SIARD-2.1.1 – Spécification de format

Titre	SIARD-2.1.1 – Spécification de format
Type	Norme
Stade	Implémentée
Version	2.1.1
Statut	Approuvé
Validation	2019-05-15
Date de publication	2019-05-15
Remplace	eCH-0165 version 2.1
Dépendances	Aucune
Annexes	metadata.xsd, ech-0165_oe.siard[1]
Langues	Allemand (original), français (traduction), anglais (traduction)
Auteurs	Marcel Büchler, Archives fédérales suisses, 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, Archives fédérales suisses, krystyna.ohnesorge@bar.admin.ch Claire Röthlisberger-Jourdan, CECO, claire.roethlisberger@kost.admin.ch Hartwig Thomas, Enter AG, hartwig.thomas@enterag.ch Andreas Voss †, Archives fédérales suisses

[1] Basé sur le modèle de base de données OE d'Oracle

Collaborateurs	Karin Bredenberg, National Archives of Sweden, karin.bredenberg@riksarkivet.se Hedi Bruggisser, Staatsarchiv Thurgau, hedi.bruggisser@tg.ch Georg Büchler, CECO, 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, Archives de l'État du Valais, alain.dubois@admin.vs.ch Arne-Kristian Groven, National Archives Norway (Riksarkivet), arngro@arkivverket.no Martin Kaiser, CECO, 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
Éditeur/distributeur	Centre de coordination pour l'archivage à long terme de documents électroniques (CECO), c/o Archives fédérales suisses, Archivstrasse 24, 3003 Berne https://kost-ceco.ch/ / info@kost-ceco.ch

Condensé

Le présent document contient la spécification du format de fichier SIARD, version 2.1. SIARD est l'abréviation de *Software-Independent Archival of Relational Databases*. La version 1.0 a été mise au point par les Archives fédérales suisses et continue d'être développée par la communauté. Il s'agit d'une description normative d'un format de fichier servant à la conservation à long terme de bases de données relationnelles.

Le format SIARD repose sur des normes – notamment sur les normes ISO Unicode, XML et SQL:2008, la norme Internet URI et la norme industrielle ZIP. L'utilisation de normes reconnues internationalement a pour but de garantir durablement la conservation et l'accès au modèle très répandu de bases de données relationnelles et de permettre d'échanger facilement les contenus de bases de données, indépendamment des formats Dump propriétaires.

Rapport entre la présente version et les précédentes :

eCH-0165 v1.0 → Remplacée par la version 2.1.1 :
La version 1.0 correspond à la norme eCH actuelle. Son usage est encore possible, mais il est plutôt recommandé d'employer le format SIARD-2.1.

eCH-0165 v2.0 → Supprimée :
La version 2.0 a été supprimée car elle générait des erreurs et des confusions qui pourrait causer des problèmes à long terme. Elle ne doit plus être employée. Il convient de recourir au nouveau format SIARD-2.1, qui est déjà implémenté[2], ou alors à la norme eCH-0165 version 1.0.

SIARD-2.1 → Remplacée par la version 2.1.1 :
La version 2.1 corrigeait les erreurs et confusions de la version 2.0. Elle a été élaborée par le groupe spécialisé Archivage électronique d'eCH, mais ne constitue pas une norme eCH officielle.

SIARD-2.1.1 La présente version 2.1.1 constitue la version la plus récente du format SIARD. Elle est identique à la version 2.1 à l'exception de quelques précisions dans les formulations.

[2] Le format SIARD-2.1 est supporté par SIARD Suite à partir de la mise à jour 2.0 et par l'application KOST-Val à partir de la version 1.8.

Sommaire

1 Introduction 6
1.1 Statut 6
1.2 Champ d'application 6
1.2.1 Destinataires/groupe cible 6
1.2.2 Situation initiale 6
1.2.3 Délimitations 7
2 Structure du document 8
2.1 Structure des chapitres 8
2.2 Exigences concernant l'identifiant 8
2.3 Distinction entre exigences obligatoires et facultatives 9
2.4 Notation des dossiers, fichiers et structures des fichiers 9
3 Exigences générales / principes 10
3.1 Utilisation des normes 10
3.2 Base de données comme document 10
3.3 Jeux de caractères et caractères 10
3.4 Schéma d'URI *file* 11
3.5 Identificateur et identificateur régulier 13
4 Exigences concernant la structure du format 14
4.1 Structure du fichier d'archive SIARD 14
4.2 Structure du fichier d'archive SIARD 14
4.3 Correspondance entre les métadonnées et les données tabulaires 17
5 Exigences concernant les métadonnées 22
5.1 Métadonnées de la base de données 22
5.2 Métadonnées du schéma 24
5.3 Métadonnées du type 25
5.4 Métadonnées de l'attribut 26
5.5 Métadonnées au niveau table 27
5.6 Métadonnées de la colonne 28
5.7 Métadonnées de champ 30
5.8 Métadonnées de la clé primaire 31
5.9 Métadonnées de la clé étrangère 31

5.10 Métadonnées de référence 32
5.11 Métadonnées de la clé candidate 32
5.12 Métadonnées de la *check restriction* 33
5.13 Métadonnées au niveau *trigger* 33
5.14 Métadonnées de la vue 34
5.15 Métadonnées du niveau de la routine 35
5.16 Métadonnées des paramètres 35
5.17 Métadonnées des utilisateurs 36
5.18 Métadonnées du niveau rôle 37
5.19 Métadonnées du niveau des privilèges 37
6 Exigences concernant les données tabulaires 38
6.1 Définition de schéma de la table 38
6.2 Cellules de données *Large Object* 39
6.3 Cellules de données date et *timestamp* 40
6.4 Données tabulaires 40
7 Version et validité de la spécification 42
8 Processus de gestion des modifications 42
9 Exclusion de responsabilité – droits de tiers 42
10 Droits d'auteur 42
Annexe A – Collaboration & vérification 43
Annexe B – Abréviations et glossaire 44
Annexe C – Documentation des normes utilisées 46
Annexe D – Extraits de l'exemple ech-0165_oe.siard 47
Annexe E – Modifications par rapport à la version 1.0 75

1 Introduction

1.1 Statut

Le présent document a été approuvé par le groupe spécialisé Archivage électronique d'eCH et peut dès lors être appliqué.

1.2 Champ d'application

1.2.1 Destinataires/groupe cible

Le présent document est un outil technique qui s'adresse aux spécialistes en informatique travaillant dans le domaine de l'archivage permanent de bases de données relationnelles.

1.2.2 Situation initiale

SIARD est l'acronyme de *Software-Independent Archival of Relational Databases* (en français « archivage de bases de données relationnelles indépendant des logiciels »). Il s'agit d'un format de fichier ouvert destiné à l'archivage permanent de bases de données relationnelles se présentant sous la forme de données textuelles basées sur XML, qui sont regroupées dans un fichier conteneur (archive SIARD)[3].

L'archivage permanent désigne la conservation de durée illimitée des informations stockées dans les fichiers SIARD en préservant le train de bits ainsi que la capacité d'interpréter et de représenter les données de façon intelligible et lisible par une personne.

Lorsque la structure et le contenu d'une base de données relationnelle sont traduits au format SIARD, il est possible d'accéder à tout moment aux données de la base de données ou de les échanger, même lorsque le logiciel d'origine de la base de données n'est plus disponible ou opérationnel. On utilise pour ce faire des normes compatibles avec le format SIARD, jouissant d'une vaste compatibilité au niveau international. Cette capacité d'interprétation à long terme des contenus de la base de données repose pour l'essentiel sur les deux normes XML et SQL:2008.

[3] Il convient d'établir une distinction entre le format d'archivage des bases de données SIARD et l'application SIARD Suite. Cette dernière a été mise au point par les Archives fédérales suisses (AFS) afin de créer des fichiers SIARD, de les modifier et de les importer à nouveau dans les environnements des bases de données.

1.2.3 Délimitations

Il est à noter que le format SIARD représente exclusivement le format d'archivage à long terme pour un type spécial de documents numériques (bases de données relationnelles) et est ainsi conçu de manière totalement indépendante des structures de paquet comme SIP (*Submission Information Package*), AIP (*Archival Information Package*) ou DIP (*Dissemination Information Package*) du modèle OAIS[4].

On part du principe qu'une base de données au format SIARD est archivée comme partie d'un tel paquet d'informations avec d'autres documents (*Large Object Files* transférés, tableau de traduction pour les noms de fichiers externes, documentation de base de données, documentation commerciale pertinente pour la compréhension de la base de données, etc.).

De la même manière qu'un fichier de messagerie ou Word basé sur XML contient une structure de fichier interne avec des métadonnées, des données primaires et différentes données auxiliaires, une base de données relationnelle archivée au format SIARD contient aussi, outre les données tabulaires effectives, ses propres métadonnées décrivant le document en détail, qui sont indépendantes du catalogue de métadonnées qu'une archive inclut dans ses paquets OAIS.

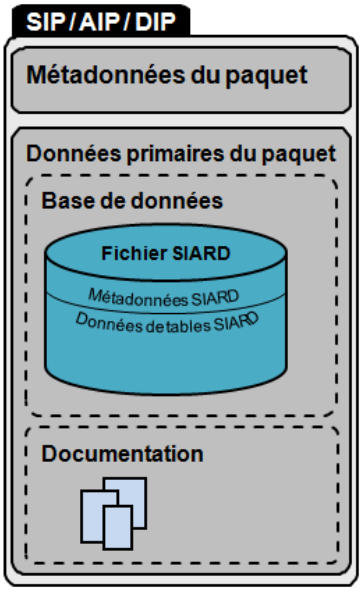


Fig. 1 : schéma d'un paquet d'informations contenant un fichier SIARD

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

2 Structure du document

2.1 Structure des chapitres

Chaque chapitre de la présente spécification est structuré selon un même modèle. Une brève introduction est suivie par un tableau exposant les exigences concernées.

ID	Description de l'exigence	O/F
Contient l'ID de l'exigence	Contient le texte de l'exigence	Précise s'il s'agit d'une exigence obligatoire ou facultative

Une exigence est souvent expliquée au moyen d'exemples, d'indications et de recommandations, lesquels sont présentés de façon spécifique.

ID	Description de l'exigence	O/F
G_2.1-1	Texte de l'exigence **Exemple** Texte de l'exemple **Indication** Texte de l'indication ***Recommandation*** *Le texte d'une recommandation est en italique.*	O

2.2 Exigences concernant l'identifiant

Un identifiant unique permet d'identifier les exigences.

ID
G_2.2-1

La structure de cet identifiant s'appuie sur le modèle suivant :

G_ Lettre + _ identifie le chapitre principal
- G_ = Principes / exigences générales
- T_ = Exigences concernant les données tabulaires
- M_ = Exigences concernant les métadonnées
- P_ = Exigences concernant la structure du paquet

2.2-1 Ce nombre commence par le numéro du chapitre (regroupe les exigences requises concernant le même thème), le chiffre qui suit le trait d'union correspond à une numérotation en continu et désigne ainsi l'ensemble des exigences requises pour le chapitre en question.

2.3 Distinction entre exigences obligatoires et facultatives

Chaque exigence est soit obligatoire, soit facultative. Une lettre sert à préciser cette caractéristique :

Abréviation	Signification
O	Exigence obligatoire Cette exigence doit impérativement être remplie pour obtenir un fichier SIARD valide.
F	Exigence facultative Cette exigence devrait être remplie. Elle simplifie l'utilisation dans l'esprit d'une bonne pratique.

2.4 Notation des dossiers, fichiers et structures des fichiers

Les symboles et paramètres suivants sont utilisés pour la notation des dossiers, des fichiers, etc.

Symbole	Signification
`/`	Dossier
`header/`	Dossier portant le nom *header*
`xy.txt`	Fichier (avec extension de fichier *.txt*)
`dir1/`	Dossier exemple (en rouge)
`abc.pdf`	Fichier exemple (en rouge)
…	Marque d'emplacement pour les fichiers ou dossiers qui ne sont pas pertinents pour l'explication
`[]`	Marque d'emplacement pour une expression ou un type de base, comme *string*, *integer*, etc.
`<xx>`	Marque d'emplacement pour chaîne de caractères quelconque

3 Exigences générales / principes

3.1 Utilisation des normes

Afin de garantir la capacité d'interprétation des contenus de la base de données sur de longues périodes, le format SIARD repose pour l'essentiel sur les deux normes ISO XML et SQL:2008.

ID	Description de l'exigence	O/F
G_3.1-1	Tous les contenus de la base de données sont enregistrés dans une collection de fichiers au format XML 1.0[5] conformes aux définitions selon XML Schema 1.0[6]. Les définitions de schéma et le code SQL doivent être conformes à la version SQL:2008 de la norme ISO/IEC 9075. La seule exception concerne les données BLOB et CLOB (*Binary Large OBjects* et *Character Large OBjects*), qui sont enregistrées dans des fichiers binaires ou textes séparés, mais référencées dans les fichiers XML.	O

3.2 Base de données comme document

Afin de conserver les relations (références) entre les données des différentes tables, une base de données relationnelle est traitée comme un seul document à archiver.

ID	Description de l'exigence	O/F
G_3.2-1	Une base de données relationnelle est archivée dans un seul fichier SIARD. Il est toutefois possible de référencer des *Large Objects* enregistrés à un autre emplacement, mais qui font partie de la base de données au sens large.	O

3.3 Jeux de caractères et caractères

ID	Description de l'exigence	O/F
G_3.3-1	Toutes les données sont enregistrées dans le jeu de caractères Unicode selon ISO 10646.	O
G_3.3-2	Lors de l'extraction à partir de bases de données qui sont compatibles avec d'autres jeux de caractères, les données sont affichées dans les jeux de caractères Unicode correspondants. C'est la raison pour laquelle les types de chaîne de caractères nationaux (NCHAR, NCHAR VARYING, NCLOB) extraits de la base de données doivent en règle générale être traduits en types non nationaux (CHAR, VARCHAR ou CLOB). Cette convention est compatible avec XML, que le fichier XML soit enregistré au format UTF-8 ou UTF-16.	O

[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>Description de l’exigence</th><th>O/F</th></tr>
<tr><td>G_3.3-3</td><td>Dans les fichiers XML du format SIARD, tous les caractères ayant une signification particulière dans la syntaxe XML sont remplacés par des entités caractère et ce, dans tous les champs de type xs:string.
En outre, les caractères de contrôle Unicode 0-31 et 127-159 sont codés à l’aide de la barre oblique inversée (« \ ») afin de garantir la validité du fichier XML.</td><td>O</td></tr>
<tr><td>G_3.3-4</td><td>Les caractères qui ne peuvent être représentés en Unicode (codes 0-8, 14-31 et 127-f159) ainsi que le caractère d’échappement « \ » et une suite de plusieurs espaces successives sont représentés dans XML par la combinaison \u00<xx>. Les guillemets ainsi que les caractères « < », « > » et « & » sont représentés comme entités de caractères dans XML.
<table>
<tr><th>Caractères originaux</th><th>Caractères du format SIARD</th></tr>
<tr><td>de 0 à 8</td><td>de \u0000 à \u0008</td></tr>
<tr><td>14-31</td><td>de \u000E à \u001F</td></tr>
<tr><td>32</td><td>\u0020, si plusieurs l’un après l’autre</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>de 127 à 159</td><td>de \u007F à \u009F</td></tr>
</table></td><td>O</td></tr>
</table>

3.4 Schéma d’URI *file*

Le schéma d’URI *file* est utilisé pour référencer les *Large Objects* transférés[7], conformément à la spécification RFC 1738.

ID	Description de l’exigence	O/F
G_3.4-1	Tous les fichiers transférés sont spécifiés par un URI *file* selon la spécification RFC 1738.	O
G_3.4-2	Les URI *file* sont enregistrés dans un fichier SIARD en tant que ASCII-Strings codés en URL.	O
G_3.4-3	Pour autant que l’URI *file* soit basé sur un système de fichiers qui permet un renvoi direct vers des fichiers individuels dans un fichier ZIP, les *Large Objects* transférés peuvent être recompressés dans des fichiers de ce format. Par exemple, l’adresse *file:///d:/sips/sip1234.zip* redirige vers le fichier ZIP, tandis que l’adresse *file:///d:/sips/sip1234.zip/* renvoie au répertoire racine dans le fichier ZIP.	F

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

3.5 Identificateur et identificateur régulier

Dans SQL:2008, il existe des identificateurs réguliers[8] sans espaces ni caractères spéciaux, pour lesquels la casse n'est pas importante mais qui sont enregistrés en majuscules dans le fichier SIARD, et des identificateurs entre guillemets[9], pour lesquels la casse importe et qui peuvent également contenir des caractères spéciaux ou être identiques à un mot clé SQL. Ceux-ci doivent être encadrés par des expressions entre guillemets doubles. Dans le fichier SIARD, ils sont enregistrés sans les guillemets.

C'est la norme SQL qui détermine ce qu'est un caractère spécial ou un mot clé. La norme Unicode définit quant à elle ce qu'est la version majuscule d'une lettre.

Dans les métadonnées, un identificateur régulier est enregistré en majuscules, tandis que tous les autres identificateurs sont enregistrés à l'identique, sans guillemets. Selon la norme SQL:2008, un identificateur doit être considéré comme un identificateur délimité s'il contient un caractère qu'un identificateur régulier ne peut pas contenir, ou s'il est identique à un mot clé SQL.

ID	Description de l'exigence	O/F
G_3.5-1	Tous les identificateurs sont enregistrés dans le jeu de caractères Unicode.	O
G_3.5-2	Les identificateurs réguliers sont en majuscules et sans guillemets.	O
G_3.5-3	Les identificateurs délimités sont enregistrés sans guillemets.	O

[8] « Identificateur régulier », *identifier* en anglais. Un identificateur SQL:2008 doit commencer par une lettre (A-Z) ou le tiret bas (_), suivi de lettres (A-Z), de chiffres (0-9) ou de tiret bas (_), et ne pas comporter plus de 128 caractères.

[9] « Identificateur entre guillemets » ou « identificateur (dé)limité », *delimited identifier* en anglais

4 Exigences concernant la structure du format

4.1 Structure du fichier d'archive SIARD

Le fichier d'archive SIARD est structuré comme une archive ZIP.

ID	Description de l'exigence	O/F
G_4.1-1	Le fichier SIARD est enregistré comme une archive ZIP unique non compressée, conformément à la spécification publiée par la société PkWare, version 6.3.2[10].	O
G_4.1-2	Les fichiers SIARD doivent être soit non compressés, soit compressés au moyen d'un algorithme *Deflate* selon la spécification RFC 1951[11]. ***Recommandation*** *Il est recommandé de recourir à l'algorithme Deflate.*	O
G_4.1-3	Le fichier SIARD n'est ni protégé par un mot de passe, ni chiffré.	O
G_4.1-4	Les deux dénominations ZIP32 et ZIP64 sont autorisées pour l'archive ZIP.	O
G_4.1-5	L'archive ZIP a l'extension de fichier "`.siard`".	O

4.2 Structure du fichier d'archive SIARD

Une base de données relationnelle archivée au format SIARD comprend deux composantes : les métadonnées, qui décrivent la structure de la base de données archivée, et les données tabulaires, qui représentent les contenus des tables. Les métadonnées indiquent en outre où trouver quelles données tabulaires dans l'archive.

ID	Description de l'exigence	O/F

P_4.2-1	Les données tabulaires se trouvent dans le dossier `content/` et les métadonnées dans le dossier `header/`. Les autres dossiers ou fichiers ne sont pas autorisés.
	
	Exemple
	Structure du fichier SIARD (schématique)
	`ech-0165_oe.siard`
	`  content/`
	`  header/`
		O

[10] Définis à l'origine par Phil Katz, les fichiers ZIP se sont imposés comme norme *de facto* et sont aujourd'hui très répandus. La version actuelle 6.3.2 de la spécification publiée par la société PkWare est disponible sur le site https://support.pkware.com/display/PKZIP/Application+Note+Archives.

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

ID	Description de l'exigence	O/F

P_4.2-2	Le dossier `content/` contient un ou plusieurs dossiers de schéma dans lesquels se trouvent les différents dossiers de table. Les autres dossiers ou fichiers ne sont pas autorisés.
	
	Exemple
	Structure du fichier SIARD (schématique)
	`ech-0165_oe.siard`
	`content/`
	`schema0/`
	`table0/`
	`table1/`
	`table2/`
	`...`
	`schema1/`
	`table0/`
	`...`
	
	Recommandation
	Il est recommandé d'uniformiser les noms des dossiers des schémas et des tables et de les utiliser à la place du nom effectif, p. ex `schema0/` *ou* `table0/` *(voir les restrictions au point P_4.2-6).*
		O

P_4.2-3	Chaque dossier de table contient un fichier XML et un fichier XSD dont les noms (nom du dossier et des deux fichiers respectifs) doivent être identiques. Les autres dossiers ou fichiers ne sont pas autorisés, à l'exception des dossiers BLOB et CLOB et de leur contenu (fichiers BIN, TXT, XML ou, si le type MIME des fichiers LOB est connu, une extension associée à celui-ci, p. ex. JPG).
	
	Exemple
	Structure du fichier SIARD (schématique)
	`ech-0165_oe.siard`
	`content/`
	`...`
	`schema1/`
	`...`
	`table6/`
	`table6.xml`
	`table6.xsd`
	`table7/`
	`table7.xml`
	`table7.xsd`
	`lob1`[12]`/`
	`record0.xml`
	`record1.xml`
	`...`
		O

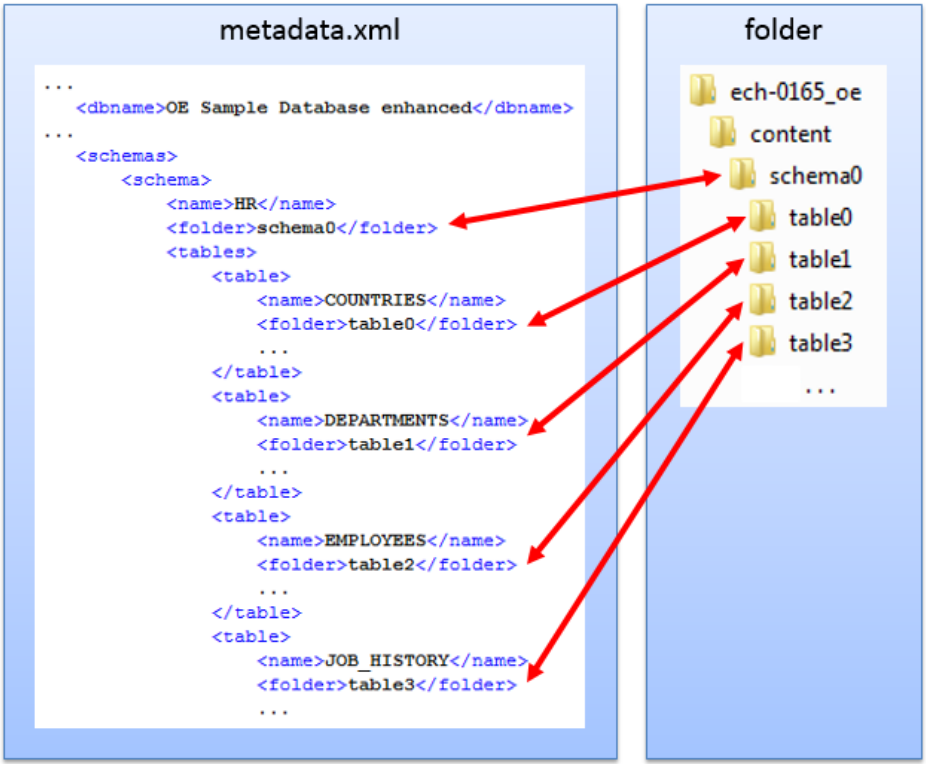
[12] Dans cet exemple, la colonne 2 contient des fichiers LOB supplémentaires, lesquels sont enregistrés dans `lob1/`.

ID	Description de l'exigence	O/F
	Recommandation *Il est recommandé d'uniformiser les noms des dossiers et des fichiers LOB et de les utiliser à la place des noms effectifs, par exemple* `lob1/` *et* `record0.bin`, `record0.txt` *ou* `record0.xml` *ou, si le type MIME des fichiers LOB est connu, d'utiliser une extension associée à celui-ci (voir les restrictions au point P_4.2-6).*	
P_4.2-4	Pour faciliter l'identification du format SIARD (p. ex. par PRONOM), un dossier vide `header/siardversion/2.1/` doit exister qui indique la version du format SIARD.	O

P_4.2-5	Les fichiers `metadata.xml` et `metadata.xsd` doivent se trouver dans le dossier `header/`. D'autres fichiers, par exemple des feuilles de style, sont autorisés.
	
	Exemple
	Structure du fichier SIARD (schématique)
	`ech-0165_oe.siard`
	`  content/`
	`...`
	`  header/`
	`  metadata.xml`
	`  metadata.xsd`
	`  siardversion/`
	`  2.1/`
	`...`
		O

P_4.2-6	Tous les noms de fichier et de dossier doivent présenter la structure suivante : Le nom doit commencer par une lettre [a-z ou A-Z] et ne peut ensuite contenir que les caractères suivants : • a-z • A-Z • 0-9 • _ • . (autorisé uniquement pour séparer le nom et l'extension) ***Recommandation*** *Dans la mesure du possible, les noms de fichier et de dossier ne devraient pas dépasser 20 caractères afin d'éviter les difficultés liées à des longueurs de chemin trop importantes sous Windows.*	O

4.3 Correspondance entre les métadonnées et les données tabulaires

ID	Description de l'exigence	O/F
P_4.3-1	La structure figurant dans `metadata.xml` doit être identique à celle du dossier `content/`. **Exemple**  	O
P_4.3-2	Le nombre de colonnes de chaque table figurant dans `metadata.xml` doit être identique à celui du fichier `table[nombre].xsd` correspondant. **Exemple** 	O

<table>
<tr><th>ID</th><th>Description de l’exigence</th><th>O/F</th></tr>
<tr><td>P_4.3-3</td><td>Dans les définitions de colonnes de <code>metadata.xml</code>, les informations sur le type de données doivent être identiques à celles du fichier <code>table[nombre].xsd</code> correspondant.

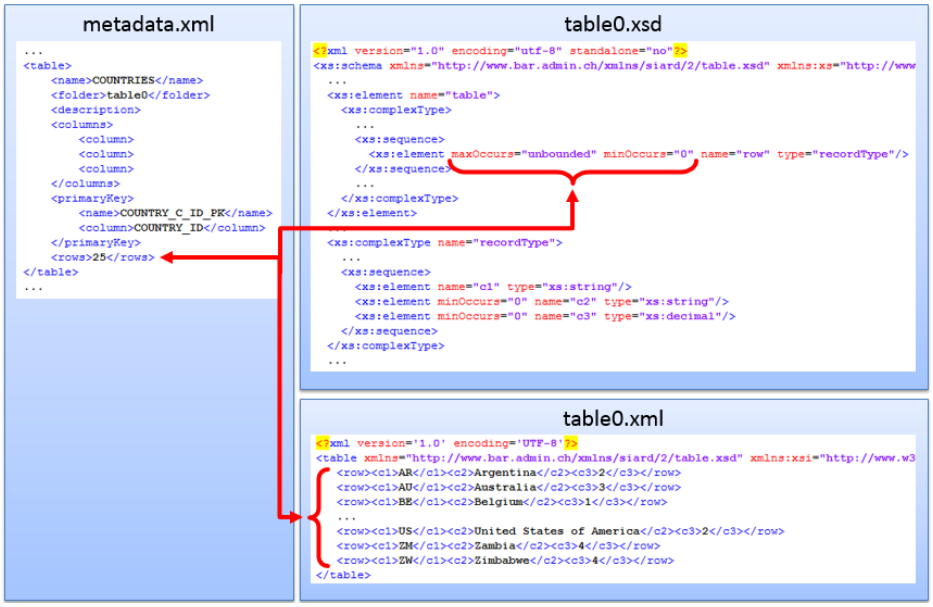
Dans les fichiers de schéma <code>table[nombre].xsd</code>, les types de données SQL:2008[13] prédéfinis sont convertis en types de données XML selon le tableau suivant :
<table>
<tr><th>SQL:2008</th><th>XML</th></tr>
<tr><td>BIGINT</td><td>xs:integer</td></tr>
<tr><td>BINARY LARGE OBJECT(...), BLOB(...)</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>O</td></tr>
</table>

[13] Les types BIT et BIT VARYING correspondent à d’anciennes définitions SQL qui ont été remplacées par BOOLEAN et BINARY dans SQL:2008. BIT(1) est converti en BOOLEAN et BIT(n) en BINARY((n+7)/8).

[14] Pour les types de données XML *blobType* et *clobType*, voir G_3.1-1

ID	Description de l'exigence	O/F
	Exemple metadata.xml table0.xsd Tous les types DATE et TIME sont spécifiés dans le fuseau horaire UTC. Il est recommandé de les faire terminer par « Z ». dateType est une restriction de xs:date en UTC aux années comprises entre 0001 et 9999 (restriction SQL:2008). timeType est une restriction de xs:time au fuseau horaire UTC. dateTimeType est une restriction de xs:dateTime dans le fuseau horaire UTC aux années comprises entre 0001 et 9999 (restriction SQL:2008). clobType est une extension de xs:string. Les valeurs *inline* ne nécessitent pas d'attribut supplémentaire et la valeur CLOB est enregistrée directement. Si les valeurs CLOB sont stockées dans un fichier séparé, il faut impérativement définir les attributs suivants : *file* et *length*. En plus de ces attributs obligatoires, il est aussi possible de définir les attributs facultatifs suivants : *digestType* et *digest*. La valeur de l'attribut *file* contient l'URI du fichier (codé en URL et si possible rattaché au *lobFolder* suivant) dans lequel est enregistré le CLOB. La valeur de l'attribut *length* correspond à la longueur en UTF-8, et l'attribut facultatif *digest* contient l'information liée à l'intégrité selon l'attribut facultatif *digestType*. blobType est une extension de xs:hexBinary. Les valeurs *inline* ne nécessitent pas d'attribut supplémentaire et la valeur BLOB est enregistrée directement. Si les valeurs BLOB sont stockées dans un fichier séparé, il faut impérativement définir les attributs suivants : *file* et *length*. En plus de ces attributs obligatoires, il est aussi possible de définir les attributs facultatifs suivants : *digestType* et *digest*. La valeur de l'attribut *file* contient l'URI du fichier (codé en URL et si possible rattraché au *lobFolder* suivant) dans lequel est enregistré le BLOB. La valeur de l'attribut *length* correspond à la longueur en UTF-8, et l'attribut facultatif *digest* contient l'information liée à l'intégrité selon l'attribut facultatif *digestType*.	
P_4.3-4	Dans les fichiers de schéma `table[nombre].xsd`, les types de données *distinct* sont convertis dans le type de données XML qui a été employé pour représenter leurs types de base.	O
P_4.3-5	Dans les fichiers de schéma `table[nombre].xsd`, les *arrays* sont convertis en une séquence d'éléments XML structurés `<a1>`, `<a2>`, etc., qui sont à leur tour convertis dans les types de données XML correspondant au type de base de l'*array*. **Exemple** Voir l'exemple `table0.xsd` dans l'annexe D.3c	O

ID	Description de l'exigence	O/F
P_4.3-6	Dans les fichiers de schéma `table[nombre].xsd`, les *User-defined Data Type (UDT)* sont convertis en une séquence d'éléments XML structurés `<u1>`, `<u2>`, etc., qui sont à leur tour convertis dans les types de données XML correspondant au type de chaque attribut. **Exemple** Voir l'exemple `table0.xsd` dans l'annexe D.3c	O
P_4.3-7	Dans les définitions de colonnes de `metadata.xml`, les informations *nullable* doivent être identiques à celles du fichier `table[nombre].xsd` correspondant. **Exemple** metadata.xml table0.xsd **Indication** La notation SQL:2008 <nullable>true</nullable> devient minOccurs="0" dans XML. <nullable>false</nullable> correspond à minOccurs="1" dans XML ; celle-ci étant toutefois la valeur par défaut, elle est bien souvent omise. En l'absence d'indication <nullable>, il faut comprendre <nullable>true</nullable>.	O
P_4.3-8	Dans `metadata.xml`, l'ordre des colonnes doit être identique à celui du fichier `table[nombre].xsd` correspondant.	O
P_4.3-9	Dans la définition de table de `metadata.xml`, l'ordre des champs doit être identique à celui du fichier `table[nombre].xsd` correspondant.	O

ID	Description de l'exigence	O/F
P_4.3-10	Dans `metadata.xml`, le nombre de lignes d'une table doit être identique au nombre de lignes dans le fichier `table[nombre].xml` correspondant. Dans `metadata.xml`, le nombre de lignes d'une table doit pouvoir entrer dans la zone spécifiée par le fichier `table[nombre].xsd` correspondant. **Exemple**  ***Recommandation*** *Il est recommandé de définir une plage de 0 à l'infini (maxOccurs="unbounded" minOccurs="0") dans le fichier* `table[nombre].xsd`*. Ceci permet d'éviter les problèmes lors de la validation du fichier* `table[nombre].xml` *par rapport au fichier* `table[nombre].xsd`*.*	O

5 Exigences concernant les métadonnées

Les métadonnées dans l'archive SIARD enregistrent la structure de la base de données archivée et indiquent où trouver quelles données tabulaires dans l'archive.

L'ensemble des métadonnées sont collectées dans un seul fichier nommé `metadata.xml` à l'intérieur du dossier `header/`. Ce fichier dispose d'une structure hiérarchique.

Il existe pour le fichier `metadata.xml` la définition de schéma `metadata.xsd`, qui est également stockée dans le dossier `header/`.

ID	Description de l'exigence	O/F
M_5.0-1	La définition de schéma `metadata.xsd` doit impérativement être respectée pour le fichier `metadata.xml`. En d'autres termes, `metadata.xml` doit pouvoir être validé avec le schéma `metadata.xsd`.	O

Les contenus des différents niveaux sont définis dans la suite du présent document.

5.1 Métadonnées de la base de données

Le fichier `metadata.xml` contient les informations générales suivantes au niveau de la base de données :

ID	Description de l'exigence	O/F
M_5.1-1	Toutes les métadonnées qui sont désignées comme obligatoires dans `metadata.xsd` au niveau de la base de données doivent être renseignées en conséquence.	O

Les métadonnées de la base de données suivantes sont enregistrées dans le fichier `metadata.xml` :

Désignation	Signification	O/F
version	Format de la version SIARD	O
dbname	Désignation concise de la base de données	O
description	Description de la signification et du contenu de la base de données dans son ensemble	F
archiver	Nom de la personne qui a effectué l'archivage des données de tables issues de la base de données	F
archiverContact	Coordonnées (téléphone, e-mail) de la personne qui a effectué l'archivage des données tabulaires issues de la base de données	F

Désignation	Signification	O/F
dataOwner	Propriétaire des données figurant dans la base de données ; institution ou personne qui, au moment de l'archivage, détient le droit d'octroyer des droits d'utilisation des données et qui est responsable du respect des obligations légales, par exemple des directives relatives à la protection des données	O
dataOriginTimespan	Période de création des données dans la base de données ; indication approximative de la période sous forme de texte	O
lobFolder	URI *file* qui sert d'URI de base pour les URI relatifs qui indiquent l'emplacement de sauvegarde externe possible des *Large Objects*. À défaut d'une telle métadonnée, le *Default* du dossier d'origine se trouve dans le fichier ZIP. Les URI relatifs *lobFolder* dans les métadonnées de colonne sont relatifs à cette valeur. **Indication** Lorsque l'URI *file* renvoie vers un *Extended File System* dans lequel les fichiers ZIP sont traités comme dossier, l'URI relatif « *..* » renvoie au dossier externe dans lequel se trouve le fichier SIARD. Faute de compatibilité avec ce système de fichier, il faut utiliser un URI *file* absolu afin d'indiquer un emplacement de sauvegarde externe pour les fichiers LOB. Il est explicitement recommandé de représenter toutes les entrées *lobFolder* des colonnes et tous les attributs de fichier LOB comme URI relatif. Ainsi, en cas de déplacement du fichier SIARD ou de son paquet d'informations, seul cet URI global doit être modifié afin de renvoyer vers le nouvel emplacement de sauvegarde.	F
producerApplication	Nom et version de l'application qui a téléchargé le fichier SIARD	F
archivalDate	Date d'archivage ; date de l'archivage des données tabulaires	O
messageDigest	Se compose de l'attribut `digestType` (MD5, SHA-1 ou SHA-256) et de l'attribut `digest` associé. L'attribut `digest` représente un tampon binaire sous forme de chaîne de caractères hexadécimale ou – pour SHA-1 ou SHA-256 – en base 64. La longueur du *digest* binaire et de la chaîne de caractères permet de déterminer si le code utilisé était hexadécimal ou en base 64. Le *digest* est calculé via le dossier `content/`. Plusieurs codes *messageDigest* peuvent être enregistrés sur la base de différents algorithmes[15]. **Exemple** Voir l'exemple `metadata.xml` dans l'annexe D.2 ***Recommandation*** *Si l'option* messageDigest *est utilisée, le point suivant s'applique :*	F

[15] Un code *messageDigest* enregistré dans le fichier SIARD ne constitue pas en soi une garantie d'intégrité, car si le fichier SIARD est falsifié, il peut l'être aussi. Seul le stockage externe d'un code *messageDigest* permet de contourner ce problème. Le cas échéant, il peut néanmoins être utile de générer un code interne *messageDigest* indépendant des métadonnées lors du téléchargement.

Désignation	Signification	O/F
	les dossiers content *et* header *sont enregistrés comme entrées séparées (vides)* `content/` *et* `header/` *dans le fichier ZIP. Afin de pouvoir contrôler l'intégrité des données primaires, il est nécessaire que l'entrée du répertoire* header *ne soit insérée dans l'entrée* `content/` *qu'après toutes les données primaires et avant toutes les autres entrées de métadonnées. Le* messageDigest *mentionné ci-dessous est calculé de l'offset 0 jusqu'à l'offset de l'entrée* `header/` *du fichier SIARD.*	
clientMachine	Nom DNS de l'ordinateur (client) sur lequel l'archivage a été effectué	F
databaseProduct	Produit utilisé comme base de données et version dans laquelle l'archivage des données tabulaires a été effectué	F
connection	*Connection String* utilisé pour l'archivage des données tabulaires	F
databaseUser	User Id dans la base de données de l'utilisateur de l'outil SIARD pour l'archivage des données tabulaires provenant de la base de données	F
schemas	Liste des schémas dans la base de données	O
users	Liste des utilisateurs de la base de données	O
roles	Liste des rôles de la base de données	F
privileges	Liste des privilèges pour les utilisateurs et les rôles	F

5.2 Métadonnées du schéma

Les métadonnées de schéma sont archivées dans le fichier `metadata.xml`, comme les informations générales sur la base de données.

ID	Description de l'exigence	O/F
M_5.2-1	Toutes les métadonnées qui sont désignées comme obligatoires dans `metadata.xsd` au niveau du schéma doivent être renseignées en conséquence.	O

Les métadonnées de schéma suivantes sont enregistrées dans le fichier `metadata.xml` :

Désignation	Signification	O/F
name	Nom de schéma dans la base de données	O
folder	Nom du dossier schéma sous `content/` dans l'archive SIARD	O
types	Liste des types avancés ou structurés dans le schéma	F
description	Description de la signification et du contenu du schéma	F
tables	Liste des tables dans le schéma	F
vues	Listes des vues stockées dans le schéma	F
routines	Liste des routines (anciennement *Stored Procedures*) dans le schéma	F

5.3 Métadonnées du type

ID	Description de l'exigence	O/F
M_5.3-1	Les métadonnées de type d'un schéma peuvent être archivées dans le fichier `metadata.xml`.	F

Les métadonnées de type suivantes sont enregistrées dans `metadata.xml` lorsqu'un type de données avancé ou structuré est archivé :

Désignation	Signification	O/F
name	Nom du type de données dans le schéma	O
category	Catégorie du type de données avancé ou structuré (*distinct* ou *udt*)	O
underSchema	Nom de schéma du supertype, si le type de données est basé sur un supertype	F
underType	Nom du supertype si le type de données est basé sur un supertype	F
instantiable	Vrai, si le type de données peut être instancé, sinon faux	O
final	Vrai, si aucun subtype ne peut être créé pour ce type de données, sinon faux.	O
base	Nom du type de base (SQL, prédéfini) si la catégorie est *distinct*	F
attributes	Liste des attributs si la catégorie est *udt*	F
description	Description de la signification et du contenu du type de données	F

5.4 Métadonnées de l'attribut

ID	Description de l'exigence	O/F
M_5.4-1	Toutes les métadonnées employées dans le type de données *udt* désignées comme obligatoires au niveau de l'attribut dans `metadata.xsd` doivent être renseignées en conséquence.	O

Les métadonnées d'attributs suivantes sont enregistrées dans le fichier `metadata.xml` :

Désignation	Signification	O/F
name	Nom de l'attribut	O
type	Type de données de l'attribut selon SQL:2008 (prédéfini)	F
typeOriginal	Type de colonne original pour le type de données standard **Indication** Étant donné que les différents programmes de base de données considérés comme conformes à SQL permettent des types de données très divers, le type *original* figure également ici, tout comme le type SQL:2008. Pour chaque programme de base de données supportant le format SIARD, il faut définir et documenter une traduction des types propriétaires en types SQL:2008 pour l'application correspondante.	F
nullable	Élément *nullable* de l'attribut ***Recommandation*** *Il est recommandé de ne pas recourir à l'élément* nullable.	F
typeSchema	Schéma du type de données avancé ou structuré	F
typeName	Nom du type de données avancé ou structuré	F
defaultValue	Valeur standard de l'attribut	F
description	Description de la signification et fonction de la routine	F
cardinality	Nombre (maximal) d'éléments si l'attribut est un *array*	F

5.5 Métadonnées au niveau table

Les métadonnées au niveau de la table sont archivées dans le fichier `metadata.xml`, comme les informations générales sur la base de données et les métadonnées de schéma.

ID	Description de l'exigence	O/F
M_5.5-1	Toutes les métadonnées qui sont désignées comme obligatoires dans `metadata.xsd` au niveau de la table doivent être renseignées en conséquence.	O

Les métadonnées du niveau de la table sont enregistrées dans le fichier `metadata.xml`, tout comme les indications globales de la base de données ainsi que les métadonnées du niveau schéma :

Désignation	Signification	O/F
name	Nom de la table dans le schéma	O
folder	Nom du dossier de la table dans le dossier du schéma	O
description	Description de la signification et du contenu de la table	F
columns	Liste des colonnes de la table	O
primaryKey	Clé primaire de la table	F
foreignKeys	Liste de la clé étrangère de la table	F
candidateKeys	Liste de la clé candidate de la table	F
checkConstraints	Liste des restrictions de la table	F
triggers	Liste des *triggers* de la table	F
rows	Nombre de jeux de données	O

5.6 Métadonnées de la colonne

Les métadonnées au niveau de la colonne sont archivées dans le fichier `metadata.xml`, comme les informations générales concernant la base de données, les métadonnées de schéma et les métadonnées de la table. Les métadonnées de colonne décrivent une colonne dans une table ou une vue.

ID	Description de l'exigence	O/F
M_5.6-1	Toutes les métadonnées qui sont désignées comme obligatoires dans `metadata.xsd` au niveau de la colonne doivent être renseignées.	O

Les métadonnées de colonne suivantes sont enregistrées dans le fichier `metadata.xml` :

Désignation	Signification	O/F
name	Nom de colonne dans la table ou la vue Le nom de colonne doit être unique à l'intérieur d'une même table.	O
lobFolder	Nom du dossier LOB comme URI *file* relatif ou absolu, le cas échéant, dans le système de fichier externe. L'élément peut être utilisé pour l'enregistrement tant interne qu'externe de *Large Objects*. **Exemple** Voir l'exemple `metadata.xml` dans l'annexe D.2 **Indication** Cette entrée n'a d'importance que si la colonne est une colonne LOB (p. ex. de type BLOB, CLOB ou XML). Si rien n'est indiqué, la valeur « . » fait office de valeur par défaut, par exemple pour renvoyer au même dossier que *lobFolder* au niveau de la base de données. Sinon, sa valeur doit être un URI *file* (si possible relatif) qui désigne le dossier dans lequel les fichiers de cette colonne LOB doivent être enregistrés. Si cette valeur est un URI relatif, il est admis que celui-ci est relatif par rapport à l'ensemble de l'entrée *lobFolder* au niveau de la base de données. Les attributs file relatifs des cellules de cette colonne sont interprétés comme relatifs à ce dossier.	F
type	Type de colonne SQL:2008 prédéfini **Indication** Si le type de données de cette colonne est un type de données prédéfini, ce champ est obligatoire. Dans le cas contraire, le champ *typeName* doit renvoyer vers un type défini dans la liste des types.	O

Désignation	Signification	O/F
typeOriginal	Type de colonne original **Indication** Étant donné que les différents programmes de base de données considérés comme conformes à SQL permettent des types de données très divers, le type *original* figure également ici, outre le type SQL:2008. Pour chaque programme de base de données supportant le format SIARD, il faut définir et documenter une traduction des types propriétaires en types SQL:2008 pour l'application correspondante.	F
nullable	Entrée non requise	F
typeSchema	Schéma du type désigné lorsque la colonne n'est pas un type de données prédéfini et que le type de données désigné n'est pas défini dans le même schéma que la table de cette colonne.	F
typeName	Nom du type de données avancé ou structuré de cette colonne	F
fields	Liste des champs dans la colonne, si la colonne est un *array* ou un type de données structuré de la catégorie *udt*	F
defaultValue	Valeur standard de la colonne	F
mimeType	Type MIME de cette colonne s'il s'agit d'une colonne BLOB et si toutes ses entrées contiennent des fichiers du même type MIME. Cet élément purement informatif aide à choisir le bon *viewer* pour les objets binaires. Il peut être soit renseigné manuellement, soit par le programme de téléchargement en utilisant un outil de détection du format.	F
description	Description de la signification et du contenu de la colonne	F
cardinality	Nombre (maximal) d'éléments si la colonne est un *array*	F

5.7 Métadonnées de champ

ID	Description de l'exigence	O/F
M_5.7-1	Les métadonnées de champ d'une colonne ou d'un champ peuvent être archivées dans `metadata.xml`.	F

Les métadonnées de champ suivantes sont enregistrées dans `metadata.xml` si une colonne ou un champ est un *array* ou un type de données avancé ou structuré de la catégorie *udt :*

Désignation	Signification	O/F
name	Nom du champ dans la colonne ou le champ Le nom du champ doit être unique à l'intérieur d'une même colonne. ***Recommandation*** *Pour les conteneurs (colonne ou champ) de type* udt*, le nom du champ devrait être identique au nom de l'attribut correspondant. Pour les conteneurs* array*, le nom du champ devrait être le nom du conteneur suivi de l'indice* array *commençant par 1 entre parenthèses carrées. P. ex. Point[1], Point[2], etc.*	O
lobFolder	Nom du dossier LOB comme URI *file* relatif ou absolu, le cas échéant dans le système de fichier externe. L'élément peut être utilisé pour l'enregistrement tant interne qu'externe de *Large Objects*. **Indication** Cette entrée n'a d'importance que si le champ est un champ LOB (p. ex. de type BLOB, CLOB ou XML). Si rien n'est indiqué, la valeur « . » fait office de valeur par défaut, par exemple pour renvoyer au même dossier que *lobFolder* au niveau de la base de données. Sinon, sa valeur doit être un URI *file* (si possible relatif) qui désigne le dossier dans lequel les fichiers de ce champ LOB doivent être enregistrés. Si cette valeur est un URI relatif, il est admis que celui-ci est relatif par rapport à l'ensemble de l'entrée *lobFolder* au niveau de la base de données. Les attributs *file* relatifs des cellules de cette colonne sont interprétés comme relatifs à ce dossier.	F
fields	Liste des champs si celui-ci est un *array* ou un type de données structuré de la catégorie *udt*	F
mimeType	Type MIME de ce champ s'il s'agit d'un champ BLOB et si toutes ses entrées contiennent des fichiers du même type MIME. Cet élément purement informatif aide à choisir le bon *viewer* pour les objets binaires. Il peut être soit renseigné manuellement, soit par le programme de téléchargement en utilisant un outil de détection du format.	F
description	Description de la signification et du contenu du champ	F

5.8 Métadonnées de la clé primaire

Par « clé primaire », on entend une clé unique qui permet d'identifier un jeu de données.

ID	Description de l'exigence	O/F
M_5.8-1	Les métadonnées de la clé primaire d'une table peuvent être archivées dans le fichier `metadata.xml`.	F

Les métadonnées de la clé primaire suivantes sont enregistrées dans le fichier `metadata.xml`, dans la mesure où une clé primaire est archivée :

Désignation	Signification	O/F
name	Nom de la clé primaire	O
column	Liste des colonnes de la clé primaire	O
description	Description de la signification et du contenu de la clé primaire	F

5.9 Métadonnées de la clé étrangère

ID	Description de l'exigence	O/F
M_5.9-1	Les métadonnées de la clé étrangère à l'intérieur d'une table peuvent être archivées dans le fichier `metadata.xml`.	F

Les métadonnées de clé étrangère sont enregistrées dans le fichier `metadata.xml`, dans la mesure où une clé étrangère est archivée :

Désignation	Signification	O/F
name	Nom de la clé étrangère	O
referenced-Schema	Schéma de la table référencée	O
referencedTable	Table référencée **Indication** Le nom de la table externe référencée peut être de type table ou schéma.table. Dans ce cas, les identificateurs délimités sont placés entre guillemets.	O
reference	Référence (liste des colonnes et colonnes référencées)	O
matchType	Type de *match* (*FULL*, *PARTIAL* ou *SIMPLE*)	F

Désignation	Signification	O/F
deleteAction	Action de suppression, par exemple : *CASCADE* **Indication** Les actions de suppression et de modification incluent les actions autorisées par la norme SQL:2008.	F
updateAction	Action de modification, par exemple : *SET DEFAULT*	F
description	Description de la signification et du contenu de la clé étrangère	F

5.10 Métadonnées de référence

ID	Description de l'exigence	O/F
M_5.10-1	Les métadonnées de référence qui sont utilisées pour la clé étrangère peuvent être archivées dans le fichier `metadata.xml`	F

Les métadonnées de référence suivantes sont enregistrées dans le fichier `metadata.xml`, dans la mesure où une clé étrangère est archivée :

Désignation	Signification	O/F
column	Nom de la colonne	O
referenced	Nom de la colonne référencée	O

5.11 Métadonnées de la clé candidate

Par « clé candidate », on entend les clés uniques qui peuvent être choisies comme clé primaire. Dans `metadata.xsd`, les clés primaires et les clés candidates sont toutes du même type, à savoir *uniqueKeyType*. C'est pourquoi les exigences envers les clés candidates sont les mêmes que pour les clés primaires (M_5.8-1).

ID	Description de l'exigence	O/F
M_5.11-1	Les métadonnées de la clé candidate d'une table peuvent être archivées dans le fichier `metadata.xml`	F

Les métadonnées de clé candidate suivantes sont enregistrées dans le fichier `metadata.xml`, dans la mesure où une clé candidate est archivée :

Désignation	Signification	O/F
name	Nom de la clé candidate	O
column	Liste des colonnes de la clé candidate	O
description	Description de la signification et du contenu de la clé candidate	F

5.12 Métadonnées de la *check restriction*

La *check restriction* correspond à une condition à vérifier. Celle-ci est indiquée comme expression de type BOOLEAN (avec la valeur *true*, *false* ou *unknown*) dans la syntaxe SQL:2008.

ID	Description de l'exigence	O/F
M_5.12-1	Les métadonnées de la *check restriction* d'une table peuvent être archivées dans le fichier `metadata.xml`.	F

Les métadonnées de *check restriction* suivantes sont enregistrées dans le fichier `metadata.xml`, dans la mesure où une *check restriction* est archivée :

Désignation	Signification	O/F
name	Nom de la *check restriction*	O
condition	Condition de la *check restriction*	O
description	Description de la signification et du contenu de la *check restriction*	F

5.13 Métadonnées au niveau *trigger*

ID	Description de l'exigence	O/F
M_5.13-1	Les métadonnées du *trigger* d'une table peuvent être archivées dans le fichier `metadata.xml`	F

Les métadonnées de *trigger* suivantes sont enregistrées dans le fichier `metadata.xml`, dans la mesure où un *trigger* est archivé :

Désignation	Signification	O/F
name	Nom du *trigger* dans la table	O
actionTime	BEFORE, AFTER ou INSTEAD OF	O
triggerEvent	INSERT, DELETE, UPDATE [OF <trigger column list>]	O
aliasList	<old or new value alias list>	F
triggeredAction	<triggered action>	O
description	Description de la signification et du contenu du *trigger*	F

5.14 Métadonnées de la vue

ID	Description de l'exigence	O/F
M_5.14-1	Les métadonnées de la vue d'un schéma peuvent être archivées dans le fichier `metadata.xml`.	F

Les métadonnées de la vue suivantes sont enregistrées dans le fichier `metadata.xml`, dans la mesure où une vue est archivée:

Désignation	Signification	O/F
name	Nom de la vue dans le schéma	O
columns	Liste du nom de la colonne de la vue **Indication** Les métadonnées des colonnes d'une vue ont une structure identique à celle d'une table.	O
query	Requête SQL:2008 qui définit la vue	F
queryOriginal	Requête SQL originale qui définit la vue **Indication** Étant donné que les différents programmes de base de données considérés comme conformes à SQL permettent une syntaxe de requête très diverse, la requête originale figure également ici, en plus de la requête SQL:2008. Pour chaque programme de base de données supportant le format SIARD, il faut définir et documenter une traduction des syntaxes de requête propriétaires en types SQL:2008 pour l'application correspondante.	F
rows	Nombre de jeux de données	F
description	Description de la signification et du contenu de la vue	F

5.15 Métadonnées du niveau de la routine

ID	Description de l'exigence	O/F
M_5.15-1	Les métadonnées de la routine d'un schéma peuvent être archivées dans le fichier `metadata.xml`.	F

Les métadonnées de routine suivantes sont archivées dans le fichier `metadata.xml`, dans la mesure où une routine est archivée :

Désignation	Signification	O/F
specificName	Nom spécifique identifiant précisément la routine dans le schéma[16]	O
name	Nom de la routine dans le schéma	O
Description	Description de la signification et du contenu de la routine	F
source	Code source original de la routine (VBA, PL/SQL, JAVA) **Indication** Étant donné que de nombreux programmes de base de données disposent de routines propriétaires qui ne peuvent être transformées en une requête conforme à SQL:2008, le code source original de la routine (p. ex. dans PL/SQL pour les bases de données Oracle, VBA pour les modules MS Access) peut être archivé ici.	F
body	Code source de la routine conforme à SQL:2008	F
characteristic	Description de la routine	F
returnType	Type de valeur de retour de la routine (dans la mesure où il s'agit d'une fonction)	F
parameters	Liste des paramètres	F

5.16 Métadonnées des paramètres

ID	Description de l'exigence	O/F
M_5.16-1	Les métadonnées des paramètres qui sont utilisés pour la routine peuvent être archivées dans le fichier `metadata.xml`	F

[16] L'introduction d'éléments orientés sur les objets dans SQL:1999 autorise l'*overloading*, qui permet à deux routines différentes (procédures ou fonctions) d'avoir le même nom pour autant qu'elles aient une liste de paramètres différents. Dès lors, l'exigence selon laquelle le nom d'une routine doit être unique dans le schéma devient caduque. C'est pourquoi l'attribut *specificName* a été introduit afin d'identifier précisément la routine dans le schéma.

Les métadonnées de paramètres suivantes sont enregistrées dans le fichier `metadata.xml`, dans la mesure où une routine est archivée :

Désignation	Signification	O/F
name	Nom du paramètre	O
mode	Mode du paramètre (IN, OUT ou INOUT)	O
type	Type SQL:2008 prédéfini du paramètre **Indication** Si le type de données de cette colonne est prédéfini, ce champ doit être utilisé. Dans le cas contraire, il faut utiliser un des types de la liste du champ *typeName*.	O
typeOriginal	Type de paramètre original **Indication** Étant donné que les différents programmes de base de données considérés comme conformes à SQL permettent des types de données très divers, le type de colonne original figure également ici, tout comme le type de données SQL:2008. Pour chaque programme de base de données supportant le format SIARD, il faut définir et documenter une traduction des types propriétaires en types SQL:2008 pour l'application correspondante.	F
typeSchema	Schéma du type désigné si le paramètre n'est pas un type de données prédéfini et si le type de données n'est pas défini dans le même schéma que la table de cette colonne.	F
typeName	Nom du type de données avancé ou structuré de ce paramètre	F
description	Description de la signification et de la fonction de la routine	F
cardinality	Nombre (maximal) d'éléments si le paramètre est un *array*	I

5.17 Métadonnées des utilisateurs

sID	Description de l'exigence	O/F
M_5.17-1	Les métadonnées des utilisateurs peuvent être archivées dans le fichier `metadata.xml`.	F

Les métadonnées d'utilisateur suivantes sont enregistrées dans le fichier `metadata.xml` :

Désignation	Signification	O/F
name	Nom de l'utilisateur	O
description	Description de la signification et de la fonction de l'utilisateur	F

5.18 Métadonnées du niveau rôle

ID	Description de l'exigence	O/F
M_5.18-1	Les métadonnées de rôle peuvent être archivées dans le fichier `metadata.xml`.	F

Les métadonnées de rôle suivantes sont archivées dans le fichier `metadata.xml` :

Désignation	Signification	O/F
name	Nom du rôle	O
admin	Administrateur du rôle (utilisateur ou rôle)	O
description	Description de la signification et de la fonction du rôle	F

5.19 Métadonnées du niveau des privilèges

ID	Description de l'exigence	O/F
M_5.19-1	Les métadonnées des privilèges peuvent être archivées dans le fichier `metadata.xml`.	K

Les métadonnées de privilèges suivantes sont archivées dans le fichier `metadata.xml` :

Désignation	Signification	O/F
type	Privilège accordé (ex. SELECT)	O
object	Objet sur lequel le privilège doit être appliqué	F
grantor	Personne qui accorde le privilège	O
grantee	Bénéficiaire du privilège (utilisateur ou rôle)	O
option	Grant-Option (ADMIN ou GRANT)	F
description	Description de la signification et de la fonction du privilège	F

6 Exigences concernant les données tabulaires

Comme expliqué précédemment, les données tabulaires d'une base de données relationnelle archivée se trouvent dans le dossier `content/` dans la racine document de l'archive SIARD. Elles y sont stockées dans les dossiers schéma et tables correspondants.

Les données tabulaires sont enregistrées dans des fichiers XML correspondants. Pour chaque table, une définition de schéma XML, qui indique le format d'enregistrement XML des données tabulaires, est générée. Pour chaque table, il existe par conséquent un fi- chier `table[nombre].xml` correspondant à la définition de schéma `table[nombre].xsd`.

ID	Description de l'exigence	O/F
T_6.0-1	L'intégralité des données tabulaires (données primaires) doit satisfaire aux exigences de cohérence de SQL:2008. Ainsi, un fichier SIARD qui valide la syntaxe par rapport aux XSD mais dont la sémantique enfreint la norme SQL n'est pas conforme à la présente description de format. Les valeurs de la table en particulier doivent correspondre aux restrictions des types SQL dans les métadonnées. En outre, les conditions des clés primaires, candidates et étrangère enregistrées dans les métadonnées, ainsi que les conditions entraînant la nullité, doivent toutes être remplies.	O
T_6.0-2	La définition de schéma `table[nombre].xsd` doit impérativement être respectée pour le fichier `table[nombre].xml`. En d'autres termes, `table[nombre].xml` doit pouvoir être validé avec le schéma `table[nombre].xsd`.	O

6.1 Définition de schéma de la table

Le fichier `table[nombre].xsd` contient les définitions de schéma suivantes de la table :

ID	Description de l'exigence	O/F
T_6.1-1	Pour chaque table, il doit exister une définition de schéma XML, qui indique le format d'enregistrement XML des données tabulaires.	O
T_6.1-2	Cette définition de schéma reflète les métadonnées schéma SQL de la table et indique que la table sera enregistré comme séquence de lignes, qui contient une séquence d'entrées de colonnes avec différents types XML. Le nom des *tags* de table est *table*, celui des *tags* de jeu de données est *row*, tandis que les *tags* de colonnes portent le nom *c1, c2*… Les *tags* de colonne commencent toujours par c1 et progressent systématiquement de 1 en 1. En d'autres termes, il ne peut y avoir aucun « trou », car une valeur NULL se traduit par la suppression de la colonne correspondante dans le fichier XML associé. **Exemple** Voir l'exemple `table2.xsd` dans l'annexe D.3a	O

ID	Description de l'exigence	O/F
T_6.1-3	Le *mapping* des types qui devrait être utilisé dans les définitions de schéma de la table est spécifié dans P_4.3-3. Outre les types standards de schéma XML, on utilise les types spéciaux suivants : clobType, blobType, dateType, timeType, dateTimeType	O
T_6.1-4	Les valeurs multiples de types avancés ou structurés doivent être enregistrées comme éléments séparés à l'intérieur du *tag* de cellule. Les noms des éléments individuels d'un *array* sont a1, a2, etc. Les noms des éléments individuels d'un *udt* sont u1, u2, etc. Les noms commencent toujours respectivement par a1 et u1, et progressent systématiquement de 1 en 1. En d'autres termes, il ne peut y avoir aucun « trou », car une valeur NULL se traduit par la suppression de la colonne correspondante dans le fichier XML associé. **Exemple** Voir l'exemple `table0.xsd` dans l'annexe D.3c	O

6.2 Cellules de données *Large Object*

ID	Description de l'exigence	O/F
T_6.2-1	Les *Large Objects* peuvent être enregistrés *inline* dans le fichier `table[nombre].xml`, ou alors comme entrées séparées dans le fichier SIARD (internes) ou encore comme fichiers autonomes indépendants du fichier SIARD (externes) dans le système de fichiers.	O

Lorsque les *Large Objects* sont enregistrés comme entrées ou fichiers séparés (internes ou externes), il faut impérativement enregistrer les attributs *file* et *length* dans une cellule LOB du fichier `table[nombre].xsd`. Si les *Large Objects* sont enregistrés *inline*, ces attributs sont facultatifs.

Désignation	Signification	O/F
file	Si le *Large Object* est stocké séparément, cet élément désigne l'emplacement où il est enregistré et le nom du fichier *Large Object* dans cette cellule ou cet attribut de cellule en tant qu'URI *file*. S'il s'agit d'un URI relatif, celui-ci est interprété relativement au *lobFolder* (de la colonne ou de l'attribut) de l'élément auquel il se rapporte.	O[17]
length	Longueur (en octets pour les BLOB, en caractères pour les CLOB et XML)	O[17]

[17] Facultatif si le *Large Object* est enregistré *inline* dans le fichier `table[nombre].xml`

Désignation	Signification	O/F
digestType	Contient le type d'information liée à l'intégrité (*digest*) : « MD5 », « SHA-1 » ou « SHA-256 ». ***Recommandation*** *Il est recommandé de définir cet attribut (en combinaison avec l'attribut* digest*) pour tous les LOB enregistrés dans des fichiers séparés.*	F
digest	Information liée à l'intégrité du CLOB ou du BLOB ***Recommandation*** *Il est recommandé de définir cet attribut (en combinaison avec l'attribut* digestType*) pour tous les LOB enregistrés dans des fichiers séparés.*	F

6.3 Cellules de données date et *timestamp*

ID	Description de l'exigence	O/F
T_6.3-1	Les dates et les *timestamps* doivent être limités aux années 0001-9999, selon la spécification SQL:2008. Cette restriction est forcée dans les définitions de *dateType* et *dateTimeType*.	O
T_6.3-2	Les dates, heures et horodatages doivent être enregistrés en temps universel coordonné (UTC) et éventuellement se terminer par « Z ». Cette restriction est obligatoire dans les définitions de *dateType*, *timeType* et *dateTimeType*. ***Recommandation*** *Il est recommandé de terminer l'ensemble des dates, heures et horodatages par « Z ».*	O

6.4 Données tabulaires

Le fichier `table[nombre].xml` contient les données tabulaires de la table :

ID	Description de l'exigence	O/F
T_6.4-1	Pour chaque table, les données tabulaires doivent être enregistrées dans un fichier XML.	O
T_6.4-2	Le fichier *table* se compose d'éléments *row* qui contiennent les données d'une ligne réparties entre les différentes colonnes (*c1, c2* ...). **Exemple** Voir l'exemple `table2.xml` dans l'annexe D.4a	O
T_6.4-3	Lorsque la cellule d'une colonne ou d'un champ a une valeur NULL, elle doit être omise. Si elle est égale à „“ (un *string* de longueur 0), elle doit être présente mais vide.	O

ID	Description de l'exigence	O/F
T_6.4-4	Lorsque la cellule d'une colonne contient une valeur complexe (*array*, *udt*), elle est représentée par une séquence de sous-éléments de la cellule (a1, a2, … pour les *array*, u1, u2, … pour les *udt*), lesquels contiennent les valeurs correspondantes. Ces valeurs peuvent être complexes à leur tour. **Exemple** Voir l'exemple `table0.xml` dans l'annexe D.4c	F
T_6.4-5	Lorsqu'une table contient des données de type *Large Object* (BLOB, CLOB, ...), des fichiers séparés peuvent être créés à cette fin et stockés à la place du contenu des cellules de l'emplacement de sauvegarde du fichier. La décision d'enregistrer des *Large Objects* dans des fichiers séparés plutôt qu'*inline* appartient au logiciel qui crée le fichier SIARD. Pour éviter des dossiers vides, il ne faut créer des dossiers que si cela s'avère nécessaire (à savoir, s'ils contiennent des données). Si les *Large Objects* sont enregistrés dans un fichier séparé, il faut impérativement définir les attributs suivants : *file* et *length*. En plus de ces attributs obligatoires, il est aussi possible de définir les attributs facultatifs suivants : *digestType* et *digest*. La valeur de l'attribut *file* contient l'URI du fichier (codé en URL et si possible lié au prochain *lobFolder*) dans lequel est enregistré le *Large Object*. La valeur de l'attribut *length* correspond à la longueur (en octets pour les BLOB et en caractères pour les CLOB et XML), et l'attribut facultatif *digest* contient l'information liée à l'intégrité selon l'attribut facultatif *digestType*. **Exemple** Voir l'exemple `table7.xml` dans l'annexe D.4b ***Recommandation*** *Il est explicitement recommandé d'enregistrer soit tous, soit aucun* Large Object *dans une colonne* inline. *Il est recommandé d'uniformiser les noms des dossiers et des fichiers lob et de les utiliser à la place du nom effectif, par exemple* `lob4/` *et* `record0.bin` *ou* `record0.txt`.	O

7 Version et validité de la spécification

La présente version de la spécification est la version 2.1.

8 Processus de gestion des modifications

Le processus de gestion des modifications de la présente norme se fonde sur [eCH-0150], scénario 3.

9 Exclusion de responsabilité – droits de tiers

Le Centre de coordination pour l'archivage à long terme de documents électroniques (CECO) exclut toute responsabilité quant aux décisions ou mesures prises par un utilisateur sur la base du présent document. Il ne peut fournir aucune assurance ou garantie quant à l'absence d'erreur, l'actualité, l'exhaustivité et l'exactitude des documents et informations mis à disposition. Toute responsabilité relative à des dommages que l'utilisateur pourrait subir par suite de l'utilisation de la présente norme est exclue, sous réserve de prescriptions légales contraires.

10 Droits d'auteur

La propriété intellectuelle de la présente norme revient entièrement à ses auteurs. Ceux-ci s'engagent à la mettre à disposition gratuitement et à autoriser son utilisation et son développement.

Annexe A – Collaboration & vérification

Karin Bredenberg, National Archives of Sweden

Hedi Bruggisser, Staatsarchiv Thurgau

Georg Büchler, CECO

Marcel Büchler, Archives fédérales suisses

Janet Delve, University of Portsmouth

Boris Domajnko, Slovenian National Archives

Alain Dubois, Archives de l'Etat du Valais

Luis Faria, KEEP SOLUTIONS, LDA

Bruno Ferreira, KEEP SOLUTIONS, LDA

Arne-Kristian Groven, National Archives Norway (Riksarkivet)

Martin Kaiser, CECO

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, Archives fédérales suisses

Lauri Rätsep, National Archives of Estonia

Claire Röthlisberger-Jourdan, CECO

Hélder Silva, KEEP SOLUTIONS, LDA

Mario Spuler, Fachlabor Gubler

Hartwig Thomas, Enter AG

Andreas Voss †, Archives fédérales suisses

Annexe B – Abréviations et glossaire

Terme	Description
AIP	*Archival Information Package* : selon la norme OAIS, les AIP sont extraits des SIP au cours du processus d'archivage des documents numériques. L'AIP représente la forme de paquets d'informations sous laquelle les documents numériques sont archivés dans le magasin numérique.
Archive	1. Institution/service qui évalue, conserve et rend accessible les fonds d'archive. 2. Documents d'une organisation qui ont été archivés. 3. Bâtiment ou institution construit ou établi pour l'archivage des documents. 4. Terme pour un fichier contenant les autres fichiers. Cf. également fichier d'archive et comme synonyme fichier conteneur.
Archivage permanent/ archivage à long terme	Conservation, pour une durée illimitée, et maintien de la disponibilité permanente des informations numériques. Outre la préservation du train de bits des informations archivées, il s'agit également de la capacité à les interpréter et les représenter à tout moment de manière lisible et intelligible par les personnes.
Base de données	Une « base de données » se compose normalement d'un ou de plusieurs schémas de base de données ainsi que des droits d'accès définis pour les différents utilisateurs et rôles à certaines parties de la base de données. Dans SQL:2008, les utilisateurs (*Users*) et les rôles (*Roles*) peuvent être les bénéficiaires d'autorisations (*privileges*). Une base de données relationnelle se compose ainsi d'une quantité d'objets de base de données structurés (p. ex. schéma, vue, etc.) ainsi que des contenus des tables. Un schéma de base de données est une sorte de préfixe de *namespace*. Un catalogue de base de données contient les métadonnées de tous les schémas du catalogue. Le catalogue des niveaux dans SQL:2008 correspond au « Document base de données », qui peut être converti en un format d'archive avec SIARD.
DIP	*Dissemination Information Package* : selon la norme OAIS, conteneur des dossiers qui ont été commandés par un utilisateur dans le cadre d'une procédure de commande.
DNS	*Domain Name System* : base de données distribuée qui gère les *noms de domaine* sur Internet.
Documentation	Ensemble des informations, tous supports confondus, qui ont été reçues ou établies lors de l'accomplissement de tâches publiques, ainsi qu'ensemble des outils et données complémentaires nécessaires à la compréhension de ces informations et à leur utilisation.
Documents d'archive	Documents qui ont été repris par les archives à des fins de conservation ou ont été archivés de façon indépendante par d'autres services selon les mêmes principes.
Données primaires	Données qui constituent le contenu du document. Dans un fichier SIARD, les données tabulaires ont la fonction de données primaires.
Dossier	Ensemble (collectif) des documents relatifs à une affaire. Un dossier correspond par principe à une affaire. En regroupant des affaires similaires ou en divisant les dossiers en sous-dossiers, cette structure de base peut toutefois être adaptée aux besoins individuels. Le dossier est créé sur la base du système de classement.
LOB	*Large Object* : terme générique désignant le contenu d'une cellule d'une colonne CLOB, BLOB ou XML, qui peut être représenté par un fichier séparé.
MD5	Message-Digest Algorithm 5, une fonction de hachage
Métadonnées	Informations concernant les données primaires (données sur les données), qui se distinguent par leur caractère descriptif.

Terme	Description
OAIS	*Open Archival Information System*, ISO 14721:2003. Ce modèle de référence décrit une archive comme une organisation dans laquelle les personnes et les systèmes œuvrent de concert pour conserver les informations et les mettre à la disposition d'un groupe défini d'utilisateurs.
Paquet d'informations	Conteneur conceptuel composé de renseignements facultatifs sur les contenus et des métadonnées de conservation qui en font partie de façon facultative. Contient aussi des informations de mise en paquet, qui délimitent et identifient les informations de contenu et la description du paquet et permettent la recherche d'information de contenu.
Producteur de documents	Service ou unité d'organisation qui a produit et géré les documents.
Routines	Les routines SQL (également nommées *Stored Procedures*) sont importantes avant tout pour la compréhension des requêtes *view*, dans lesquelles elles peuvent apparaître sous forme d'expressions partielles.
Schémas	Conteneurs de tables, de vues et de routines.
SHA1	Algorithme de hachage sûr (*Secure Hash Algorithm*)
SIP	*Submission Information Package* : selon la norme OAIS, paquets d'informations transmis aux archives par les services constituant les dossiers. Ils contiennent les documents numériques (données primaires et métadonnées).
Table	Élement composé d'une définition de table comportant des champs qui attribuent à chaque colonne un nom et un type, de jeux de données qui contiennent les données tabulaires effectives, d'une clé primaire facultative, de clés étrangères, qui garantissent l'intégrité référentielle, de clés candidates, qui servent à l'identification d'un jeu de données, et de restrictions, qui garantissent la cohérence. Une table peut également comporter des *trigger* (déclencheur).
UTF	Unicode *Transformation Format*
Vue	Requête standard enregistrée dans la base de données. Le résultat des requêtes est une table qui contient également des champs et des jeux de données.
XSD	Définition de schéma XML

Annexe C – Documentation des normes utilisées

eCH-0150	eCH-0150 *Change et Release Management* des normes eCH http://www.ech.ch/
RFC 1738	Spécification des *Uniform Resource Locators* (URL), notamment de l'URL/URI *file* https://www.ietf.org/rfc/rfc1738.txt
RFC 1951	Spécification de l'algorithme Deflate https://www.ietf.org/rfc/rfc1951.txt
SQL:2008	Norme ISO/IEC 9075(1-4,9-11,13,14):2008 : Technologies de l'information – Langages de base de données – SQL https://www.iso.org/fr/standard/53681.html
Unicode	Unicode 6.1.0 Unicode, Inc. http://www.unicode.org/versions/Unicode6.1.0/ (correspond à ISO/IEC 10646:2012 : Technologies de l'information – Jeu universel de caractères codés (JUC) https://www.iso.org/fr/standard/56921.html)
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

Annexe D – Extraits de l'exemple ech-0165_oe.siard

Le fichier SIARD ech-0165_oe.siard constitue une annexe au document SIARD 2.1.
Les données de l'annexe D sont tirées de ce fichier.

D.1 metadata.xsd

La définition de schéma `metadata.xsd` définit la structure du fichier `metadata.xml` dans le dossier `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:documentation>
    </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 Exemple de metadata.xml

Une description de métadonnées pour le schéma XML conforme au format SIARD d'une base de données se présente par exemple comme suit :


```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>
```

```xml
<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 Exemple de définition de schéma XML d'une table

Pour chaque table, SIARD crée une définition de schéma XML, qui affecte correctement les types de données XML aux colonnes.

D.3a table2.xsd (définition de schéma d'une table simple)

```
<?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 (définition de schéma d'une table avec *Large Objects* internes)

```
<?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 (définition de schéma d'une table avec *udt* et *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"
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"/>
      <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 Exemples de données tabulaires d'une table

Les données tabulaires sont enregistrées dans un fichier XML qui satisfait à la définition de schéma XML de la table.

D.4a table2.xml (table simple)


```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 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 (table avec *Large Objects* internes)


```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 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 (table avec *udt* et *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>
```

Annexe E – Modifications par rapport à la version 1.0

Les modifications suivantes ont été effectuées entre la version 1.0 et la version 2.1[18] :

Chapitre / ID / document	Adaptation	RFC
Passim	Exigences facultatives : il est clairement spécifié si un champ est obligatoire ou non, et s'il peut aussi être laissé vide.	2013-23
Passim, chap. 5.3, 5.4, 5.7, 6.2 & 6.3	Le passage de SQL:1999 à SQL:2008 se répercute quasiment partout et engendre de nouveaux chapitres	2014-110
Page de titre, chap. 3 à 6, annexe D	Tous les exemples se réfèrent au nouvel exemple ech-0165_oe.siard.	
Condensé	Actualisé et enrichi d'explications concernant le recours aux versions précédentes	
Chap. 1 & 2	Numérotation des chapitres selon le nouveau modèle	
Chap. 2.2, passim	Exigence ID G au lieu de A	
G_3.2-1, G_3.4-3, P_4.2-3, passim, metadata.xsd	Adaptations dans le domaine des *Large Objects*, y c. attributs *digest* et *digestType*. Il est désormais possible d'enregistrer des BLOB et des LOB dans un emplacement externe, à savoir en dehors du fichier SIARD.	2015-29
G_3.2-2	Supprimé, car il ne s'agit pas d'une définition de format mais d'une exigence liée à l'organisation	
G_4.1-2	Compression Deflate autorisée et recommandée	Addendum
P_4.2-4, P_4.2-5	Identification du format. Pour faciliter l'identification du format SIARD (p. ex. par PRONOM), le fichier doit contenir un dossier vide header/siardversion/2.1/ qui indique la version du format SIARD.	2015-12
P_4.3-3, passim, metadata.xsd	Passage de SQL:1999 à SQL:2008. Les abréviations SQL:2008 ont été intégrées.	2014-110
Chap. 5.4, metadata.xsd	Élément *nullable* de l'attribut ajouté	
Chap. 5.6 & 5.7	*folder* remplacé par *lobFolder* dans les sections relatives aux colonnes et aux champs	
T_6.1-2, T_6.1-4	Nombre de départ de la numérotation indiqué	
T_6.1-3, T_6.3-2, passim, metadata.xsd	Adaptations dans les sections relatives aux dates, aux heures et à l'horodatage	
Chap. 5.13, metadata.xsd	INSTEAD OF ajouté dans la signification de l'attribut *actionTime* du *trigger*	
metadata.xsd	Le fichier metadata.xsd a été adapté en plusieurs endroits, notamment pour les motifs suivants : harmonisation avec la norme SQL:2008 (descriptions et éléments *type*), adaptation des expressions régulières pour les types de données prédéfinis, mise en œuvre des modifications décrites ci-dessus, adaptation à la spécification et au regroupement de *primaryKeyType* et de *candidateKeyType* en *uniqueKeyType*.	

[18] Les modifications entre les versions 2.1 et 2.1.1 se limitent à des précisions de formulation dans le condensé ainsi que dans G_3.3-4.