

## PostGIS 2.0 Topology Cheatsheet

New in this release <sup>1</sup> Enhanced in this release <sup>2</sup> Requires GEOS 3.3 or higher<sup>3,3</sup>

### Topology Types

**getfaceedges\_returntype** A composite type that consists of a sequence number and edge number. This is the return type for ST\_GetFaceEdges

**topogeometry** A composite type that refers to a topology geometry in a specific topology, layer, having specific type (1:[multi]point, 2:[multi]line, 3:[multi]poly, 4:collection) with specific identifier id in the topology. The id uniquely defines the topogeometry in the topology.

**validate\_topology\_returntype** A composite type that consists of an error message and id1 and id2 to denote location of error. This is the return type for ValidateTopology

### Topology Domains

**TopoElement** An array of 2 integers generally used to identify a TopoGeometry component.

**topoelementarray** An array of element\_id,element\_type values. a bidimensional array of integers: '{{id,type},{id,type}, ...}'

### Topology and TopoGeometry Management

**AddTopoGeometryColumn** Adds a topogeometry column to an existing table, registers this new column as a layer in topology.layer and returns the new layer\_id.

1. topology\_name, schema\_name, table\_name, column\_name, feature\_type
2. topology\_name, schema\_name, table\_name, column\_name, feature\_type, child\_layer

**DropTopology** (topology\_schema\_name) Use with caution: Drops a topology schema and deletes its reference from topology.topology table and references to tables in that schema from the geometry\_columns table.

**DropTopoGeometryColumn** (schema\_name, table\_name, column\_name) Drops the topogeometry column from the table named table\_name in schema schema\_name and unregisters the columns from topology.layer table.

**TopologySummary**<sup>1</sup> (topology\_schema\_name) Takes a topology name and provides summary totals of types of objects in topology

**ValidateTopology**<sup>2</sup> (topology\_schema\_name) Returns a set of validate\_topology\_returntype objects detailing issues with topology

### Topology Constructors

**CreateTopology** Creates a new topology schema and registers this new schema in the topology.topology table.

1. topology\_schema\_name
2. topology\_schema\_name, srid
3. topology\_schema\_name, srid, tolerance
4. topology\_schema\_name, srid, tolerance, hasz

**CopyTopology**<sup>1</sup> (existing\_topology\_name, new\_name) Makes a copy of a topology structure (nodes, edges, faces, layers and TopoGeometries).

**ST\_InitTopoGeo**<sup>mmm</sup> (topology\_schema\_name) Creates a new topology schema and registers this new schema in the topology.topology table and details summary of process.

**ST\_CreateTopoGeo**<sup>1 mmm</sup> (atopology, acollection) Adds a collection of geometries to a given empty topology and returns a message detailing success.

**TopoGeo\_AddPoint**<sup>1</sup> (toponame, apoint, tolerance) Adds a point to an existing topology using a tolerance and possibly splitting an existing edge.

**TopoGeo\_AddLineString**<sup>1</sup> (toponame, aline, tolerance) Adds a linestring to an existing topology using a tolerance and possibly splitting existing edges/faces.

**TopoGeo\_AddPolygon**<sup>1</sup> (atopology, aline, atolerance) Adds a polygon to an existing topology using a tolerance and possibly splitting existing edges/faces.

### Topology Editors

**ST\_AddIsoNode**<sup>mmm</sup> (atopology, aface, apoint) Adds an isolated node to a face in a topology and returns the nodeid of the new node. If face is null, the node is still created.

**ST\_AddIsoEdge**<sup>mmm</sup> (atopology, anode, anothernode, alinestring) Adds an isolated edge defined by geometry alinestring to a topology connecting two existing isolated nodes anode and anothernode and returns the edge id of the new edge.

**ST\_AddEdgeNewFaces**<sup>1 mmm</sup> (atopology, anode, anothernode, acurve) Add a new edge and, if in doing so it splits a face, delete the original face and replace it with two new faces.

**ST\_AddEdgeModFace**<sup>1 mmm</sup> (atopology, anode, anothernode, acurve) Add a new edge and, if in doing so it splits a face, modify the original face and add a new face.

**ST\_RemEdgeNewFace**<sup>1 mmm</sup> (atopology, anedge) Removes an edge and, if the removed edge separated two faces, delete the original faces and replace them with a new face.

**ST\_RemEdgeModFace**<sup>1 mmm</sup> (atopology, anedge) Removes an edge and, if the removed edge separated two faces, delete one of the them and modify the other to take the space of both.

**ST\_ChangeEdgeGeom**<sup>2 mmm</sup> (atopology, anedge, acurve) Changes the shape of an edge without affecting the topology structure.

**ST\_ModEdgeSplit**<sup>mmm</sup> (atopology, anedge, apoint) Split an edge by creating a new node along an existing edge, modifying the original edge and adding a new edge.

**ST\_ModEdgeHeal**<sup>1 mmm</sup> (atopology, anedge, anotheredge) Heal two edges by deleting the node connecting them, modifying the first edge and deleting the second edge. Returns the id of the deleted node.

**ST\_NewEdgeHeal**<sup>1 mmm</sup> (atopology, anedge, anotheredge) Heal two edges by deleting the node connecting them, deleting both edges, and replacing them with an edge whose direction is the same as the first edge provided.

**ST\_MoveIsoNode**<sup>mmm</sup> (atopology, anedge, apoint) Moves an isolated node in a topology from one point to another. If new apoint geometry exists as a node an error is thrown. REturns description of move.

**ST\_NewEdgesSplit**<sup>mmm</sup> (atopology, anedge, apoint) Split an edge by creating a new node along an existing edge, deleting the original edge and replacing it with two new edges. Returns the id of the new node created that joins the new edges.

**ST\_RemoveIsoNode**<sup>mmm</sup> (atopology, anode) Removes an isolated node and returns description of action. If the node is not isolated (is start or end of an edge), then an exception is thrown.

### Topology Accessors

**GetEdgeByPoint**<sup>1 g3.3</sup> (atopology, apoint, tol) Find the edge-id of an edge that intersects a given point

**GetFaceByPoint**<sup>1 g3.3</sup> (atopology, apoint, tol) Find the face-id of a face that intersects a given point

**GetNodeByPoint**<sup>1 g3.3</sup> (atopology, point, tol) Find the id of a node at a point location

**GetTopologyID** (toponame) Returns the id of a topology in the topology.topology table given the name of the topology.

**GetTopologySRID**<sup>1</sup> (toponame) Returns the SRID of a topology in the topology.topology table given the name of the topology.

**GetTopologyName** (topology\_id) Returns the name of a topology (schema) given the id of the topology.

**ST\_GetFaceEdges**<sup>1 mmm</sup> (atopology, aface) Returns a set of ordered edges that bound aface includes the sequence order.

**ST\_GetFaceGeometry**<sup>mmm</sup> (atopology, aface) Returns the polygon in the given topology with the specified face id.

**GetRingEdges**<sup>1</sup> (atopology, aring, max\_edges=null) Returns an ordered set of edges forming a ring with the given edge .

**GetNodeEdges**<sup>1</sup> (atopology, anode) Returns an ordered set of edges incident to the given node.

### Topology Processing

**Polygonize**<sup>1</sup> (toponame) Find and register all faces defined by topology edges

**AddNode**<sup>1</sup> (toponame, apoint, allowEdgeSplitting=false, computeContainingFace=false) Adds a point node to the node table in the specified topology schema and returns the nodeid of new node. If point already exists as node, the existing nodeid is returned.

**AddEdge**<sup>1 g3.3</sup> (toponame, aline) Adds a linestring edge to the edge table and associated start and end points to the point nodes table of the specified topology schema using the specified linestring geometry and returns the edgeid of the new (or existing) edge.

**AddFace**<sup>1</sup> (toponame, apolygon, force\_new=false) Registers a face primitive to a topology and get its identifier.

### TopoGeometry Constructors

**CreateTopoGeom** Creates a new topo geometry object from topo element array - tg\_type: 1:[multi]point, 2:[multi]line, 3:[multi]poly, 4:collection

1. toponame, tg\_type, layer\_id, tg\_objs
2. toponame, tg\_type, layer\_id

**toTopoGeom**<sup>1</sup> (geom, toponame, layer\_id, tolerance) Creates a new topo geometry from a simple geometry

**TopoElementArray\_Agg**<sup>1</sup> (tefield) Returns a topelementarray for a set of element\_id, type arrays (topelements)

### TopoGeometry Accessors

**GetTopoGeomElementArray** Returns a topelementarray (an array of topelements) containing the topological elements and type of the given TopoGeometry (primitive elements)

1. toponame, layer\_id, tg\_id
2. tg

**GetTopoGeomElements** Returns a set of topelement objects containing the topological element\_id,element\_type of the given TopoGeometry (primitive elements)

1. toponame, layer\_id, tg\_id
2. tg

### TopoGeometry Outputs

**AsGML**<sup>1</sup> Returns the GML representation of a topogeometry.

1. tg
2. tg, nsprefix\_in
3. tg, visitedTable
4. tg, visitedTable, nsprefix
5. tg, nsprefix\_in, precision, options
6. tg, nsprefix\_in, precision, options, visitedTable
7. tg, nsprefix\_in, precision, options, visitedTable, idprefix
8. tg, nsprefix\_in, precision, options, visitedTable, idprefix, gmlversion



This work is licensed under a Creative Commons Attribution

Feel free to use this material for private or commercial purposes, but we ask that you please retain the <http://www.postgis.us> PostGIS in Action website link.

PostGIS in Action BostonGIS.com Paragon Corporation Postgres OnLine Journal