SNA PROJECT

ANALYSIS OF FREQUENTLY BOUGHT AMAZON PRODUCTS

Exploring the Amazon Product Co-Purchasing Network: A Social Network Analysis Approach

INTRODUCTION

The Amazon product co-purchasing network is a rich and complex dataset that provides valuable insights into the behavior of customer purchasing patterns on amazon.com. In this project report, we use this dataset to explore the co-purchasing relationships between products sold on Amazon and investigate the structure of the underlying graph. Specifically, we aim to answer the following research questions:

- What is the overall structure of the Amazon product co-purchasing network?
- What are the most frequently co-purchased products on Amazon?
- Are there any communities or clusters of products that tend to be co-purchased together?
- How can we use the co-purchasing network to make recommendations for new products to Amazon customers?

- To answer these questions, we use a combination of network analysis, graph mining, and machine learning techniques. We first visualize the co-purchasing network using various graph visualization tools and identify the key structural properties of the network, such as degree distribution, clustering coefficient, and centrality measures. We then use community detection algorithms to identify clusters of products that are frequently co-purchased together.

About Dataset

Network was collected by crawling Amazon website. It is based on Customers Who Bought This Item Also Bought feature of the Amazon website. If a product I is frequently co-purchased with product j, the graph contains a directed edge from i to j.

The data was collected on March 02, 2003.

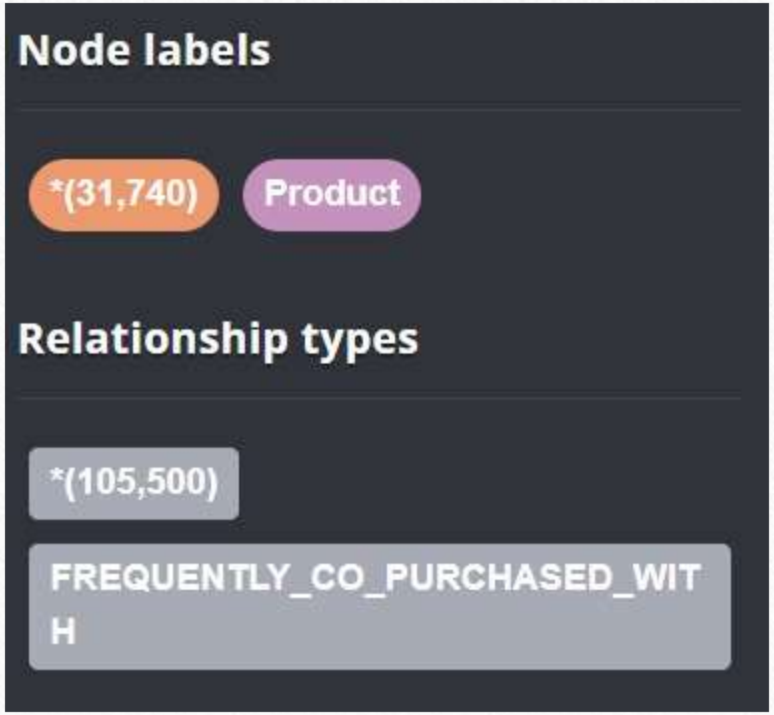
Dataset statistics

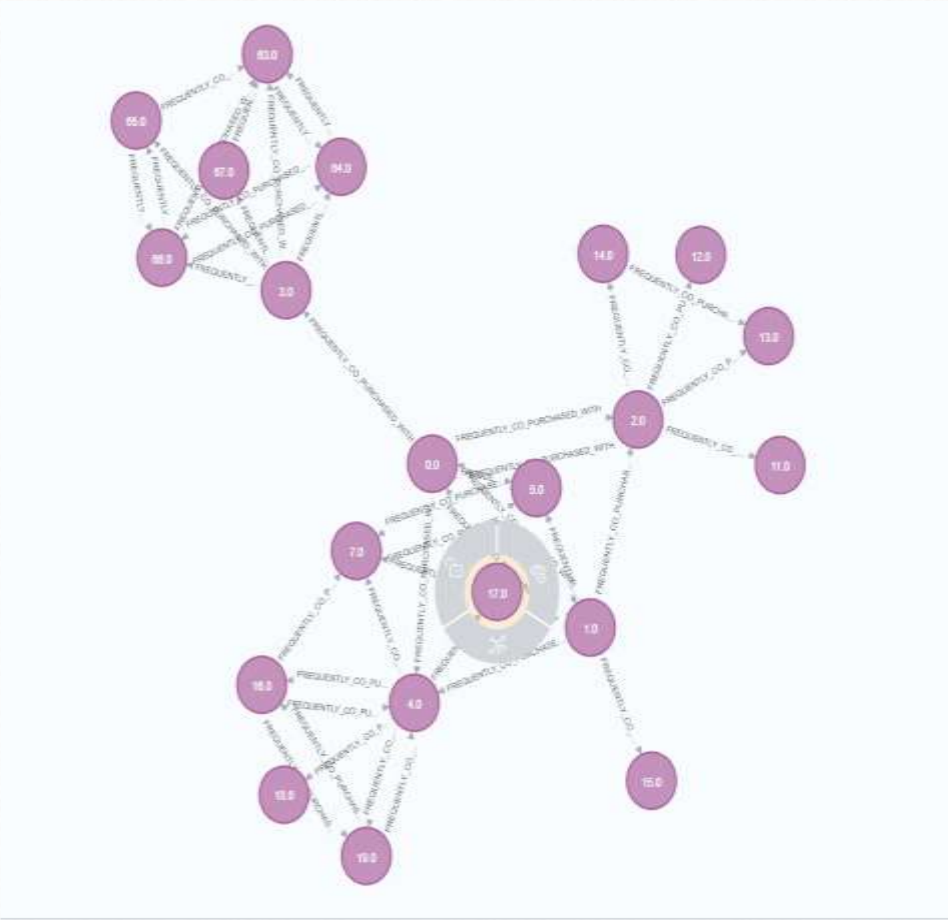
Nodes	262111
Edges	1234877
Nodes in largest WCC	262111 (1.000)
Edges in largest WCC	1234877 (1.000)
Nodes in largest SCC	241761 (0.922)
Edges in largest SCC	1131217 (0.916)
Average clustering coefficient	0.4198
Number of triangles	717719
Fraction of closed triangles	0.09339
Diameter (longest shortest path)	32
90-percentile effective diameter	11

Platform/Technologies

- Python
- Gephi
- Neo4j
- Apache Spark
- Graph Data Science Plugin
- AOC Plugin

LOAD DATASET INTO NEO4J AND SELECT 1,00,000 NODES





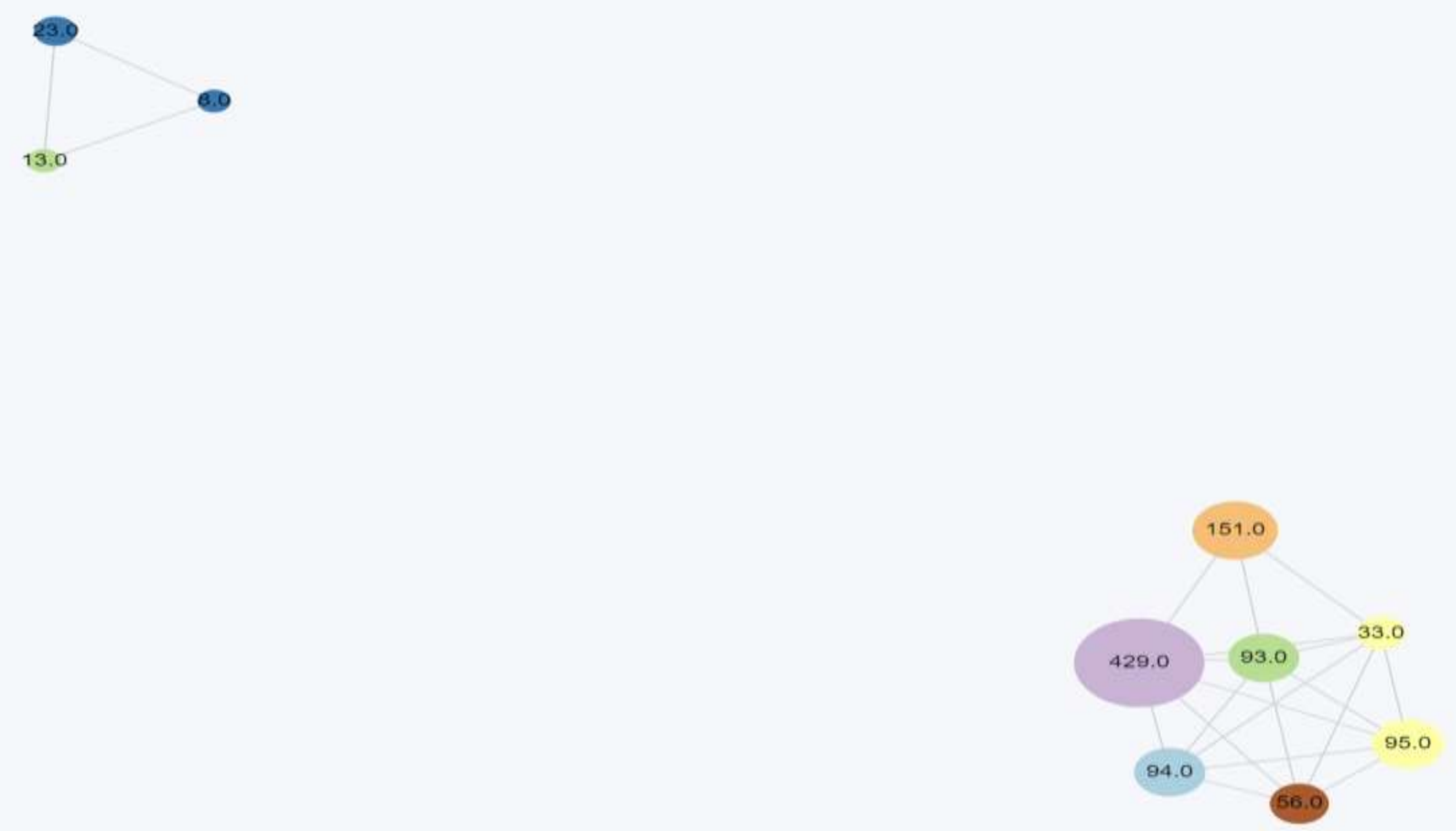
Subgraph Of 25 Nodes

TOP 10 NODES -> PAGE RANK ALGORITHM

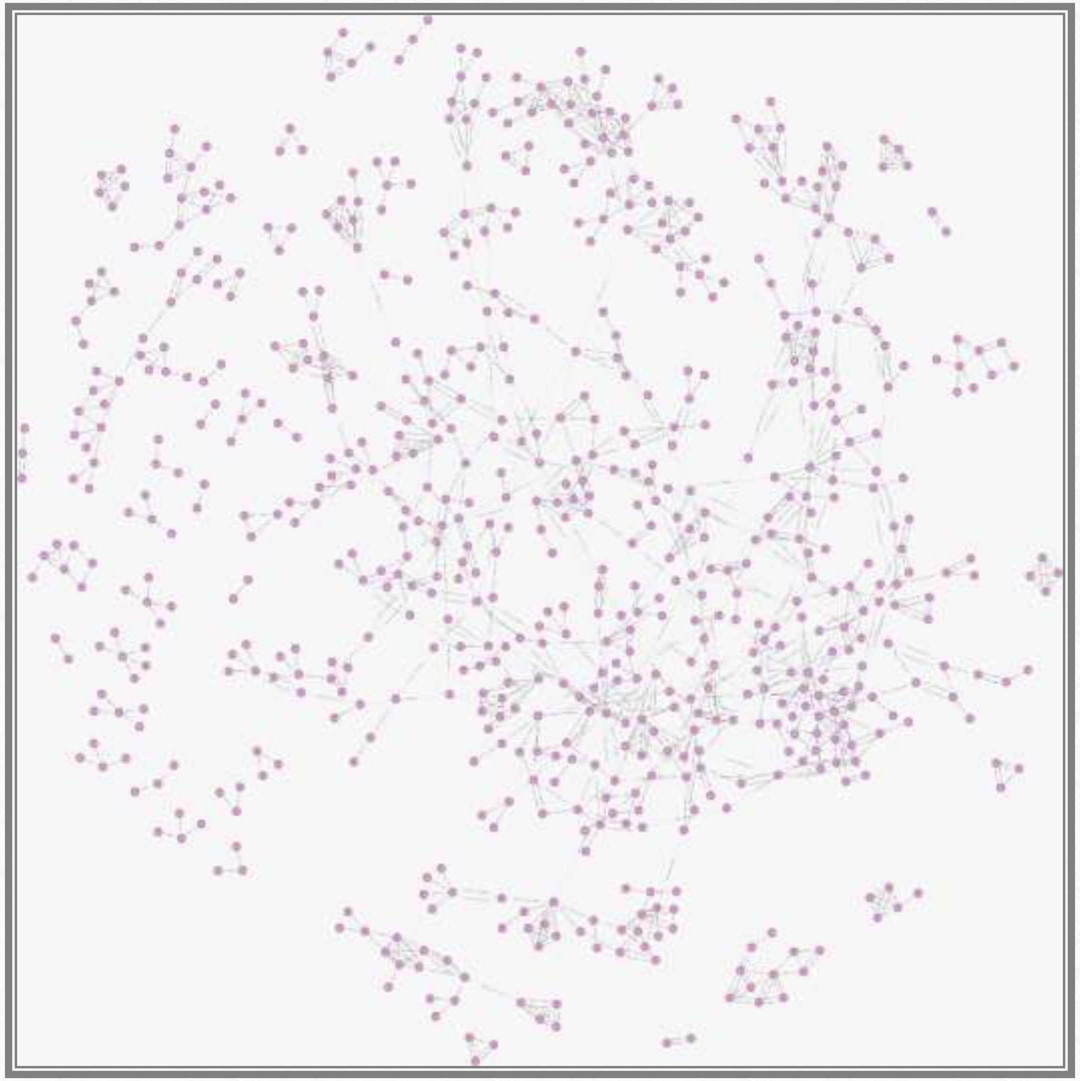
Product

Node	Score
33.0	74.1792168932472
8.0	66.8589496038479
93.0	66.39815629429279
94.0	54.44812982903213
56.0	47.896279133504315
151.0	46.847550216383965
95.0	45.982990338867786
23.0	44.651698930314616
429.0	35.00443519436009
13.0	32.537582883822076

GRAPH OF TOP 10 NODES – Page Rank



- Sub-Graph Of 1000 Nodes



LABEL PROPAGATION – A FAST COMMUNITY FINDING ALGORITHM

LABEL PROPAGATION

The Label Propagation Algorithm is a fast algorithm for finding communities in a graph, which it does by propagating labels and forming communities based on this process of label propagation

On initialisation, every node has its own label, but as labels propagate, densely connected groups of nodes quickly reach a consensus on a unique label. At the end of the propagation only a few labels will remain.

1. Configure | 2. Results | 3. Code

Table

Visualisation

Product

Community	Size	Nodes
401	4619	21505.0 22009.0 22478.0 23122.0 23125.0 23148.0 0.0 1.0 2.0 3.0
455	1553	21506.0 21780.0 22221.0 22237.0 22474.0 22502.0 22696.0 22698.0 22897.0 22898.0
666	413	170.0 213.0 214.0 215.0 216.0 217.0 324.0 325.0 326.0 327.0
431	363	18.0 32.0 34.0 35.0 36.0 51.0 52.0 53.0 54.0 55.0
1291	227	788.0 927.0 928.0 930.0 931.0 1019.0 1020.0 1061.0 1062.0 1063.0
657	188	21804.0 150.0 204.0 205.0 206.0 207.0 233.0 234.0 246.0 247.0
2826	166	23327.0 23328.0 2353.0 2501.0 2505.0 3587.0 3588.0 3589.0 4223.0 4429.0
467	130	15.0 69.0 70.0 71.0 72.0 102.0 103.0 68.0 105.0 106.0
1245	123	702.0 879.0 880.0 881.0 882.0 883.0 973.0 974.0 975.0 976.0
3539	113	2439.0 2440.0 3230.0 3231.0 3232.0 3233.0 3234.0 3316.0 3318.0 3338.0
15749	95	23102.0 12289.0 12371.0 12372.0 12373.0 12374.0 12375.0 12769.0 12770.0 13433.0

TOP 10 COMMUNITIES

LOUVIAN - COMMUNITY FINDING ALGORITHM

CATEGORISE UNSTRUCTURED DATA

LOUVAIN

The Louvain method for community detection is an algorithm for detecting communities in networks. It maximizes a modularity score for each community, where the modularity quantifies the quality of an assignment of nodes to communities.

1. Configure 2. Results 3. Code

Table

Visualisation

Product

Community	Communities	Size	Nodes
7039		1939	21505.0 21506.0 21779.0 21780.0 21978.0 22221.0 22237.0 22474.0 22502.0 22696.0
6903		1918	21976.0 21977.0 22242.0 22511.0 23122.0 23329.0 23330.0 23334.0 23335.0 23336.0
16633		1008	21514.0 21515.0 21803.0 21998.0 22496.0 22497.0 22498.0 22499.0 22512.0 22737.0
5330		1007	21984.0 21985.0 22240.0 22245.0 230.0 231.0 302.0 303.0 333.0 335.0
7926		912	22002.0 22226.0 22238.0 22753.0 23148.0 23180.0 23181.0 23182.0 1094.0 1118.0
5683		903	21509.0 21511.0 21512.0 21800.0 21969.0 21970.0 21993.0 21994.0 21995.0 21996.0
18154		860	22490.0 22491.0 23135.0 23184.0 23185.0 23191.0 23192.0 23193.0 1119.0 2655.0
8084		834	21999.0 22000.0 22896.0 22915.0 22916.0 22917.0 22918.0 23149.0 23150.0 23151.0
2941		809	22473.0 22738.0 22741.0 22742.0 22743.0 23152.0 23153.0 23154.0 23155.0 22.0
5652		805	22938.0 15.0 69.0 70.0 71.0 72.0 102.0 103.0 68.0 105.0

KNN ALGORITHM- JACCORD SIMILARITY METRIC

K-NEAREST NEIGHBORS

The K-Nearest Neighbors algorithm computes a distance value for all node pairs in the graph and creates new relationships between each node and its k nearest neighbors. The distance is calculated based on node properties.

1. Configure 2. Results 3. Code

Table Product

From	Nodes
21051.0	57700.0 1 25137.0 1 34081.0 1 20608.0 1 54487.0 1 57699.0 1 11438.0 1 24906.0 1 25105.0 1 30248.0 1
21053.0	29872.0 1 8370.0 1 29871.0 1 26789.0 1 36686.0 1 28428.0 1 21271.0 1 20818.0 1 26808.0 1 28121.0 1
21054.0	17571.0 1 34039.0 1 22999.0 1 13884.0 1 14423.0 1 29672.0 1 55741.0 1 16667.0 1 21472.0 1 43300.0 1
21055.0	28814.0 1 20021.0 1 14422.0 1 35198.0 1 37102.0 1 36075.0 1 23867.0 1 39961.0 1 28583.0 1 13976.0 1
21056.0	54473.0 1 20843.0 1 21951.0 1 8367.0 1 41128.0 1 20941.0 1 27643.0 1 11914.0 1 18259.0 1 24700.0 1
21058.0	41451.0 1 53501.0 1 12559.0 1 19897.0 1 17105.0 1 55719.0 1 22549.0 1 21296.0 1 17581.0 1 53718.0 1
21060.0	20329.0 1 34848.0 1 35600.0 1 25324.0 1 28422.0 1 16085.0 1 19244.0 1 46298.0 1 18084.0 1 18105.0 1
21062.0	261702.0 1 57695.0 1 41137.0 1 31187.0 1 49973.0 1 37445.0 1 261703.0 1 37444.0 1 31185.0 1 23725.0 1
21063.0	29190.0 1 16853.0 1 28889.0 1 41137.0 1 14861.0 1 24031.0 1 24985.0 1 31189.0 1 24028.0 1 20291.0 1
21064.0	23186.0 1 21029.0 1 16852.0 1 13972.0 1 41445.0 1 29226.0 1 35817.0 1 18368.0 1 32079.0 1 26838.0 1

KNN ALGORITHM- EUCLIDEAN SIMILARITY METRIC

K-NEAREST NEIGHBORS

The K-Nearest Neighbors algorithm computes a distance value for all node pairs in the graph and creates new relationships between each node and its k nearest neighbors. The distance is calculated based on node properties.

1. Configure 2. Results 3. Code

Table: Product

From	Nodes
21051.0	17512.0 1 · 47835.0 1 · 22096.0 1 · 34080.0 1 · 24891.0 1 · 20761.0 1 · 18081.0 1 · 20608.0 1 · 22724.0 1 · 54112.0 1
21053.0	72990.0 1 · 41130.0 1 · 13882.0 1 · 23868.0 1 · 53317.0 1 · 27897.0 1 · 33419.0 1 · 32354.0 1 · 37723.0 1 · 31666.0 1
21054.0	6491.0 1 · 15219.0 1 · 15350.0 1 · 21472.0 1 · 18260.0 1 · 20817.0 1 · 72990.0 1 · 26754.0 1 · 25347.0 1 · 41128.0 1
21055.0	27896.0 1 · 38098.0 1 · 23869.0 1 · 35198.0 1 · 28670.0 1 · 28285.0 1 · 11912.0 1 · 39069.0 1 · 22658.0 1 · 23887.0 1
21056.0	16666.0 1 · 37539.0 1 · 22521.0 1 · 48841.0 1 · 19236.0 1 · 11779.0 1 · 19703.0 1 · 36466.0 1 · 11021.0 1 · 24113.0 1
21058.0	55720.0 1 · 41450.0 1 · 29784.0 1 · 16182.0 1 · 55719.0 1 · 29479.0 1 · 12886.0 1 · 45103.0 1 · 18890.0 1 · 20124.0 1
21060.0	30534.0 1 · 12302.0 1 · 22959.0 1 · 36918.0 1 · 6459.0 1 · 12376.0 1 · 22218.0 1 · 45092.0 1 · 25087.0 1 · 34530.0 1
21062.0	7094.0 1 · 34445.0 1 · 37445.0 1 · 21063.0 1 · 29510.0 1 · 21031.0 1 · 20291.0 1 · 30620.0 1 · 57726.0 1 · 23724.0 1
21063.0	29510.0 1 · 31189.0 1 · 21062.0 1 · 261702.0 1 · 261703.0 1 · 37445.0 1 · 34445.0 1 · 36062.0 1 · 29190.0 1 · 7094.0 1
21064.0	23553.0 1 · 261702.0 1 · 23188.0 1 · 16853.0 1 · 37445.0 1 · 29266.0 1 · 21061.0 1 · 12856.0 1 · 21030.0 1 · 24029.0 1

KNN ALGORITHM – COSINE SIMILARITY

K-NEAREST NEIGHBORS

The K-Nearest Neighbors algorithm computes a distance value for all node pairs in the graph and creates new relationships between each node and its k nearest neighbors. The distance is calculated based on node properties.

1. Configure 2. Results 3. Code

Table Product

From	Nodes
21051.0	14873.0 1 54114.0 1 31294.0 1 12406.0 1 22723.0 1 20612.0 1 35148.0 1 25137.0 1 35147.0 1 21504.0 1
21053.0	27633.0 1 22658.0 1 28121.0 1 7939.0 1 26754.0 1 24712.0 1 36119.0 1 41131.0 1 13882.0 1 53340.0 1
21054.0	53317.0 1 19703.0 1 36075.0 1 27897.0 1 23869.0 1 33087.0 1 33420.0 1 54473.0 1 29727.0 1 55742.0 1
21055.0	6493.0 1 20186.0 1 28122.0 1 28258.0 1 29873.0 1 82759.0 1 9437.0 1 45732.0 1 28428.0 1 1874.0 1
21056.0	20546.0 1 28672.0 1 16808.0 1 27635.0 1 19704.0 1 41128.0 1 21066.0 1 16016.0 1 40380.0 1 6491.0 1
21058.0	53288.0 1 56072.0 1 33942.0 1 9386.0 1 7585.0 1 6963.0 1 38453.0 1 31501.0 1 13294.0 1 15170.0 1
21060.0	35185.0 1 49405.0 1 29894.0 1 52566.0 1 29782.0 1 51480.0 1 39370.0 1 16549.0 1 30535.0 1 41468.0 1
21062.0	35816.0 1 23724.0 1 18368.0 1 20565.0 1 36062.0 1 21029.0 1 21063.0 1 23725.0 1 20389.0 1 30621.0 1
21063.0	16854.0 1 23186.0 1 61370.0 1 21062.0 1 41137.0 1 32079.0 1 31185.0 1 20933.0 1 16852.0 1 23553.0 1
21064.0	57324.0 1 29192.0 1 25577.0 1 34445.0 1 29263.0 1 30620.0 1 14555.0 1 27768.0 1 21061.0 1 12856.0 1

FUTURE WORK

- INCLUDE PRODUCT META DATA IN THIS DATABASE AND MATCH THEIR ID INDEX WI.