Open Geospatial Consortium

Submission Date: 2018-08-13

Approval Date: 2019-03-26

Publication Date: 2019-05-08

External identifier of this OGC® document: http://www.opengis.net/doc/IS/gpkg-rte/1.0

Internal reference number of this OGC® document: 18-000

Version: 1.0

Category: OGC® Encoding Standard

Editor: Jeff Yutzler

OGC GeoPackage Related Tables Extension

Warning

This document is an OGC Member approved international standard. This document is available on a royalty free, non-discriminatory basis. Recipients of this document are invited to submit, with their comments, notification of any relevant patent rights of which they are aware and to provide supporting documentation.

Document type:	OGC® Standard
Document subtype:	
Document stage:	Approved for public release
Document language:	English

License Agreement

Permission is hereby granted by the Open Geospatial Consortium, ("Licensor"), free of charge and subject to the terms set forth below, to any person obtaining a copy of this Intellectual Property and any associated documentation, to deal in the Intellectual Property without restriction (except as set forth below), including without limitation the rights to implement, use, copy, modify, merge, publish, distribute, and/or sublicense copies of the Intellectual Property, and to permit persons to whom the Intellectual Property is furnished to do so, provided that all copyright notices on the intellectual property are retained intact and that each person to whom the Intellectual Property is furnished agrees to the terms of this Agreement.

If you modify the Intellectual Property, all copies of the modified Intellectual Property must include, in addition to the above copyright notice, a notice that the Intellectual Property includes modifications that have not been approved or adopted by LICENSOR.

THIS LICENSE IS A COPYRIGHT LICENSE ONLY, AND DOES NOT CONVEY ANY RIGHTS UNDER ANY PATENTS THAT MAY BE IN FORCE ANYWHERE IN THE WORLD.

THE INTELLECTUAL PROPERTY IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, AND NONINFRINGEMENT OF THIRD PARTY RIGHTS. THE COPYRIGHT HOLDER OR HOLDERS INCLUDED IN THIS NOTICE DO NOT WARRANT THAT THE FUNCTIONS CONTAINED IN THE INTELLECTUAL PROPERTY WILL MEET YOUR REQUIREMENTS OR THAT THE OPERATION OF THE INTELLECTUAL PROPERTY WILL BE UNINTERRUPTED OR ERROR FREE. ANY USE OF THE INTELLECTUAL PROPERTY SHALL BE MADE ENTIRELY AT THE USER'S OWN RISK. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR ANY CONTRIBUTOR OF INTELLECTUAL PROPERTY RIGHTS TO THE INTELLECTUAL PROPERTY BE LIABLE FOR ANY CLAIM, OR ANY DIRECT, SPECIAL, INDIRECT OR CONSEQUENTIAL DAMAGES, OR ANY DAMAGES WHATSOEVER RESULTING FROM ANY ALLEGED INFRINGEMENT OR ANY LOSS OF USE, DATA OR PROFITS, WHETHER IN AN ACTION OF CONTRACT, NEGLIGENCE OR UNDER ANY OTHER LEGAL THEORY, ARISING OUT OF OR IN CONNECTION WITH THE IMPLEMENTATION, USE, COMMERCIALIZATION OR PERFORMANCE OF THIS INTELLECTUAL PROPERTY.

This license is effective until terminated. You may terminate it at any time by destroying the Intellectual Property together with all copies in any form. The license will also terminate if you fail to comply with any term or condition of this Agreement. Except as provided in the following sentence, no such termination of this license shall require the termination of any third party end-user sublicense to the Intellectual Property which is in force as of the date of notice of such termination. In addition, should the Intellectual Property, or the operation of the Intellectual Property, infringe, or in LICENSOR's sole opinion be likely to infringe, any patent, copyright, trademark or other right of a third party, you agree that LICENSOR, in its sole discretion, may terminate this license without any compensation or liability to you, your licensees or any other party. You agree upon termination of any kind to destroy or cause to be destroyed the Intellectual Property together with all copies in any form, whether held by you or by any third party.

Except as contained in this notice, the name of LICENSOR or of any other holder of a copyright in all or part of the Intellectual Property shall not be used in advertising or otherwise to promote the sale, use or other dealings in this Intellectual Property without prior written authorization of LICENSOR or such copyright holder. LICENSOR is and shall at all times be the sole entity that may authorize

you or any third party to use certification marks, trademarks or other special designations to indicate compliance with any LICENSOR standards or specifications. This Agreement is governed by the laws of the Commonwealth of Massachusetts. The application to this Agreement of the United Nations Convention on Contracts for the International Sale of Goods is hereby expressly excluded. In the event any provision of this Agreement shall be deemed unenforceable, void or invalid, such provision shall be modified so as to make it valid and enforceable, and as so modified the entire Agreement shall remain in full force and effect. No decision, action or inaction by LICENSOR shall be construed to be a waiver of any rights or remedies available to it.

OGC GeoPackage Related Tables Extension

Table of Contents

1. Scope . . . 6
2. Conformance . . . 7
3. References . . . 8
4. Terms and Definitions . . . 9
   4.1. **attributes data** . . . 9
   4.2. **base data** . . . 9
   4.3. **cardinality** . . . 9
   4.4. **geospatial data** . . . 9
   4.5. **one-to-many** . . . 9
   4.6. **many-to-many** . . . 9
   4.7. **many-to-one** . . . 9
   4.8. **related data** . . . 10
   4.9. **relationship** . . . 10
   4.10. **user-defined attributes table** . . . 10
   4.11. **user-defined mapping table** . . . 10
   4.12. **user-defined media table** . . . 10
5. Conventions . . . 11
   5.1. Identifiers . . . 11
6. Design (Informative) . . . 12
   6.1. Overview . . . 12
   6.2. Requirements Classes . . . 14
      6.2.1. Media . . . 14
      6.2.2. Simple Attributes . . . 14
      6.2.3. Features . . . 15
      6.2.4. Attributes . . . 15
      6.2.5. Tiles . . . 15
   6.3. Usage scenario . . . 15
7. Requirements (Normative) . . . 20
   7.1. Common Requirements – Table Definitions . . . 20
      7.1.1. `gpkg_extensions` . . . 20
      7.1.2. `gpkgext_relations` . . . 21
      7.1.3. User-Defined Mapping Tables . . . 22
      7.1.4. User-Defined Related Data Tables . . . 24
   7.2. Media Requirements Class . . . 24
      7.2.1. User-Defined Media Tables . . . 24
   7.3. Simple Attributes Requirements Class . . . 25
      7.3.1. User-Defined Simple Attribute Tables . . . 25
   7.4. Related Features Requirements Class . . . 26

7.4.1. User-Defined Related Features Tables . . . 26
7.5. Related Attributes Requirements class . . . 27
7.5.1. User-Defined Related Attributes Tables . . . 27
7.6. Related Tiles Requirements class . . . 28
7.6.1. User-Defined Related Tiles Tables . . . 28
8. Media Types for any data encoding(s) . . . 29
Annex A: Conformance Class Abstract Test Suite (Normative) . . . 30
A.1. Common Requirements . . . 30
A.1.1. gpkg_extensions Row . . . 30
A.1.2. gpkgext_relations Row . . . 30
A.1.3. gpkg_extensions User Defined Mapping Table Rows . . . 31
A.1.4. gpkgext_relations Table . . . 31
A.1.5. gpkgext_relations Base Table . . . 32
A.1.6. gpkgext_relations Related Table . . . 32
A.1.7. gpkgext_relations User Defined Mapping Table . . . 33
A.1.8. gpkgext_relations Relation Name . . . 34
A.1.9. User Defined Mapping Table . . . 34
A.1.10. User Defined Mapping Table Base ID . . . 34
A.1.11. User Defined Mapping Table Related ID . . . 35
A.1.12. Media Relation Name . . . 35
A.1.13. Media Table Definition . . . 35
A.1.14. Simple Attributes Relation Name . . . 36
A.1.15. Simple Attributes Table Definition . . . 36
A.1.16. Features Relation Name . . . 37
A.1.17. Features Table Definition . . . 37
Annex B: Example (Informative) . . . 38
Annex C: Dublin Core Profile (Informative) . . . 40
Annex D: Table Definition SQL . . . 41
Annex E: Revision History . . . 43
Annex F: Bibliography . . . 44

i. Abstract

A GeoPackage [geopackage] is a platform-independent SQLite [sqlite] database file that contains GeoPackage data and metadata tables. GeoPackages, as described by the GeoPackage Encoding Standard [GPKG1_2] are designed to be extensible, including support for additional data types. This document defines the Related Tables Extension (RTE) for the GeoPackage Encoding Standard.

The RTE defines the rules and requirements for creating relationships in a GeoPackage data store between geospatial data tables and other tables that contain or reference related content such as attributes or media. Geospatial data tables (such as features or tiles tables) contain location information and/or geometries. There are many examples of where the RTE can be used including relating parcel (land lot) features to pictures of that parcel or linking census boundaries to the related demographic census data.

ii. Keywords

The following are keywords to be used by search engines and document catalogues.

ogcdoc, OGC document, GeoPackage, extension, related tables, multimedia

iii. Preface

The GeoPackage Related Tables Extension (RTE) defines the rules and requirements for creating relationships in a GeoPackage [geopackage] data store between geospatial or attributes data tables and other tables that contain or reference related content such as attributes or media. Geospatial data tables (such as features or tiles tables) contain location information and/or geometries. Conceptually, this extension is similar to the OGC Table Joining Service Interface Standard [OGC-TJS-IS].

One use case for this extension is to associate features with related multimedia content such as:

- photographs;
- audio or video files; and
- PDFs.

There are many examples of where the RTE can be used including relating parcel (land lot) features to pictures of that parcel or linking census boundaries to the related demographic census data. To support this capability, an ancillary data table (`gpkgext_relations`) is used in conjunction with user-defined mapping tables.

This extension, like all GeoPackage extensions, is intended to be transparent and to not interfere with GPKG compliant, but non-supporting, software packages. The OGC GeoPackage (GPKG) Related Tables Extension Interoperability Experiment (GPKG-RTE IE) [GPKG-RTE_IE] verified that the extension is correctly designed to meet the design goals and to be transparent in this manner. The goal of the IE was achieved by building GeoPackages containing embedded multimedia content and sharing those GeoPackages with other software products. This IE produced an OGC Engineering Report [GPKG-RTE_IE_ER] that discusses whether the extension is fit for use and adoption by the OGC.

The following information is provided in compliance with the GeoPackage Extension template as

defined in Annex E: GeoPackage Extension Template of the GeoPackage Standard. These additional introductory clauses are not elements in the standard OGC document template.

- Extension Author: GeoPackage SWG; author_name `gpkg` to be used upon adoption.
- Extension Name or Template: `related_tables`; upon adoption the alias `gpkg_related_tables` MAY be used
- Extension Type: This extension provides new requirements dependent on GeoPackage Clause 2.1, Clause 2.2, and Clause 2.4.
- Applicability: This extension defines relationships between feature tables and tables that hold related content, including multimedia, simple attributes, and other features.
- Scope: read-write

Attention is drawn to the possibility that some of the elements of this document may be the subject of patent rights. The Open Geospatial Consortium shall not be held responsible for identifying any or all such patent rights.

Recipients of this document are requested to submit, with their comments, notification of any relevant patent claims or other intellectual property rights of which they may be aware that might be infringed by any implementation of the standard set forth in this document, and to provide supporting documentation.

iv. Submitting organizations

The following organizations submitted this Document to the Open Geospatial Consortium (OGC):

- Image Matters LLC
- Compusult
- Radiant Solutions
- SOFWERX

v. Submitters

All questions regarding this submission should be directed to the editor or the submitters:

Name	Affiliation
Rob Cass	Compusult
Jeff Yutzler	Image Matters
Tracey Birch	SOFWERX
Jason MacDonald	Compusult
Ashley Antonides	Radiant Solutions
Brad Hards	—

Chapter 1. Scope

A GeoPackage [geopackage] is a platform-independent SQLite [sqlite] database file that contains GeoPackage data and metadata tables. GeoPackage is designed to be extensible, including support for additional data types.

This document defines an extension that allows a GeoPackage to contain additional data that is related to geospatial (generally, but not exclusively, features) or attributes data. As an example, this can be used to establish a many-to-many relationship between features (e.g. points, lines, or areas) and multimedia files.

Chapter 2. Conformance

This standard defines requirements for specialized attribute tables (containing information such as multimedia content to be stored in a GeoPackage, or simple attributes) and mapping tables to relate information in separate tables, along with metadata to describe those tables and the relationships between them.

Conformance with this standard shall be checked using all the relevant tests specified in Annex A (normative) of this document. The framework, concepts, and methodology for testing, and the criteria to be achieved to claim conformance are specified in the OGC Compliance Testing Policies and Procedures and the OGC Compliance Testing web site [1: www.opengeospatial.org/cite].

Chapter 3. References

The following normative documents contain provisions that, through reference in this text, constitute provisions of this document. For dated references, subsequent amendments to, or revisions of, any of these publications do not apply. For undated references, the latest edition of the normative document referred to applies.

- [] OGC 12-128r14, *GeoPackage Encoding Standard 1.2.1*, September 6, 2018. OGC 12-128r15 OGC® GeoPackage Encoding Standard v1.2.1 - with Corrigendum (Online)
- [] OGC 06-121r9, *OGC Web Service Common Implementation Specification*, April 7, 2010. http://portal.opengeospatial.org/files/?artifact_id=38867

Chapter 4. Terms and Definitions

This document uses the terms defined in Sub-clause 5.3 of [OWS_COMMON], which is based on the ISO/IEC Directives, Part 2, Rules for the structure and drafting of International Standards. In particular, the word “shall” (not “must”) is the verb form used to indicate a requirement to be strictly followed to conform to this standard.

For the purposes of this document, the following additional terms and definitions apply.

4.1. attributes data

Non-spatial tabular data that is designed to be joined with geospatial data for analysis. In a GeoPackage, attributes data is stored in attributes tables as per http://www.geopackage.org/spec121/#attributes.

4.2. base data

Data that is linked in some way to related data (in other words, the *left* side of the A→B relationship). In this extension, base data is stored in geospatial or attributes data tables.

4.3. cardinality

The property of a relationship between two entities, specifying whether it is one-to-one, one-to-many, many-to-one, or many-to-many.

4.4. geospatial data

Data containing location information and/or geometries. In a GeoPackage, geospatial data may be stored in features or tiles tables.

4.5. one-to-many

A type of cardinality in which an element of `A` may be linked to zero or more elements of `B`, but an element of `B` is linked to one and only one element of `A`.

4.6. many-to-many

A type of cardinality in which an element of `A` may be linked to zero or more elements of `B` and an element of `B` may be linked to zero or more elements of `A`.

4.7. many-to-one

A type of cardinality in which an element of `A` is linked to one and only one element of `B`, but an element of `B` may be linked to zero or more elements of `A`.

4.8. related data

Data that is linked to in some way from base data (in other words, the *right* side of the A→B relationship). In this extension, related data is stored in a user-defined attributes table (of which user-defined media table is a special case) or feature table.

4.9. relationship

For the purposes of this extension, a link between two entities `A` and `B`. `A` refers to base data and `B` refers to related data.

4.10. user-defined attributes table

In this extension, a user-defined attributes table is a table that contains data that is related to existing geospatial data.

4.11. user-defined mapping table

In this extension, a user-defined mapping table is a join table that links geospatial data and related data.

4.12. user-defined media table

In this extension, a user-defined media table is a user-defined attributes table that is specifically designed to contain multimedia content (including, but not limited to, images, diagrams, formatted text, moving images or audio).

Chapter 5. Conventions

This sections provides details and examples for any conventions used in the document. Examples of conventions are symbols, abbreviations, use of XML schema, or special notes regarding how to read the document.

5.1. Identifiers

The normative provisions in this specification are denoted by the URI prefix

http://www.opengis.net/spec/gpkg-rt/1.0/

All requirements and conformance tests that appear in this document are denoted by partial URIs which are relative to this base. For example, if the requirement identifier is shown as `/req/table-defs/extensions-ger` then the unique identifier for that requirement is `http://www.opengis.net/spec/gpkg-rt/1.0/req/table-defs/extensions-ger`.

Chapter 6. Design (Informative)

6.1. Overview

The core of the Related Tables Extension is a mapping between existing table types defined by GPKG 1.2.1 - features, tiles, and attributes. The mapping is defined by a new kind of table defined by the Related Tables Extension. The mapping table links related rows in those tables of those types by reference to their primary keys. For example, to link a row in Table A to a row in Table B, the mapping table includes a row that has two values - the primary key of the row from Table A, and the primary key of the row from Table B.

The mapping table allows many-to-many relationships. For example, to relate another row in Table B to the same row in Table A, the mapping table would simply include another row with the appropriate primary keys. See Figure 1.

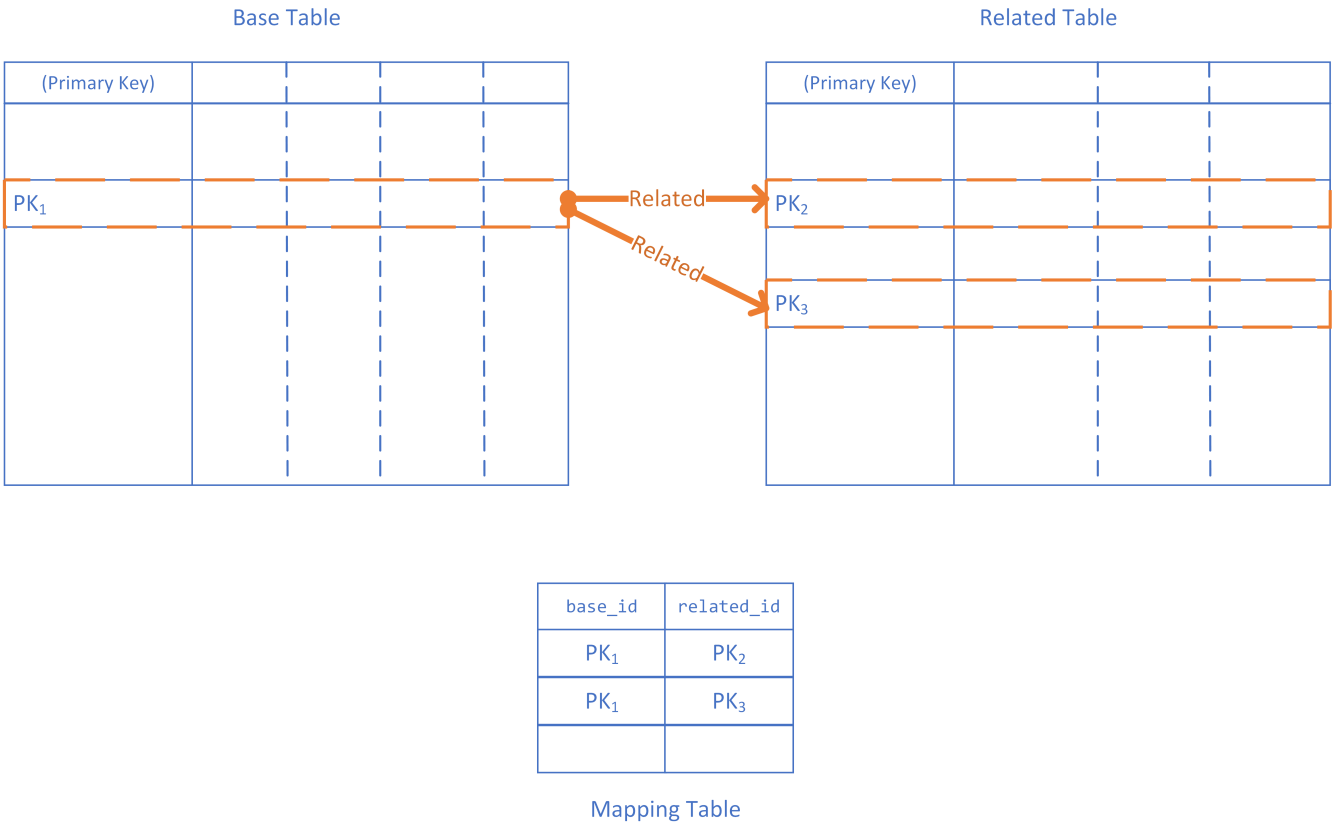


Figure 1. Related Tables concept

Mapping tables are unique to each pair of tables. The appropriate mapping table for each table pair (if any) is identified in a new table `gpkgext_relations`, which also specifies the name of the primary key column and the type of related data. This version of the Related Tables Extension supports five types of related data, which are separate conformance classes:

- media
- simple attributes
- features
- attributes

- tiles

The relationships can be considered directional, in that they relate primary keys of two tables in terms of base (the "left" or "from" side of the mapping) and related (the "right" or "to" side of the mapping). Since the related tables are valid GPKG 1.2 table types (potentially with some additional constraints), they can form the base side of another mapping. This allows chaining (directed graph) of relationships as appropriate to represent the modeled data. See Figure 2.

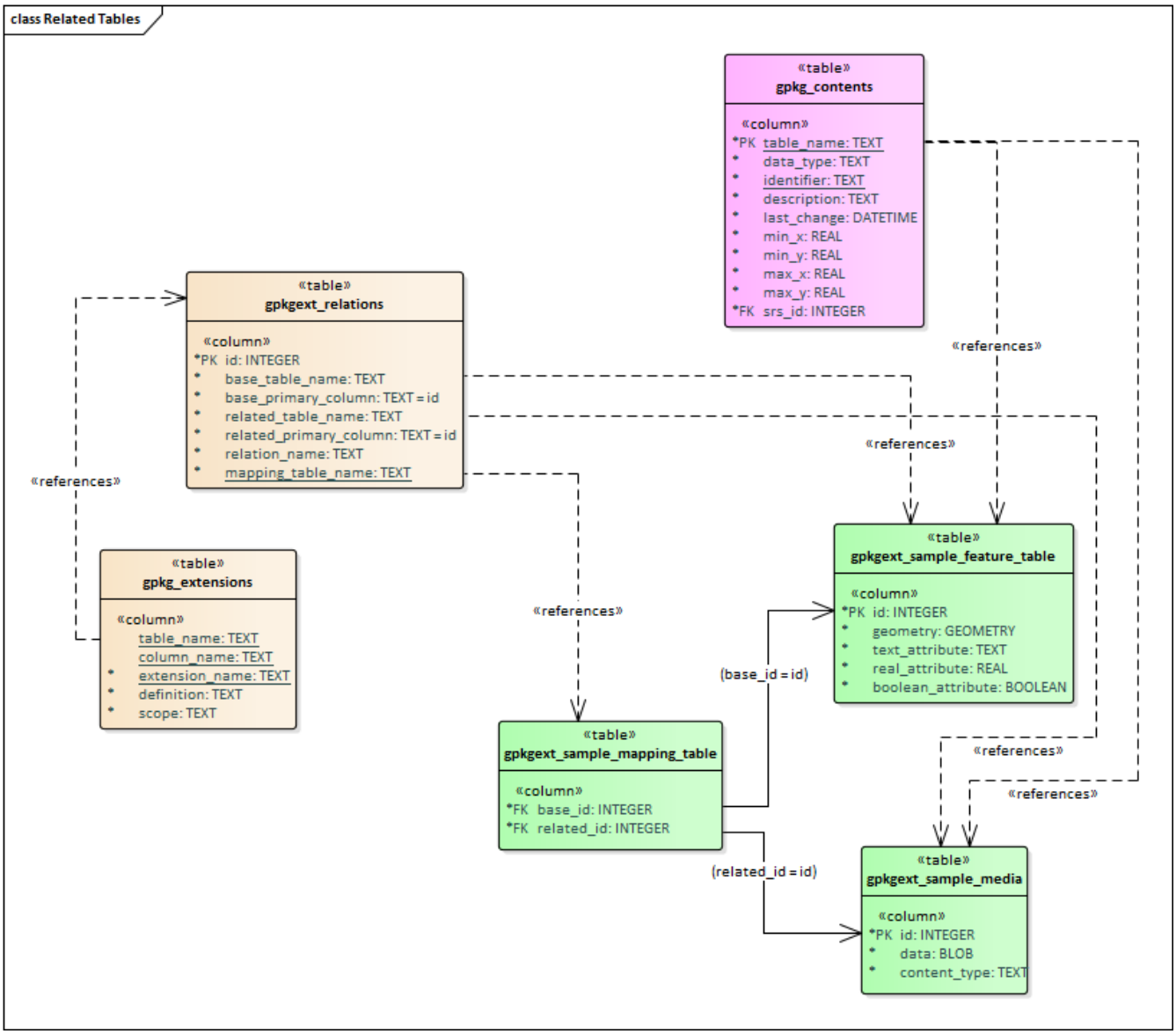


Figure 2. Related Tables UML Diagram

The Related Tables Extension makes no constraints on the base table; it can be any table type supported by GPKG 1.2 - tiles, attributes or features. The related ("right" / "to") table is constrained by defined values of `relation_name` which is a TEXT value in the `gpkgext_relations` table. The constraining of relationships serves two purposes - it allows clients to provide appropriate rendering of content, and it communicates the intent of the relationship. Since the relationship is text, values other than those defined by the Related Tables Extension document can be used, however this will not be interoperable without some other coordination mechanism.

6.2. Requirements Classes

6.2.1. Media

The Media conformance class is used for related tables that provide multimedia content. The GPKG table type is attributes. This was the original intent of the Related Tables Interoperability Experiment, and remains an important use. For example, using a `relation_name` of `media` provides the ability to link a set of photographs, line diagrams, documents, videos and/or audio files to a specific location (typically a point or polygon feature; but the Related Tables Extension does not prohibit some other kind of feature, or a row in an attribute table, or a row in a tile table being used as the base side of the mapping to the media table). The minimum content of the user defined media table is a primary key, a BLOB containing the media content (conceptually a byte array in the GeoPackage), and the IANA Media Type type for the media content (e.g., `image/jpeg` for a photograph).

An example of this is a land parcel (land lot) as the feature (base table), and photographs of the location (house, commercial property, etc.) as the related media.

Note that the related table does not need to include additional columns, although additional columns are permitted in the related table definition, so they can be added if desired. The Related Tables Extension does not constrain or codify what the additional columns can be. Specific communities of interest may wish to provide usage profiles of the Media conformance class to meet specific operational or business needs. Clients that intend to display GeoPackages that make use of the Media conformance class of the Related Tables Extension may wish to provide additional attribute display on a "best efforts" basis (e.g., view with the column names as labels for the text and numeric row values).

For example, additional column content might include:

- An indicator of the size of the media content (although this can be determined using the SQLite `length()` function);
- A title or description of the content of the media BLOB; and/or
- License information, usage restrictions, or security constraints.

6.2.2. Simple Attributes

The Simple Attributes conformance class is used for related tables that include only "simple attributes" - those SQLite values that are part of the TEXT, INTEGER and REAL storage classes. The GPKG table type is attributes. This is intended to support data that would typically be represented in Comma Separated Value (CSV) or spreadsheet formats, such as reference tables or observations. The simple attributes related table is not permitted to contain BLOB data (such as multimedia content, or feature GEOMETRY - these are covered by other conformance classes).

Only two columns are required in the related attributes table - the primary key, and one other column (which can be of TEXT, INTEGER, REAL, or a type that maps to one of those storage classes). As for Media, the Simple Attributes related table does not need to include additional columns, although additional columns are permitted in the related table definition, so they can be added if desired. The Related Tables Extension does not constrain or codify what the additional columns can

be. Specific communities of interest may wish to provide usage profiles of the Simple Attributes conformance class to meet specific operational or business needs. Clients that intend to display GeoPackages that make use of the Simple Attributes conformance class of the Related Tables Extension may wish to provide additional attribute display on a "best efforts" basis (e.g., view with the column names as labels for the text and numeric row values; or a spreadsheet-style table representation).

An example of this is a land parcel (land lot) as the feature (base table), and contact details for the managing agent as the related table. While this could be supported by embedding the contact details for each land parcel, this could be a lot of duplication and require update if a phone number or email address changes.

Note that the feature (base table) could link to many attribute table rows. An example of this would be for a set of valuations for the property, or records of property inspections or maintenance work conducted on the property.

6.2.3. Features

The Features conformance class is used for related tables that are GPKG 1.2.1 vector feature tables. The GPKG table type is features. This is intended to support defining relationships between feature types. No changes or constraints are made on the extant definition of the features tables.

An example of this is linking the location of a condominium (town house) or apartment with the locations of associated parking places or individual garden plots.

6.2.4. Attributes

The Attributes conformance class is used for related tables that comply with the GPKG attributes table type. This is intended to support additional relationships to data which may already be stored as GPKG metadata.

As with the Simple Attributes conformance class, only two columns are required in the related attributes table - the primary key, and one other column. The Related Tables Extension does not constrain or codify what additional columns can be. Unlike a Simple Attributes table, an Attributes table may include all data types allowed in a GPKG attributes metadata table (e.g. BLOB data types).

6.2.5. Tiles

The Tiles conformance class is used for related tables that are GPKG tiles tables, specifically tile pyramid tables.

6.3. Usage scenario

A single GeoPackage could include each of these relationships. For example, an airport can be considered as a point location with some attributes, which would be represented in GeoPackage as a features table. Similarly, the runways may be considered as polygons with attributes, which would be represented in GeoPackage as a different features table. See Figure 3. The mapping between those feature tables can be represented using the Related Tables Extension, so that a graphical user interface could identify and select the runways for a particular airport, including

associated attributes and metadata.

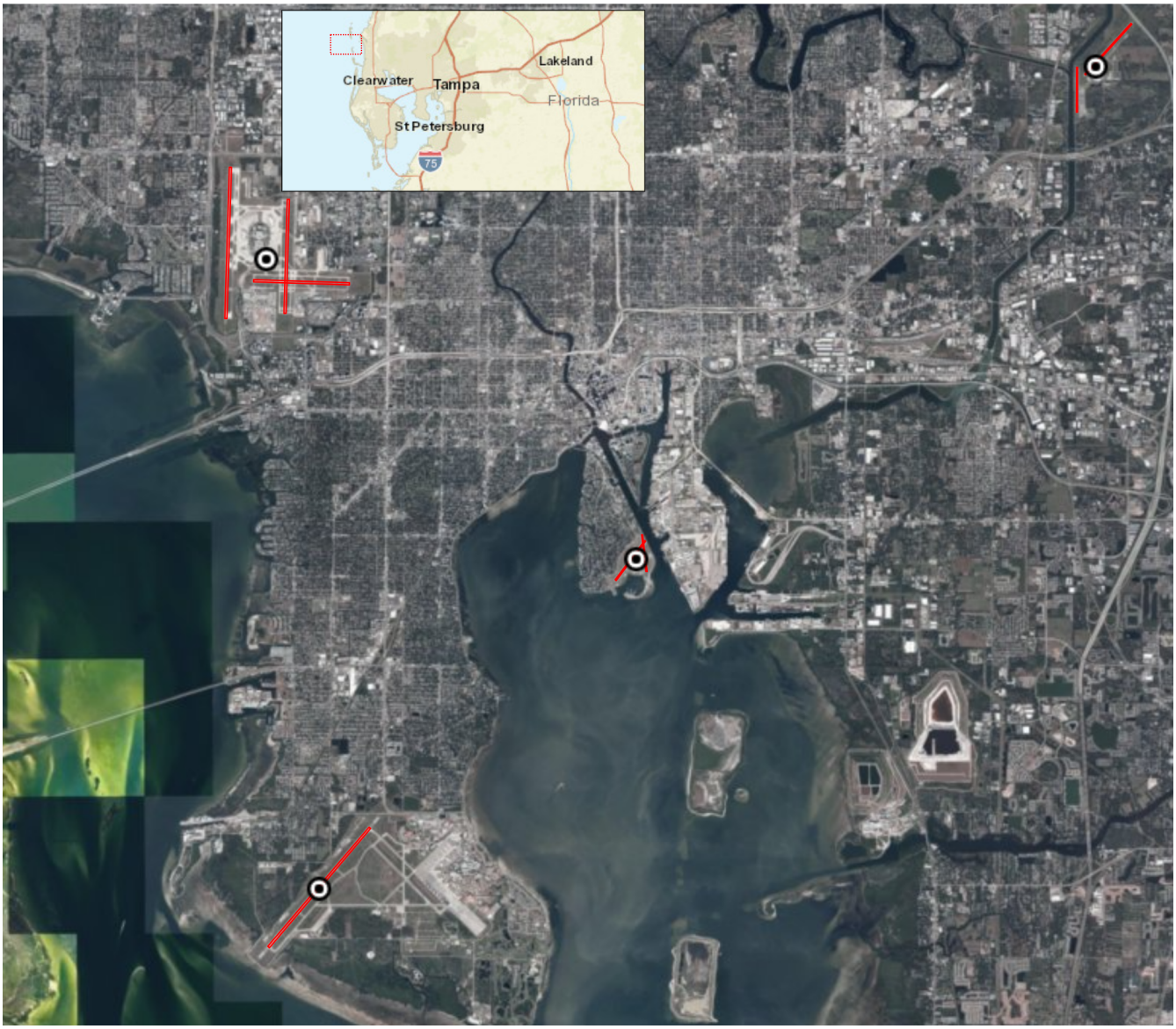


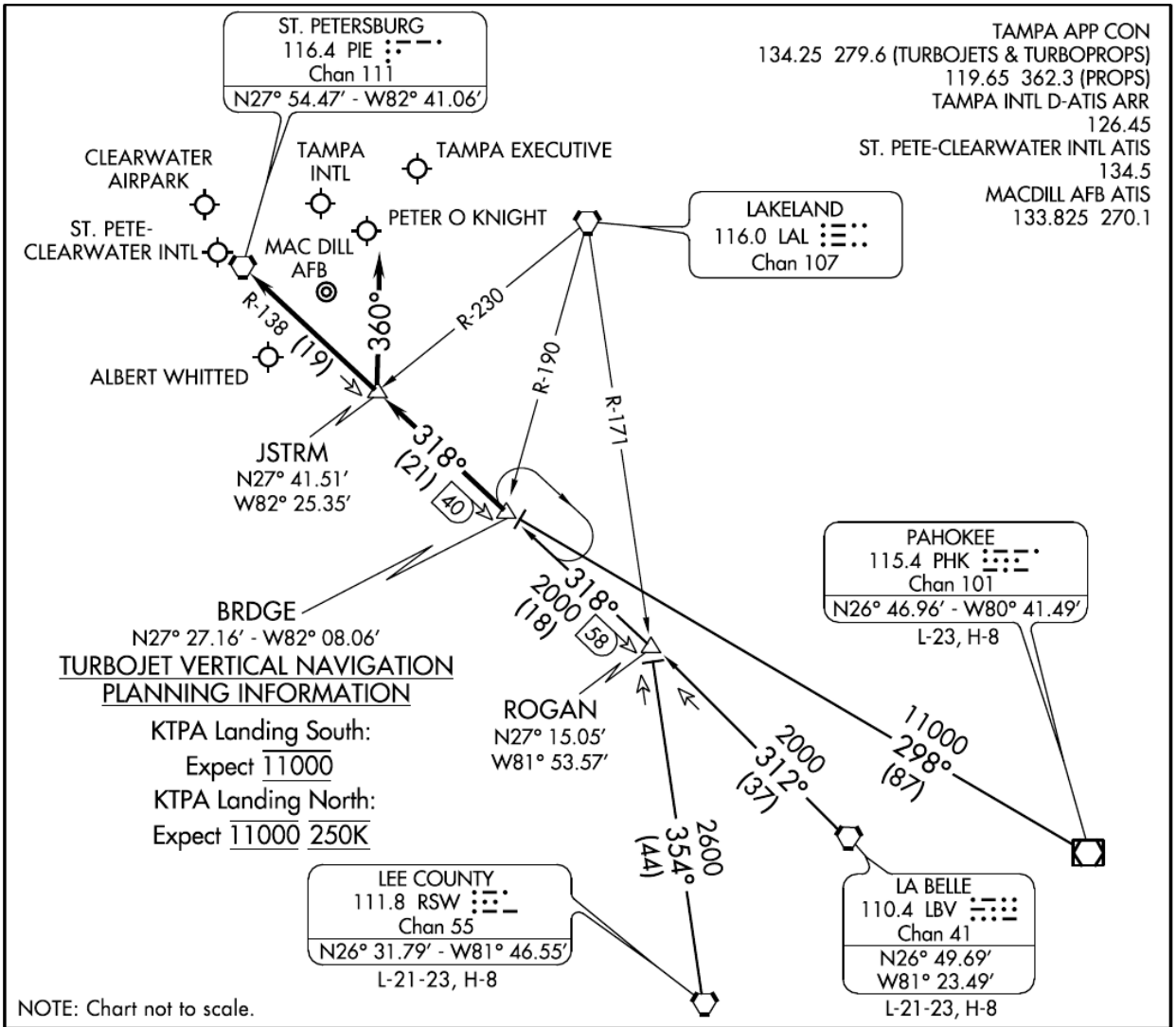
Figure 3. Airports and runways for Tampa area (from FAA data, AIRAC cycle 1802)

In addition to feature geometry, an airport may have associated documents, such as terminal procedures. These are typically provided as PDF documents containing a mix of text and diagrams, as shown in Figure 4. These could be common to a range of airports in a close area (which is the case for that Arrival procedure), specific to a particular airport, or they could be specific to a particular runway, as shown in Figure 5.

(BRDGE.BRDGE8) 17173 AL-416 (FAA)

BRDGE EIGHT ARRIVAL

TAMPA, FLORIDA



ARRIVAL ROUTE DESCRIPTION

LA BELLE TRANSITION (LBV.BRDGE8): From over LBV VORTAC on LBV R-312 & PIE R-138 to BRDGE INT. Thence. . . .

LEE COUNTY TRANSITION (RSW.BRDGE8): From over RSW VORTAC on RSW R-354 to ROGAN INT, then on PIE R-138 to BRDGE INT. Thence. . . .

PAHOKEE TRANSITION (PHK.BRDGE8): From over PHK VOR/DME on PHK R-298 to BRDGE INT. Thence. . . .

KTPA:

. . . .RWY 19L/R: From over BRDGE INT on PIE R-138 to JSTRM INT. Depart JSTRM INT heading 360° for vector to final approach course.

. . . .RWY 01L/R: From over BRDGE INT on PIE R-138 to PIE VORTAC. Expect RADAR vector to final approach course after BRDGE INT.

KPIE, KCLW, KTPF, KSPG, KMCF, KVDF:

. . . .From over BRDGE INT on PIE R-138 to PIE VORTAC. Expect RADAR vector to final approach course/airport after BRDGE INT.

LOST COMMUNICATIONS: STANDARD.

BRDGE EIGHT ARRIVAL

TAMPA, FLORIDA

(BRDGE.BRDGE8) 27APR17

Figure 4. Bridge Eight Arrival (from FAA data, AIRAC cycle 1802)

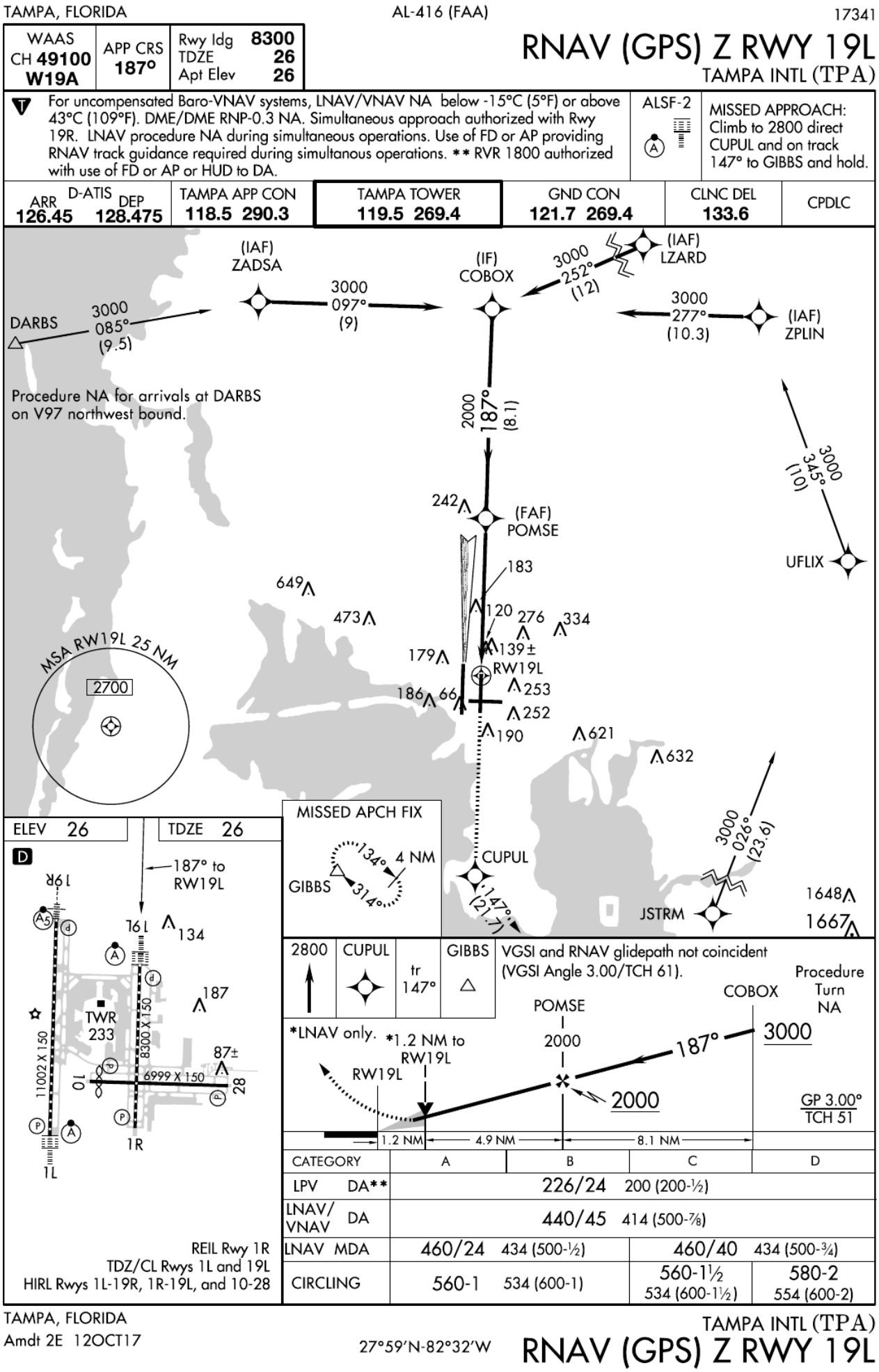


Figure 5. Tampa International RWY 19L GPS Instrument Approach(from FAA data, AIRAC cycle 1802)

This information can be represented using the user defined media table, either by incorporating the original PDF as the content of the data BLOB, or by rendering it to an image format such as PNG and using that as the content of the data BLOB. Mapping tables can relate the feature tables (e.g., airport geometry as points and runways as polygons, as described above) to the media table.

Airports may also have associated attributes that are not geospatial or media, such as the communications frequencies that are required. There are often multiple frequencies, and they are often common to multiple airports in an area. The frequency information can be represented as an attributes table, with the mapping from airport to communication frequency through a simple attributes mapping. There could well be additional attribute information, such as a mapping from the terminal procedures media table to currency (validity dates, last change) or to the responsible information provider and associated contact details.

Chapter 7. Requirements (Normative)

7.1. Common Requirements – Table Definitions

Requirements Class : Table Definitions	
http://www.opengis.net/spec/gpkg-rt/1.0/table-defs	
Target type	Token
Dependency	http://www.geopackage.org/spec121/#features
Dependency	http://www.geopackage.org/spec121/#tiles
Dependency	http://www.geopackage.org/spec121/#attributes

7.1.1. `gpkg_extensions`

Table Values

Requirement 1 – gpkg_extensions Row	*/req/table-defs/extensions-ger* A GeoPackage that contains a row in the `gpkg_extensions` table for `gpkgext_relations` as described in Extensions Table Record SHALL comply with the Related Tables Extension as described by this document.

Requirement 2 – gpkgext_relations Row	/req/table-defs/extensions-gerr A GeoPackage that contains a row in the `gpkg_extensions` table for `gpkgext_relations` SHALL contain at least one related table relationship.

WARNING	The intent of this requirement is that a GeoPackage shall not declare the Related Tables Extension unless it is used within the GeoPackage. As a consequence, if all rows are removed from `gpkgext_relations`, that table must be dropped and all references to the extension must be removed from `gpkg_extensions` as well. However, because a related tables schema may be established without any data, empty user-defined mapping tables and/or related data tables are permitted.

Requirement 3 – gpkg_extensions User Defined Mapping Table Rows	/req/table-defs/extensions-udmt A GeoPackage that complies with the Related Tables Extension SHALL contain rows in the `gpkg_extensions` table for each User Defined Mapping Table as described in Extensions Table Record.

Table 1. Extensions Table Record

table_name	column_name	extension_name	definition	scope
`gpkgext_relations`	null	`related_tables`	TBD	`read-write`
name of actual User Defined Mapping Table	null	`related_tables`	TBD	`read-write`

7.1.2. `gpkgext_relations`

Requirement 4 – gpkgext_relations Table	/req/table-defs/ger A GeoPackage that complies with this extension SHALL contain a `gpkgext_relations` table as per Extended Relations Table Definition and Extended Relations Table Definition SQL (Normative).

Table Definition

Table 2. Extended Relations Table Definition

Column Name	Column Type	Column Description	Null	Default	Key	Unique
`id`	INTEGER	Autoincrement primary key	no		PK	yes
`base_table_name`	TEXT	Name of the table containing the base data (e.g., features) to relate	no			no
`base_primary_column`	TEXT	Name of the primary key column in `base_table_name`	no	`id`		no
`related_table_name`	TEXT	Name of the table containing the related content	no			no
`related_primary_column`	TEXT	Name of the primary key column in `related_table_name`	no	`id`		no
`relation_name`	TEXT	Name (profile) of the relationship	no			no
`mapping_table_name`	TEXT	Name of a mapping table	no			yes

Table Values

Requirement 5 – gpkgext_relations Base Table	/req/table-defs/ger-base For each row in `gpkgext_relations`, there SHALL be a table or view of the name referenced in `base_table_name` and that table or view SHALL have an entry in `gpkg_contents`.

NOTE	See Example User Defined Features Table Definition SQL (Informative) for example SQL for creating a features table for use with this extension.

Requirement 6 – gpkgext_relations Related Table	/req/table-defs/ger_related For each row in `gpkgext_relations`, there SHALL be a table or view of the name referenced in `related_table_name` and that table or view SHALL have an entry in `gpkg_contents`.

Requirement 7 – gpkgext_relations User Defined Mapping Table	/req/table-defs/ger_udmt For each row in `gpkgext_relations`, the `mapping_table_name` column SHALL contain the name of a user-defined mapping table or view as described by User-Defined Mapping Tables.

Requirement 8 – gpkgext_relations Relation Name	/req/table-defs/ger_relname Each `relation_name` column in a `gpkgext_relations` row SHALL either match a `relation_name` from the Requirements Classes for User-Defined Related Data Tables in this or other OGC standards (e.g. `media` for [Media Requirement Class]), or be of the form `x-`<author>_<relation_name> where <author> indicates the person or organization that developed and maintains this set of User-Defined Related Tables.

7.1.3. User-Defined Mapping Tables

Table Definition

Requirement 9 – User Defined Mapping Table	/req/table-defs/udmt A user-defined mapping table or view SHALL contain all of the columns described in User-Defined Mapping Table Definition.

NOTE A user-defined mapping table MAY contain no rows. This is intended to support declaring a conceptual relationship where no instances of that relationship currently exist in the GeoPackage (e.g. no related data has been added, or all applicable data has been deleted).

Table 3. User-Defined Mapping Table Definition

Column Name	Column Type	Column Description	Null	Default	Key	Unique
`base_id`	INTEGER	The primary key value of the base data table	no			no
`related_id`	INTEGER	The primary key value of the related data table	no			no

NOTE See Extended Relations Mapping Table SQL (Informative) for example SQL for defining a user-defined mapping table. A user-defined mapping table MAY be implemented as a GPKG attributes table type, and MAY contain other columns not listed here.

Table Values

Requirement 10 – User Defined Mapping Table Base ID	/req/table-defs/udmt_base For each row of a user-defined mapping table, the `base_id` column SHALL correlate to the primary key of the corresponding base table (as identified by the `base_primary_column` of the associated row in `gpkgext_relations`).

Requirement 11 – User Defined Mapping Table Related ID	/req/table-defs/udmt_related For each row of a user-defined mapping table, the `related_id` column SHALL correlate to the primary key of the corresponding related data table (as identified by the `related_primary_column` of the associated row in `gpkgext_relations`).

NOTE This specification makes no statement on the cardinality of a user-defined mapping table. It MAY contain no rows or have a one-to-many, many-to-one, or many-to-many relationship. While it is possible to enforce a one-to-many or many-to-one relationship by applying a UNIQUE constraint to the `attributes_id` or `base_id` respectively, this is NOT RECOMMENDED because the presence of these constraints is not exposed by SQLite in an easy-to-query manner.

7.1.4. User-Defined Related Data Tables

There are no explicit requirements for a generic user-defined related data table. However, interoperability is improved if a specific requirements class is used. The following subsections describe currently defined requirements classes.

- Media Requirements Class (for multimedia files)
- Simple Attributes Requirements Class (for text and numeric data)
- Related Features Requirements Class (for features)
- Related Attributes Requirements class (for attributes metadata)
- Related Tiles Requirements class (for tiles)

7.2. Media Requirements Class

Requirements Class : Media	
http://www.opengis.net/spec/gpkg-rt/1.0/media	
Target type	Token
Dependency	www.opengis.net/spec/gpkg-rt/1.0/table-defs

This requirements class allows a GeoPackage producer or editor to relate features or attributes to multimedia files such as pictures and videos. For example, this could relate building features with pictures or videos taken of those areas.

7.2.1. User-Defined Media Tables

Table Definition

Requirement 12 – Media Relation Name	/req/media/udmt A user-defined related data table or view SHALL be a user-defined media table or view if the row in `gpkgext_relations` with a corresponding `related_table_name` has a `relation_name` of "media". A user-defined media table or view SHALL meet all requirements of a GPKG attributes table type.
Requirement 13 – Media Table Definition	/req/media/table_def A user-defined media table or view SHALL contain all of the columns described in User-Defined Media Table Definition.

Table 4. User-Defined Media Table Definition

Column Name	Column Type	Column Description	Null	Key
any	INTEGER	Autoincrement primary key	no	PK
`data`	BLOB	Multimedia content	no	
`content_type`	TEXT	Mime-type of data	no	

NOTE	See Example User Defined Media Table Definition SQL (Informative) for example SQL for defining a user-defined media table. A user-defined media table MAY also contain other columns not listed here. Refer to the Dublin Core profile (Informative) for a set of additional columns that may be provided.

7.3. Simple Attributes Requirements Class

Requirements Class : Simple Attributes	
http://www.opengis.net/spec/gpkg-rt/1.0/simpleattr	
Target type	Token
Dependency	www.opengis.net/spec/gpkg-rt/1.0/table-defs

This requirements class allows a GeoPackage producer or editor to relate sets of tabular text or numeric data. For example, this could relate land parcels with statistical information that is collected over different boundaries, or at different times.

7.3.1. User-Defined Simple Attribute Tables

Table Definition

Requirement 14 – Simple Attributes Relation Name	/req/simpleattr/udat A user-defined related data table or view SHALL be a user-defined simple attribute table or view if the row in `gpkgext_relations` with a corresponding `related_table_name` has a `relation_name` of "simple_attributes". A user-defined simple attributes table or view SHALL meet all requirements of a GPKG attributes table type.

Requirement 15 – Simple Attributes Table Definition	/req/simpleattr/table_def A user-defined simple attribute table or view SHALL contain the primary key column and at least one other column as defined in User-Defined Simple Attributes Table Definition. A user-defined simple attribute table SHALL only contain data belonging to the TEXT, INTEGER, or REAL storage classes and SHALL NOT contain NULL or BLOB storage classes (e.g., GEOMETRY).

Table 5. User-Defined Simple Attributes Table Definition

Column Name	Column Type	Column Description	Null	Key
any	INTEGER	Autoincrement primary key	no	PK
any	TEXT	Text attribute content	no	
any	INTEGER	Integer attribute content	no	
any	REAL	Floating point number attribute content	no	

NOTE See Example User Defined Simple Attribute Table Definition SQL (Informative) for example SQL for defining a user-defined simple attributes table.

7.4. Related Features Requirements Class

Requirements Class : Related Features	
http://www.opengis.net/spec/gpkg-rt/1.0/relatedfeat	
Target type	Token
Dependency	www.opengis.net/spec/gpkg-rt/1.0/table-defs

This requirements class allows a GeoPackage producer or editor to link features with other features. For example, this could link road features with related pylon features.

7.4.1. User-Defined Related Features Tables

Table Definition

Requirement 16 – Features Relation Name	/req/relatedfeat/udat A user-defined related data table or view SHALL be a user-defined related features table or view if the row in `gpkgext_relations` with a corresponding `related_table_name` has a `relation_name` of "features".

Requirement 17 – Features Table Definition	/req/relatedfeat/table_def A user-defined related features table or view SHALL be a GPKG vector feature table type.

7.5. Related Attributes Requirements class

Requirements Class : Attributes	
http://www.opengis.net/spec/gpkg-rt/1.0/attr	
Target type	Token
Dependency	www.opengis.net/spec/gpkg-rt/1.0/table-defs

This requirements class allows a GeoPackage producer or editor to link metadata which may already exist in a GPKG Attributes table, including data types beyond the Simple Attributes set (e.g. BLOB data).

7.5.1. User-Defined Related Attributes Tables

Table Definition

Requirement 18 – Attributes Relation Name	/req/relatedattr/udat A user-defined related data table or view SHALL be a user-defined related attributes table or view if the row in `gpkgext_relations` with a corresponding `related_table_name` has a `relation_name` of "attributes".

Requirement 19 – Attributes Table Definition	/req/relatedattr/table_def A user-defined related attributes table or view SHALL be a GPKG attributes table type.

7.6. Related Tiles Requirements class

Requirements Class : Tiles	
http://www.opengis.net/spec/gpkg-rt/1.0/tiles	
Target type	Token
Dependency	www.opengis.net/spec/gpkg-rt/1.0/table-defs

This requirements class allows a GeoPackage producer or editor to link to individual tiles in a GPKG tile pyramid user data table.

7.6.1. User-Defined Related Tiles Tables

Table Definition

Requirement 20 – Tiles Relation Name	/req/relatedtiles/udat A user-defined related data table or view SHALL be a user-defined related tiles table or view if the row in `gpkgext_relations` with a corresponding `related_table_name` has a `relation_name` of "tiles".

Requirement 21 – Tiles Table Definition	/req/relatedtiles/table_def A user-defined related features table or view SHALL be a GPKG tile pyramid table type.

Chapter 8. Media Types for any data encoding(s)

This standard provides an extension to GeoPackage and does not affect the GeoPackage data encoding.

Annex A: Conformance Class Abstract Test Suite (Normative)

NOTE	Ensure that there is a conformance class for each requirements class and a test for each requirement (identified by requirement name and number)

A.1. Common Requirements

A.1.1. gpkg_extensions Row

Test id:	/conf/table-defs/applicability
Requirement:	/req/table-defs/extensions-ger
Test purpose:	Verify whether the Related Tables Extension is applicable.
Test method:	`SELECT COUNT(*) FROM gpkg_extensions WHERE extension_name IN ('related_tables', 'gpkg_related_tables')` Extension not in use if count is empty

Test id:	/conf/table-defs/extensions-ger
Requirement:	/req/table-defs/extensions-ger
Test purpose:	Verify that the `gpkg_extensions` table contains the required `gpkgext_relations` table.
Test method:	`SELECT * FROM gpkg_extensions WHERE table_name = 'gpkgext_relations'` Fail if number of rows returned is not 1. Pass if results match required content from Extensions Table Record, else Fail.

A.1.2. gpkgext_relations Row

Test id:	/conf/table-defs/extensions-gerr
Requirement:	/req/table-defs/extensions-gerr
Test purpose:	Verify that the `gpkg_extensions` table contains at least one user defined mapping table.

Test method:	1. `SELECT table_name FROM gpkg_extensions WHERE (extension_name IN ('related_tables', 'gpkg_related_tables') AND table_name != 'gpkgext_relations')` 2. Fail if number of rows returned is 0. 3. For each `table_name` row returned in Step 1: - `SELECT tbl_name FROM sqlite_master WHERE tbl_name = '{table_name}'` - Fail if no rows returned. 4. Pass if no fails.

A.1.3. gpkg_extensions User Defined Mapping Table Rows

Test id:	/conf/table-defs/extensions-udmt
Requirement:	/req/table-defs/extensions-udmt
Test purpose:	Verify that the `gpkg_extensions` table contains the required user defined mapping table contents
Test method:	1. `SELECT column_name, definition, scope FROM gpkg_extensions WHERE (extension_name IN ('related_tables', 'gpkg_related_tables') AND table_name != 'gpkgext_relations')` 2. Fail if number of rows returned is 0. 3. For each row returned in Step 1, fail if `column_name` is not null, `definition` is not "TBD" or `scope` is not "read-write". 4. Pass if no fails.

A.1.4. gpkgext_relations Table

Test id:	/conf/table-defs/ger
Requirement:	/req/table-defs/ger
Test purpose:	Verify that the `gpkgext_relations` table has the required table definition.
Test method:	1. `PRAGMA table_info(gpkgext_relations)` 2. Fail if returns an empty result set. 3. Pass if the column names, types, nullability, default values, primary key, and unique key constraints match all of those in the contents of Extended Relations Table Definition. Column order, check constraint and trigger definitions, and other column definitions in the returned sql are irrelevant. 4. Fail otherwise.

A.1.5. gpkgext_relations Base Table

Test id:	/conf/table-defs/ger-base
Requirement:	/req/table-defs/ger-base
Test purpose:	Verify that the base tables listed in `gpkgext_relations` table are in the GeoPackage database
Test method:	1. `SELECT base_table_name FROM gpkgext_relations` 2. Fail if returns an empty result set. 3. For each `base_table_name` row returned in Step 1: - `SELECT tbl_name FROM sqlite_master WHERE tbl_name = '{base_table_name}'` - Fail if no rows returned. 4. Pass if no fails.

Test id:	/conf/table-defs/ger-base-contents
Requirement:	/req/table-defs/ger-base
Test purpose:	Verify that the base tables listed in `gpkgext_relations` table are in the GeoPackage contents table
Test method:	1. `SELECT base_table_name FROM gpkgext_relations` 2. Fail if returns an empty result set. 3. For each `base_table_name` row returned in Step 1: - `SELECT table_name FROM gpkg_contents WHERE table_name = '{base_table_name}'` - Fail if no rows returned. 4. Pass if no fails.

A.1.6. gpkgext_relations Related Table

Test id:	/conf/table-defs/ger-related
Requirement:	/req/table-defs/ger_related
Test purpose:	Verify that the related tables listed in `gpkgext_relations` table are in the GeoPackage database

Test method:	1. `SELECT related_table_name FROM gpkgext_relations` 2. Fail if returns an empty result set. 3. For each `related_table_name` row returned in Step 1: - `SELECT tbl_name FROM sqlite_master WHERE tbl_name = '{related_table_name}'` - Fail if no rows returned. 4. Pass if no fails.

Test id:	/conf/table-defs/ger-related-contents
Requirement:	/req/table-defs/ger_related
Test purpose:	Verify that the related tables listed in `gpkgext_relations` table are in the GeoPackage contents table
Test method:	1. `SELECT related_table_name FROM gpkgext_relations` 2. Fail if returns an empty result set. 3. For each `related_table_name` row returned in Step 1: - `SELECT table_name FROM gpkg_contents WHERE table_name = '{related_table_name}'` - Fail if no rows returned. 4. Pass if no fails.

A.1.7. gpkgext_relations User Defined Mapping Table

Test id:	/conf/table-defs/ger-udmt
Requirement:	/req/table-defs/ger_udmt
Test purpose:	Verify that the mapping tables listed in `gpkgext_relations` table are in the GeoPackage database
Test method:	1. `SELECT mapping_table_name FROM gpkgext_relations` 2. Fail if returns an empty result set. 3. For each `mapping_table_name` row returned in Step 1: - `SELECT tbl_name FROM sqlite_master WHERE tbl_name = '{mapping_table_name}'` - Fail if no rows returned. 4. Pass if no fails.

A.1.8. gpkgext_relations Relation Name

Test id:	/conf/table-defs/ger-relname
Requirement:	/req/table-defs/ger_relname
Test purpose:	Verify that the relation_name entries listed in `gpkgext_relations` table are valid.
Test method:	1. `SELECT base_table_name, relation_name FROM gpkgext_relations WHERE (relation_name NOT IN ('features', 'simple_attributes', 'media') AND relation_name NOT LIKE 'x-%\_%' ESCAPE '\')` 2. Fail if returns any rows

A.1.9. User Defined Mapping Table

Test id:	/conf/table-defs/udmt
Requirement:	/req/table-defs/udmt
Test purpose:	Verify that the mapping tables listed in `gpkgext_relations` table have the correct structure
Test method:	1. `SELECT mapping_table_name FROM gpkgext_relations` 2. Fail if returns an empty result set. 3. For each `mapping_table_name` row returned in Step 1: - `PRAGMA table_info({mapping_table_name})` - Fail if returns an empty result set. - Fail if the column names, types, nullability, default values, primary key, and unique key constraints do match all of those in the contents of User-Defined Mapping Table Definition. Additional columns, column order, check constraint and trigger definitions, and other column definitions in the returned sql are irrelevant. 4. Pass if no fails.

A.1.10. User Defined Mapping Table Base ID

Test id:	/conf/table-defs/udmt-base
Requirement:	/req/table-defs/udmt_base
Test purpose:	Verify that the contents of mapping tables listed in `gpkgext_relations` table correlate to base table rows

Test method:	1. `SELECT base_table_name, base_primary_column, mapping_table_name FROM gpkgext_relations` 2. Fail if returns an empty result set. 3. For each row returned in Step 1, fail if any `base_id` value in {mapping_table_name} does not match an entry in the {base_primary_column} column of the {base_table_name} table. 4. Pass if no fails.

A.1.11. User Defined Mapping Table Related ID

Test id:	/conf/table-defs/udmt-related
Requirement:	/req/table-defs/udmt_related
Test purpose:	Verify that the contents of mapping tables listed in `gpkgext_relations` table correlate to related table rows
Test method:	1. `SELECT related_table_name, related_primary_column, mapping_table_name FROM gpkgext_relations` 2. Fail if returns an empty result set. 3. For each row returned in Step 1, fail if any `related_id` value in {mapping_table_name} does not match an entry in the {related_primary_column} column of the {related_table_name} table. 4. Pass if no fails.

A.1.12. Media Relation Name

Test id:	/conf/media/udmt
Requirement:	/req/media/udmt
Test purpose:	Verify whether the requirements class applies for user-defined media tables
Test method:	`SELECT COUNT(*) FROM gpkgext_relations WHERE relation_name = 'media'` Requirement class is not in use if count is zero

A.1.13. Media Table Definition

Test id:	/conf/media/table_def
Requirement:	/req/media/table_def
Test purpose:	Verify that the user-defined media tables have the correct structure

Test method:	1. `SELECT related_table_name FROM gpkgext_relations WHERE relation_name = 'media'` 2. For each related_table_name row returned in Step 1: - `PRAGMA table_info({related_table_name})` - Fail if returns an empty result set. - Fail if the column names, types, nullability, default values, primary key, and unique key constraints do match all of those in the contents of User-Defined Media Table Definition. Additional columns, column order, check constraints, trigger definitions, and other column definitions in the returned sql are irrelevant. 3. Pass if no fails.

A.1.14. Simple Attributes Relation Name

Test id:	/conf/simpleattr/udat
Requirement:	/req/simpleattr/udat
Test purpose:	Verify whether the requirements class applies for user-defined simple attributes tables
Test method:	`SELECT COUNT(*) FROM gpkgext_relations WHERE relation_name = 'simple_attributes'` Requirement class is not in use if count is zero

A.1.15. Simple Attributes Table Definition

Test id:	/conf/simpleattr/table_def
Requirement:	/req/simpleattr/table_def
Test purpose:	Verify that the user-defined simple attribute tables have the correct structure
Test method:	1. `SELECT related_table_name FROM gpkgext_relations WHERE relation_name = 'simple_attributes'` 2. For each related_table_name row returned in Step 1: - `PRAGMA table_info({related_table_name})` - Fail if returns an empty result set. - Fail if the column names, types, nullability, default values, primary key, and unique key constraints do match all of those in the contents of User-Defined Simple Attributes Table Definition. - Fail if `Column Type` includes storage classes other than `TEXT`, `INTEGER`, or `REAL`. 3. Pass if no fails.

A.1.16. Features Relation Name

Test id:	/conf/relatedfeat/udat
Requirement:	/req/relatedfeat/udat
Test purpose:	Verify whether the requirements class applies for user-defined features tables
Test method:	`SELECT COUNT(*) FROM gpkgext_relations WHERE relation_name = 'features'` Requirement class is not in use if count is zero

A.1.17. Features Table Definition

Test id:	/conf/relatedfeat/table_def
Requirement:	/req/relatedfeat/table_def
Test purpose:	Verify that the user-defined related features tables have the correct structure
Test method:	1. `SELECT related_table_name FROM gpkgext_relations WHERE relation_name = 'features'` 2. For each related_table_name row returned in Step 1: - `PRAGMA table_info({related_table_name})` - Fail if returns an empty result set. - Fail if the column names, types, nullability, default values, primary key, and unique key constraints do match all of those in the contents of [http://www.geopackage.org/spec/#feature_user_tables]. 3. Pass if no fails.

Annex B: Example (Informative)

This example illustrates support for many-to-many relationships but the concept may be used in a degenerative way to support one-to-many or many-to-one relationships. The content of the `gpkgext_relations` includes a features_to_media table that relates the features table values and media table values using their respective `id` columns.

In this example, there are four features (ID 1, 2, 3 and 4) and three PNG media items (ID 17, 18, and 19). Using the features_to_media table,

- feature 1 relates to media 17 and 18
- feature 2 relates to media 18
- feature 3 relates to media 18
- feature 4 relates to media 17 and 19

Table 6. gpkgext_relations table values

base_table_name	base_column	related_table_name	related_column	relation_name	mapping_table_name
features	id	media	id	media	features_to_media

The following SQL query will reveal this information:

```
select base_table_name,
    base_primary_column,
    related_table_name,
    related_primary_column,
    mapping_table_name
from gpkgext_relations;
```

Table 7. features table values

id	geom
1	<BLOB>
2	<BLOB>
3	<BLOB>
4	<BLOB>

Table 8. media table values

id	data	content_type
17	<BLOB>	image/png
18	<BLOB>	image/png
19	<BLOB>	image/png

Table 9. features_to_media table

base_id	related_id
4	17
4	19
3	18
2	18
1	18
1	17

The features_to_media table table relates the `id` columns between the features table and the media table.

The following SQL query will reveal this information:

```
select features.id, features_to_media.related_id
    from features, features_to_media
where features.id=features_to_media.base_id;
```

Annex C: Dublin Core Profile (Informative)

The Related Tables Extension intends to be flexible in which columns are required. The goal is to support a wide variety of input data. However, with loose control over the columns it can be difficult for users to understand why a specific relationship exists. This annex presents a recommendation that is designed to improve interoperability while maintaining as flexible a structure as possible. It is based on Dublin Core elements.

Columns for the following Dublin Core elements are RECOMMENDED to be present in a mapping table or a related data table. Clients SHOULD look for these elements and attempt to present them to the user where possible.

- date
- description
- source
- title

Some columns use different names from Dublin Core elements, as shown in Column Name / Dublin Core Synonyms.

Table 10. Column Name / Dublin Core Synonyms

Related Table Column Name	Synonymous Dublin Core Element
`id`	`identifier`
`content_type`	`format`

Table 11. Example User Defined Media Table Definition SQL - Dublin Core (Informative)

```
CREATE TABLE 'sample_media_with_metadata' (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  data BLOB NOT NULL,
  content_type TEXT NOT NULL,
  title TEXT NOT NULL,
  description TEXT NOT NULL,
  date INTEGER NOT NULL,
  source TEXT NOT NULL)
```

NOTE There are three ways to store dates in SQLite (and in GeoPackage) - as TEXT (ISO-8601), as REAL (days relative to Gregorian calendar start) and INTEGER (seconds relative to Unix epoch). See SQLite documentation for more information including conversion between forms.

Annex D: Table Definition SQL

Table 12. Extended Relations Table Definition SQL (Normative)

```
CREATE TABLE 'gpkgext_relations' (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  base_table_name TEXT NOT NULL,
  base_primary_column TEXT NOT NULL DEFAULT 'id',
  related_table_name TEXT NOT NULL,
  related_primary_column TEXT NOT NULL DEFAULT 'id',
  relation_name TEXT NOT NULL,
  mapping_table_name TEXT NOT NULL UNIQUE
 );
```

Table 13. Extended Relations Mapping Table SQL (Informative)

```
CREATE TABLE 'sample_mapping_table' (
  base_id INTEGER NOT NULL,
  related_id INTEGER NOT NULL
 );
```

Table 14. Example User Defined Features Table Definition SQL (Informative)

```
CREATE TABLE 'sample_feature_table' (
  id INTEGER PRIMARY KEY AUTOINCREMENT NOT NULL,
  geometry GEOMETRY,
  text_attribute TEXT,
  real_attribute REAL,
  boolean_attribute BOOLEAN)
```

This table is a modified version of the informative example in the core document. The `raster_or_photo` column is removed with the intent of making the binary data available as per this extension.

Table 15. Example User Defined Media Table Definition SQL (Informative)

```
CREATE TABLE 'sample_media' (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  data BLOB NOT NULL,
  content_type TEXT NOT NULL)
```

Table 16. Example User Defined Simple Attribute Table Definition SQL (Informative)

```
CREATE TABLE 'sample_attributes' (
  id INTEGER PRIMARY KEY AUTOINCREMENT,
  label TEXT NOT NULL,
  description TEXT NOT NULL,
  arisings INTEGER NOT NULL)
```

Annex E: Revision History

Date	Release	Editor	Primary clauses modified	Description
2017-12	0.1	J. Yutzler	all	initial version
2018-06	0.2	J. Yutzler	all	post-IE

Annex F: Bibliography

- [] http://www.geopackage.org/
- [] http://www.sqlite.org/
- [] GeoPackage Related Tables Extension Interoperability Experiment
- [] OGC: OGC 17-093r1, GeoPackage Related Tables Extension Interoperability Experiment Engineering Report, 2018
- [] OGC: OGC 10-070r2 Table Joining Service Interface Standard, 2010