Oracle Spatial 11g: Build Web Services, GeoRaster, Network and 3D Applications
Dr. Siva Ravada
Director of Development
Oracle
Oracle Spatial Features

- **Oracle Locator**: Feature of Oracle Database XE, SE, EE
- **Oracle Spatial**: Priced option to Oracle Database EE
- **MapView**: Java application and map rendering feature of Fusion Middleware
- **Bundled Map Content**: Major roads, administrative boundaries (city, county, state, country) - worldwide coverage from Navteq
Supports all Geospatial Data types

- Locations (points of interest)
- Networks (roads, utilities)
- Imagery (satellite imagery)
- Topology (data provider)
- Polygons (admin, sales territories, high risk zones)
- 3D data type (city models)
- LIDAR Data Type
- TIN Data Type
Open Platform for Leading Applications and Tools

- Autodesk MapGuide
- Bentley
- Manifold
- MapInfo
- ESR| ArcMap
- Leica ADE
- eSpatial
- Intergraph GeoMedia
Oracle Spatial 11g Enabled Applications

Scrollable, Interactive Maps

Spatial Web Services

3D, Point Clouds, and LIDAR

Geocoding & Routing

Oracle BI Dashboards

Raster Imagery
GeoRaster

- A data type to store raster data
  - Satellite images, remote sensing data
  - Multi-band, multi-layer
  - An XML schema to store Metadata
    - Data source, layer information
    - Geo Referencing information

- Functionality
  - Storage and indexing of raster data
    - No size limit for each raster object
  - Generate resolution pyramid
  - Query and analysis
  - Delivering to external consumers
    - Publish as JPEG, GIFF images
A Raster Application

Raw imagery: Airborne imagery, Satellite imagery, DEM

ETL

Replicated geo-spatial data

Publish data over the internet (read-only)
ETL tool – the GDAL GeoRaster Driver

- GDAL is the best open source geospatial ETL tool/API for raster data
  - It now natively supports importing and exporting many formats, to/from SDO_GEORASTER, including GeoTIFF, JPEG2000, ECW, NITF, HDF, NetCDF, ERDAS IMG, USGS DEM, SPOT, and more.
  - GDAL is written in C++ and has great performance
  - It provides C/C++, Java, Python API for accessing GeoRaster
  - It provides many tools. Two of the important ones are:
    - gdal_translate – utility to translate raster formats to/from GeoRaster objects
    - gdalinfo – utility to view information about a raster, such as a GeoRaster object
Raster Bands, Layers, and Blocks

- RGB Image: 3 Bands (Red, Green, Blue)
  - Consider an Image with 8096x8096 pixels
  - Each block is 256x256 pixels
  - This results in 1024 blocks for the whole image
- Hyperspectral image: hundreds of bands
Georeferencing Using GCP

- Georeferencing is the process of mapping pixels on the image to ground coordinates
- GCP: stands for Ground Control Points
- **GCP Model**: GeoRaster supports a generic GCP model. In the current release, 2D cell coordinates, 2D and 3D model coordinates are supported.
- **GCP Storage**: GeoRaster defines a GCP XML schema and can (optionally) store GCP natively in the metadata of GeoRaster objects.
- **GCP Manipulation**: GeoRaster provides a set of update and query functions to manipulate GCP’s and related data.
The GeoRaster Java API

- Sample Applications: using this Java API, users can easily develop simple applications, particularly web-based applications. GeoRaster viewer is a simple application built with this API.
The GeoRaster Java API

- The GeoRaster JAVA API
  - `oracle.spatial.georaster`: Provides a complete mapping of the SDO_GEORASTER object type and its metadata to Java objects, and provides support for the core GeoRaster features
  - `oracle.spatial.georaster.sql`: Provides a Java wrapper of the GeoRaster PL/SQL API for some server-side operations
  - `oracle.spatial.georaster.image`: Provides support for generating Java images from a GeoRaster object and for processing the images

- The core georaster package and the sql package are implemented in pure Java. It doesn’t depend upon Java 2D and JAI.
- The image package is based on Java 2D and JAI. This allows users to leverage all the strength and advanced capabilities from Java 2D and JAI. Users can easily develop web applications and other image processing applications.
Blocking Size Optimizer – Minimize Padding

- GeoRaster supports regular blocking and padding is applied to the right and lower boundary blocks if necessary. However, padding wastes some storage space.

- The blocking size optimizer would automatically optimize the blocking size based on the GeoRaster dimension sizes and pyramid levels so that the padding can be minimized:
  - "blocking=optimalPadding" in storageParam. This applies to any functions which use storageParam, such as mosaic, subset, scaleCopy, mergeLayers, reproject. It automatically adjusts blocking sizes.
  - SDO_GEOR_UTL.calcOptimizedBlockSize Users can use it to pre-compute the optimal blocking sizes and then apply.
Blocking Size Optimizer – An Example

-- original image size
518x518x3 blocked into 256x256x3

sdo_geor.changeFormatCopy
(gr1,
  'blocking=optimalPadding
  blocksize=(128,96,1)',
  gr2);

-- result image blocked into 130x74, near zero padding
**Raster CS Transformations**

**SDO_GEOR.reproject:** transform GeoRaster raster data from one projection to another projection. All oracle spatial supported coordinate systems are supported. Five re sampling methods are supported

- NN, Bilinear, Cubic, Avarage4 and Average16.

Supports two options

- Reproject persistently. Reprojects a GeoRaster object and stores the result as a new GeoRaster.

- Reproject on-the-fly. This is equivalent to getRasterSubset except the window-based cropping result is transformed into a different projection. The result is stored as a single BLOB.

- Both options support window and pyramid level based query (subsetting using AOI)
Raster Reprojection – An Example

```sql
sdo_geor.reproject ( gr1, 'resampling=cubic', 'blocksize=(256,256,3) interleaving=BIP', 26988, gr2 );
```

From: SRID 26986
"NAD83 / Massachusetts Island"

To: SRID 26988
"NAD83 / Michigan North"
Raster Distribution Service

Raw imagery: Airborne imagery, Satellite imagery, DEM

Replicated geo-spatial data

Publish data over the internet (read-only)
Users can order images for a custom area of interest
Polygon-based Raster Clipping

SDO_GEOR.getRasterSubset
- Allows users to clip the query result along the polygon (irregular) boundary
Networks and Routing
Sample Application: Shortest Path Analysis

Street Addresses
747 Howard Street, San Francisco, CA → 1099 Lombard Street, San Francisco, CA

Network Nodes and Links
5% on link 1234567 → 10% on link 89101112

Path Represented in Nodes and Links
Path consisting 500 nodes […] and 499 links […]

Path Geometry
Path geometry is ….

Path Displayed On Map
NDM Application Architecture

**DB Layer**
- NDM PL/SQL API
  - SDO_NET Package
    - Create Network
    - Validate Network
    - Partition Network
    - Generate Partition BLOBs
- Network Data Model
  - Network Metadata
    - Node Table
    - Link Table
    - Path Table
    - User Data
    - Partition Table
    - Partition BLOB Table
    - Component Table
  - NAVTEQ ODF Data

**Mid-tier Layer**
- NDM JAVA API
- Demo JSPs and Java Beans
- J2EE
- Mapviewer Server
- Geocoder Server

**Client Layer**
- NDM Demo Web application
NDM Application: Mid-tier Architecture

NDM JAVA API
- oracle.spatial.network.lod package
  - Network Analysis API
  - Cached Network Read/Write
  - Partition BLOB Translator Interface
  - Link/Node Cost Calculator Interface
  - Network Constraint Interface
  - Goal Node Filter Interface
  - User Data IO Interface

NDM Demo Web application

Initialization:
- Get database connection
- Get network analyst
- Get geocoder http client

Network Analysis:
- If input is address or lat/lon, call geocoder client API.
- Call analysis methods in Network Analyst
- Read path geometry
- Display results with Mapviewer

Customization:
- Link Cost Calculator Impl.
- Network Constraint Impl.
- Goal Node Filter Impl.
- User Data IO Impl, e.g. trucking

Demo JSPs and Java Beans

Geocoder Server

Mapviewer Server

J2EE

Oracle.
NDM Customization (I)

- Users can easily plug in their implementations for
  - Network constraints
    - Trucking restrictions
    - Turn restrictions
    - Avoid toll road
    - Avoid/use Highway
  - Multiple Cost calculators
    - Travel distance
    - Travel time
    - Local/Highway/Scenic Drive preference
    - Penalty of road under construction/traffic congestion
  - Goal nodes
    - Restaurants within 10 min’s drive
NDM Customization (II)

- Our data model also supports
  - Application specific data
    - Trucking data
    - Application attributes/logic
  - Dynamic updates to the network
    - Temporarily disable links/nodes due to roadblock/accidents
    - Temporarily increase cost of a link due to traffic congestion
Multiple Cost Support in Path Analysis

- Path/Subpath analysis now supports multiple costs in a single analysis
- First cost is the one NDM optimizes
- Examples
  - Shortest distance paths with travel time
  - Fastest paths with travel distance
User Object Support in Network Constraints

- User defined object can be accumulated and used in network constraints during network analysis
- Available in network constraint implementation
- Examples:
  - Previously visited link/node information
  - Logical restrictions based on links
Hierarchical A* shortest path Analysis

- Support A* shortest path algorithm
- Provide user defined heuristic cost function
- Support hierarchical shortest path analysis
- Better performance than Dijkstra Algorithm as less nodes are explored
Traveling Sales Man (TSP) Analysis

- Minimum cost tour that includes all given nodes
- Support Points on Network as nodes to be visited
- Node visit order can be enforced using network constraints
- This is useful scheduling problems where a service representative has to visit a number of customers
Drive Time Polygon Generation

- A spatial representation (polygon) based on minimum cost network coverage
- Concavehull polygon or convexhull polygon (accuracy and performance)
- Example
  - Compute Drive Time Polygon (with travel time as link cost) from a service station
  - Use the polygon to determine if a given address can be reached within a given time using point in polygon operation
Concave Hull
Convex Hull vs Concave Hull
Routing Engine

- Routing Engine is an application built on top of NDM LOD engine
- This is a custom application that ships with Spatial
- Routing Engine uses all the extensibility of the underlying NDM engine
- Routine engine is also customized to understand truck routing attributes
  - Truck routing is an example of customizing the NDM engine specific to an application based on user data
Routing engine vs NDM API

- **NDM API** is a tool kit for developing network applications
  - Provides a Java API
  - The tool kit can be used to build thick client or thin client applications
  - Completely customizable to suit the users application data
  - Provides customization for network search based on user data

- **Routing engine** is a J2EE application specifically built for point to point routing
  - Built on top of NDM API
  - Provides XML api
  - Not customizable with user data
  - Supports Auto and Truck routing on the street network
  - Works out of the box with NAVTEQ ODF data product
XML API for Routing

- `<xml version="1.0" standalone="yes"?>
  <route_request id="8" route_preference="shortest"
  road_preference="highway" vehicle_type="truck"
  return_driving_directions="true" distance_unit="mile"
  time_unit="hour" return_route_geometry="false" />

- Route Request vehicle_type
  - (auto|truck) optional, defaults to auto

- Route Request truck_type
  - truck_type: (delivery|public|trailer) optional, no default

- Works out of the box with NAVTEQ ODF data product
Truck Routing

- truck_height: (positive float) optional, no default
- truck_length: (positive float) optional, no default
- truck_per_axle_weight: (positive float) optional, no default
- truck_weight: (positive float) optional, no default
- truck_width: (positive float) optional, no default
- length_unit: (metric|us) optional, default US
- weight_unit: (metric|us) optional, default US
- Truck height, length and width are specified in length_unit units
- Truck per axle weight and weight and specified in weight_unit units
- Point Addressing data has an exact long/lat for each address
  - This is different from the range based addressing where each road segment has an address range

Address range: 1 to 100

- Oracle Parkway
  - House# 1
  - Actual House #50
  - House# 50
  - House# 100
Point Address Geocoding

- Support for this feature requires a new data table in addition to the current set of Geocoder tables.
- No interface change required, if this table exists we use it to refine the result using the exact long/lat provided in the table.
- Customers can buy this new data set from NAVTEQ.
- NAVTEQ Point Addressing data product.
- This is in addition to the NAVTEQ Geocoding data in Oracle Data Format (ODF).
Moving Objects
Tracking of Moving Objects
Oracle CEP - Overview

• Designed to target event processing applications
  – Network monitoring and traffic engineering
  – Smart Meters
  – Manufacturing Execution systems
  – Supply chain planning
  – Web logs & Click stream analysis

• Oracle CEP provides a platform to process in-flight & Reference data
  – Domain specific application server capable of extreme low latencies and high throughput
  – Designed to scale-out and be highly available
  – SQL like Query language a.k.a Continuous Query Language
Moving Objects Demo Application

- Polygon Manager
- Oracle CEP Application
- Dashboard

Oracle CEP Application (EPN)
Moving Objects Demo Application

- Polygon Manager
- Oracle CEP Application
- Dashboard

Oracle CEP Application (EPN)

SQL

Polygon Manager

Zone Definitions
Moving Objects Demo Application

- Polygon Manager
- Oracle CEP Application
- Dashboard

Oracle CEP Application (EPN)
Moving Objects Demo Application

- Polygon Manager
- Oracle CEP Application
- Dashboard

Oracle CEP Application (EPN)
• Polygon Manager
• Oracle CEP Application
• Dashboard

Oracle CEP Application (EPN)
Moving Objects Demo Application

- Polygon Manager
- Oracle CEP Application
- Dashboard

Oracle CEP Application (EPN)

Resource Locations

SQL

AQ

Polygon Manager

Zone Definitions
Moving Objects Demo Application

- Polygon Manager
- Oracle CEP Application
- Dashboard

Oracle CEP Application (EPN)

Resource Locations

Zone Definitions

SQL

AQ

Polygon Manager
Moving Objects Demo Application

- Polygon Manager
- Oracle CEP Application
- Dashboard

Oracle CEP Application (EPN)

Resource Locations

SQL

AQ

Polygon Manager

Zone Definitions
Moving Objects Demo Application

- Polygon Manager
- Oracle CEP Application
- Dashboard

Oracle CEP Application (EPN)

- Resource Locations
- Matches and Alerts

SQL

Zone Definitions

Polygon Manager
Polygon Manager

- Polygon zones defined in the database as Spatial objects
- MapViewer can be used to create and maintain polygon zones
- CEP joins with the polygon objects cached within the grid and determines point-in-polygon matches and fires alerts when a boundary is breached
  - e.g. Enter and Exit events
- A web-based dashboard consumes the events and lists them within a table
Oracle Spatial Web Service Framework

Oracle Application Server

Oracle Database / Spatial

- WFS Client
- CSW Client
- OpenLS Client
- WMS Client

- Standard Interface
- SOAP/XML
- XML

- JDBC

- Components
  - WFS
  - CSW
  - OpenLS
  - MapViewer
  - WMS
  - GML
  - HDM
  - GeoRaster
  - SDO_GEOMETRY
  - TIN/Point Clouds
3D Applications

- Location-based services
  - Augmented reality
- GIS Analytical Modeling
  - Terrain (2.5D) and 3D objects
- City Planning/Administration
- Infrastructure Design
  - Accurate descriptions of objects
3D Mash-ups
3D in Spatial 11g

Types

- SDO_GEOMETRY (3D)
- SDO_TIN
- SDO_POINT CLOUD

Building Models,

Surface Modeling

Scene,

Object Modeling

Efficient Storage

Query

Analysis
High Performance Dissemination of 3D data

Web 3D Services

Image server
WCS

Feature server
WFS

2DMap server
WMS

Oracle Spatial DB

GeoRaster
2D Data

3D Data: Geometry, PointClouds, TINs

Graphics data: Textures, Colors

• Deliver 3D display elements
  • VRML97; X3D; KML, etc.

• Generate 3D scenes with predefined initial viewpoint
Visualization Support

- DB Schema for visual elements
  - Combine visualization elements from
    - COLLADA (Collaboative Digital Asset Exchange Format)
    - X3D (latest incarnation of VRML)
    - Java3D
  - Easily associate Textures with Buildings
- All the required elements for visualization persistently stored in the database
- Support multiple rendering engines
  - Java 3D, X3D, OpenGL
A thick client 3D visualizer

Client Viewer
1. Picks view frame in DB
2. Derives associated scene & LOD-0-themes
3. Queries & streams geoms from theme tables
4. Queries & streams textures from texture tables

Java3D
Scene Graph

JDBC

User theme tables
- BUILDINGS_LOD3
- BUILDINGS_LOD2
...
- GROUND

Texture tables
- TEXTURES
- TEXTURE_COORDS

Spatial 3D Metadata
- SDO_3DTHEMES
- SDO_SCENES
- SDO_VIEWFRAMES
- SDO_STYLES
- SDO_LIGHTSOURCES

SDO_GEOMETRY, BLOB (textures), Texture coordinates
For more Information on Spatial and MapViewer

search.oracle.com
Oracle Spatial
ORACLE IS THE INFORMATION COMPANY