

RIGA TECHNICAL UNIVERSITY

FACULTY OF COMPUTER SCIENCE AND INFORMATION TECHNOLOGY

INSTITUTE OF APPLIED COMPUTER SYSTEMS

“Technology of Large database”

Practical task #4

**Design of XML Database**

Author: Batuhan Arslanoglu

Studentcardno.: 151ADB096

2017 / 2018 study year

Contents

[1. Task description 3](#_Toc505963836)

[2. Design and create XML documents and schema files 4](#_Toc505963837)

[2.1 XML files 4](#_Toc505963838)

[a) Customer.xml 4](#_Toc505963839)

[b) Hotel.xml 5](#_Toc505963840)

[2.2 XML Schemas 6](#_Toc505963841)

[a) Customer.xsd 6](#_Toc505963842)

[b) Hotel.xsd 8](#_Toc505963843)

[2. Creating oracle directory to store files 9](#_Toc505963844)

[2.1 Creating directory: 9](#_Toc505963845)

[2.2 XML schema registration using BFILE: 9](#_Toc505963846)

[3. Unstructured type (CLOB type) 11](#_Toc505963847)

[3.1 Creation 11](#_Toc505963848)

[3.2 Input of data 11](#_Toc505963849)

[3.3 Output of data 12](#_Toc505963850)

[a) EXTRACT() 12](#_Toc505963851)

[b) EXTRACTVALUE() 13](#_Toc505963852)

[c) GLOBAL() 15](#_Toc505963853)

[4. Relational type of data from XML stored data 16](#_Toc505963854)

[4.1 Creation 16](#_Toc505963855)

[a) XML creation 16](#_Toc505963856)

[b) Uploading xml file: 17](#_Toc505963857)

[d) Table creation: 17](#_Toc505963858)

[4.2 Loading XML in Relation Table 18](#_Toc505963859)

[5. XML type of data from relational type of data 19](#_Toc505963860)

[Creation 19](#_Toc505963861)

[Input of data 19](#_Toc505963862)

[Generating XML data from relational data: 19](#_Toc505963863)

[6. Conclusion 20](#_Toc505963864)

# Task description

1. With XML editor (EditX, Oxygen, …) design and create XML documents and schema files.
2. Design and creation of XML database data storage object-relational tables with XMLType use:  
   – unstructured type (CLOB type);  
   – structured type (OR type);  
   – if You use Oracle12, binary type.
3. Perform input of data (as text and use **bfile** (directory creation and use of large object function **bfile**)).
4. Perform extraction of data (queries (4)) from tables.
5. Perform registration of schema.
6. Create table for structured type of storage with use of schema.
7. Perform 4 queries.
8. Perform queries:  
   – to achieve relational type of data from XML stored data;  
   – to achieve XML type of data from relational type of data.
9. Conclusions.

# Design and create XML documents and schema files

I will create two xml and the schemas for these files:

## 2.1 XML files

### Customer.xml

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?> <Customers>  <Customer id="101">  <Fullname>Merry kandie</Fullname>  <Booking>  <Days>5</Days>  <Check-in>05-01-2017</Check-in>  <Check-out>10-01-2017</Check-out>  </Booking>  <Address>  <Email>meery@outlook.com</Email>  <Phone>+178 45698715</Phone>  <Country>Canada</Country>  </Address>  </Customer>  <Customer id="102">  <Fullname>John Smith</Fullname>  <Booking>  <Days>3</Days>  <Check-in>05-12-2017</Check-in>  <Check-out>08-12-2017</Check-out>  </Booking>  <Address>  <Email>john@gmail.com</Email>  <Phone>+1 75689792566</Phone>  <Country>USA</Country>  </Address>  </Customer>  <Customer id="103">  <Fullname>Chungi Kunge</Fullname>  <Booking>  <Days>10</Days>  <Check-in>25-03-2017</Check-in>  <Check-out>05-04-2017</Check-out>  </Booking>  <Address>  <Email>chungi@gmail.com</Email>  <Phone>+878 5648976256</Phone>  <Country>Japan</Country>  </Address>  </Customer> </Customers> |

### Hotel.xml

|  |
| --- |
| 1. <?xml version="1.0" encoding="UTF-8"?> <Hotels>  <Hotel id="10001">  <Name>Mercury Hotel</Name>  <Address>  <Street>Elizabetes 103</Street>  <City>Riga</City>  <Country>Latvia</Country>  </Address>  <Contact>  <Phone>+371 689547598</Phone>  <Email>mercury@hotels.com</Email>  </Contact>  </Hotel>  <Hotel id="10002">  <Name>London Eye Hotel</Name>  <Address>  <Street>Brige street 14</Street>  <City>London</City>  <Country>United Kingdom</Country>  </Address>  <Contact>  <Phone>+44 787859695</Phone>  <Email>londoneye@gmail.com</Email>  </Contact>  </Hotel>  <Hotel id="10003">  <Name>Effils tour Hotel</Name>  <Address>  <Street>Burifge street 56</Street>  <City>Paris</City>  <Country>France</Country>  </Address>  <Contact>  <Phone>+49 5645856312</Phone>  <Email>effilehotel@gmail.com</Email>  </Contact>  </Hotel> </Hotels> |

## 2.2 XML Schemas

### a) Customer.xsd

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?> <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">   <xs:element name="Customers">  <xs:complexType>  <xs:sequence>  <xs:element name="Customer">  <xs:complexType>  <xs:sequence>  <xs:element name="Fullname"/>  <xs:element name="Booking">  <xs:complexType>  <xs:sequence>  <xs:element name="Days"/>  <xs:element name="Check-in"/>  <xs:element name="Check-out"/>  </xs:sequence>  </xs:complexType>  </xs:element>  <xs:element name="Address">  <xs:complexType>  <xs:sequence>  <xs:element name="Email"/>  <xs:element name="Phone"/>  <xs:element name="Country"/>  </xs:sequence>  </xs:complexType>  </xs:element>  </xs:sequence>  <xs:attribute name="ID"/>  </xs:complexType>  </xs:element>  </xs:sequence>  </xs:complexType>  </xs:element> </xs:schema>  **Graphical representation:**  **A screenshot of a cell phone  Description generated with very high confidence** |

### b) Hotel.xsd

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?> <xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">  <xs:element name="Hotels">  <xs:complexType>  <xs:sequence>  <xs:element name="Hotel">  <xs:complexType>  <xs:sequence>  <xs:element name="Name"/>  <xs:element name="Address">  <xs:complexType>  <xs:sequence>  <xs:element name="Street"/>  <xs:element name="City"/>  <xs:element name="Country"/>  </xs:sequence>  </xs:complexType>  </xs:element>  <xs:element name="Contact">  <xs:complexType>  <xs:sequence>  <xs:element name="Email"/>  <xs:element name="Phone"/>  </xs:sequence>  </xs:complexType>  </xs:element>  </xs:sequence>  <xs:attribute name="ID"/>  </xs:complexType>  </xs:element>  </xs:sequence>  </xs:complexType>  </xs:element> </xs:schema>  **Graphical representation:**  A screenshot of a cell phone  Description generated with very high confidence |

# Creating oracle directory to store files

As I showed the creations of XML document and XML schemas in last section, now I will upload those files into database by creating **directory**:

## 2.1 Creating **directory**:

|  |
| --- |
| SQL> CREATE OR REPLACE directory XMLFILES AS 'C:\Database\_course\XML\_files\Practical work\_3' ;    In SQL Developer:  A screenshot of a cell phone  Description generated with very high confidence |

## 2.2 XML schema registration using BFILE:

|  |
| --- |
| I am going to upload xml file which I have created in the first section:These files will be uploaded:  **A screenshot of a cell phone  Description generated with very high confidence**  **PL/SQL code:**  begin  DBMS\_XMLSCHEMA.REGISTERSCHEMA(  SCHEMAURL => 'http://localhost:8080/public/Customer.xsd',  SCHEMADOC => bfilename('XMLFILES', 'Customer.xsd'),  LOCAL => TRUE,  CSID => nls\_charset\_id('AL32UTF8') ) ;  end;  A screenshot of a cell phone  Description generated with very high confidence  **Hotel.xsd** is being uploaded:  **PL/SQL code:**  begin  DBMS\_XMLSCHEMA.REGISTERSCHEMA(  SCHEMAURL => 'http://localhost:8080/public/Hotel.xsd',  SCHEMADOC => bfilename('XMLFILES', 'Hotel.xsd'),  LOCAL => TRUE,  CSID => nls\_charset\_id('AL32UTF8') ) ;  end;  A screenshot of a cell phone  Description generated with very high confidence  **XML Schemas** from SQL Developer:  A screenshot of a cell phone  Description generated with very high confidence |

# Unstructured type (CLOB type)

## 3.1 Creation

|  |
| --- |
| **Customer\_table** is created:  create table Customer\_table of XMLType  XMLTYPE store as CLOB;  /  A screenshot of a cell phone  Description generated with very high confidence  **Hotel\_table** is created:  create table Hotel\_table of XMLType  XMLTYPE STORE as CLOB  XMLSCHEMA  "http://localhost:8080/public/Hotel.xsd"  ELEMENT "Hotels";  /  A screenshot of a cell phone  Description generated with very high confidence |

## 3.2 Input of data

|  |
| --- |
| Data entry From **Customer.xml** to **Customer**\_**table:**  insert into Customer\_table values (xmltype(  bfilename('XMLFILES', 'customer.xml'),  nls\_charset\_id('AL32UTF8') ) );  /  A screenshot of a cell phone  Description generated with very high confidence  Data entry from **Hotel.xml**  to **Hotel\_table**:  insert into Hotel\_table values (xmltype(  bfilename('XMLFILES', 'hotel.xml'),  nls\_charset\_id('AL32UTF8') ) );  /  A screenshot of a cell phone  Description generated with very high confidence |

## 3.3 Output of data

### EXTRACT()

|  |
| --- |
| **QUERY\_1#:**  select EXTRACT(OBJECT\_VALUE, '/Customers/Customer/Fullname') "Fullnames"  from Customer\_table;/  A screenshot of a cell phone  Description generated with very high confidence  **QUERY\_2#:**  select EXTRACT(OBJECT\_VALUE, '/Customers/Customer[1]/Fullname') "Fullname"  from Customer\_table;  /  A screenshot of a cell phone  Description generated with very high confidence  **Query\_3#:**  select EXTRACT(OBJECT\_VALUE, '/Hotels/Hotel[1]/\*') "First\_hotel"  from HOTEL\_TABLE;  /    **Query\_4#:**  select EXTRACT(OBJECT\_VALUE, '/Customers/Customer[2]/Fullname') "Fullname",  EXTRACT(OBJECT\_VALUE, '/Customers/Customer[2]/Address/Phone') "Phone",  EXTRACT(OBJECT\_VALUE, '/Customers/Customer[2]/Address/Email') "Email"  from Customer\_table;  /  A screenshot of a cell phone  Description generated with very high confidence |

### EXTRACTVALUE()

|  |
| --- |
| **QUERY\_1#**  SELECT EXTRACTVALUE(OBJECT\_VALUE, '/Hotels/Hotel[1]/Address/Country') as Country  FROM Hotel\_table;  A screenshot of a cell phone  Description generated with very high confidence  **Query\_2#:**  SELECT EXTRACTVALUE(OBJECT\_VALUE, '/Hotels/Hotel[1]/Name') as Name,  EXTRACTVALUE(OBJECT\_VALUE, '/Hotels/Hotel[1]/Address/City') as City,  EXTRACTVALUE(OBJECT\_VALUE, '/Hotels/Hotel[1]/Address/Country') as Country,  EXTRACTVALUE(OBJECT\_VALUE, '/Hotels/Hotel[1]/Contact/Email') as Email  FROM Hotel\_table;  A screenshot of a cell phone  Description generated with very high confidence  **Query\_3#:**  SELECT EXTRACTVALUE(OBJECT\_VALUE, '/Customers/Customer[2]/Fullname') as Name,  EXTRACTVALUE(OBJECT\_VALUE, '/Customers/Customer[2]/Booking/Check-in') as Coming\_date,  EXTRACTVALUE(OBJECT\_VALUE, '/Customers/Customer[2]/Booking/Check-out') as Leaving\_date,  EXTRACTVALUE(OBJECT\_VALUE, '/Customers/Customer[2]/Address/Email') as Email  FROM Customer\_table;  A screenshot of a cell phone  Description generated with very high confidence |

### GLOBAL()

|  |
| --- |
| **Query\_1#:**  select X.OBJECT\_VALUE.GETCLOBVAL()  FROM Hotel\_table X;  A screenshot of a cell phone  Description generated with very high confidence  **Single Row mode:**  A screenshot of a cell phone  Description generated with very high confidence |

# Relational type of data from XML stored data

## 4.1 Creation

### a) XML creation

Here, I have created new XML file name “Employees.xml” for this task and I will use it:

|  |
| --- |
| <?xml version="1.0" encoding="UTF-8"?> <Cars>  <Car>  <Name>BWM auto</Name>  <Hourse\_power>200</Hourse\_power>  <Year>2015</Year>  <Condition>New</Condition>  <Price>20000 euro</Price>  <Type>Automatic</Type>  </Car>  <Car>  <Name>Mercedce</Name>  <Hourse\_power>250</Hourse\_power>  <Year>2014</Year>  <Condition>New</Condition>  <Price>15000 euro</Price>  <Type>Automatic</Type>  </Car>  <Car>  <Name>BWM X6</Name>  <Hourse\_power>350</Hourse\_power>  <Year>2018</Year>  <Condition>New</Condition>  <Price>25000 euro</Price>  <Type>Automatic</Type>  </Car>  <Car>  <Name>Audio</Name>  <Hourse\_power>240</Hourse\_power>  <Year>2016</Year>  <Condition>Used</Condition>  <Price>15000 euro</Price>  <Type>Automatic</Type>  </Car> </Cars> |

### b) Uploading xml file:

I have created another directory for this xml file so that I will try to show clear implementation for this task:

|  |
| --- |
| create or replace directory MY\_DIR as 'C:\Database\_course\XML\_files\Practical work\_3\xml files';  A screenshot of a cell phone  Description generated with very high confidence |

### Table creation:

|  |
| --- |
| CREATE TABLE Cars\_from\_xml(  name varchar2(20),  hourse\_power number,  year number,  condition varchar2(20),  price varchar2(20),  type varchar(20)  );  A screenshot of a cell phone  Description generated with very high confidence |

## 4.2 Loading XML in Relation Table

|  |
| --- |
| insert into Cars\_from\_xml (name, hourse\_power, year, condition, price, type )  select x.Name, x.Hourse\_power, x.Year, x.Condition, x.Price, x.Type  from xmltable(  '/Cars/Car'  passing xmltype(  bfilename('MY\_DIR', 'Cars.xml'),  nls\_charset\_id('AL32UTF8')  )  columns id integer path '@cid',  name varchar2(20) path 'Name',  hourse\_power number path 'Horse\_power',  year number path 'Year',  condition varchar2(30) path 'Condition',  price varchar2(20) path 'Price',  type varchar2(20) path 'Type'  ) x  ;  A screenshot of a cell phone  Description generated with very high confidence |

# XML type of data from relational type of data

### Creation

|  |
| --- |
| CREATE TABLE Product(  id NUMBER,  name VARCHAR2(20),  price VARCHAR2(20),  made\_year NUMBER,  valid\_year NUMBER,  made\_country VARCHAR2(20)  );  A screenshot of a cell phone  Description generated with very high confidence |

### Input of data

|  |
| --- |
| INSERT INTO Product values(1, 'T-shirt', '23 EURO', 2015, 2020, 'LATVIA');  INSERT INTO Product values(2, 'Shampoo', '2 EURO', 2015, 2018, 'Turkey');  INSERT INTO Product values(3, 'Milk', '4 EURO', 2016, 2018, 'LATVIA');  INSERT INTO Product values(4, 'Chocolate', '4 EURO', 2017, 2020, 'Gemany');  INSERT INTO Product values(5, 'Cheese', '3 EURO', 2016, 2019, 'LATVIA');  A screenshot of a cell phone  Description generated with very high confidence |

### Generating XML data from relational data:

|  |
| --- |
| SELECT XMLElement("Product", XMLAttributes(P.id as "iD"),  XMLForest(P.name as "name", P.price as "price",  P.made\_year as "made\_year", P.valid\_year as "P.valid\_year",  P.made\_country as "made\_country")).EXTRACT('/\*') AS XML  FROM Product P;  A screenshot of a social media post  Description generated with very high confidence |

# Conclusion

Finally, I should mention that this task is quite easier than others but while I was doing this practical work, I had a lot of difficulties and errors. For example, While I was trying to create directory on the server.

But, working with XML files was easy and I have learned many thing about XML also. Such as how to create XML schema and registering those files to the server.

I followed the slides which is provided in datubaze.wordpress.com and they helped me a lot.