

eCH-0165 SIARD-Formatspezifikation

Name	SIARD-Formatspezifikation
eCH-Nummer	eCH-0165
Kategorie	Standard
Reifegrad	Experimentell
Version	2.0
Status	Genehmigt
Genehmigt am	2016-06-01
Ausgabedatum	2016-06-15
Ersetzt Version	1.0
Voraussetzungen	-
Beilagen	metadata.xsd
Sprachen	Deutsch (Original), Französisch (Uebersetzung), Englisch (Uebersetzung)
Autoren	<p>Luis Faria, KEEP SOLUTIONS, LDA, lfaria@keep.pt</p> <p>Anders Bo Nielsen, Danish National Archives (Rigsarkivet), abn@sa.dk</p> <p>Krystyna Ohnesorge, Schweizerisches Bundesarchiv, krystyna.ohnesorge@bar.admin.ch</p> <p>Claire Röthlisberger-Jourdan, KOST, claire.roethlisberger@kost.admin.ch</p> <p>Hartwig Thomas, Enter AG, hartwig.thomas@enterag.ch</p> <p>Andreas Voss †, Schweizerisches Bundesarchiv, andreas.voss@bar.admin.ch</p>
Mitarbeitende	<p>Karin Bredenberg, National Archives of Sweden, karin.bredenberg@riksarkivet.se</p> <p>Hedi Bruggisser, Staatsarchiv Thurgau, hedi.bruggisser@tg.ch</p> <p>Georg Büchler, KOST, georg.buechler@kost.admin.ch</p> <p>Janet Delve, University of Portsmouth, janet.delve@port.ac.uk</p>

	<p>Boris Domajnko, Slovenian National Archives, boris.domajnko@gov.si</p> <p>Alain Dubois, Staatsarchiv Wallis, alain.dubois@admin.vs.ch</p> <p>Bruno Ferreira, KEEP SOLUTIONS, LDA, bferreira@keep.pt</p> <p>Arne-Kristian Groven, National Archives Norway (Riksarkivet), arngro@arkivverket.no</p> <p>Martin Kaiser, KOST, martin.kaiser@kost.admin.ch</p> <p>Lambert Kansy, Staatsarchiv Basel Stadt, lambert.kansy@bs.ch</p> <p>Markus Lischer, Staatsarchiv Luzern, markus.lischer@lu.ch</p> <p>Zoltán Lux, National Archives of Hungary, lux.zoltan@mnl.gov.hu</p> <p>Lauri Rätsep, National Archives of Estonia, lauri.ratsep@ra.ee</p> <p>Hélder Silva, KEEP SOLUTIONS, LDA, hsilva@keep.pt</p>
Herausgeber / Vertrieb	<p>Verein eCH, Mainaustrasse 30, Postfach, 8034 Zürich T 044 388 74 64, F 044 388 71 80 www.ech.ch / info@ech.ch</p>

Zusammenfassung

Dieses Dokument enthält die Spezifikation des SIARD-Dateiformats, Version 2.0. SIARD steht für *Software-Independent Archival of Relational Databases*. Das Format wurde vom Schweizerischen Bundesarchiv entwickelt. Es handelt sich um eine normative Beschreibung eines Dateiformats für die langfristige Erhaltung von relationalen Datenbanken.

Das SIARD-Format basiert auf Standards – u. a. auf den ISO-Normen Unicode, XML und SQL:2008, dem Internetstandard URI und dem Industriestandard ZIP. Die Verwendung international anerkannter Standards zielt darauf hin, die langfristige Erhaltung von und den Zugang zu dem weitverbreiteten relationalen Datenbankmodell zu gewährleisten sowie den einfachen Austausch von Datenbankinhalten zu ermöglichen, unabhängig von proprietären Dump-Formaten.

Inhaltsverzeichnis

1	Status des Dokuments	6
2	Einleitung	6
2.1	Struktur des Dokuments	6
2.1.1	Aufbau Kapitel	6
2.1.2	ID Anforderungen	7
2.1.3	Unterscheidung zwischen Muss- und Kann-Anforderungen.....	7
2.1.4	Notation Ordner, Dateien und Ordnerstrukturen	7
2.2	Adressaten/Zielgruppe.....	8
2.3	Ausgangslage	8
2.4	Abgrenzungen	8
3	Allgemeine Anforderungen / Grundsätze	10
3.1	Verwendung von Standards.....	10
3.2	Datenbanken als Unterlagen	10
3.3	Zeichensätze und Zeichen.....	10
3.4	File-URI-Schema	11
3.5	Bezeichner und reguläre Bezeichner	11
4	Anforderungen an die Formatstruktur	13
4.1	Aufbau der SIARD-Archivdatei.....	13
4.2	Struktur der SIARD-Archivdatei	13
4.3	Korrespondenz zwischen Metadaten und Tabellendaten	16
5	Anforderungen an die Metadaten	21
5.1	Metadaten auf der Ebene Datenbank	21
5.2	Metadaten auf der Ebene Schema	23
5.3	Metadaten auf der Ebene Type.....	23
5.4	Metadaten auf der Ebene Attribut	24
5.5	Metadaten auf der Ebene Tabelle	25
5.6	Metadaten auf der Ebene Spalte	25
5.7	Metadaten für Felder	27
5.8	Metadaten des Primärschlüssels	29
5.9	Metadaten der Fremdschlüssel	29
5.10	Referenz-Metadaten	30

5.11 Metadaten des Kandidatenschlüssels	30
5.12 Metadaten der Check-Einschränkung	30
5.13 Metadaten auf der Ebene Trigger	31
5.14 Metadaten auf der Ebene View	31
5.15 Metadaten auf der Ebene Routine	32
5.16 Metadaten der Parameter	33
5.17 Metadaten von ParameterField	33
5.18 Metadaten auf der Ebene des Benutzers	34
5.19 Metadaten auf der Ebene Rolle	34
5.20 Metadaten auf der Ebene der Privilegien	35
6 Anforderungen an die Tabellendaten	36
6.1 Tabellen-Schemadefinition	36
6.2 <i>Large-Object</i> -Datenzellen	37
6.3 Datums- und Timestamp-Datenzellen	38
6.4 Tabellendaten	38
7 Version und Gültigkeit der Spezifikation	41
8 Change-Management-Prozess	41
9 Haftungsausschluss/Hinweise auf Rechte Dritter	41
10 Urheberrechte	42
Anhang A – Mitarbeit & Überprüfung	43
Anhang B – Abkürzungen und Glossar	44
Anhang C – Nachweis der verwendeten Standards	46
Anhang D – XML-Schemadefinitionen	46
D.1 metadata.xsd	46
D.2 Beispiel für metadata.xml	62
D.3 Beispiel für die XML-Schemadefinition einer Tabelle: table0.xsd	71
D.4 Beispiel für die Tabellendaten einer Tabelle: table0.xml	73
D.5 Beispiel für die XML-Schemadefinition einer Tabelle mit fortgeschrittenen und strukturierten Datentypen: table1.xsd	74
D.6 Beispiel für die Tabellendaten einer Tabelle mit fortgeschrittenen und strukturierten Datentypen: table1.xml	76
Anhang E – Änderungen gegenüber Version 1.0	77

1 Status des Dokuments

Genehmigt: Das Dokument wurde vom Expertenausschuss genehmigt. Es hat für das definierte Einsatzgebiet im festgelegten Gültigkeitsbereich normative Kraft.

2 Einleitung

2.1 Struktur des Dokuments

2.1.1 Aufbau Kapitel

Jedes Kapitel in dieser Spezifikation ist nach demselben Muster aufgebaut. Nach einer kurzen Einleitung werden die Anforderungen in einer Tabelle aufgeführt.

ID	Beschreibung Anforderung	M/K
Enthält die ID der Anforderung	Enthält den Anforderungstext	Definiert, ob es sich um eine Muss- oder Kann-Anforderung handelt

Eine Anforderung wird häufig durch Empfehlungen, Hinweise und Beispiele erläutert. Empfehlungen, Hinweise und Beispiele sind speziell gekennzeichnet.

ID	Beschreibung Anforderung	M/K
G_3.1-1	<p>Anforderungstext</p> <p>Beispiel Beispieltext</p> <p>Hinweis Hinweistext</p> <p><i>Empfehlung</i> <i>Empfehlungstext ist kursiv gesetzt.</i></p>	M

2.1.2 ID Anforderungen

Die Anforderungen sind über eine ID eindeutig identifizierbar.

ID
G_3.1-1

Diese ID ist nach dem folgenden Muster aufgebaut:

G_	Buchstabe + _ identifiziert Hauptkapitel
G_	= Grundsätze / Allgemeine Anforderungen
T_	= Anforderungen an die Tabellendaten
M_	= Anforderungen an die Metadaten
P_	= Anforderungen an die Paketstruktur
3.1-1	Die Nummer beginnt mit der Angabe des Kapitels (Gruppierung der Anforderungen zum gleichen Thema), die Zahl hinter dem Bindestrich wird durchnummeriert und kennzeichnet so alle Anforderungen des Kapitels.

2.1.3 Unterscheidung zwischen Muss- und Kann-Anforderungen

Jede Anforderung ist entweder eine Muss- oder eine Kann-Anforderung. Dies wird mit einem Buchstaben kenntlich gemacht, der auf die Verbindlichkeit verweist:

Abkürzung	Bedeutung
M	Muss-Anforderung Diese Anforderung muss erfüllt sein, um eine gültige SIARD-Datei zu erhalten.
K	Kann-Anforderung Diese Anforderung sollte erfüllt sein. Sie vereinfacht das Handling im Sinne von <i>Best Practice</i> .

2.1.4 Notation Ordner, Dateien und Ordnerstrukturen

Für die Notation von Ordnern, Dateien etc. werden die folgenden Symbole und Parameter verwendet.

Symbol	Bedeutung
/	Ordner
header/	Ein Ordner mit dem Namen «header»
xy.txt	Datei (mit Datei-Endung «txt»)
dir1/	Beispiel-Ordner (in roter Farbe)
abc.pdf	Beispiel-Dateien (in roter Farbe)
...	Platzhalter für Dateien oder Ordner, die für die Erklärung nicht relevant sind.

Symbol	Bedeutung
[]	Platzhalter für einen Ausdruck oder einen Basistyp wie «string», «integer» etc.
<xx>	Platzhalter für beliebige Zeichenkette

2.2 Adressaten/Zielgruppe

Dies ist ein technisches Dokument für IT-Spezialisten, die im Bereich der dauerhaften Archivierung von relationalen Datenbanken tätig sind.

2.3 Ausgangslage

Die Bezeichnung SIARD steht für Software-Independent Archival of Relational Databases (engl. für „software-unabhängige Archivierung von relationalen Datenbanken“). Es handelt sich um ein offenes Dateiformat zur dauerhaften Archivierung von relationalen Datenbanken in Form von Textdaten, basierend auf XML, die in eine Containerdatei (SIARD-Archiv) gepackt werden¹.

Dauerhafte Archivierung meint die grundsätzlich unbegrenzte Aufbewahrung der in SIARD-Dateien gespeicherten Informationen unter Erhalt des Bitstroms sowie der Fähigkeit, die Daten menschenlesbar und verständlich zu interpretieren und darzustellen.

Wenn Struktur und Inhalt einer relationalen Datenbank ins SIARD-Format übersetzt werden, wird es später jederzeit möglich sein, auf die Daten der Datenbank zuzugreifen oder diese auszutauschen, selbst wenn die ursprüngliche Datenbanksoftware nicht mehr verfügbar oder nicht mehr lauffähig sein wird. Dies wird erreicht, indem für das SIARD-Format geeignete Standards verwendet wurden, die international breit abgestützt sind. Diese langfristige Interpretierbarkeit der Datenbankinhalte beruht im Wesentlichen auf den beiden Standards XML und SQL:2008.

2.4 Abgrenzungen

Es ist festzuhalten, dass das SIARD-Format nur das Langzeitspeicherformat für eine spezielle Sorte von digitalen Unterlagen (relationale Datenbanken) darstellt und somit völlig unabhängig von Paketstrukturen wie SIP (Submission Information Package), AIP (Archival Information Package) und DIP (Dissemination Information Package) des OAIS-Modells konzipiert ist².

¹ Das Datenbank-Archivierungsformat SIARD ist zu unterscheiden von der Applikation SIARD Suite. Diese wurde vom Schweizerischen Bundesarchiv BAR entwickelt, um SIARD-Dateien zu erzeugen, zu editieren und wieder in Datenbankumgebungen zu importieren.

² <http://public.ccsds.org/publications/archive/650x0m2.pdf>

Es wird davon ausgegangen, dass eine Datenbank im SIARD-Format als Teil eines solchen Informationspakets zusammen mit anderen Unterlagen (ausgelagerte *Large Object Files*, Übersetzungstabelle für externe Dateinamen, Datenbank-Dokumentation, für das Verständnis der Datenbank relevante Geschäftsunterlagen, ...) archiviert wird.

Ähnlich wie eine XML-basierte Word- oder Mail-Datei eine interne Dateistruktur mit Metadaten, Primärdaten und verschiedenen Hilfsdaten enthält, enthält auch eine archivierte relationale Datenbank im SIARD-Format neben den eigentlichen Tabellendaten auch eigene Metadaten, welche die Unterlage näher beschreiben – ohne Rücksicht auf den Metadatenkatalog, den ein Archiv in seinen OAIS-Paketen erfasst.

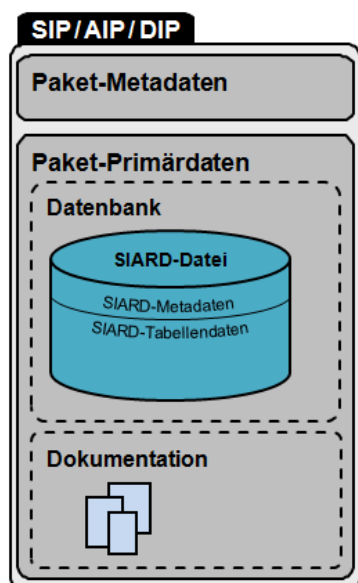


Abb. 1: Symbolbild eines Informationspakets mit einer enthaltenen SIARD-Datei

3 Allgemeine Anforderungen / Grundsätze

3.1 Verwendung von Standards

Um die Interpretierbarkeit der Datenbankinhalte über lange Zeiträume zu gewährleisten, beruht das SIARD-Format im Wesentlichen auf den beiden ISO-Standards XML sowie SQL:2008.

ID	Beschreibung Anforderung	M/K
G_3.1-1	Sämtliche Datenbankinhalte werden in einer Kollektion von XML-Dateien gemäss ISO/IEC 19503:2005 gespeichert. Schemadefinitionen und SQL-Code müssen jeweils SQL:2008-konform sein gemäss ISO/IEC 9075. Einzige Ausnahme sind BLOB- und CLOB-Daten (Binary Large Objects und Character Large Objects), die in separaten binären oder Text-Dateien gespeichert, aber in den XML-Dateien referenziert werden.	M

3.2 Datenbanken als Unterlagen

Eine relationale Datenbank wird wie eine einzige zu archivierende Unterlage behandelt, damit die Bezüge (Referenzen) zwischen den Daten einzelner Tabellen erhalten bleiben.

ID	Beschreibung Anforderung	M/K
G_3.2-1	Eine relationale Datenbank wird in einer einzigen SIARD-Datei archiviert.	M
G_3.2-2	Die Primärdaten einer relationalen Datenbank werden vollständig in einer SIARD-Datei archiviert. Das heisst, dass jede SELECT-Abfrage auf den originalen und auf den archivierten Daten dieselben Resultate ergibt ³ .	M

3.3 Zeichensätze und Zeichen

ID	Beschreibung Anforderung	M/K
G_3.3-1	Alle Daten werden im Unicode-Zeichensatz gemäss ISO 10646 gespeichert.	M
G_3.3-2	Beim Extrahieren aus Datenbanken, welche andere Zeichensätze unterstützen, wird die Abbildung in die entsprechenden Unicode-Zeichensätze vorgenommen. Aus diesem Grund müssen die nationalen Zeichenketten-Typen (NCHAR, NVARCHAR, NCLOB) aus dem Datenbank-Produkt generell in nicht-nationale (CHAR, VARCHAR bzw. CLOB) übersetzt werden. Diese Konvention wird von XML unterstützt, unabhängig davon, ob eine XML-Datei im UTF-8-Format oder im UTF-16-Format gespeichert wird.	M
G_3.3-3	In den XML-Dateien des SIARD-Formats werden alle Zeichen, welche in der XML-Syntax eine spezielle Bedeutung haben, durch Einheitenreferenzen ersetzt und zwar in allen Feldern vom Typ xs:string. Zusätzlich werden die Unicode-Steuerzeichen 0-31 und 127-159 mit Hilfe des Solidus ("") codiert, damit die Gültigkeit der XML-Datei garantiert bleibt.	M

³ Für Details siehe http://www.enterag.ch/hartwig/SIARD_Criterion.pdf.

ID	Beschreibung Anforderung	M/K																						
G_3.3-4	<p>Zeichen, die nicht in UNICODE dargestellt werden können (Codes 0-8, 14-31, 127-159), sowie das Escapezeichen '\' und mehrere aufeinanderfolgende Leer-schläge werden mit Escape als \u00<xx> in XML dargestellt. Anführungszeichen, Kleiner und Et-Zeichen werden in XML als Einheitsreferenzen dargestellt.</p> <table><tr><th>Ursprüngliche Zeichen</th><th>Zeichen im SIARD- Format</th></tr><tr><td>0 bis 8</td><td>\u0000 bis \u0008</td></tr><tr><td>14-31</td><td>\u000E bis \u001F</td></tr><tr><td>32</td><td>\u0020, falls mehrere aufeinanderfolgen</td></tr><tr><td>"</td><td>&quot;</td></tr><tr><td>&</td><td>&amp;</td></tr><tr><td>'</td><td>&apos;</td></tr><tr><td><</td><td>&lt;</td></tr><tr><td>></td><td>&gt;</td></tr><tr><td>\</td><td>\u005c</td></tr><tr><td>127 bis 159</td><td>\u007F bis \u009F</td></tr></table>	Ursprüngliche Zeichen	Zeichen im SIARD- Format	0 bis 8	\u0000 bis \u0008	14-31	\u000E bis \u001F	32	\u0020, falls mehrere aufeinanderfolgen	"	"	&	&	'	'	<	<	>	>	\	\u005c	127 bis 159	\u007F bis \u009F	M
Ursprüngliche Zeichen	Zeichen im SIARD- Format																							
0 bis 8	\u0000 bis \u0008																							
14-31	\u000E bis \u001F																							
32	\u0020, falls mehrere aufeinanderfolgen																							
"	"																							
&	&																							
'	'																							
<	<																							
>	>																							
\	\u005c																							
127 bis 159	\u007F bis \u009F																							

3.4 File-URI-Schema

Zur Referenzierung von ausgelagerten *Large Objects* wird das File-URI-Schema gemäss RFC 1738 verwendet⁴.

ID	Beschreibung Anforderung	M/K
G_3.4-1	Alle ausgelagerten Dateien werden mit einer File-URI gemäss RFC 1738 spezi-fiziert.	M
G_3.4-2	File-URI werden in einer SIARD-Datei als URL-codierte ASCII-Strings gespei-chert.	M
G_3.4-3	Es wird empfohlen, für die File-URI ein Dateisystem zu verwenden, welches die direkte Adressierung individueller Dateieinträge innerhalb einer ZIP-Datei er-möglicht. Beispielsweise würde <i>file:///d:/sips/sip1234.zip</i> auf die ZIP-Datei ver-weisen, während <i>file:///d:/sips/sip1234.zip/</i> auf den Stammordner innerhalb der ZIP-Datei verweist.	K

3.5 Bezeichner und reguläre Bezeichner

In SQL:2008 gibt es reguläre Bezeichner⁵ ohne Leerschläge und Sonderzeichen, für welche Gross- und Kleinschreibung unwichtig ist, die aber in der SIARD-Datei in Grossbuchstaben

⁴ http://en.wikipedia.org/wiki/File_URI_scheme , <http://tools.ietf.org/html/rfc1738>.

⁵ „Regulärer Bezeichner“, engl.: *identifier* . Ein SQL:2008-Bezeichner muss mit einem Buchstaben (A-Z) oder dem Tiefstrich (_) beginnen, gefolgt von Buchstaben (A-Z), Ziffern (0-9) oder Tiefstrich (_), maximal 128 Zeichen.

gespeichert werden, und Bezeichner in Anführungszeichen⁶, für welche die Schreibweise eindeutig ist, und die auch Sonderzeichen enthalten oder mit einem SQL-Schlüsselwort identisch sein dürfen. Diese müssen in Ausdrücken von doppelten Anführungszeichen umrahmt werden. In der SIARD-Datei werden sie ohne die Anführungszeichen gespeichert.

Was ein Sonderzeichen ist, bestimmt der SQL-Standard. Was die Grossbuchstabenversion eines Buchstabens ist, wird vom Unicode-Standard bestimmt.

In den Metadaten wird ein regulärer Bezeichner in Grossbuchstaben gespeichert, während ein begrenzter (delimitierter) Bezeichner ohne Anführungszeichen gespeichert wird. Der SQL:2008-Standard hält fest, dass ein Bezeichner als begrenzter Bezeichner gelten soll, falls er ein Zeichen enthält, das er als regulärer Bezeichner nicht enthalten darf, oder falls er mit einem SQL-Schlüsselwort identisch ist.

ID	Beschreibung Anforderung	M/K
G_3.5-1	Alle Bezeichner werden im Unicode-Zeichensatz gespeichert.	M
G_3.5-2	Reguläre Bezeichner sind in Grossbuchstaben und ohne Anführungszeichen.	M
G_3.5-3	Begrenzte (delimitierte) Bezeichner werden ohne Anführungszeichen gespeichert.	M

⁶ „Bezeichner in Anführungszeichen“ bzw. „begrenzter (delimitierter) Bezeichner“, engl.: *delimited identifier*.

4 Anforderungen an die Formatstruktur

4.1 Aufbau der SIARD-Archivdatei

Die SIARD-Archivdatei wird als ZIP-Archiv realisiert.

ID	Beschreibung Anforderung	M/K
G_4.1-1	Die SIARD-Datei wird als ein einziges ZIP-Archiv gemäss der von der Firma PkWare publizierten Spezifikation, Version 6.3.2 gespeichert ⁷ .	M
G_4.1-2	SIARD-Dateien müssen entweder unkomprimiert oder mit dem Deflate-Algorithmus gemäss RFC 1951 ⁸ komprimiert sein.	
G_4.1-3	Die SIARD-Datei ist nicht passwortgeschützt oder verschlüsselt.	M
G_4.1-4	Für das ZIP-Archiv sind beide Ausprägungen erlaubt, ZIP32 und ZIP64.	M
G_4.1-5	Das ZIP-Archiv hat die Dateierweiterung ".siard".	M

4.2 Struktur der SIARD-Archivdatei

Eine im SIARD-Format archivierte relationale Datenbank besteht aus zwei Komponenten: den Metadaten, welche die Struktur der archivierten Datenbank beschreiben, und den Tabellendaten, welche die Tabelleninhalte repräsentieren. Die Metadaten geben weiterhin an, welche Tabellendaten wo im Archiv zu finden sind.

ID	Beschreibung Anforderung	M/K
P_4.2-1	<p>Die Tabellendaten befinden sich im Ordner <code>content/</code> und die Metadaten im Ordner <code>header/</code>. Weitere Ordner oder Dateien sind nicht erlaubt.</p> <p>Beispiel Aufbau der SIARD-Datei (schematisch)</p> <pre> Northwind.siard content/ header/ </pre>	M

⁷ ZIP-Dateien wurden ursprünglich von Phil Katz definiert und sind heute als De-facto-Standard sehr weit verbreitet. Die aktuelle Version 6.3.2 der von der Firma PkWare publizierten Spezifikation findet man unter <http://www.pkware.com/documents/casestudies/APPNOTE.TXT>.

⁸ <https://www.ietf.org/rfc/rfc1951.txt>.


ID	Beschreibung Anforderung	M/K
P_4.2-2	<p>Im Ordner <code>content/</code> befinden sich ein oder mehrere Schema-Ordner, welche die einzelnen Tabellen-Ordner enthalten. Andere Ordner oder Dateien sind nicht erlaubt.</p> <p>Beispiel Aufbau der SIARD-Datei (schematisch)</p> <pre> Northwind.siard content/ schema0/ table0/ table1/ table2/ ... schema1/ table0/ ... </pre> <p>Empfehlung <i>Es wird empfohlen, die Namen der Schema- und Tabellenordner zu normalisieren und anstelle des eigentlichen Namen z.B. <code>schema0/</code> und <code>table0/</code> zu verwenden (siehe Einschränkungen unter P_4.2-6).</i></p>	M
P_4.2-3	<p>In den einzelnen Tabellen-Ordnern sind eine XML- und eine XSD-Datei enthalten, wobei die Namen (Ordnerbezeichnung und beide Dateinamen) identisch sein müssen. Weitere Ordner oder Dateien sind mit Ausnahme von BLOB- und CLOB-Ordern samt deren Inhalt (BIN-oder TXT-Dateien) nicht erlaubt.</p> <p>Beispiel Aufbau der SIARD-Datei (schematisch)</p> <pre> Northwind.siard content/ schema0/ table0/ table0.xml table0.xsd lob4⁹/ record0.bin record1.bin table1/ table1.xml table1.xsd ... </pre> <p>Empfehlung <i>Es wird empfohlen, die Namen der LOB-Ordner und LOB-Dateien zu normalisieren und anstelle des eigentlichen Namens z.B. <code>lob4/</code> und <code>record0.bin</code> oder <code>record0.txt</code> zu verwenden (siehe Einschränkungen unter P_4.2-6).</i></p>	M

⁹ Bei diesem Beispiel enthält die Spalte 4 zusätzliche lob-Dateien, die entsprechend in `lob4/` abgelegt werden.

ID	Beschreibung Anforderung	M/K
P_4.2-4	Zur einfacheren Erkennung des SIARD-Formats (z.B. durch PRONOM) muss der Ordner <code>header/</code> einen leeren Unterordner <code>header/version/2.0</code> enthalten, welcher die Version des SIARD-Formats identifiziert.	M
P_4.2-5	<p>Im Ordner <code>header/</code> müssen die Dateien <code>metadata.xml</code> und <code>metadata.xsd</code> vorhanden sein. Weitere Dateien, zum Beispiel Stylesheets, sind erlaubt.</p> <p>Beispiel Aufbau der SIARD-Datei (schematisch)</p> <pre> Northwind.siard content/ ... header/ metadata.xml metadata.xsd ... </pre>	M
P_4.2-6	<p>Alle Datei- und Ordernamen müssen wie folgt aufgebaut sein: Der Name muss mit einem Buchstaben [a-z respektive A-Z] beginnen und darf anschliessend nur folgende Zeichen enthalten:</p> <ul style="list-style-type: none"> • a-z • A-Z • 0-9 • – • . (darf nur für die Trennung zwischen Namen und Extension verwendet werden) <p>Empfehlung <i>Die Länge der Datei- und Ordernamen sollte möglichst 20 Zeichen nicht überschreiten, damit Schwierigkeiten mit zu grossen Pfadlängen unter Windows vermieden werden können.</i></p>	M

4.3 Korrespondenz zwischen Metadaten und Tabellendaten

P_4.3-1	<p>Die in metadata.xml vorgegebene Struktur muss identisch sein mit jener im Ordner content/.</p> <p>Beispiel</p> <div data-bbox="435 439 1305 1406"> <div> <p>SIARD-Metadaten</p> <pre> ... <dbname>northwind</dbname> ... <schemas> <schema> <name>admin</name> <folder>schema0</folder> <tables> <table> <name>Products</name> <folder>table0</folder> ... </table> <table> <name>Shippers</name> <folder>table1</folder> ... </table> <table> <name>Orders</name> <folder>table2</folder> ... </table> <table> <name>Categories</name> <folder>table3</folder> ... </table> <table> <name>Customers</name> <folder>table4</folder> ... </table> ... </schemas> ... </db> </pre> </div> <div> <p>content-Struktur</p> </div> </div>	M
---------	---	---

P_4.3-2	<p>Die in <code>metadata.xml</code> angegebene Anzahl der Spalten einer Tabelle muss identisch sein mit jener der entsprechenden Datei <code>table[Zahl].xsd</code>.</p> <p>Beispiel</p> <div style="display: flex; justify-content: space-around;"> <div style="border: 1px solid black; padding: 5px; width: 45%;"> <p style="text-align: center;">SIARD-Metadaten</p> <pre> ... <dbname>northwind</dbname> ... <schemas> <schema> <name>admin</name> <folder>schema0</folder> <tables> <table> <name>Products</name> <folder>table0</folder> <description/> <columns> <column> <column> <column> <column> <column> <column> <column> <column> </columns> <primaryKey> <foreignKeys> <rows>77</rows> </table> ... </pre> </div> <div style="border: 1px solid black; padding: 5px; width: 45%;"> <p style="text-align: center;">content – table0.xsd</p> <pre> <?xml version="1.0" encoding="UTF-8"?> <xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"> <xs:element name="table"> <xs:complexType name="rowType"> <xs:sequence> <xs:element name="c1" type="xs:integer"/> <xs:element minOccurs="0" name="c2" type="xs:string"/> <xs:element minOccurs="0" name="c3" type="xs:integer"/> <xs:element minOccurs="0" name="c4" type="xs:integer"/> <xs:element minOccurs="0" name="c5" type="xs:string"/> <xs:element minOccurs="0" name="c6" type="xs:decimal"/> <xs:element minOccurs="0" name="c7" type="xs:integer"/> <xs:element minOccurs="0" name="c8" type="xs:integer"/> <xs:element minOccurs="0" name="c9" type="xs:integer"/> <xs:element name="c10" type="xs:boolean"/> </xs:sequence> </xs:complexType> </xs:schema> </pre> </div> </div> 	M
---------	--	---

P_4.3-3	<p>Die Datentyp-Angaben zu den Spaltendefinitionen in <code>metadata.xml</code> müssen identisch sein mit jenen der entsprechenden Datei <code>table[Zahl].xsd</code>.</p> <p>Die vordefinierten SQL:2008-Datentypen werden in den Schemadateien <code>table[Zahl].xsd</code> gemäss der folgenden Tabelle in XML-Datentypen umgewandelt.</p> <table border="1"> <thead> <tr> <th>SQL:2008</th><th>XML</th></tr> </thead> <tbody> <tr> <td>BIGINT</td><td>xs:integer</td></tr> <tr> <td>BINARY LARGE OBJECT</td><td>blobType¹⁰</td></tr> <tr> <td>BINARY VARYING</td><td>xs:hexBinary</td></tr> <tr> <td>BINARY[...]</td><td>xs:hexBinary</td></tr> <tr> <td>BIT VARYING(...)</td><td>xs:hexBinary</td></tr> <tr> <td>BIT(...)</td><td>xs:hexBinary</td></tr> <tr> <td>BOOLEAN</td><td>xs:boolean</td></tr> <tr> <td>CHARACTER LARGE OBJECT</td><td>clobType¹⁰</td></tr> <tr> <td>CHARACTER VARYING(...)</td><td>xs:string</td></tr> <tr> <td>CHARACTER(...)</td><td>xs:string</td></tr> <tr> <td>DATE</td><td>dateType</td></tr> <tr> <td>DECIMAL(...)</td><td>xs:decimal</td></tr> <tr> <td>DOUBLE PRECISION</td><td>xs:float</td></tr> <tr> <td>FLOAT(...)</td><td>xs:float</td></tr> <tr> <td>INTEGER</td><td>xs:integer</td></tr> <tr> <td>INTERVAL</td><td>xs:duration</td></tr> </tbody> </table>	SQL:2008	XML	BIGINT	xs:integer	BINARY LARGE OBJECT	blobType ¹⁰	BINARY VARYING	xs:hexBinary	BINARY[...]	xs:hexBinary	BIT VARYING(...)	xs:hexBinary	BIT(...)	xs:hexBinary	BOOLEAN	xs:boolean	CHARACTER LARGE OBJECT	clobType ¹⁰	CHARACTER VARYING(...)	xs:string	CHARACTER(...)	xs:string	DATE	dateType	DECIMAL(...)	xs:decimal	DOUBLE PRECISION	xs:float	FLOAT(...)	xs:float	INTEGER	xs:integer	INTERVAL	xs:duration	M
SQL:2008	XML																																			
BIGINT	xs:integer																																			
BINARY LARGE OBJECT	blobType ¹⁰																																			
BINARY VARYING	xs:hexBinary																																			
BINARY[...]	xs:hexBinary																																			
BIT VARYING(...)	xs:hexBinary																																			
BIT(...)	xs:hexBinary																																			
BOOLEAN	xs:boolean																																			
CHARACTER LARGE OBJECT	clobType ¹⁰																																			
CHARACTER VARYING(...)	xs:string																																			
CHARACTER(...)	xs:string																																			
DATE	dateType																																			
DECIMAL(...)	xs:decimal																																			
DOUBLE PRECISION	xs:float																																			
FLOAT(...)	xs:float																																			
INTEGER	xs:integer																																			
INTERVAL	xs:duration																																			

¹⁰ Zu den XML-Datentypen *blobType* und *clobType* siehe G_3.1-1.

NATIONAL CHARACTER LARGE OBJECT	clobType ¹⁰
NATIONAL CHARACTER VARYING(...)	xs:string
NATIONAL CHARACTER(...)	xs:string
NUMERIC(...)	xs:decimal
REAL	xs:float
SMALLINT	xs:integer
TIME	timeType
TIME WITH TIME ZONE	timeType
TIMESTAMP	dateTimeType
TIMESTAMP WITH TIME ZONE	dateTimeType
XML	clobType

Beispiel

Alle DATE- und TIME-Typen sind in der UTC-Zeitzone spezifiziert. Es wird empfohlen, sie mit einem „Z“ zu beenden.


dateTimeType ist eine Restriktion von xs:date in UTC auf Jahre zwischen 0001 und 9999 (SQL:2008-Restriktion).

timeType ist eine Restriktion von xs:time auf die UTC-Zeitzone.

dateTimeType ist eine Restriktion von xs:dateTime in der UTC-Zeitzone auf Jahre zwischen 0001 und 9999 (SQL:2008-Restriktion).

clobType ist eine Extension von xs:string (für Inline-Werte) mit den optionalen Attributen *file*, *length* und *messageDigest*, welche notwendig sind, wenn der CLOB-Wert nicht einfach als String inline gespeichert ist. In diesem Fall ist der Wert des *file*-Attributs die Datei-URI (URL-encodiert, wenn möglich auf den nächstliegenden lobFolder bezogen), wo das CLOB gespeichert ist, der Wert des *length*-Attributs ist die Länge in UTF-8, und das optionale *messageDigest*-Attribut ist ein String, der dem gleichen Muster (Algorithmus und Wert) folgt wie der globale *messageDigest* (siehe 5.1-1). blobType ist eine Extension von xs:hexBinary (für Inline-Werte) mit den optionalen Attributen *file*, *length* und *messageDigest*, welche notwendig sind, wenn der BLOB-Wert nicht einfach als hexadezimaler String inline gespeichert werden kann. In diesem Fall ist der Wert des *file*-Attributs die Datei-URI (URL-encodiert, wenn möglich auf den nächstliegenden lobFolder bezogen), wo das BLOB gespeichert ist, der Wert des *length*-Attributs ist die Länge in Bytes, und das optionale *messageDigest*-Attribut ist ein String, der dem gleichen Muster (Algorithmus und Wert) folgt wie der globale *messageDigest* (siehe 5.1-1).

P_4.3-4	Die benannten DISTINCT-Datentypen werden in denjenigen <code>table[number].xsd</code> -Schemadateien in den XML-Datentyp konvertiert, welche zur Darstellung ihrer Basistypen verwendet würden.	M
---------	---	---

P_4.3-5	<p>Der benannte ROW-Containertyp wird in den <code>table[number].xsd</code>-Schemadateien in eine Sequenz strukturierter XML-Elemente <code><r1></code>, <code><r2></code>, ... konvertiert, welche einen Eintrag pro Feld enthalten. Der Datentyp jedes Feldes wird analog zu einem Spalten-Datentyp in den XML-Datentypen konvertiert. Siehe das Beispiel <code>table0.xsd</code> im Anhang D.3.</p>	M
P_4.3-6	<p>Der ARRAY-Containertyp wird in den <code>table[number].xsd</code>-Schemadateien in eine Sequenz strukturierter XML-Elemente <code><a1></code>, <code><a2></code>, ... konvertiert, welche ihrerseits in den XML-Datentypen konvertiert werden, der dem Basistyp des Array entspricht. Siehe das Beispiel <code>table0.xsd</code> im Anhang D.3.</p>	
P_4.3-7	<p>Der benannte User-defined Data Type (UDT) wird in den <code>table[number].xsd</code>-Schemadateien in eine Sequenz strukturierter XML-Elemente <code><u1></code>, <code><u2></code>, ... konvertiert, welche ihrerseits in den XML-Datentypen konvertiert werden, der dem Typen jedes Attributs entspricht. Siehe das Beispiel <code>table0.xsd</code> im Anhang D.3.</p>	
P_4.3-8	<p>Die Nullable-Angaben zu den Spaltendefinitionen in <code>metadata.xml</code> müssen identisch sein mit jenen der entsprechenden Datei <code>table[Zahl].xsd</code>.</p> <p>Beispiel</p>  <p>Hinweis</p> <p>Die SQL:2008-Notation "<code><nullable>true</nullable></code>" wird in XML zu "<code>minOccurs='0'</code>". "<code><nullable>false</nullable></code>" entspricht "<code>minOccurs='1'</code>" in XML; da dies jedoch der Standardwert ist, wird er oftmals weggelassen.</p>	M
P_4.3-9	<p>Die Spaltenreihenfolge in <code>metadata.xml</code> muss identisch sein mit der Spaltenreihenfolge in der entsprechenden <code>table[number].xsd</code>.</p>	M
P_4.3-10	<p>Die Feldreihenfolge in der Typdefinition einer Spalte in <code>metadata.xml</code> muss identisch sein mit der entsprechenden Attributreihenfolge in der Typdefinition von <code>metadata.xml</code>.</p>	M
P_4.3-11	<p>Die Feldreihenfolge in der Tabellendefinition von <code>metadata.xml</code> muss identisch sein mit der Feldreihenfolge der entsprechenden <code>table[Zahl].xsd</code>.</p>	M

<p>P_4.3-7</p>	<p>Die Anzahl Zeilen einer Tabelle in metadata.xml muss in den von der entsprechenden table[Zahl].xsd spezifizierten Bereich hineinpassen.</p> <p>Die Anzahl Zeilen einer Tabelle in metadata.xml muss identisch sein mit der Anzahl Zeilen in der entsprechenden table[Zahl].xml.</p> <p>Beispiel</p>  <p>Empfehlung</p> <p>Es wird empfohlen, in table[Zahl].xsd den Bereich 0 bis unendlich (maxOccurs="unbounded" minOccurs="0") zu verwenden. So werden Probleme bei der Validierung von table[Zahl].xml gegenüber table[Zahl].xsd vermieden.</p>	<p>M</p>
----------------	--	----------

5 Anforderungen an die Metadaten

Die Metadaten im SIARD-Archiv speichern die Struktur der archivierten Datenbank und geben an, welche Tabellendaten wo im Archiv zu finden sind.

Sämtliche Metadaten werden in einer einzigen Datei `metadata.xml` im Ordner `header/` versammelt. Die Datei ist im Gegensatz zu einer relationalen Datenbank hierarchisch aufgebaut.

Für die Datei `metadata.xml` existiert die Schemadefinition `metadata.xsd`, welche ebenfalls im Ordner `header/` abgelegt ist.

ID	Beschreibung Anforderung	M/K
M_5.0-1	Die Schemadefinition <code>metadata.xsd</code> ist für die Datei <code>metadata.xml</code> verbindlich einzuhalten. Das heisst, <code>metadata.xml</code> muss gegenüber <code>metadata.xsd</code> positiv validiert werden können.	M

Die Inhalte der einzelnen Ebenen werden im Folgenden definiert.

5.1 Metadaten auf der Ebene Datenbank

Die Datei `metadata.xml` enthält folgende globalen Angaben auf der Ebene Datenbank:

ID	Beschreibung Anforderung	M/K
M_5.1-1	Alle Metadaten, die in <code>metadata.xsd</code> auf der Ebene Datenbank als Muss bezeichnet sind, müssen entsprechend ausgefüllt sein.	M

Die folgenden Datenbank-Metadaten werden in der Datei `metadata.xml` gespeichert:

Bezeichnung	Bedeutung	M/K
version	SIARD-Format-Version	M
dbname	Kurze Bezeichnung der Datenbank	M
description	Beschreibung der Bedeutung und des Inhalts der Datenbank als Ganzes.	K
archiver	Name der Person, welche die Archivierung der Tabellendaten aus der Datenbank durchführte	K
archiverContact	Kontaktdaten (Telefon, E-Mail) zur Person, welche die Archivierung der Tabellendaten aus der Datenbank durchführte	K
dataOwner	Eigentümer der Daten in der Datenbank; die Institution oder Person, welche zum Zeitpunkt der Archivierung das Recht besitzt, Lizenzrechte an der Nutzung der Daten zu vergeben und die für die Einhaltung gesetzlicher Auflagen wie Datenschutzrichtlinien verantwortlich ist	M
dataOriginTimespan	Entstehungszeitraum der Daten in der Datenbank; eine ungefähre Zeitangabe als Text	M

Bezeichnung	Bedeutung	M/K
lobFolder	<p>Eine „file:“-URI, die als Basis-URI dient für relative URI, welche den möglichen externen Speicherort von <i>Large Objects</i> angeben. Wenn dieses Metadatum fehlt, ist der Default der Stammordner in der ZIP-Datei. Relative lobFolder-URI in den Spalten-Metadaten sind relativ zu diesem Wert.</p> <p>Anmerkung Wenn die „file:“-URI auf ein <i>Extended File System</i> verweist, in welchem ZIP-Dateien als Ordner behandelt werden, verweist die relative URI „..“ auf den externen Ordner, in welchem die SIARD-Datei liegt. Wenn eine solche Dateisystem-Extension nicht unterstützt wird, müssen absolute „file:“-URI zur Angabe eines externen Speicherorts für LOB-Dateien verwendet werden. Es wird ausdrücklich empfohlen, alle <i>lobFolder</i>-Einträge in Spalten und alle LOB-Dateiattribute als relative URI abzubilden. So muss bei einer Verschiebung der SIARD-Datei oder ihres Informationspakets nur diese absolute URI geändert werden, um auf den neuen Speicherort zu verweisen.</p>	K
producerApplication	Name und Version der Anwendung, welche die SIARD-Datei heruntergeladen hat.	K
archivalDate	Archivierungsdatum; Datum der Archivierung der Tabellendaten	M
messageDigest	<p>Hexadezimaler Message-Digest-Code über den Ordner <code>content/</code> mit Präfix, welches den Typ des Digest-Algorithmus angibt (wie MD5, SHA-1 oder SHA-256), gefolgt vom hexadezimalen Message Digest-Code, zum Beispiel „MD55234Fd874EFADFC8A86E5CDDC97398991“. Es kann mehr als ein Message-Digest-Code gespeichert werden, basierend auf verschiedenen Algorithmen. Wenn kein Message-Digest gespeichert ist, muss die Integrität über die Speicherung eines Message-Digest ausserhalb der SIARD-Datei gesichert werden.</p> <p>Empfehlung <i>Wird die Option MessageDigest verwendet, muss folgendes umgesetzt werden:</i> <i>Die Verzeichnisse content und header werden als separate (leere) Einträge content/ und header/ in der ZIP-Datei gespeichert. Damit die Integrität der Primärdaten überprüft werden kann, ist es notwendig, dass der Eintrag des header-Verzeichnisses erst nach allen Primärdaten im content/-Eintrag und vor allen anderen Metadateneinträgen eingefügt wird. Der unten erwähnte MessageDigest wird von Offset 0 bis zum Offset des header/-Eintrags der SIARD-Datei berechnet.</i></p>	K
clientMachine	DNS-Name des (Client-)Rechners, auf welchem die Archivierung durchgeführt wurde	K
databaseProduct	Datenbank-Produkt und Version, aus welchem die Archivierung der Tabellendaten erfolgte	K
connection	Verwendeter Connection String für die Archivierung der Tabellendaten	K
databaseUser	Datenbank-UserId des Benutzers des SIARD-Werkzeugs für das Archivieren der Tabellendaten aus der Datenbank	K
schemas	Liste der Schemas in der Datenbank	M

Bezeichnung	Bedeutung	M/K
users	Liste der Datenbank-Benutzer	M
roles	Liste der Datenbank-Rollen	K
privileges	Liste der Privilegien für Benutzer und Rollen	K

5.2 Metadaten auf der Ebene Schema

Die Schema-Metadaten werden wie schon die globalen Angaben zur Datenbank in der Datei `metadata.xml` archiviert.

ID	Beschreibung Anforderung	M/K
M_5.2-1	Alle Metadaten, die in <code>metadata.xsd</code> auf der Ebene Schema als Muss bezeichnet sind, müssen entsprechend ausgefüllt sein.	M

Die folgenden Schema-Metadaten werden in der Datei `metadata.xml` gespeichert:

Bezeichnung	Bedeutung	M/K
name	Schemaname in der Datenbank	M
folder	Name des Schemaordners unter <code>content/</code> im SIARD-Archiv	M
types	Liste der (benannten) fortgeschrittenen oder strukturierten Typen im Schema	K
description	Beschreibung der Bedeutung und des Inhalts des Schemas	K
tables	Liste der Tabellen in der Datenbank	M
views	Liste der in der Datenbank gespeicherten Queries (Abfragen)	K
routines	Liste der Routinen (früher Stored Procedures genannt) im Schema	K

5.3 Metadaten auf der Ebene Type

ID	Beschreibung Anforderung	M/K
M_5.3-1	Die Type-Metadaten eines Schemas können in der Datei <code>metadata.xml</code> archiviert werden.	K

Die folgenden Type-Metadaten werden in `metadata.xml` gespeichert, wenn ein fortgeschrittener oder strukturierter Datentyp archiviert wird:

Bezeichnung	Bedeutung	M/K
name	Name des Datentyps im Schema.	M

Bezeichnung	Bedeutung	M/K
category	Kategorie des fortgeschrittenen oder strukturierten Datentyps ("distinct", "row", "array" oder "udt").	M
underSchema	Schemaname des Supertyps, falls der Datentyp auf einem Supertyp basiert.	K
underName	Name des Supertyps, falls der Datentyp auf einem Supertyp basiert.	K
instantiable	Wahr, falls der Datentyp instanziiert werden kann, sonst falsch.	M
final	Wahr, falls keine Subtypen zu diesem Datentyp geschaffen werden können, sonst falsch.	M
base	Name des (primitiven) Basistyps, falls die Kategorie "distinct" ist.	K
cardinality	(Maximale) Anzahl Elemente, falls die Kategorie "array" ist.	K
attributes	Liste der Attribute, falls die Kategorie "row" oder "udt" ist	K
description	Beschreibung von Bedeutung und Inhalt des Datentyps.	K

5.4 Metadaten auf der Ebene Attribut

ID	Beschreibung Anforderung	M/K
M_5.4-1	Die im Datentyp verwendeten Attribut-Metadaten können in <code>metadata.xml</code> archiviert werden.	K

Bezeichnung	Bedeutung	M/K
name	Name des Attributs.	M
type	Vordefinierter SQL:2008-Datentyp des Attributs gemäss SQL:2008.	K
typeOriginal	Originaler Spaltentyp für den standardmässigen Datentyp. Hinweis Da die verschiedenen sich SQL-konform nennenden Datenbank-Programme sehr verschiedene Datentypen zulassen, ist hier der originale Spaltentyp ebenso aufgelistet wie der SQL:2008-Datentyp. In jedem Datenbank-Programm, das das SIARD-Format unterstützt, muss eine Übersetzung der proprietären Datentypen in SQL:2008-Datentypen definiert und in der jeweiligen Applikation dokumentiert werden.	K
typeSchema	Schema des fortgeschrittenen oder strukturierten Datentyps.	K
typeName	Name des fortgeschrittenen oder strukturierten Datentyps.	K
attributes	Liste der Attribute, falls die Kategorie "row" oder "udt" ist	K
defaultValue	Standardwert des Attributs,	K
description	Beschreibung der Bedeutung und Funktion der Routine.	K

5.5 Metadaten auf der Ebene Tabelle

Die Metadaten auf der Ebene Tabelle werden wie schon die globalen Angaben zur Datenbank und die Schema-Metadaten in der Datei `metadata.xml` archiviert.

ID	Beschreibung Anforderung	M/K
M_5.5-1	Alle Metadaten, die in <code>metadata.xsd</code> auf der Ebene Tabelle als Muss bezeichnet sind, müssen entsprechend ausgefüllt sein.	M

Die folgenden Tabellen-Metadaten werden in der Datei `metadata.xml` gespeichert:

Bezeichnung	Bedeutung	M/K
name	Tabellenname im Schema	M
folder	Name des Tabellenordners im Schemaordner	M
description	Beschreibung der Bedeutung und des Inhalts der Tabelle	K
columns	Liste der Spalten der Tabelle	M
primaryKey	Primärschlüssel der Tabelle	K
foreignKeys	Liste der Fremdschlüssel der Tabelle	K
candidateKeys	Liste der Kandidatenschlüssel der Tabelle	K
checkConstraints	Liste der Einschränkungen der Tabelle	K
triggers	Liste der Triggers der Tabelle	K
rows	Anzahl Datensätze	M

5.6 Metadaten auf der Ebene Spalte

Die Metadaten auf der Ebene Spalte werden wie schon die globalen Angaben zur Datenbank, die Schema-Metadaten und die Metadaten auf der Ebene Tabelle in der Datei `metadata.xml` archiviert. Spalten-Metadaten beschreiben eine Spalte in einer Tabelle oder View.

ID	Beschreibung Anforderung	M/K
M_5.6-1	Alle Metadaten, die in <code>metadata.xsd</code> auf der Ebene Spalte als Muss bezeichnet sind, müssen ausgefüllt sein.	M

Die folgenden Spalten-Metadaten werden in der Datei `metadata.xml` gespeichert:

Bezeichnung	Bedeutung	M/K
name	Spaltenname in der Tabelle oder View Innerhalb der gleichen Tabelle muss der Spaltenname eindeutig sein.	M

Bezeichnung	Bedeutung	M/K
folder	<p>Eingangsname des LOB- Ordners im Tabellenordner, falls die Spalte in der SIARD-Datei gespeichert ist.</p> <p>Diese Methode zur Angabe des Speicherorts des LOB-Ordners aus Version 1.0 der SIARD-Formatspezifikation wird durch das neue <i>lobFolder</i>-Element obsolet (siehe unten). Sie wird hier beibehalten, um die Validität bestehender SIARD-Dateien unter der neuen Version 2.0 der SIARD-Formatspezifikation sicherzustellen. Anwendungen sollten sie beim Lesen einer SIARD-Datei unterstützen, aber beim Schreiben einer SIARD-Datei <i>lobFolder</i> benutzen. N.B.: Die Semantik von <i>folder</i> unterscheidet sich von derjenigen von <i>lobFolder</i>. <i>lobFolder</i> beinhaltet eine „file:“-URI, <i>folder</i> beinhaltet einen ZIP-Eingangsnamen.</p> <p>Hinweis</p> <p>Der optionale LOB-Ordnername wird nur für Spalten der <i>Large-Object</i>-Typen (z.B BLOB, CLOB oder XML) benötigt, welche in der SIARD-Datei gespeichert sind.</p> <p>Die Dateien, welche die <i>Large-Object</i>-Felder repräsentieren, werden in diesen Ordnern angelegt und heissen <i>record0.txt</i>, <i>record1.txt</i>, bzw. <i>record0.bin</i>, <i>record1.bin</i> ... oder <i>a1/record2.bin</i>, <i>r1/record234.txt</i>, ... Diese werden in der Daten-XML-Datei referenziert.</p>	K
lobFolder	<p>Name des LOB-Ordners als relative oder absolute „file:“-URI, gegebenenfalls im externen Dateisystem. Das Element kann sowohl für interne als auch für externe Speicherung von <i>Large Objects</i> benutzt werden.</p> <p>Das Element <i>lobFolder</i> kann nicht gleichzeitig mit dem Element <i>folder</i> benutzt werden, welches es ablöst. Beim Schreiben einer SIARD-Datei muss das Element <i>lobFolder</i> benutzt werden.</p> <p>Hinweis</p> <p>Dieser Eintrag ist nur von Bedeutung, wenn die Spalte eine LOB-Spalte ist (z.B. vom Typ BLOB, CLOB oder XML).</p> <p>Wenn es fehlt, wird als Defaultwert „.“ angenommen, z.B. als Verweis auf den gleichen Ordner wie <i>lobFolder</i> auf der Ebene Datenbank. Andernfalls muss sein Wert eine (wenn möglich relative) „file:“-URI sein, welche den Ordner bezeichnet, in dem die Dateien dieser LOB-Spalte gespeichert werden sollen.</p> <p>Wenn dieser Wert eine relative URI ist, wird angenommen, dass sie relativ ist zum globalen <i>lobFolder</i>-Eintrag auf der Ebene Datenbank.</p> <p>Die relativen <i>file</i>-Attribute der Zellen dieser Spalte werden als relativ zu diesem Ordner interpretiert.</p>	
type	<p>Vordefinierter SQL:2008-Typ der Spalte</p> <p>Hinweis</p> <p>Wenn der Datentyp dieser Spalte ein vordefinierter Datentyp ist, ist dieses Feld obligatorisch. Andernfalls muss das Feld <i>typeName</i> auf einen definierten Typ in der Typenliste verweisen.</p>	M

Bezeichnung	Bedeutung	M/K
typeOriginal	<p>Originaler Spaltentyp</p> <p>Hinweis Da die verschiedenen sich SQL-konform nennenden Datenbank-Programme sehr unterschiedliche Datentypen zulassen, wird hier neben dem SQL:2008-Typ auch der <i>originale</i> Typ aufgeführt. Für jedes das SIARD-Format unterstützende Datenbank-Programm ist eine Übersetzung der proprietären Typen zu SQL:2008-Typen bei der entsprechenden Applikation zu definieren und zu dokumentieren.</p>	K
nullable	Eintrag nicht erforderlich	K
typeSchema	Schema des benannten Typs wenn die Spalte kein vordefinierter Datentyp ist und der benannte Datentyp nicht im gleichen Schema definiert ist wie die Tabelle dieser Spalte.	K
typeName	Name des fortgeschrittenen oder strukturierten Datentyps dieser Spalte.	K
fields	Liste der Felder in der Spalte, falls die Spalte ein strukturierter Datentyp der Kategorie „row“ oder „udt“ ist.	K
defaultValue	Standardwert der Spalte	K
contentType	MIME Type dieser Spalte, falls es eine BLOB-Spalte ist und alle Einträge dieser Spalte Dateien vom gleichen MIME-Type enthalten. Dieses rein informative Element hilft bei der Auswahl des korrekten Viewers für Binärobjekte. Es kann entweder manuell ausgefüllt werden oder durch das herunterladende Programm unter Benutzung eines Mechanismus zur Formaterkennung.	K
description	Beschreibung der Bedeutung und des Inhalts der Spalte	K

5.7 Metadaten für Felder

ID	Beschreibung Anforderungen	M/K
M_5.7-1	Die Feldmetadaten einer Spalte oder eines Feldes können in <code>metadata.xml</code> archiviert werden.	O

Die folgenden Feldmetadaten werden in `metadata.xml` gespeichert wenn eine Spalte oder ein Feld ein fortgeschrittener oder strukturierter Datentyp der Kategorie "row" oder "udt" ist:

Bezeichnung	Bedeutung	M/K
folder	<p>Eingangsname des LOB- Ordners im Tabellenordner, falls die Spalte in der SIARD-Datei gespeichert ist.</p> <p>Hinweis</p> <p>Der optionale LOB-Ordnername wird nur für Spalten der <i>Large-Object</i>-Typen (z.B. BLOB, CLOB oder XML) benötigt, welche in der SIARD-Datei gespeichert sind..</p> <p>Die Dateien, welche die <i>Large-Object</i>-Felder repräsentieren, werden in diesen Ordnern angelegt und heissen <code>record0.txt</code>, <code>record1.txt</code>, bzw. <code>record0.bin</code>, <code>record1.bin</code> ... oder <code>a1/record2.bin</code>, <code>r1/record234.txt</code>, ... Diese werden in der Daten-XML-Datei referenziert.</p>	K
lobFolder	<p>Name des LOB-Ordners als relative oder absolute „file:“-URI, gegebenenfalls im externen Dateisystem. Das Element kann sowohl für interne als auch für externe Speicherung von <i>Large Objects</i> benutzt werden.</p> <p>Hinweis</p> <p>Dieser Eintrag ist nur von Bedeutung, wenn die Spalte eine LOB-Spalte ist (z.B. vom Typ BLOB, CLOB oder XML).</p> <p>Wenn es fehlt, wird als Defaultwert „.“ angenommen, z.B. als Verweis auf den gleichen Ordner wie <i>lobFolder</i> auf der Ebene Datenbank. Andernfalls muss sein Wert eine (wenn möglich relative) „file:“-URI sein, welche den Ordner bezeichnet, in dem die Dateien dieser LOB-Spalte gespeichert werden sollen.</p> <p>Wenn dieser Wert eine relative URI ist, wird angenommen, dass sie relativ ist zum globalen <i>lobFolder</i>-Eintrag auf der Ebene Datenbank.</p> <p>Die relativen <i>file</i>-Attribute der Zellen dieser Spalte werden als relativ zu diesem Ordner interpretiert.</p>	K
fields	Liste der Felder in der Spalte, falls die Spalte ein strukturierter Datentyp der Kategorie „row“ oder „udt“ ist.	K
contentType	MIME Type dieser Spalte, falls es eine BLOB-Spalte ist und alle Einträge dieser Spalte Dateien vom gleichen MIME-Type enthalten. Dieses rein informative Element hilft bei der Auswahl des korrekten Viewers für Binärobjekte. Es kann entweder manuell ausgefüllt werden oder durch das herunterladende Programm unter Benutzung eines Mechanismus zur Formaterkennung.	K
description	Beschreibung der Bedeutung und des Inhalts der Spalte.	K

5.8 Metadaten des Primärschlüssels

ID	Beschreibung Anforderung	M/K
M_5.8-1	Die Metadaten des Primärschlüssels einer Tabelle können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden Primärschlüssel-Metadaten werden in der Datei `metadata.xml` gespeichert, sofern ein Primärschlüssel archiviert wird:

Bezeichnung	Bedeutung	M/K
name	Name des Primärschlüssels	M
column	Liste der Spalten des Primärschlüssels	M
description	Beschreibung der Bedeutung und des Inhalts des Primärschlüssels	K

5.9 Metadaten der Fremdschlüssel

ID	Beschreibung Anforderung	M/K
M_5.9-1	Die Metadaten der Fremdschlüssel innerhalb einer Tabelle können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden Fremdschlüssel-Metadaten werden in der Datei `metadata.xml` gespeichert, sofern ein Fremdschlüssel archiviert wird:

Bezeichnung	Bedeutung	M/K
name	Name des Fremdschlüssels	M
referencedSchema	Schema der referenzierten Tabelle	M
referencedTable	Referenzierte Tabelle Hinweis Der referenzierte externe Tabellename kann vom Typ <code>tabelle</code> oder <code>schema.tabelle</code> sein. Dabei sind delimitierte Bezeichner in Anführungszeichen gesetzt.	M
reference	Referenz (Liste von Spalten und referenzierten Spalten)	M
matchType	Matchtyp (FULL, PARTIAL oder SIMPLE)	K
deleteAction	Löschaktion, z.B.: CASCADE Hinweis Die Lös- und Änderungsaktion enthalten die vom SQL:2008-Standard zugelassenen Aktionen.	K

Bezeichnung	Bedeutung	M/K
updateAction	Änderungsaktion, z.B.: SET DEFAULT	K
description	Beschreibung der Bedeutung und des Inhalts des Fremdschlüssels	K

5.10 Referenz-Metadaten

ID	Beschreibung Anforderung	M/K
M_5.10-1	Die Metadaten der Referenzen, welche beim Fremdschlüssel verwendet werden, können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden Referenzen-Metadaten werden in der Datei `metadata.xml` gespeichert, sofern ein Fremdschlüssel archiviert wird:

Bezeichnung	Bedeutung	M/K
column	Name der Spalte	M
referenced	Name der referenzierten Spalte	M

5.11 Metadaten des Kandidatenschlüssels

ID	Beschreibung Anforderung	M/K
M_5.11-1	Die Metadaten des Kandidatenschlüssels einer Tabelle können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden Kandidatenschlüssel-Metadaten werden in der Datei `metadata.xml` gespeichert, sofern ein Kandidatenschlüssel archiviert wird:

Bezeichnung	Bedeutung	M/K
name	Name des Kandidatenschlüssels	M
column	Liste der Spalten des Kandidatenschlüssels	M
description	Beschreibung der Bedeutung und des Inhalts des Kandidatenschlüssels	K

5.12 Metadaten der Check-Einschränkung

Die Check-Einschränkung besteht aus einer zu prüfenden Bedingung. Diese ist als Ausdruck vom Typ BOOLEAN (mit Wert *true*, *false* oder *unknown*) in SQL:2008-Syntax angegeben.

ID	Beschreibung Anforderung	M/K
M_5.12-1	Die Metadaten der Check-Einschränkung einer Tabelle können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden Check-Einschränkung-Metadaten werden in der Datei `metadata.xml` gespeichert, sofern eine Check-Einschränkung archiviert wird:

Bezeichnung	Bedeutung	M/K
name	Name der Check-Einschränkung	M
condition	Bedingung der Check-Einschränkung	M
description	Beschreibung der Bedeutung und des Inhalts der Check-Einschränkung	K

5.13 Metadaten auf der Ebene Trigger

ID	Beschreibung Anforderung	M/K
M_5.13-1	Die Metadaten des Triggers einer Tabelle können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden Trigger-Metadaten werden in der Datei `metadata.xml` gespeichert, sofern ein Trigger archiviert wird:

Bezeichnung	Bedeutung	M/K
name	Triggernamen in der Tabelle	M
actionTime	BEFORE oder AFTER	M
triggerEvent	INSERT, DELETE, UPDATE [OF <trigger column list>]	M
aliasList	<old or new value alias list>	K
triggeredAction	<triggered action>	M
description	Beschreibung der Bedeutung und des Inhalts des Triggers	K

5.14 Metadaten auf der Ebene View

ID	Beschreibung Anforderung	M/K
M_5.14-1	Die Metadaten der View eines Schemas können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden View-Metadaten werden in der Datei `metadata.xml` gespeichert, sofern eine View archiviert wird:

Bezeichnung	Bedeutung	M/K
name	Name der View im Schema	M

Bezeichnung	Bedeutung	M/K
columns	Liste der Spaltennamen der View Hinweis Die Metadaten der Spalten einer View sind identisch strukturiert wie diejenigen einer Tabelle.	M
query	SQL:2008-Abfrage, welche die View definiert	K
queryOriginal	Originale SQL-Abfrage, welche die View definiert Hinweis Da die verschiedenen sich SQL-konform nennenden Datenbank-Programme sehr unterschiedliche Abfrage-Syntax zulassen, wird hier neben der SQL:2008-Abfrage auch die originale Abfrage aufgeführt. Für jedes das SIARD-Format unterstützende Datenbank-Programm ist eine Übersetzung der proprietären Abfragesyntax zu SQL:2008-Typen bei der entsprechenden Applikation zu definieren und zu dokumentieren.	K
description	Beschreibung der Bedeutung und des Inhalts der View	K

5.15 Metadaten auf der Ebene Routine

ID	Beschreibung Anforderung	M/K
M_5.15-1	Die Metadaten der Routine eines Schemas können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden Routine-Metadaten werden in der Datei `metadata.xml` gespeichert, sofern eine Routine archiviert wird:

Bezeichnung	Bedeutung	M/K
name	Routinename im Schema	M
description	Beschreibung der Bedeutung und des Inhalts der Routine	K
source	originaler Quellcode der Routine (VBA, PL/SQL, JAVA) Hinweis Da viele Datenbank-Programme über proprietäre Routinen verfügen, die nicht in eine SQL:2008-konforme Abfrage transformiert werden können, kann hier der originale Quellcode der Routine (z.B. in PL/SQL bei Oracle-Datenbanken, VBA bei MS Access Modulen) archiviert werden.	K
body	SQL:2008-konformer Quellcode der Routine	K
characteristic	Charakteristik der Routine	K
returnType	Rückgabebetyp der Routine (sofern es sich um eine Funktion handelt)	K
parameters	Liste der Parameter	K

5.16 Metadaten der Parameter

ID	Beschreibung Anforderung	M/K
M_5.16-1	Die Metadaten der Parameter, welche bei der Routine verwendet werden, können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden Parameter-Metadaten werden in der Datei `metadata.xml` gespeichert, sofern eine Routine archiviert wird:

Bezeichnung	Bedeutung	M/K
name	Name des Parameters	M
mode	Mode des Parameters (IN, OUT oder INOUT)	M
type	Vordefinierter SQL:2008-Typ des Parameters. Hinweis Wenn der Datentyp dieser Spalte ein vordefinierter Datentyp ist, muss dieses Feld benutzt werden. Andernfalls muss das Feld <code>typeName</code> auf einen definierten Typen in der Typenliste verweisen.	M
typeOriginal	originaler Parametertyp Hinweis Da die verschiedenen sich SQL-konform nennenden Datenbank-Programme sehr verschiedene Datentypen zulassen, ist hier der originale Spaltentyp ebenso aufgelistet wie der SQL:2008-Datentyp. In jedem Datenbank-Programm, das das SIARD-Format unterstützt, muss eine Übersetzung der proprietären Datentypen in SQL:2008-Datentypen definiert und in der jeweiligen Applikation dokumentiert werden.	K
typeSchema	Schema des benannten Typs, falls der Parameter kein vordefinierter Datentyp und der benannte Datentyp nicht im gleichen Schema wie die Tabelle dieser Spalte definiert ist.	K
typeName	Name des fortgeschrittenen oder strukturierten Datentyps dieses Parameters.	K
parameterFields	Liste der Felder im Parameter, falls der Parameter ein strukturierter Datentyp der Kategorie „row“ oder „udt“ ist.	K
description	Beschreibung der Bedeutung und der Funktion der Routine	K

5.17 Metadaten von ParameterField

ID	Beschreibung Anforderung	M/K
M_5.17-1	Die Feldmetadaten eines Parameters oder ein <code>parameterField</code> kann in <code>metadata.xml</code> archiviert werden.	K

Die folgenden *parameterField*-Metadaten werden in `metadata.xml` gespeichert, wenn ein Parameter oder ein *parameterField* ein fortgeschrittener oder strukturierter Datentyp der Kategorie "row" oder "udt" ist:

Bezeichnung	Bedeutung	M/K
parameterFields	Liste der parameterFields im parameterField, falls parameterField ein strukturierter Datentyp der Kategorie "row" oder "udt" ist.	K
description	Beschreibung der Bedeutung und der Funktion des parameterField	K

Weder Name noch Typ des Parameter-Felds müssen festgehalten werden, weil sie in der entsprechenden Attributliste des Typs gespeichert sind. Wenn der Parameter ein einfacher Typ ist, wird die *parameterFields*-Liste weggelassen. Ein Container für die Beschreibung wird jedoch gleichwohl gebraucht.

5.18 Metadaten auf der Ebene des Benutzers

ID	Beschreibung Anforderung	M/K
M_5.18-1	Die Metadaten der Benutzer können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden User-Metadaten werden in der Datei `metadata.xml` gespeichert:

Bezeichnung	Bedeutung	M/K
name	Name des Benutzers	M
description	Beschreibung der Bedeutung und der Funktion des Benutzers	K

5.19 Metadaten auf der Ebene Rolle

ID	Beschreibung Anforderung	M/K
M_5.19-1	Die Metadaten der Rolle können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden Role-Metadaten werden in der Datei `metadata.xml` gespeichert:

Bezeichnung	Bedeutung	M/K
name	Name der Rolle	M
admin	Administrator der Rolle (Benutzer oder Rolle)	M
description	Beschreibung der Bedeutung und der Funktion der Rolle	K

5.20 Metadaten auf der Ebene der Privilegien

ID	Beschreibung Anforderung	M/K
M_5.20-1	Die Metadaten der Privilegien können in der Datei <code>metadata.xml</code> archiviert werden	K

Die folgenden Privilegien-Metadaten werden in der Datei `metadata.xml` gespeichert:

Bezeichnung	Bedeutung	M/K
type	eingewäumtes Privileg (z.B. SELECT)	M
object	Objekt, auf welches das Privileg anzuwenden ist	K
grantor	Berechtigter, der das Privileg einräumt	M
grantee	Empfänger des Privilegs (Benutzer oder Rolle)	M
option	Grant-Option (ADMIN oder GRANT)	K
description	Beschreibung der Bedeutung und der Funktion des Grants	K

6 Anforderungen an die Tabellendaten

Wie bereits beschrieben, befinden sich die Tabellendaten einer archivierten relationalen Datenbank im Ordner `content/` in der Dokument-Root des SIARD-Archivs. Sie werden dort in dem jeweiligen Schema- und Tabellenordner abgelegt.

Die Tabellendaten sind jeweils in einer XML-Datei gespeichert. Pro Tabelle wird eine XML-Schemadefinition erzeugt, welche das XML-Speicherformat der Tabellendaten angibt. Entsprechend existiert für jede Tabelle die Datei `table[Zahl].xml` zur Schemadefinition `table[Zahl].xsd`.

ID	Beschreibung Anforderung	M/K
T_6.0-1	Die Gesamtheit der Tabellendaten (Primärdaten) muss den Konsistenzanforderungen von SQL:2008 entsprechen. Eine SIARD-Datei, die zwar syntaktisch gegen die verschiedenen XSDs validiert, aber semantisch gegen den SQL-Standard verstösst, ist nicht konform zu der vorliegenden Formatbeschreibung. Insbesondere müssen die Tabellenwerte den Einschränkungen der SQL-Typen in den Metadaten entsprechen. Ausserdem müssen die in den Metadaten gespeicherten Primär-, Kandidaten- und Fremdschlüsselbedingungen und die Nullabilitätsbedingungen alle erfüllt sein.	M
T_6.0-2	Die Schemadefinition <code>table[Zahl].xsd</code> ist für die Datei <code>table[Zahl].xml</code> verbindlich einzuhalten. Das heisst, <code>table[Zahl].xml</code> muss gegenüber <code>table[Zahl].xsd</code> positiv validiert werden können.	M

6.1 Tabellen-Schemadefinition

Die Datei `table[Zahl].xsd` enthält folgende Schemadefinitionen zu einer Tabelle:

ID	Beschreibung Anforderung	M/K
T_6.1-1	Pro Tabelle muss eine XML-Schemadefinition existieren, welche das XML-Speicherformat der Tabellendaten angibt.	M

ID	Beschreibung Anforderung	M/K
T_6.1-2	<p>Diese Schemadefinition spiegelt die SQL-Schema-Metadaten der Tabelle wider und gibt an, dass die Tabelle als Sequenz von Zeilen gespeichert wird, welche eine Sequenz von Spalteneinträgen mit verschiedenen XML-Typen enthalten. Der Name des Tabellen-Tags ist <i>table</i>, derjenige des Datensatz-Tags ist <i>row</i>, die Spalten-Tags heissen <i>c1</i>, <i>c2</i>,</p> <p>Beispiel</p> <pre> content – table0.xsd <?xml version="1.0" encoding="UTF-8"?> <xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified" targetNamespace= "http://www.admin.ch/xmlns/siard/1.0/schema0/table0.xsd" xmlns= "http://www.admin.ch/xmlns/siard/1.0/schema0/table0.xsd" xmlns:xs="http://www.w3.org/2001/XMLSchema"> <xs:element name="table"> <xs:complexType> <xs:sequence> <xs:element maxOccurs="unbounded" minOccurs="0" name="row" type="rowType"/> </xs:sequence> </xs:complexType> </xs:element> <xs:complexType name="rowType"> <xs:sequence> <xs:element name="c1" type="xs:integer"/> <xs:element minOccurs="0" name="c2" type="xs:string"/> <xs:element minOccurs="0" name="c3" type="xs:integer"/> <xs:element minOccurs="0" name="c4" type="xs:integer"/> <xs:element minOccurs="0" name="c5" type="xs:string"/> <xs:element minOccurs="0" name="c6" type="xs:decimal"/> <xs:element minOccurs="0" name="c7" type="xs:integer"/> <xs:element minOccurs="0" name="c8" type="xs:integer"/> <xs:element minOccurs="0" name="c9" type="xs:integer"/> <xs:element name="c10" type="xs:boolean"/> </xs:sequence> </xs:complexType> </xs:schema> </pre>	M
T_6.1-3	<p>Das Typen-Mapping, das in Tabellenschemadefinitionen verwendet werden soll, ist in P_4.3-3 spezifiziert. Zusätzlich zu den Standardtypen von XML Schema werden die folgenden speziellen Typen benutzt:</p> <p>clobType, blobType, dateType, dateTimeType</p>	M
T_6.1-4	<p>Mehrfachwerte von fortgeschrittenen oder strukturierten Typen müssen als separate Elemente innerhalb der Zellen-Tags gespeichert werden.</p> <p>Die Namen der individuellen Elemente eines ARRAY sind a1, a2, ...</p> <p>Die Namen der individuellen Elemente einer ROW sind r1, r2, ...</p> <p>Die Namen der individuellen Elemente eines UDT sind u1, u2, ...</p> <p>Siehe das Beispiel in Anhang D.3.</p>	?

6.2 Large-Object-Datenzellen

ID	Beschreibung Anforderung	M/K
T_6.2-1	<p><i>Large Objects</i> können inline in der <code>table[number].xml</code> Datei gespeichert werden, als separate Dateieinträge in der SIARD-Datei oder extern als Dateien im Dateisystem.</p>	M

Die folgenden *Large-Object*-Daten werden in einer LOB-Zelle der `table[number].xsd`-Datei gespeichert:

Bezeichnung	Bedeutung	M/K
file	Wenn das <i>Large Object</i> nicht inline gespeichert ist, bezeichnet dieses Element den Speicherort und den Namen der <i>Large-Object</i> -Datei in dieser Zelle oder diesem Zellattribut als "file:"-URI. Wenn es sich um eine relative URI handelt, wird diese relativ zum lobFolder (der Spalte oder des Attributs) des einschliessenden Elements interpretiert.	K
length	Länge (für BLOBs in Bytes, für CLOBs und XML in Zeichen)	K
digestType	Empfohlen für extern gespeicherte <i>Large Objects</i> . "MD5", "SHA-1" oder "SHA-256"	K
messageDigest	Empfohlen für extern gespeicherte <i>Large Objects</i> . Message Digest (MD5, SHA-1 oder SHA-256) über den Bytes des <i>Large Objects</i> .	K

6.3 Datums- und Timestamp-Datenzellen

ID	Beschreibung Anforderung	M/K
T_6.3-1	Daten und Zeitstempel müssen auf die Jahre 0001-9999 beschränkt sein, gemäss SQL:2008-Spezifikation. Diese Beschränkung wird in den Definitionen von <i>dateType</i> und <i>dateTimeType</i> erzwungen.	M

6.4 Tabellendaten

Die Datei `table[Zahl].xml` enthält die Tabellendaten zu dieser Tabelle:

ID	Beschreibung Anforderung	M/K
T_6.4-1	Pro Tabelle müssen die Tabellendaten jeweils in einer XML-Datei gespeichert sein.	M
T_6.4-2	<p>Die Datei <i>table</i> besteht aus <i>row</i>-Elementen, welche die Daten einer Zeile unterteilt in die verschiedenen Spalten (<i>c1</i>, <i>c2</i> ...) enthalten.</p> <p>Beispiel</p> <div style="border: 1px solid black; padding: 10px; margin: 10px 0;"> <p style="text-align: center;">content – table0.xml</p> <pre><?xml version="1.0" encoding="utf-8"> <table xsi:schemaLocation="http://www.admin.ch/xmlns/siard/1.0/schema0/table0.xsd table0.xsd" xmlns="http://www.admin.ch/xmlns/siard/1.0/schema0/table0.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"> <row><c1>1</c1><c2>Chai</c2><c3>1</c3><c4>1</c4> ... <c10>>false</c10></row> <row><c1>2</c1><c2>Chang</c2><c3>1</c3><c4>1</c4> ... <c10>>false</c10></row> <row><c1>3</c1><c2>Aniseed Syrup</c2><c3>1</c3><c4>2</c4> ... <c10>>false</c10></row> ... <row><c1>75</c1><c2>Rhönbräu Klosterbier</c2><c3>12</c3><c4>1</c4> ... <c10>>false</c10></row> <row><c1>76</c1><c2>Lakkalikööri</c2><c3>23</c3><c4>1</c4> ... <c10>>false</c10></row> <row><c1>77</c1><c2>Frankfurter grüne Soße</c2><c3>12</c3><c4>2</c4> ... <c10>>false</c10></row> </table></pre> </div>	M

ID	Beschreibung Anforderung	M/K
T_6.4-3	<p>Wenn eine Zelle einer Spalte oder eines Feldes NULL ist, muss sie weggelassen werden. Wenn sie gleich „“ ist (ein String der Länge 0), muss sie vorhanden, aber leer sein.</p> <p>Beispiel</p> <div style="border: 1px solid #add8e6; padding: 10px; margin: 10px 0;"> <p style="text-align: center;">content – table1.xml</p> <pre><?xml version="1.0" encoding="utf-8"?> <table xsi:schemaLocation="http://www.admin.ch/xmlns/siard/1.0/schema0/table1.xsd table1.xsd" xmlns="http://www.admin.ch/xmlns/siard/1.0/schema0/table1.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"> <row><c1>1</c1><c2>Speedy Express</c2></row> <row><c1>2</c1><c2>United Package</c2><c3/></row> <row><c1>3</c1><c2>Federal Shipping</c2><c3></c3></row> </table></pre> </div>	M
T_6.4-4	<p>Wenn eine Zelle einer Spalte einen komplexen Wert enthält (ARRAY, ROW, UDT), wird sie durch eine Sequenz von Subelementen der Zelle repräsentiert (a1, a2, ... für ARRAYS, r1, r2, ... für ROWs, u1, u2, ... für UDTs), welche ihrerseits die entsprechenden Werte enthalten. Diese Werte können wiederum komplex sein.</p> <p>Siehe das Beispiel in Anhang D4.</p>	O

ID	Beschreibung Anforderung	M/K
T_6.4-5	<p>Wenn eine Tabelle Daten der <i>Large-Object</i>-Typen (BLOB, CLOB, ...) enthält, können hierfür separate Dateien erzeugt und anstelle des Zelleninhalts der Speicherort der Datei abgelegt werden.</p> <p>Die Entscheidung, <i>Large Objects</i> in separaten Dateien statt inline zu speichern, obliegt der Software, welche die SIARD-Datei erzeugt.</p> <p>Um zu vermeiden, dass leere Ordner entstehen, werden die Ordner nur angelegt, wenn sie notwendig sind, also Daten beinhalten.</p> <p>Wenn ein <i>Large Object</i> in einer separaten Datei gespeichert wird, muss sein Zelelement die Attribute file, length und digest haben. Dabei ist file eine zum lob-Folder-Element der Spalten- oder Attributmetadaten relative „file“-URI. <i>length</i> enthält die Länge in Bytes (für BLOBs) oder Zeichen (für CLOBs oder XMLs). <i>digest</i> hält einen Message Digest über der LOB-Datei fest und ermöglicht den Integritätscheck der SIARD-Datei auch bei externer Speicherung gewisser LOBs.</p> <p>Beispiel</p> <div data-bbox="432 828 1303 1167"> <p>content – table3.xml</p> <pre> <?xml version="1.0" encoding="utf-8"> <table xsi:schemaLocation="http://www.admin.ch/xmlns/siard/1.0/schema0/table3.xsd table3.xsd" xmlns="http://www.admin.ch/xmlns/siard/1.0/schema0/table3.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"> <row><c1>1</c1><c2>Beverages</c2><c3>Soft drinks, coffees, teas, beers, and ales</c3><c4 length= "10746" file="content/schema0/table3/lob4/record0.bin"/></row> <row><c1>2</c1><c2>Condiments</c2><c3>Sweet and savory sauces, relishes, spreads, and seasonings </c3><c4 length="10746" file="content/schema0/table3/lob4/record1.bin"/></row> ... <row><c1>7</c1><c2>Produce</c2><c3>Dried fruit and bean curd</c3><c4 length="10746" file= "content/schema0/table3/lob4/record6.bin"/></row> <row><c1>8</c1><c2>Seafood</c2><c3>Seaweed and fish</c3><c4 length="10746" file= "content/schema0/table3/lob4/record7.bin"/></row> </table> </pre> </div> <p>Empfehlung</p> <p>Es wird ausdrücklich empfohlen, entweder alle oder kein Large Object in einer Spalte inline zu speichern.</p> <p>Es wird empfohlen, die lob-Ordner und lob-Dateien zu normalisieren und anstelle des eigentlichen Namens z.B. lob4/ und record0.bin oder record0.txt zu verwenden.</p>	M

7 Version und Gültigkeit der Spezifikation

Die Spezifikation liegt in der Version 2.0 vor. Die Inhalte der Spezifikation werden periodisch durch die eCH-Fachgruppe *Digitale Archivierung* überprüft und wenn nötig angepasst.

8 Change-Management-Prozess

Das Change-Management dieses Standards folgt [eCH-0150], Szenario 3. Als *Change Manager* dient die Fachgruppenleitung, als *Change Board* ein von der Fachgruppe mandatierter Ausschuss oder eine Themengruppe.

9 Haftungsausschluss/Hinweise auf Rechte Dritter

eCH-Standards, welche der Verein **eCH** dem Benutzer zur unentgeltlichen Nutzung zur Verfügung stellt, oder welche **eCH** referenziert, haben nur den Status von Empfehlungen. Der Verein **eCH** haftet in keinem Fall für Entscheidungen oder Massnahmen, welche der Benutzer auf Grund dieser Dokumente trifft und / oder ergreift. Der Benutzer ist verpflichtet, die Dokumente vor deren Nutzung selbst zu überprüfen und sich gegebenenfalls beraten zu lassen. **eCH**-Standards können und sollen die technische, organisatorische oder juristische Beratung im konkreten Einzelfall nicht ersetzen.

In **eCH**-Standards referenzierte Dokumente, Verfahren, Methoden, Produkte und Standards sind unter Umständen markenrechtlich, urheberrechtlich oder patentrechtlich geschützt. Es liegt in der ausschliesslichen Verantwortlichkeit des Benutzers, sich die allenfalls erforderlichen Rechte bei den jeweils berechtigten Personen und/oder Organisationen zu beschaffen.

Obwohl der Verein **eCH** all seine Sorgfalt darauf verwendet, die **eCH**-Standards sorgfältig auszuarbeiten, kann keine Zusicherung oder Garantie auf Aktualität, Vollständigkeit, Richtigkeit bzw. Fehlerfreiheit der zur Verfügung gestellten Informationen und Dokumente gegeben werden. Der Inhalt von **eCH**-Standards kann jederzeit und ohne Ankündigung geändert werden.

Jede Haftung für Schäden, welche dem Benutzer aus dem Gebrauch der **eCH**-Standards entstehen ist, soweit gesetzlich zulässig, wegbedungen.

10 Urheberrechte

Wer **eCH**-Standards erarbeitet, behält das geistige Eigentum an diesen. Allerdings verpflichtet sich der Erarbeitende, sein betreffendes geistiges Eigentum oder seine Rechte an geistigem Eigentum anderer, sofern möglich, den jeweiligen Fachgruppen und dem Verein **eCH** kostenlos zur uneingeschränkten Nutzung und Weiterentwicklung im Rahmen des Vereinszweckes zur Verfügung zu stellen.

Die von den Fachgruppen erarbeiteten Standards können unter Nennung der jeweiligen Urheber von **eCH** unentgeltlich und uneingeschränkt genutzt, weiterverbreitet und weiterentwickelt werden.

eCH-Standards sind vollständig dokumentiert und frei von lizenz- und/oder patentrechtlichen Einschränkungen. Die dazugehörige Dokumentation kann unentgeltlich bezogen werden.

Diese Bestimmungen gelten ausschliesslich für die von **eCH** erarbeiteten Standards, nicht jedoch für Standards oder Produkte Dritter, auf welche in den **eCH**-Standards Bezug genommen wird. Die Standards enthalten die entsprechenden Hinweise auf die Rechte Dritter.

Anhang A – Mitarbeit & Überprüfung

Karin Bredenberg, National Archives of Sweden

Hedi Bruggisser, Staatsarchiv Thurgau

Georg Büchler, KOST

Janet Delve, University of Portsmouth

Boris Domajnko, Slovenian National Archives

Alain Dubois, Staatsarchiv Wallis

Luis Faria, KEEP SOLUTIONS, LDA

Bruno Ferreira, KEEP SOLUTIONS, LDA

Arne-Kristian Groven, National Archives Norway (Riksarkivet)

Martin Kaiser, KOST

Lambert Kansy, Staatsarchiv Basel Stadt

Markus Lischer, Staatsarchiv Luzern

Zoltán Lux, National Archives of Hungary

Anders Bo Nielsen, Danish National Archives (Rigsarkivet)

Krystyna Ohnesorge, Schweizerisches Bundesarchiv

Lauri Rätsep, National Archives of Estonia

Claire Röthlisberger-Jourdan, KOST

Hélder Silva, KEEP SOLUTIONS, LDA

Hartwig Thomas, Enter AG

Andreas Voss †, Schweizerisches Bundesarchiv

Anhang B – Abkürzungen und Glossar

Begriff	Beschreibung
AIP	Archival Information Package: AIP entstehen gemäss OAIS aus SIP im Laufe des Archivierungsprozesses der digitalen Unterlagen. AIP stellen diejenige Form der Informationspakete dar, in welcher die digitalen Unterlagen im digitalen Magazin gespeichert werden.
Aktenbildner	Bezeichnung der Stelle bzw. Organisationseinheit, welche die Unterlagen gebildet und geführt hat.
Archiv	<ol style="list-style-type: none"> 1. Institution/Stelle, die Archivgut erfasst, aufbewahrt, konserviert und zugänglich macht. 2. Archivierte Unterlagen einer Organisation. 3. Gebäude oder Institution, das/die für die Archivierung von Unterlagen gebaut oder hergerichtet wurde. 4. Begriff für eine Datei, die andere Dateien beinhaltet. Vgl. auch Archivdatei und als Synonym Containerdatei.
Archivgut	Als Archivgut gelten Unterlagen, die vom Archiv zur Aufbewahrung übernommen worden sind oder von anderen Stellen nach den gleichen Grundsätzen selbständig archiviert werden.
Datenbank	<p>Eine "Datenbank" besteht normalerweise aus einem oder mehreren Datenbank-Schemas sowie definierten Zugriffsrechten einzelner Benutzer und Rollen auf gewisse Teile der Datenbank. In SQL:2008 können Benutzer (Users) und Rollen (Roles) Träger von Berechtigungen (Privilegien) sein.</p> <p>Eine relationale Datenbank besteht somit aus einer Menge strukturierter Datenbankobjekte (z.B. Schema, View etc.) sowie den Tabelleninhalten.</p> <p>Ein Datenbankschema ist eine Art Namespace-Präfix. Ein Datenbankkatalog enthält die Metadaten aller Schemas im Katalog. Die Ebene Katalog in SQL: 2008 entspricht der „Unterlage Datenbank“, die man mit SIARD in ein Archivformat umwandeln kann.</p>
Dauerhafte Archivierung / Langzeitarchivierung	Bezeichnung für die grundsätzlich unbegrenzte Aufbewahrung und die Erhaltung der dauerhaften Verfügbarkeit von digitalen Informationen. Neben der Erhaltung des Bitstroms der archivierten Information fällt darunter auch die Fähigkeit, denselben menschenlesbar und verständlich jederzeit interpretieren und darstellen zu können.
DIP	Dissemination Information Package: Ein DIP ist gemäss OAIS der Behälter für diejenigen Dossiers, welche von einem Benutzer in einem Bestellvorgang bestellt werden.
DNS	Domain Name System: eine verteilte Datenbank, die den Namensraum im Internet verwaltet.
Dossier	Als Dossier gilt die Gesamtheit (Kollektiv) der Unterlagen zu einem Geschäft. Grundsätzlich entspricht ein Dossier einem Geschäft. Durch Zusammenfassen artverwandter Geschäfte bzw. durch Aufteilung von Dossiers in Subdossiers kann diese Grundstruktur aber den jeweiligen Bedürfnissen angepasst werden. Die Dossierbildung erfolgt auf der Grundlage des Ordnungssystems.

Begriff	Beschreibung
Informationspaket	Ein konzeptioneller Container, der sich aus optionaler Inhaltsinformation und optional dazugehörigen Erhaltungsmetadaten zusammensetzt. Zu diesem Informationspaket gehört Verpackungsinformation, welche die Inhaltsinformation und die Paketbeschreibung voneinander abgrenzt und identifiziert sowie die Suche nach der Inhaltsinformation ermöglicht.
LOB	Large Object: generischer Begriff für Zellinhalt einer CLOB-, BLOB- oder XML-Spalte, welcher durch eine separate Datei repräsentiert werden kann.
MD5	Message-Digest Algorithm 5
Metadaten	Metadaten können als «Informationen über die Primärdaten» (Daten über Daten) bezeichnet werden, da sie einen beschreibenden Charakter haben.
OAIS	Open Archival Information System, ISO 14721:2003. Das OAIS beschreibt als Referenzmodell ein Archiv als Organisation, in der Menschen und Systeme mit der Aufgabenstellung zusammenwirken, Informationen zu erhalten und einer definierten Nutzergruppe verfügbar zu machen.
Primärdaten	Primärdaten sind die Daten, welche die inhaltliche Substanz von Unterlagen ausmachen. Innerhalb einer SIARD-Datei nehmen die Tabellendaten die Funktion von Primärdaten ein.
Routinen	SQL-Routinen (auch unter der Bezeichnung Stored Procedures bekannt) sind vor allem zum Verständnis der View-Abfragen wichtig, bei welchen sie in Teilausdrücken vorkommen können.
Schemas	Schemas sind Behälter der Tabellen, Views und Routinen.
SHA1	sicherer Hash-Algorithmus (Secure Hash Algorithm)
SIP	Submission Information Package: SIP sind gemäss OAIS Informationspakete, die von den aktenbildenden Stellen an das Archiv übermittelt werden. Sie enthalten die digitalen Unterlagen (Primärdaten und Metadaten).
Tabellen	Tabellen bestehen aus einer Tabellendefinition mit Feldern, die jeder Spalte der Tabelle einen Namen und einen Typ zuordnen, aus Datensätzen, welche die eigentlichen Tabellendaten enthalten, aus einem optionalen Primärschlüssel, aus Fremdschlüsseln, welche die referenzielle Integrität sicherstellen, aus Kandidatenschlüsseln, welche zur Identifizierung eines Datensatzes dienen, und aus Einschränkungen, welche die Konsistenz garantieren. Optional können zu einer Tabelle sogenannte Triggers (Auslöser) definiert sein.
Unterlagen	Unterlagen sind alle aufgezeichneten Informationen, unabhängig vom Informationsträger, welche bei der Erfüllung öffentlicher Aufgaben empfangen oder erstellt worden sind, sowie alle Hilfsmittel und ergänzenden Daten, die für das Verständnis dieser Informationen und deren Nutzung notwendig sind.
UTF	Unicode Transformation Format

Begriff	Beschreibung
Views	Views sind in der Datenbank gespeicherte Standardabfragen. Das Abfrageresultat ist eine Tabelle, welche ebenfalls Felder und Datensätze enthält.
XSD	XML Schema Definition

Anhang C – Nachweis der verwendeten Standards

eCH-0150	eCH-0150 Change und Release Management von eCH-Standards http://www.ech.ch/
RFC 1738	URL specification – in particular the “file:” URL/URI https://www.ietf.org/rfc/rfc1738.txt
RFC 1951	Specification of the “deflate” algorithm. https://www.ietf.org/rfc/rfc1951.txt
SQL: 2008	ISO/IEC 9075(1-4,9-11,13,14): 2008: Information technology -- Database languages – SQL http://www.iso.org/iso/home/store/catalogue_ics/catalogue_detail_ics.htm?csnumber=53681
Unicode	Unicode 6.1.0 Unicode, Inc. http://www.unicode.org/versions/Unicode6.1.0/ (entspricht ISO/IEC 10646:2012: Information technology -- Universal Coded Character Set (UCS) http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=56921)
XML	Extensible Markup Language (XML), 1.1 (Second Edition) W3C Recommendation 16 August 2006, edited in place 29 September 2006 http://www.w3.org/TR/2006/REC-xml11-20060816/ (entspricht ISO/IEC 19503:2005: Information technology -- XML Metadata Interchange (XMI), http://www.iso.org/iso/home/store/catalogue_tc/catalogue_detail.htm?csnumber=32622)
ZIP	.ZIP File Format Specification, Version 6.3.3 September 1, 2012 PKWARE Inc. http://www.pkware.com/documents/casestudies/APPNOTE.TXT

Anhang D – XML-Schemadefinitionen

D.1 metadata.xsd

Die XML-Schemadefinition `metadata.xsd` definiert die Struktur der Datei `metadata.xml` im Ordner `header/`.

```
<?xml version="1.0" encoding="utf-8" ?>
<!-- $Workfile: metadata.xsd $ =====
Metadata schema for SIARD-E (SIARD 2.0)
Version      : $Id: metadata.xsd 1205 2010-06-17 16:54:52Z hartwig $
Application: Software-Independent Archival of Relational Databases
Platform     : XML 1.0, XML Schema 2001
Description: This XML schema definition defines the structure
              of the metadata in the SIARD format
=====
Copyright  : 2007, 2014, 2015, Swiss Federal Archives, Berne, Switzerland
===== -->
<xs:schema id="metadata"
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns="http://www.bar.admin.ch/xmlns/siard/1.0/metadata.xsd"
  targetNamespace="http://www.bar.admin.ch/xmlns/siard/2.0/metadata.xsd"
  elementFormDefault="qualified"
  attributeFormDefault="unqualified">

  <!-- root element of an XML file conforming to this XML schema -->
  <xs:element name="siardArchive">
    <xs:complexType>
      <xs:annotation>
        <xs:documentation>
          Root element of meta data of the SIARD archive
        </xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <!-- name of the archived database -->
        <xs:element name="dbname" type="mandatoryString"/>
        <!-- short free form description of the database content -->
        <xs:element name="description" type="xs:string" minOccurs="0"/>
        <!-- name of person responsible for archiving the database -->
        <xs:element name="archiver" type="xs:string" minOccurs="0"/>
        <!-- contact data (telephone number or email address) of archiver -->
        <xs:element name="archiverContact" type="xs:string" minOccurs="0"/>
        <!-- name of data owner (section and institution responsible for data)
              of database when it was archived -->
        <xs:element name="dataOwner" type="mandatoryString"/>
        <!-- time span during which data where entered into the database -->
        <xs:element name="dataOriginTimespan" type="mandatoryString"/>
        <!-- root folder for external files (new in version 2.0) -->
        <xs:element name="lobFolder" type="xs:anyURI" minOccurs="0"/>
        <!-- name and version of program that generated the metadata file -->
        <xs:element name="producerApplication" type="xs:string" minOccurs="0"/>
        <!-- date of creation of archive (automatically generated by SIARD) -->
        <xs:element name="archivalDate" type="xs:date"/>
        <!-- message digest code over all primary data in folder "content" -->
        <xs:element name="messageDigest" type="xs:string" minOccurs="0"
maxOccurs="unbounded"/>
        <!-- DNS name of client machine from which SIARD was running for archiving
-->
        <xs:element name="clientMachine" type="xs:string" minOccurs="0"/>
        <!-- name of database product and version from which database originates -
->
        <xs:element name="databaseProduct" type="xs:string" minOccurs="0"/>
        <!-- connection string used for archiving -->
        <xs:element name="connection" type="xs:string" minOccurs="0"/>

```

```

    <!-- database user used for archiving -->
    <xs:element name="databaseUser" type="xs:string" minOccurs="0"/>
    <!-- list of schemas in database -->
    <xs:element name="schemas" type="schemasType"/>
    <!-- list of users in the archived database -->
    <xs:element name="users" type="usersType"/>
    <!-- list of roles in the archived database -->
    <xs:element name="roles" type="rolesType" minOccurs="0"/>
    <!-- list of privileges in the archived database -->
    <xs:element name="privileges" type="privilegesType" minOccurs="0"/>
  </xs:sequence>
  <!-- constraint: version number must be 1.0 or 2.0 -->
  <xs:attribute name="version" type="versionType" use="required" />
</xs:complexType>
</xs:element>

<!-- complex type schemas -->
<xs:complexType name="schemasType">
  <xs:annotation>
    <xs:documentation>
      List of schemas
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="schema" type="schemaType" minOccurs="1"
maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<!-- complex type schema -->
<xs:complexType name="schemaType">
  <xs:annotation>
    <xs:documentation>
      Schema element in siardArchive
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <!-- database name of the schema -->
    <xs:element name="name" type="xs:string" />
    <!-- archive name of the schema folder -->
    <xs:element name="folder" type="fsName"/>
    <!-- description of the schema's meaning and content -->
    <xs:element name="description" type="xs:string" minOccurs="0"/>
    <!-- list of advanced and structured types in the schema (new in version
2.0) -->
    <xs:element name="types" type="typesType" minOccurs="0"/>
    <!-- list of tables in the schema -->
    <xs:element name="tables" type="tablesType"/>
    <!-- list of views in the schema -->
    <xs:element name="views" type="viewsType" minOccurs="0"/>
    <!-- list of routines in the schema -->
    <xs:element name="routines" type="routinesType" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<!-- complex type types (new in version 2.0) -->
<xs:complexType name="typesType">
  <xs:annotation>

```



```

        <xs:documentation>
            List of advanced or structured data types types
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:element name="type" type="typeType" minOccurs="1" maxOccurs="unbounded"
/>
    </xs:sequence>
</xs:complexType>

<!-- complex type type (new in version 2.0) -->
<xs:complexType name="typeType">
    <xs:annotation>
        <xs:documentation>
            Advanced or structured data tape type
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <!-- name of data type -->
        <xs:element name="name" type="xs:string"/>
        <!-- category of data type -->
        <xs:element name="category" type="categoryType"/>
        <!-- schema of supertype -->
        <xs:element name="underSchema" type="xs:string" minOccurs="0"/>
        <!-- name of supertype -->
        <xs:element name="underType" type="xs:string" minOccurs="0"/>
        <!-- instantiability if data type (never true for DISTINCT) -->
        <xs:element name="instantiable" type="xs:boolean"/>
        <!-- finality (always true for DISTINCT, never true for structured UDTs) -->
        <xs:element name="final" type="xs:boolean"/>
        <!-- primitive base SQL:2008 type of (DISTINCT, ARRAY) type -->
        <xs:element name="base" type="xs:string" minOccurs="0"/>
        <!-- SQL_2008 cardinality of ARRAY type -->
        <xs:element name="cardinality" type="xs:integer" minOccurs="0"/>
        <!-- alternatively list of attributes (ROW, UDT) -->
        <xs:element name="attributes" type="attributesType" minOccurs="0"/>
        <!-- description of the parameter's meaning and content -->
        <xs:element name="description" type="xs:string" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>

<!-- complex type attributes (new in version 2.0) -->
<xs:complexType name="attributesType">
    <xs:annotation>
        <xs:documentation>
            List of attributes of a type
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:element name="attribute" type="attributeType" minOccurs="1"
maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<!-- complex type attribute (new in version 2.0) -->
<xs:complexType name="attributeType">
    <xs:annotation>
        <xs:documentation>

```

```

        Attribute of a type
    </xs:documentation>
</xs:annotation>
<xs:sequence>
    <!-- database name of the attribute -->
    <xs:element name="name" type="xs:string" />
    <xs:choice> <!-- new in version 2.0: either built-in or structured -->
        <xs:sequence>
            <!-- SQL:2008 data type of the column -->
            <xs:element name="type" type="builtinTypeType" />
            <!-- original data type of the column -->
            <xs:element name="typeOriginal" type="xs:string" minOccurs="0"/>
        </xs:sequence>
        <xs:sequence>
            <!-- SQL:2008 schema of advanced or structured data type of the
attribute -->
            <xs:element name="typeSchema" type="xs:string" minOccurs="0" />
            <!-- SQL:2008 name of advanced or structured data type of the
attribute -->
            <xs:element name="typeName" type="xs:string" />
            <!-- SQL:2008 attribute list of the column (recursive) -->
            <xs:element name="attributes" type="attributesType" minOccurs="0"/>
        </xs:sequence>
    </xs:choice>
    <!-- default value -->
    <xs:element name="defaultValue" type="xs:string" minOccurs="0"/>
    <!-- description of the attributes's meaning and content -->
    <xs:element name="description" type="xs:string" minOccurs="0"/>
</xs:sequence>
</xs:complexType>

<!-- complex type tables -->
<xs:complexType name="tablesType">
    <xs:annotation>
        <xs:documentation>
            List of tables
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:element name="table" type="tableType" minOccurs="1"
maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<!-- complex type table -->
<xs:complexType name="tableType">
    <xs:annotation>
        <xs:documentation>
            Table element in siardArchive
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <!-- database name of the table -->
        <xs:element name="name" type="xs:string"/>
        <!-- archive name of the table folder -->
        <xs:element name="folder" type="fsName"/>
        <!-- description of the table's meaning and content -->
        <xs:element name="description" type="xs:string" minOccurs="0"/>

```

```

    <!-- list of columns of the table -->
    <xs:element name="columns" type="columnsType"/>
    <!-- primary key -->
    <xs:element name="primaryKey" type="primaryKeyType" minOccurs="0"/>
    <!-- foreign keys -->
    <xs:element name="foreignKeys" type="foreignKeysType" minOccurs="0"/>
    <!-- candidate keys (unique constraints) -->
    <xs:element name="candidateKeys" type="candidateKeysType" minOccurs="0"/>
    <!-- list of (check) constraints -->
    <xs:element name="checkConstraints" type="checkConstraintsType"
minOccurs="0"/>
    <!-- list of triggers -->
    <xs:element name="triggers" type="triggersType" minOccurs="0"/>
    <!-- number of rows in the table -->
    <xs:element name="rows" type="xs:integer"/>
  </xs:sequence>
</xs:complexType>

<!-- complex type views -->
<xs:complexType name="viewsType">
  <xs:annotation>
    <xs:documentation>
      List of views
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="view" type="viewType" minOccurs="1" maxOccurs="unbounded"
/>
  </xs:sequence>
</xs:complexType>

<!-- complex type view -->
<xs:complexType name="viewType">
  <xs:annotation>
    <xs:documentation>
      View element in siardArchive
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <!-- database name of the view -->
    <xs:element name="name" type="xs:string" />
    <!-- SQL query string defining the view -->
    <xs:element name="query" type="xs:string" minOccurs="0"/>
    <!-- original query string defining the view -->
    <xs:element name="queryOriginal" type="xs:string" minOccurs="0"/>
    <!-- description of the view's meaning and content -->
    <xs:element name="description" type="xs:string" minOccurs="0"/>
    <!-- list of columns of the view -->
    <xs:element name="columns" type="columnsType"/>
    <!-- number of rows in the view - added in 2014! -->
    <xs:element name="rows" type="xs:integer" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<!-- complex type columns -->
<xs:complexType name="columnsType">
  <xs:annotation>
    <xs:documentation>

```

```

        List of columns
    </xs:documentation>
</xs:annotation>
<xs:sequence>
    <xs:element name="column" type="columnType" minOccurs="1"
maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>

<!-- complex type column -->
<xs:complexType name="columnType">
    <xs:annotation>
        <xs:documentation>
            Column element in siardArchive
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <!-- database name of the column -->
        <xs:element name="name" type="xs:string" />
        <!-- archive name of the internally stored LOBs within containing
            element -->
        <xs:element name="folder" type="fsName" minOccurs="0"/>
        <!-- folder for LOBs relative to lobFolder of nearest containing
            element for internally or externally stored LOBs
            (new in version 2.0) -->
        <xs:element name="lobFolder" type="xs:anyURI" minOccurs="0"/>
        <xs:choice> <!-- new in version 2.0: either built-in or structured -->
            <xs:sequence>
                <!-- SQL:2008 data type of the column -->
                <xs:element name="type" type="builtinTypeType" />
                <!-- original data type of the column -->
                <xs:element name="typeOriginal" type="xs:string" minOccurs="0"/>
                <!-- nullability (default: true) -->
                <xs:element name="nullable" type="xs:boolean" minOccurs="0"/>
            </xs:sequence>
            <xs:sequence> <!-- new in version 2.0 -->
                <!-- SQL:2008 schema of UDT name of the column (new in version 2.0)
-->
                <xs:element name="typeSchema" type="xs:string" minOccurs="0" />
                <!-- SQL:2008 name of UDT of the column (new in version 2.0) -->
                <xs:element name="typeName" type="xs:string" />
                <!-- SQL:2008 attribute list of the column (new in version 2.0) -->
                <xs:element name="fields" type="fieldsType" minOccurs="0"/>
            </xs:sequence>
        </xs:choice>
        <!-- default value -->
        <xs:element name="defaultValue" type="xs:string" minOccurs="0"/>
        <!-- unique, references, check column constraints
            are stored as table constraints -->
        <!-- mimeType makes sense only for LOBs and is only informative-->
        <xs:element name="mimeType" type="xs:string" minOccurs="0"/>
        <!-- description of the column's meaning and content -->
        <xs:element name="description" type="xs:string" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>

<!-- complex type fields -->
<xs:complexType name="fieldsType">

```

```

<xs:annotation>
  <xs:documentation>
    List of fields of a column or field
  </xs:documentation>
</xs:annotation>
<xs:sequence>
  <xs:element name="field" type="fieldType" minOccurs="1"
maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>

<!-- complex type for type of a column or a field -->
<xs:complexType name="fieldType">
  <xs:annotation>
    <xs:documentation>
      Field element describing the type of a field
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <!-- archive name of the internally stored LOBs within containing
      element -->
    <xs:element name="folder" type="fsName" minOccurs="0"/>
    <!-- folder for LOBs relative to lobFolder of nearest containing
      element for internally or externally stored LOBs
      (new in version 2.0) -->
    <xs:element name="lobFolder" type="xs:anyURI" minOccurs="0"/>
    <!-- SQL:2008 attribute list of the column (new in version 2.0) -->
    <xs:element name="fields" type="fieldsType" minOccurs="0"/>
    <!-- mimeType makes sense only for LOBs and is only informative -->
    <xs:element name="mimeType" type="xs:string" minOccurs="0"/>
    <!-- description of the field's meaning and content -->
    <xs:element name="description" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<!-- complex type primaryKey -->
<xs:complexType name="primaryKeyType">
  <xs:annotation>
    <xs:documentation>
      primaryKey element in siardArchive
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <!-- database name of the primary key -->
    <xs:element name="name" type="xs:string" minOccurs="0" />
    <!-- description of the primary key's meaning and content -->
    <xs:element name="description" type="xs:string" minOccurs="0"/>
    <!-- columns belonging to the primary key -->
    <xs:element name="column" type="xs:string" minOccurs="1"
maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>

<!-- complex type foreignKeys -->
<xs:complexType name="foreignKeysType">
  <xs:annotation>
    <xs:documentation>
      List of foreign key constraints

```

```

        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:element name="foreignKey" type="foreignKeyType" minOccurs="1"
maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<!-- complex type foreignKey -->
<xs:complexType name="foreignKeyType">
    <xs:annotation>
        <xs:documentation>
            foreignKey element in siardArchive
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <!-- database name of the foreign key -->
        <xs:element name="name" type="xs:string" />
        <!-- referenced schema -->
        <xs:element name="referencedSchema" type="xs:string"/>
        <!-- referenced table -->
        <xs:element name="referencedTable" type="xs:string"/>
        <!-- references -->
        <xs:element name="reference" type="referenceType" minOccurs="1"
maxOccurs="unbounded"/>
        <!-- match type (FULL, PARTIAL, SIMPLE) -->
        <xs:element name="matchType" type="matchTypeType" minOccurs="0"/>
        <!-- ON DELETE action e.g. ON DELETE CASCADE -->
        <xs:element name="deleteAction" type="xs:string" minOccurs="0"/>
        <!-- ON UPDATE action e.g. ON UPDATE SET DEFAULT -->
        <xs:element name="updateAction" type="xs:string" minOccurs="0"/>
        <!-- description of the foreign key's meaning and content -->
        <xs:element name="description" type="xs:string" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>

<!-- complex type reference -->
<xs:complexType name="referenceType">
    <xs:annotation>
        <xs:documentation>
            reference element in siardArchive
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <!-- referencing column -->
        <xs:element name="column" type="xs:string"/>
        <!-- referenced column (table.column) -->
        <xs:element name="referenced" type="xs:string"/>
    </xs:sequence>
</xs:complexType>

<!-- complex type candidateKeys -->
<xs:complexType name="candidateKeysType">
    <xs:annotation>
        <xs:documentation>
            List of candidate key (unique) constraints
        </xs:documentation>
    </xs:annotation>

```

```

    <xs:sequence>
      <xs:element name="candidateKey" type="candidateKeyType" minOccurs="1"
maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>

  <!-- complex type candidateKey -->
  <xs:complexType name="candidateKeyType">
    <xs:annotation>
      <xs:documentation>
        candidate key (unique) element in siardArchive
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <!-- database name of the candidate key -->
      <xs:element name="name" type="xs:string"/>
      <!-- description of the candidate key's meaning and content -->
      <xs:element name="description" type="xs:string" minOccurs="0"/>
      <!-- columns belonging to the candidate key -->
      <xs:element name="column" type="xs:string" minOccurs="1"
maxOccurs="unbounded"/>
    </xs:sequence>
  </xs:complexType>

  <!-- complex type check constraints -->
  <xs:complexType name="checkConstraintsType">
    <xs:annotation>
      <xs:documentation>
        List of check constraints
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="checkConstraint" type="checkConstraintType" minOccurs="1"
maxOccurs="unbounded" />
    </xs:sequence>
  </xs:complexType>

  <!-- complex type check constraint -->
  <xs:complexType name="checkConstraintType">
    <xs:annotation>
      <xs:documentation>
        Check constraint element in siardArchive
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <!-- database name of the constraint -->
      <xs:element name="name" type="xs:string"/>
      <!-- check condition -->
      <xs:element name="condition" type="xs:string"/>
      <!-- description of the constraint's meaning and content -->
      <xs:element name="description" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>

  <!-- complex type triggers -->
  <xs:complexType name="triggersType">
    <xs:annotation>
      <xs:documentation>

```



```

        List of triggers
    </xs:documentation>
</xs:annotation>
<xs:sequence>
    <xs:element name="trigger" type="triggerType" minOccurs="1"
maxOccurs="unbounded" />
</xs:sequence>
</xs:complexType>

<!-- complex type trigger -->
<xs:complexType name="triggerType">
    <xs:annotation>
        <xs:documentation>
            Trigger element in siardArchive
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <!-- database name of the trigger -->
        <xs:element name="name" type="xs:string" />
        <!-- action time -->
        <xs:element name="actionTime" type="actionTimeType"/>
        <!-- trigger event INSERT, DELETE, UPDATE [OF <trigger column list>] -->
        <xs:element name="triggerEvent" type="xs:string"/>
        <!-- alias list <old or new values alias> -->
        <xs:element name="aliasList" type="xs:string" minOccurs="0"/>
        <!-- triggered action -->
        <xs:element name="triggeredAction" type="xs:string"/>
        <!-- description of the trigger's meaning and content -->
        <xs:element name="description" type="xs:string" minOccurs="0"/>
    </xs:sequence>
</xs:complexType>

<!-- complex type routines -->
<xs:complexType name="routinesType">
    <xs:annotation>
        <xs:documentation>
            List of routines
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <xs:element name="routine" type="routineType" minOccurs="1"
maxOccurs="unbounded" />
    </xs:sequence>
</xs:complexType>

<!-- complex type routine -->
<xs:complexType name="routineType">
    <xs:annotation>
        <xs:documentation>
            Routine
        </xs:documentation>
    </xs:annotation>
    <xs:sequence>
        <!-- database name of routine in schema -->
        <xs:element name="name" type="xs:string"/>
        <!-- description of the routines's meaning and content -->
        <xs:element name="description" type="xs:string" minOccurs="0"/>
        <!-- original source code (VBA, PL/SQL, ...) defining the routine -->

```



```

    <xs:element name="source" type="xs:string" minOccurs="0"/>
    <!-- SQL:2008 body of routine -->
    <xs:element name="body" type="xs:string" minOccurs="0"/>
    <!-- routine characteristic -->
    <xs:element name="characteristic" type="xs:string" minOccurs="0"/>
    <!-- SQL:2008 data type of the return value (for functions) -->
    <xs:element name="returnType" type="xs:string" minOccurs="0"/>
    <!-- list of parameters -->
    <xs:element name="parameters" type="parametersType" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<!-- complex type parameters -->
<xs:complexType name="parametersType">
  <xs:annotation>
    <xs:documentation>
      List of parameters of a routine
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="parameter" type="parameterType" minOccurs="1"
maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<!-- complex type parameter -->
<xs:complexType name="parameterType">
  <xs:annotation>
    <xs:documentation>
      Parameter of a routine
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <!-- name of parameter -->
    <xs:element name="name" type="xs:string"/>
    <!-- mode of parameter (IN, OUT, INOUT) -->
    <xs:element name="mode" type="xs:string"/>
    <xs:choice> <!-- new in version 2.0: either built-in or structured -->
      <xs:sequence>
        <!-- SQL:2008 data type of the column -->
        <xs:element name="type" type="builtinTypeType" />
        <!-- original data type of the column -->
        <xs:element name="typeOriginal" type="xs:string" minOccurs="0"/>
      </xs:sequence>
      <xs:sequence> <!-- new in version 2.0 -->
        <!-- SQL:2008 schema of UDT name of the column (new in version 2.0)
-->
        <xs:element name="typeSchema" type="xs:string" minOccurs="0" />
        <!-- SQL:2008 name of UDT of the column (new in version 2.0) -->
        <xs:element name="typeName" type="xs:string" />
        <!-- SQL:2008 attribute list of the column (new in version 2.0) -->
        <xs:element name="parameterFields" type="parameterFieldsType"
minOccurs="0"/>
      </xs:sequence>
    </xs:choice>
    <!-- description of the parameter's meaning and content -->
    <xs:element name="description" type="xs:string" minOccurs="0"/>
  </xs:sequence>

```

```

</xs:complexType>

<!-- complex type parameterFields (new in version 2.0) -->
<xs:complexType name="parameterFieldsType">
  <xs:annotation>
    <xs:documentation>
      List of fields of a parameter
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="parameterField" type="parameterFieldType" minOccurs="1"
maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<!-- complex type for type of a parameter or a parameter field (new in version
2.0) -->
<xs:complexType name="parameterFieldType">
  <xs:annotation>
    <xs:documentation>
      Field element describing the type of a parameter or a parameter field
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <!-- SQL:2008 attribute list of the column (new in version 2.0) -->
    <xs:element name="parameterFields" type="parameterFieldsType"
minOccurs="0"/>
    <!-- description of the parameter field's meaning and content -->
    <xs:element name="description" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<!-- complex type users -->
<xs:complexType name="usersType">
  <xs:annotation>
    <xs:documentation>
      List of users
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="user" type="userType" minOccurs="1" maxOccurs="unbounded"
/>
  </xs:sequence>
</xs:complexType>

<!-- complex type user -->
<xs:complexType name="userType">
  <xs:annotation>
    <xs:documentation>
      User
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <!-- user name -->
    <xs:element name="name" type="xs:string"/>
    <!-- description of the user's meaning and content -->
    <xs:element name="description" type="xs:string" minOccurs="0"/>
  </xs:sequence>

```

```

</xs:complexType>

<!-- complex type roles -->
<xs:complexType name="rolesType">
  <xs:annotation>
    <xs:documentation>
      List of roles
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="role" type="roleType" minOccurs="1" maxOccurs="unbounded"
  />
  </xs:sequence>
</xs:complexType>

<!-- complex type role -->
<xs:complexType name="roleType">
  <xs:annotation>
    <xs:documentation>
      Role
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <!-- role name -->
    <xs:element name="name" type="xs:string"/>
    <!-- role ADMIN (user or role) -->
    <xs:element name="admin" type="xs:string"/>
    <!-- description of the role's meaning and content -->
    <xs:element name="description" type="xs:string" minOccurs="0"/>
  </xs:sequence>
</xs:complexType>

<!-- complex type privileges -->
<xs:complexType name="privilegesType">
  <xs:annotation>
    <xs:documentation>
      List of grants
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:element name="privilege" type="privilegeType" minOccurs="1"
maxOccurs="unbounded" />
  </xs:sequence>
</xs:complexType>

<!-- complex type privilege -->
<xs:complexType name="privilegeType">
  <xs:annotation>
    <xs:documentation>
      Grant (incl. grant of role)
    </xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <!-- privilege type (incl. ROLE privilege or "ALL PRIVILEGES" -->
    <xs:element name="type" type="xs:string"/>
    <!-- privilege object (may be omitted for ROLE privilege) -->
    <xs:element name="object" type="xs:string" minOccurs="0"/>
    <!-- GRANTED BY -->

```

```

<xs:element name="grantor" type="xs:string"/>
<!-- user list of users or roles or single value "PUBLIC" -->
<xs:element name="grantee" type="xs:string"/>
<!-- optional option "GRANT" or "ADMIN" -->
<xs:element name="option" type="privOptionType" minOccurs="0"/>
<!-- description of the grant's meaning and content -->
<xs:element name="description" type="xs:string" minOccurs="0"/>
</xs:sequence>
</xs:complexType>

<xs:simpleType name="builtinTypeType">
  <xs:annotation>
    <xs:documentation>
      builtinTypeType is constrained to valid SQL:2008 data type values
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <!-- exact numerics (BIGINT from SQL:2008) -->
    <xs:pattern value="INTEGER/INT/SMALLINT/BIGINT"/>
    <xs:pattern value="NUMERIC(\([1-9]\d*(,[1-9]\d*)?\))?" />
    <xs:pattern value="DECIMAL(\([1-9]\d*(,[1-9]\d*)?\))?" />
    <!-- approximate numerics -->
    <xs:pattern value="REAL/DOUBLE PRECISION"/>
    <xs:pattern value="FLOAT(\([1-9]\d*\))?" />
    <!-- character strings -->
    <xs:pattern value="(CHARACTER/CHAR)(\([1-9]\d*\))?" />
    <xs:pattern value="(CHARACTER VARYING/CHAR VARYING/VARCHAR)(\([1-9]\d*\))?" />
    <xs:pattern value="(CHARACTER LARGE OBJECT/CLOB)(\([1-9]\d*(K|M|G)?\))?" />
    <xs:pattern value="(NATIONAL CHARACTER/NATIONAL CHAR/NCHAR)(\([1-
9]\d*\))?" />
    <xs:pattern value="(NATIONAL CHARACTER VARYING/NATIONAL CHAR VARYING/NCHAR
VARYING)(\([1-9]\d*\))?" />
    <xs:pattern value="(NATIONAL CHARACTER LARGE OBJECT/NCHAR LARGE
OBJECT/NCLOB)(\([1-9]\d*(K|M|G)?\))?" />
    <xs:pattern value="XML" />
    <!-- BIT and BINARY strings -->
    <xs:pattern value="BIT(\([1-9]\d*\))?" />
    <xs:pattern value="BIT VARYING(\([1-9]\d*\))?" />
    <xs:pattern value="(BINARY LARGE OBJECT/BLOB)(\([1-9]\d*(K|M|G)?\))?" />
    <!-- BINARY strings from SQL:2008 -->
    <xs:pattern value="BINARY(\([1-9]\d*\))?" />
    <xs:pattern value="(BINARY VARYING/VARBINARY)(\([1-9]\d*\))?" />
    <!-- datetimes -->
    <xs:pattern value="DATE" />
    <xs:pattern value="(TIME/TIME WITH TIME ZONE)(\([1-9]\d*\))?" />
    <xs:pattern value="(TIMESTAMP/TIMESTAMP WITH TIME ZONE)(\([1-9]\d*\))?" />
    <!-- intervals -->
    <xs:pattern value="INTERVAL (YEAR/MONTH/DAY/HOUR/MINUTE/SECOND)(\([1-
9]\d*\))?( TO (MONTH/DAY/HOUR/MINUTE/SECOND)(\([1-9]\d*\))?)?" />
    <!-- BOOLEAN -->
    <xs:pattern value="BOOLEAN" />
  </xs:restriction>
</xs:simpleType>

<!-- simple type for version number -->
<xs:simpleType name="versionType">

```

```
<xs:annotation>
  <xs:documentation>
    valueType must be constrained to "1.0" or "2.0"
    for conformity with this XML schema
  </xs:documentation>
</xs:annotation>
<xs:restriction base="xs:string">
  <xs:whiteSpace value="collapse"/>
  <xs:enumeration value="1.0"/>
  <xs:enumeration value="2.0"/>
</xs:restriction>
</xs:simpleType>

<!-- simple type for privilege option -->
<xs:simpleType name="privOptionType">
  <xs:annotation>
    <xs:documentation>
      privOptionType must be "ADMIN" or "GRANT"
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:whiteSpace value="collapse"/>
    <xs:enumeration value="ADMIN"/>
    <xs:enumeration value="GRANT"/>
  </xs:restriction>
</xs:simpleType>

<!-- simple type for mandatory string
  which must contain at least 1 character -->
<xs:simpleType name="mandatoryString">
  <xs:annotation>
    <xs:documentation>
      mandatoryString must contain at least 1 character
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:whiteSpace value="preserve"/>
    <xs:minLength value="1" />
  </xs:restriction>
</xs:simpleType>

<!-- simple type of a filesystem (file or folder) name -->
<xs:simpleType name="fsName">
  <xs:annotation>
    <xs:documentation>
      fsNames may only consist of ASCII characters and digits
      and must start with a non-digit
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:pattern value="([a-z]/[A-Z])([a-z]/[A-Z]/[0-9]).*" />
    <xs:minLength value="1" />
  </xs:restriction>
</xs:simpleType>

<!-- simple type for action time of a trigger -->
<xs:simpleType name="actionTimeType">
  <xs:annotation>
```

```

    <xs:documentation>
      actionTime is BEFORE or AFTER
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:enumeration value="BEFORE" />
    <xs:enumeration value="AFTER" />
  </xs:restriction>
</xs:simpleType>

<!-- simple type for match type of a foreign key -->
<xs:simpleType name="matchTypeType">
  <xs:annotation>
    <xs:documentation>
      matchType is FULL, PARTIAL or SIMPLE
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:enumeration value="FULL" />
    <xs:enumeration value="PARTIAL" />
    <xs:enumeration value="SIMPLE" />
  </xs:restriction>
</xs:simpleType>

<!-- simple type for the category of a column or a parameter (new in version
2.0) -->
<xs:simpleType name="categoryType">
  <xs:annotation>
    <xs:documentation>
      category of advanced or structured data types is "distinct",
      "row", "array", or "udt"
      for conformity with this XLM schema
    </xs:documentation>
  </xs:annotation>
  <xs:restriction base="xs:string">
    <xs:whiteSpace value="collapse"/>
    <xs:enumeration value="distinct"/>
    <xs:enumeration value="row"/>
    <xs:enumeration value="array"/>
    <xs:enumeration value="udt"/>
  </xs:restriction>
</xs:simpleType>

</xs:schema>

```

D.2 Beispiel für metadata.xml

Eine zum XML-Schema für SIARD konforme Metadatenbeschreibung einer Datenbank sieht beispielsweise so aus:

```

<?xml version="1.0" encoding="utf-8" standalone="no"?>
<?xml-stylesheet type="text/xsl" href="metadata.xsl"?>
<siardArchive
  xmlns="http://www.bar.admin.ch/xmlns/siard/2.0/metadata.xsd"
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"

```

```
xsi:schemaLocation="http://www.bar.admin.ch/xmlns/siard/1.0/metadata.xsd
metadata.xsd"
version="2.0">
<dbname>SIARD Format 2 with SQL:2008 Standard Types</dbname>
<dataOwner>Enter AG, Zurich</dataOwner>
<dataOriginTimespan>2015</dataOriginTimespan>
<lobFolder>file:///D:/Projekte/SIARD/SIARD%20Suite/</lobFolder>
<producerApplication>Manually constructed</producerApplication>
<archivalDate>2015-01-19</archivalDate>
<messageDigest>MD5D1C411FC45542DCA86EEB1CC9B4B596C</messageDigest>
<clientMachine>celsius.enterag.ch</clientMachine>
<databaseProduct>SQL:2011</databaseProduct>
<connection>jdbc:manually:SQL:2011</connection>
<databaseUser>SIARDLOGIN</databaseUser>
<schemas>
  <schema>
    <name>SIARDSHEMA</name>
    <folder>schema0</folder>

    <!-- version 2.0 advanced or structured types -->
    <types>
      <type>
        <name>TDISTINCT</name>
        <category>distinct</category>
        <instantiable>false</instantiable>
        <final>true</final>
        <base>INTEGER</base>
        <description>Example of a DISTINCT type</description>
      </type>
      <type>
        <name>TROW</name>
        <category>row</category>
        <instantiable>true</instantiable>
        <final>true</final>
        <attributes>
          <attribute>
            <name>TAPEID</name>
            <type>INTEGER</type>
            <defaultValue>0</defaultValue>
          </attribute>
          <attribute>
            <name>TRANSCRIPTION</name>
            <type>CLOB</type>
          </attribute>
          <attribute>
            <name>SOUND</name>
            <type>BLOB</type>
          </attribute>
        </attributes>
        <description>Example of a ROW type</description>
      </type>
      <type>
        <name>TARRAY</name>
        <category>array</category>
        <instantiable>false</instantiable>
        <final>true</final>
        <base>CHARACTER VARYING(255)</base>
        <cardinality>4</cardinality>
```

```
<description>Example of an ARRAY type</description>
</type>
<type>
  <name>TUDT</name>
  <category>udt</category>
  <instantiable>true</instantiable>
  <final>true</final>
  <attributes>
    <attribute>
      <name>ID</name>
      <type>INTEGER</type>
    </attribute>
    <attribute>
      <name>NESTEDROW</name>
      <typeName>TROW</typeName>
      <attributes>
        <attribute>
          <name>first</name>
          <type>INTEGER</type>
        </attribute>
        <attribute>
          <name>second</name>
          <type>VARCHAR(255)</type>
        </attribute>
      </attributes>
    </attribute>
    <description>Example of nested attributes</description>
  </attribute>
</attributes>
<description>Example of a UDT</description>
</type>
</types>

<tables>
  <!-- version 1.0 table -->
  <table>
    <name>TABLETEST1</name>
    <folder>table0</folder>
    <description />
    <columns>
      <!-- binary types -->
      <column>
        <name>CBIT</name>
        <type>BIT</type>
        <typeOriginal>BIT</typeOriginal>
        <nullable>true</nullable>
      </column>
      <column>
        <name>CBIT_32</name>
        <type>BIT(32)</type>
        <typeOriginal>BIT(32)</typeOriginal>
        <nullable>true</nullable>
      </column>
      <column>
        <name>CBIT_VARYING_160</name>
        <type>BIT VARYING(160)</type>
        <typeOriginal>BIT VARYING(160)</typeOriginal>
        <nullable>true</nullable>
      </column>
    </columns>
  </table>
</tables>
```



```
<column>
  <name>CBINARY</name>
  <type>BINARY</type>
  <typeOriginal>BINARY</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CBINARY_5</name>
  <type>BINARY(5)</type>
  <typeOriginal>BINARY(5)</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CBINARY_VARYING_32</name>
  <type>BINARY VARYING(32)</type>
  <typeOriginal>BINARY VARYING(32)</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CBINARY_LARGE_OBJECT</name>
  <!-- <folder>lob8</folder> -->
  <lobFolder>schema0/table0/lob8/</lobFolder>
  <type>BINARY LARGE OBJECT</type>
  <typeOriginal>BINARY LARGE OBJECT</typeOriginal>
  <nullable>true</nullable>
</column>
<!-- character types -->
<column>
  <name>CCHARACTER</name>
  <type>CHARACTER</type>
  <typeOriginal>CHARACTER</typeOriginal>
  <nullable>false</nullable>
</column>
<column>
  <name>CCHARACTER_5</name>
  <type>CHARACTER(5)</type>
  <typeOriginal>CHARACTER(5)</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CCHARACTER_VARYING_32</name>
  <type>CHARACTER VARYING(32)</type>
  <typeOriginal>CHARACTER VARYING(32)</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CCHARACTER_LARGE_OBJECT</name>
  <!-- <folder>lob12</folder> -->
  <lobFolder>schema0/table0/lob12/</lobFolder>
  <type>CHARACTER LARGE OBJECT</type>
  <typeOriginal>CHARACTER LARGE OBJECT</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CNATIONAL_CHARACTER</name>
  <type>NATIONAL CHARACTER</type>
  <typeOriginal>NATIONAL_CHARACTER</typeOriginal>
  <nullable>true</nullable>
```

```
</column>
<column>
  <name>CNATIONAL_CHARACTER_5</name>
  <type>NATIONAL CHARACTER(5)</type>
  <typeOriginal>NATIONAL_CHARACTER(5)</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CNATIONAL_CHARACTER_VARYING_32</name>
  <type>NATIONAL CHARACTER VARYING(32)</type>
  <typeOriginal>NATIONAL_CHARACTER VARYING(32)</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CNATIONAL_CHARACTER_LARGE_OBJECT</name>
  <!-- <folder>lob16</folder> -->
  <lobFolder>schema0/table0/lob16/</lobFolder>
  <type>NATIONAL CHARACTER LARGE OBJECT</type>
  <typeOriginal>NATIONAL_CHARACTER LARGE OBJECT</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CXML</name>
  <!-- <folder>lob17</folder> -->
  <lobFolder>schema0/table0/lob17/</lobFolder>
  <type>XML</type>
  <typeOriginal>XML</typeOriginal>
  <nullable>true</nullable>
</column>
<!-- integer types -->
<column>
  <name>CSMALLINT</name>
  <type>SMALLINT</type>
  <typeOriginal>SMALLINT</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CINTEGER</name>
  <type>INTEGER</type>
  <typeOriginal>INTEGER</typeOriginal>
  <nullable>false</nullable>
</column>
<column>
  <name>CBIGINT</name>
  <type>BIGINT</type>
  <typeOriginal>BIGINT</typeOriginal>
  <nullable>true</nullable>
</column>
<!-- fixed numeric types -->
<column>
  <name>CDECIMAL</name>
  <type>DECIMAL</type>
  <typeOriginal>DECIMAL</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CDECIMAL_3</name>
  <type>DECIMAL(3)</type>
```

```
<typeOriginal>DECIMAL(3)</typeOriginal>
<nullable>true</nullable>
</column>
<column>
  <name>CDECIMAL_5_2</name>
  <type>DECIMAL(5,2)</type>
  <typeOriginal>DECIMAL(5,2)</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CNUMERIC</name>
  <type>NUMERIC</type>
  <typeOriginal>NUMERIC</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CNUMERIC_3</name>
  <type>NUMERIC(3)</type>
  <typeOriginal>NUMERIC(3)</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CNUMERIC_5_2</name>
  <type>NUMERIC(5,2)</type>
  <typeOriginal>NUMERIC(5,2)</typeOriginal>
  <nullable>true</nullable>
</column>
<!-- approximate numerics types -->
<column>
  <name>CREAL</name>
  <type>REAL</type>
  <typeOriginal>REAL</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CFLOAT</name>
  <type>FLOAT</type>
  <typeOriginal>FLOAT</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CFLOAT_13</name>
  <type>FLOAT(13)</type>
  <typeOriginal>FLOAT(13)</typeOriginal>
  <nullable>true</nullable>
</column>
<column>
  <name>CDOUBLE_PRECISION</name>
  <type>DOUBLE PRECISION</type>
  <typeOriginal>DOUBLE PRECISION</typeOriginal>
  <nullable>true</nullable>
</column>
<!-- date and time types -->
<column>
  <name>CDATE</name>
  <type>DATE</type>
  <typeOriginal>DATE</typeOriginal>
  <nullable>true</nullable>
```

```

    </column>
    <column>
      <name>CTIME</name>
      <type>TIME</type>
      <typeOriginal>TIME</typeOriginal>
      <nullable>true</nullable>
    </column>
    <column>
      <name>CTIME_5</name>
      <type>TIME(5)</type>
      <typeOriginal>TIME(5)</typeOriginal>
      <nullable>true</nullable>
    </column>
    <column>
      <name>CTIMESTAMP</name>
      <type>TIMESTAMP</type>
      <typeOriginal>TIMESTAMP</typeOriginal>
      <nullable>true</nullable>
    </column>
    <column>
      <name>CTIMESTAMP_7</name>
      <type>TIMESTAMP(7)</type>
      <typeOriginal>TIMESTAMP(7)</typeOriginal>
      <nullable>true</nullable>
    </column>
  </columns>
  <primaryKey>
    <name>TABLETEST1PK</name>
    <column>CCHARACTER</column>
    <column>CINTEGER</column>
  </primaryKey>
  <rows>1</rows>
</table>

<!-- advanced table -->
<table>
  <name>TABLETEST2</name>
  <folder>table1</folder>
  <description />
  <columns>
    <column>
      <name>ID</name>
      <type>INTEGER</type>
      <nullable>false</nullable>
    </column>
    <column>
      <name>CDISTINCT</name>
      <typeName>TDISTINT</typeName>
    </column>
    <column>
      <name>CROW</name>
      <lobFolder>lob2</lobFolder>
      <typeName>TROW</typeName>
      <fields>
        <!-- TAPEID -->
        <field>
          <description>Tape ID</description>
        </field>
      </fields>
    </column>
  </columns>

```

```

<!-- TRANSCRIPTION -->
<field>
  <lobFolder>field1</lobFolder>
  <description>Tape transcription</description>
</field>
<!-- SOUND -->
<field>
  <lobFolder>field2</lobFolder>
  <mimeType>audio/mpeg</mimeType>
  <description>Digitized tape</description>
</field>
</fields>
<description>Tape recordings</description>
</column>
<column>
  <name>CARRAY</name>
  <typeName>TARRAY</typeName>
  <description>Up to 4 fields for complex foreign names</description>
</column>
<column>
  <name>CU DT</name>
  <lobFolder>lob4</lobFolder>
  <typeName>TU DT</typeName>
  <fields>
    <!-- ID -->
    <field></field>
    <!-- NESTEDROW -->
    <field>
      <lobFolder>field1</lobFolder>
      <fields>
        <!-- TAPEID -->
        <field>
          <description>Tape ID</description>
        </field>
        <!-- TRANSCRIPTION -->
        <field>
          <lobFolder>field1</lobFolder>
          <description>Tape transcription</description>
        </field>
        <!-- SOUND -->
        <field>
          <lobFolder>field2</lobFolder>
          <mimeType>audio/mpeg</mimeType>
          <description>Digitized tape</description>
        </field>
      </fields>
      <description>Tape recordings nested</description>
    </field>
  </fields>
  <description>UDT type with field of advanced type</description>
</column>
</columns>
<primaryKey>
  <name>TABLETEST2PK</name>
  <column>ID</column>
</primaryKey>
<rows>1000000</rows>
</table>

```

```
    </tables>
  </schema>
</schemas>
<users>
  <user>
    <name>SIARDLOGIN</name>
  </user>
  <user>
    <name>SIARDUSER</name>
  </user>
</users>
<roles>
  <role>
    <name>public</name>
    <admin />
  </role>
</roles>
<privileges>
  <privilege>
    <type>UPDATE</type>
    <object>TABLE SIARDSHEMA.TABLETEST2</object>
    <grantor>dbo</grantor>
    <grantee>SIARDUSER</grantee>
    <option>ADMIN</option>
  </privilege>
  <privilege>
    <type>INSERT</type>
    <object>TABLE SIARDSHEMA.TABLETEST2</object>
    <grantor>dbo</grantor>
    <grantee>SIARDUSER</grantee>
    <option>ADMIN</option>
  </privilege>
  <privilege>
    <type>REFERENCES</type>
    <object>TABLE SIARDSHEMA.TABLETEST2</object>
    <grantor>dbo</grantor>
    <grantee>SIARDUSER</grantee>
    <option>ADMIN</option>
  </privilege>
  <privilege>
    <type>SELECT</type>
    <object>TABLE SIARDSHEMA.TABLETEST2</object>
    <grantor>dbo</grantor>
    <grantee>SIARDUSER</grantee>
    <option>ADMIN</option>
  </privilege>
  <privilege>
    <type>DELETE</type>
    <object>TABLE SIARDSHEMA.TABLETEST2</object>
    <grantor>dbo</grantor>
    <grantee>SIARDUSER</grantee>
    <option>ADMIN</option>
  </privilege>
</privileges>
</siardArchive>
```

D.3 Beispiel für die XML-Schemadefinition einer Tabelle: table0.xsd

Für jede Tabelle wird von SIARD eine XML-Schemadefinition erzeugt, welche den Spalten die richtigen XML-Datentypen zuordnet.

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns="http://www.admin.ch/xmlns/siard/2.0/schema0/table0.xsd"
  attributeFormDefault="unqualified"
  elementFormDefault="qualified"
  targetNamespace="http://www.admin.ch/xmlns/siard/2.0/schema0/table0.xsd">
  <xs:element name="table">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" minOccurs="0" name="row"
type="rowType"/></xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="rowType">
    <xs:sequence>
      <xs:element minOccurs="0" name="c1" type="xs:boolean" /> <!-- CBOOLEAN -->
      <xs:element minOccurs="0" name="c2" type="xs:hexBinary" /> <!-- CBIT -->
      <xs:element minOccurs="0" name="c3" type="xs:hexBinary" /> <!-- CBIT_32 -->
      <xs:element minOccurs="0" name="c4" type="xs:hexBinary" /> <!--
CBIT_VARYING_160 -->
      <xs:element minOccurs="0" name="c5" type="xs:hexBinary" /> <!-- CBINARY -->
      <xs:element minOccurs="0" name="c6" type="xs:hexBinary" /> <!-- CBINARY_5 -
->
      <xs:element minOccurs="0" name="c7" type="xs:hexBinary" /> <!--
CBINARY_VARYING_32 -->
      <xs:element minOccurs="0" name="c8" type="blobType" /> <!--
CBINARY_LARGE_OBJECT -->
      <xs:element
        name="c9" type="xs:string" /> <!-- CCHARACTER -->
      <xs:element minOccurs="0" name="c10" type="xs:string" /> <!-- CCHARACTER_5 -
->
      <xs:element minOccurs="0" name="c11" type="xs:string" /> <!--
CCHARACTER_VARYING_32 -->
      <xs:element minOccurs="0" name="c12" type="clobType" /> <!--
CCHARACTER_LARGE_OBJECT -->
      <xs:element minOccurs="0" name="c13" type="xs:string" /> <!--
CNATIONAL_CHARACTER -->
      <xs:element minOccurs="0" name="c14" type="xs:string" /> <!--
CNATIONAL_CHARACTER_5 -->
      <xs:element minOccurs="0" name="c15" type="xs:string" /> <!--
CNATIONAL_CHARACTER_VARYING_32 -->
      <xs:element minOccurs="0" name="c16" type="clobType" /> <!--
CNATIONAL_CHARACTER_LARGE_OBJECT -->
      <xs:element minOccurs="0" name="c17" type="clobType" /> <!-- CXML -->
      <xs:element minOccurs="0" name="c18" type="xs:integer" /> <!-- CSMALLINT -->
      <xs:element
        name="c19" type="xs:integer" /> <!-- CINTEGER -->
      <xs:element minOccurs="0" name="c20" type="xs:integer" /> <!-- CBIGINT -->
      <xs:element minOccurs="0" name="c21" type="xs:decimal" /> <!-- CDECIMAL -->
      <xs:element minOccurs="0" name="c22" type="xs:decimal" /> <!-- CDECIMAL_3 --
>
```

```

    <xs:element minOccurs="0" name="c23" type="xs:decimal" /> <!-- CDECIMAL_5_2
-->
    <xs:element minOccurs="0" name="c24" type="xs:decimal" /> <!-- CNUMERIC -->
    <xs:element minOccurs="0" name="c25" type="xs:decimal" /> <!-- CNUMERIC_3 --
>
    <xs:element minOccurs="0" name="c26" type="xs:decimal" /> <!-- CNUMERIC_5_2
-->
    <xs:element minOccurs="0" name="c27" type="xs:float" /> <!-- CREAL -->
    <xs:element minOccurs="0" name="c28" type="xs:float" /> <!-- CFLOAT -->
    <xs:element minOccurs="0" name="c29" type="xs:float" /> <!-- CFLOAT_13 -->
    <xs:element minOccurs="0" name="c30" type="xs:float" /> <!--
CDOUBLE_PRECISION -->
    <xs:element minOccurs="0" name="c31" type="dateTime" /> <!-- CDATE -->
    <xs:element minOccurs="0" name="c32" type="xs:time" /> <!-- CTIME -->
    <xs:element minOccurs="0" name="c33" type="xs:time" /> <!-- CTIME_5 -->
    <xs:element minOccurs="0" name="c34" type="dateTimeType" /> <!-- CTIMESTAMP
-->
    <xs:element minOccurs="0" name="c35" type="dateTimeType" /> <!--
CTIMESTAMP_7 -->
  </xs:sequence>
</xs:complexType>
<xs:complexType name="clobType">
  <xs:simpleContent>
    <xs:extension base="xs:string">
      <xs:attribute name="file" type="xs:anyURI" />
      <xs:attribute name="length" type="xs:integer" />
      <xs:attribute name="digestType" type="digestTypeType" />
      <xs:attribute name="messageDigest" type="xs:string"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<xs:complexType name="blobType">
  <xs:simpleContent>
    <xs:extension base="xs:hexBinary">
      <xs:attribute name="file" type="xs:anyURI" use="required" />
      <xs:attribute name="length" type="xs:integer" use="required" />
      <xs:attribute name="digestType" type="digestTypeType" />
      <xs:attribute name="messageDigest" type="xs:string"/>
    </xs:extension>
  </xs:simpleContent>
</xs:complexType>
<!-- type for message digest type -->
<xs:simpleType name="digestTypeType">
  <xs:restriction base="xs:string">
    <xs:whiteSpace value="collapse"/>
    <xs:enumeration value="MD5"/>
    <xs:enumeration value="SHA-1"/>
    <xs:enumeration value="SHA-256"/>
  </xs:restriction>
</xs:simpleType>
<!-- date type between 0001 and 9999 restricted to UTC -->
<xs:simpleType name="dateTime">
  <xs:restriction base="xs:date">
    <xs:minInclusive value="0001-01-01Z"/>
    <xs:maxExclusive value="10000-01-01Z"/>
    <xs:pattern value="\d{4}-\d{2}-\d{2}Z"/>
  </xs:restriction>
</xs:simpleType>

```



```
<!-- time type restricted to UTC -->
<xs:simpleType name="timeType">
  <xs:restriction base="xs:time">
    <xs:pattern value="\d{2}:\d{2}:\d{2}(\.\d+)?Z?" />
  </xs:restriction>
</xs:simpleType>
<!-- dateTime type between 0001 and 9999 restricted to UTC -->
<xs:simpleType name="dateTimeType">
  <xs:restriction base="xs:dateTime">
    <xs:minInclusive value="0001-01-01T00:00:00.000000000Z" />
    <xs:maxExclusive value="1000-01-01T00:00:00.000000000Z" />
    <xs:pattern value="\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}(\.\d*)Z?" />
  </xs:restriction>
</xs:simpleType>
</xs:schema>
```

D.4 Beispiel für die Tabellendaten einer Tabelle: `table0.xml`

Die Tabellendaten werden in einer XML-Datei gespeichert, die der XML-Schemadefinition der Tabelle genügt.

```
<?xml version="1.0" encoding="utf-8"?>
<table
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.admin.ch/xmlns/siard/2.0/schema0/table0.xsd"
  xsi:schemaLocation="http://www.admin.ch/xmlns/siard/2.0/schema0/table0.xsd
table0.xsd">
  <row>

    <c1>true</c1> <!-- CBOOLEAN -->
    <c2>01</c2> <!-- CBIT -->
    <c3>31323334</c3> <!-- CBIT_32 -->
    <c4>1E1F2021222324252627282930313233</c4> <!-- CBIT_VARYING_160 -->
    <c5>1E</c5> <!-- CBINARY -->
    <c6>1E1F202122</c6> <!-- CBINARY_5 -->
    <c7>1E1F2021222324252627</c7> <!-- CBINARY_VARYING_32 -->
    <c8 file="record0.bin" length="16000" digestType="MD5"
messageDigest="D1C411FC45542DCA86EEB1CC9B4B596C" /> <!-- CBINARY_LARGE_OBJECT -->
    <c9>A</c9> <!-- CCHARACTER -->
    <c10>ABCDE</c10> <!-- CCHARACTER_5 -->
    <c11>A varchar(32) text</c11> <!-- CCHARACTER_VARYING_32 -->
    <c12 file="record0.txt" length="84000" digestType="MD5"
messageDigest="D1341187455adfCA86E5666C974B5950" /> <!-- CCHARACTER_LARGE_OBJECT -->
  >
    <c13>Ä</c13> <!-- CNATIONAL_CHARACTER -->
    <c14>ABCDÖ</c14> <!-- CNATIONAL_CHARACTER_5 -->
    <c15>A national varchar(32) text</c15> <!-- CNATIONAL_CHARACTER_VARYING -->
    <c16 file="record0.txt" length="84000" digestType="MD5"
messageDigest="243561FE455EBFCAA1256667774B5831" /> <!--
CNATIONAL_CHARACTER_LARGE_OBJECT -->
    <c17 file="record0.xml" length="10424" digestType="MD5"
messageDigest="5234Fd874EFADFC86E5CDDC97398991" /> <!-- CXML -->
    <c18>4321</c18> <!-- SMALLINT -->
    <c19>5</c19> <!-- CINTEGER -->
    <c20>98765432101234</c20> <!-- CBIGINT -->
```

```

<c21>12345</c21> <!-- CDECIMAL -->
<c22>123</c22> <!-- CDECIMAL_3 -->
<c23>123.45</c23> <!-- CDECIMAL_5_2 -->
<c24>54321</c24> <!-- NUMERIC -->
<c25>123</c25> <!-- NUMERIC_3 -->
<c26>123.45</c26> <!-- NUMERIC_5_2 -->
<c27>3.14159</c27> <!-- CREAL -->
<c28>3.14159</c28> <!-- CFLOAT -->
<c29>3.141</c29> <!-- CFLOAT_13 -->
<c30>987654321.0900</c30> <!-- CDOUBLE_PRECISION -->
<c31>2013-10-18</c31> <!-- CDATE -->
<c32>17:24:33</c32> <!-- CTIME -->
<c33>17:24:33.12345</c33> <!-- CTIME_5 -->
<c34>2011-12-05T16:24:33.123456</c34> <!-- CTIMESTAMP -->
<c35>2011-12-05T16:24:33.1234567</c35> <!-- CTIMESTAMP_7 -->
</row>
</table>

```

D.5 Beispiel für die XML-Schemadefinition einer Tabelle mit fortgeschrittenen und strukturierten Datentypen: `table1.xsd`

```

<?xml version="1.0" encoding="utf-8" standalone="no"?>
<xs:schema
  xmlns:xs="http://www.w3.org/2001/XMLSchema"
  xmlns="http://www.admin.ch/xmlns/siard/2.0/schema0/table1.xsd"
  attributeFormDefault="unqualified"
  elementFormDefault="qualified"
  targetNamespace="http://www.admin.ch/xmlns/siard/2.0/schema0/table1.xsd">
  <xs:element name="table">
    <xs:complexType>
      <xs:sequence>
        <xs:element maxOccurs="unbounded" minOccurs="0" name="row"
type="rowType"/></xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
  <xs:complexType name="rowType">
    <xs:sequence>
      <xs:element name="c1" type="xs:integer" /> <!-- ID -->
      <xs:element minOccurs="0" name="c2" type="xs:integer" /> <!-- CDISTINCT -->
      <xs:element minOccurs="0" name="c3"> <!-- CROW -->
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" name="r1" type="xs:integer" /> <!-- TABLEID --
>
            <xs:element minOccurs="0" name="r2" type="clobType" /> <!--
TRANSCRIPTION -->
            <xs:element minOccurs="0" name="r3" type="blobType" /> <!-- SOUND -->
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element minOccurs="0" name="c4"> <!-- CARRAY -->
        <xs:complexType>
          <xs:sequence>
            <xs:element minOccurs="0" name="a1" type="xs:string" />
            <xs:element minOccurs="0" name="a2" type="xs:string" />

```

```

        <xs:element minOccurs="0" name="a3" type="xs:string" />
        <xs:element minOccurs="0" name="a4" type="xs:string" />
    </xs:sequence>
</xs:complexType>
</xs:element>
<xs:element minOccurs="0" name="c5" > <!-- CUDT -->
    <xs:complexType>
        <xs:sequence>
            <xs:element minOccurs="0" name="u1" type="xs:integer" /> <!-- ID -->
            <xs:element minOccurs="0" name="u2"> <!-- NESTEDROW -->
                <xs:complexType>
                    <xs:sequence>
                        <xs:element minOccurs="0" name="r1" type="xs:integer" /> <!-- TABLEID --
>
                    <xs:element minOccurs="0" name="r2" type="clobType" /> <!--
TRANSCRIPTION -->
                    <xs:element minOccurs="0" name="r3" type="blobType" /> <!-- SOUND -->
                </xs:sequence>
            </xs:complexType>
        </xs:element>
    </xs:sequence>
</xs:complexType>
<xs:complexType name="clobType">
    <xs:simpleContent>
        <xs:extension base="xs:string">
            <xs:attribute name="file" type="xs:anyURI" />
            <xs:attribute name="length" type="xs:integer" />
            <xs:attribute name="digestType" type="digestTypeType" />
            <xs:attribute name="messageDigest" type="xs:string"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>
<xs:complexType name="blobType">
    <xs:simpleContent>
        <xs:extension base="xs:hexBinary">
            <xs:attribute name="file" type="xs:anyURI" />
            <xs:attribute name="length" type="xs:integer" />
            <xs:attribute name="digestType" type="digestTypeType" />
            <xs:attribute name="messageDigest" type="xs:string"/>
        </xs:extension>
    </xs:simpleContent>
</xs:complexType>
<!-- type for message digest type -->
<xs:simpleType name="digestTypeType">
    <xs:restriction base="xs:string">
        <xs:whiteSpace value="collapse"/>
        <xs:enumeration value="MD5"/>
        <xs:enumeration value="SHA-1"/>
        <xs:enumeration value="SHA-256"/>
    </xs:restriction>
</xs:simpleType>
<!-- date type between 0001 and 9999 restricted to UTC -->
<xs:simpleType name="dateType">
    <xs:restriction base="xs:date">
        <xs:minInclusive value="0001-01-01"/>

```

```

        <xs:maxExclusive value="10000-01-01Z"/>
        <xs:pattern value="\d{4}-\d{2}-\d{2}Z?"/>
    </xs:restriction>
</xs:simpleType>
<!-- time type restricted to UTC -->
<xs:simpleType name="timeType">
    <xs:restriction base="xs:time">
        <xs:pattern value="\d{2}:\d{2}:\d{2}(\.\d+)?Z?"/>
    </xs:restriction>
</xs:simpleType>
<!-- dateTime type between 0001 and 9999 restricted to UTC -->
<xs:simpleType name="dateTimeType">
    <xs:restriction base="xs:dateTime">
        <xs:minInclusive value="0001-01-01T00:00:00.000000000Z"/>
        <xs:maxExclusive value="10000-01-01T00:00:00.000000000Z"/>
        <xs:pattern value="\d{4}-\d{2}-\d{2}T\d{2}:\d{2}:\d{2}(\.\d*)Z?"/>
    </xs:restriction>
</xs:simpleType>
</xs:schema>

```

D.6 Beispiel für die Tabellendaten einer Tabelle mit fortgeschrittenen und strukturierten Datentypen: `table1.xml`

```

<?xml version="1.0" encoding="utf-8"?>
<table
  xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
  xmlns="http://www.admin.ch/xmlns/siard/2.0/schema0/table1.xsd"
  xsi:schemaLocation="http://www.admin.ch/xmlns/siard/2.0/schema0/table1.xsd
table1.xsd">
  <row>
    <c1>5</c1> <!-- ID -->
    <c2>5555</c2> <!-- CDISTINCT -->
    <c3> <!-- CROW -->
      <r1>1234</r1> <!-- TAPEID -->
      <r2 file="record0.txt" length="84000" digestType="MD5"
messageDigest="D1341187455adfCA86E5666C974B5950"/> <!-- TRANSCRIPTION -->
      <r3 file="record0.bin" length="16000" digestType="MD5"
messageDigest="D1C411FC45542DCA86EEB1CC9B4B596C"/> <!-- SOUND -->
    </c3>
    <c4> <!-- CARRAY -->
      <a1>Vorname1</a1>
      <!-- a2 is NULL! and therefore missing -->
      <a3>Patronymic</a3>
      <a4>Name</a4>
    </c4>
    <c5> <!-- CUDT -->
      <u1>9876</u1> <!-- ID -->
      <u2> <!-- NESTEDROW -->
        <r1>71934576</r1> <!-- TAPEID -->
        <!-- r2 is NULL and therefore missing -->
        <r3 file="sub1000/record0.bin" length="16000" digestType="MD5"
messageDigest="D1C411FC45542DCA86EEB1CC9B4B596C"/> <!-- SOUND -->
      </u2>
    </c5>
  </row>
</table>

```

Anhang E – Änderungen gegenüber Version 1.0

Folgende Änderungen wurden von der Version 1.0 zur Version 2.0 vorgenommen.

Kapitel / ID / Dokument	Anpassung	RFC
passim	Anpassung von SQL:1999 zu SQL:2008	
passim	Kann-Anforderungen: Es wird klar festgehalten, ob ein Feld zwingend ist oder nicht oder ob es auch leer gelassen werden kann.	2013-23
3.4, 5.1, 5.2, 5.3, 5.4, 5.6, 6.2,	Externer Speicherort für BLOBs und LOBs. Neu ist es möglich, BLOBS und LOBS extern, das heisst ausserhalb der SIARD-Datei zu speichern. Die Referenz auf die externe Datei erfolgt mit einer URL oder einer Dateireferenz in table.xml. Diese Lösung erlaubt eine separate Bewirtschaftung der BLOB-Dateien (z.B. Office-Dokumente oder Bilder) im Archivfindmittel, d.h. diese Dokumente können einzeln im Findmittel erschlossen und angesprochen werden. Die Formatmigration ist zudem einfacher.	2015-29
4.1	Deflate-Komprimierung zulassen	Addendum
4.2	Formaterkennung. Neben der Formaterkennung (PK\003\004) und nach XMLNS String (xmlns=http://www.bar.admin.ch/xmlns/siard/1.0/metadata.xsd bzw. xmlns=http://www.bar.admin.ch/xmlns/siard/2.0/metadata.xsd) kann eine leere Versionsdatei mit dem Namen «siard.version.2.0» im Ordner "header" gespeichert werden.	2015-12
4.3, 5.6, 5.7, 5.16, 5.17, 6.4,	SQL:1999- bzw. SQL:2008-Kompatibilität wird vollständig realisiert: User-Defined Types, Datentyp ARRAY.	2014-110
4.3, 6.1	Data Type Mapping wird als externer Anhang bzw. als Addendum Teil der Spezifikation.	2015-13
6	Tippfehler in der ID 6.0-1	2014-1
metadata.xsd	Nullable in columnType ist obligatorisch.	2015-11
metadata.xsd	Validierungsregeln für Datentypen sind mittels <i>Regular Expressions</i> expliziter formuliert	