x='variant_count', y='variant_trace', data=variants_df[:10],␣
↪palette='viridis').set(title='Top 10 Variants', xlabel='Occurrences',␣
↪ylabel='');
```

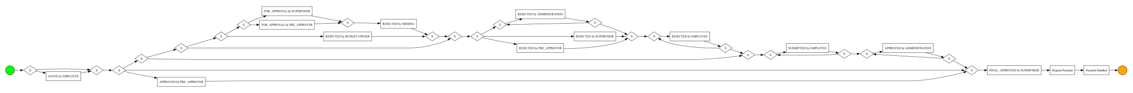


```
[15]: cases = log['case:id'].unique()
      count_cases_top_10 = variants_df[:10]['variant_count'].sum()
      print(f'Top 10 variants account for {count_cases_top_10:,} cases out of␣
      ↪{len(cases):,}.')
```

```
Top 10 variants account for 10,033 cases out of 10,500.
```

0.4 Check out BPMN Model

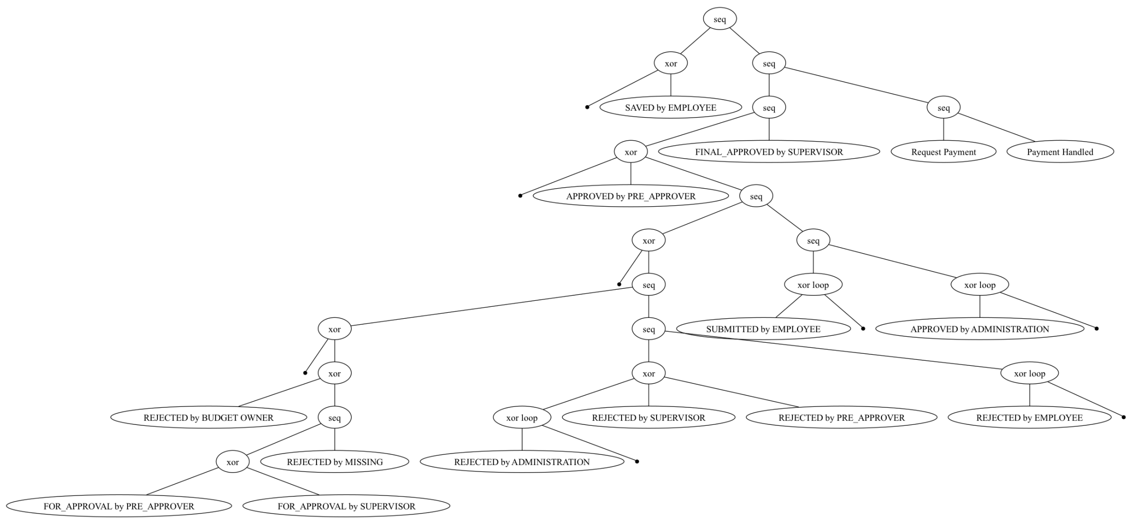
```
[22]: bpmn_model = pm4py.discover_bpmn_inductive(
          log=log,
          noise_threshold=0.8,
          activity_key='concept:name',
          timestamp_key='time:timestamp',
          case_id_key='case:id'
      )
      pm4py.view_bpmn(bpmn_model)
```



In BPMN model "x" stands for choice. We may observe that algorithm mined a model with a lot of choices and shortcuts. But on the end of the process it needs to be approved by supervisor. Last two stepps are done by system.

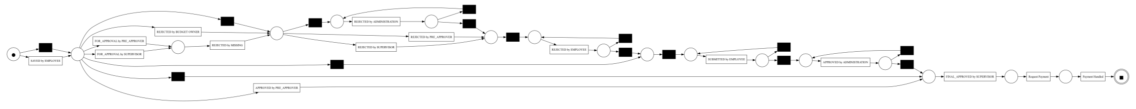
0.5 Process Tree

[23]:
```python
process_tree = pm4py.discover_process_tree_inductive(
    log=log,
    noise_threshold=.8,
    activity_key='concept:name',
    timestamp_key='time:timestamp',
    case_id_key='case:id'
)
pm4py.view_process_tree(process_tree)
```



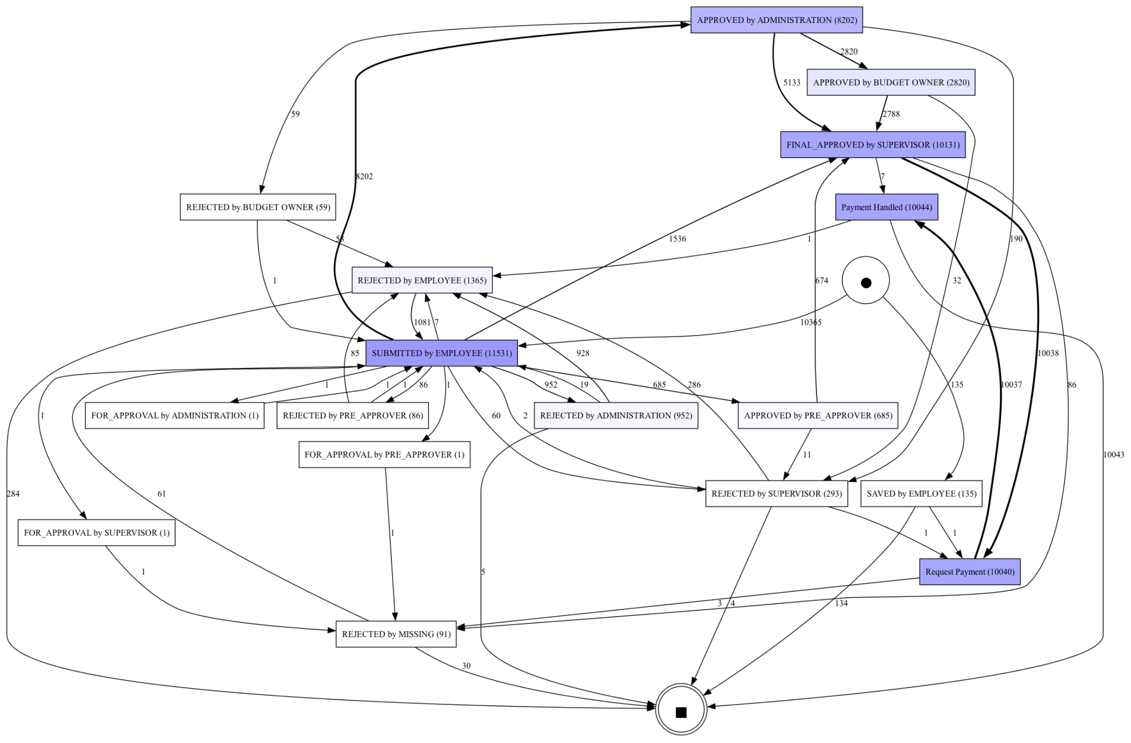
0.6 Petri Net

[28]:
```python
p_net, im, fm = pm4py.discover_petri_net_inductive(
    log=log,
    noise_threshold=.8,
    activity_key='concept:name',
    timestamp_key='time:timestamp',
    case_id_key='case:id'
)
pm4py.view_petri_net(p_net, im, fm)
```



0.7 Directly-Follows Graph

[29]:
```python
dfg, sa, ea = pm4py.discover_dfg(
    log=log,
    activity_key='concept:name',
    timestamp_key='time:timestamp',
    case_id_key='case:id'
)
pm4py.view_dfg(dfg, sa, ea)
```


APPROVED by ADMINISTRATION (8202)
APPROVED by BUDGET OWNER (2820)
FINAL_APPROVED by SUPERVISOR (10131)
Payment Handled (10044)
REJECTED by BUDGET OWNER (59)
REJECTED by EMPLOYEE (1365)
SUBMITTED by EMPLOYEE (11531)
FOR_APPROVAL by ADMINISTRATION (1)
REJECTED by PRE_APPROVER (86)
REJECTED by ADMINISTRATION (952)
APPROVED by PRE_APPROVER (685)
FOR_APPROVAL by PRE_APPROVER (1)
REJECTED by SUPERVISOR (293)
SAVED by EMPLOYEE (135)
FOR_APPROVAL by SUPERVISOR (1)
Request Payment (10040)
REJECTED by MISSING (91)

Data Granularity mismatch?

TODO: Try to reshuffle events into new categories and create models again.

0.8 Statistics

TODO