GeoPackage Specifications

Table of Contents

Background

Mobile device users who require map/geospatial application services and operate in disconnected or limited network connected environments are challenged by having open, available geospatial data to support these applications. Further challenging mobile device users are the limited storage available and the likelihood that each map/geospatial application will require their own potentially proprietary geospatial data store. An open, standards-based, self-describing GeoPackage (GPKG) data container, manifest, and API are needed to overcome these challenges and effectively support multiple map/geospatial applications such as fixed product distribution, local data collection, and geospatially enabled analytics.

This GPKG data container will act as an exchange and direct-use format for multiple types of geospatial data. Specifically, a GeoPackage will be capable of holding multiple vector feature types, rasters from various sources, and multiple tile pyramids. An individual GPKG may contain one, some or all of these types of geospatial data. Future GeoPackage Service Specification(s) will include requirements for elevation data, routes, and web services to support provisioning of GeoPackages throughout an enterprise.

The structure of the GeoPackage container and GeoPackage services will be specified by reference to existing international standards and public open source specifications to the greatest practicable extent.

GeoPackage Container

A self-contained, single-file, cross-platform, serverless, open source RDBMS container is desired to simplify production, distribution and use of GeoPackages. Conformance with current ISO/IEC 9075 (SQL) standards would be optimal, but at a minimum the GeoPackage container must support SQL-92, including BLOB data types, stored procedures, and FOR EACH ROW triggers.

Reference Implementation

SQLite will be the initial reference implementation of the GeoPackage container. It has been used as the base for a number of vector, raster and tile storage specifications, and commercial and open-source implementations. It is deployed and supported by Google on Android and Apple on IOS mobile devices. Testing on a laptop indicates that its performance scales well for databases in excess of 200GB containing vector and raster tables of more than 4 million rows. Some evolution in Mobile / Handheld Computing Environment file system capabilities will be necessary to allow such large files.

SQLite does not support updateable views, which must be simulated using FOR EACH ROW triggers.

Vector Feature Store

An RDBMS container store with SQL access for simple features with geometry is desired to manage (create, update, delete as well as search and retrieve) both geospatial foundation data for multiple types of features, and newly collected feature observation data. Initial support is required for the basic simple feature geometry types – Geometry, Point, Curve, LineString, LinearRing, Surface, Polygon, GeometryCollection, MultiPoint, MultiCurve, MultiLineString, MultiSurface and MultiPolygon. Subsequent GPKG specification versions may require support for Polyhedral Surfaces, TINS, and Full 3D. In both cases spatial indexing and SQL spatial routines are required for access, transform, and relational functions and predicates on geometry types to support direct use by geospatial applications.

Fortunately there are applicable international specifications that have standardized practices for the storage, access and use of vector geospatial features and geometries via SQL in relational databases. The original Simple Features for SQL specification OGC 99-049 and its successors OGC 06-103r4 and 06-104r4 (ISO 19125) describe the common architecture for simple feature geometry and define a standard Structured Query Language (SQL) schema that supports storage, retrieval, query and update of feature collections via the SQL Call-Level Interface (SQL/CLI) (ISO/IEC 9075-3:2003).

Spatial Reference Systems

The first component of the standard SQL schema for simple features is a table or updateable view specified in OGC 06-104r4 section 7.1.2.2 containing data that defines spatial reference systems. This table shall contain a record for auth_name EPSG and auth_srid 4326 for WGS-84.

Table or View Name	Column Name	Column Type	Column Description	Key
spatial_ref_sys	srid	integer	Unique identifier for each Spatial Reference System within a database	PK
spatial_ref_sys	auth_name	text	the name of the standard or standards body that is being cited for this reference system, e.g. EPSG	
spatial_ref_sys	auth_srid	integer	the ID of the Spatial Reference System as defined by the Authority cited in AUTH_NAME	
spatial_ref_sys	srtext	text	Well-known Text Representation of the Spatial Reference System as specified in OGC 06-103r4 section 9	

Geometry Columns

The second component of the standard SQL schema is a table or updateable view specified in OGC 06-104r4 section 7.1.3.1 that identifies the geometry columns in tables that contain data representing simple features.

Table or View Name	Column Name	Column Type	Column Description	Key
geometry_columns	f_table_catalog	text	Catalog containing the table containing the geometry column	PK
geometry_columns	f_table_schema	text	Schema containing the table containing the geometry column	PK
geometry_columns	f_table_name	text	Name of the table containing the geometry column	PK
geometry_columns	f_geometry_column	text	Name of the column in the feature table that is the Geometry Column	
geometry_columns	g_table_catalog	text	Catalog containing separate geometry table, or f_table_catalog if no separate geometry table	
geometry_columns	g_table_schema	text	Schema containing separate geometry table, or f_table_schema if no separate geometry table.	
geometry_columns	g_table_name	text	Name of separate geometry table, or f_table_name if no separate geometry table	
geometry_columns	storage_type	integer	Storage type; 0=normalized (separate geometry table), 1=binary "gB", 2=geometry types "gS"	
geometry_columns	geometry_type	integer	Code from OGC 06-104r4 Table 4: 0=geometry, 1=point …	
geometry_columns	coord_dimension	integer	Number of ordinates, +1 if geometry_type includes an "M" Measure dimension.	
geometry_columns	max_ppr	integer	Points per row for storage_type=0, otherwise NULL	
geometry_columns	srid	integer	Spatial Reference System ID: spatial_ref_sys.id	FK

SQLite does not have separate catalogs or schemas, so the f_table_catalog and f_table_schema columns are meaningless in an SQLite GPKG container. As described below, the GPKG container does not use a separate geometry table, so the g_table_catalog, g_table_schema, g_table_name, and max_ppr colomns are also meaningless in a GPKG container. As described below, the initial GPKG container will only support binary "gB" geometry representations, so the storage_type column is also meaningless until GPKG containers can also support "gS" geometry representations. Initial SQLite GPKG implementations may omit these columns with names in blue in the table above. The foreign key (FK) on geometry_columns.srid references the primary key (PK) on spatial_ref_sys.srid to ensure that geometry columns are only defined in feature tables for defined spatial reference systems.

Feature Tables

The third component of the standard SQL schema for simple features includes two different storage architectures for tables that contain data representing simple features and geometries. The first architecture represents geometries using predefined data types in a separate geometry table. The second architecture uses SQL geometry types for geometry columns in feature tables.

There are two variants of the "SQL with geometry types" architecture. The first, called "gB", uses binary types (SQL BLOBs) to contain geometries, and is supported by SQL-92. These BLOBs are required to contain values of the corresponding geometry_columns.geometry_type that include "Well Known Binary (WKB)" format data specified in OGC 06-103r4. The second, called "gS", uses SQL user-defined types as specified by SQL/MM in ISO/IEC 13249-3, which defines spatial user-defined types (for the corresponding geometry_columns.geometry_type) and their associated routines. User defined types were introduced in SQL 3, and so the "gS" geometry types are not supported by SQL-92.

GeoPackages will initially use the "SQL with geometry types" "gB" architecture with BLOB geometry columns in feature tables supported by SQL-92. GPKG "gB" feature tables shall be defined as specified in OGC 99-049 section 2.3.8, including the use of stored procedures to add and remove geometry columns.

When GeoPackage containers that support current ISO/IEC 9075 standards become available, GeoPackages will also use the "SQL with geometry types" "gS" architecture with the user-defined geometry types from ISO/IEC 13249-3. GPKG "gS" feature tables shall be defined as specified in OGC 06-104r4 section 7.2.3.

All geometry types and relational operators implemented by a GeoPackage for either the "gB" or "gS" architecture shall conform to the geometry architecture model specified in section 6.1 of OGC 06-103r4.

GeoPackage implementations of the "gB" and "gS" architectures shall assure that the geometry values stored in a geometry column of a feature table are of the geometry type specified for the column in the geometry_columns.geometry_type value for the geometry column, and are of the srid specified for the geometry column in the geometry_columns.srid for the column by implementation of appropriate SQL constraints and triggers.

GeoPackage implementations of the "gB" and "gS" architectures shall have the capability to create, delete and use a spatial index on any geometry column of a feature table to improve the performance of spatial queries that use the spatial methods listed below to select features based on spatial relationships. These

indexes may be constructed using any applicable geometry routines listed below, e.g. Envelope, and any appropriate indexing mechanisms, such as quad trees or R trees.

Geometry Routines SQL API

In both the "gB" and "gS" architecture implementations, GeoPackages shall provide the geometry-type routines specified by ISO/IEC 13249-3 and the OGC Simple Features for SQL specifications listed in the following tables with names in black. GeoPackages may optionally provide the geometry-type routines with names in blue, which may be required by a future GeoPackage specification version.

These routines may be provided by alias of GeoPackage implementation routines that have the same parameters and produce the specified result but have different, non-standard names. Note: in the following table, for brevity the "ST_" prefix is removed from the names of Geometry types.

Routine Name	**Description**	**13249-3 Clause**	**99-049 Clause**	**06-103r4 Clause**	**06-104r4 Clause**
ST_Dimension	returns the dimension of a Geometry value	4.1.1.1	2.1.1.1	6.1.2.2	7.2.8.1
ST_GeometryType	returns the type of the Geometry value as a String	4.1.1.1	2.1.1.1	6.1.2.2	7.2.8.1
ST_SRID	returns the spatial reference system identifier of an Geometry value	4.1.1.1	2.1.1.1	6.1.2.2	7.2.8.1
ST_Transform	returns the Geometry value in the specified spatial reference system	4.1.1.1			
ST_IsEmpty	tests if a Geometry value corresponds to the empty set	4.1.1.1	2.1.1.1	6.1.2.2	7.2.8.1
ST_IsSimple	tests if a Geometry value has no anomalous geometric point, such as self intersection or self tangency	4.1.1.1	2.1.1.1	6.1.2.2	7.2.8.1
ST_IsValid	tests if a Geometry value is well formed	4.1.1.1			
ST_Is3D	tests whether a Geometry value has z coordinates	4.1.1.1		6.1.2.2	
ST_IsMeasured	tests whether a Geometry value has m coordinate values	4.1.1.1		6.1.2.2	
ST_LocateAlong	returns a derived geometry collection value that matches the specified m coordinate value	4.1.1.1		6.1.2.6	
ST_LocateBetween	returns a derived geometry collection value that matches the specified range of m coordinate values inclusively	4.1.1.1		6.1.2.6	
ST_Boundary	returns the boundary of a Geometry value	4.1.1.1	2.1.1.1	6.1.2.2	7.2.8.1
ST_Envelope	returns the bounding rectangle of a Geometry value	4.1.1.1	2.1.1.1	6.1.2.2	7.2.8.1
ST_ConvexHull	returns the convex hull of a Geometry value	4.1.1.1	2.1.1.3	6.1.2.4	7.2.8.1
ST_Buffer	returns the Geometry value that represents all points whose distance from any point of a Geometry value is less than or equal to a specified distance	4.1.1.1	2.1.1.3	6.1.2.4	7.2.8.1
ST_Intersection	returns the Geometry value that represents the point set intersection of two Geometry values	4.1.1.1	2.1.1.3	6.1.2.4	7.2.8.1
ST_Union	returns the Geometry value that represents the point set union of two ST_Geometry values	4.1.1.1	2.1.1.3	6.1.2.4	7.2.8.1
ST_Difference	returns the Geometry value that represents the point set difference of two Geometry values	4.1.1.1	2.1.1.3	6.1.2.4	7.2.8.1
ST_SymDifference	returns the Geometry value that represents the point set symmetric difference of two Geometry values	4.1.1.1	2.1.1.3	6.1.2.4	7.2.8.1
ST_Distance	returns the distance between two geometries	4.1.1.1	2.1.1.3	6.1.2.4	7.2.8.1
ST_AsText	returns the well-known text representation for the specified Geometry value	4.1.1.1	2.1.1.1	6.1.2.2	7.2.8.1
ST_AsBinary	returns the well-known binary representation for the	4.1.1.1	2.1.1.1	6.1.2.2	

	specified Geometry value				
ST_AsGML	returns the GML representation for the specified Geometry value	4.1.1.1			
ST_GeomFromText	returns a Geometry value from its well-known text representation	4.1.1.2	3.2.6.2		
ST_GeomFromWKB	returns a Geometry value from its well-known binary representation	4.1.1.2	3.2.7.2		
ST_GeomFromGML	returns a Geometry value from its GML representation	4.1.1.2			

Point Routines

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause
ST_X	returns the x coordinate value of a Point value	4.1.3.1	3.2.11.2	6.1.4.2	7.2.9.1
ST_Y	returns the y coordinate value of a Point value	4.1.3.1	3.2.11.2	6.1.4.2	7.2.9.1
ST_Z	returns the z coordinate value of a Point value	4.1.3.1		6.1.4.2	7.2.9.1
ST_M	returns the m coordinate value of a Point value	4.1.3.1		6.1.4.2	7.2.9.1
ST_PointFromText	returns a Point value from the well-known text representation of a Point	4.1.3.2	3.2.6.2		
ST_PointFromWKB	returns a Point value from the well-known binary representation of a Point.	4.1.3.2	3.2.7.2		

Curve Routines

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause
ST_Length	returns the length of a Curve value	4.1.4.1	2.1.5.1	6.1.6.2	7.2.10.1
ST_StartPoint	returns the Point value that is the start point of a Curve value	4.1.4.1	2.1.5.1	6.1.6.2	7.2.10.1
ST_EndPoint	returns the Point value that is the end point of a Curve value	4.1.4.1	2.1.5.1	6.1.6.2	7.2.10.1
ST_IsClosed	tests if a Curve value is closed	4.1.4.1	2.1.5.1	6.1.6.2	7.2.10.1
ST_IsRing	tests if an Curve value is a ring	4.1.4.1	2.1.5.1	6.1.6.2	7.2.10.1

LineString Routines

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause
ST_NumPoints	returns the cardinality of the Point collection in the LineString value	4.1.5.1	2.1.6.1	6.1.7.2	7.2.11.1
ST_PointN	returns the specified element in the Point collection in the LineString value	4.1.5.1	2.1.6.1	6.1.7.2	7.2.11.1
ST_LineFromText	returns a LineString value from the well-known text representation of a LineString	4.1.5.2	3.2.6.2		
ST_LineFromWKB	returns a LineString value from the well-known binary representation of a LineString	4.1.5.2	3.2.7.2		

Surface Routines

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause

ST_Area	returns the area of a Surface value	4.1.8.1	2.1.9.1	6.1.10.2	7.2.12.1
ST_Centroid	returns the Point value that is the mathematical centroid of the Surface value	4.1.8.1	2.1.9.1	6.1.10.2	7.2.12.1
ST_PointOnSurface	returns a Point value that is guaranteed to be on the Surface value	4.1.8.1	2.1.9.1	6.1.10.2	7.2.12.1

Polygon Routines

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause
ST_ExteriorRing	returns the exterior ring of a Polygon value	4.1.9.1	2.1.10.1	6.1.11.2	7.2.13.1
ST_NumInteriorRing	returns the cardinality of the collection of interior rings of a Polygon value	4.1.9.1	2.1.10.1	6.1.11.2	7.2.13.1
ST_InteriorRingN	returns the specified element in the collection of interior rings of a Polygon value	4.1.9.1	2.1.10.1	6.1.11.2	7.2.13.1
ST_PolyFromText	returns a Polygon value from the well-known text representation of a Polygon	4.1.10.2	3.2.6.2		
ST_PolyFromWKB	returns a Polygon value from the well-known binary representation of a Polygon	4.1.10.2	3.2.7.2		
ST_BdPolyFromText	returns a Polygon value from a well-known text representation of a MultiLineString	4.1.10.2	3.2.6.2		
ST_BdPolyFromWKB	returns a Polygon value from a well-known binary representation of a MultiLineString	4.1.10.2	3.2.7.2		

MultiGeometry Routines

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause
ST_NumGeometries	returns the cardinality of the Geometry collection	4.1.11.1	3.2.16.2	6.1.3.2	7.2.15
ST_GeometryN	returns the specified element in the Geometry collection	4.1.11.1	3.2.16.2	6.1.3.2	7.2.15
ST_GeomCollFromTxt	returns a GeomCollection value from the well-known text representation of a GeomCollection	4.1.11.2	3.2.6.2		
ST_GeomCollFromWKB	returns a GeomCollection value from the well-known binary representation of GeomCollection	4.1.11.2	3.2.7.2		

MultiPoint Routines

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause
ST_MPointFromText	returns a MultiPoint value from the well-known text representation of a MultiPoint	4.1.12.2	3.2.6.2		
ST_MPointFromWKB	returns a MultiPoint value from the well-known binary representation of a MultiPoint	4.1.12.2	3.2.7.2		

MultiCurve Routines

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause
ST_IsClosed	tests if a MultiCurve value is closed	4.1.13.1	3.2.17.2	6.1.8.2	7.2.17.1
ST_Length	returns the length of a MultiCurve value	4.1.13.1	3.2.17.2	6.1.8.2	7.2.17.1

MultiLineString Routines

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause
ST_MLineFromText	returns a MultiLineString value from the well-known text representation of a MultiLineString	4.1.14.2	3.2.6.2		
ST_MLineFromWKB	returns a MultiLineString value from the well-known binary representation of a MultiLineString	4.1.14.2	3.2.7.2		

MultiSurface Routines

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause
ST_Area	returns the area of a MultiSurface value	4.1.15.1	3.2.18.2	6.1.13.2	7.2.19.1
ST_Centroid	returns the Point value that is the mathematical centroid of the MultiSurface value	4.1.15.1	3.2.18.2	6.1.13.2	7.2.19.1
ST_PointOnSurface	returns a Point value that is guaranteed to be on the MultiSurface value	4.1.15.1	3.2.18.2	6.1.13.2	7.2.19.1

MultiPolygon Routines

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause
ST_MPolyFromText	returns a MultiPolygon value from the well-known text representation of a MultiPolygon	4.1.16.2	3.2.6.2		
ST_MPolyFromWKB	returns a MultiPolygon value from the well-known binary representation of a MultiPolygon	4.1.16.2	3.2.7.2		
ST_BdMPolyFromText	returns a MultiPolygon value from a well-known text representation of a MultiLineString	4.1.16.2	3.2.6.2		
ST_BdMPolyFromWKB	returns a MultiPolygon value from a well-known binary representation of a MultiLineString	4.1.16.2	3.2.7.2		

Spatial Predicates

Routine Name	Description	13249-3 Clause	99-049 Clause	06-103r4 Clause	06-104r4 Clause
ST_Equals	tests if a Geometry value is spatially equal to another Geometry value.	4.1.2.3	3.2.19.2	6.1.15.3	7.2.8.1
ST_Relate	tests if a Geometry value is spatially related to another Geometry value by testing for intersections between the interior, boundary and exterior of the two Geometry values as specified by the intersection matrix.	4.1.2.3	3.2.19.2	6.1.15.3	7.2.8.1
ST_Disjoint	tests if a Geometry value is spatially disjoint from another Geometry value	4.1.2.3	3.2.19.2	6.1.15.3	7.2.8.1
ST_Intersects	tests if a Geometry value spatially intersects another Geometry value	4.1.2.3	3.2.19.2	6.1.15.3	7.2.8.1
ST_Touches	tests if a Geometry value spatially touches another Geometry value	4.1.2.3	3.2.19.2	6.1.15.3	7.2.8.1
ST_Crosses	tests if a Geometry value spatially crosses another Geometry value	4.1.2.3	3.2.19.2	6.1.15.3	7.2.8.1

ST_Within	tests if a Geometry value is spatially within another Geometry value	4.1.2.3	3.2.19.2	6.1.15.3	7.2.8.1
ST_Contains	tests if a Geometry value spatially contains another Geometry value	4.1.2.3	3.2.19.2	6.1.15.3	7.2.8.1
ST_Overlaps	tests if a Geometry value spatially overlaps another Geometry value	4.1.2.3	3.2.19.2	6.1.15.3	7.2.8.1

Reference Implementation

SpatiaLite with additional geometry routines and spatial predicates provided by the Geometry Engine Open Source (GEOS) library will be used as the initial reference implementation of the GeoPackage Vector Feature Store. SpatiaLite is currently the only known vector feature store based on SQLite that meets the specifications documented above. It is a supported format for the OGR Simple Features Library. GEOS is widely used by both free and commercial software packages. Quantum GIS (QGIS) is an example of an open source GIS that can manage data in SpatiaLite and other spatial databases using GEOS. QGIS has been ported to Android-based tablets. Luciad Lightspeed is a situational awareness application that uses SpatiaLite for vector feature storage. Use of SpatiaLite for point feature storage is planned for FalconView version 5.1.

Raster / Tile Store

There are a wide variety of commercial and open source conventions for storing, indexing and accessing individual rasters and tiles in tile matrix pyramids. Unfortunately, no applicable existing consensus, national or international specifications have standardized practices in this domain. In addition, various image file formats have different representational capabilities, and include different self-descriptive metadata.

The Raster / Tile Store data model described below attempts to include and expose enough metadata information at both the dataset and record levels to allow direct use of the rasters and tiles in a GeoPackage by existing applications that follow different interface conventions. Following a convention used by MB-Tiles, the Raster / Tile Store data model may be implemented directly as SQL tables in an SQLite database for maximum performance, or as SQL views on top of tables in an existing SQLite Raster / Tile store for maximum adaptability and loose coupling to enable widespread implementation. Following a convention used by RasterLite, tables or views containing rasters, tiles, and record-level metadata are named with a raster layer name prefix, e.g. {RasterLayerName}{_rasters | _tiles | _rt_metadata}, to enable storage of multiple raster and tile pyramid data sets in the same container. These tables or views are described and discussed individually in the following subsections.

Raster Table Metadata

The raster_table_metadata table or view shall contain one record describing each raster or tile table. Note that this data structure can be implemented as a table in absence of geometry data types or spatial indexes. When implemented as a view, the min/max x/y columns could reference ordinates of a bounding box geometry in an underlying table when geometry data types are available. The version column facilitates change-only update of rasters and tiles. i.e. replacement of only those rasters or tiles for which newer versions are available, instead of replacing all rasters or tiles in a table.

Table or View Name	Column Name	Column Type	Column Description	Key
raster_table_metadata	r_table_name	text	{RasterLayerName}{_rasters | _tiles}	PK
raster_table_metadata	srid	integer	spatial reference system id from spatial_ref_sys table (-1 if not georeferenced)	FK
raster_table_metadata	min_x	double	Bounding box for all rasters in r_table_name	
raster_table_metadata	min_y	double	Bounding box for all rasters in r_table_name	
raster_table_metadata	max_x	double	Bounding box for all rasters in r_table_name	
raster_table_metadata	max_y	double	Bounding box for all rasters in r_table_name	
raster_table_metadata	name	text	Full name of the contents of r_table_name . (RasterLayerName may be abbreviated)	
raster_table_metadata	description	text	Text description of the contents of r_table_name	
raster_table_metadata	type	text	"baselayer" or "overlay" (map view may have only one baselayer, 0-n overlays)	
raster_table_metadata	version	text	Version of the contents of r_table_name	

Raster Table Metadata Sample SQL

```
CREATE TABLE raster_table_metadata (
r_table_name TEXT NOT NULL PRIMARY KEY,
srid INTEGER NOT NULL,
min_x DOUBLE NOT NULL,
min_y DOUBLE NOT NULL,
max_x DOUBLE NOT NULL,
max_y DOUBLE NOT NULL,
name TEXT NOT NULL,
description TEXT NOT NULL,
type TEXT NOT NULL,
version TEXT NOT NULL,
CONSTRAINT fk_rtm_srs FOREIGN KEY (srid) REFERENCES spatial_ref_sys (srid))
```

```
INSERT INTO raster_table_metadata VALUES (
"sample_matrix_tiles",
4326,
-179.0,
-89.0,
 179.0,
 89.0,
"sample_matrix_tiles",
"sample matrix tiles table for GeoPackage",
"baselayer",
"1.0.0")
```

Raster Format Metadata

The raster_format_metadata table or view shall contain one record describing each raster or tile image MIME type for each raster or tile table.

Note that images of multiple MIME types may be stored in given table. For example, in a tiles table, image/png format tiles without compression could be used for transparency where there is no data on the tile edges, and image/jpeg format tiles with compression could be used for storage efficiency where there is image data for all pixels.

Table or View Name	Column Name	Column Type	Column Description	Key
raster_format_metadata	r_table_name	text	{RasterLayerName}{_rasters | _tiles}	PK, FK
raster_format_metadata	mime_type	text	Raster or tile image MIME type (image/jpeg | image/png | image/tiff | image/x-webp)	PK
raster_format_metadata	compression_type	text	lossless | lossy | none	
raster_format_metadata	compr_qual_factor	integer	Compression quality factor: 1 (lowest) to 100 (highest) for compression_type lossy; always 100 for types lossless or none.	
raster_format_metadata	num_bands	integer	Number of bands	
raster_format_metadata	image_mode	text	Image mode, how the image is stored: B | P | R | S	
raster_format_metadata	image_rep	text	Image representation: MONO | RGB | RGB/LUT | MULTI | NODISPLY | NVECTOR | VPH | YCbCr601	
raster_format_metadata	image_category	text	VIS | SL | TI | FL | RD | EO | OP | HR | HS | CP | BP | SAR | SARIQ | IR | MS | FP | MRI | XRAY | CAT | VD | BARO | CURRENT | DEPTH | WIND | MAP | PAT | LEG | DTEM | MATR | LOCG	

The following definitions are taken from the National Imagery Transmission Format (MIL-STD-2500C). Image modes are defined as follows:

- B = Band Interleaved by block
- P = Band Interleaved by pixel
- R = Band Interleaved by row
- S = Band Sequential
- N = Not Applicable (for PNG, JPEG, WebP: not in standard)

Image representations are defined as follows:

- MONO = Monochrome
- RGB = Red, Green, Blue true color
- RGB/LUT = Red, Green, Blue mapped color to a Look Up Table
- MULTI = Multi-band imagery
- NODISPLY = Image not intended for display
- NVECTOR = Vectors with Cartesian coordinates
- POLAR = Vectors with polar coordinates
- VPH = SAR video phase history
- YcbCr601 = compressed in the ITU-R Recommendation BT.601-5 color space using JPEG

Image Categories are defined as follows:

- VIS = Visible imagery
- SL = Side-looking radar
- TI = Thermal infrared
- FL = Forward looking infrared
- RD = Radar
- EO = Electro-optical
- OP = Optical

- HR = High-resolution radar
- HS = Hyperspectral
- CP = Color frame photography
- BP = Black and white frame photography
- SAR = Syntheticf aperture radar
- SARIQ = SAR radio hologram
- IR = Infrared
- MS = Multispectral
- FP = Fingerprints
- MRI = Magnetic resonance imagery
- XRAY = X-rays
- CAT = Computer aided tomography scans
- VD = Video
- BARO = Barometric pressure
- CURRENT = Water current
- DEPTH = Water depth
- WIND = Air wind chart
- MAP = Raster map
- PAT = Color patch
- LEG = Legend
- DTEM = Elevation model
- MATR = Other types of matrix data
- LOCG = Location Grid
-

Sample Raster Format Metadata SQL

```
CREATE TABLE raster_format_metadata (
r_table_name TEXT NOT NULL,
mime_type TEXT NOT NULL,
compression_type TEXT NOT NULL,
compr_qual_factor INTEGER NOT NULL,
num_bands INTEGER NOT NULL,
image_mode TEXT NOT NULL,
image_rep TEXT NOT NULL,
image_category TEXT NOT NULL,
CONSTRAINT pk_rfm PRIMARY KEY (r_table_name, mime_type) ON CONFLICT ROLLBACK,
CONSTRAINT fk_rfm_r_table_name FOREIGN KEY (r_table_name) REFERENCES
raster_table_metadata(r_table_name))
```

```
INSERT INTO raster_format_metadata VALUES (
"sample_matrix_tiles",
"image/jpeg",
"lossy",
75,
3,
"N",
```



```
"YCbCr601",
"VIS")
```

Raster Band Metadata

The raster_band_metadata table or view may contain one record describing each band of the rasters or tiles in each raster or tile table. These records are intended to support discovery and interpretation of data that requires specialized processing or analysis before it can be rendered or visualized.

Table or View Name	Column Name	Column Type	Column Description	Key
raster_band_metadata	r_table_name	text	{RasterLayerName}{_rasters | _tiles}	PK, FK
raster_band_metadata	mime_type	text	Raster image MIME type (image/jpeg | image/png | image/tiff | image/x-webp)	PK, FK
raster_band_metadata	band_number	integer	1 to raster_format_metadata.num_bands	PK
raster_band_metadata	band_name	text	Name of band (e.g. elevation, time, alpha, nodata)	
raster_band_metadata	width	integer	In pixels	
raster_band_metadata	height	integer	In pixels	
raster_band_metadata	number_bits_per_pixel	integer	1 | 8 | 12 | 16 | 32 | 64	
raster_band_metadata	actual_bits_per_pixel	integer	1 | 2-8 | 8-12 | 9-16 | 17-32 | 33-64	
raster_band_metadata	pixel_value_type	text	INT | B | SI | R	
raster_band_metadata	no_data_value	text	NoData pixel value	
raster_band_metadata	image_subcategory	text	Wavelength | I | Q | M | P | " " | SPEED | DIRECT | CGX | CGY | GGX | GGY | FACC code | Length Unit	
raster_band_metadata	color_interpretation	text	none | unknown | monochrome | grayscale | paletteIndex | red | green | blue | alpha | hue | saturation | brightness ChrominanceBlue | ChrominanceRed | cyan | magenta | yellow | black|	

The following definitions are taken from the National Imagery Transmission Format (MIL-STD-2500C). Pixel value types are defined as follows:

- INT = Integer (unsigned)
- B = Bi-level (single bit 0 or 1)
- SI = 2's complement signed integer
- R = Real (IEEE 32 or 64-bit floating point representation (IEEE 754))
- C = Complex (Real and Imaginary parts each IEEE 32 or 64-bit floating point representation (IEEE 754)

Image sub-categories are defined as follows:

Image Category	Image Subcategory	Description
MS, HS, IR	wavelength in nanometers	wavelength in nanometers
SAR, SARIQ	I	Inphase band
SAR, SARIQ	Q	Quadrature components band
SAR, SARIQ	M	Magnitude band
SAR, SARIQ	P	Phase components
SAR, SARIQ	Spaces	All other cases – not applicable
WIND, CURRENT	SPEED	Wind or water speed

WIND, CURRENT	DIRECT	Wind or water direction
LOCG	CGX	Cartographic X (Easting)
LOCG	CGY	Cartographic Y (Northing)
LOCG	GGX	Geographic X (Longitude)
LOCG	GGY	Geography Y (Latitude)
MATR	FACC codes from DIGEST part 4 Annex B	Standard values for digital terrain elevation models
DTEM	Units of Length from DIGEST Parts 3-7	
All Others	Spaces	All other cases – not applicable

Sample Raster Band Metadata SQL

```
CREATE TABLE raster_band_metadata (
r_table_name TEXT NOT NULL,
mime_type TEXT NOT NULL,
band_number INTEGER NOT NULL,
band_name TEXT NOT NULL,
width INTEGER NOT NULL,
height INTEGER NOT NULL,
number_bits_per_pixel INTEGER NOT NULL,
actual_bits_per_pixel INTEGER NOT NULL,
pixel_value_type TEXT NOT NULL,
no_data_value TEXT NOT NULL,
image_subcategory TEXT NOT NULL,
color_interpretation TEXT NOT NULL,
CONSTRAINT pk_rbm PRIMARY KEY (r_table_name, mime_type, band_number) ON
CONFLICT ROLLBACK,
CONSTRAINT fk_rbm_table_mime_type FOREIGN KEY (r_table_name, mime_type)
REFERENCES raster_format_metadata(r_table_name, mime_type))
```

```
INSERT INTO raster_band_metadata VALUES (
"sample_matrix_tiles",
"image/jpeg",
1,
"brightness",
512,
512,
8,
8,
"INT",
"0",
" ",
"brightness")
```

Tiles Table Metadata

The tiles_table_metadata table or view shall contain one record describing each tile table. It documents the origin and zoom level conventions followed for tiles in the tile table. Most tile pyramids have an origin at the upper left, but some such as MB-Tiles have an an origin at the lower left. Most tile pyramids, such as Open Street Map, OSMDroidAtlas, and FalconView use a zoom_out_level of 0 for the smallest map scale "whole world" zoom level view, but some such as Big Planet Tracks invert this convention and use 0 or 1 for the largest map scale "local detail" zoom level view. The

tiles_table_metadata table or view supports storage, indexing and use of tile matrix pyramids that follow different origin and zoom level conventions, and allows application software to readily discover the conventions in use.

Table or View Name	Column Name	Column Type	Column Description	Key
tiles_table_metadata	t_table_name	text	{RasterLayerName}_tiles	PK, FK
tiles_table_metadata	origin_y	double	raster_table_metadata. max_y for upper left or raster_table_metadata. min_y for lower left (origin_x = raster_table_metadata. min_x)	
tiles_table_metadata	zoom_out_level	integer	Overview, small scale zoom level for a single tile. Zero for most common zoom level convention; usually > 12 if inverted zoom level convention.	
tiles_table_metadata	zoom_min_level	integer	Minimum (overview, small scale) zoom level of rasters currently in table. May be greater than zoom_max_level for inverted zoom level convention.	
tiles_table_metadata	zoom_max_level	integer	Maximum (detail, large scale) zoom level of rasters currently in table. May be less than zoom_min_level for inverted zoom level convention.	
tiles_table_metadata	scale_set_name	text	Name of a WMTS (OGC 07-057r7 Annex E) or other well known scale set (e.g. TMS) defined in tile_matrix_metadata for t_table_name	

Sample Tiles Table Metadata SQL

```
CREATE TABLE tiles_table_metadata (
t_table_name TEXT NOT NULL PRIMARY KEY,
origin_y DOUBLE NOT NULL,
zoom_out_level INTEGER NOT NULL,
zoom_min_level TEXT NOT NULL,
zoom_max_level INTEGER NOT NULL,
scale_set_name TEXT NOT NULL,
CONSTRAINT fk_ttm_t_table_name FOREIGN KEY (t_table_name) REFERENCES
raster_table_metadata(r_table_name))
```

```
INSERT INTO tiles_table_metadata VALUES (
"sample_matrix_tiles",
89.0,
0,
0,
17,
"urn:ogc:def:wkss:OGC:1.0:GlobalCRS84Pixel")
```

Tile Matrix Metadata

The tile_matrix_metadata table or view shall contain one record for each zoom level in each tiles table. It documents the structure of the tile matrix at each zoom level in each tiles table, and allows GeoPackages to contain rectangular as well as square tiles (e.g. for better representation of polar regions), and tile pyramids with zoom levels that differ in resolution by irregular intervals or regular intervals other than powers of 2, as well as those.

Table or View Name	Column Name	Column Type	Column Description	Key
tile_matrix_metadata	t_table_name	text	{RasterLayerName}_tiles	PK, FK
tile_matrix_metadata	zoom_level	integer	tiles_table_metadata.zoom_min_level <= zoom_level <= tiles_table_metadata.zoom_max_level for t_table_name	PK
tile_matrix_metadata	matrix_width	integer	Number of columns in tile matrix at this zoom level	
tile_matrix_metadata	matrix_height	integer	Number of rows in tile matrix at this zoom level	
tile_matrix_metadata	tile_width	integer	Tile width in pixels for this zoom level	
tile_matrix_metadata	tile_height	integer	Tile height in pixels for this zoom level	
tile_matrix_metadata	pixel_x_size	double	In srid units default meters	
tile_matrix_metadata	pixel_y_size	double	In srid units default meters	

Sample Tile Matrix Metadata SQL

```
CREATE TABLE tile_matrix_metadata (
t_table_name TEXT NOT NULL,
zoom_level INTEGER NOT NULL,
matrix_width INTEGER NOT NULL,
matrix_height INTEGER NOT NULL,
tile_width INTEGER NOT NULL,
tile_height INTEGER NOT NULL,
pixel_x_size DOUBLE NOT NULL,
pixel_y_size DOUBLE NOT NULL,
CONSTRAINT pk_ttm PRIMARY KEY (t_table_name, zoom_level) ON CONFLICT
ROLLBACK,
CONSTRAINT fk_ttm_t_table_name FOREIGN KEY (t_table_name) REFERENCES
raster_table_metadata(r_table_name))
```

```
INSERT INTO tile_matrix_metadata VALUES (
"sample_matrix_tiles",
0,
1,
1,
512,
512,
2.0,
2.0)
```

Tiles Table

Each {RasterLayerName}_tiles table or view contains tile matrices at one or more zoom levels of different spatial resolution (map scale). The id primary key allows tiles table views to be created on RasterLite version 1 raster table implementations, where the tiles are selected based on a spatially indexed bounding box in a separate metadata table. The zoom_level / tile_column / tile_row unique key allows tiles to be selected and accessed by "z, x, y", a common convention used by MB-Tiles, Big Planet TA, and other implementations. This table / view definition may also follow RasterLite version 1 conventions, where the tiles are selected based on a spatially indexed bounding box in a separate metadata table.

GeoPackage implementations shall ensure that only rasters of types stored in the raster_format_metadata table are stored in this table by implementation of appropriate SQL constraints and triggers.

Table or View Name	Column Name	Column Type	Column Description	Key
{RasterLayerName}_tiles	id	integer	Autoincrement primary key	PK
{RasterLayerName}_tiles	zoom_level	integer	tiles_table_metadata.zoom_min_level <= zoom_level <= tiles_table_metadata.zoom_max_level for t_table_name	UK
{RasterLayerName}_tiles	tile_column	integer	0 to tile_matrix_metadata.matrix_width – 1	UK
{RasterLayerName}_tiles	tile_row	integer	0 to tile_matrix_metadata.matrix_height - 1	UK
{RasterLayerName}_tiles	tile_data	BLOB	Of type raster_table_metadata.format	

Sample Tiles Table SQL

```
CREATE TABLE sample_matrix_tiles (
id INTEGER PRIMARY KEY AUTOINCREMENT,
zoom_level INTEGER NOT NULL,
tile_column INTEGER NOT NULL,
tile_row INTEGER NOT NULL,
tile_data BLOB NOT NULL)
```

```
INSERT INTO sample_matrix_tiles VALUES (
1,
1,
1,
1,
"BLOB VALUE")
```

Rasters Table

Each {RasterLayerName}_rasters table or view contains rasters that are not part of tile matrices. GeoPackage implementations shall ensure that only rasters of types stored in the raster_format_metadata table are stored in this table by implementation of appropriate SQL constraints and triggers.

This table / view definition follows RasterLite version 1 conventions, where the rasters are selected based on a spatially indexed bounding box in a separate metadata table. The design objective of having a separate metadata table is to reduce the I/O required to search through the metadata columns. Much more I/O is required to identify the desired tiles when the metadata columns are part of the table that includes the raster BLOBs.

Table or View Name	Column Name	Column Type	Column Description	Key
{RasterLayerName}_rasters	id	integer	Autoincrement primary key	PK
{RasterLayerName}_rasters	raster	BLOB	Of type raster_format_metadata.mime_type	

Sample Rasters Table SQL

```
CREATE TABLE sample_rasters (
id INTEGER PRIMARY KEY AUTOINCREMENT,
raster BLOB NOT NULL)
```

```
INSERT INTO SAMPLE_RASTERS VALUES (
1,
"RASTER BLOB")
```

Rasters or Tiles Table Metadata

Each {RasterLayerName}_rt_metadata table or view contains bounding box ordinates, a version, and a timestamp for a raster or tile. The version and timestamp facilitate change only update of stored rasters and tiles. Note that this data structure can be implemented as a table in absence of geometry data types or spatial indexes. When implemented as a view, the min/max x/y columns could reference ordinates of a bounding box geometry in an underlying table when geometry data types are available.

Table or View Name	Column Name	Column Type	Column Description	Key
{RasterLayerName}_rt_metadata	id	integer	{RasterLayerName}{_rasters | _tiles} id	PK, FK
{RasterLayerName}_rt_metadata	min_x	double	In raster_table_metadata.srid	
{RasterLayerName}_rt_metadata	min_y	double	In raster_table_metadata.srid	
{RasterLayerName}_rt_metadata	max_x	double	In raster_table_metadata.srid	
{RasterLayerName}_rt_metadata	max_y	double	In raster_table_metadata.srid	
{RasterLayerName}_rt_metadata	version	text	Version of raster or tile	
{RasterLayerName}_rt_metadata	image_timestamp	DateTime	Timestamp of raster or tile	

Sample Rasters or Tiles Table Metadata SQL

```
CREATE TABLE sample_matrix_tiles_rt_metadata (
id INTEGER NOT NULL,
min_x DOUBLE NOT NULL,
min_y DOUBLE NOT NULL,
max_x DOUBLE NOT NULL,
max_y DOUBLE NOT NULL,
version TEXT NOT NULL,
image_timestamp TIMESTAMP NOT NULL,
CONSTRAINT fk_smtrm_id FOREIGN KEY (id) REFERENCES sample_matrix_tiles(id))
```

```
INSERT INTO sample_matrix_tiles_rt_metadata VALUES (
1,
-77.0,
38.0,
-75.0,
40.0,
"1.0.0",
"2012-04-26 14:30:00")
```

```
CREATE VIEW TrueMarble_rt_metadata AS
SELECT id,
MbrMinX(geometry),
MbrMinY(geometry),
MbrMaxX(geometry),
MbrMaxY(geometry),
"1.0.0",
"2010-06-15 11:30:00"
FROM TrueMarble_metadata
```

```
SELECT * FROM TrueMarble_rt_metadata WHERE id = 1

1 -180.000000 88.952083 -178.952083 90.000000   1.0.0 2010-06-15 11:30:00
```

Image Routines SQL API

GeoPakages shall provide the following image routine support for rasters and tiles.

Routine Name	Description	Input Param	Input Params	Input Params	Input Params	Input Param	Returns
get_zoom	Returns correct zoom level for specified table, converting between normal convention (0 = whole world) and inverted convention (0 = most detailed) as necessary.	tile table name	input zoom convention: "normal" or "inverted"	input zoom level			Integer zoom level for specified table
get_row	Returns correct row for specified table, converting between origin conventions (upper left or lower left) as necessary	tile table name	origin: "upperLeft" or "lowerLeft"	Input row number			Integer row number for specified table
isJpegBlob	Returns true if the BLOB contains a JPEG image	image BLOB					Integer 1 =true, 0=false
isPngBlob	Returns true if the BLOB contains a PNG image	image BLOB					Integer 1 =true, 0=false
isTiffBlob	Returns true if the BLOB contains a TIFF image	image BLOB					Integer 1 =true, 0=false
isWebpBlob	Returns true if the BLOB contains a WebP image	image BLOB					Integer 1 =true, 0=false
bbox-to-tiles	Returns tiles from tile matrix for the specified srid bounding box and zoom level	tile table name	srid	min-x, max-x	min-y, max-y	zoom	result Set of id, image BLOB
geo-to-tile	Returns tile from tile matrix for specified srid, point and zoom level	tile table name	srid	x	y	zoom	image BLOB
geo-to-pixel	Returns pixel from image for specified srid and point	raster or tile table name	id, srid	x	y		8 byte pixel
pixel-to-geo	Returns ground point for specified srid and pixel	raster or tile table name	id, srid	pixel-row	pixel-col	output format	WKT for point, WKB for point, or Point BLOB

Manifest

The GeoPackage manifest serves as a table of contents and metadata store for the GeoPackage data container. It enables a GeoPackage client application to avoid having to read through all of the contents of a database catalog to find the feature, raster, and tile tables contained in a database. The manifest metadata provides descriptive information to enable a GeoPackage client application to present a descriptive menu of available geospatial data to users of the application. It also provides the query parameters used to obtain the geospatial data contents in the feature, raster, and tile tables in a GeoPackage from an online service.

A GeoPackage manifest is represented by a single XML document in a manifest table with one row and one column.

Table or View Name	Column Name	Column Type	Column Description	Key
manifest	manifest	text	XML manifest document	

The GeoPackage manifest is defined in a geoPackageManifest.xsd XML schema document shown below as an extension of the OGC Manifest defined in http://schemas.opengis.net/ows/2.0/owsManifest.xsd by OGC 06-121r9 OGC Web Services Common Standard Version: 2.0.0. The OGC Manifest contains a Reference Group which in turn contains Reference elements, each of which describes a geospatial resource in a container. These data structures are described in section 13.3, figure 19, and tables 50-53 of OGC 06-121r9. The GeoPackage manifest extends the ows:ReferenceType by adding a gpkg:Table element, and by defining elements to contain ISO and DublinCore metadata. Additional metadata elements may be added to future versions of this specification to contain metadata conforming to other widely used metadata standard models.

Manifest XML Schema

```
<?xml version="1.0" encoding="UTF-8"?>
<schema
   xmlns:gpkg="http://www.opengis.net/gpkg/1.0"
   xmlns:ows="http://www.opengis.net/ows/2.0"
   xmlns:xlink="http://www.w3.org/1999/xlink"
   xmlns:gmd="http://www.isotc211.org/2005/gmd"
   xmlns:dct="http://purl.org/dc/terms/"
   xmlns="http://www.w3.org/2001/XMLSchema"
   targetNamespace="http://www.opengis.net/gpkg/1.0"
   elementFormDefault="qualified" version="2.0.1" xml:lang="en">
        <annotation>
                <documentation>Manifest schema for OGC GeoPackage</documentation>
        </annotation>
        <!-- ==========================================================
         includes and imports
    ========================================================== -->
        <import namespace="http://www.w3.org/1999/xlink" schemaLocation="http://schemas.opengis.net/xlink/1.0.0/xlinks.xsd"/>
        <import namespace="http://www.opengis.net/ows/2.0" schemaLocation="http://schemas.opengis.net/ows/2.0/owsManifest.xsd"/>
        <import namespace="http://www.isotc211.org/2005/gmd"
                schemaLocation="http://schemas.opengis.net/iso/19139/20070417/gmd/gmd.xsd"/>
        <import namespace="http://purl.org/dc/terms/" schemaLocation="http://schemas.opengis.net/csw/2.0.2/rec-dcterms.xsd"/>
        <!-- ======================================================
```

```
    Types and elements
  ========================================================== -->
  <simpleType name="TableTypeNameType">
        <annotation>
              <documentation>Names of the types GeoPackage tables that contain geospatial data content.</documentation>
        </annotation>
        <restriction base="string">
              <enumeration value="features"/>
              <enumeration value="rasters"/>
              <enumeration value="tiles"/>
        </restriction>
  </simpleType>
  <!-- ========================================================== -->
  <complexType name="PostQueryType">
        <annotation>
              <documentation>XML document fragment containing query parameters used to obtain geospatial data content
                    via an XML POST or SOAP POST HTTP request.</documentation>
        </annotation>
        <sequence>
              <any processContents="lax"/>
        </sequence>
  </complexType>
  <!-- ========================================================== -->
  <simpleType name="VersionType">
        <restriction base="string">
              <!-- <pattern value="[1-9][0-9]*\.[0-9]+\.[0-9]+"/> -->
        </restriction>
  </simpleType>
  <!-- ========================================================== -->
  <complexType name="SourceType">
        <annotation>
              <documentation>Source for a GeoPackage table or row from some external Web service.  If the postQuery
                    element is present, the xlink:href attribute shall contain a connection URL for the appropriate
                    corresponding XML or SOAP POST service.  If the postQuery is not present, the xlink:href
                    attribute shall contain a complete KVP GET or RESTful query URL.</documentation>
        </annotation>
        <complexContent>
              <extension base="ows:AbstractReferenceBaseType">
                    <sequence>
                          <element name="identification" type="ows:IdentificationType" minOccurs="0"/>
                          <element name="postQuery" type="gpkg:PostQueryType" minOccurs="0"/>
                    </sequence>
              </extension>
        </complexContent>
  </complexType>
  <!-- ========================================================== -->
  <complexType name="RowSourceType">
        <annotation>
              <documentation>Source for a GeoPackage row.</documentation>
        </annotation>
        <complexContent>
              <extension base="gpkg:SourceType">
                    <attribute name="primaryKey" type="string" use="required"/>
              </extension>
        </complexContent>
  </complexType>
```

```
	<!-- ================================================================== -->
	<element name="Table" type="gpkg:TableType"/>
	<!-- ================================================================== -->
	<complexType name="TableType">
		<annotation>
			<documentation>Identification and description of an RDBMS table containing geospatial data
				in a GeoPackage.</documentation>
		</annotation>
		<sequence>
			<element name="name" type="string"/>
			<element name="type" type="gpkg:TableTypeNameType"/>
			<element name="version" type="gpkg:VersionType"/>
			<choice minOccurs="0">
				<element name="TableSource" type="gpkg:SourceType"/>
				<element name="RowSource" type="gpkg:RowSourceType" maxOccurs="unbounded"/>
			</choice>
		</sequence>
	</complexType>
	<!-- ================================================================== -->
	<element name="Reference" type="gpkg:ReferenceType" substitutionGroup="ows:Reference"/>
	<!-- ================================================================== -->
	<complexType name="ReferenceType">
		<annotation>
			<documentation>GeoPackage reference type with additional gpkg:Table elements
				to describe database tables.</documentation>
		</annotation>
		<complexContent>
			<extension base="ows:ReferenceType">
				<sequence>
					<element ref="gpkg:Table"/>
				</sequence>
			</extension>
		</complexContent>
	</complexType>
	<!-- ================================================================== -->
	<element name="ISO19139Metadata" type="gmd:MD_Metadata_Type" substitutionGroup="ows:AbstractMetaData"/>
	<!-- ================================================================== -->
	<element name="DublinCoreMetadata" type="gpkg:DublinCoreMetadataType" substitutionGroup="ows:AbstractMetaData"/>
	<!-- ================================================================== -->
	<complexType name="DublinCoreMetadataType">
		<group ref="dct:DCMI-terms"/>
	</complexType>
	<!-- ================================================================== -->
</schema>
```

Sample Manifest XML Document

```
<?xml version="1.0" encoding="UTF-8"?>
<Manifest
	xmlns="http://www.opengis.net/ows/2.0"
	xmlns:gpkg="http://www.opengis.net/gpkg/1.0"
	xmlns:xlink="http://www.w3.org/1999/xlink"
	xmlns:gmd="http://www.isotc211.org/2005/gmd"
	xmlns:dct="http://purl.org/dc/terms/"
```

```
        xmlns:dc="http://purl.org/dc/elements/1.1/"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation="http://www.opengis.net/gpkg/1.0 geoPackagemanifest.xsd">
        <ReferenceGroup>
                <Abstract>This ReferenceGroup identifies the contents of a sample GeoPackage</Abstract>
                <Identifier>SampleGeoPackage</Identifier>

                <gpkg:Reference xlink:href="GeoPackage/roads" >
                        <Identifier>RoadFeatures</Identifier>
                        <Abstract>Primary and secondary roads, no state or interstate highways.</Abstract>
                        <Metadata >
                                <gpkg:ISO19139Metadata>  <!-- schema-valid skeleton, more content required! -->
                                        <gmd:contact></gmd:contact>
                                        <gmd:dateStamp></gmd:dateStamp>
                                        <gmd:identificationInfo></gmd:identificationInfo>
                                </gpkg:ISO19139Metadata>
                        </Metadata>
                        <Metadata>
                                <gpkg:DublinCoreMetadata>
                                        <dc:publisher>U.S. Census Bureau</dc:publisher>
                                        <dc:coverage>conus</dc:coverage>
                                </gpkg:DublinCoreMetadata>
                        </Metadata>
                        <gpkg:Table>
                                <gpkg:name>roads</gpkg:name>
                                <gpkg:type>features</gpkg:type>
                                <gpkg:version>2011.03.01</gpkg:version>
                                <gpkg:TableSource xlink:href="http://www.census.gov/cgi-bin/geo/shapefiles2011/layers.cgi"/>
                        </gpkg:Table>
                </gpkg:Reference>

                <gpkg:Reference xlink:href="GeoPackage/streams">
                        <Identifier>StreamFeatures</Identifier>
                        <Abstract></Abstract>
                        <Metadata>
                                <gpkg:DublinCoreMetadata>
                                        <dc:publisher>MassGIS</dc:publisher>
                                        <dc:coverage>Massachusetts</dc:coverage>
                                </gpkg:DublinCoreMetadata>
                        </Metadata>
                        <gpkg:Table>
                                <gpkg:name>streams</gpkg:name>
                                <gpkg:type>features</gpkg:type>
                                <gpkg:version>1.0.0</gpkg:version>
                                <gpkg:TableSource xlink:href="http%3A%2F%2Fgiswebservices.massgis.state.ma.us%2Fgeoserver
%2Fwfs%3Fservice%3Dwfs%26version%3D1.1.0%26request%3DGetFeature%26typeName%3Dmassgis%3AGISDATA.NWI_ARC"/>
                        </gpkg:Table>
                </gpkg:Reference>

                <gpkg:Reference xlink:href="GeoPackage/wfstest">
                        <Identifier>WFSTestFeatures</Identifier>
                        <Abstract>Features from WFS 1.1.0 Test Suite</Abstract>
                        <gpkg:Table>
                                <gpkg:name>wfstest</gpkg:name>
                                <gpkg:type>features</gpkg:type>
                                <gpkg:version>1.1.0</gpkg:version>
```

```
                        <gpkg:RowSource primaryKey="1" xlink:href="http://www.deegree.org/services/wfs">
                            <gpkg:postQuery>
                                <wfs:GetFeature
                                    xmlns:wfs="http://www.opengis.net/wfs"
                                    xmlns:gml=http://www.opengis.net/gml
                                    xmlns:ogc="http://www.opengis.net/ogc" version="1.1.0" service="WFS">
                                    <wfs:Query
                                        xmlns:sf="http://cite.opengeospatial.org/gmlsf"
                                        typeName="sf:AggregateGeoFeature"
                                        srsName="urn:ogc:def:crs:EPSG::4326">
                                        <ogc:Filter>
                                            <ogc:PropertyIsEqualTo>

<ogc:PropertyName>sf:strProperty</ogc:PropertyName>
                                                <ogc:Literal>Ma quande lingues coalesce,
                                                    li grammatica del
resultant.</ogc:Literal>
                                            </ogc:PropertyIsEqualTo>
                                        </ogc:Filter>
                                    </wfs:Query>
                                </wfs:GetFeature>
                            </gpkg:postQuery>
                        </gpkg:RowSource>
                    </gpkg:Table>
                </gpkg:Reference>

                <gpkg:Reference xlink:href="GeoPackage/maps_rasters">
                    <Identifier>MapImages</Identifier>
                    <gpkg:Table>
                        <gpkg:name>maps</gpkg:name>
                        <gpkg:type>rasters</gpkg:type>
                        <gpkg:version>2012.04.26</gpkg:version>
                        <gpkg:RowSource primaryKey="1"
                            xlink:href="http%3A%2F%2Fwww.openstreetmap.org%2F%3Flat
                                %3D39.094%26lon%3D-76.776%26zoom%3D10%26layers%3DM">
                            <gpkg:identification>
                                <Abstract>Standard OSM map with M layers at zoom level 10</Abstract>
                                <Metadata>
                                    <gpkg:DublinCoreMetadata>
                                        <dc:publisher>Open Street Map</dc:publisher>
                                        <dc:coverage>Baltimore / Washington D.C.</dc:coverage>
                                    </gpkg:DublinCoreMetadata>
                                </Metadata>
                            </gpkg:identification>
                        </gpkg:RowSource>
                        <gpkg:RowSource primaryKey="2"
                            xlink:href="http%3A%2F%2Fapps1.gdr.nrcan.gc.ca%2Fcgi-bin%2Fworldmin_en-ca_ows
                                %3Fservice%3Dwms%26version%3D1.1.1%26request%3DGetMap%26layers
                                %3DGSC%3AWORLD_MineralDeposits">
                            <gpkg:identification>
                                <Abstract>World Mineral Deposits</Abstract>
                                <Metadata>
                                    <gpkg:DublinCoreMetadata>
                                        <dc:publisher>Natural Resources Canada</dc:publisher>
                                        <dc:coverage>world</dc:coverage>
                                    </gpkg:DublinCoreMetadata>
```

```
                    </Metadata>
                </gpkg:identification>
                </gpkg:RowSource>
            </gpkg:Table>
        </gpkg:Reference>

        <gpkg:Reference xlink:href="xlink:href=GeoPackage/coverages_rasters">
            <Identifier>Coverages</Identifier>
            <gpkg:Table>
                <gpkg:name>coverages</gpkg:name>
                <gpkg:type>rasters</gpkg:type>
                <gpkg:version>1.0.0</gpkg:version>
                <gpkg:RowSource primaryKey="1"
                    xlink:href="http%3A%2F%2Fsdf.ndbc.noaa.gov%2Fthredds%2Fwcs%2Fhfradar_usegc_1km
                        %3Fservice%3DWCS%26request%3DGetCoverage%26coverage%3Du%26bbox
                        %3D-98%2C21%2C-57%2C47%26time%3D2012-04-25T00%3A00%3A00Z
                        %26format%3DGeoTIFF">
                    <gpkg:identification>
                        <Abstract>High Frequency Radar Ocean Currents Coverage</Abstract>
                        <Identifier>surface_eastward_sea_water_velocity</Identifier>
                    </gpkg:identification>
                </gpkg:RowSource>
            </gpkg:Table>
        </gpkg:Reference>

        <gpkg:Reference xlink:href="GeoPackage/denmark_tiles">
            <Identifier>DenmarkTiles</Identifier>
            <Metadata>
                <gpkg:ISO19139Metadata>  <!-- schema-valid skeleton, more content required! -->
                    <gmd:contact></gmd:contact>
                    <gmd:dateStamp></gmd:dateStamp>
                    <gmd:identificationInfo></gmd:identificationInfo>
                </gpkg:ISO19139Metadata>
            </Metadata>
            <gpkg:Table>
                <gpkg:name>denmark_tiles</gpkg:name>
                <gpkg:type>tiles</gpkg:type>
                <gpkg:version>1.0.0</gpkg:version>
                <gpkg:RowSource primaryKey="1"
                    xlink:href="http%3A%2F%2Fkortforsyningen.kms.dk%2Ftopo_skaermkort%3Fclient%3DGaia
                        %26service%3DWMTS%26request%3DGetTile%26version%3D1.0.0%26style
                        %3Ddefault%26format%3Dimage%2Fjpeg%26TileMatrixSet%3DView1
                        %26TileMatrix%3DL05%26TileRow%3D1%26TileCol%3D1"/>
                    <gpkg:RowSource primaryKey="2"
                    xlink:href="http%3A%2F%2Fkortforsyningen.kms.dk%2Ftopo_skaermkort%3Fclient%3DGaia
                        %26service%3DWMTS%26request%3DGetTile%26version%3D1.0.0%26style
                        %3Ddefault%26format%3Dimage%2Fjpeg%26TileMatrixSet%3DView1
                        %26TileMatrix%3DL05%26TileRow%3D1%26TileCol%3D2"/>
                    <!-- many more gpkg:RowSource required here; wouldn't it be nice
                        if WMTS had a GetMatrix operation so this could be a gpkg:TableSource ? -->
            </gpkg:Table>
        </gpkg:Reference>

    </ReferenceGroup>
</Manifest>
```

References

1. ISO/IEC 9075:1992 Information Technology - Database Language SQL (SQL92)
2. ISO/IEC 9075-1:2011 Information Technology - Database Language SQL - Part 1: Framework
3. ISO/IEC 9075-2:2011 Information Technology - Database Language SQL - Part 2: Foundation
4. ISO/IEC 9075-3:2008 Information Technology - Database Language SQL - Part 3: Call-Level Interface
5. ISO/IEC 9075-4:2011 Information Technology - Database Language SQL - Part 4: Persistent Stored Modules
6. http://www.sqlite.org/
7. http://www.sqlite.org/cintro.html (CLI)
8. http://www.sqlite.org/lang.html (SQL-92)
9. http://www.sqlite.org/omitted.html (SQL-92)
10. http://developer.android.com/guide/topics/data/data-storage.html#db
11. https://developer.apple.com/technologies/ios/data-management.html
12. http://portal.opengeospatial.org/files/?artifact_id=25355

    OGC 06-103r4 OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 1: Common architecture Version: 1.2.1 2011-05-28 (also ISO/TC211 19125 Part 1)
13. http://portal.opengeospatial.org/files/?artifact_id=25354

    OGC 06-104r4 OpenGIS® Implementation Standard for Geographic information - Simple feature access - Part 2: SQL option Version: 1.2.1 2010-08-04 (also ISO/TC211 19125 Part 2)
14. http://portal.opengeospatial.org/files/?artifact_id=829

    OGC 99-049 OpenGIS® Simple Features Specification for SQL Revision 1.1 May 5, 1999, Clause 2.3.8
15. ISO/IEC 13249-3:2011 Information technology — SQL Multimedia and Application Packages - Part 3: Spatial (SQL/MM)
16. https://www.gaia-gis.it/fossil/libspatialite/index
17. http://www.gaia-gis.it/gaia-sins/spatialite-sql-3.0.0.html
18. http://trac.osgeo.org/geos/
19. http://trac.osgeo.org/geos/wiki/Applications
20. http://www.qgis.org/
21. http://hub.qgis.org/projects/android-qgis

22. http://www.gdal.org/ogr/

23. http://www.gdal.org/ogr/drv_sqlite.html

24. http://www.luciadlightspeed.com/

25. http://www.falconview.org/trac/FalconView/downloads/26

26. https://nsgreg.nga.mil/NSGDOC/files/doc/Document/MIL-STD-2500C.pdf

27. http://www.dgiwg.org/dgiwg/htm/documents/historical_documents.htm

    STANAG 7074 Digital Geographic Information Exchange Standard (DIGEST) - AGeoP-3A, edition 1, 19 October 1994

28. https://github.com/mapbox/mbtiles-spec

29. http://code.google.com/p/osmdroid/

30. http://code.google.com/p/big-planet-tracks/

31. http://www.falconview.org/trac/FalconView/doxygen/page06.htm

32. https://www.gaia-gis.it/fossil/librasterlite/index

33. http://www.gdal.org/frmt_rasterlite.html

34. http://wiki.openstreetmap.org/wiki/Main_Page

35. http://wiki.openstreetmap.org/wiki/TMS

36. http://portal.opengeospatial.org/files/?artifact_id=35326

    OGC 07-057r7_Web_Map_Tile_Service_Standard.pdf

37. http://2010.foss4g.org/presentations_show.php?id=3653

38. Portable Network Graphics http://libpng.org/pub/png/

39. ITU-T Recommendation T.81 (09/92) with Corrigendum (JPEG)

40. https://developers.google.com/speed/webp/

41. http://www.gdal.org/index.html

42. http://www.ossim.org/OSSIM/OSSIM_Home.html

43. W3C Recommendation January 1999, *Namespaces In XML*, http://www.w3.org/TR/2000/REC-xml-names.

44. W3C Recommendation 4 February 2004, *Extensible Markup Language (XML) 1.0* (Third Edition), http://www.w3.org/TR/REC-xml

45. W3C Recommendation 2 May 2001: *XML Schema Part 0: Primer*, http://www.w3.org/TR/2001/REC-xmlschema-0-20010502/

46. W3C Recommendation 2 May 2001: *XML Schema Part 1: Structures*, http://www.w3.org/TR/2001/REC-xmlschema-1-20010502/
47. W3C Recommendation 2 May 2001: *XML Schema Part 2: Datatypes*, http://www.w3.org/TR/2001/REC-xmlschema-2-20010502/

48. http://portal.opengeospatial.org/files/?artifact_id=38867

    OGC 06-121r9 OGC Web Services Common Standard Version: 2.0.0 2010-04-07 (Manifest)

49. http://schemas.opengis.net/xlink/1.0.0/xlinks.xsd

50. http://schemas.opengis.net/ows/2.0/owsManifest.xsd