



ORACLE®

Geodaten und XML: XML in der Oracle-Datenbank

Geodaten und XML



- XML-Standards und Geodaten
 - WMS: getCapabilities, getFeatureInfo
 - WFS
 - KML, GML, GPX, ...
- XML-Verarbeitung daher häufig gefordert
 - XML verarbeiten (Informationsextraktion)
 - XML konstruieren

Eingebaute Funktionen



- PL/SQL-Paket SDO_UTIL ...
 - TO_GMLGEOMETRY
 - TO_GML311GEOMETRY
 - TO_KMLGEOMETRY

 - FROM_GMLGEOMETRY
 - FROM_GML311GEOMETRY
 - FROM_KMLGEOMETRY

KML Geometrie erzeugen ...

Built-In-Funktion TO_KMLGEOMETRY



```
select sdo_util.to_kmlgeometry(geometry) KML
from geo_staten
where feature_name='LUXEMBOURG'
```

KML

```
-----
<Polygon><extrude>0</extrude><tessellate>0</tessellate><altitudeMo
de>relativeToGround</altitudeMode><outerBoundaryIs><LinearRing><co
ordinates>5.74006284,49.862178 5.77917684,49.84760601 5.78176092,4
9.83479298 5.77622484,49.82968197 5.75738316,49.83354297 5.7602019
6,49.81635801 5.75054784,49.81374504 5.759748,49.80879 5.75207496,
:
:
49.980375 5.84045712,49.968207 5.83958988,49.95673596 5.81707116,
49.94938701 5.81195484,49.93768701 5.78109816,49.93548003 5.764208
04,49.88585898 5.74006284,49.862178</coordinates></LinearRing></o
uterBoundaryIs></Polygon>
```

KML-Datei per HTTP verfügbar machen

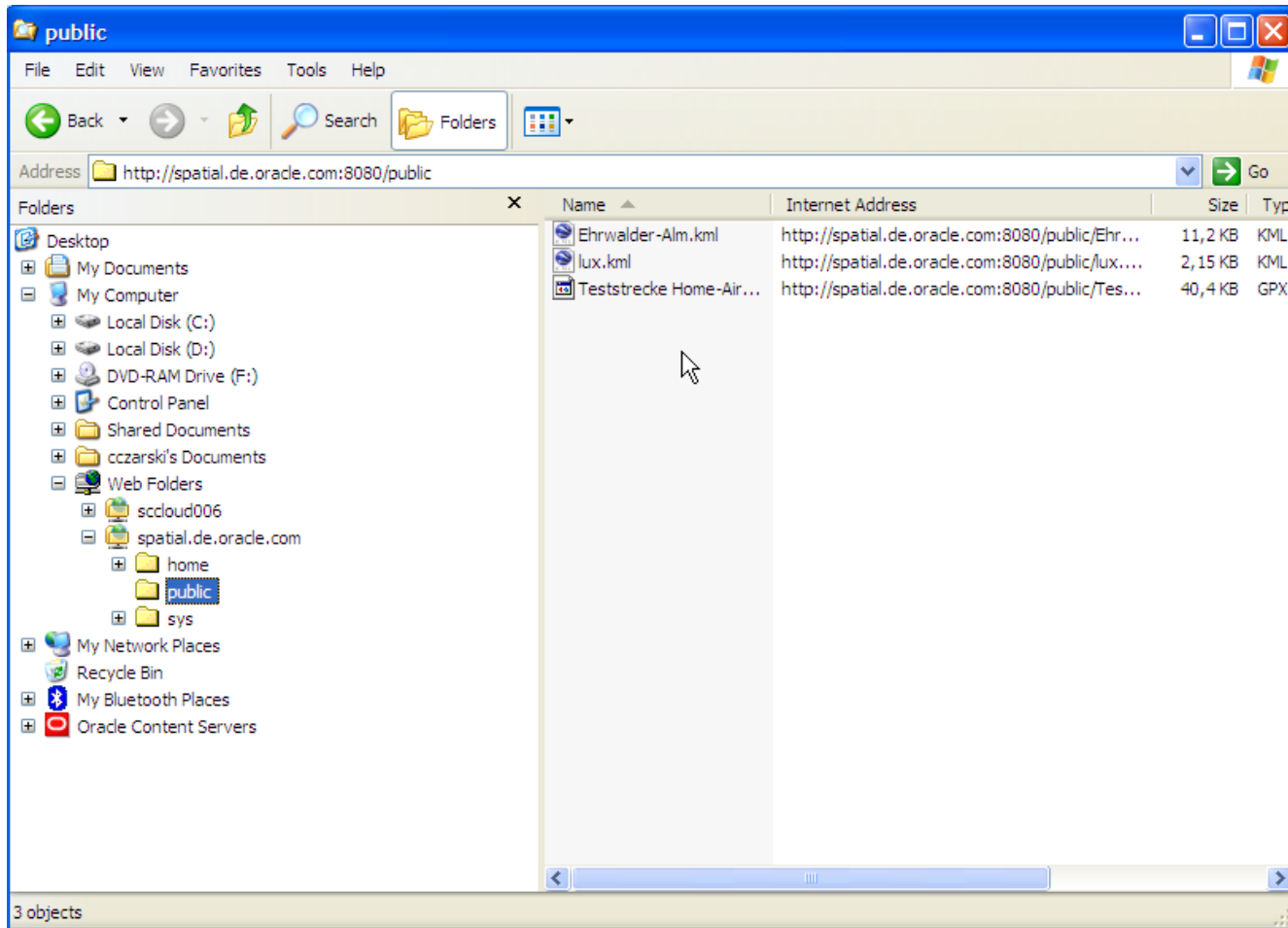
... mit dem *XML DB Repository*

```
declare
  v_create    boolean;
  v_kml       clob;
begin
  select sdo_util.to_kmlgeometry(geometry) into v_kml
  from geo_staat
  where feature_name='LUXEMBOURG';

  v_create := dbms_xdb.createresource(
    abspath => '/public/lux.kml',
    data    => v_kml
  );
end;
/
```

Zugriff mit WebDAV ...

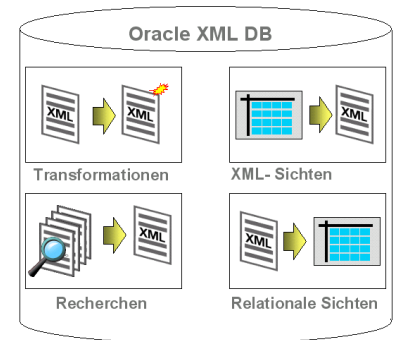
Alternativ FTP oder HTTP möglich.



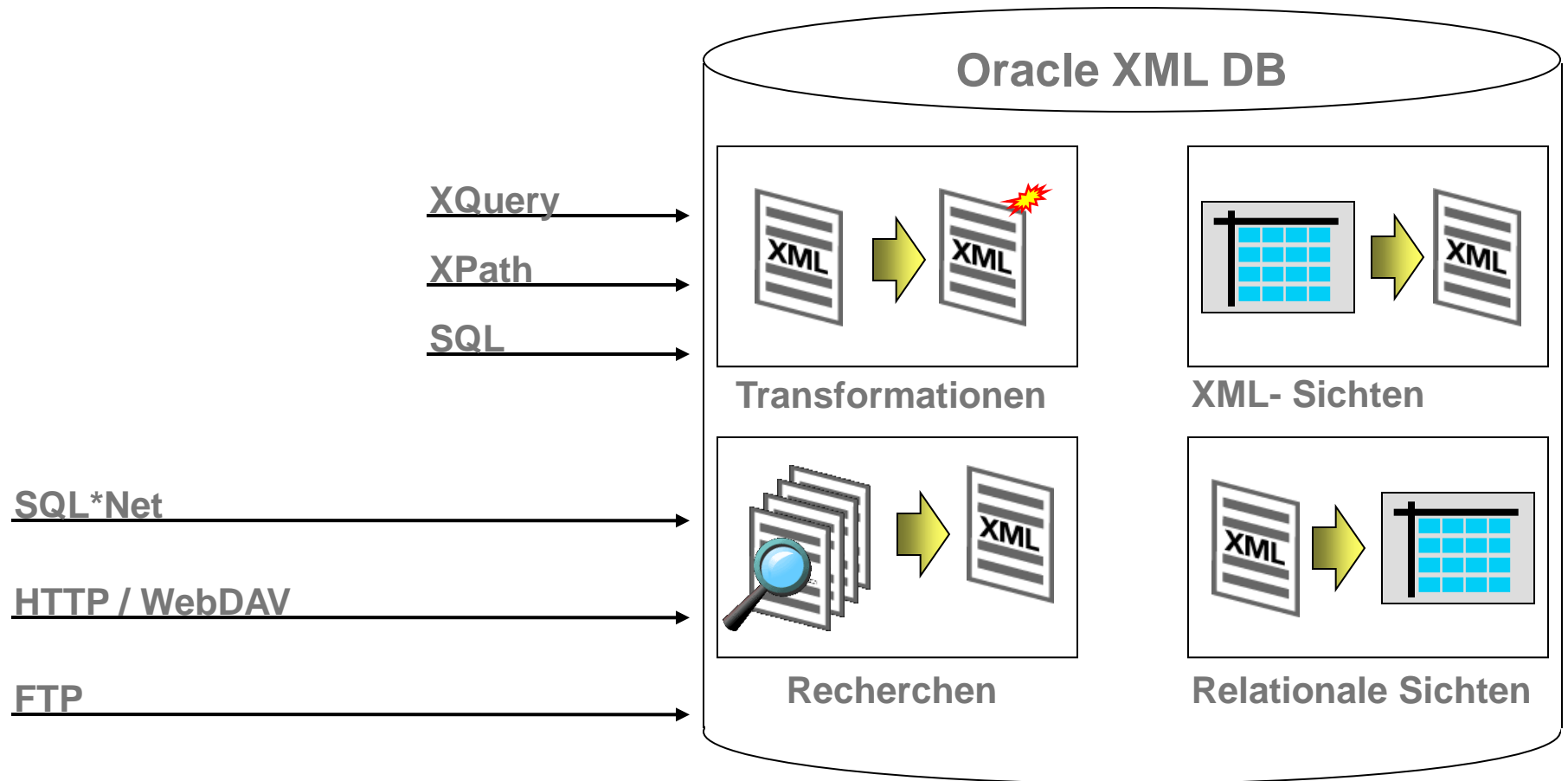
Grundlage: Oracle XML DB

Kurzprofil

- XML und SQL in einer Datenbank
- Standardkonform (W3C, SQL:2003...)
 - XML/SQL
 - XQuery
 - XML Schema, DOM
- Verfügbar ab Oracle 9i Release 2
- Alle Datenbankeditionen
- Keine zusätzliche Installation erforderlich

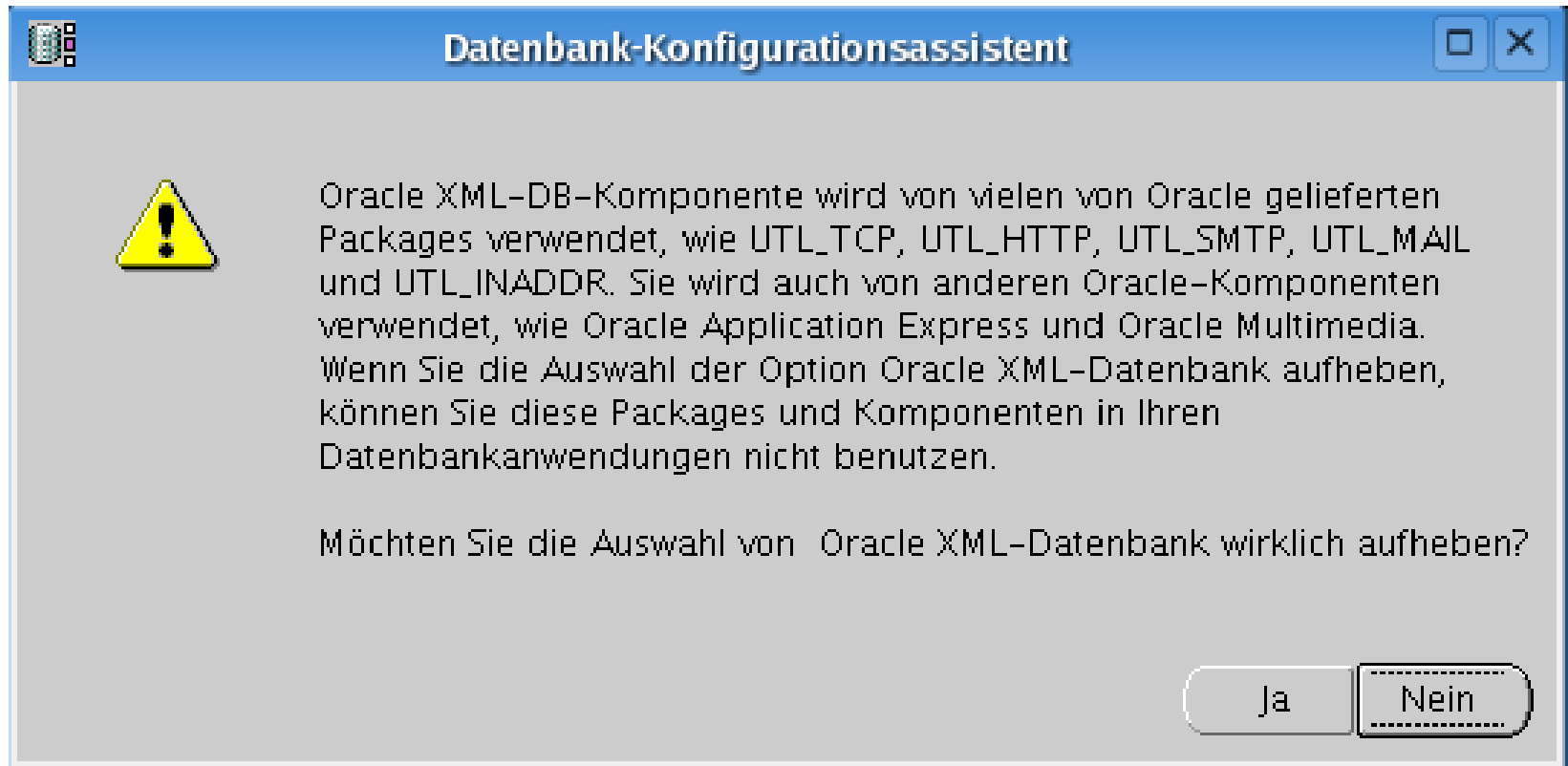


Oracle XML DB



XML DB in Oracle11g

PL/SQL ACLs via XML DB



XML DB Installation

- Überprüfen ob Funktionalität installiert ist mit `dba_registry`
- Falls Funktionalität nicht installiert
 1. XML DB Repository installieren mit `$ORACLE_HOME/rdbms/admin/catqm.sql`
 2. HTTP- und FTP -Listener aktivieren `$ORACLE_HOME/rdbms/admin/catxdbdbca.sql`

XML DB Protokollserver

Kontrolle

```
$ lsnrctl status

LSNRCTL for Linux: Version 11.1.0.6.0 - Production on 22-FEB-2008 16:14:08

Copyright (c) 1991, 2007, Oracle. All rights reserved.

Connecting to (DESCRIPTION=(ADDRESS=(PROTOCOL=IPC)(KEY=EXTPROC1522)))
STATUS of the LISTENER
-----
Alias                     LISTENER
Version                   TNSLSNR for Linux: Version 11.1.0.6.0 - Production
Start Date                15-FEB-2008 11:56:54
Uptime                    7 days 4 hr. 17 min. 14 sec
Trace Level               off
Security                  ON: Local OS Authentication
SNMP                      OFF
Listener Parameter File   /space/orabase/11gR1/network/admin/listener.ora
Listener Log File         /space/orabase/diag/tnslsnr/stuamdmuc3/listener/alert/log.xml
Listening Endpoints Summary...
  (DESCRIPTION=(ADDRESS=(PROTOCOL=ipc)(KEY=EXTPROC1522)))
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=stuamdmuc3)(PORT=1522)))

  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=stuamdmuc3)(PORT=1111))(Presentation=FTP)(Session=RAW))

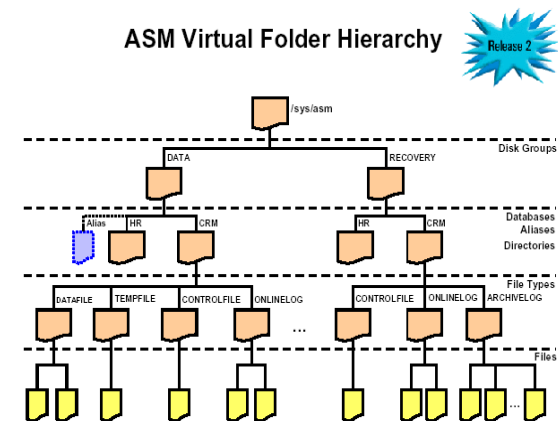
  (DESCRIPTION=(ADDRESS=(PROTOCOL=tcp)(HOST=stuamdmuc3)(PORT=2222))(Presentation=HTTP)(Session=RAW))

Services Summary...
Service "orallgR1" has 1 instance(s).....
```

XML DB in der Oracle DB

Dateien und Ordner im XML DB Repository

- File/Folder-Sicht auf eine Repository-Tabelle
- PL/SQL-Zugriff via DBMS_XDB
- Basis für FTP und WebDAV
 - Zugriffskontrolle mit ACL
 - Versionierung
 - Links
 - Metadatenverwaltung



Datentyp XMLTYPE

- Verwendbar wie jeder andere Datentyp
 - als Tabellenspalte
 - in PL/SQL Logik
- XML-Know How in der Datenbank
 - EXTRACT, EXTRACTVALUE
 - XSLTRANSFORM
 - UPDATEXML
 - APPENDCHILDXML
 - SCHEMAVALIDATE
 - ...



Zugriff mit SQL: SQL-Funktionen

- EXTRACT, EXTRACTVALUE
- EXISTSNODE

```
select
  extractvalue
    (value(po), '/PurchaseOrder/Reference') reference
,extractvalue
    (value(po), '/PurchaseOrder/User') as username
,extractvalue
    (value(po), '/PurchaseOrder/CostCenter') as cc
from PURCHASEORDER_TAB po
```

Zugriff mit SQL

Umgang mit XML-Hierarchien

- SQL-Funktionen XMLSEQUENCE, XMLTABLE
- Auch mehrere Hierarchieebenen möglich

```
select
  extractvalue
    (value(po), '/PurchaseOrder/Reference') reference
,extractvalue
    (value(po), '/PurchaseOrder/User') as username
,count(value(li)) as lineitems
from
  PURCHASEORDER_TAB po,
  table(xmlsequence(extract
    (value(po), '/PurchaseOrder/LineItems/LineItem')
  )) li
where existsnode
  (value(po), '/PurchaseOrder/[CostCenter="A10"]')=1
```

Beispiel: WMS getCapabilities-Request

Verarbeitung mit der XML DB

```
<?xml version="1.0" encoding="UTF-8" standalone="no" ?>
<!DOCTYPE WMT_MS_Capabilities (View Source for full doctype...)>
<!-- end of DOCTYPE declaration -->
- <WMT_MS_Capabilities version="1.1.1">
  <!-- MapServer version 5.2.0 OUTPUT=GIF OUTPUT=PNG OUTPUT=JPEG OUTPUT=WBMP OUTPUT=SVG SUPPORTS=PROJ SUPPORTS=AGG SUPPORTS=FREETYPE SUPPORTS=ICONV SUPPORTS=WM
  SUPPORTS=WFS_SERVER SUPPORTS=WFS_CLIENT SUPPORTS=WCS_SERVER SUPPORTS=GEOIS INPUT=TIFF INPUT=EPPL7 INPUT=POSTGIS INPUT=OGR INPUT=GDAL INPUT=SHAPEFILE -->
  - <Service>
    <Name>OGC:WMS</Name>
    <Title>OSM_Basic</Title>
    <Abstract>Open Street Map</Abstract>
  - <KeywordList>
    <Keyword>osm basic</Keyword>
    <Keyword>WhereGroup</Keyword>
    <Keyword>OpenStreetMap</Keyword>
  </KeywordList>
  <OnlineResource xmlns:xlink="http://www.w3.org/1999/xlink" xlink:href="http://osm.wherogroup.com/cgi-bin/osm_basic.xml?SERVICE=WMS&" xlink:type="simple" />
  - <ContactInformation>
    - <ContactPersonPrimary>
      <ContactPerson>Olaf Knopp</ContactPerson>
      <ContactOrganization>Wheregroup GmbH & Co.KG</ContactOrganization>
    </ContactPersonPrimary>
    <ContactPosition>Development</ContactPosition>
  - <ContactAddress>
    <AddressType>postal</AddressType>
    <Address>Siemensstrasse 8</Address>
    <City>Bonn</City>
    <StateOrProvince>NRW</StateOrProvince>
    <PostCode>53121</PostCode>
    <Country>Germany</Country>
  </ContactAddress>
  <ContactVoiceTelephone />
  <ContactFacsimileTelephone>0049-2289090380</ContactFacsimileTelephone>
  <ContactElectronicMailAddress>info@wherogroup.com</ContactElectronicMailAddress>
```


WMS Capabilities auswerten ...

1. Dokument abrufen

```
select httpuritype(  
  'http://osm.wherogroup.com/cgi-bin/osm_basic.xml?REQUEST=GetCapabilities  
  &SERVICE=WMS&VERSION=1.1.1'  
) .getxml() from dual;  
  
HTTPURITYPE('HTTP://OSM.WHEREGROUP.COM/CGI-BIN/OSM_BASIC  
-----  
<?xml version="1.0" encoding="CP850" standalone='no'?>  
<!DOCTYPE WMT_MS_Capabilities[  
<!ELEMENT VendorSpecificCapabilities EMPTY>]>  
<!-- end of DOCTYPE declaration -->  
<WMT_MS_Capabilities version="1.1.1">  
  <!-- MapServer version 5.2.0 OUTPUT=GIF OUTPUT=PNG OUT  
  :
```

WMS Capabilities auswerten ...

2. Dokument verarbeiten

```
with wms as (  
  select httpuritype(  
    'http://osm.wherogroup.com/cgi-bin/osm_basic.xml?REQUEST=GetCapabilities&SERVICE=WMS&VERSION=1.1.1'  
  ).getxml() xml  
  from dual  
)  
select  
  layer_name, layer_srs  
from wms, xmltable(  
  'for $i in /WMT_MS_Capabilities/Capability/Layer  
  for $j in $i/SRS  
  return <layer><name>{$i/Name/text()}</name>  
         <srs>{$j/text()}</srs></layer>'  
  passing xml  
  columns  
    layer_name varchar2(40) path '/layer/name/text()',  
    layer_srs varchar2(40) path '/layer/srs/text()'  
)
```

WMS Capabilities auswerten ...

3. Ergebniss ansehen

LAYER_NAME	LAYER_SRS
OSM_Basic	EPSG:31467
OSM_Basic	EPSG:31466
OSM_Basic	EPSG:31468
OSM_Basic	EPSG:31469
OSM_Basic	EPSG:31492
OSM_Basic	EPSG:31493
OSM_Basic	EPSG:31494
OSM_Basic	EPSG:31495
OSM_Basic	EPSG:31462
OSM_Basic	EPSG:31463
OSM_Basic	EPSG:31464
OSM_Basic	EPSG:31465
OSM_Basic	EPSG:4326
OSM_Basic	EPSG:25832
OSM_Basic	EPSG:25833
OSM_Basic	EPSG:31257



... mehr dazu in Kürze auf [oracle-spatial.blogspot.com!](http://oracle-spatial.blogspot.com/)

Aufbereitet in APEX ...

WMS Tester

WMS URL

Q-

☐  Layer Srs = 'EPSG:4326' 

Layer Name	Layer Srs	Bbox Minx	Bbox Miny	Bbox Maxx	Bbox Maxy
OGC:WMS	EPSG:4326	8.89189	47.082	13.9764	50.6257
copyright	EPSG:4326	-	-	-	-
TK50	EPSG:4326	8.89189	47.082	13.9764	50.6257
UK500	EPSG:4326	8.89189	47.082	13.9764	50.6257

1 - 4

Beispiel: GPX-Dateien verarbeiten

- GPS-Punkte im XML-Format ...

```
<gpx xmlns="http://www.topografix.com/GPX/1/1" tc2=" ...
  <trk>
    <name>2010-11-16T03:46:55Z</name>
    <trkseg>
      <trkpt lat="51.4799802" lon="7.9678522">
        <ele>270.7484131</ele>
        <time>2010-11-16T03:46:55Z</time>
      </trkpt>
      <trkpt lat="51.4799436" lon="7.9678679">
        <ele>264.0192871</ele>
        <time>2010-11-16T03:47:06Z</time>
      </trkpt>
      <trkpt lat="51.4797696" lon="7.9681059">
        <ele>264.4998779</ele>
        <time>2010-11-16T03:47:12Z</time>
      </trkpt>
      <trkpt lat="51.4797497" lon="7.9681919">
        <ele>264.9805908</ele>
        <time>2010-11-16T03:47:14Z</time>
      </trkpt>
    </trkseg>
  </trk>
</gpx>
```

Beispiel II: GPX-Dateien verarbeiten

- Zugriff auf Einzelpunkte mit SQL
 - Danach einfaches Umwandeln in SDO_GEOMETRY

```
select
  extractvalue(value(tp), '/trkpt/@lon', c_gpxns) x,
  extractvalue(value(tp), '/trkpt/@lat', c_gpxns) y,
  extractvalue(value(tp), '/trkpt/time', c_gpxns) m
from
  table(xmlsequence(extract(
    gpx_as_xml,
    '//trkpt',
    'xmlns="http://www.topografix.com/GPX/1/1"'
  ))) tp
```

<http://oracle-spatial.blogspot.com/2010/11/jungst-habe-ich-fur-einen-kollegen.html>

ORACLE

XML generieren

SQL/XML Standard (SQL:2003)

- XMLElement()
- XMLForest()
- XMLAgg()
- XMLComme
- XMLCDATA()
- XMLPI()
- XMLRo
- XMLSerialize()

XML sollte *immer* so generiert werden ...

Alte PL/SQL-Pakete wie
DBMS_XMLGEN oder DBMS_XMLQUERY,
Funktionen wie SYS_XMLGEN
oder gar "Stringverkettung" in
PL/SQL sind **keine**
empfehlenswerten Methoden.

Eingangsbeispiel: KML ...

- Problem:

KML wird in Google Earth nicht angezeigt ...

```
select sdo_util.to_kmlgeometry(geometry) KML
from geo_staat
where feature_name='LUXEMBOURG'
```

KML

```
-----
<Polygon><extrude>0</extrude><tessellate>0</tessellate><altitudeMo
de>relativeToGround</altitudeMode><outerBoundaryIs><LinearRing><co
ordinates>5.74006284,49.862178 5.77917684,49.84760601 5.78176092,4
9.83479298 5.77622484,49.82968197 5.75738316,49.83354297 5.7602019
:
:
49.94938701 5.81195484,49.93768701 5.78109816,49.93548003 5.764208
04,49.88585898 5.74006284,49.862178</coordinates></LinearRing></o
uterBoundaryIs></Polygon>
```


KML, KML und KML ...

- Da fehlt doch was ...

```
<kml xmlns="http://www.opengis.net/kml/2.2">
  <Document>
    <Placemark>
      <name>CDATA example</name>
      <description> ...</description>
      <Polygon><extrude>0</extrude><tessellate>0</tessellate><altitudeMode>relativeToGround</altitudeMode><outerBoundaryIs><LinearRing><coordinates>5.74006284,49.862178 5.77917684,49.84760601 5.78176092,49.83479298 5.77622484,49.82968197 5.75738316,49.83354297 5.7602019
      :
      :
      49.94938701 5.81195484,49.93768701 5.78109816,49.93548003 5.76420804,49.88585898 5.74006284,49.862178</coordinates></LinearRing></outerBoundaryIs></Polygon>
    </Placemark>
  </Document>
</kml>
```

Ein "richtiges" KML erzeugen ...

- Anwendung der SQL/XML Funktionen ...

```
select
  xmlelement("kml",
    xmlattributes('http://www.opengis.net/kml/2.2' as "xmlns"),
    xmlelement("Document",
      xmlelement("Placemark",
        xmlelement("name", 'Luxembourg'),
        xmlelement("Description", 'Landesgrenze'),
        XMLTYPE(SDO_UTIL.TO_KMLGEOMETRY(GEOMETRY))
      )
    )
  )
from geo_staaten
where feature_name = 'LUXEMBOURG';
```

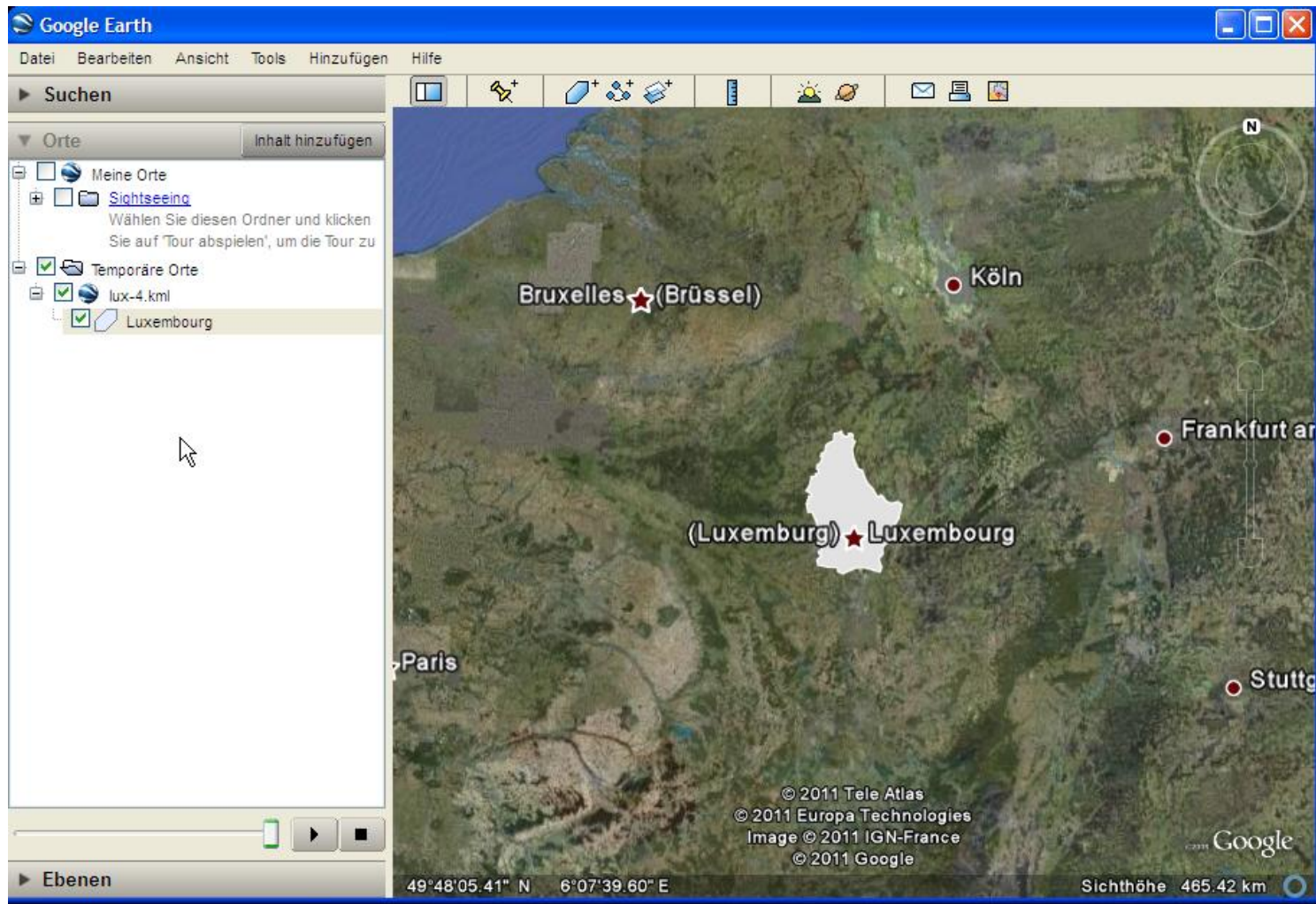
Ein "richtiges" KML erzeugen

Das Ergebnis ... als XML

```
- <kml xmlns="http://www.opengis.net/kml/2.2" >
- <Document >
- <Placemark >
  <name>Luxembourg </name>
  <Description>Landesgrenze </Description >
- <Polygon >
  <extrude>0</extrude >
  <tessellate>0</tessellate >
  <altitudeMode>relativeToGround </altitudeMode >
- <outerBoundaryIs >
  - <LinearRing >
    <coordinates >5.74006284,49.862178 5.77917684,49.84760601 5.78176092,49.
      5.759748,49.80879 5.75207496,49.80544902 5.75261712,49.79914101 5.74
      5.82932304,49.69949499 5.82407604,49.67679303 5.84414316,49.673025 5
      5.90255712,49.61116503 5.89926996,49.58347599 5.88950388,49.57964604
      5.84163684,49.53292902 5.86332792,49.53125403 5.85803592,49.52507103
      5.99592384,49.45874103 5.99964804,49.43804004 6.03934812,49.42805004
      6.32881188,49.50547497 6.36260184,49.53255003 6.36318216,49.571865 6
      6.48472212,49.80464496 6.432273,49.79607804 6.39222516,49.80625803 6
      6.19679088,49.97065698 6.16223916,49.97577798 6.12810504,50.028588 6
      6.10613496,50.16332097 6.08328504,50.16390102 6.08712516,50.15212103
      5.96692584,50.16407004 5.959755,50.12143902 5.91802884,50.11284798 5
      5.87133612,50.03679501 5.86036404,50.02004502 5.86127808,50.012145 5
      5.81707116,49.94938701 5.81195484,49.93768701 5.78109816,49.93548003
    </coordinates >
  </LinearRing >
- </outerBoundaryIs >
- </Polygon >
- </Placemark >
- </Document >
- </kml >
```

Ein "richtiges" KML erzeugen

Das Ergebnis ... in Google Earth



Ein weiteres Beispiel

GPX-Dateien erzeugen

- Das Zielformat ...

```
<?xml version="1.0" encoding="utf -8" ?>
- <gpx xmlns="http://www.topografix.com/GPX/1/1" xmlns:tc2="http://www.garmin.com/xmlschemas/TrackPointExtension/v1" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:tp1="http://www.garmin.com/xmlschemas/TrackPointExtension/v1" version="1.1" xsi:schemaLocation="http://www.topografix.com/GPX/1/1 http://www.topografix.com/GPX/1/1 http://www.garmin.com/xmlschemas/TrackPointExtension/v1 http://www.garmin.com/xmlschemas/TrackPointExtension/v1">
- <trk>
  <name>2010-11-16T03:46:55Z </name>
- <trkseg>
  - <trkpt lat="51.4799802" lon="7.9678522">
    <ele>270.7484131 </ele>
    <time>2010-11-16T03:46:55Z </time>
  </trkpt>
  + <trkpt lat="51.4799436" lon="7.9678679">
  + <trkpt lat="51.4797696" lon="7.9681059">
  + <trkpt lat="51.4797497" lon="7.968191">
  + <trkpt lat="51.4798410" lon="7.9683517">
  + <trkpt lat="51.4803376" lon="7.9686642">
  + <trkpt lat="51.4804108" lon="7.9686944">
  + <trkpt lat="51.4806803" lon="7.9686530">
  + <trkpt lat="51.4808056" lon="7.9686819">
  + <trkpt lat="51.4808884" lon="7.9688763">
  + <trkpt lat="51.4809137" lon="7.9690126">
  - <trkpt lat="51.4809888" lon="7.9695931">
```

SDO_GEOMETRY nach GPX ...

- SQL-Abfrage ...

```
select
  xmlelement("gpx",
    xmlattributes('http://www.topografix.com/GPX/1/1' as "xmlns"),
    xmlelement("trk",
      xmlelement("name", 'GPX Name'),
      xmlelement("trkseg",
        (
          select
            xmlagg(
              xmlelement("trkpt", xmlattributes(v.y as "lat", v.x as "lon"))
            )
          from table(sdo_util.getvertices(v_geom)) v
        )
      )
    ) into v_gpx
from dual;
```

SDO_GEOMETRY nach GPX

Als PL/SQL-Funktion ...

```
FUNCTION make_gpx RETURNS XMLTYPE
```

Argument Name	Typ	In/Out
P_NAME	VARCHAR2	IN
P_GEOM	SDO_GEOMETRY	IN

- Anwendungsgebiete
 - Linestrings konvertieren und auf GPS-Gerät laden
 - Bereitstellung von Routen
 - Zeitinformationen per LRS ermitteln
 - Auch Höheninformationen denkbar

<http://oracle-spatial.blogspot.com/2011/03/sdo-nach-gpx-konvertieren-ein-beispiel.html>

ORACLE

Weitere Informationen

- Blogs zum Thema Oracle Spatial und Oracle MAPS
 - <http://oracle-spatial.blogspot.com>
 - <http://oracle-maps.blogspot.com>
- Blog zum Thema SQL und PL/SQL
 - <http://sql-plsql-de.blogspot.com>
- Dokumentation: Oracle XML DB
 - <http://www.oracle.com/technetwork/database/features/xmldb/index.html>

Fragen & Antworten

