Facilty Location Problem

A large company wishes to open new depots to deliver to its sales centers. Every new setup of a depot has a fixed cost. Goods are delivered from a depot to the sales centers close to the site. Every delivery has a cost that depends on the distance covered.

The list of depots and sales centers and their related costs are provided in the file ``facility.gdx''. Furthermore, the capacity of each depot and the demand at each sales center is provided in that file.

In [10]:
```
import sys
import pandas as pd
import numpy as np

from gamspy import (
    Container,Set,Alias,Parameter,Variable,Equation,Model,Problem,Sense,Opti
    Domain,Number,Sum,Product,Smax,Smin,Ord,Card,SpecialValues,
    ModelStatus,SolveStatus,
)

# any optimum within <1 of the true optimum must BE the true optimum!
options = Options(equation_listing_limit=0,absolute_optimality_gap=0.999)

m = Container(options=options,load_from="facility.gdx")

Clients, Depots, Cap, Copen, Cship, Dem = m.getSymbols(['Clients','Depots','
```

In [11]:
```
# PUT YOUR CODE HER-/+/E
depot = Alias(m, 'd', Depots)
client = Alias(m, 'c', Clients)
arcs = Set(m, 'arcs', domain=[depot, client])
arcs.setRecords(Cship.records)

amount = m.addVariable('amount','positive',domain=[depot, client])
useDepot = m.addVariable('useDepot','binary',domain=depot)

demand = m.addEquation('demand',domain=[client])
demand[client]= Sum(arcs[depot, client], amount[depot, client]) >= Dem[clier

capacity = m.addEquation('capacity', domain=[depot])
capacity[depot] = Sum(arcs[depot, client], amount[depot, client]) <= Cap[dep

indicator = m.addEquation('indicator', domain=[depot, client])
indicator[arcs[depot, client]] = Cap[depot] * useDepot[depot] >= amount[depo

cost = Sum(arcs[depot, client], Cship[depot, client] * amount[depot, client]

fixedcost = m.addModel('fixedCost',
    equations=m.getEquations(),
```

```python
    problem=Problem.MIP,
    sense=Sense.MIN,
    objective=cost,
)

fixedcost.solve()
```

Out[11]:

	Solver Status	Model Status	Objective	Num of Equations	Num of Variables	Model Type	Solver	Solve Tim
0	Normal	OptimalGlobal	42122	77	64	MIP	CPLEX	0.06

In [12]:

```python
# What if Depot 1 has to be open?
# PUT YOUR CODE HERE  (Hint: one or two lines in each case)
useDepot.lo['DEP1'] = 1
fixedcost.solve()
print(f'Depot 1 open: {fixedcost.objective_value}')
# CODE TO PRINT OPEN Depots
display(useDepot.records)


# What if Depot 2 has to be open?
useDepot.lo['DEP1'] = 0
useDepot.lo['DEP2'] = 1
# PUT YOUR CODE HERE
fixedcost.solve()
print(f'Depot 2 open: {fixedcost.objective_value}')
display(useDepot.records)
# CODE TO PRINT OPEN Depots

# What if both 1 & 2 have to be open?
useDepot.lo['DEP1'] = 1
useDepot.lo['DEP2'] = 1

# PUT YOUR CODE HERE
fixedcost.solve()
print(f'Depot 1 and 2 open: {fixedcost.objective_value}')
# CODE TO PRINT OPEN Depots
display(useDepot.records)

useDepot.lo['DEP1'] = 0
useDepot.lo['DEP2'] = 0
```

```
Depot 1 open: 42371.0
```

	d	level	marginal	lower	upper	scale
0	DEP1	1.0	3200.0	1.0	1.0	1.0
1	DEP2	0.0	-31500.0	0.0	1.0	1.0
2	DEP3	1.0	-7000.0	0.0	1.0	1.0
3	DEP5	1.0	4000.0	0.0	1.0	1.0
4	DEP6	1.0	2100.0	0.0	1.0	1.0
5	DEP9	0.0	-3760.0	0.0	1.0	1.0

```
Depot 2 open: 42129.0
```

	d	level	marginal	lower	upper	scale
0	DEP1	0.0	-22400.0	0.0	1.0	1.0
1	DEP2	1.0	2500.0	1.0	1.0	1.0
2	DEP3	1.0	-11800.0	0.0	1.0	1.0
3	DEP5	1.0	4000.0	0.0	1.0	1.0
4	DEP6	1.0	-900.0	0.0	1.0	1.0
5	DEP9	0.0	-16500.0	0.0	1.0	1.0

```
Depot 1 and 2 open: 42835.0
```

	d	level	marginal	lower	upper	scale
0	DEP1	1.0	3200.0	1.0	1.0	1.0
1	DEP2	1.0	2500.0	1.0	1.0	1.0
2	DEP3	1.0	-7000.0	0.0	1.0	1.0
3	DEP5	1.0	4000.0	0.0	1.0	1.0
4	DEP6	1.0	2100.0	0.0	1.0	1.0
5	DEP9	0.0	-3760.0	0.0	1.0	1.0