SIARD-2.1-Formatspezifikation

Name	SIARD-2.1-Formatspezifikation
Kategorie	Standard
Reifegrad	Implementiert
Version	2.1
Status	Stabile Version
Beschluss am	2018-12-04
Ausgabedatum	2018-12-04
Ersetzt Version	eCH-0165 Version 2.0
Voraussetzungen	Keine
Beilagen	metadata.xsd, ech-0165_oe.siard[1]
Sprachen	Deutsch (Original), Französisch (Übersetzung), Englisch (Übersetzung)
Autoren	Marcel Büchler, Schweizerisches Bundesarchiv, marcel.buechler@bar.admin.ch Luis Faria, KEEP SOLUTIONS, LDA, lfaria@keep.pt Bruno Ferreira, KEEP SOLUTIONS, LDA, bferreira@keep.pt Anders Bo Nielsen, Danish National Archives (Rigsarkivet), abn@sa.dk Krystyna Ohnesorge, Schweizerisches Bundesarchiv, krystyna.ohnesorge@bar.admin.ch Claire Röthlisberger-Jourdan, KOST, claire.roethlisberger@kost.admin.ch Hartwig Thomas, Enter AG, hartwig.thomas@enterag.ch Andreas Voss †, Schweizerisches Bundesarchiv

[1] Basiert auf der OE-Beispieldatenbank von Oracle.

Mitarbeitende	Karin Bredenberg, National Archives of Sweden, karin.bredenberg@riksarkivet.se Hedi Bruggisser, Staatsarchiv Thurgau, hedi.bruggisser@tg.ch Georg Büchler, KOST, georg.buechler@kost.admin.ch Janet Delve, University of Portsmouth, janet.delve@port.ac.uk Boris Domajnko, Slovenian National Archives, boris.domajnko@gov.si Alain Dubois, Staatsarchiv Wallis, alain.dubois@admin.vs.ch Arne-Kristian Groven, National Archives Norway (Riksarkivet), arngro@arkivverket.no Martin Kaiser, KOST, martin.kaiser@kost.admin.ch Lambert Kansy, Staatsarchiv Basel Stadt, lambert.kansy@bs.ch Markus Lischer, Staatsarchiv Luzern, markus.lischer@lu.ch Zoltán Lux, National Archives of Hungary, lux.zoltan@mnl.gov.hu Rebekka Plüss, Staatsarchiv Zürich, rebekka.pluess@ji.zh.ch Lauri Rätsep, National Archives of Estonia, lauri.ratsep@ra.ee Hélder Silva, KEEP SOLUTIONS, LDA, hsilva@keep.pt Mario Spuler, Fachlabor Gubler, m.spuler@fachlabor-gubler.ch
Herausgeber / Vertrieb	Koordinationsstelle für die dauerhafte Archivierung elektronischer Unterlagen (KOST), c/o Schweizerisches Bundesarchiv, Archivstrasse 24, 3003 Bern https://kost-ceco.ch/ / info@kost-ceco.ch

Zusammenfassung

Dieses Dokument enthält die Spezifikation des SIARD-Dateiformats, Version 2.1. SIARD steht für *Software-Independent Archival of Relational Databases*. Die Version 1.0 wurde vom Schweizerischen Bundesarchiv entwickelt und wird seither durch die Community weiterentwickelt. 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 unabhängig von proprietären Dump-Formaten zu ermöglichen.

Verhältnis der vorliegenden Version zu Vorversionen:

eCH-0165 v1.0 → Abgelöst durch Version 2.1:
Die Version 1.0 ist die aktuelle eCH-Version des Standards. Die Benutzung ist zwar noch möglich, es wird aber empfohlen, SIARD-2.1 einzusetzen.

eCH-0165 v2.0 → Aufgehoben:
Die Version 2.0 wurde ersetzt, da es bei der Implementierung zu Fehlern und Unklarheiten gekommen ist, welche eine Implementierung verunmöglichten. Die Version 2.0 darf nicht mehr verwendet werden. Es muss die neue SIARD-2.1 verwendet werden, welche bereits implementiert ist[2], oder alternativ eCH-0165 Version 1.0.

SIARD-2.1 Die vorliegende Version 2.1 stellt den aktuellen Entwicklungsstand des SIARD-Formats dar. Sie wurde von der eCH-Fachgruppe Digitale Archivierung erarbeitet, aber ist kein offizieller eCH-Standard.

[2] SIARD-2.1 wird von Siard Suite ab dem Release 2.0 und von KOST-Val ab der Version 1.8 unterstützt.

Inhaltsverzeichnis

1 Einleitung ... 6
1.1 Status ... 6
1.2 Anwendungsgebiet ... 6
1.2.1 Adressaten/Zielgruppe ... 6
1.2.2 Ausgangslage ... 6
1.2.3 Abgrenzungen ... 7
2 Struktur des Dokuments ... 8
2.1 Aufbau Kapitel ... 8
2.2 ID Anforderungen ... 8
2.3 Unterscheidung zwischen Muss- und Kann-Anforderungen ... 9
2.4 Notation Ordner, Dateien und Ordnerstrukturen ... 9
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 ... 12
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 ... 24
5.4 Metadaten auf der Ebene Attribut ... 25
5.5 Metadaten auf der Ebene Tabelle ... 26
5.6 Metadaten auf der Ebene Spalte ... 27
5.7 Metadaten für Felder ... 29
5.8 Metadaten des Primärschlüssels ... 30
5.9 Metadaten der Fremdschlüssel ... 30

5.10 Referenz-Metadaten 31
5.11 Metadaten des Kandidatenschlüssels 31
5.12 Metadaten der Check-Einschränkung 32
5.13 Metadaten auf der Ebene Trigger 32
5.14 Metadaten auf der Ebene View 33
5.15 Metadaten auf der Ebene Routine 34
5.16 Metadaten der Parameter 34
5.17 Metadaten auf der Ebene des Benutzers 35
5.18 Metadaten auf der Ebene Rolle 36
5.19 Metadaten auf der Ebene der Privilegien 36
6 Anforderungen an die Tabellendaten 37
6.1 Tabellen-Schemadefinition 37
6.2 *Large-Object*-Datenzellen 38
6.3 Datums- und Timestamp-Datenzellen 39
6.4 Tabellendaten 39
7 Version und Gültigkeit der Spezifikation 41
8 Change-Management-Prozess 41
9 Haftungsausschluss/Hinweise auf Rechte Dritter 41
10 Urheberrechte 41
Anhang A – Mitarbeit & Überprüfung 42
Anhang B – Abkürzungen und Glossar 43
Anhang C – Nachweis der verwendeten Standards 45
Anhang D – Auszüge aus Beispiel ech-0165_oe.siard 46
Anhang E – Änderungen gegenüber Version 1.0 74

1 Einleitung

1.1 Status

Das Dokument wurde von der eCH-Fachgruppe Digitale Archivierung genehmigt und kann implementiert werden.

1.2 Anwendungsgebiet

1.2.1 Adressaten/Zielgruppe

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

1.2.2 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 werden[3].

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 werden, die international breit abgestützt sind. Diese langfristige Interpretierbarkeit der Datenbankinhalte beruht im Wesentlichen auf den beiden Standards XML und SQL:2008.

[3] 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.

1.2.3 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 ist[4].

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.

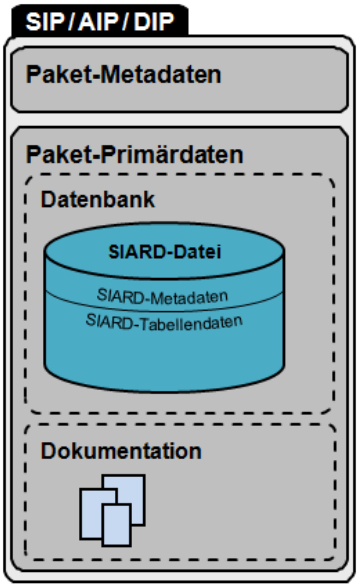


Abb. 1: Symbolbild eines Informationspakets mit einer enthaltenen SIARD-Datei

[4] http://public.ccsds.org/pubs/650x0m2.pdf

2 Struktur des Dokuments

2.1 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_2.1-1	Anforderungstext **Beispiel** Beispieltext **Hinweis** Hinweistext ***Empfehlung*** *Empfehlungstext ist kursiv gesetzt.*	M

2.2 ID Anforderungen

Die Anforderungen sind über eine ID eindeutig identifizierbar.

ID
G_2.2-1

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

2.2-1 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.

2.3 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 *Best Practice*.

2.4 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.
`[]`	Platzhalter für einen Ausdruck oder einen Basistyp wie «string», «integer» etc.
`<xx>`	Platzhalter für beliebige Zeichenkette

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 Dateien im Format XML 1.0[5] gespeichert, die konform zu Schema Definitionen gemäss XML Schema 1.0[6] sind. 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 Dateien 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. In dieser können allenfalls extern gespeicherte Large Objects referenziert sein, die im weiteren Sinn zur Datenbank gehören.	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, NCHAR VARYING, 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

[5] https://www.w3.org/TR/REC-xml/

[6] https://www.w3.org/TR/xmlschema-1/, https://www.w3.org/TR/xmlschema-2/, https://www.w3.org/TR/xmlschema-ref/

<table>
<tr><th>ID</th><th>Beschreibung Anforderung</th><th>M/K</th></tr>
<tr><td>G_3.3-3</td><td>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.</td><td>M</td></tr>
<tr><td>G_3.3-4</td><td>Zeichen, die nicht in UNICODE dargestellt werden können (Codes 0-8, 14-31, 127-159), sowie das Escapezeichen '\' und mehrere aufeinanderfolgende Leerschläge werden mit Escape als \u00<xx> in XML dargestellt. Anführungszeichen, Kleiner und Et-Zeichen werden in XML als Einheitsreferenzen dargestellt.
<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>"</td></tr>
<tr><td>&</td><td>&</td></tr>
<tr><td>'</td><td>'</td></tr>
<tr><td><</td><td><</td></tr>
<tr><td>></td><td>></td></tr>
<tr><td>\</td><td>\u005c</td></tr>
<tr><td>127 bis 159</td><td>\u007F bis \u009F</td></tr>
</table></td><td>M</td></tr>
</table>

3.4 File-URI-Schema

Zur Referenzierung von ausgelagerten *Large Objects* wird das File-URI-Schema gemäss RFC 1738 verwendet[7].

ID	Beschreibung Anforderung	M/K
G_3.4-1	Alle ausgelagerten Dateien werden mit einer File-URI gemäss RFC 1738 spezifiziert.	M
G_3.4-2	File-URI werden in einer SIARD-Datei als URL-codierte ASCII-Strings gespeichert.	M
G_3.4-3	Ausgelagerte Large Objects können wieder in ZIP-Dateien zusammengefasst werden, sofern das File-URI einem Dateisystem zugrunde gelegt wird, welches die direkte Adressierung individueller Dateien innerhalb einer ZIP-Datei ermöglicht. Beispielsweise würde *file:///d:/sips/sip1234.zip* auf die ZIP-Datei verweisen, während *file:///d:/sips/sip1234.zip/* auf den Stammordner innerhalb der ZIP-Datei verweist.	K

[7] http://en.wikipedia.org/wiki/File_URI_scheme, http://tools.ietf.org/html/rfc1738.

3.5 Bezeichner und reguläre Bezeichner

In SQL:2008 gibt es reguläre Bezeichner[8] ohne Leerschläge und Sonderzeichen, für welche Gross- und Kleinschreibung unwichtig ist, die aber in der SIARD-Datei in Grossbuchstaben gespeichert werden, und Bezeichner in Anführungszeichen[9], 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 oder ein Schlüsselwort 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 alle anderen Bezeichner unverändert ohne Anführungszeichen gespeichert werden. 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

[8] „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.

[9] „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[10].	M
G_4.1-2	SIARD-Dateien müssen entweder unkomprimiert oder mit dem Deflate-Algorithmus gemäss RFC 1951[11] komprimiert sein. ***Empfehlung*** *Es wird empfohlen, den Deflate-Algorithmus zu verwenden.*	M
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	Die Tabellendaten befinden sich im Ordner `content/` und die Metadaten im Ordner `header/`. Weitere Ordner oder Dateien sind nicht erlaubt.
	
	Beispiel
	Aufbau der SIARD-Datei (schematisch)
	`ech-0165_oe.siard`
	`  content/`
	`  header/`
		M

[10] 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 https://support.pkware.com/display/PKZIP/Application+Note+Archives.

[11] https://www.ietf.org/rfc/rfc1951.txt.

ID	Beschreibung Anforderung	M/K

P_4.2-2	Im Ordner `content/` befinden sich ein oder mehrere Schema-Ordner, welche die einzelnen Tabellen-Ordner enthalten. Andere Ordner oder Dateien sind nicht erlaubt.
	
	Beispiel
	Aufbau der SIARD-Datei (schematisch)
	`ech-0165_oe.siard`
	`content/`
	`schema0/`
	`table0/`
	`table1/`
	`table2/`
	`...`
	`schema1/`
	`table0/`
	`...`
	
	Empfehlung
	Es wird empfohlen, die Namen der Schema- und Tabellenordner zu normalisieren und anstelle des eigentlichen Namen z.B. `schema0/` *und* `table0/` *zu verwenden (siehe Einschränkungen unter P_4.2-6).*
		M

P_4.2-3	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-Ordnern samt deren Inhalt (BIN-, TXT-, XML-Dateien oder, falls der MIME Type der LOB-Dateien bekannt ist, eine mit diesem assoziierte Dateierweiterung wie zum Beispiel JPG) nicht erlaubt.
	
	Beispiel
	Aufbau der SIARD-Datei (schematisch)
	`ech-0165_oe.siard`
	`content/`
	`...`
	`schema1/`
	`...`
	`table6/`
	`table6.xml`
	`table6.xsd`
	`table7/`
	`table7.xml`
	`table7.xsd`
	`lob1`[12]`/`
	`record0.xml`
	`record1.xml`
	`...`
		M

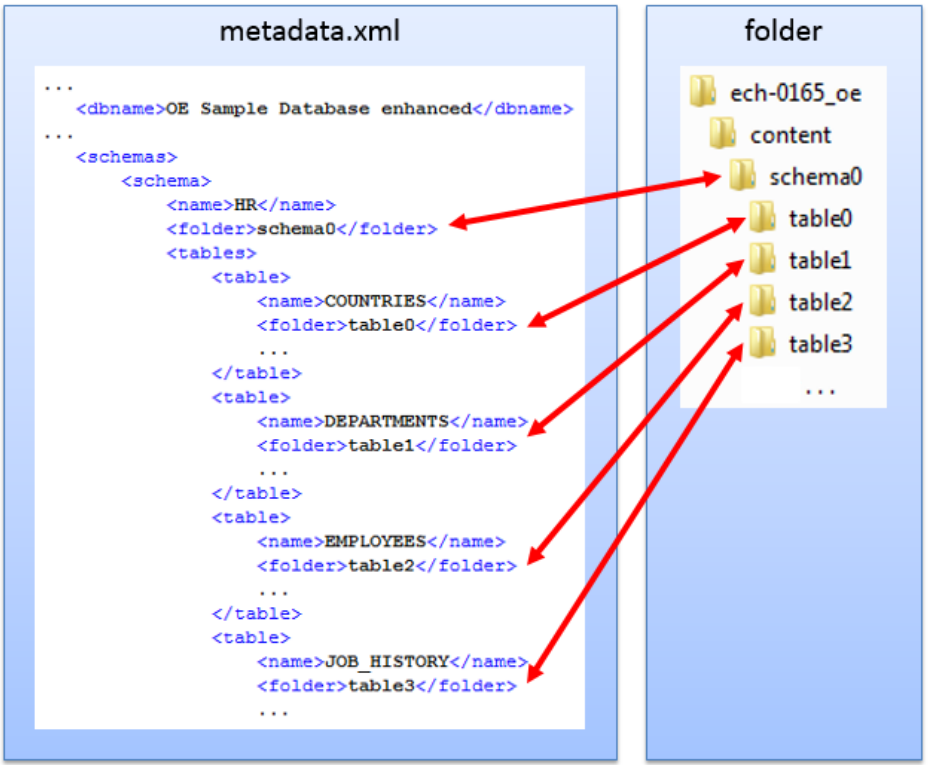
[12] Bei diesem Beispiel enthält die Spalte 2 zusätzliche lob-Dateien, die entsprechend in `lob1/` abgelegt werden.

ID	Beschreibung Anforderung	M/K
	Empfehlung *Es wird empfohlen, die Namen der LOB-Ordner und LOB-Dateien zu normalisieren und anstelle des eigentlichen Namens z.B.* `lob1/` *und* `record0.bin`, `record0.txt` *oder* `record0.xml` *oder, falls der MIME Type der LOB-Dateien bekannt ist, eine mit diesem assoziierte Dateierweiterung zu verwenden (siehe Einschränkungen unter P_4.2-6).*	
P_4.2-4	Zur einfacheren Erkennung des SIARD-Formats (z.B. durch PRONOM) muss ein leerer Ordner `header/siardversion/2.1/` existieren, welcher die Version des SIARD-Formats identifiziert.	M

P_4.2-5	Im Ordner `header/` müssen die Dateien `metadata.xml` und `metadata.xsd` vorhanden sein. Weitere Dateien, zum Beispiel Stylesheets, sind erlaubt.
	
	Beispiel
	Aufbau der SIARD-Datei (schematisch)
	`ech-0165_oe.siard`
	`  content/`
	`...`
	`  header/`
	`  metadata.xml`
	`  metadata.xsd`
	`  siardversion/`
	`  2.1/`
	`...`
		M

P_4.2-6	Alle Datei- und Ordnernamen müssen wie folgt aufgebaut sein: Der Name muss mit einem Buchstaben [a-z respektive A-Z] beginnen und darf anschliessend nur folgende Zeichen enthalten: • a-z • A-Z • 0-9 • _ • . (darf nur für die Trennung zwischen Namen und Extension verwendet werden) ***Empfehlung*** *Die Länge der Datei- und Ordnernamen sollte möglichst 20 Zeichen nicht überschreiten, damit Schwierigkeiten mit zu grossen Pfadlängen unter Windows vermieden werden können.*	M

4.3 Korrespondenz zwischen Metadaten und Tabellendaten

ID	Beschreibung Anforderung	M/K
P_4.3-1	Die in `metadata.xml` vorgegebene Struktur muss identisch sein mit jener im Ordner `content/`. **Beispiel**  	M
P_4.3-2	Die in `metadata.xml` angegebene Anzahl der Spalten einer Tabelle muss identisch sein mit jener der entsprechenden Datei `table[Zahl].xsd`. **Beispiel** 	M

<table>
<tr><th>ID</th><th>Beschreibung Anforderung</th><th>M/K</th></tr>
<tr><td>P_4.3-3</td><td>Die Datentyp-Angaben zu den Spaltendefinitionen in <code>metadata.xml</code> müssen identisch sein mit jenen der entsprechenden Datei <code>table[Zahl].xsd</code>.

Die vordefinierten SQL:2008-Datentypen[13] werden in den Schemadateien <code>table[Zahl].xsd</code> gemäss der folgenden Tabelle in XML-Datentypen umgewandelt.
<table>
<tr><th>SQL:2008</th><th>XML</th></tr>
<tr><td>BIGINT</td><td>xs:integer</td></tr>
<tr><td>BINARY LARGE OBJECT(...), BLOB(...)</td><td>blobType[14]</td></tr>
<tr><td>BINARY VARYING(...), VARBINARY(...)</td><td>xs:hexBinary / blobType[14]</td></tr>
<tr><td>BINARY(…)</td><td>xs:hexBinary / blobType[14]</td></tr>
<tr><td>BOOLEAN</td><td>xs:boolean</td></tr>
<tr><td>CHARACTER LARGE OBJECT(...), CLOB(...)</td><td>clobType[14]</td></tr>
<tr><td>CHARACTER VARYING(...),
CHAR VARYING(...), VARCHAR(...)</td><td>xs:string / clobType[14]</td></tr>
<tr><td>CHARACTER(...), CHAR(...)</td><td>xs:string / clobType[14]</td></tr>
<tr><td>DATE</td><td>dateType</td></tr>
<tr><td>DECIMAL(...), DEC(...)</td><td>xs:decimal</td></tr>
<tr><td>DOUBLE PRECISION</td><td>xs:double</td></tr>
<tr><td>FLOAT(p)</td><td>xs:double</td></tr>
<tr><td>INTEGER, INT</td><td>xs:integer</td></tr>
<tr><td>INTERVAL <start> [TO <end>]</td><td>xs:duration</td></tr>
<tr><td>NATIONAL CHARACTER LARGE OBJECT(...),
NCHAR LARGE OBJECT(...), NCLOB(...)</td><td>clobType[14]</td></tr>
<tr><td>NATIONAL CHARACTER VARYING(...),
NATIONAL CHAR VARYING(...),
NCHAR VARYING(...)</td><td>xs:string / clobType[14]</td></tr>
<tr><td>NATIONAL CHARACTER(...), NCHAR(...),
NATIONAL CHAR(...),</td><td>xs:string / clobType[14]</td></tr>
<tr><td>NUMERIC(...)</td><td>xs:decimal</td></tr>
<tr><td>REAL</td><td>xs:float</td></tr>
<tr><td>SMALLINT</td><td>xs:integer</td></tr>
<tr><td>TIME(...)</td><td>timeType</td></tr>
<tr><td>TIME WITH TIME ZONE(...)</td><td>timeType</td></tr>
<tr><td>TIMESTAMP(...)</td><td>dateTimeType</td></tr>
<tr><td>TIMESTAMP WITH TIME ZONE(...)</td><td>dateTimeType</td></tr>
<tr><td>XML</td><td>clobType[14]</td></tr>
</table></td><td>M</td></tr>
</table>

[13] BIT und BIT VARYING sind beides Datentypen aus alten SQL-Definitionen und wurden in SQL:2008 durch BOOLEAN und BINARY ersetzt. BIT(1) wird nach BOOLEAN und BIT(n) nach BINARY((n+7)/8) konvertiert.

[14] Zu den XML-Datentypen *blobType* und *clobType* siehe G_3.1-1.

ID	Beschreibung Anforderung	M/K
	Beispiel metadata.xml table0.xsd Alle DATE- und TIME-Typen sind in der UTC-Zeitzone spezifiziert. Es wird empfohlen, sie mit einem „Z“ zu beenden. dateType 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 sind keine zusätzlichen Attribute nötig, und der CLOB-Wert wird direkt gespeichert. Werden die CLOB-Werte in einer separaten Datei gespeichert, müssen folgende Attribute zwingend gesetzt werden: *file* und *length*. Nebst diesen zwingenden Attributen existieren noch die folgenden optionalen Attribute *digestType* und *digest*. Der Wert des *file*-Attributs enthält die Datei-URI (URL-enkodiert, 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 Attribut *digest* enthält Integritätsinformation gemäss dem optionalen Attribut *digestType*. blobType ist eine Extension von xs:hexBinary. Für Inline-Werte sind keine zusätzlichen Attribute nötig, und der BLOB-Wert wird direkt gespeichert. Werden die BLOB-Werte in einer separaten Datei gespeichert, müssen folgende Attribute zwingend gesetzt werden: *file* und *length*. Nebst diesen zwingenden Attributen existieren noch die folgenden optionalen Attribute *digestType* und *digest*. Der Wert des *file*-Attributs enthält die Datei-URI (URL-enkodiert, wenn möglich auf den nächstliegenden lobFolder bezogen), wo das BLOB gespeichert ist. Der Wert des *length*-Attributs ist die Länge in UTF-8, und das optionale Attribut *digest* enthält Integritätsinformation gemäss dem optionalen Attribut *digestType*.	
P_4.3-4	Die benannten DISTINCT-Datentypen werden in derjenigen `table[Zahl].xsd`-Schemadateien in den XML-Datentyp konvertiert, welche zur Darstellung ihrer Basistypen verwendet würden.	M
P_4.3-5	Arrays werden in den `table[Zahl].xsd`-Schemadateien in eine Sequenz strukturierter XML-Elemente `<a1>, <a2>, …` konvertiert, welche ihrerseits in den XML-Datentypen konvertiert werden, der dem Basistyp des Array entspricht. **Beispiel** Siehe das Beispiel `table0.xsd` im Anhang D.3c.	M

ID	Beschreibung Anforderung	M/K
P_4.3-6	Der benannte User-defined Data Type (UDT) wird in den `table[Zahl].xsd`-Schemadateien in eine Sequenz strukturierter XML-Elemente `<u1>, <u2>, …` konvertiert, welche ihrerseits in den XML-Datentypen konvertiert werden, der dem Typen jedes Attributs entspricht. **Beispiel** Siehe das Beispiel `table0.xsd` im Anhang D.3c.	M
P_4.3-7	Die Nullable-Angaben zu den Spaltendefinitionen in `metadata.xml` müssen identisch sein mit jenen der entsprechenden Datei `table[Zahl].xsd`. **Beispiel** metadata.xml table0.xsd **Hinweis** Die SQL:2008-Notation "<nullable>true</nullable>" wird in XML zu "minOccurs="0". "<nullable>false</nullable>" entspricht "minOccurs="1" in XML. Da "minOccurs="1"jedoch der Standardwert ist, wird er oftmals weggelassen. Steht keine Angabe zu "<nullable>" bedeutet dies "<nullable>true</nullable>".	M
P_4.3-8	Die Spaltenreihenfolge in `metadata.xml` muss identisch sein mit der Spaltenreihenfolge in der entsprechenden `table[Zahl].xsd`.	M
P_4.3-9	Die Feldreihenfolge in der Tabellendefinition von `metadata.xml` muss identisch sein mit der Feldreihenfolge der entsprechenden `table[Zahl].xsd`.	M

ID	Beschreibung Anforderung	M/K
P_4.3-10	Die Anzahl Zeilen einer Tabelle in `metadata.xml` muss identisch sein mit der Anzahl Zeilen in der entsprechenden `table[Zahl].xml`. Die Anzahl Zeilen einer Tabelle in `metadata.xml` muss in den von der entsprechenden `table[Zahl].xsd` spezifizierten Bereich hineinpassen. **Beispiel** metadata.xml table0.xsd table0.xml ***Empfehlung*** *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.*	M

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. Diese Datei ist 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 `metadata.xsd` ist für die Datei `metadata.xml` verbindlich einzuhalten. Das heisst, `metadata.xml` muss die Validierung mit dem Schema `metadata.xsd` bestehen.	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 `metadata.xsd` 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	Eine „file:“-URI, die als Basis-URI dient für relative URI, welche den möglichen externen Speicherort von *Large Objects* 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. **Hinweis** Wenn die „file:“-URI auf ein *Extended File System* 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 *lobFolder*-Einträge in Spalten und alle LOB-Dateiattribute als relative URI abzubilden. So muss bei einer Verschiebung der SIARD-Datei oder ihres Informationspakets nur diese globale URI geändert werden, um auf den neuen Speicherort zu verweisen.	K
producerApplication	Name und Version der Anwendung, welche die SIARD-Datei heruntergeladen hat.	K
archivalDate	Archivierungsdatum; Datum der Archivierung der Tabellendaten	M
messageDigest	Besteht aus `digestType` (MD5, SHA-1 oder SHA-256) und dem zugehörigen `digest`. Der `digest` repräsentiert einen binären Buffer als hexadezimale oder alternativ – für SHA-1 oder SHA-256 – eine base64-Zeichenkette. Ob hexadezimale oder base64-Codierung benutzt wurde, entscheidet man anhand der Länge des binären Digest und der Zeichenkette. Der Digest wird über den Ordner `content/` berechnet. Es können mehrere Message-Digest-Codes gespeichert werden, basierend auf verschiedenen Algorithmen[15]. **Beispiel** Siehe das Beispiel `metadata.xml` im Anhang D.2. ***Empfehlung*** *Wird die Option MessageDigest verwendet, muss folgendes umgesetzt werden:* *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.*	K

[15] Ein in der SIARD-Datei gespeicherter Message-Digest-Code garantiert für sich keine Integrität. Denn er kann von einem Fälscher der SIARD-Datei mitgefälscht werden. Dagegen hilft nur die externe Speicherung eines Message-Digest-Codes. Die interne Erzeugung eines metadatenunabhängigen Message-Digest-Codes beim Herunterladen kann dies aber unterstützen.

Bezeichnung	Bedeutung	M/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
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 `metadata.xsd` 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 `content/` 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 im Schema	K
views	Liste der im Schema gespeicherten Views	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 `metadata.xml` 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
category	Kategorie des fortgeschrittenen oder strukturierten Datentyps (“distinct” oder “udt”).	M
underSchema	Schemaname des Supertyps, falls der Datentyp auf einem Supertyp basiert.	K
underType	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 (vordefinierten SQL-) Basistyps, falls die Kategorie “distinct” ist.	K
attributes	Liste der Attribute, falls die Kategorie “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	Alle im "udt" Datentyp verwendeten Metadaten, die in `metadata.xsd` auf der Ebene Attribut als Muss bezeichnet sind, müssen entsprechend ausgefüllt sein.	M

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

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
nullable	Nullable-Element des Attributs. ***Empfehlung*** *Es wird empfohlen das nullable-Element nicht einzusetzen.*	K
typeSchema	Schema des fortgeschrittenen oder strukturierten Datentyps.	K
typeName	Name des fortgeschrittenen oder strukturierten Datentyps.	K
defaultValue	Standardwert des Attributs.	K
description	Beschreibung der Bedeutung und Funktion der Routine.	K
cardinality	(Maximale) Anzahl Elemente, falls das Attribut ein Array ist.	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 `metadata.xsd` 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 `metadata.xsd` 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
lobFolder	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 *Large Objects* benutzt werden. **Beispiel** Siehe das Beispiel `metadata.xml` im Anhang D.2. **Hinweis** Dieser Eintrag ist nur von Bedeutung, wenn die Spalte eine LOB-Spalte ist (z.B. vom Typ BLOB, CLOB oder XML). Wenn es fehlt, wird als Defaultwert „.“ angenommen, z.B. als Verweis auf den gleichen Ordner wie *lobFolder* 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. Wenn dieser Wert eine relative URI ist, wird angenommen, dass sie relativ ist zum globalen *lobFolder*-Eintrag auf der Ebene Datenbank. Die relativen *file*-Attribute der Zellen dieser Spalte werden als relativ zu diesem Ordner interpretiert.	K
type	Vordefinierter SQL:2008-Typ der Spalte **Hinweis** Wenn der Datentyp dieser Spalte ein vordefinierter Datentyp ist, ist dieses Feld obligatorisch. Andernfalls muss das Feld *typeName* auf einen definierten Typ in der Typenliste verweisen.	M

Bezeichnung	Bedeutung	M/K
typeOriginal	Originaler Spaltentyp **Hinweis** Da die verschiedenen sich SQL-konform nennenden Datenbank-Programme sehr unterschiedliche Datentypen zulassen, wird hier neben dem SQL:2008-Typ auch der *originale* 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.	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 Array oder ein strukturierter Datentyp der Kategorie „udt“ ist.	K
defaultValue	Standardwert der Spalte	K
mimeType	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
cardinality	(Maximale) Anzahl Elemente, falls die Spalte ein Array ist.	K

5.7 Metadaten für Felder

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

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

Bezeichnung	Bedeutung	M/K
name	Feldname in der Spalte oder Feld Innerhalb der gleichen Spalte muss der Feldnname eindeutig sein. ***Empfehlung*** *Für Container (Spalte oder Feld) vom Typ “udt” sollte der Feldname identisch mit dem entsprechenden Attributenamen sein. Für Array-Container sollte der Feldname der Name des Containers gefolgt vom mit 1 beginnenden Array-Index in eckigen Klammern sein. Also z.B. „Punkt[1]“, „Punkt[2]“ usw.*	M
lobFolder	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 *Large Objects* benutzt werden. **Hinweis** Dieser Eintrag ist nur von Bedeutung, wenn das Feld ein LOB-Feld ist (z.B. vom Typ BLOB, CLOB oder XML). Wenn es fehlt, wird als Defaultwert „.“ angenommen, z.B. als Verweis auf den gleichen Ordner wie *lobFolder* auf der Ebene Datenbank. Andernfalls muss sein Wert eine (wenn möglich relative) „file:“-URI sein, welche den Ordner bezeichnet, in dem die Dateien dieses LOB-Feldes gespeichert werden sollen. Wenn dieser Wert eine relative URI ist, wird angenommen, dass sie relativ ist zum globalen *lobFolder*-Eintrag auf der Ebene Datenbank. Die relativen *file*-Attribute der Zellen dieser Spalte werden als relativ zu diesem Ordner interpretiert.	K
fields	Liste der Felder im Feld, falls das Feld ein Array oder ein strukturierter Datentyp der Kategorie „udt“ ist.	K
mimeType	MIME Type dieses Feldes, falls es ein BLOB-Feld ist und alle Einträge dieses Feldes 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 des Feldes.	K

5.8 Metadaten des Primärschlüssels

Als Primärschlüssel wird ein eindeutiger (UNIQUE) Schlüssel bezeichnet, über welchen ein Datensatz identifiziert wird.

ID	Beschreibung Anforderung	M/K
M_5.8-1	Die Metadaten des Primärschlüssels einer Tabelle können in der Datei `metadata.xml` 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 `metadata.xml` 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
referenced-Schema	Schema der referenzierten Tabelle	M
referencedTable	Referenzierte Tabelle **Hinweis** Der referenzierte externe Tabellenname kann vom Typ tabelle oder schema.tabelle sein. Dabei sind delimitierte Bezeichner in Anführungs-zeichen gesetzt.	M
reference	Referenz (Liste von Spalten und referenzierten Spalten)	M
matchType	Matchtyp (FULL, PARTIAL oder SIMPLE)	K

Bezeichnung	Bedeutung	M/K
deleteAction	Löschaktion, z.B.: CASCADE **Hinweis** Die Lösch- und Änderungsaktion enthalten die vom SQL:2008-Standard zugelassenen Aktionen.	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 `metadata.xml` 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

Als Kandidatenschlüssel werden eindeutige (UNIQUE) Schlüssel bezeichnet, weil sie als Kandidaten für den Primärschlüssel in Frage kommen. Im `metadata.xsd` sind sowohl Primärschlüssel als auch Kandidatenschlüssel vom selben Typ uniqueKeyType. Deshalb sind auch die Anforderungen an den Kandidatenschlüssel identisch mit denjenigen an den Primärschlüssel (M_5.8-1).

ID	Beschreibung Anforderung	M/K
M_5.11-1	Die Metadaten des Kandidatenschlüssels einer Tabelle können in der Datei `metadata.xml` 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

Bezeichnung	Bedeutung	M/K
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 `metadata.xml` 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 `metadata.xml` archiviert werden	K

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

Bezeichnung	Bedeutung	M/K
name	Triggername in der Tabelle	M
actionTime	BEFORE, AFTER oder INSTEAD OF	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 `metadata.xml` 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
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
rows	Anzahl Datensätze	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 `metadata.xml` archiviert werden	K

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

Bezeichnung	Bedeutung	M/K
specificName	Spezifischer Name, welcher die Routine eindeutig im Schema identifiziert[16].	M
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	Rückgabetyp 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 `metadata.xml` archiviert werden	K

[16] Mit der Einführung objekt-orientierter Elemente in SQL:1999 ist auch das „Overloading“ möglich geworden, welches erlaubt, dass zwei verschiedene Routinen (Prozeduren oder Funktionen) denselben Namen haben, sofern diese eine unterschiedliche Parameterliste aufweisen. Deshalb muss die Anforderung fallengelassen werden, dass der Name einer Routine im Schema eindeutig ist. Stattdessen wurde der „spezifische Name“ eingeführt, welcher die Routine eindeutig im Schema identifiziert.

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 *typeName* 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
description	Beschreibung der Bedeutung und der Funktion der Routine	K
cardinality	(Maximale) Anzahl Elemente, falls der Parameter ein Array ist.	K

5.17 Metadaten auf der Ebene des Benutzers

ID	Beschreibung Anforderung	M/K
M_5.17-1	Die Metadaten der Benutzer können in der Datei `metadata.xml` 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

5.18 Metadaten auf der Ebene Rolle

ID	Beschreibung Anforderung	M/K
M_5.18-1	Die Metadaten der Rolle können in der Datei `metadata.xml` 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

5.19 Metadaten auf der Ebene der Privilegien

ID	Beschreibung Anforderung	M/K
M_5.19-1	Die Metadaten der Privilegien können in der Datei `metadata.xml` archiviert werden	K

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

Bezeichnung	Bedeutung	M/K
type	eingeräumtes Privileg (z.B. SELECT)	M
object	Objekt, auf welches das Privileg anzuwenden ist	K
grantor	Berechtiger, 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 `table[Zahl].xsd` ist für die Datei `table[Zahl].xml` verbindlich einzuhalten. Das heisst, `table[Zahl].xml` muss die Validierung mit dem Schema `table[Zahl].xsd` bestehen.	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
T_6.1-2	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 *table*, derjenige des Datensatz-Tags ist *row*, die Spalten-Tags heissen *c1, c2, ...* Die Spalten-Tags beginnen immer mit c1 und erhöhen sich um 1. Das heisst, dass keine Lücke existieren darf. Denn ein NULL-Wert wird in der zugehörigen XML-Datei durch Fehlen der entsprechenden Spalte ausgedrückt. **Beispiel** Siehe das Beispiel `table2.xsd` im Anhang D.3a.	M

ID	Beschreibung Anforderung	M/K
T_6.1-3	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: clobType, blobType, dateType, timeType, dateTimeType	M
T_6.1-4	Mehrfachwerte von fortgeschrittenen oder strukturierten Typen müssen als separate Elemente innerhalb der Zellen-Tags gespeichert werden. Die Namen der individuellen Elemente eines ARRAY sind a1, a2, … Die Namen der individuellen Elemente eines UDT sind u1, u2, … Die Namen beginnen immer mit a1 bzw. u1 und erhöhen sich um 1. Das heisst, dass keine Lücke existieren darf. Denn ein NULL-Wert wird in der zugehörigen XML-Datei durch Fehlen der entsprechenden Spalte ausgedrückt. **Beispiel** Siehe das Beispiel `table0.xsd` im Anhang D.3c.	M

6.2 *Large-Object*-Datenzellen

ID	Beschreibung Anforderung	M/K
T_6.2-1	*Large Objects* können inline in der `table[Zahl].xml` Datei gespeichert werden oder als separate Dateieinträge in der SIARD-Datei (intern) oder als eigenstehende Dateien im Dateisystem ausserhalb der SIARD-Datei (extern).	M

Wenn die *Large Objects* als separate Dateien (intern oder extern) gespeichert werden, sind *file* und *length* zwingend in einer LOB-Zelle der `table[Zahl].xsd` Datei zu speichern. Werden *Large Objects* inline gespeichert, sind diese optional:

Bezeichnung	Bedeutung	M/K
file	Wenn das *Large Object* separat gespeichert ist, bezeichnet dieses Element den Speicherort und den Namen der *Large-Object*-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.	M[17]
length	Länge (für BLOBs in Bytes, für CLOBs und XML in Zeichen)	M[17]

[17] K, wenn das *Large Object* inline in der table[zahl].xml Datei gespeichert wird.

Bezeichnung	Bedeutung	M/K
digestType	Enthält den Typ der Integritätsinformation (digest): “MD5”, “SHA-1” oder “SHA-256”. ***Empfehlung*** *Für alle LOBs die in separaten Dateien gespeichert werden, sollte das Attribut (in Kombination mit dem digest) gesetzt werden.*	K
digest	Integritätsinformation zum CLOB oder BLOB. ***Empfehlung*** *Für alle LOBs die in separaten Dateien gespeichert werden, sollte das Attribut (in Kombination mit dem digestType) gesetzt werden.*	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 *dateType* und *dateTimeType* erzwungen.	M
T_6.3-2	Daten, Zeiten und Zeitstempel müssen in UTC allenfalls mit terminierendem Z gespeichert sein. Diese Beschränkung wird in den Definitionen von dateType, timeType und dateTimeType erzwungen. ***Empfehlung*** *Alle Daten, Zeiten und Zeitstempel mit terminierendem Z abspeichern.*	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	Die Datei *table* besteht aus *row*-Elementen, welche die Daten einer Zeile unterteilt in die verschiedenen Spalten (*c1, c2 ...*) enthalten. **Beispiel** Siehe das Beispiel `table2.xml` im Anhang D.4a.	M
T_6.4-3	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.	M

ID	Beschreibung Anforderung	M/K
T_6.4-4	Wenn eine Zelle einer Spalte einen komplexen Wert enthält (ARRAY, UDT), wird sie durch eine Sequenz von Subelementen der Zelle repräsentiert (a1, a2, … für ARRAYs, u1, u2, … für UDTs), welche ihrerseits die entsprechenden Werte enthalten. Diese Werte können wiederum komplex sein. **Beispiel** Siehe das Beispiel `table0.xml` im Anhang D.4c.	K
T_6.4-5	Wenn eine Tabelle Daten der *Large-Object*-Typen (BLOB, CLOB, ...) enthält, können hierfür separate Dateien erzeugt und anstelle des Zelleninhalts der Speicherort der Datei abgelegt werden. Die Entscheidung, *Large Objects* in separaten Dateien statt inline zu speichern, obliegt der Software, welche die SIARD-Datei erzeugt. Um zu vermeiden, dass leere Ordner entstehen, werden die Ordner nur angelegt, wenn sie notwendig sind (also Daten beinhalten). Werden die *Large Objects* in einer separaten Datei gespeichert, müssen folgende Attribute zwingend gesetzt werden: *file* und *length*. Nebst diesen zwingenden Attributen existieren noch die folgenden optionalen Attribute *digestType* und *digest*. Der Wert des *file*-Attributs enthält die Datei-URI (URL-enkodiert, wenn möglich auf den nächstliegenden lobFolder bezogen), wo das LOB gespeichert ist. Der Wert des *length*-Attributs ist die Länge (für BLOBs in Bytes, für CLOBs und XML in Zeichen), und das optionale Attribut *digest* enthält Integritätsinformation gemäss dem optionalen Attribut *digestType*. **Beispiel** Siehe das Beispiel `table7.xml` im Anhang D.4b. ***Empfehlung*** *Es wird ausdrücklich empfohlen, entweder alle oder kein Large Object in einer Spalte inline zu speichern.* *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.*	M

7 Version und Gültigkeit der Spezifikation

Die Spezifikation liegt in der Version 2.1 vor.

8 Change-Management-Prozess

Das Change-Management dieses Standards lehnt sich an [eCH-0150], Szenario 3, an.

9 Haftungsausschluss/Hinweise auf Rechte Dritter

Die KOST haftet in keinem Fall für Entscheidungen oder Massnahmen, welche der Benutzer auf Grund dieses Dokuments trifft und / oder ergreift. Sie kann keine Zusicherung oder Garantie auf Aktualität, Vollständigkeit, Richtigkeit bzw. Fehlerfreiheit der zur Verfügung gestellten Informationen und Dokumente geben. Jede Haftung für Schäden, welche dem Benutzer aus dem Gebrauch des vorliegenden Standards entstehen, ist, soweit gesetzlich zulässig, wegbedungen.

10 Urheberrechte

Der vorliegende Standard ist das geistige Eigentum seiner Autorinnen und Autoren. Sie verpflichten sich, dieses kostenlos zur uneingeschränkten Nutzung und Weiterentwicklung zur Verfügung zu stellen.

Anhang A – Mitarbeit & Überprüfung

Karin Bredenberg, National Archives of Sweden

Hedi Bruggisser, Staatsarchiv Thurgau

Georg Büchler, KOST

Marcel Büchler, Schweizerisches Bundesarchiv

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)

Rebekka Plüss, Staatsarchiv Zürich

Krystyna Ohnesorge, Schweizerisches Bundesarchiv

Lauri Rätsep, National Archives of Estonia

Claire Röthlisberger-Jourdan, KOST

Hélder Silva, KEEP SOLUTIONS, LDA

Mario Spuler, Fachlabor Gubler

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	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	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. Eine relationale Datenbank besteht somit aus einer Menge strukturierter Datenbankobjekten (z.B. Schema, View) sowie den Tabelleninhalten. 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.
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.
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.

Begriff	Beschreibung
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
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), 1.0 (Fifth Edition) W3C Recommendation 26 November 2008 https://www.w3.org/TR/REC-xml/
ZIP	ZIP File Format Specification, Version 6.3.3 September 1, 2012 PKWARE Inc. http://www.pkware.com/documents/casestudies/APPNOTE.TXT

Anhang D – Auszüge aus Beispiel ech-0165_oe.siard

Zu SIARD- 2.1 existiert als Beilage die SIARD-Datei ech-0165_oe.siard.
Die Daten im Anhang D sind Teile aus dieser Datei.

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"?>
<!-- ==================================================================
XML schema for meta data of the SIARD Format 2.1
Application: Software-Independent Archival of Relational Databases
Platform   : XML 1.0, XML Schema 2001
Description: This XML schema definition defines the structure
             of the meta data in the SIARD format 2.1.
=======================================================================
Copyright  : 2007, 2014, 2018, Swiss Federal Archives, Berne, Switzerland
=================================================================== -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="http://www.bar.admin.ch/xmlns/siard/2/metadata.xsd"
targetNamespace="http://www.bar.admin.ch/xmlns/siard/2/metadata.xsd" elementFormDefault="qualified" attributeFormDefault="unqualified"
version="2.1" id="metadata">
  <!-- 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 -->
        <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 codes over all primary data in folder "content" -->
        <xs:element name="messageDigest" type="messageDigestType" minOccurs="0" maxOccurs="unbounded"/>
        <!-- DNS name of client machine from which connection to the database was established 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 (JDBC URL) 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>
    <xs:attribute name="version" type="versionType" use="required"/>
    <!-- constraint: version number with release -->
  </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" 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 -->
    <xs:element name="types" type="typesType" minOccurs="0"/>
    <!-- list of tables in the schema -->
    <xs:element name="tables" type="tablesType" minOccurs="0"/>
    <!-- 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 -->
<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" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<!-- complex type type -->
<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"/>
    <!--  predefined base SQL:2008 type of (DISTINCT) type -->
    <xs:element name="base" type="predefinedTypeType" minOccurs="0"/>
    <!--  alternatively list of attributes (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 -->
<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" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<!-- complex type attribute -->
<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>
      <!-- either predefined or structured -->
      <xs:sequence>
        <!-- SQL:2008 data type of the column -->
        <xs:element name="type" type="predefinedTypeType"/>
      </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"/>
      </xs:sequence>
    </xs:choice>
    <!-- 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"/>
    <!-- default value -->
    <xs:element name="defaultValue" type="xs:string" minOccurs="0"/>
    <!--  SQL_1999 cardinality for ARRAY type -->
    <xs:element name="cardinality" type="xs:integer" 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" 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="uniqueKeyType" 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" 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" 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"/>
    <!-- folder for LOBs relative to lobFolder of nearest containing
      element for internally or externally stored LOBs -->
    <xs:element name="lobFolder" type="xs:anyURI" minOccurs="0"/>
    <xs:choice>
      <!-- either predefined or structured -->
      <xs:sequence>
        <!-- SQL:2008 predefined data type of the column -->
        <xs:element name="type" type="predefinedTypeType"/>
        <!-- mimeType makes sense only for LOBs and is only informatory -->
        <xs:element name="mimeType" type="xs:string" minOccurs="0"/>
      </xs:sequence>
      <xs:sequence>
```

```
        <!-- SQL:2008 schema of UDT name of the column -->
        <xs:element name="typeSchema" type="xs:string" minOccurs="0"/>
        <!-- SQL:2008 name of UDT of the column -->
        <xs:element name="typeName" type="xs:string"/>
      </xs:sequence>
    </xs:choice>
    <!-- original data type of the column -->
    <xs:element name="typeOriginal" type="xs:string" minOccurs="0"/>
    <!-- SQL:2008 attribute list of the column -->
    <xs:element name="fields" type="fieldsType" minOccurs="0"/>
    <!-- nullability (default: true) -->
    <xs:element name="nullable" type="xs:boolean" minOccurs="0"/>
    <!-- default value -->
    <xs:element name="defaultValue" type="xs:string" minOccurs="0"/>
    <!--  SQL_1999 cardinality for ARRAY type -->
    <xs:element name="cardinality" type="xs:integer" minOccurs="0"/>
    <!-- unique, references, check column constraints
     are stored as table constraints -->
    <!-- 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" 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>
    <!-- attribute name or array element position (1-based) -->
    <xs:element name="name" type="xs:string"/>
    <!--  folder for LOBs relative to lobFolder of nearest containing
      element for internally or externally stored LOBs -->
    <xs:element name="lobFolder" type="xs:anyURI" minOccurs="0"/>
    <!-- SQL:2008 sub field list of the field -->
    <xs:element name="fields" type="fieldsType" minOccurs="0"/>
    <!--  mimeType makes sense only for LOBs and is only informatory-->
    <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 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" 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" 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="referentialActionType" minOccurs="0"/>
    <!-- ON UPDATE action e.g. ON UPDATE SET DEFAULT -->
    <xs:element name="updateAction" type="referentialActionType" 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="uniqueKeyType" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<!-- complex type uniqueKey -->
<xs:complexType name="uniqueKeyType">
  <xs:annotation>
    <xs:documentation>unique (primary or candidate) key element in siardArchive</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <!-- database name of the unique key -->
    <xs:element name="name" type="xs:string"/>
    <!-- description of the unique key's meaning and content -->
    <xs:element name="description" type="xs:string" minOccurs="0"/>
    <!-- columns belonging to the unique key -->
    <xs:element name="column" type="xs:string" 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" 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" 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 BEFORE, AFTER or INSTEAD OF -->
    <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" maxOccurs="unbounded"/>
  </xs:sequence>
</xs:complexType>
<!-- complex type routine -->
<xs:complexType name="routineType">
  <xs:annotation>
    <xs:documentation>Routine</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <!--  specific (unique) name of routine in schema -->
    <xs:element name="specificName" type="xs:string"/>
    <!-- database (possible overloaded) 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" 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>
      <!-- either predefined or structured -->
      <xs:sequence>
        <!-- SQL:2008 data type of the column -->
        <xs:element name="type" type="predefinedTypeType"/>
      </xs:sequence>
      <xs:sequence>
        <!-- SQL:2008 schema of UDT name of the column -->
        <xs:element name="typeSchema" type="xs:string" minOccurs="0"/>
        <!-- SQL:2008 name of UDT of the column -->
        <xs:element name="typeName" type="xs:string"/>
      </xs:sequence>
    </xs:choice>
    <!-- original data type of the column -->
    <xs:element name="typeOriginal" type="xs:string" minOccurs="0"/>
    <!--  SQL_1999 cardinality for ARRAY type -->
    <xs:element name="cardinality" type="xs:integer" 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 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="0" 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" 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" 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>
  <!--  complex type for messageDigest with separate algorithm field -->
  <xs:complexType name="messageDigestType">
    <xs:annotation>
      <xs:documentation>Message digests with algorithm ("MD5", "SHA-1" or "SHA-256") and hexadecimal or - for the SHA variants - Base64
code.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="digestType" type="digestTypeType"/>
      <xs:element name="digest" type="xs:string"/>
    </xs:sequence>
  </xs:complexType>
  <!-- simple type fpr predefined SQL:2008 types -->
  <xs:simpleType name="predefinedTypeType">
    <xs:annotation>
      <xs:documentation>predefinedTypeType is constrained to valid SQL:2008 data type values</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:pattern value="INTEGER|INT|SMALLINT|BIGINT"/>
      <xs:pattern value="(NUMERIC|DECIMAL|DEC)(\s*\(\s*[1-9]\d*\s*(,\s*\d+\s*)?\))?"/>
      <xs:pattern value="REAL|DOUBLE PRECISION"/>
      <xs:pattern value="FLOAT(\s*\(\s*[1-9]\d*\s*\))?"/>
      <xs:pattern value="(CHARACTER|CHAR)(\s*\(\s*[1-9]\d*\s*\))?"/>
      <xs:pattern value="(CHARACTER\s+VARYING|CHAR\s+VARYING|VARCHAR)(\s*\(\s*[1-9]\d*\s*\))?"/>
      <xs:pattern value="(CHARACTER\s+LARGE\s+OBJECT|CLOB)(\s*\(\s*[1-9]\d*(\s*(K|M|G))?\s*\))?"/>
      <xs:pattern value="(NATIONAL\s+CHARACTER|NATIONAL\s+CHAR|NCHAR)(\s*\(\s*[1-9]\d*\s*\))?"/>
      <xs:pattern value="(NATIONAL\s+CHARACTER\s+VARYING|NATIONAL\s+CHAR\s+VARYING|NCHAR VARYING)(\s*\(\s*[1-9]\d*\s*\))?"/>
      <xs:pattern value="(NATIONAL\s+CHARACTER\s+LARGE\s+OBJECT|NCHAR\s+LARGE\s+OBJECT|NCLOB)(\s*\(\s*[1-
9]\d*(\s*(K|M|G))?\s*\))?"/>
      <xs:pattern value="XML"/>
      <xs:pattern value="BINARY(\s*\(\s*[1-9]\d*\s*\))?"/>
      <xs:pattern value="(BINARY\s+VARYING|VARBINARY)(\s*\(\s*[1-9]\d*\s*\))?"/>
      <xs:pattern value="(BINARY\s+LARGE\s+OBJECT|BLOB)(\s*\(\s*[1-9]\d*(\s*(K|M|G))?\s*\))?"/>
      <xs:pattern value="DATE"/>
      <xs:pattern value="(TIME|TIME\s+WITH\s+TIME\s+ZONE)(\s*\(\s*[1-9]\d*\s*\))?"/>
```

```
      <xs:pattern value="(TIMESTAMP|TIMESTAMP\s+WITH\s+TIME\s+ZONE)(\s*\(\s*(0|([1-9]\d*))\s*\))?"/>
      <xs:pattern value="INTERVAL\s+(((YEAR|MONTH|DAY|HOUR|MINUTE)(\s*\(\s*[1-
9]\d*\s*\))?(\s+TO\s+(MONTH|DAY|HOUR|MINUTE|SECOND)(\s*\(\s*[1-9]\d*\s*\))?)?)|(SECOND(\s*\(\s*[1-9]\d*\s*(,\s*\d+\s*)?\))?))"/>
      <xs:pattern value="BOOLEAN"/>
      <!-- exact numerics (BIGINT from SQL:2008) -->
      <!-- approximate numerics -->
      <!-- character strings -->
      <!-- BINARY strings from SQL:2008 -->
      <!-- datetimes -->
      <!-- BOOLEAN -->
    </xs:restriction>
  </xs:simpleType>
  <!--  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>
  <!-- simple type for version number -->
  <xs:simpleType name="versionType">
    <xs:annotation>
      <xs:documentation>versionType is constrained to "2.1" for conformity with this XML schema</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="collapse"/>
      <xs:enumeration value="2.1"/>
      <!--  to be extended later with
    <xs.enumeration value="2.2"/>
    etc. -->
    </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:minLength value="1"/>
      <xs:pattern value="([a-z]|[A-Z])([a-z]|[A-Z]|[0-9]).*"/>
    </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="INSTEAD OF"/>
      <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 referential action of a foreign key -->
  <xs:simpleType name="referentialActionType">
    <xs:annotation>
      <xs:documentation>referential action is CASCADE, SET NULL, SET DEFAULT, RESTRICT, or NO ACTION</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="CASCADE"/>
      <xs:enumeration value="SET NULL"/>
      <xs:enumeration value="SET DEFAULT"/>
      <xs:enumeration value="RESTRICT"/>
      <xs:enumeration value="NO ACTION"/>
    </xs:restriction>
  </xs:simpleType>
  <!-- simple type for the category of a column or a parameter -->
  <xs:simpleType name="categoryType">
    <xs:annotation>
      <xs:documentation>category of advanced or structured data types is "distinct" or "udt" for conformity with this XML schema</xs:documenta-
tion>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="collapse"/>
      <xs:enumeration value="distinct"/>
      <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
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<siardArchive xmlns="http://www.bar.admin.ch/xmlns/siard/2/metadata.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" version="2.1" xsi:schemaLocation="http://www.bar.admin.ch/xmlns/siard/2/metadata.xsd metadata.xsd">
  <dbname>OE Sample Database enhanced</dbname>
  <description>Record with PRODUCT_ID 4000 in table PRODUCT_INFORMATION has a picture.version with updated .xsd</description>
  <archiver>Claire Roethlisberger</archiver>
  <archiverContact>claire.roethlisberger@kost.admin.ch</archiverContact>
  <dataOwner>Oracle (OE database) and Swiss Federal Archives (enhancement)</dataOwner>
  <dataOriginTimespan>2000-2007</dataOriginTimespan>
  <producerApplication>SiardGui 2.1.89 Swiss Federal Archives, Berne, Switzerland, 2007-2018</producerApplication>
  <archivalDate>2018-01-30Z</archivalDate>
  <clientMachine>VMW10.enterag.ch</clientMachine>
  <databaseProduct>Oracle Oracle Database 12c Release 12.1.0.1.0 - 64bit Production</databaseProduct>
  <connection>jdbc:oracle:thin:@localhost:1521:ORCL</connection>
  <databaseUser>OE</databaseUser>
  <schemas>
    <schema>
      <name>HR</name>
      <folder>schema0</folder>
      <tables>
        ...
        <table>
          <name>EMPLOYEES</name>
          <folder>table2</folder>
          <description>employees table. Contains 107 rows. References with departments, jobs, job_history tables. Contains a self reference.
          </description>
          <columns>
            <column>
              <name>EMPLOYEE_ID</name>
              <type>DECIMAL(6)</type>
              <typeOriginal>"NUMBER"</typeOriginal>
              <nullable>false</nullable>
              <description>Primary key of employees table.</description>
            </column>
            <column>
              <name>FIRST_NAME</name>
              <type>VARCHAR(20)</type>
              <typeOriginal>"VARCHAR2"</typeOriginal>
              <description>First name of the employee. A not null column.</description>
            </column>
            <column>
              <name>LAST_NAME</name>
              <type>VARCHAR(25)</type>
              <typeOriginal>"VARCHAR2"</typeOriginal>
              <nullable>false</nullable>
              <description>Last name of the employee. A not null column.</description>
            </column>
            <column>
              <name>EMAIL</name>
              <type>VARCHAR(25)</type>
              <typeOriginal>"VARCHAR2"</typeOriginal>
              <nullable>false</nullable>
              <description>Email id of the employee</description>
            </column>
            <column>
              <name>PHONE_NUMBER</name>
              <type>VARCHAR(20)</type>
              <typeOriginal>"VARCHAR2"</typeOriginal>
              <description>Phone number of the employee; includes country code and area code</description>
            </column>
            <column>
              <name>HIRE_DATE</name>
              <type>DATE</type>
              <typeOriginal>"DATE"</typeOriginal>
```

```
      <nullable>false</nullable>
      <description>Date when the employee started on this job. A not null column.</description>
    </column>
    <column>
      <name>JOB_ID</name>
      <type>VARCHAR(10)</type>
      <typeOriginal>"VARCHAR2"</typeOriginal>
      <nullable>false</nullable>
      <description>Current job of the employee; foreign key to job_id column of the jobs table. A not null column.</description>
    </column>
    <column>
      <name>SALARY</name>
      <type>DECIMAL(8, 2)</type>
      <typeOriginal>"NUMBER"</typeOriginal>
      <description>Monthly salary of the employee. Must be greater than zero (enforced by constraint emp_salary_min)</description>
    </column>
    <column>
      <name>COMMISSION_PCT</name>
      <type>DECIMAL(2, 2)</type>
      <typeOriginal>"NUMBER"</typeOriginal>
      <description>Commission percentage of the employee; Only employees in sales department elgible for commission percentage
      </description>
    </column>
    <column>
      <name>MANAGER_ID</name>
      <type>DECIMAL(6)</type>
      <typeOriginal>"NUMBER"</typeOriginal>
      <description>Manager id of the employee; has same domain as manager_id in departments table. Foreign key to employee_id
        column of employees table. (useful for reflexive joins and CONNECT BY query)</description>
    </column>
    <column>
      <name>DEPARTMENT_ID</name>
      <type>DECIMAL(4)</type>
      <typeOriginal>"NUMBER"</typeOriginal>
      <description>Department id where employee works; foreign key to department_id column of the departments table</description>
    </column>
  </columns>
  <primaryKey>
    <name>EMP_EMP_ID_PK</name>
    <column>EMPLOYEE_ID</column>
  </primaryKey>
  <foreignKeys>
    <foreignKey>
      <name>EMP_DEPT_FK</name>
      <referencedSchema>HR</referencedSchema>
      <referencedTable>DEPARTMENTS</referencedTable>
      <reference>
        <column>DEPARTMENT_ID</column>
        <referenced>DEPARTMENT_ID</referenced>
      </reference>
      <deleteAction>RESTRICT</deleteAction>
      <updateAction>CASCADE</updateAction>
    </foreignKey>
    <foreignKey>
      <name>EMP_MANAGER_FK</name>
      <referencedSchema>HR</referencedSchema>
      <referencedTable>EMPLOYEES</referencedTable>
      <reference>
        <column>MANAGER_ID</column>
        <referenced>EMPLOYEE_ID</referenced>
      </reference>
      <deleteAction>RESTRICT</deleteAction>
      <updateAction>CASCADE</updateAction>
    </foreignKey>
    <foreignKey>
      <name>EMP_JOB_FK</name>
      <referencedSchema>HR</referencedSchema>
      <referencedTable>JOBS</referencedTable>
      <reference>
        <column>JOB_ID</column>
```

```
            <referenced>JOB_ID</referenced>
          </reference>
          <deleteAction>RESTRICT</deleteAction>
          <updateAction>CASCADE</updateAction>
        </foreignKey>
      </foreignKeys>
      <candidateKeys>
        <candidateKey>
          <name>EMP_EMAIL_UK</name>
          <column>EMAIL</column>
        </candidateKey>
      </candidateKeys>
      <rows>107</rows>
    </table>
    ...
  </tables>
</schema>
<schema>
  <name>OE</name>
  <folder>schema1</folder>
  <types>
    <type>
      <name>CUST_ADDRESS_TYP</name>
      <category>udt</category>
      <instantiable>true</instantiable>
      <final>true</final>
      <attributes>
        <attribute>
          <name>STREET_ADDRESS</name>
          <type>VARCHAR(40)</type>
        </attribute>
        <attribute>
          <name>POSTAL_CODE</name>
          <type>VARCHAR(10)</type>
        </attribute>
        <attribute>
          <name>CITY</name>
          <type>VARCHAR(30)</type>
        </attribute>
        <attribute>
          <name>STATE_PROVINCE</name>
          <type>VARCHAR(10)</type>
        </attribute>
        <attribute>
          <name>COUNTRY_ID</name>
          <type>CHARACTER(2)</type>
        </attribute>
      </attributes>
    </type>
    <type>
      <name>ORDER_TYP</name>
      <category>udt</category>
      <instantiable>true</instantiable>
      <final>true</final>
      <attributes>
        <attribute>
          <name>ORDER_ID</name>
          <type>DECIMAL(12)</type>
        </attribute>
        <attribute>
          <name>ORDER_MODE</name>
          <type>VARCHAR(8)</type>
        </attribute>
        <attribute>
          <name>CUSTOMER_REF</name>
          <typeSchema>OE</typeSchema>
          <typeName>CUSTOMER_TYP</typeName>
        </attribute>
        <attribute>
          <name>ORDER_STATUS</name>
```

```
        <type>DECIMAL(2)</type>
      </attribute>
      <attribute>
        <name>ORDER_TOTAL</name>
        <type>DECIMAL(8, 2)</type>
      </attribute>
      <attribute>
        <name>SALES_REP_ID</name>
        <type>DECIMAL(6)</type>
      </attribute>
      <attribute>
        <name>ORDER_ITEM_LIST</name>
        <typeSchema>OE</typeSchema>
        <typeName>ORDER_ITEM_TYP</typeName>
        <cardinality>2147483647</cardinality>
      </attribute>
    </attributes>
  </type>
  <type>
    <name>CUSTOMER_TYP</name>
    <category>udt</category>
    <instantiable>true</instantiable>
    <final>true</final>
    <attributes>
      <attribute>
        <name>CUSTOMER_ID</name>
        <type>DECIMAL(6)</type>
      </attribute>
      <attribute>
        <name>CUST_FIRST_NAME</name>
        <type>VARCHAR(20)</type>
      </attribute>
      <attribute>
        <name>CUST_LAST_NAME</name>
        <type>VARCHAR(20)</type>
      </attribute>
      <attribute>
        <name>CUST_ADDRESS</name>
        <typeSchema>OE</typeSchema>
        <typeName>CUST_ADDRESS_TYP</typeName>
      </attribute>
      <attribute>
        <name>PHONE_NUMBERS</name>
        <type>VARCHAR(25)</type>
        <cardinality>5</cardinality>
      </attribute>
      <attribute>
        <name>NLS_LANGUAGE</name>
        <type>VARCHAR(3)</type>
      </attribute>
      <attribute>
        <name>NLS_TERRITORY</name>
        <type>VARCHAR(40)</type>
      </attribute>
      <attribute>
        <name>CREDIT_LIMIT</name>
        <type>DECIMAL(9, 2)</type>
      </attribute>
      <attribute>
        <name>CUST_EMAIL</name>
        <type>VARCHAR(40)</type>
      </attribute>
      <attribute>
        <name>CUST_ORDERS</name>
        <typeSchema>OE</typeSchema>
        <typeName>ORDER_TYP</typeName>
        <cardinality>2147483647</cardinality>
      </attribute>
    </attributes>
  </type>
```

```
<type>
  <name>ORDER_ITEM_TYP</name>
  <category>udt</category>
  <instantiable>true</instantiable>
  <final>true</final>
  <attributes>
    <attribute>
      <name>ORDER_ID</name>
      <type>DECIMAL(12)</type>
    </attribute>
    <attribute>
      <name>LINE_ITEM_ID</name>
      <type>DECIMAL(3)</type>
    </attribute>
    <attribute>
      <name>UNIT_PRICE</name>
      <type>DECIMAL(8, 2)</type>
    </attribute>
    <attribute>
      <name>QUANTITY</name>
      <type>DECIMAL(8)</type>
    </attribute>
    <attribute>
      <name>PRODUCT_REF</name>
      <typeSchema>OE</typeSchema>
      <typeName>PRODUCT_INFORMATION_TYP</typeName>
    </attribute>
  </attributes>
</type>
<type>
  <name>PRODUCT_INFORMATION_TYP</name>
  <category>udt</category>
  <instantiable>true</instantiable>
  <final>true</final>
  <attributes>
    <attribute>
      <name>PRODUCT_ID</name>
      <type>DECIMAL(6)</type>
    </attribute>
    <attribute>
      <name>PRODUCT_NAME</name>
      <type>VARCHAR(50)</type>
    </attribute>
    <attribute>
      <name>PRODUCT_DESCRIPTION</name>
      <type>VARCHAR(2000)</type>
    </attribute>
    <attribute>
      <name>CATEGORY_ID</name>
      <type>DECIMAL(2)</type>
    </attribute>
    <attribute>
      <name>WEIGHT_CLASS</name>
      <type>DECIMAL(1)</type>
    </attribute>
    <attribute>
      <name>WARRANTY_PERIOD</name>
      <type>INTERVAL YEAR TO MONTH</type>
    </attribute>
    <attribute>
      <name>SUPPLIER_ID</name>
      <type>DECIMAL(6)</type>
    </attribute>
    <attribute>
      <name>PRODUCT_STATUS</name>
      <type>VARCHAR(20)</type>
    </attribute>
    <attribute>
      <name>LIST_PRICE</name>
      <type>DECIMAL(8, 2)</type>
```

```
        </attribute>
        <attribute>
          <name>MIN_PRICE</name>
          <type>DECIMAL(8, 2)</type>
        </attribute>
        <attribute>
          <name>CATALOG_URL</name>
          <type>VARCHAR(50)</type>
        </attribute>
        <attribute>
          <name>INVENTORY_LIST</name>
          <typeSchema>OE</typeSchema>
          <typeName>INVENTORY_TYP</typeName>
          <cardinality>2147483647</cardinality>
        </attribute>
      </attributes>
    </type>
    <type>
      <name>INVENTORY_TYP</name>
      <category>udt</category>
      <instantiable>true</instantiable>
      <final>true</final>
      <attributes>
        <attribute>
          <name>PRODUCT_ID</name>
          <type>DECIMAL(6)</type>
        </attribute>
        <attribute>
          <name>WAREHOUSE</name>
          <typeSchema>OE</typeSchema>
          <typeName>WAREHOUSE_TYP</typeName>
        </attribute>
        <attribute>
          <name>QUANTITY_ON_HAND</name>
          <type>DECIMAL(8)</type>
        </attribute>
      </attributes>
    </type>
    <type>
      <name>WAREHOUSE_TYP</name>
      <category>udt</category>
      <instantiable>true</instantiable>
      <final>true</final>
      <attributes>
        <attribute>
          <name>WAREHOUSE_ID</name>
          <type>DECIMAL(3)</type>
        </attribute>
        <attribute>
          <name>WAREHOUSE_NAME</name>
          <type>VARCHAR(35)</type>
        </attribute>
        <attribute>
          <name>LOCATION_ID</name>
          <type>DECIMAL(4)</type>
        </attribute>
      </attributes>
    </type>
  </types>
  <tables>
    <table>
      <name>CUSTOMERS</name>
      <folder>table0</folder>
      <description>Contains customers data either entered by an employee or by the customer him/herself over the Web.</description>
      <columns>
        <column>
          <name>CUSTOMER_ID</name>
          <type>DECIMAL(6)</type>
          <typeOriginal>"NUMBER"</typeOriginal>
          <nullable>false</nullable>
```

```
    <description>Primary key column.</description>
  </column>
  <column>
    <name>CUST_FIRST_NAME</name>
    <type>VARCHAR(20)</type>
    <typeOriginal>"VARCHAR2"</typeOriginal>
    <nullable>false</nullable>
    <description>NOT NULL constraint.</description>
  </column>
  <column>
    <name>CUST_LAST_NAME</name>
    <type>VARCHAR(20)</type>
    <typeOriginal>"VARCHAR2"</typeOriginal>
    <nullable>false</nullable>
    <description>NOT NULL constraint.</description>
  </column>
  <column>
    <name>CUST_ADDRESS</name>
    <typeSchema>OE</typeSchema>
    <typeName>CUST_ADDRESS_TYP</typeName>
    <fields>
      <field>
        <name>STREET_ADDRESS</name>
      </field>
      <field>
        <name>POSTAL_CODE</name>
      </field>
      <field>
        <name>CITY</name>
      </field>
      <field>
        <name>STATE_PROVINCE</name>
      </field>
      <field>
        <name>COUNTRY_ID</name>
      </field>
    </fields>
    <description>Object column of type address_typ.</description>
  </column>
  <column>
    <name>PHONE_NUMBERS</name>
    <type>VARCHAR(25)</type>
    <typeOriginal>VARCHAR(25) ARRAY[5]</typeOriginal>
    <fields>
      <field>
        <name>PHONE_NUMBERS[1]</name>
      </field>
      <field>
        <name>PHONE_NUMBERS[2]</name>
      </field>
      <field>
        <name>PHONE_NUMBERS[3]</name>
      </field>
      <field>
        <name>PHONE_NUMBERS[4]</name>
      </field>
      <field>
        <name>PHONE_NUMBERS[5]</name>
      </field>
    </fields>
    <cardinality>5</cardinality>
    <description>Varray column of type phone_list_typ</description>
  </column>
  <column>
    <name>NLS_LANGUAGE</name>
    <type>VARCHAR(3)</type>
    <typeOriginal>"VARCHAR2"</typeOriginal>
  </column>
  <column>
    <name>NLS_TERRITORY</name>
```

```
    <type>VARCHAR(30)</type>
    <typeOriginal>"VARCHAR2"</typeOriginal>
  </column>
  <column>
    <name>CREDIT_LIMIT</name>
    <type>DECIMAL(9, 2)</type>
    <typeOriginal>"NUMBER"</typeOriginal>
    <description>Check constraint.</description>
  </column>
  <column>
    <name>CUST_EMAIL</name>
    <type>VARCHAR(40)</type>
    <typeOriginal>"VARCHAR2"</typeOriginal>
  </column>
  <column>
    <name>ACCOUNT_MGR_ID</name>
    <type>DECIMAL(6)</type>
    <typeOriginal>"NUMBER"</typeOriginal>
    <description>References hr.employees.employee_id.</description>
  </column>
  <column>
    <name>CUST_GEO_LOCATION</name>
    <typeSchema>MDSYS</typeSchema>
    <typeName>SDO_GEOMETRY</typeName>
    <fields>
      <field>
        <name>SDO_GTYPE</name>
      </field>
      <field>
        <name>SDO_SRID</name>
      </field>
      <field>
        <name>SDO_POINT</name>
        <fields>
          <field>
            <name>X</name>
          </field>
          <field>
            <name>Y</name>
          </field>
          <field>
            <name>Z</name>
          </field>
        </fields>
      </field>
      <field>
        <name>SDO_ELEM_INFO</name>
      </field>
      <field>
        <name>SDO_ORDINATES</name>
      </field>
    </fields>
    <description>SDO (spatial) column.</description>
  </column>
  <column>
    <name>DATE_OF_BIRTH</name>
    <type>DATE</type>
    <typeOriginal>"DATE"</typeOriginal>
  </column>
  <column>
    <name>MARITAL_STATUS</name>
    <type>VARCHAR(20)</type>
    <typeOriginal>"VARCHAR2"</typeOriginal>
  </column>
  <column>
    <name>GENDER</name>
    <type>VARCHAR(1)</type>
    <typeOriginal>"VARCHAR2"</typeOriginal>
  </column>
  <column>
```

```
        <name>INCOME_LEVEL</name>
        <type>VARCHAR(20)</type>
        <typeOriginal>"VARCHAR2"</typeOriginal>
      </column>
    </columns>
    <primaryKey>
      <name>CUSTOMERS_PK</name>
      <column>CUSTOMER_ID</column>
    </primaryKey>
    <foreignKeys>
      <foreignKey>
        <name>CUSTOMERS_ACCOUNT_MANAGER_FK</name>
        <referencedSchema>HR</referencedSchema>
        <referencedTable>EMPLOYEES</referencedTable>
        <reference>
          <column>ACCOUNT_MGR_ID</column>
          <referenced>EMPLOYEE_ID</referenced>
        </reference>
        <deleteAction>SET NULL</deleteAction>
        <updateAction>CASCADE</updateAction>
      </foreignKey>
    </foreignKeys>
    <rows>319</rows>
  </table>
...
  <table>
    <name>WAREHOUSES</name>
    <folder>table7</folder>
    <description>Warehouse data unspecific to any industry.</description>
    <columns>
      <column>
        <name>WAREHOUSE_ID</name>
        <type>DECIMAL(3)</type>
        <typeOriginal>"NUMBER"</typeOriginal>
        <nullable>false</nullable>
        <description>Primary key column.</description>
      </column>
      <column>
        <name>WAREHOUSE_SPEC</name>
        <type>XML</type>
        <typeOriginal>"SYS"."XMLTYPE"</typeOriginal>
      </column>
      <column>
        <name>WAREHOUSE_NAME</name>
        <type>VARCHAR(35)</type>
        <typeOriginal>"VARCHAR2"</typeOriginal>
      </column>
      <column>
        <name>LOCATION_ID</name>
        <type>DECIMAL(4)</type>
        <typeOriginal>"NUMBER"</typeOriginal>
        <description>Primary key column, references hr.locations.location_id.</description>
      </column>
      <column>
        <name>WH_GEO_LOCATION</name>
        <typeSchema>MDSYS</typeSchema>
        <typeName>SDO_GEOMETRY</typeName>
        <fields>
          <field>
            <name>SDO_GTYPE</name>
          </field>
          <field>
            <name>SDO_SRID</name>
          </field>
          <field>
            <name>SDO_POINT</name>
            <fields>
              <field>
                <name>X</name>
              </field>
```

```
                <field>
                  <name>Y</name>
                </field>
                <field>
                  <name>Z</name>
                </field>
              </fields>
            </field>
            <field>
              <name>SDO_ELEM_INFO</name>
            </field>
            <field>
              <name>SDO_ORDINATES</name>
            </field>
          </fields>
          <description>SDO (spatial) column.</description>
        </column>
      </columns>
      <primaryKey>
        <name>WAREHOUSES_PK</name>
        <column>WAREHOUSE_ID</column>
      </primaryKey>
      <foreignKeys>
        <foreignKey>
          <name>WAREHOUSES_LOCATION_FK</name>
          <referencedSchema>HR</referencedSchema>
          <referencedTable>LOCATIONS</referencedTable>
          <reference>
            <column>LOCATION_ID</column>
            <referenced>LOCATION_ID</referenced>
          </reference>
          <deleteAction>SET NULL</deleteAction>
          <updateAction>CASCADE</updateAction>
        </foreignKey>
      </foreignKeys>
      <rows>9</rows>
    </table>
  </tables>
  <views>
    <view>
      <name>ACCOUNT_MANAGERS</name>
      <queryOriginal>SELECT .account_mgr_id _MGR, .region_id, .cust_address.country_id , .cust_address.state_province , (*)
        _CUSTOMERSFROM c, countries crWHERE .cust_address.country_id = cr.country_idGROUP BY ROLLUP (c.account_mgr_id,
        cr.region_id, c.cust_address.country_id, c.cust_address.state_province)</queryOriginal>
      <columns>
        <column>
          <name>ACCT_MGR</name>
          <type>DECIMAL(6)</type>
          <typeOriginal>"NUMBER"</typeOriginal>
        </column>
        <column>
          <name>REGION</name>
          <type>DECIMAL(22)</type>
          <typeOriginal>"NUMBER"</typeOriginal>
        </column>
        <column>
          <name>COUNTRY</name>
          <type>CHARACTER(2)</type>
          <typeOriginal>"CHAR"</typeOriginal>
        </column>
        <column>
          <name>PROVINCE</name>
          <type>VARCHAR(10)</type>
          <typeOriginal>"VARCHAR2"</typeOriginal>
        </column>
        <column>
          <name>NUM_CUSTOMERS</name>
          <type>DECIMAL(22)</type>
          <typeOriginal>"NUMBER"</typeOriginal>
        </column>
```

```
        </columns>
        <rows>0</rows>
      </view>
      ...
    </views>
    <routines>
      <routine>
        <specificName>CATEGORY_DESCRIBE.CATALOG_TYP</specificName>
        <name>CATALOG_TYP</name>
      </routine>
      ...
      <routine>
        <specificName>GET_PHONE_NUMBER_F</specificName>
        <name>GET_PHONE_NUMBER_F</name>
        <returnType>VARCHAR</returnType>
        <parameters>
          <parameter>
            <name>P_IN</name><mode>IN</mode><type>DECIMAL(38)</type><typeOriginal>NUMBER</typeOriginal>
          </parameter>
          <parameter>
            <name>P_PHONELIST</name><mode>IN</mode><type>VARCHAR(25)</type><cardinality>5</cardinality>
          </parameter>
        </parameters>
      </routine>
      ...
    </routines>
  </schema>
  <schema>
    ...
  </schema>
</schemas>
<users>
  <user><name>OE</name></user>
  <user><name>HR</name></user>
</users>
<roles>
  <role><name>BI</name><admin>OE</admin></role>
  <role><name>PM</name><admin>OE</admin></role>
</roles>
<privileges>
  <privilege>
    <type>REFERENCES</type>
    <object>HR.COUNTRIES</object>
    <grantor>HR</grantor>
    <grantee>OE</grantee>
  </privilege>
  ...
</privileges>
</siardArchive>
```

D.3 Beispiele für die XML-Schemadefinition einer Tabelle

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

D.3a table2.xsd (Schemadefinition einer einfache Tabelle)

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<xs:schema xmlns="http://www.bar.admin.ch/xmlns/siard/2/table.xsd" xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.bar.admin.ch/xmlns/siard/2/table.xsd" elementFormDefault="qualified" attributeFormDefault="unqualified"
version="2.1">
  <!-- root element is the table element -->
  <xs:element name="table">
    <xs:complexType>
      <xs:annotation>
        <xs:documentation>Root element of a table of the SIARD archive. A table consists of rows.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="row" type="recordType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" type="versionType" use="required"/>
    </xs:complexType>
  </xs:element>
  <!-- simple type for version number -->
  <xs:simpleType name="versionType">
    <xs:annotation>
      <xs:documentation>versionType is constrained to "2.1" for conformity with this XML schema</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="collapse"/>
      <xs:enumeration value="2.1"/>
      <!--  to be extended later with  <xs.enumeration value="2.2"/>  etc. -->
    </xs:restriction>
  </xs:simpleType>
  <!-- complex type record -->
  <xs:complexType name="recordType">
    <xs:annotation>
      <xs:documentation>row type of a table of the SIARD archive. A row consists of columns.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="c1" type="xs:decimal"/>
      <xs:element name="c2" type="xs:string" minOccurs="0"/>
      <xs:element name="c3" type="xs:string"/>
      <xs:element name="c4" type="xs:string"/>
      <xs:element name="c5" type="xs:string" minOccurs="0"/>
      <xs:element name="c6" type="dateType"/>
      <xs:element name="c7" type="xs:string"/>
      <xs:element name="c8" type="xs:decimal" minOccurs="0"/>
      <xs:element name="c9" type="xs:decimal" minOccurs="0"/>
      <xs:element name="c10" type="xs:decimal" minOccurs="0"/>
      <xs:element name="c11" type="xs:decimal" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
  <!-- date type between 0001 and 9999 restricted to UTC -->
  <xs:simpleType name="dateType">
    <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>
</xs:schema>
```

D.3b table7.xsd (Schemadefinition einer Tabelle mit internen Large Objects)

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<xs:schema xmlns="http://www.bar.admin.ch/xmlns/siard/2/table.xsd" xmlns:xs="http://www.w3.org/2001/XMLSchema"
targetNamespace="http://www.bar.admin.ch/xmlns/siard/2/table.xsd" elementFormDefault="qualified" attributeFormDefault="unqualified"
version="2.1">
  <!-- root element is the table element -->
  <xs:element name="table">
    <xs:complexType>
      <xs:annotation>
        <xs:documentation>Root element of a table of the SIARD archive. A table consists of rows.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="row" type="recordType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" type="versionType" use="required"/>
    </xs:complexType>
  </xs:element>
  <!-- simple type for version number -->
  <xs:simpleType name="versionType">
    <xs:annotation>
      <xs:documentation>versionType is constrained to "2.1" for conformity with this XML schema</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="collapse"/>
      <xs:enumeration value="2.1"/>
      <!--  to be extended later with  <xs.enumeration value="2.2"/>  etc. -->
    </xs:restriction>
  </xs:simpleType>
  <!-- complex type record -->
  <xs:complexType name="recordType">
    <xs:annotation>
      <xs:documentation>row type of a table of the SIARD archive. A row consists of columns.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="c1" type="xs:decimal"/>
      <xs:element name="c2" type="clobType" minOccurs="0"/>
      <xs:element name="c3" type="xs:string" minOccurs="0"/>
      <xs:element name="c4" type="xs:decimal" minOccurs="0"/>
      <xs:element name="c5" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="u1" type="xs:decimal" minOccurs="0"/>
            <xs:element name="u2" type="xs:decimal" minOccurs="0"/>
            <xs:element name="u3" minOccurs="0">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="u1" type="xs:decimal" minOccurs="0"/>
                  <xs:element name="u2" type="xs:decimal" minOccurs="0"/>
                  <xs:element name="u3" type="xs:decimal" minOccurs="0"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element name="u4" type="xs:decimal" minOccurs="0"/>
            <xs:element name="u5" type="xs:decimal" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
    </xs:sequence>
  </xs:complexType>
  <!-- type for text large objects -->
  <xs:complexType name="clobType">
    <xs:annotation>
      <xs:documentation>a text large object can either be stored inline (as xs:string) or externally (addressed by URI). The digest makes sure,
        that the connection to the external object is not completely lost. The length is in characters, not in bytes.</xs:documentation>
    </xs:annotation>
    <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="digest" 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>
</xs:schema>
```

D.3c table0.xsd (Schemadefinition einer Tabelle mit “udt” und Array)

```
<?xml version="1.0" encoding="utf-8" standalone="no"?>
<xs:schema xmlns="http://www.bar.admin.ch/xmlns/siard/2/table.xsd" xmlns:xs="http://www.w3.org/2001/XMLSchema" target-
Namespace="http://www.bar.admin.ch/xmlns/siard/2/table.xsd" elementFormDefault="qualified" attributeFormDefault="unqualified" version="2.1">
  <!-- root element is the table element -->
  <xs:element name="table">
    <xs:complexType>
      <xs:annotation>
        <xs:documentation>Root element of a table of the SIARD archive. A table consists of rows.</xs:documentation>
      </xs:annotation>
      <xs:sequence>
        <xs:element name="row" type="recordType" minOccurs="0" maxOccurs="unbounded"/>
      </xs:sequence>
      <xs:attribute name="version" type="versionType" use="required"/>
    </xs:complexType>
  </xs:element>
  <!-- simple type for version number -->
  <xs:simpleType name="versionType">
    <xs:annotation>
      <xs:documentation>versionType is constrained to "2.1" for conformity with this XML schema</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:whiteSpace value="collapse"/>
      <xs:enumeration value="2.1"/>
      <!-- to be extended later with <xs.enumeration value="2.2"/> etc. -->
    </xs:restriction>
  </xs:simpleType>
  <!-- complex type record -->
  <xs:complexType name="recordType">
    <xs:annotation>
      <xs:documentation>row type of a table of the SIARD archive. A row consists of columns.</xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="c1" type="xs:decimal"/>
      <xs:element name="c2" type="xs:string"/>
      <xs:element name="c3" type="xs:string"/>
      <xs:element name="c4" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="u1" type="xs:string" minOccurs="0"/>
            <xs:element name="u2" type="xs:string" minOccurs="0"/>
            <xs:element name="u3" type="xs:string" minOccurs="0"/>
            <xs:element name="u4" type="xs:string" minOccurs="0"/>
            <xs:element name="u5" type="xs:string" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="c5" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="a1" type="xs:string" minOccurs="0"/>
            <xs:element name="a2" type="xs:string" minOccurs="0"/>
            <xs:element name="a3" type="xs:string" minOccurs="0"/>
```

```
            <xs:element name="a4" type="xs:string" minOccurs="0"/>
            <xs:element name="a5" type="xs:string" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="c6" type="xs:string" minOccurs="0"/>
      <xs:element name="c7" type="xs:string" minOccurs="0"/>
      <xs:element name="c8" type="xs:decimal" minOccurs="0"/>
      <xs:element name="c9" type="xs:string" minOccurs="0"/>
      <xs:element name="c10" type="xs:decimal" minOccurs="0"/>
      <xs:element name="c11" minOccurs="0">
        <xs:complexType>
          <xs:sequence>
            <xs:element name="u1" type="xs:decimal" minOccurs="0"/>
            <xs:element name="u2" type="xs:decimal" minOccurs="0"/>
            <xs:element name="u3" minOccurs="0">
              <xs:complexType>
                <xs:sequence>
                  <xs:element name="u1" type="xs:decimal" minOccurs="0"/>
                  <xs:element name="u2" type="xs:decimal" minOccurs="0"/>
                  <xs:element name="u3" type="xs:decimal" minOccurs="0"/>
                </xs:sequence>
              </xs:complexType>
            </xs:element>
            <xs:element name="u4" type="xs:decimal" minOccurs="0"/>
            <xs:element name="u5" type="xs:decimal" minOccurs="0"/>
          </xs:sequence>
        </xs:complexType>
      </xs:element>
      <xs:element name="c12" type="dateType" minOccurs="0"/>
      <xs:element name="c13" type="xs:string" minOccurs="0"/>
      <xs:element name="c14" type="xs:string" minOccurs="0"/>
      <xs:element name="c15" type="xs:string" minOccurs="0"/>
    </xs:sequence>
  </xs:complexType>
  <!-- date type between 0001 and 9999 restricted to UTC -->
  <xs:simpleType name="dateType">
    <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>
</xs:schema>
```

D.4 Beispiele für die Tabellendaten einer Tabelle

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

D.4a table2.xml (einfache Tabelle)


```xml
<?xml version="1.0" encoding="UTF-8"?>
<table xmlns="http://www.bar.admin.ch/xmlns/siard/2/table.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLoca-
tion="http://www.bar.admin.ch/xmlns/siard/2/table.xsd table2.xsd" version="2.1">
  <row><c1>100</c1><c2>Steven</c2><c3>King</c3><c4>SKING</c4><c5>515.123.4567</c5>
    <c6>2003-06-16Z</c6><c7>AD_PRES</c7><c8>24000</c8><c11>90</c11></row>
  <row><c1>101</c1><c2>Neena</c2><c3>Kochhar</c3><c4>NKOCHHAR</c4><c5>515.123.4568</c5>
    <c6>2005-09-20Z</c6><c7>AD_VP</c7><c8>17000</c8><c10>100</c10><c11>90</c11></row>
  <row><c1>102</c1><c2>Lex</c2><c3>De Haan</c3><c4>LDEHAAN</c4><c5>515.123.4569</c5>
    <c6>2001-01-12Z</c6><c7>AD_VP</c7><c8>17000</c8><c10>100</c10><c11>90</c11></row>
  <row><c1>103</c1><c2>Alexander</c2><c3>Hunold</c3><c4>AHUNOLD</c4><c5>590.423.4567</c5>
    <c6>2006-01-02Z</c6><c7>IT_PROG</c7><c8>9000</c8><c10>102</c10><c11>60</c11></row>
  <row><c1>104</c1><c2>Bruce</c2><c3>Ernst</c3><c4>BERNST</c4><c5>590.423.4568</c5>
    <c6>2007-05-20Z</c6><c7>IT_PROG</c7><c8>6000</c8><c10>103</c10><c11>60</c11></row>
  <row><c1>105</c1><c2>David</c2><c3>Austin</c3><c4>DAUSTIN</c4><c5>590.423.4569</c5>
    <c6>2005-06-24Z</c6><c7>IT_PROG</c7><c8>4800</c8><c10>103</c10><c11>60</c11></row>
  <row><c1>106</c1><c2>Valli</c2><c3>Pataballa</c3><c4>VPATABAL</c4><c5>590.423.4560</c5>
    <c6>2006-02-04Z</c6><c7>IT_PROG</c7><c8>4800</c8><c10>103</c10><c11>60</c11></row>
  <row><c1>107</c1><c2>Diana</c2><c3>Lorentz</c3><c4>DLORENTZ</c4><c5>590.423.5567</c5>
    <c6>2007-02-06Z</c6><c7>IT_PROG</c7><c8>4200</c8><c10>103</c10><c11>60</c11></row>
  <row><c1>108</c1><c2>Nancy</c2><c3>Greenberg</c3><c4>NGREENBE</c4><c5>515.124.4569</c5>
    <c6>2002-08-16Z</c6><c7>FI_MGR</c7><c8>12008</c8><c10>101</c10><c11>100</c11></row>
  ...
  <row><c1>206</c1><c2>William</c2><c3>Gietz</c3><c4>WGIETZ</c4><c5>515.123.8181</c5><c6>2002-06-
06Z</c6><c7>AC_ACCOUNT</c7><c8>8300</c8><c10>205</c10><c11>110</c11></row>
</table>
```


D.4b table7.xml (Tabelle mit internen Large Objects)


```xml
<?xml version="1.0" encoding="UTF-8"?>
<table xmlns="http://www.bar.admin.ch/xmlns/siard/2/table.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xsi:schemaLoca-
tion="http://www.bar.admin.ch/xmlns/siard/2/table.xsd table7.xsd" version="2.1">
  <row>
    <c1>1</c1>
    <c2 file="content/schema1/table7/lob1/record0.xml" length="270" digestType="MD5" digest="BCA4FB6D6898A2F42C624839B431C386"/>
    <c3>Southlake, Texas</c3>
    <c4>1400</c4>
    <c5><u1>2001</u1><u2>8307</u2><u3><u1>-103.00195</u1><u2>36.500374</u2></u3></c5>
  </row>
  <row>
    <c1>2</c1>
    <c2 file="content/schema1/table7/lob1/record1.xml" length="268" digestType="MD5" digest="7E99F05D8C4D7D3909D3F20987A0DE41"/>
    <c3>San Francisco</c3>
    <c4>1500</c4>
    <c5><u1>2001</u1><u2>8307</u2><u3><u1>-124.21014</u1><u2>41.998016</u2></u3></c5>
  </row>
  <row>
    <c1>3</c1>
    <c2 file="content/schema1/table7/lob1/record2.xml" length="235" digestType="MD5" digest="C495BB25A6EDBFE829DDB9B28C027DC3"/>
    <c3>New Jersey</c3>
    <c4>1600</c4>
    <c5><u1>2001</u1><u2>8307</u2><u3><u1>-74.695305</u1><u2>41.35733</u2></u3></c5>
  </row>
  ...
  <row>
    <c1>9</c1><c3>Bombay</c3><c4>2100</c4>
  </row>
</table>
```

D.4c table0.xml (Tabelle mit “udt” und Array)


```xml
<?xml version="1.0" encoding="UTF-8"?>
<table xmlns="http://www.bar.admin.ch/xmlns/siard/2/table.xsd" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:schemaLocation="http://www.bar.admin.ch/xmlns/siard/2/table.xsd table0.xsd" version="2.1">
  <row>
    <c1>232</c1>
    <c2>Donald</c2>
    <c3>Hunter</c3>
    <c4><u1>5122 Sinclair Ln</u1><u2>21206</u2><u3>Baltimore</u3><u4>MD</u4><u5>US</u5></c4>
    <c5><a1>+1 410 123 4795</a1></c5>
    <c6>us</c6>
    <c7>AMERICA</c7>
    <c8>2400</c8>
    <c9>Donald.Hunter@CHACHALACA.EXAMPLE.COM</c9>
    <c10>145</c10>
    <c11><u1>2001</u1><u2>8307</u2><u3><u1>-76.545732</u1><u2>39.322775</u2></u3></c11>
    <c12>1960-01-19Z</c12>
    <c13>married</c13>
    <c14>M</c14>
    <c15>G: 130,000 - 149,999</c15>
  </row>
  <row>
    <c1>233</c1>
    <c2>Graham</c2>
    <c3>Spielberg</c3>
    <c4><u1>680 Bel Air Rd</u1><u2>21014</u2><u3>Bel Air</u3><u4>MD</u4><u5>US</u5></c4>
    <c5><a1>+1 410 123 4800</a1></c5>
    <c6>us</c6>
    <c7>AMERICA</c7>
    <c8>2400</c8>
    <c9>Graham.Spielberg@CHUKAR.EXAMPLE.COM</c9>
    <c10>145</c10>
    <c11><u1>2001</u1><u2>8307</u2><u3><u1>-76.357073</u1><u2>39.523878</u2></u3></c11>
    <c12>1970-01-28Z</c12>
    <c13>single</c13>
    <c14>M</c14>
    <c15>D: 70,000 - 89,999</c15>
  </row>
  ...
  </row>
    <row>
    <c1>235</c1>
    <c2>Edward</c2>
    <c3>Oates</c3>
    <c4><u1>8004 Stansbury Rd</u1><u2>21222</u2><u3>Baltimore</u3><u4>MD</u4><u5>US</u5></c4>
    <c5><a1>+1 410 012 4715</a1><a2>+1 410 083 4715</a2></c5>
    <c6>us</c6>
    <c7>AMERICA</c7>
    <c8>2400</c8>
    <c9>Edward.Oates@OVENBIRD.EXAMPLE.COM</c9>
    <c10>145</c10>
    <c11><u1>2001</u1><u2>8307</u2><u3><u1>-76.500344</u1><u2>39.25618</u2></u3></c11>
    <c12>1955-03-20Z</c12>
    <c13>married</c13>
    <c14>M</c14>
    <c15>E: 90,000 - 109,999</c15>
  </row>
  ...
</table>
```

Anhang E – Änderungen gegenüber Version 1.0

Folgende Änderungen wurden von der Version 1.0 zur Version 2.1 vorgenommen.

Kapitel / ID / Dokument	Anpassung	RFC
passim	Kann-Anforderungen: Es wird klar festgehalten, ob ein Feld zwingend ist oder nicht oder ob es auch leer gelassen werden kann.	2013-23
Passim, Kap 5.3, 5.4, 5.7, 6.2 & 6.3	Der Wechsel von SQL:1999 zu SQL:2008 hat Einfluss auf fast alles und bedingt auch neue Kapitel	2014-110
Titelblatt, Kap 3-6, Anhang D	Alle Beispiele beziehen sich auf das neu beigelegte Beispiel ech-0165_oe.siard	
Zusammenfassung	Aktualisiert und erläutert betreffend der Verwendung der Vorgängerversionen.	
Kap 1 & 2	Kapitelnummerierung anhand der neuen Vorlage	
Kap 2.2, passim	ID Anforderung G anstelle A	
G_3.2-1, G_3.4-3, P_4.2-3, passim, metadata.xsd	Anpassungen im Bereich LargeObjects inkl. Attribut *digest* und *digestType* Neu ist es möglich, BLOBS und LOBS extern, das heisst ausserhalb der SIARD-Datei zu speichern.	2015-29
G_3.2-2	Gelöscht, da es keine Formatdefinition ist sondern eine organisatorische Anforderung	
G_4.1-2	Deflate-Komprimierung erlaubt und empfohlen.	Addendum
P_4.2-4, P_4.2-5	Formaterkennung. Zur einfacheren Erkennung des SIARD-Formats (z.B. durch PRONOM) muss neu ein leerer Ordner header/siardversion/2.1/ existieren, welcher die Version des SIARD-Formats identifiziert.	2015-12
P_4.3-3, passim, metadata.xsd	Wechsel von SQL:1999 zu SQL:2008. Die SQL:2008 Abkürzungen wurden integriert.	2014-110
Kap 5.4, metadata.xsd	Nullable-Element des Attributs eingefügt	
Kap 5.6 & 5.7	Folder wurde im Bereich Spalten und Felder durch lobfolder ersetzt	
T_6.1-2, T_6.1-4	Präzisierung Start der fortlaufenden Nummerierung	
T_6.1-3, T_6.3-2, passim, metadata.xsd	Anpassungen im Bereich „Daten, Zeiten und Zeitstempel“	
Kap 5.13, metadata.xsd	actionTime des Triggers mit INSTEAD OF erweitert	
metadata.xsd	Das metadata.xsd wurde an mehreren Orten angepasst. Gründe sind unter anderem: Angleichung an den SQL:2008 Standard (Bezeichnungen und Type-Elemente), Anpassung der regulären Ausdrücke für die vordefinierten Datentypen, Umsetzung der oben genannten Änderungen, Anpassung an die Spezifikation und die Zusammenfassung von primaryKeyType und candidateKeyType zu uniqueKeyType.	