## Guide - MITO Topology Import

### General Overview

This document provides a comprehensive guide for importing data into the **MITO Topology** system using Neo4j. The system manages network inventory through a graph model that represents configuration items (CiItems), their relationships, and logical organization.

**Key Features**

* **Database**: Neo4j (Graph Database)
* **Model**: Graph with nodes and relationships
* **Import**: Batch processing via CSV
* **Scalability**: Optimized for large data volumes
* **Integrity**: Constraints and indexes to ensure consistency

**Important Note on Sample Data**

The provided CSV files contain:

* **First row**: Header with required data model (DO NOT MODIFY)
* **Subsequent rows**: Sample data to be replaced with your real data

### System Architecture

**Main Entities**

| **Entity** | **Description** | **Primary Key** |
| --- | --- | --- |
| **Topology** | Logical groupings of network elements | name |
| **Layer** | Network abstraction levels | name |
| **ItemType** | Types of configurable elements | itemTypeName |
| **CiItem** | Configuration items | name |
| **Location** | Geographic positions | name |

**Main Relationships**

Topology -[HAS\_ELEMENT]-> CiItem

Layer -[IS\_LAYER\_OF]-> CiItem

Layer -[IS\_PARENT\_OF]-> Layer

Location -[IS\_LOCATION\_OF]-> CiItem

CiItem -[*business\_model\_relations*]-> CiItem

**Layer Hierarchical Model**

Service Circuit (Layer 3)

↓ IS\_PARENT\_OF

Ethernet (Layer 2)

↓ IS\_PARENT\_OF

Physical (Layer 1)

### Prerequisites

**Required Software**

* **Neo4j Desktop/Server** (version 4.4+)
* **APOC Plugin** installed and configured
* **Neo4j Spatial Plugin** for geographic coordinate management

**Neo4j Configuration**

# neo4j.conf

dbms.security.procedures.unrestricted=apoc.\*,spatial.\*

dbms.security.procedures.allowlist=apoc.\*,spatial.\*

### Complete Summary Table

Refer to table.html

### CSV Data Models

#### PHASE 1 - Base Entities

* **layer.csv**

*operation,name,mandatory*

**Required Fields**:

* operation:

A for Add

U for Update

D for delete

* name: Unique layer name

**Optional Fields**:

* mandatory: Y/N indicates if layer is mandatory
* **itemType.csv**

*operation,itemTypeName,defaultLayers*

**Required Fields**:

* operation:

A for Add

U for Update

D for delete

* + itemTypeName: Unique element type name

**Optional Fields**:

* dafaultLayers: list of layers separated by delimiter

#### PHASE 2 - Inventory

**locations.csv**

* *operation,name,latitude,longitude,country\_name,state\_name,city\_name,address,postal\_code*

**Required Fields**:

* operation:

A for Add

U for Update

D for delete

* name: Unique location name
* latitude/longitude: Geographic coordinates (decimal format)

**Optional Fields**:

* *country\_name*
* *state\_name*
* *city\_name*
* *address*
* *postal\_code*

**ciItems.csv**

*operation,name,itemTypeName,status,accountName,importance,isLink,fromElement,toElement,latitude,longitude,layers,url,propKey\_1,propVal\_1,propKey\_2,propVal\_2………………………*

**Required Fields**:

* operation:

A for Add

U for Update

D for delete

* name: Unique element name MUST BE USED FOR ALL FOREING KEYS of other entity
* itemTypeName: Must exist in itemType.csv
* accountName: Account name must exist and defined in mito
* isLink: Y/N indicates if it's a link
* fromElement/toElement: IDs of connected elements (for links only)

**Optional fields Fields**:

* importance: Importance level
* layers: list of the layer names where ciitem belongs to separated by delimiter
* url : callback url
* status: Element status

**Dynamic Properties**:

* propKey\_N / propVal\_N: Key-value pairs for additional properties
* N must be sequential (1,2,3,...)
* **topology.csv**

*operation,name,description,propKey\_1,propVal\_1,propKey\_2,propVal\_2………………………*

**Required Fields**:

* operation:

A for Add

U for Update

D for delete

* + name: Unique topology name

**Optional Fields**:

* description: Text description

**Dynamic Properties**:

* propKey\_N / propVal\_N: Key-value pairs for additional properties
* N must be sequential (1,2,3,...)

#### PHASE 3 - Relations

* **location\_rel.csv**

location\_name,ciitem\_name

* **ciitem\_rel.csv**

*source\_name,target\_name,relation*

* **topology\_rel.csv**

*topology\_name,ciitem\_name*

### DRAFT!!! --- Execution Procedure -ONLY FOR MITO TEAM

##### PHASE 0: Database Setup and Configuration

**Purpose**: Initialize database with indexes, constraints, and spatial capabilities

**Dependencies**: None (first execution)

**Scripts**:

* F0\_SETUP.txt - Creates all indexes and constraints
* F0\_B.spatial\_layer\_F0.txt - Initializes spatial layer for geographic data

##### PHASE 1: Import Base Entities

**Purpose**: Import fundamental system entities and establish layer hierarchy

**Dependencies**: PHASE 0

**Required CSV Files**:

* layer.csv - Network layer definitions
* itemType.csv - Equipment and service type catalog
* layer\_hierarchy.csv - Parent-child layer relationships

**Scripts**:

* 1.F1.import\_basic.txt - Import Layer, ItemType, Account entities
* 2.F1.import\_layer\_hierarchy\_rel.txt - Create layer hierarchy relationships

##### PHASE 2: Import Inventory

**Purpose**: Import physical and logical network inventory

**Dependencies**: PHASE 0 + PHASE 1

**Required CSV Files**:

* locations.csv - Geographic locations with coordinates
* ciItems.csv - Network elements with properties and relationships
* topology.csv - Topology definitions and metadata

**Scripts**:

* 1.F2.import\_location.txt - Import locations with spatial indexing
* 2.F2.import\_items.txt - Import network elements with dynamic properties
* 3.F2.import\_topology.txt - Import topology definitions

##### PHASE 3: Import Base Relations

**Purpose**: Establish default layer assignments and location mappings

**Dependencies**: PHASE 0 + PHASE 1 + PHASE 2

**Required CSV Files**:

* layer\_rel.csv - Manual layer assignments (optional, supplements defaults)
* location\_rel.csv - Element-to-location mappings

**Scripts**:

* 1.F3.import\_defaultLayer\_rel.txt - Auto-assign elements to default layers
* 2.F3.import\_location\_rel.txt - Create location-element relationships

##### PHASE 4: Import Items Relations

**Purpose**: Create network interconnections and topology assignments

**Dependencies**: PHASE 0 + PHASE 1 + PHASE 2

**Required CSV Files**:

* ciitem\_rel.csv - Element-to-element business relationships
* topology\_rel.csv - Topology-to-element assignments

**Scripts**:

* 1.F4.import\_items\_rel.txt - Create network interconnections
* 2.F4.import\_topology\_rel.txt - Assign elements to topologies

#### Cleanup Scripts

**DELETE\_ALL.txt**

//DELETE ENTIRE GRAPH (node + relation)

//MATCH (n) DETACH DELETE n

//OR

//DELETE RELATIONS ONLY

//MATCH ()-[r]->() DELETE r;

**DELETE\_INDEXES.txt**

// WARNING: Use only if you need to remove indexes

/\*

DROP CONSTRAINT topology\_id\_unique;

DROP CONSTRAINT layer\_id\_unique;

DROP CONSTRAINT itemtype\_id\_unique;

DROP CONSTRAINT account\_id\_unique;

DROP INDEX topology\_name\_index;

DROP INDEX layer\_name\_index;

DROP INDEX itemtype\_name\_index;

DROP INDEX account\_name\_index;

DROP INDEX itemtype\_defaultlayer\_index;

DROP INDEX topology\_createdat\_index;

DROP INDEX topology\_updatedat\_index;

\*/