GEO - Graph Evolution Operation

Dominique Hausler

University of Regensburg
Data Engineering Group, Faculty of Informatics and Data Science
dominique.hausler@ur.de

September 4, 2023

```
(*add operation*)
add ::= addReplicable | addUnique
addReplicable ::= addEntities | addFeatures
addEntityTypes ::= "add" entityTypes "to" database
addFeatures ::= "add" features "to" entityTypes
addUnique ::= (addEntityTypes | addFeatures) ("if created add" properties)? ("if unique add"
    properties)?

(*delete operation*)
delete ::= restrictedDelete | cascadeDelete | remove
restrictedDelete ::= "delete" entityTypes "from" database
cascadeDelete ::= "delete" connectedNodes "from" database
remove ::= "remove" features "from" entityTypes

(*rename operation*)
rename ::= "rename" features "to new" name
name ::= String

(*transform operation*)
transform :: = nodeToRel | relToNode | propToNode
nodeToRel ::= "transform" node "with its features into a" relation
relToNode ::= "transform a" relation "with its features into a" node
propToNode ::= "select a" property "of n" nodes "and transform it into a" node "connected by a"
    relation "to the nodes prior containing this property"

(*merge operation*)
join ::= innerJoin | fullOuterExclusive | fullOuterInclusive | leftJoin | rightJoin
innerJoin ::= "merge duplicate" properties "of" entityTypes "," add (labels | types) "and" (
    cascadeDelete | restrictedDelete) "originals"
fullOuterExclusive ::= "merge different" properties "of" entityTypes "," remove "duplicate"
    properties "and" (copy labels | add labels)
fullOuterInclusive ::= copy properties "of" entityTypes "to" entityTypes "and" (cascadeDelete |
    restrictedDelete) "originals"
leftJoin ::= "keep all features of left" entityTypes "add duplicates from right" entityTypes
rightJoin ::= "keep all features of right" entityTypes "add duplicates from right" entityTypes
```

```
(*copy operation*)
copy ::= copyEntityTypes | copyFeatures | copySubgraph
copyEntityTypes ::= "copy" (connectedNodes | nodes) ("keep" relations | delete relations)
copyFeatures ::= "copy" features "to" entityTypes
copySubgraph ::= "copy" connectedNodes "linked as" subgraph

(*split operation*)
split ::= splitNodes | splitRelations
splitNodes ::= "split" nodes "at" propertyName ("keep" relations | delete relations | "keep"
     relations "of" (partA | partB))
splitRelations ::= "split" relations "at" propertyName (copy endNodes | remove properties "from"
     endNodes "of" (partA | partB) | "overwrite" properties "of" endNodes "of" (partA | partB))
partA ::= properties "till" propertyName
partB ::= properties "from" propertyName

(*move operation*)
move ::= moveNodes | moveRelations | moveFeatures
moveNodes ::= "move" connectedNodes "to new" node
moveRelations ::= "move" realtions "from old" ((startNode "to new" startNode) | (endNode "to new"
     endNode))
moveFeatures ::= "move" features "to" entityTypes


entityType ::= node | relation
entityTypes ::= nodes | relations
node ::= variable (labels (properties)*)*
connectedNodes ::= node "through" relations (rightDirected | leftDirected) "to" node
subgraph ::= connectedNodes (connectedNodes)* "of" database
direction ::= "-"
rightDirected ::= "→"
leftDirected ::= "←"
nodes ::= node ("," node )*
startNode ::= "outgoing relation from" node
endNode ::= "ingoing reation towards" node

relation ::= variable (types (properties)*)*
relations ::= relation ("," relation )*
variable ::= String | Char
feature ::= label | type | property
features ::= labels | types |properties
label ::= String
labels ::= label ("," label )*
type ::= String
types ::= type ("," type )*
property ::= key propertyValue
key ::= String
propertyValue ::= String | Char | Integer | Long | Float | Double | Boolean | Date | Array;
properties ::= property ("," property )*
```