“While hardcore GIS professionals may start their work in other applications, when they want to solve spatial problems in production and with web- and IoT-scale data, Oracle gives them the platform to do so.”

Analysts: Rowan Curran with Holger Kisker, Ph.D. and Emily Miller
September 1, 2016
Build Applications on Spark, Hadoop, and NoSQL with Oracle Big Data Spatial and Graph

Jim Steiner
Vice President, Product Management

Siva Ravada
Senior Director, Product Development

Zhe Wu
Architect, Product Development

September 22, 2016
Safe Harbor Statement

The following is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions. The development, release, and timing of any features or functionality described for Oracle’s products remains at the sole discretion of Oracle.
Program Agenda

1. Big Data Spatial and Graph feature overview
2. Common Use Cases and Coding Examples
3. Benefits for the Developer
4. Questions and Answers
Oracle Big Data Spatial And Graph
A Complete Big Data Portfolio

Oracle Data Integrator
Oracle GoldenGate
Oracle Big Data Preparation
Oracle Big Data SQL
Oracle Big Data Connectors
Oracle NoSQL Database

Oracle BI EE + Visual Analyzer
Oracle Big Data Discovery
Oracle Big Data Spatial and Graph
Oracle Advanced Analytics
Oracle Data-as-a-Service
Oracle Applications
Oracle Big Data Appliance
Oracle Public Cloud
Oracle Big Data Cloud
Big Data Defined

“Big Data: Techniques and Technologies that Enable Enterprises to Effectively and Economically Analyze All of their Data”
Big Data Challenges

Requires more development resources and data scientists

Build your own environment from commodity hardware and open source software.
Big Data Solution

Make developers and data scientists more productive

- Popular languages, development frameworks and service APIs
- In-Memory, parallel OOTB Graph Algorithms
- Massively-scalable Graph Database
- Pre-built, parallel, MapReduce and Spark Spatial Algorithms
- Raster and Vector processing Frameworks

Optimized, pre-configured Big Data Appliance and Cloud platforms
Graph and Spatial Analysis – It is about relationships

- Are things in the same location? Who is the nearest? What tax zone is this in? Where can deliver in 35 minutes? What is in my sales territory? Is this built in a flood zone?

- Which supplier am I most dependent upon? Who is the most influential customer? Do my products appeal to certain communities? What patterns are there in fraudulent behavior?
Big Data Spatial and Graph

Big Data often needs an Organizing principle – Data Harmonization

Big Data analysis is often about relationships not aggregation

Big Data platform is economically compelling for working with massive data sets found in spatial and graph workflows
Motivation for Oracle Big Data Spatial

Emergence of Hadoop for spatial analysis in business and spatial workflows

Existing Hadoop-based Spatial technologies are GeoSpatial-centric not Application-centric

No significant commercial offerings
What problems can Big Data Spatial analysis address?

- **Data Harmonization** using any location attribute (address, postal code, lat/long, placename, etc).
- **Categorization and filtering** based on location and proximity.
- **Preparation, validation and cleansing** of Spatial and Raster data.
- **Visualizing** and displaying results on a map.
- **Spatial querying and analysis** of Hadoop data with SQL.
Big Data Spatial Features

Data enrichment service API using GeoNames and geometry hierarchy data

MapReduce routines for distance calculations, PointInPolygon, buffer creation, Categorization, KMeansClustering, Binning

Spatial processing of data stored in HDFS or NoSQL. Raster processing operations: Mosaic and sub-set operations. Geodetic and Cartesian data

HTML5 Map Visualization API

Hive SQL API
Query from Oracle DB with Big Data SQL & Oracle SQL Connectors for Hadoop
Graph Data Model

What is a graph?

- A set of links and nodes (and optionally attributes)
- A graph is simply **linked data**

Why do we care?

- Graphs are everywhere
  - Social networks/Social Web (Facebook, Linkedin, Twitter, Baidu, Google+,...)
  - Cyber networks, power grids, protein interaction graphs
  - Knowledge graphs (IBM Watson, Apple SIRI, Google Knowledge Graph)
- Graphs are intuitive and flexible
  - Easy to navigate, easy to form a path, natural to visualize
  - Do not require a predefined schema
Property Graph: Big Data Spatial and Graph
Fast analytics with horizontally scalable storage

- **Parallel In-memory-based Analyst (PGX)**
  - 39 built-in memory-based graph analysis algorithms
  - Property Graph Query Language (PGQL)
  - Smart filtering of large graphs

- **Flexible interfaces**
  - Python, Groovy
  - Java, Tinkerpop, Blueprints, Gremlin
  - Apache Lucene and SolrCloud

- **Massively-Scalable Graph Database**
  - Multiple back-ends: NoSQL, Hbase, Oracle Database
  - Scales securely to **10s of billions** of nodes/edges
Computational Analytics: Built-in Package

Rich set of built-in parallel graph algorithms

- Detecting Components and Communities
  - Tarjan’s, Kosaraju’s, Weakly Connected Components, Label Propagation (w/ variants), Soman and Narang’s Spacification

- Ranking and Walking
  - Pagerank, Personalized Pagerank, Betweenness Centrality (w/ variants), Closeness Centrality, Degree Centrality, Eigenvector Centrality, HITS, Random walking and sampling (w/ variants)

- Evaluating Community Structures
  - Conductance, Modularity, Clustering Coefficient (Triangle Counting), Adamic-Adar

- Path-Finding
  - Hop-Distance (BFS), Dijkstra’s, Bi-directional Dijkstra’s Bellman-Ford’s

- Other Classics
  - Vertex Cover, Minimum Spanning-Tree (Prim’s)

- Link Prediction
  - SALSA (Twitter’s Who-to-follow)

... and parallel graph mutation operations

- The original graph
  - Left Set: “a,b,e”

- Create Bipartite Graph
  - Filter-Expression

- Create Undirected Graph

- Sort-By-Degree (Renumbering)

- Filtered Subgraph

- Simplify Graph

Lei Set: ”a,b,e”

Create Bipartite Graph

Sort-By-Degree (Renumbering)

Filtered Subgraph

Simplify Graph
Modeling and Analyzing The Internet of Things

Massive scale – Trillions of Connections

Dynamic Network

Includes metadata as well as events/enriched data

Extensible by other data source
On Premise or Cloud
Oracle Big Data Cloud Service

• Fast, reliable and secure service on Oracle Big Data Appliance
• Fully automated service for Hadoop and Spark
• Includes Big Data Spatial and Graph in subscription
Program Agenda

1. Big Data Spatial and Graph feature overview
2. Common Use Cases and Coding Examples
3. Benefits for the Developer
4. Questions and Answers
Demo: Tax Fraud Detection and Visualization with Oracle Big Data Spatial and Graph Property Graph

Java, Groovy, BDSG, Oracle NoSQL Database, PGQL, Cytoscape Visualization, Graph Analytics
Demo: Build Recommender System using Personalized PageRank, a built-in analytics in Oracle Big Data Spatial and Graph Property Graph

Java, Groovy, BDSG, Apache HBase, Solr, Graph Analytics, Graph Navigation, and Python
Building Spatial Analytical Applications with Java and Python
Image Processing with feature highlighting

• Given a set of raster data for elevation, calculate the HillShade from the raster data

• Input: Single band digital elevation model data

• Output: Single band raster image showing the hill shade
Shaded Relief calculation

- Input: NXM pixels where each pixel is a floating point number denoting elevation
- Find the shaded relief from the DEM
- Algorithm
  - Look at the values of 8 neighbors and the current pixel value and generate a new pixel
  - Needs the neighbor’s pixel value to calculate the new pixel value for the current pixel
Raster Data Analysis Framework

• Traditional algorithm will work very well in Map-Reduce framework

• Once the data is loaded into HDFS rest of the processing can be done using standard algorithms

• Very effective for raster data processing as many map-reduce nodes can work together to produce the result in a short amount of time

• Customers don’t need to write map-reduce algorithms for image processing

• All the operations are performed on a catalogue of images
  – Customers can logically combine any number of images into a catalogue or into a virtual mosaic
Raster Data processing workflow

• Copy the image files from NFS to HDFS
  – Use the provided ETL tools, no need to write code for this

• Process the images with Map-Reduce
  – We provide the framework to convert single node image processing classes to map-reduce
  – Users need to provide the image processing class for any custom image processing and analysis
Copy the input Raster data to HDFS

Using PYTHON script

```python
# ********** Execute the script with a valid yarn user with access **********
if __name__ == "__main__":
    if len(sys.argv) != 3:
        print "usage: imageloader.py <innputLocalFolder> <relativeHDFSPath> where \n"
        "<innputLocalFolder> is a comma separated string with multiple files or folders with images.\n"
        "<relativeHDFSPath> HDFS relative path of the user executing the job to write the output.\n"
    else:
        try: printConf ()
        doLoader (sys.argv[1] , sys.argv[2])
        listResult (sys.argv[2])
```
Copy the input Raster data to HDFS

Using PYTHON scripts

#**********************************************************
# TO-DO Configure these properties as necessary
#**********************************************************

SHARE_DIR    = "/net/host100/scratch/abcduser/test/
INSTALLATION = "/opt/oracle/oracle-spatial-graph/
ALL_ACCESS_DIR = SHARE_DIR + "write/"
GDAL_LIB      = ALL_ACCESS_DIR + "gdal"
GDAL_DATA     = SHARE_DIR + "data"
RASER_INST    = INSTALLATION + "spatial/raster/
LOADER_JAR    = RASER_INST + "jlib/" + "hadoop-imageloader.jar"

#Add more hadoop_classpath jars as necessary
CLASSPATH_JARS  = ["hadoop-imageprocessor.jar", "hadoop-raster-fwk-api.jar", "gdal.jar"]
Calculate the slope

```java
private double calculateSlope(double[] neighbors, double pixelWidthX, double pixelWidthY) {
    double slope = 0;
    dx = ((neighbors[0] + (2 * neighbors[3]) + neighbors[6]) -
           (neighbors[2] + (2 * neighbors[5]) + neighbors[8])) / (8 * pixelWidthX);
    dy = ((neighbors[6] + (2 * neighbors[7]) + neighbors[8]) -
           (neighbors[0] + (2 * neighbors[1]) + neighbors[2])) / (8 * pixelWidthY);
    slope = Math.sqrt(Math.pow(dx, 2) + Math.pow(dy, 2));
    return slope;
}
```
Calculate Hillshade

double zenithRadians = 90-altitude * Math.PI / 180;
double azimuthRadians = azimuth_math * Math.PI / 180;
clearNeighborhood(neighborhood);
slopeRadians = Math.atan(slope * zFactor);
aspectRadians = calculateAspectRadians();
hillShade = calculateHillShade(zenithRadians, azimuthRadians, slopeRadians, aspectRadians);
finalFloatData[position] = hillShade;
Executing the analysis process as a map-reduce job

Now that the catalog and mosaic configuration are set, execute the job using the hadoop jar command with the `hadoop-imageprocessor.jar` and all the options described in the last steps and wait until it finishes.

```
```
Categorize data based on a hierarchy -- Java

• Data containing location references is processed and categorized based on a custom geometry hierarchy

• Input: Tweet data for January 2016 time period

• Output: categorized data that associates each tweet to a geometry in the hierarchy
Load the input data on HDFS for spatial processing

• Assume that the data is stored on an NFS file system that is also accessible from the HDFS server

• The input data is stored as a text file in the JSON format with one record for one tweet

  hadoop fs -put /opt/oracle/oracle-spatial-graph/spatial/vector/HOL/data/tweets.json
  /user/oracle/HOL/tweets.json

• It is as simple as copying the data from the NFS to HDFS using haddop file system commands
  – No pre-processing required so the data is ready to go once the copy is done
How to read Tweet data to extract geometry from it

• If the data is in standard known formats users don’t need to write any code for reading the records and extracting geometry from it

• Standard formats supported
  – JSON, ShapeFiles

• For others, users need to provide custom custom RecordInfoReader class
  – We will show an example of this using JSON format
RecordInfoProvider Example

```java
public JGeometry getGeometry() {
    //returns the geometry of the record
    return JsonUtils.readGeometry(
        recordNode.get("geometry"), //the GeoJSON geometry
        2, //the dimensions of the geometry
        8307 //the SRID of the geometry
    )
}

@Override
public boolean getExtraFields(Map<String, String> extraFields) {
    //Find the followers count
    String followersCount = recordNode.get("properties").get("followers_count").getValueAsText();
    //and add it to the extra fields that will be available when performing an analysis
    extraFields.put("followers_count", followersCount);

    //return true since we added an extra field
    return true;
}
```
Setup the custom hierarchy using JSON files

```java
/**
 * Subclass of DynaAdminHierarchyInfo that setups the hierarchy composed of the layers
 * eurozone_countries and eurozone_provinces.
 *
 */

public class EuroHierarchyInfo extends DynaAdminHierarchyInfo {

    public EuroHierarchyInfo()
    {
        super();
        addLevel(
            1, //hierarchy level number
            "eurozone_countries", //hierarchy level name
            "id", //path to the JSON field used to associate entries of this level with their children.
            null
        );

        addLevel(
            2, //hierarchy level number
            "eurozone_provinces", //hierarchy level name
            null,
            "properties.ISO" //path to the JSON field used to associate entries of this level with their parent.
        );
    }
}
```
Example JSON data for custom hierarchy

Example of record in the file eurozone_countries.json:
{"type":"Feature","_id":"FRA","geometry":{"type":"Polygon","coordinates":
[[1.44136,42.60366,1.47851,42.65168,...,1.44136,42.60366]],"properties":
{"Continent":"EU","Name":"France","Alt_Region":"EMEA","Country Code":"FRA"},"label_box":[-1.12061,45.13915,6.02255,49.19591]}

And example of a record representing a France province in eurozone_provinces.json:
{"type":"Feature","_id":"3023519","geometry":{"type":"Polygon","coordinates":
[[9.19977,41.36465,9.25876,...,41.36465]],"properties":
{"Country":"France","ISO":"FRA","State Province Name":"Corse","Country Code_State Province Name":"FRA_Corse","Country Name_State Province Name":"France_Corse"},"label_box":[8.70974,42.08453,9.53075,42.54732]}
Invoke the Java classes to run the categorization job

```java
hadoop jar $API_LIB_DIR/sdohadoop-vector.jar
  oracle.spatial.hadoop.vector.mapred.job.Categorization -libjars $HADOOP_LIB_JARS
  spatialOperation=IsInside
  input=/user/oracle/HOL/tweets.json
  output=/user/oracle/HOL/euroZoneCatOutput
  inputFormat=oracle.spatial.hadoop.vector.geojson.mapred.GeoJsonInputFormat
  recordInfoProvider=oracle.spatial.hadoop.vector.geojson.GeoJsonRecordInfoProvider
  srid=8307
gеodetic=true
tolerance=0.5
  hierarchyInfo=hol.EuroHierarchyInfo
  hierarchyIndex=/user/oracle/HOL/hierarchyIndex
```

Confidential – Oracle Internal/Restricted/Highly Restricted
Simple HTML5 application to show the results on a map

```javascript
map.addLayer(tileLayer);
var cs = new OM.style.Color({strokeThickness: 1, stroke: "#009400",
                              fill: "#5c85ad", fillOpacity: 0.75});
var layer = new OM.layer.VectorLayer("datapack_layer",
                                      {def: {type: OM.layer.VectorLayer.TYPE_DATAPACK,
                                              url: "saved_results.json", labelColumn: "county" },
                                       renderingStyle: cs });
layer.setLabelsVisible(true);
layer.setBoundingTheme(true);
```
Image Processing with feature Extraction -- Python

• Given a set of raster data for elevation, calculate the HillShade from the raster data

• Input: Single band digital elevation model data

• Output: Single band raster image showing the hill shade
Copy the input Raster data to HDFS

Using PYTHON scripts

```python
# ********* Execute the script with a valid yarn user with access ***********

if __name__ == "__main__":
    if len(sys.argv) != 3:
        print "usage: imageloader.py <inputLocalFolder> <relativeHDFSPath> where \n"
        "<inputLocalFolder> is a comma separated string with multiple files or folders with images.\n"
        "<relativeHDFSPath> HDFS relative path of the user executing the job to write the output.\n"
    else:
        try:
            printConf ()
            doLoader (sys.argv[1], sys.argv[2])
            resultList (sys.argv[2])
        except Exception as inst:
            print inst.args
            print inst
```

Copy the input Raster data to HDFS

Using PYTHON scripts

#******************************************************************************
#
# TO-DO Configure these properties as necessary
#
#******************************************************************************

SHARED_DIR = "/net/host100/scratch/abcduser/test/"
INSTALLATION = "/opt/oracle/oracle-spatial-graph/"
ALL_ACCESS_DIR = SHARED_DIR + "write/"
GDAL_LIB = ALL_ACCESS_DIR + "gdal"
GDAL_DATA = SHARED_DIR + "data"
RASTER_INST = INSTALLATION + "spatial/raster/"
LOADER_JAR = RASTER_INST + "jlib/" + "hadoop-imageloader.jar"
OVERLAP = "10"

#Add more hadoop_classpath jars as necessary

CLASSPATH_JARS = ["hadoop-imageprocessor.jar", "hadoop-raster-fwk-api.jar", "gdal.jar"]
Calculate the slope

private double calculateSlope(double[] neighboors, double pixelWidthX, double pixelWidthY) {
    double slope = 0;
    dx = ((neighboors[0] + (2 * neighboors[3]) + neighboors[6]) -
          (neighboors[2] + (2 * neighboors[5]) + neighboors[8])) / (8 * pixelWidthX);
    dy = ((neighboors[6] + (2 * neighboors[7]) + neighboors[8]) -
          (neighboors[0] + (2 * neighboors[1]) + neighboors[2])) / (8 * pixelWidthY);
    slope = Math.sqrt(Math.pow(dx, 2) + Math.pow(dy, 2));
    return slope;
}
Calculate Hillshade

double zenithRadians = 90-altitude * Math.PI / 180;
double azimuthRadians = azimuth_math * Math.PI / 180;
clearNeighborhood(neighborhood);
slopeRadians = Math.atan(slope * zFactor);
aspectRadians = calculateAspectRadians();
hillShade = calculateHillShade(zenithRadians, azimuthRadians, slopeRadians, aspectRadians);
finalFloatData[position] = hillShade;
Executing the analysis process as a map-reduce job

Now that the catalog and mosaic configuration are set, execute the job using the hadoop jar command with the `hadoop-imageprocessor.jar` and all the options described in the last steps and wait until it finishes.

```
```
Differentiators -- Spatial

• Commercial, supported software
• Big Data / Map Reduce approach vs. GIS-relational approach
  – Works on any data that includes location info, datatype and file format
• GeoEnrichment services including global geographic hierarchy
• Very rich set (~50) of spatial operators and functions
• Supports both spatial processing and spatial analytics
• Both Vector and Raster services
• Includes map visualization
Differentiators -- Graph

• Commercial, supported software
• “Best of Both Worlds” Graph DB
  – In-memory graph analysis algorithms
  – Distributed graph database model
• SQL-like Parallel Graph Query Language (PGQL)
• Dozens of pre-built in-memory graph analysis algorithms
• 10-50x faster analytics than competitors’ offerings
• Analyze 20-30 Billion edge graph in memory on a single BDA node
Spatial and Graph at OpenWorld 2016

Subtitle

Oracle Spatial and Graph Product Team
The Spatial & Graph SIG User Community

The SIG connects and exchanges knowledge via online communities and at conferences and events

• Chat & Coffee at OOW
  Join us & chat with other S&G users about your experiences! Bring your own coffee 😊
  Wednesday at 8:00 am
  Meet in front of Oracle Bookstore – Moscone South Lobby

• Join us online tinyurl.com/oraclespatialcommunity

LinkedIn, Google+ & IOUG SIG groups

@oraspatsialsig | oraclespatialsig@gmail.com
BIWA SUMMIT 2017
WITH SPATIAL SUMMIT
+ YESSQL SUMMIT

THE Big Data + Analytics + Spatial + Cloud + IoT + Everything Cool User Conference
January 31 - February 2, 2017

Call for Speakers Now Open!
Visit www.biwasummit.org
Learn More About Oracle’s Spatial and Graph Technologies

- [blogs.oracle.com ➔ oraclespatial ➔ oracle_maps_blog ➔ bigdataspacialgraph](blogs.oracle.com/oraclespatial/oracle_maps_blog/bigdataspacialgraph)
More Resources
Resources on Big Data Spatial and Graph

• Oracle Big Data Spatial and Graph on Oracle.com:  

• OTN product page (white papers, software downloads, documentation, tutorials):  
  www.oracle.com/technetwork/database/database-technologies/bigdata-spatialandgraph

• Oracle Big Data Lite Virtual Machine - a free sandbox to get started:  

• Hands On Lab for Big Data Spatial:  tinyurl.com/BDSG-HOL

• Blog – examples, tips & tricks:  blogs.oracle.com/bigdataspatialgraph

• @OracleBigData, @SpatialHannes, @JeanIhm