GBIF
www.gbif.org

# Darwin Core Archive Format

## Reference Guide to the XML Descriptor File

**Version 1.0**

**April 2011**

This document incorporates and extends the Darwin Core Text Guide located at http://rs.tdwg.org/dwc/terms/guides/text/index.htm. It provides a detailed overview of the XML descriptor file used to document the Darwin Core Archive format. The descriptor file describes how biodiversity data files are organized.

**Suggested citation:** GBIF (2011). Darwin Core Archive Format, Reference Guide to the XML Descriptor File, April 2011, (contributed by Döring, M., Robertson, T., Remsen, D.), Copenhagen: Global Biodiversity Information Facility,16 pp.

**ISBN:** 87-92020-22-4

**Persistent URI:** http://links.gbif.org/gbif_dwc-a_metafile_en_v1/

**Language:** English



**Document Control:**

| Version | Description | Date of release | Author(s) |
|---|---|---|---|
| 1.0 | Release version for evaluation | 1 April 2011 | DPR |

**GBIF Contact**

David Remsen (DR) (dremsen@gbif.org) regarding the composition and format of this document and its source files.

Markus Döring (MD) regarding additional technical details.

**Cover Art Credit**: Gregory Basco

Squirrel monkey, *Saimiri sciureus*

This document is also part of the 'GBIF Data Publishing Manual version 1.0, ISBN 87-92020-31-3, available at <http://links.gbif.org/data_publishing_manual

**About GBIF**

The Global Biodiversity Information Facility (GBIF) was established as a global mega-science initiative to address one of the great challenges of the 21st century – harnessing knowledge of the Earth's biological diversity. GBIF envisions 'a world in which biodiversity information is freely and universally available for science, society, and a sustainable future'. GBIF's mission is to be the foremost global resource for biodiversity information, and engender smart solutions for environmental and human well-being . To achieve this mission, GBIF encourages a wide variety of data publishers across the globe to discover and publish data through its network.

**Table of Contents**

# Introduction

The Darwin Core Archive (DwC-A) format is an international biodiversity data publishing standard recommended by the Global Biodiversity Information Facility. It provides a simplified method for sharing biodiversity data using basic text data files. A complete description of Darwin Core Archives can be found online at: http://links.gbif.org/gbif_dwca_how_to_guide_en_v1

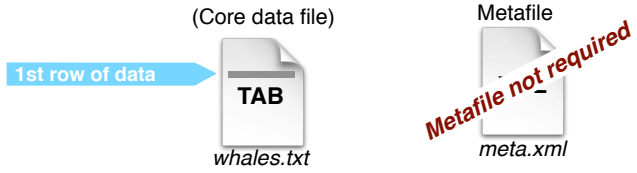
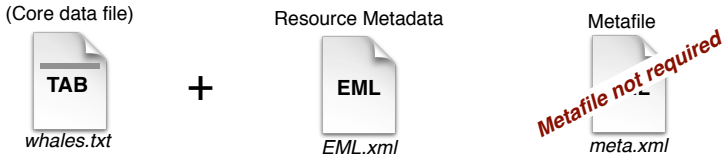
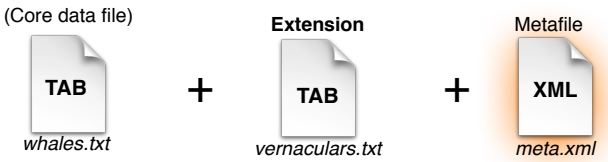
The Darwin Core Archive format relies on a special XML file, called a ***metafile*** (typically named "meta.xml"), whenever the data is published across more than one file. This file serves as a map that enables parsers and human readers to properly interpret the published files. It provides a means to link data in a particular column of a file to a specific defined term in the standard set of supported terms in the GNA Simple Exchange Format.

A biologists database, for example, may store the scientific name for a taxon in a column labeled "SCINAME." The Darwin Core provides a term called "scientificName" that is intended to store a scientific name value. The metafile provides the means to map a column in the datafile to a specific standard term, in this case the "SCINAME" column would be mapped to the *scientificName* term. This is repeated for any data element in the database that conforms to the standard terms.

This guide provides a detailed overview of the XML metafile format. It is intended for those interested in taking full advantage of the Darwin Core archive format, particularly those who may produce data files directly, without using dedicated installed software.

A metafile is not always required to be included in an archive. The following table lists the conditions where it is or is not required.

**Table 1**

| | |
|---|---|
|  <br>**Figure 1** | A metafile is not required if there is a single core data file where the first line provides the names of the Darwin Core terms represented in the published data. The column labels must exactly match the supported Darwin Core terms.<br><br>If the core data file does not contain such a "header row" then the columns names must be stored in a metafile. |
|  <br>**Figure 2** | A metafile is not required if an archive contains a resource metadata document that is named "EML.xml" If the resource metadata document is named other than "EML.xml," a metafile is required. The requirements are based on the same conditions as above concerning the presence/absence of a 'header row' in the core data file. |
|  <br>**Figure 3** | A metafile ***is required*** when one or more extensions are used to extend the data published in the core data file. The metafile provides the map that links the two data files together. |

A metafile only needs to be created once, to match the data being published. So long as the output does not change, the same metafile can be used whenever a Darwin Core Archive is updated.

### *Example Data*

The following simple example will be used to illustrate the metafile format through this document. It represents a simple species checklist stored in one table and a second table that stores common names for some of the species in the checklist. In this example, we assume the tables can be exported as basic tab-separated text files.

**Table 2**

| taxonID | kingdom | phylum | class | order | family | genus | species | authorship |
|---|---|---|---|---|---|---|---|---|
| 1 | Animalia | Chordata | Aves | Struthioniformes | Struthionidae | Struthio | camelus | Linnaeus, 1758 |
| 2 | Animalia | Chordata | Aves | Galliformes | Phasianidae | Alectoris | chukar | (A.E. Gray 1830) |
| 3 | | Chordata | Aves | Galliformes | Phasianidae | Peliperdix | coqui | (Smith, 1836) |
| 4 | Animalia | Chordata | Aves | Galliformes | Phasianidae | Dendroperdix | sephaena | A. Smith, 1836 |

**Figure 4 – Example checklist table**

The table above represents a simple de-normalised species checklist of some South African birds in a tabular format similar to what you might find in a spreadsheet or a simple database management system. The data in this table represent concepts that would be mapped to Darwin Core terms in the core data file. This file will be called “taxa.txt”

| taxonID | vernacularName | language | country |
|---|---|---|---|
| 1 | ostrich | english | US |
| 1 | volstruis | afrikaans | ZA |
| 1 | strutsi | finnish | FI |
| 2 | Asiatiese patrys | afrikaans | ZA |

**Figure 5 – Example vernacular names table**

The table above represents a second table that stores vernacular name information for some of the taxa referenced in the table above. This simple relational structure supports the publication of multiple vernacular names for a single taxon using the *taxonID* as the link (the foreign key). The data in this table conforms to terms represented by the GNA Vernacular Names extension. The corresponding file will be called “vernaculars.txt”

Assume, for this example, that we also have created a resource metadata document that we have named “EML.xml.” for a total of three files in the starting set of documents.

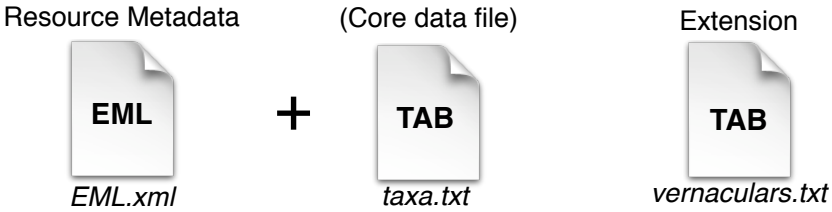



**Figure 6 - Three files in example**

A meta.xml descriptor of the example from above is illustrated below. It assumes the data files are formatted as tab-separated, UTF-8 encoded text files entitled “taxa.txt” and “vernaculasr.txt” respectively.

```
<archive xmlns="http://rs.tdwg.org/dwc/text/"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation=http://rs.tdwg.org/dwc/text/
http://rs.tdwg.org/dwc/text/tdwg_dwc_text.xsd
        metadata=”http://www.biodiv.org/docs/metadata/EML.eml”>

 <core encoding="UTF-8" fieldsTerminatedBy="\t" linesTerminatedBy="\n"
fieldsEnclosedBy=" ignoreHeaderLines="0"
rowType="http://rs.tdwg.org/dwc/terms/Taxon">
  <files>
   <location>taxa.txt</location>
    </files>

    <id index="0" />
    <field default="Animalia index="1" term="http://rs.tdwg.org/dwc/terms/kingdom"/>
    <field index="2" term="http://rs.tdwg.org/dwc/terms/phylum"/>
    <field index="3" term="http://rs.tdwg.org/dwc/terms/class"/>
    <field index="4" term="http://rs.tdwg.org/dwc/terms/order"/>
    <field index="5" term="http://rs.tdwg.org/dwc/terms/family"/>
    <field index="6" term="http://rs.tdwg.org/dwc/terms/genus"/>
    <field index="7" term="http://rs.tdwg.org/dwc/terms/species"/>
<field index="10" term="http://rs.tdwg.org/dwc/terms/scientificNameAuthorship"/>
    <field default="ICZN" term="http://rs.tdwg.org/dwc/terms/nomenclaturalCode"/>
  </core>

  <extension encoding="UTF-8" fieldsTerminatedBy="\t" linesTerminatedBy="\n"
fieldsEnclosedBy=" ignoreHeaderLines="0"
rowType="http://rs.gbif.org/terms/1.0/VernacularName">
    <files>
     <location>vernaculars.txt</location>
    </files>
    <coreid index="0" />
```

```
    <field index="1" term="http://rs.tdwg.org/dwc/terms/vernacularName"/>
    <field index="2" term="http://purl.org/dc/terms/language"/>
    <field index="3" term="http://rs.tdwg.org/dwc/terms/countryCode"/>
  </extension>
</archive>
```

**Figure 7 - XML Metafile example**

This XML file forms the basis for discussion in the following sections.

### The <archive> element

This element includes a number of attributes that are static components of the metafile. They identify namespaces and the location of the schema on which the metafile is based :

```
<archive xmlns="http://rs.tdwg.org/dwc/text/"
        xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
        xsi:schemaLocation=http://rs.tdwg.org/dwc/text/
        http://rs.tdwg.org/dwc/text/tdwg_dwc_text.xsd
        metadata="http://www.biodiv.org/docs/metadata/whale_catalogue.eml">
```

There is one important attribute included in the *<archive>* element. The **metadata** attribute stores a qualified Uniform Resource Locator (URL) defining the location of a metadata description of the entire archive. The format of the metadata is not prescribed, but a standardized format such as Ecological Metadata Language (EML), Federal Geographic Data Committee (FGDC), or ISO 19115 family is preferred.

GBIF has defined an recommends a GBIF metadata profile[1]. The GBIF Integrated Publishing Toolkit provides an interface for creating a metadata document that conforms to this schema. There is also a metadata template available with the GBIF Spreadsheet Processor service.[2] Dedicated metadata authoring tools such as Morpho[3], can also be used to author an EML metadata document.

### *The <Core> and <Extension> Elements*

The elements for describing the core data file and extension files are grouped together because they are nearly identical. An archive *must* contain exactly one <core> element that references the core data file (or files). Extensions are optional. If extensions are being used, each record in the core data must have a unique identifier, referred to using the *<id>* field. If extensions are being used, the *<extension>* element must contain a

---

[1] GNA EML Example - http://code.google.com/p/gbif-providertoolkit/source/browse/trunk/gbif-providertool/src/test/resources/dwc-archives/eml/eml.xml

[2] GBIF Spreadsheet Processor – http://tools.gbif.org/spreadsheet-processor

[3] Morpho Metadata Editor - http://knb.ecoinformatics.org/morphoportal.jsp

<*coreid*> element that indicates the column in the extension file that contains the core record identifier (the matching <*id*> in the core file). The extension itself does not have to have a unique ID field and many rows can point to the same core record.

Each block of XML within the <*core*> and <*extension*> elements is used to describe the format and content types contained in the published file. The file description can be divided into three major components described below.

1. Physical properties of the published data file (either the core data file or an extension).

```
<core encoding="UTF-8" fieldsTerminatedBy="\t" linesTerminatedBy="\n" fieldsEnclosedBy="
ignoreHeaderLines="0" rowType="http://rs.tdwg.org/dwc/terms/Taxon">
```

OR

```
<extension encoding="UTF-8" fieldsTerminatedBy="\t" linesTerminatedBy="\n"
fieldsEnclosedBy=" ignoreHeaderLines="0"
rowType="http://rs.gbif.org/terms/1.0/VernacularName">
```

The <*core*> and <*extension*> elements may contain a number of attributes which are detailed in the table below.

**Table 3**

| Attribute | Definition |
|---|---|
| encoding | The character encoding of the output text file. Recommended best practice is UTF-8. Other options include UTF-8, ISO-8859-1 (Latin-1), and Windows 1252- (WinLatin). |
| fieldsTerminatedBy | The character(s) used to separate columns. Recommended best practice is tabs "\t" or commas "," |
| linesTerminatedBy | The character(s) used to end a line, or record, of data. End of line characters vary on different operating systems[4]. Mac OS typically use a Return ("CR") character "\r" while most Unix systems output a Newline (or Line Break) "\n". Windows systems terminate lines with a combination of both "\r\n" Recommended best practice is to use a newline "\n" |
| fieldsEnclosedBy | The characters used to enclose data values. Recommended best practice is to use quotes ("") when using commas to separate columns. Tab-delimited |

[4] End of Line characters - http://en.wikipedia.org/wiki/Newline

| | |
|---|---|
| | files generally do not require enclosing quotes. |
| ignoreHeaderLines | This is a simple Boolean flag (1 or 0) that indicates if the first row in the file contains column names (a value of 1) or data values (a value of 0). A value of 1 enables file parsers to not interpret the first row as part of the published data. |
| dateFormat | Not included in the example. When verbatim dates are consistent in format, this field can be used to indicate the format represented.<br>Examples:<br>"DDMMYYYY"<br>For dates of the form 21121978<br>"DD-MM-YYYY"<br>For dates of the form 21-12-1978<br>"MMDDYYYY"<br>For dates of the form 12211978<br>"MM-DD-YYYY"<br>For dates of the form 12-21-1978<br>"YYYYMMDD"<br>For dates of the form 19781221 |
| rowType | This value refers to the unique URI that identifies the specific data type (or *class*) represented in a row of data in the specified file. For the GNA Simple Exchange Format a row of data is linked to either a Taxon in the core data file<br>http://rs.tdwg.org/dwc/terms/Taxon<br>or to one of the data types represented in the taxon-level extensions stored in the GBIF Registry (http://rs.gbif.org/extension/)<br>This includes data types such as<br>http://rs.gbif.org/terms/1.0/VernacularName<br>http://rs.gbif.org/terms/1.0/Reference<br>http://rs.gbif.org/terms/1.0/Specimens<br>http://rs.gbif.org/terms/1.0/Identifier<br>http://rs.gbif.org/terms/1.0/Description<br>http://rs.gbif.org/terms/1.0/Distribution |

2. Identification of core data file or files.

```
<files>
    <location>taxa.txt</location>
 </files>
```

The <*files*> element declares the name of the core data file. The <*core*> or <*extension*> elements must contain one or more <*files*> elements to locate the data being described. If multiple files are used, multiple location elements list the individual file names.

Note that a location is generally a file or files that are local to the metafile (they exist in the same directory or zipped archive) and are referenced by a simple filename or relative filepath ("data/taxon.txt"). A location, however, can also be a web-accessible URL such as "http://www.gbif.org/data/specimen.csv" or "ftp://ftp.gbif.org/tim/specimen.txt". Best practice is to include all files in the archive.

3. Description of specific data columns (or *fields*) in each file using the <*fields*> tag.

```
<id index="0" />
<field index="1" term="http://rs.tdwg.org/dwc/terms/kingdom"/>
<field index="2" term="http://rs.tdwg.org/dwc/terms/phylum"/>
```

The remainder of the core or extension element provides a listing of the data elements published in the core data file as data fields. As noted above, if extensions are being used the core data file must contain an <*id*> element to indicate a unique identifier. It is recommended that this column be the first column (*index=0*) in a data file. In addition, each extension must identify a <*coreId*> column that acts as a foreign key to the matching <*id*> element in the core. It is also recommended that this column be the first column (*index=0*) in a extension file.

A <*core*> or <*extension*> element must contain one or more <*field*> elements, each representing a 'column' in a row of data.

A data record is divided into fields, based on the value of the *fieldsTerminatedBy* property. In the core data example above, the 11 data elements in the table are output as 11 fields, separated by a tab. The unique identifier would be contained in the <*id*> element and the remaining 10 fields would be contained in 10 separate <field> elements.

A <field> element specifies the location and content of a data field and provides the following 4 attributes.

**Table 4**

| Attribute | Definition |
|---|---|
| index | Specifies the relative column position of the field in a row of data. Numbering |

| | |
|---|---|
| | starts with the number, 0.<br><br>The numbering continues from 0 so that the 2nd physical column in the file is referenced as "index=1." In the core data example, this column stores the taxonomic Kingdom values and is mapped to the Darwin Core term, kingdom.<br><field index="1" term="http://rs.tdwg.org/dwc/terms/kingdom"/><br>This continues for all of the columns in the core data or extension file. If the files contain data that do not conform to the supported standard but are nonetheless part of the published file, they can be included but not explicitly referenced simply by omitting a <field> element for the column specified by the corresponding index number. |
| term | A Unified Resource Identifier (URI) for the term represented by this field. For example, a field containing the scientific name would have term="http://rs.tdwg.org/dwc/terms/scientificName". The complete set of terms supported in the GNA Simple Exchange Format can be found in the document **"GBIF Global Names Architecture Simple Exchange Format: Core Terms and Extensions[5]"** |
| type | Specifies the type of the data content in the column. This is restricted to any simple type (xs:integer, xs:nonNegativeInteger, xs:date, etc.; see guidelines for field types). Default is "string" |
| default | Specifies value to use if one is not supplied for the field in a given row. If no index is supplied, the default can be used to define a constant for all rows for a field that is not in the data file. |

### *Static Mappings*

Some terms in a dataset may be the same for all rows in the data. In the example above, all taxa are animals that share the same Kingdom, "*Animalia*". Note that record #3 is missing a value "Animalia." A default value can be set for this row. The default value can be inserted for any blank value in the specified column.

<field default="Animalia" index="1" term="http://rs.tdwg.org/dwc/terms/kingdom"/>

---

[5] GNA Simple Exchange Format: Core Terms and Extensions: http://code.google.com/p/gbif-ecat/downloads/detail?name=GNA-Terms-Extensions.pdf

A global value may also be specified. Global values do not contain a column number. Instead, they effectively add a new column into the output file that is not contained explicitly in the source data but may be useful for users. For example, since all the taxa in the example are animal names, they all fall under the jurisdiction of the zoological code of nomenclature governed by the "International Commission of Zoological Nomenclature" ("*ICZN* "for short). Instead of repeating the value "ICZN" for each record, a global value is set and can be applied to each record whenever the data is parsed and used.

```
<field default="ICZN" term="http://rs.tdwg.org/dwc/terms/nomenclaturalCode"/>
```

Similarly, default values can be applied to extension files. For example, a list of vernacular names used in a specific country may provide a default value for the country code term instead of listing the same value in each record. The format is exactly the same as a default recorded in the core data file.

### *Variables in Static Mappings*

An advanced use of static mappings allows the insertion of variables in the default value rather than static text such as "ICZN" from the example above. A common use for such variables is to compose a link (a URL) to a web page or web service call that uses the taxon identifier or the taxon name as one of the parameters in the URL. Any column in the published data can be referenced by enclosing the index number in curly braces "{}". The taxon identifier in the core data file can also be referenced with the via the variable "{id}."

1. The Integrated Taxonomic Information System (ITIS) uses Taxonomic Serial Numbers (TSN) to provide links to taxon pages on its web site.
http://www.itis.gov/servlet/SingleRpt/SingleRpt?search_topic=TSN&search_value=174375
If a core data file is published using the ITIS TSN system a link can be composed and tied to the "identifier" term in the core data standard using the following syntax.

```
# a record id based link to the species page:
<field
default="http://www.itis.gov/servlet/SingleRpt/SingleRpt?search_topic=TSN&search_valu
e={id}" term="http://purl.org/dc/terms/identifier"/>
```

where the original numeric value is replaced by the variable “{id}”. This value would be derived from the core ID.

2. The 2010 Catalogue of Life Annual Checklist provides similar identifiers. It also supports name-based searches that can also be encoded as URLs. For example,
http://www.catalogueoflife.org/annual-checklist/2010/search/all/key/Struthio+camelus/match/1
embeds a scientific name “Struthio camelus” into a URL. Full scientific name combinations can be published in the core data file using the Darwin Core term “scientificName.” If we assume that this term represented the 12th column in our core data file we could use the syntax

```
# a record id based link to the species page:
<field default="http://www.catalogueoflife.org/annual-checklist/2010/search/all/key/{12}/match/1"
term="http://purl.org/dc/terms/identifier"/>
```

where {12} represents the 12th column value that will be substituted in the URL.

## Creating a metafile

The metafile that documents the data files in a Darwin Core Archive must

- Accurately and completely represent the data that a data publisher wishes to document
- Must comply with the metafile XML schema[6] (included as an Annex to this document).

There are three ways a metafile can be created

1. A metafile can be hand-coded, using the examples provided in this document as a template, or composed using custom software. This method, however, is not recommended or necessary.

2. The GBIF Integrated Publishing Toolkit outputs Darwin Core Archive files as a default feature, automatically generating a metafile as part of the publication process.

3. GBIF provides a Darwin Core Archive Assistant that greatly simplifies the creation of a metafile by providing a service for generating the metafile. See http://tools.gbif.org/dwca-assistant/

## Validating a metafile.

The best way to validate a metafile is to use the Darwin Core Archive Validator at http://tools.gbif.org/dwca-validator/
There are several options for validating an XML metafile. A local solution exists using XML editors such as Jedit[7], oXygen[8], XML Spy[9] or similar tools.

---

[6] Metafile XML Schema - http://darwincore.googlecode.com/svn/trunk/text/tdwg_dwc_text.xsd

[7] Jedit - http://www.jedit.org/index.php?page=download

[8] Oxygen - http://www.oxygenxml.com/download.html

[9] xml Spy - http://www.altova.com/xml-editor/

## Annex A – the metafile XML Schema

http://darwincore.googlecode.com/svn/trunk/text/tdwg_dwc_text.xsd

```
<?xml version='1.0' encoding='utf-8'?>
<xs:schema version="0.1"
        xmlns:xs="http://www.w3.org/2001/XMLSchema"
        xmlns:arch="http://rs.tdwg.org/dwc/text/"
        xmlns:dwc="http://rs.tdwg.org/dwc/terms/"
    targetNamespace="http://rs.tdwg.org/dwc/text/"
    attributeFormDefault="unqualified"
    elementFormDefault="qualified">
        <xs:import namespace="http://www.w3.org/2001/XMLSchema"
schemaLocation="http://www.w3.org/2001/XMLSchema.xsd"/>
        <xs:import namespace="http://rs.tdwg.org/dwc/terms/"
schemaLocation="http://darwincore.googlecode.com/svn/trunk/xsd/tdwg_dwcterms.xsd"
/>

        <!-- The root element of the document is an archive -->
        <xs:element name="archive">
                <xs:complexType>
                        <xs:sequence>
                                <xs:element name="core" type="arch:coreFileType"
maxOccurs="1" minOccurs="1"/>
                                <xs:element name="extension" type="arch:extensionFileType"
maxOccurs="unbounded" minOccurs="0"/>
                        </xs:sequence>
                        <xs:attribute name="metadata" type="xs:anyURI" use="optional"/>
                </xs:complexType>
        </xs:element>

        <!-- attributes shared across all file types, core or extensions -->
        <xs:attributeGroup name="fileAttributes">
                <xs:attribute name="linesTerminatedBy" type="xs:string" use="optional"
default="\n"/>
```

```
            <xs:attribute name="fieldsTerminatedBy" type="xs:string" use="optional"
default=","/>
            <xs:attribute name="fieldsEnclosedBy" type="xs:string" use="optional"
default=""/>
            <xs:attribute name="ignoreHeaderLines" type="xs:integer" use="optional"
default="0"/>
            <xs:attribute name="rowType" type="xs:string" use="optional"
default="http://rs.tdwg.org/dwc/xsd/simpledarwincore/SimpleDarwinRecord"/>
            <xs:attribute name="encoding" type="arch:encodingEnum" use="optional"
default="ISO-8859-1"/>
            <xs:attribute name="compression" type="arch:compressionEnum"
use="optional"/>
            <xs:attribute name="dateFormat" type="xs:string" use="optional"/>
        </xs:attributeGroup>

        <!-- The file within an archive defines the description and it's fields -->
        <xs:complexType name="fileType">
            <xs:sequence>
                <xs:element name="files" minOccurs="1" maxOccurs="1">
                    <xs:complexType>
                        <xs:sequence>
                            <xs:element name="location" type="xs:string"
minOccurs="1" maxOccurs="unbounded"/>
                        </xs:sequence>
                    </xs:complexType>
                </xs:element>
            </xs:sequence>
            <xs:attributeGroup ref="arch:fileAttributes"/>
        </xs:complexType>

        <!-- a file representing the core file in a star schema -->
        <xs:complexType name="coreFileType">
            <xs:complexContent>
                <xs:extension base="arch:fileType">
                    <xs:sequence>
                        <xs:element name="id" type="arch:idFieldType"
minOccurs="0" maxOccurs="1"/>
```

```
                                        <xs:element name="field" type="arch:fieldType"
minOccurs="0" maxOccurs="unbounded"/>
                                    </xs:sequence>
                            </xs:extension>
                    </xs:complexContent>
            </xs:complexType>

            <!-- a file representing an extension file in a star schema -->
            <xs:complexType name="extensionFileType">
                    <xs:complexContent>
                            <xs:extension base="arch:fileType">
                                    <xs:sequence>
                                        <xs:element name="coreid" type="arch:idFieldType"
minOccurs="1" maxOccurs="1"/>
                                        <xs:element name="field" type="arch:fieldType"
minOccurs="1" maxOccurs="unbounded"/>
                                    </xs:sequence>
                            </xs:extension>
                    </xs:complexContent>
            </xs:complexType>

            <!-- A field represents a column within the file -->
            <xs:complexType name="idFieldType">
                    <xs:attribute name="index" type="xs:integer" use="optional"/>
            </xs:complexType>

            <!-- A field represents a column within the file -->
            <xs:complexType name="fieldType">
                    <xs:attribute name="index" type="xs:integer" use="optional"/>
                    <xs:attribute name="term" type="xs:anyURI" use="required"/>
                    <xs:attribute name="type" type="xs:anySimpleType" use="optional"
default="xs:string"/>
                    <xs:attribute name="default" type="xs:string" use="optional"/>
            </xs:complexType>

            <!-- Enumeration for supported compression types -->
            <xs:simpleType name="compressionEnum">
```

```
            <xs:restriction base="xs:string">
                <xs:enumeration value="GZIP"/>
                <xs:enumeration value="ZIP"/>
            </xs:restriction>
        </xs:simpleType>

        <!-- Enumeration for supported encodings -->
        <xs:simpleType name="encodingEnum">
            <xs:restriction base="xs:string">
                <xs:enumeration value="windows-1252"/>
                <xs:enumeration value="ISO-8859-1"/>
                <xs:enumeration value="UTF-8"/>
                <xs:enumeration value="UTF-16"/>
            </xs:restriction>
        </xs:simpleType>
</xs:schema>
```