Management

AddGeometryColumn
DropGeometryColumn
DropGeometryTable
populate_geometry_columns[3]
postgis_full_version
postgis_geos_version
postgis_lib_version
postgis_proj_version
postgis_version
probe_geometry_columns
ST_SetSRID
UpdateGeometrySRID

Load/Dump Tools

--PostGIS tools --

shp2pgsql
shp2pgsql-gui [3]
pgsql2shp

--PostgreSQL --
pg_dump
pg_restore
psql

Meta Tables/Views

spatial_ref_sys
geometry_columns
geography_columns[1]

Geometry Creation

ST_GeomFromEWKB
ST_GeomFromEWKT
ST_GeogFromText[1]
ST_GeomFromGML[1]
ST_GeomFromKML[1]
ST_GeomFromText
ST_GeomFromWKB
ST_GeogFromWKB[1]
ST_MakeEnvelope[1]
ST_MakeLine
ST_MakePolygon
ST_MakePoint

Relationship

ST_Contains*[2]
ST_ContainsProperly*[1,2,G3.1]
ST_CoveredBy[2]
ST_Covers[2]
ST_Crosses*[2]
ST_Disjoint*
ST_DWithin[2,3]
ST_DFullyWithin[1]
ST_Equals*
ST_LineCrossingDirection[1]
ST_Intersects*[2,3]
ST_Overlaps*[2]
ST_Relate*
ST_Touches*[2]
ST_Within*[2]

Spatial Aggregates

ST_Accum
ST_Collect
ST_Extent
ST_Union*
ST_MakeLine
ST_Polygonize*

Geometry Editors

ST_AddMeasure
ST_AddPoint
ST_Affine
ST_Collect
ST_CollectionExtract[1]
ST_Force_collection
ST_Force_2d
ST_Force_3d, ST_Force_3dm
ST_Force_3dz
ST_Force_4d
ST_LineMerge
ST_Multi
ST_RemovePoint
ST_Segmentize
ST_SetPoint
ST_SnapToGrid

Linear Referencing

ST_Line_Interpolate_Point
ST_Line_Substring
ST_Line_Locate_Point
ST_Locate_Along_Measure
ST_Locate_Between_Measures
ST_LocateBetweenElevations[1]

PostGIS ver. 1.5 Quick Guide - Cheatsheet

This list is not comprehensive but tries to cover at least 80%.
*Uses GEOS, Requires Geos 3.2+ to take advantage of new or improved features[G3.2]

Most commonly used functions and operators

Measurement functions return in same units geometry SRID except
for the *sphere and *spheroid versions and new Geography type which return in meters

Denotes a new item in this version [1]
Denotes automatically uses spatial indexes [2]
Denotes enhanced in this version [3]

GEOMETRY/GEOGRAPHY TYPES - WKT REPRESENTATION

```
POINT(0 0)
LINESTRING(0 0,1 1,1 2)
POLYGON((0 0,4 0,4 4,0 4,0 0),(1 1, 2 1, 2 2, 1 2,1 1))
MULTIPOINT(0 0,1 2)
MULTILINESTRING((0 0,1 1,1 2),(2 3,3 2,5 4))
MULTIPOLYGON(((0 0,4 0,4 4,0 4,0 0),(1 1,2 1,2 2,1 2,1 1)), ..)
GEOMETRYCOLLECTION(POINT(2 3),LINESTRING((2 3,3 4)))
```

BBOX AND GEOMETRY OPERATORS

A &< B (A overlaps or is to the left of B) [2]
A &> B (A overlaps or is to the right of B) [2]
A << B (A is strictly to the left of B) [2]
A >> B (A is strictly to the right of B)[2]
A &<| B (A overlaps B or is below B)[2]
A |&> B (A overlaps or is above B)[2]
A <<| B (A strictly below B)[2]
A |>> B (A strictly above B)[2]
A = B (A bbox same as B bbox)
A @ B (A completely contained by B)[2]
A ~ B (A completely contains B)[2]
A && B (A and B bboxes intersect)[2]
A ~= B - true if A and B boxes are equal[2] [3]

COMMON USE SFSQL EXAMPLES

```
 --Create a geometry column named geom in a
--table called testtable located in schema public
-- to hold point geometries of dimension 2 in WGS84 longlat
SELECT AddGeometryColumn('public', 'testtable', 'geom', 4326, 'POINT', 2);

--Insert a record into the new table
 INSERT INTO testtable(description, geom)
VALUES('center of boston',
        ST_GeomFromText('POINT(-71.0891380310059 42.3123226165771)', 4326));

--Insert a point record into the new table - faster than st_geomfromtext for points
 INSERT INTO testtable(description, geom)
VALUES('center of boston',
        ST_SetSRID(ST_MakePoint(-71.0891380310059, 42.3123226165771), 4326));

 --Create a geography column named geog in a
--table called testtable located in schema public
-- to hold point geographies of dimension 2 in WGS84 longlat
CREATE TABLE testtable(test_id serial primary key, description text, geog geography(POINT,4326));

--Insert a record into the new table
 INSERT INTO testtable(description, geog)
VALUES('center of boston',
        ST_GeogFromText('SRID:4326;POINT(-71.0891380310059 42.3123226165771)'));

--Create a spatial index on the new geometry column
 ALTER TABLE testtable ALTER COLUMN geom SET NOT NULL;
 CREATE INDEX idx_testtable_geom ON testtable USING gist(geom);
 ALTER TABLE testtable CLUSTER ON idx_testtable_geom;

--Find the neighborhood with the smallest area
 SELECT neigh_name, ST_Area(geom)
        FROM neighborhoods
        ORDER BY ST_Area(geom) limit 1;

--Find the total area of each ward in square feet of wards in Boston,
--the extent (bounding box) of each ward, average sqft per precinct in each ward
 SELECT ward, sum(ST_Area(ST_Transform(geom,2249))) as totarea,
  avg(ST_Area(ST_Transform(geom,2249))) as avgarea_precinct,
        ST_Extent(ST_Transform(geom,2249)) as wardextent
        FROM wardprecincts WHERE city = 'Boston'
        GROUP BY ward;

--Find all land parcels within 100 units of a specific parcel.
SELECT l2.parcel_id, l2.st_num, l2.st_name
 FROM landparcels l , landparcels l2
  WHERE ST_DWithin(l.geom, l2.geom, 100)
        AND l.parcel_id = '1234560000';

--Break up multipolygons into individual polygons
SELECT neigh_name,
(ST_Dump(geom)).geom  As polygeom
        FROM neighborhoods;

--Take individual polygons and create one multipolygon for each neighborhood
 --Note if you have a mixed collection of geometries, will return a geometry collection
SELECT neigh_name, ST_Collect(polygeom) as geom
        FROM neighborhoods
        GROUP BY neigh_name;
```

USING SHAPE DUMPER/LOADER COMMANDLINE TOOLS

```
Load data into PostgreSQL from ESRI shape file to geometry data type
shp2pgsql -s 4326 neighborhoods public.neighborhoods > neighborhoods.sql
psql -h myserver -d mydb -U myuser -f neighborhoods.sql

Load data into PostgreSQL from ESRI shape file into geography type
shp2pgsql -G -s 4326 neighborhoods public.neighborhoods > neighborhoods.sql
psql -h myserver -d mydb -U myuser -f neighborhoods.sql


Exporting data from PostgreSQL to ESRI Shape file
pgsql2shp -f jpnei -h myserver -u apguser -P apgpassword mygisdb
        "SELECT neigh_name, geom FROM neighborhoods WHERE neigh_name = 'Jamaica Plain'"
```

Boston GIS   Paragon Corporation
Postgres OnLine Journal PostGIS in Action

Accessors

ST_CollectionExtract[1]
ST_Dimension
ST_Dump
ST_DumpPoints[1]
ST_DumpRings
ST_EndPoint
ST_Envelope
ST_ExteriorRing
ST_GeometryN
ST_GeometryType
ST_InteriorRingN
ST_IsClosed
ST_IsEmpty
ST_IsRing
ST_IsSimple
ST_IsValid
ST_IsValidReason
ST_mem_size
ST_M
ST_NumGeometries

ST_NumInteriorRings
ST_NumPoints
ST_npoints
ST_PointN
ST_SetSRID
ST_StartPoint

ST_Summary
ST_X
ST_XMin,ST_XMax
ST_Y
YMin,YMax
ST_Z
ZMin,ZMax

Measurement

ST_Area[3]
ST_Azimuth
ST_Distance
ST_HausdorffDistance[1G3.2]
ST_distance_sphere[3]
ST_distance_spheroid[3]

ST_Length_Spheroid
ST_Length[3]
ST_MaxDistance[1]
ST_Perimeter

Outputs

ST_AsBinary[3]
ST_AsText[3]
ST_AsEWKB
ST_AsEWKT
ST_AsHEXEWKB
ST_GeoJSON
ST_AsGML[3]
ST_AsKML[3]
ST_AsSVG[3]
ST_GeoHash

Geometry Processors

ST_MinimumBoundingCircle
ST_Boundary*
ST_Buffer*[3 G3.2]
ST_BuildArea*
ST_Centroid
ST_ClosestPoint[1]
ST_ConvexHull
ST_Difference*
ST_Expand
ST_ForceRHR
ST_LongestLine[1]
ST_Intersection*
ST_PointOnSurface*
ST_Reverse
ST_RotateX
ST_RotateY
ST_RotateZ
ST_Scale
ST_ShortestLine[1]
ST_Simplify
ST_SimplifyPreserveTopology
ST_SymDifference
ST_Transform
ST_Translate
ST_TransScale
ST_Union