Geodaten für Alle!
Mit Oracle Spatial, Oracle Maps und APEX

Carsten Czarski
ORACLE Deutschland B.V. & Co KG
München

Schlüsselworte:
Geodaten, Spatial, Locator, SDO_GEOMETRY, APEX, MapViewer, Oracle MAPS

Einleitung
Dienste wie Google Earth oder Google Maps kennt nahezu jedermann. Und so wird immer häufiger die Frage gestellt, ob ähnliche Dienste auch im eigenen Unternehmen und mit eigenen Datenbeständen möglich sind. Dass man räumliche Daten in der Oracle-Datenbank speichern kann, ist hinreichend bekannt – schließlich gibt es "Spatial" seit Oracle7.
Allerdings bietet die Oracle Plattform noch wesentlich mehr: Nimmt man Komponenten wie *Oracle Maps* zur Kartendarstellung oder *Application Express (APEX)* als Entwickler-Framework hinzu, lassen sich recht schnell "Geschäftsanwendungen mit Karte" erstellen - mit einem Look & Feel, wie man es aus dem Internet gewohnt ist. Der Vortrag gibt einen kurzen Überblick über die Geodatenhaltung in der Datenbank, stellt Oracle Maps zur Kartendarstellung vor und beschreibt, wie das alles in APEX-Anwendungen genutzt werden kann.

Geodaten in der Oracle-Plattform
Die Unterstützung für Geodaten wird meistens mit der *Spatial Option* der Oracle-Datenbank gleichgesetzt. Tatsächlich verteilt sich die Geodaten-Technologie jedoch auf verschiedene Oracle-Produkte.

Oracle Locator
Der *Oracle Locator* ist Teil aller Datenbankeditionen und enthält die grundliegenden Funktionen zum Umgang mit Geodaten; dazu gehören unter anderem Speichern, Indizieren und Abrufen von Geodaten, räumliche Abfragen, einige PL/SQL-Funktionen des Pakets SDO_GEOM oder die Transformation von Koordinaten in ein anderes Koordinatensystem.

Oracle Spatial Option
Die Oracle Spatial Option muss separat lizenziert werden und enthält die über den Locator hinausgehende Geodaten-Funktionalität. Eine vollständige Aufzählung würde hier zu weit gehen; als Beispiele seien *Linear Referencing*, *Geocoding* (Umwandeln einer postalischen Adresse in eine Koordinate) oder *Topologie- und Netzwerkdatenmodelle* genannt. Im Handbuch (*Oracle Spatial Developers' Guide*) ist im "Anhang B" eine genaue Aufstellung der Unterschiede zwischen Locator und der Spatial Option enthalten.

Oracle Fusion Middleware MapViewer mit Oracle Maps

Die Oracle Fusion Middleware enthält ebenfalls Komponenten zum Umgang mit Geodaten. Als wichtigste ist hier der *MapViewer* zu nennen: Der MapViewer trägt einen etwas irreführenden Namen, denn es ist ein Kartenserver. Im MapViewer enthalten ist das AJAX-Framework *Oracle Maps*, mit dem intuitive und interaktive Karten á la Google Maps erstellt werden können.

Bereits mit einer *Standard Edition One* der Datenbank kann man demnach bereits Geodaten speichern, verwalten und mit anderen Daten kombinieren. Nimmt man die "kleinste" Edition der Fusion Middleware hinzu, so ist man in der Lage, Karten in seine Anwendung zu integrieren.

Geodaten speichern: SDO_GEOMETRY

Für Geodaten stehen in der Datenbank zwei native Datentypen zur Verfügung: **SDO_GEOMETRY** speichert Vektordaten, während **SDO_GEORASTER** für Rasterdaten, also Luft- oder Satellitenbilder, zuständig ist. Beide Datentypen sind vollständig dokumentiert, so können die Vektorkoordinaten eines SDO_GEOMETRY auf Wunsch im Klartext ausgelesen werden. Zum Erstellen eines SDO_GEOMETRY kann man natürlich GIS-Werkzeuge verwenden, nimmt man die Dokumentation zu Hilfe, kann man es mit Java, SQL oder PL/SQL auch selbst tun. Beide Datentypen können beliebig in Tabellen oder PL/SQL-Logik verwendet werden (Listing 1).

```
CREATE TABLE kunden(
  id       NUMBER(10),
  name     VARCHAR2(200),
  umsatz   NUMBER(15,2),
  lokation SDO_GEOMETRY
);

INSERT INTO kunden values (
  1,
  'Max Mustermann',
  SDO_GEOMETRY(2001, 4326, SDO_POINT_TYPE(10, 50, null), null, null)
);
```

Listing 1: Erstellen einer Tabelle mit Vektordaten-Spalte und Speicherung einer Zeile mit einem Punkt

Räumliche Daten nutzen: Auswertungen und Analysen

Werden Abfragen auf die gespeicherten Geodaten getätigt, kann die Oracle-Datenbank ihre Stärken voll ausspielen – denn die Geodaten werden gemeinsam mit den anderen Tabellenspalten in den gleichen Tablespaces und damit in der gleichen Datenbank abgelegt. Kombinierte Abfragen sind somit kein Problem. Die Abfrage in Listing 2 stellt anhand der Tabellen VERTRIEBSGEBIETE und KUNDEN fest, welche Kunden im einem Vertriebsgebiet mehr als 1 Mio. € Umsatz gemacht haben (die Daten aus den Tabellen VERTRIEBSGEBIETE und KUNDEN werden mit SDO_RELATE räumlich zusammengeführt).

```
SELECT
  k.name,
  k.umsatz
FROM kunden k, vertriebsgebiete vg
WHERE SDO_RELATE(k.lokation, vg.grenze, 'mask=anyinteract') = 'TRUE'
AND vg.name = 'V01' AND k.umsatz > 1000000
```

Listing 2: Abfrage mit räumlichen und fachlichen Kriterien

SDO_RELATE testet verschiedene topologische Beziehungen zwischen Geometrien; dabei wird das *Nine-Intersection Pattern* unterstützt. So kann unter anderem getestet werden, ob Geometrien sich berühren (*touch*), ob sie deckungsgleich sind (*equal*) oder ob sich eine innerhalb einer anderen befindet (*inside*). Abbildung 1 zeigt die möglichen topologischen Beziehungen.

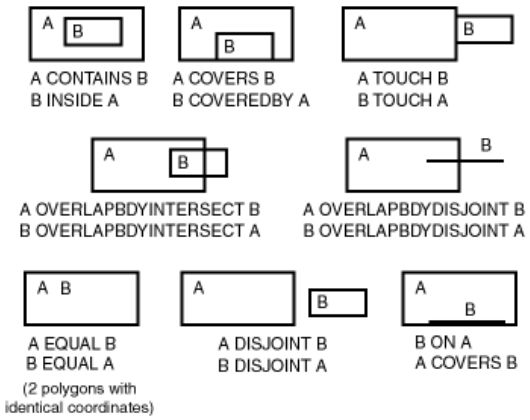

Abbildung 1: Nine Intersection Pattern

Die Existenz eines räumlichen Index (Spatial-Index) ist entscheidend für die Ausführung einer solchen Abfrage – diese werden in der Oracle-Datenbank als *R-Tree Indizes* erzeugt. Neben den SQL-Operatoren stehen im PL/SQL Paket SDO_GEOM eine Reihe Funktionen für unterschiedlichste geometrische Aufgaben zur Verfügung. Einige Beispiele hierfür sind:

- Funktionen zum Verschneiden von Geometrien:
  SDO_UNION, SDO_INTERSECTION, SDO_XOR
- Ermitteln des geometrischen Schwerpunkts einer Geometrie:
  SDO_CENTROID
- Berechnungen mit Geometrien:
  SDO_AREA, SDO_DISTANCE, SDO_LENGTH, SDO_VOLUME
- Werkzeugfunktionen:
  VALIDATE_GEOMETRY_WITH_CONTEXT

Visualisierung der Geodaten als Karte: Oracle MapViewer

Die Anforderungen an Karten in Web-Anwendungen haben sich in den letzten Jahren massiv geändert. Endanwender erwarten intuitive, leicht bedienbare Oberflächen – Dienste wie Google Maps haben quasi den Standard gesetzt. Solche Anwendungen können mit der Oracle-Technologie auch mit eigenen Daten im Inter- oder Intranet bereitgestellt werden. Oracle Maps als Teil des *Oracle Fusion Middleware MapViewer* übernimmt dabei das Erstellen der Karte.

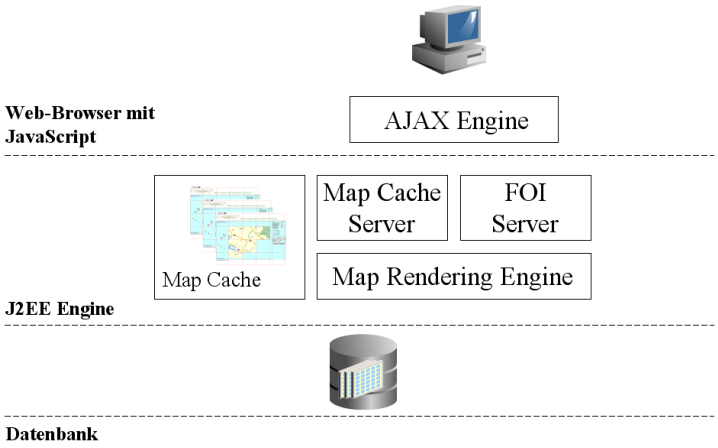

Abbildung 2: Oracle Maps-Architektur

Abbildung 2 stellt die Architektur des MapViewer dar. Kernstück ist die *Map Rendering Engine* – sie generiert anhand der Daten in der Datenbank die Karte als Bild. Dazu muss die Engine natürlich "wissen", wie die Daten in der Karte dargestellt werden sollen. Diese Definitionen werden mit dem gemeinsam mit dem MapViewer ausgelieferten Werkzeug *Oracle MapBuilder* (Abbildung 3) erstellt und in den Datenbank-Tabellen **USER_SDO_STYLES**, **USER_SDO_THEMES** und **USER_SDO_MAPS** gespeichert.

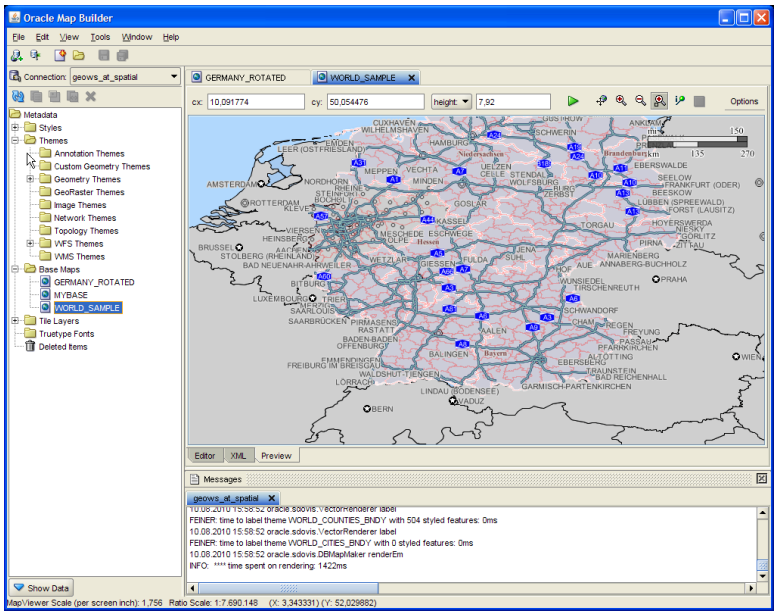

Abbildung 3: Kartendefinitionen werden im Oracle MapBuilder erstellt und in der Datenbank gespeichert

Oracle Maps stellt diese Karten schließlich intuitiv und anwenderfreundlich dar. Oracle Maps läuft auf eigenen Unternehmensdaten in der Datenbank der eigenen IT-Infrastruktur - es ist Software, kein Dienst.
Der "Maps-Ansatz" basiert darauf, dass dem Nutzer standardisierte Karten bereitgestellt werden. Der geforderte Kartenausschnitt wird nicht als ein Bild generiert, sondern in mehrere Kacheln (*Tiles*) unterteilt. Im Browser werden diese Kacheln dann zu einer Karte zusammengesetzt. Zusätzlich werden die Kacheln im *Map Cache* abgelegt: Werden die gleichen Kacheln später nochmals benötigt, können Sie aus dem nach Koordinaten und Zoom-Level aufgebauten Cache geholt werden. Und da das Abrufen einer Kachel aus dem Dateisystem wesentlich schneller ist als das erneute Generieren derselben, werden die Karten für den Endanwender sehr performant und flüssig dargestellt..

Dynamische Informationen: Features Of Interest
Die Kacheln der Karte allein dürften allerdings nur in den seltensten Fällen ausreichen; der Nutzen einer Karte ergibt sich aus den *dynamisch dargestellten* fachlichen Informationen. Und dabei möchten unterschiedliche Anwender auch unterschiedliche Daten sehen. Hierfür stehen die *Features Of Interest (FOI)* zur Verfügung. FOI werden vom Kartenserver "on-the-fly" mit einer räumlichen Abfrage aus der Datenbank abgerufen, an den Browser übermittelt und von diesem in der Karte dargestellt - Caching findet nicht statt. Jedes FOI enthält automatisch ein *Information Window*, welches bei Klick weitere Details anzeigt.
Möglich wird das durch die Verwendung von JavaScript und der AJAX-Technologie – insofern werden zur Entwicklung einer Oracle-Maps-Anwendung auch entsprechende Kenntnisse benötigt. Die Navigation in der Karte, das Ein- und Ausblenden von Informationen wird nicht mehr in "klassischer" Manier in Java, PHP oder .NET auf dem Server, sondern in JavaScript auf dem Browser implementiert.

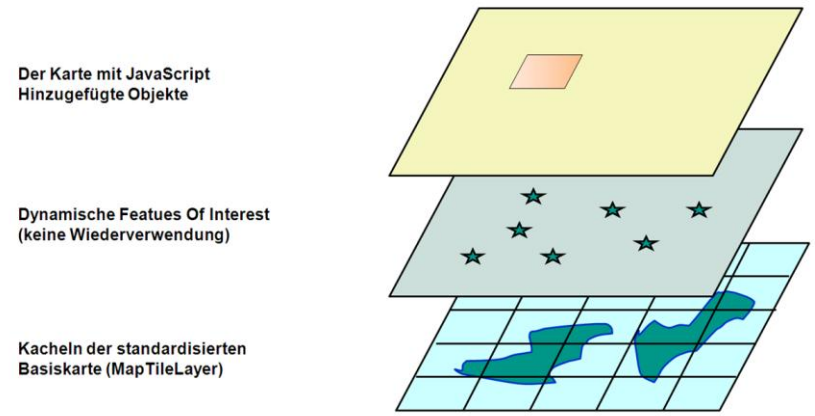

Abbildung 4: Aufbau einer Karte in Oracle Maps

Alles zusammen in einer Anwendung: Oracle Spatial, Oracle Maps und APEX

Damit das alles für den Endanwender nutzbar wird, müssen Geodaten und Karten in einen Anwendungskontext eingebunden werden. Dies sei am Beispiel von Application Express (APEX) dargestellt. APEX-Anwendungen laufen wie APEX selbst, vollständig in der Datenbank ab. Somit ist es ein Leichtes, mit den vorne angesprochenen räumlichen SQL-Funktionen einen APEX-Bericht (Abbildung 5) aufzubauen und Informationen so nach räumlichen Kriterien auszuwerten. Da alle räumlichen Funktionen als SQL-Funktionen oder PL/SQL-Packages bereitstehen, können Sie in den APEX-Komponenten frei genutzt werden.

Abbildung 5: APEX-Bericht mit räumlichen Funktionen SDO_DISTANCE und SDO_WITHIN_DISTANCE

Das Einbinden der Karte in die APEX-Anwendungsseite geschieht mit HTML- und JavaScript-Code. Es wird davon ausgegangen, dass der MapViewer-Server korrekt installiert und konfiguriert ist, und dass die Kartendefinitionen bereits vorliegen. Der Entwickler muss nun sicherstellen, dass die folgenden vier Schritte in der APEX-Anwendungsseite durchgeführt werden.

1. Die JavaScript-Bibliothek für Oracle Maps muss vom Server geladen werden.
2. Eine JavaScript-Funktion zur Initialisierung, Konfiguration und Darstellung der Karte muss enthalten sein.
3. Wenn die Seite vollständig im Browser geladen ist (*onLoad*-Ereignis), muss die JavaScript-Funktion aufgerufen werden.
4. Die Seite muss einen benannten "DIV-Container" als Platzhalter für die Karte enthalten.

Das Laden der Javascript-Bibliothek geschieht mit einem <SCRIPT>-Tag im **APEX-Seiten-Header** (Abbildung 6).

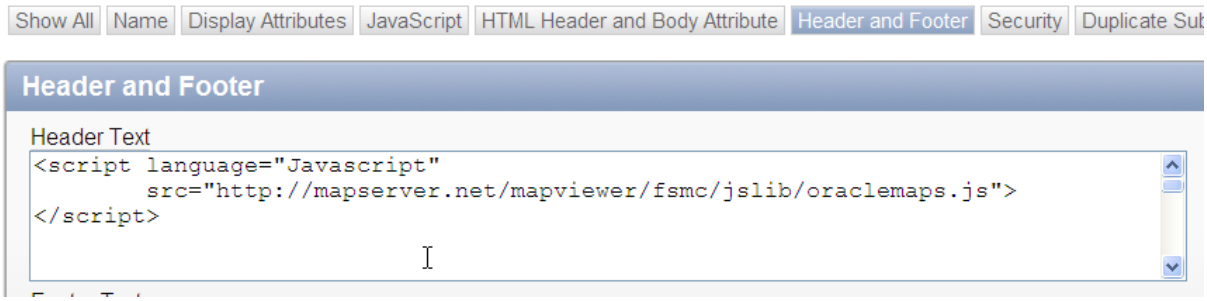

Abbildung 6: Die Oracle Maps JavaScript-Bibliothek wird geladen

Ebenfalls in den **Seiten-Header** platziert man die als zweiten Punkt genannte JavaScript-Funktion **display_map**, die in Listing 3 dargestellt ist. Das darin erzeugte Objekt der Klasse **MVMapView** repräsentiert die Karte. Diese besteht hier aus einer *Base Map*, wird mit einem Kartenzentrum, einem Zoom-Level und einer Navigationsleiste konfiguriert und mit der Funktion **display()** schließlich dargestellt.

```
<script language="JavaScript">
  var baseURL  = "http://mapserver.net/mapviewer";
  var mapview;

  function display_map() {
      mapview = new MVMapView(
        document.getElementById("map"),
        baseURL
      );
      mapview.addBaseMapLayer(new MVBaseMap("GEOWS.ELOC_NAVTEQ_MAP"));
      mapview.setCenter(MVSdoGeometry.createPoint(10, 50, 8307));
      mapview.setZoomLevel(1);
      mapview.addNavigationPanel("east");
      mapview.display();
  }
</script>
```

Listing 3: JavaScript-Funktion display_map() zur Darstellung der Oracle Maps-Karte in APEX

Wichtig ist, dass diese Funktion erst aufgerufen wird, wenn die Anwendungsseite vollständig vom Browser geladen wurde. Dazu dient das *onLoad*-Ereignis, das im HTML Body-Tag festgelegt wird.

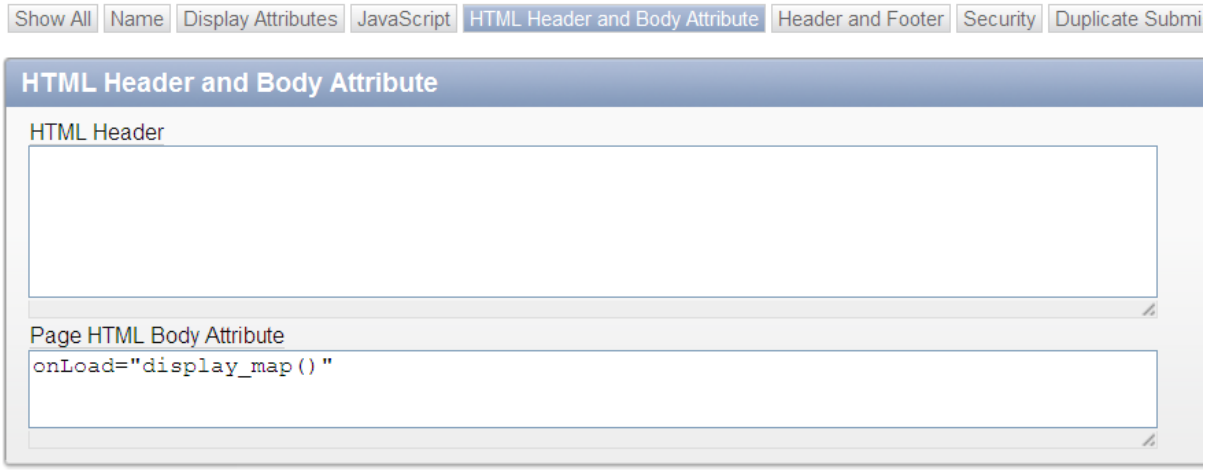

Abbildung 7: Aufruf der JavaScript-Funktion im onLoad-Handler des HTML-Body Tags

Und schließlich muss der Platzhalter für die Karte, der in der JavaScript-Funktion mit **getElementById("map")** angesprochen wird, auch vorhanden sein. Das geschieht durch Erzeugen einer **APEX-HTML-Region** mit dem in Listing 4 dargestellten HTML-Code.

```
<div id="map" style="width: 600px; height: 400px;"></div>
```

Listing 4: Im HTML-Code für den Karten-Platzhalter wird die Größe der Karte definiert.

Das Ergebnis kann dann etwa wie in Abbildung 8 aussehen.

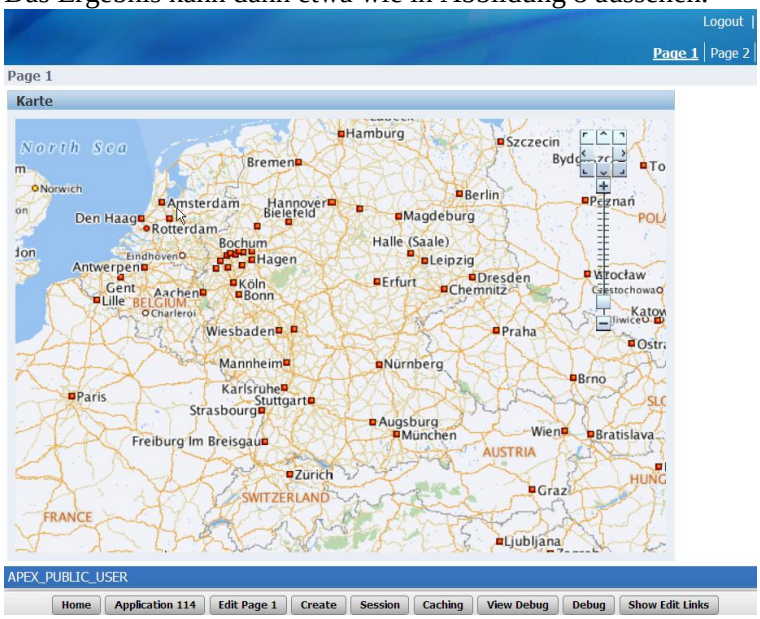

Abbildung 8: Die Oracle Maps-Karte als Teil der APEX-Anwendung

Die dynamisch dargestellten *Features Of Interest (FOI)* werden ebenfalls mit Hilfe von JavaScript-Funktionen eingebunden. Jedes mit dem *Oracle MapBuilder* erstellte *Theme* kann als FOI auf die Karte gelegt werden. Listing 5 zeigt den JavaScript-Code, der in die bereits vorhandene Funktion **display_map** eingebaut wird, allerdings vor dem finalen Aufruf von **mapview.display()**. Das FOI wird nur in den Zoom-Levels 4 bis 10 dargestellt und bei Bedarf, also wenn es aus zu vielen Einzelpunkten bestehen würde, als ein einziges Bild generiert (**enableAutoWholeImage()**).

```
:
var foiWahlGew = new MVThemeBasedFOI("foiWahlGew", "geows.GUTE_KUNDEN");
foiWahlGew.enableAutoWholeImage(true);
foiWahlGew.setMinVisibleZoomLevel(4);
foiWahlGew.setMaxVisibleZoomLevel(10);
mapview.addThemeBasedFOI(foiWahlGew);
:
```

Listing 5: Features Of Interest werden stets neu aus der Datenbank gelesen

Die JavaScript Bibliothek von Oracle Maps ist sehr umfangreich. Neben dem einfachen Darstellen von statischen Kacheln und dynamischen FOI können mit der *Redlining*-Funktion auch neue Objekte erstellt werden. Programmiert man noch etwas JavaScript-Code dazu, so lassen sich diese Objekte wiederum in Datenbanktabellen speichern. Neben vielen anderen sind auch Funktionen zur Integration von externen Diensten wie Google Maps (**MVGoogleMapTileLayer**) oder Microsoft Bing Maps als Hintergrundkarte vorhanden. Mit dem Kartenserver wird stets ein Tutorial installiert, welches die Möglichkeiten von Oracle Maps vorstellt. Es ist unter der URL **http://{host}:{port}/mapviewer/fsmc/tutorial/index.html** erreichbar.

Ausblick: Oracle MAPS-Plugin in APEX 4.0

Die jüngste APEX-Version 4.0 zeichnet sich unter anderem durch neue Plugin-Technologie aus. Die hier vorgestellte Vorgehensweise ist für jede MapViewer-Umgebung und jede Karte nahezu identisch. Insofern kann ein Plugin erstellt werden, welches die Schritte anhand weniger Parameter automatisch durchführt. Die Abbildungen 9 und 10 zeigen den Parameter-Dialog und das Ergebnis des Plugins, das von **http://apex.oracle.com/url/apxmap** heruntergeladen werden kann.

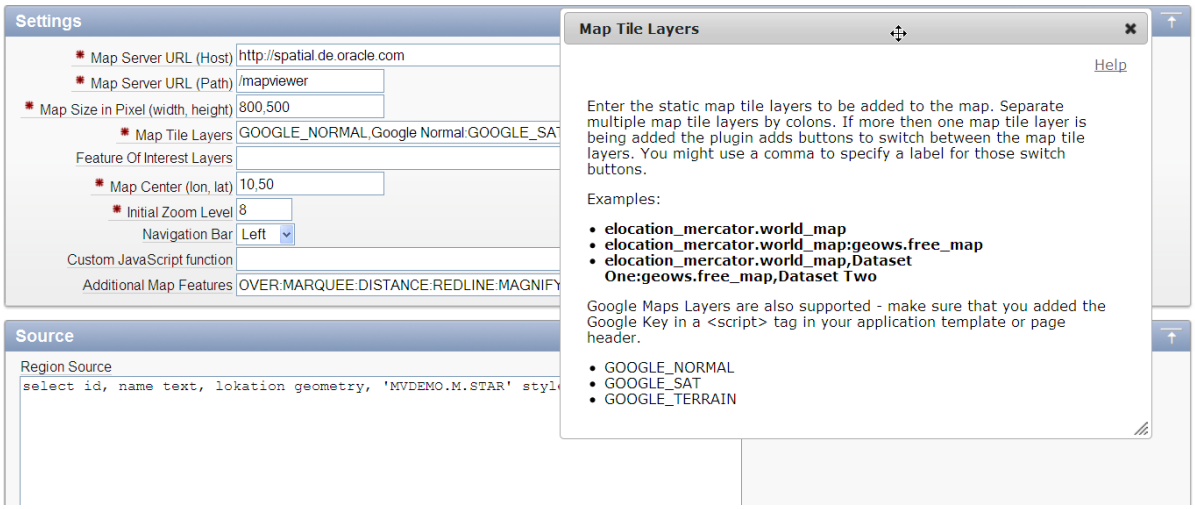

Abbildung 9: Das APEX-Plugin "Oracle Maps" wird eingerichtet ...

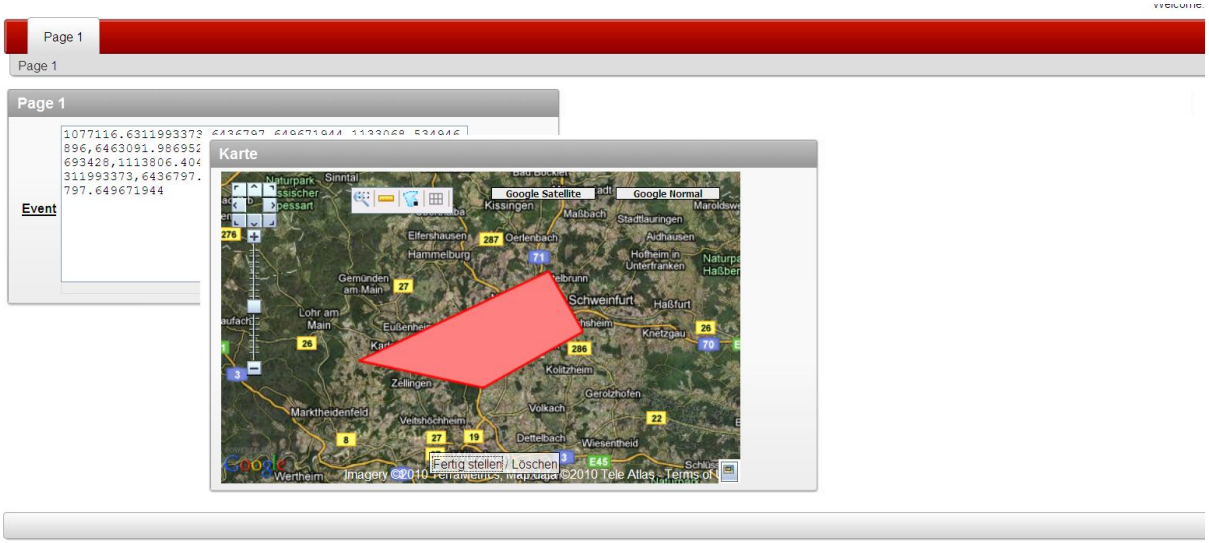

Abbildung 10: ... und bietet neben der reinen Karte auch zusätzliche Werkzeuge (bspw. Redlining) an.

Fazit

Mit dem Kartenserver Oracle MapViewer und Oracle Maps können Geodaten, aus der eigenen Datenbank, dem Endanwender genauso intuitiv präsentiert werden, wie dieser es vom Internet her kennt. Kombiniert man diese Fähigkeit mit der hohen Entwicklerproduktivität von APEX, so bietet sich die Möglichkeit, Anwendungen nicht nur agil, schnell und flexibel bereitzustellen, sondern diese zusätzlich noch mit Karten auszustatten. Informationen können so noch ansprechender aufbereitet werden.

Weitere Informationen:

Tutorial der deutschen APEX Community: Geodaten und APEX
http://apex.oracle.com/url/apxgeo
Oracle Maps-Plugin für APEX 4.0
http://apex.oracle.com/url/apxmap
Blog des Autors
http://sql-plsql-de.blogspot.com

Kontaktadresse:

Carsten Czarski
ORACLE Deutschland B.V. & Co KG
Riesstr. 25, D-80992 München

Telefon: +49 (0)89 1430 2116
E-Mail : carsten.czarski@oracle.com
Internet: http://www.oracle.com/global/de/community