```
def similarity(a1, a2, types):
  """
  a = '河南省郑州市金水区中科大厦10楼202'
  b1 = '河南省郑州市金水区中科大厦'
  b2 = ''

  c = '中原银行'
  d1 = '中原银行股份有限公司'
  d2 = '郑州银行有限公司'

  print('%.6f' % faddr(a, b1, types='addr'))
  print('%.6f' % faddr(a, b2, types='addr'))
  print('%.6f' % faddr(c, d1, types='comp'))
  print('%.6f' % faddr(c, d2, types='comp'))

  """
  def t_addr(addr):
    addr = addr.replace('省', '').replace('市', '').replace('区', '')
    addr = addr.replace('村', '').replace('镇', '').replace('乡', '')
    return addr

  def t_comp(comp):
    comp = comp.replace('股份有限公司', '').replace('有限公司', '').replace('公司', '')
    return comp

  if types == 'addr_similarity':
    a1, a2 = t_addr(a1), t_addr(a2)
  if types == 'comp_similarity':
    a1, a2 = t_comp(a1), t_comp(a2)

  n = 0
  l1, l2 = len(a1), len(a2)
  for i in a1:
    if i in a2:
      n += 1
  for i in a2:
    if i in a1:
      n += 1

  if (l1+l2) == 0:
    return np.nan
  else:
    k = n/(l1+l2)

  if types == 'name_similarity':
    if (k == 1) and (a1 != a2):
      k = 0.99
  return k
```

相似度计算

假设进行计算的两个字段为A（进件）和B（历史库内按照规则匹配后字段）：

l 步骤一

赋初始值： -9990996；

l 步骤二

如果A为空（即进件字段为空）或B为空（即库内未匹配此对应计算信息），则相似度= -9990996（默认值）；

l 步骤三

进行相似度计算的两个字段进行数据清洗；

\1. 若是地址类计算相似度，A和B字段同时进行如下清洗：

1) 去掉空格

2) 去掉 '省', '市',‘’区', '村', '镇', '乡'

\2. 若是公司名称类计算相似度，A和B字段同时进行如下清洗：

1) 去掉空格

2) 去掉‘股份有限公司’、‘有限公司’、‘公司’

\3. 若是姓名类（申请人、紧急联系人、直系亲属）计算相似度，A和B字段同时进行如下清洗：

1） 去掉空格

l 步骤四

相似度计算

数据清洗后，按照如下示例进行相似度计算：

\1. 按照以下算法计算：

A=呜呜呜呜呜呜呜

B=呜

A里面找B，B里面找A

相似度=(1+7)/两个比较字符串的总长度（1+7）；

\2. 若计算姓名类相似度则增加如下判断：

相似度通过1）计算后为1且A^=B,则相似度修正为0.99，如A=张杰，B=张杰杰（0.99待优化）

考虑方案:

1、min(n1,n2) *2/(n1+n2)*

2、1-(max(n1,n2)-min(n1,n2))/(2 max(n1,n2))

3、0.99阈值

4、根据字符长度分段定义值