## СОДЕРЖАНИЕ

Сессия 1 данного Конкурсного задания состоит из следующей документации / файлов:

1. WSR2017\_BigData\_С1.pdf (Инструкция к первой сессии)
2. obv.xml\_204.gz (Архив с исходными данными)
3. OBV\_full.xsd (Схема данных для исходного набора)

## ВВЕДЕНИЕ

На этой сессии вы выполните подготовку репозитория хранения данных.

Вам будет предложено определить набор итоговых полей и создать базу данных для данного набора.

В файл: obv.xml\_204.gz предоставлены исходные данные для выполнения данной сессии.

## ИНСТРУКЦИЯ участнику

К концу этой сессии, у вас должны быть достигнуты следующие результаты, необходимые для того, чтобы заказчик был спокоен, что работа будут завершена вовремя.

### ПРАКТИЧЕСКИЕ РЕЗУЛЬТАТЫ

|  |
| --- |
| 1.1 Выделение итогового набора полей для одной записи |
| Архив obv.xml\_204.gz содержит в себе xml файл с исходным набором записей. Каждая запись представлена набором полей. Так же имеется xsd файл, описывающий предоставленный xml файл. Необходимо проанализировать перечисленные поля, их тип, содержание и определить итоговый набор полей для реляционной или нереляционной структуры данных, в которую преобразованные записи будут импортированы |

1. Открываем xsd файл с помощью блокнота или любой другой программы для чтения текста.

**Простой пример xsd файла**

<xs:schema targetNamespace="http://www.w3schools.com" elementFormDefault="qualified">

<xs:element name="соискатель">

<xs:complexType>

<xs:sequence>

<xs:element name="ФИО" type="xs:string"/>

<xs:element name="телефон" type="xs:string"/>

</xs:sequence>

</xs:complexType>

</xs:element>

В этом примере первая строка является заголовком xsd файлов. Затем объявлено, что будет объяснен элемент “соискатель”. В третьей строке выясняется, что элемент “соискатель” имеет сложный тип. В последующих строках объяснено то, что элемент “соискатель” является списком ФИО и телефонов

Тэг простого элемента может содержать атрибут “по умолчанию”.

<xs:element name="телефон" type="xs:string" default="неизвестен"/>

или фиксированное значение

<xs:element name="город проживания" type="xs:string" fixed="город Якутск"/>

В 1.1 необходимо понять xsd = схему xml и дать объяснение.

Основные трудности:

Строковые данные могут быть заданы паттернами.

Пример задаяет строку из трех букв.

<xs:element name="initials">

<xs:simpleType>

<xs:restriction base="xs:string">

<xs:pattern value="[a-zA-Z][a-zA-Z][a-zA-Z]"/>

</xs:restriction>

</xs:simpleType>

</xs:element>

Список может содержат список.

|  |
| --- |
| 1.2 Разбиение поля с множеством значений на несколько полей |
| В исходных записях имеются поля, конкатенирующие в себе несколько значений, либо указывающие диапазон. Необходимо выделить такие поля и разбить на несколько других. |

Список может содержат список.

|  |
| --- |
| 1.3 Описание схемы данных для итоговой модели данных |
| Для итогового набора полей необходимо описать xsd файл, сделать это можно по примеру исходного xsd файла. |

Ответ будет иметь такой же вид как простой пример выше.

|  |
| --- |
| 1.4 Создать базу данных для итогового набора полей |
| Необходимо создать реляционную базу данных или любую другую структуру данных, которая позволит сохранить записи, состоящие из итогового набора полей. Так же необходимо предусмотреть разделение на тестовую и обучающую выборки. |

Пример решения

**import** **xml.etree.ElementTree** **as** **ET**

tree = ET.parse('country\_data.xml')

root = tree.getroot()

print(root.tag)

print(root.attrib)

**for** child **in** root:

print(child.tag, child.attrib)

dataFrame = []

columns = []

**for** i, child **in** enumerate(root):

record = []

**for** subchild **in** child:

record.append(subchild.text)

**if** subchild.tag **not** **in** columns:

columns.append(subchild.tag)

dataFrame.append(record)

Ключевые слова парсить (parse), dataframe, “import xml.etree.ElementTree as ET”