# Network Interdependency Database Schema Documentation

# Overview

This document details the functions that exist within the base network\_interdependency database, including any trigger functions that execute on specific tables. The document separates the callable functions from these auto-operating trigger functions.

# Callable Functions

**Function Name:** ni\_reset\_database

**Function Description:** Resets the network interdependency database by removing all tables other than Nodes, Edges, Edge\_Geometry, Graphs and Global Interdependency, as well as resetting the serial keys for Graphs and Global Interdependency. Finally all records from the geometry\_columns table are also deleted.

**Parameters in:**

* Schema\_name: string

**Returns:**

* Void

**Function Name:** ni\_delete\_network

**Function Description:** Deletes all tables associated with the network prefix supplied, including interdependencies as well as removing associated records from the Graphs and Global Interdependency tables.

**Parameters in:**

* Network\_name\_prefix: string

**Returns:**

* Void

**Function Name:** ni\_check\_srid

**Function Description:** Checks against the PostGIS spatial\_ref\_sys table that the supplied EPSG number e.g. 27700 exists i.e. is a valid SRID value.

**Parameters in:**

* SRID: integer

**Returns:**

* Boolean

**Function Name:** ni\_create\_network\_tables

**Function Description:** Function to create empty inherited Nodes, Edges and Edge\_Geometry tables for a particular network, based on the table prefix and SRID values supplied e.g. if the table prefix is ***transport***, then ***transport\_Nodes, transport\_Edges*** and ***transport\_Edge\_Geometry*** tables will be created. The function calls the individual functions ni\_create\_network\_table\_nodes, ni\_create\_network\_table\_edge\_geometry and ni\_create\_network\_table\_edges to create the tables (see later function definitions for more details).

**Parameters in:**

* Table\_prefix: string
* SRID: integer

**Returns:**

* Boolean

**Function Name:** ni\_create\_network\_table\_nodes

**Function Description:** Function to create an empty inherited Nodes table, based on the table prefix and SRID supplied e.g. if the table prefix is ***transport***, then ***transport\_Nodes*** will be created.

**Parameters in:**

* Table\_prefix: string
* SRID: integer

**Returns:**

* Boolean

**Function Name:** ni\_create\_network\_table\_edges

**Function Description:** Function to create an empty inherited Edge table, based on the table prefix supplied e.g. if the table prefix is ***transport***, then ***transport\_Edges*** will be created. No SRID is required here because the geometry for the Edge is stored in a separate table Edge\_Geometry

**Parameters in:**

* Table\_prefix: string

**Returns:**

* Boolean

**Function Name:** ni\_create\_network\_table\_edge\_geometry

**Function Description:** Function to create an empty edge\_geometry table, based on the table prefix and SRID supplied e.g. if the table prefix is ***transport***, then ***transport\_Edge\_Geometry*** will be created.

**Parameters in:**

* Table\_prefix: string

**Returns:**

* Boolean

**Function Name:** ni\_add\_geometry\_columns

**Function Description:** Function to add the references to the inherited Nodes and Edge\_Geometry table to the PostGIS geometry\_columns table, based on the table prefix supplied e.g. if the table prefix is ***transport*** then ***transport\_Nodes*** and ***transport\_Edge\_Geometry*** will be created.

**Parameters in:**

* Table\_prefix: string
* SRID: integer

**Returns:**

* Boolean

**Function Name:** ni\_add\_fr\_constraints

**Function Description:** Function to add the appropriate foreign key constraints to the Nodes, Edges, and Edge\_Geometry tables.

* Foreign key from Nodes table to Graphs table (GraphID to GraphID)
* Foreign key from Edges table to Nodes table (Node\_F\_ID to NodeID)
* Foreign key from Edges table to Nodes table (Node\_T\_ID to NodeID)
* Foreign key from Edges table to Edge\_Geometry table (Edge\_GeomID to GeomID)
* Foreign key from Edges table to Graphs table (GraphID to GraphID)

**Parameters in:**

* Table\_prefix: string

**Returns:**

* Boolean

**Function Name:** ni\_add\_graph\_record

**Function Description:** Function to add a new record to the graphs table based on the table prefix supplied, and Boolean values indicating if the graph is directed or a multigraph.

**Parameters in:**

* Table\_prefix: string
* Directed: Boolean (true if directed, false otherwise)
* MultiGraph: Boolean (true if directed, false otherwise)

**Returns:**

* Void

**Function Name:** ni\_create\_edge\_view

**Function Description:** Function to create a view that joins the Edge and Edge\_Geometry tables based on the supplied table prefix. It also sets the appropriate geometry constraints on the view, and adds its table to the PostGIS geometry\_columns table.

**Parameters in:**

* Table\_prefix: string

**Returns:**

* New\_edge\_edge\_geometry\_view table name: string

**Function Name:** ni\_create\_interdependency\_edge\_view

**Function Description:** Function to create a view that joins the Interdependency and Interdependency\_Edge tables based on the supplied table prefix. It also sets the appropriate geometry constraints on the view, and adds its table to the PostGIS geometry\_columns table.

**Parameters in:**

* Table\_prefix: string

**Returns:**

* New\_interdependency and interdependency\_edge \_view table name: string

**Function Name:** ni\_create\_interdependency\_tables

**Function Description:** Function to create the empty interpendency and interdependency\_edge tables, based on the supplied table prefixes e.g. if the table prefixes are ***transport*** and ***energy,*** then the resulting tables will be ***transport\_energy\_Interdependency*** and ***transport\_energy\_Interdependency\_Edge.*** This function also adds the appropriate foreign key constraints:

* New Interdependency table to Graphs table (Interdependency\_Graphs\_F\_GraphID to GraphID
* New Interdependency table to Graphs table (Interdependency\_Graphs\_T\_GraphID to GraphID
* New Interdependency table to the equivalent Nodes table (Interdependency\_Nodes\_F\_NodeID to NodeID)
* New Interdependency table to the equivalent Nodes table (Interdependency\_Nodes\_T\_NodeID to NodeID)

This function also adds a record to the Global\_Interdependency tables once they have been created.

**Parameters in:**

* Table\_prefix\_network\_1: string
* Table\_prefix\_network\_2: string

**Returns:**

* Boolean

**Function Name: ni\_edge\_geometry\_equality\_check**

**Function Description:** A function to determine if the input edge geometry matches an edge geometry that already exists in the edge geometry table. This exploits the ST\_Equals PostGIS function to determine equality.

**Parameters in:**

* Table prefix: string
* Geometry WKT: string
* SRID: integer

**Returns:**

* Matched geometry id: integer (-1 if not matched)

**Function Name: ni\_node\_geometry\_equality\_check**

**Function Description:** A function to determine if the input node geometry matches a node that already exists in the Node table. This exploits the ST\_Equals PostGIS function to determine equality.

**Parameters in:**

* Table prefix: string
* Geometry WKT: string
* SRID: integer

**Returns:**

* Matched geometry id: integer (-1 if not matched)

**Returns:**

**Function Name: ni\_check\_network\_tables**

**Function Description:** A function that checks whether the 3 necessary tables forming a network or graph within the database schema are present i.e. \_Nodes, \_Edges, \_Edge\_Geometry.

**Parameters in:**

* Table prefix: string

**Returns:**

* Boolean

**Function Name: ni\_check\_interdependency\_tables**

**Function Description:** A function that checks whether the 2 necessary tables exist that form a valid interdependency within the database schema i.e. \_Interdependency and \_Interdependency\_Edge.

**Parameters in:**

* Table prefix network 1: string
* Table prefix network 2: string

**Returns:**

* Boolean

**Function Name: ni\_graph\_to\_csv**

**Function Description:** A function to return all tables related to the specified input table prefix as .csv files, including all interdependency tables and views

**Parameters in:**

* Table prefix: string
* CSV File prefix: string
* CSV File path: string

**Returns:**

* Void

# Trigger Functions

**Function Name**: check\_record\_geometry\_columns\_table\_post\_graph\_insert

**Function Description:** A function to check that the values for Nodes and Edges in the newly added Graphs record, are tables that have been added to the geometry\_columns table. If the tables have not been added, this trigger will add the reference.

**Acts on:** Graphs

**When:** After Insert

**Row / Statement:** Row

**Function Name: check\_record\_geometry\_columns\_table\_post\_graph\_nodes\_update**

**Function Description:** A function to check that the newly updated value for the Nodes column of the Graphs table i.e. name of a Node table, has been added to the geometry\_columns table. If the table has not been added, this trigger will add the reference.

**Acts on:** Nodes of Graphs

**When:** After Update

**Row / Statement:** Row

**Function Name: check\_record\_geometry\_columns\_table\_post\_graph\_edges\_update**

**Function Description:** A function to check that the corresponding Edge\_Geometry table to the newly updated value for the Edges column of the Graphs table, has been added to the geometry\_columns table. If the table has not been added, this trigger will add the reference to the Edge\_Geometry table.

**Acts on:** Edges of Graphs

**When:** After Update

**Row / Statement:** Row

**Function Name:** delete\_record\_geometry\_columns\_table\_post\_delete\_graph

**Function Description:** A function to delete the tables in the geometry\_columns table once a record has been deleted from the Graphs table i.e. will remove any Node, Edge\_Geometry references

**Acts on:** Graphs

**When:** After Delete

**Row / Statement:** Row

**Function Name:** delete\_int\_tables\_post\_delete\_graph

**Function Description:** A function to remove the interdependency, interdependency\_edge tables and any related views when a record is deleted from the Graphs table.

**Acts on:** Graphs

**When:** After Delete

**Row / Statement:** Row

**Function Name:** delete\_nodes\_table\_post\_delete\_graph

Function Description: A function to delete the Nodes table, based on the recent value stored in the Nodes column of the recently deleted Graph record.

**Acts on:** Graphs

**When:** After Delete

**Row / Statement:** Row

**Function Name:** delete\_edges\_table\_post\_delete\_graph

**Function Description:** A function to delete the Edges table, based on the recent value stored in the Edges column of the recently deleted Graph record.

**Acts on:** Graphs

**When:** After Delete

**Row / Statement:** Row

**Function Name: delete\_edges\_geometry\_table\_post\_delete\_graph**

**Function Description:** A function to delete the Edge\_Geometry table, based on the recent value stored in the Edges column of the recently deleted Graph record.

**Acts on:** Graphs

**When:** After Delete

**Row / Statement:** Row

**Function Name: delete\_int\_table\_post\_int\_record\_delete**

**Function Description:** A function to delete the corresponding interdependency table based on the recent value stored in InterdependencyTableName of the deleted record from Global\_Interdependency

**Acts on:** Global\_Interdependency

**When:** After Delete

**Row / Statement:** Row

**Function Name:** **delete\_global\_int\_record\_post\_graph\_record\_delete**

**Function Description:** A function to delete the records from the Global\_Interdependency table equivalent to the recently deleted graphs deleted from the Graphs table.

**Acts on:** Graphs

**When:** After Delete

**Row / Statement:** Row