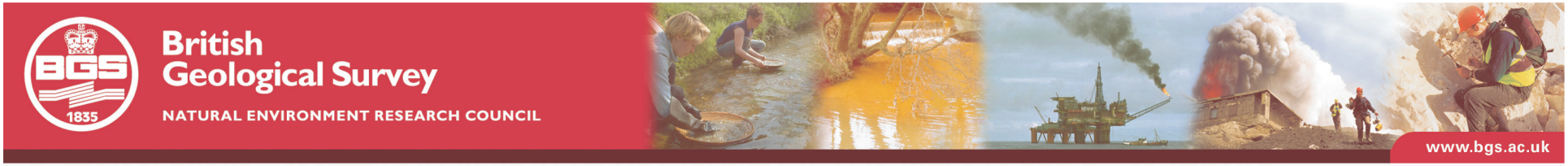


Oracle Spatial
Special Interest Group Presentation

Keith A.M. Adlam & Martin L. Nayembil

Kingsley Dunham Centre
Keyworth
Nottingham NG12 5GG
Tel 0115 936 3100

NATURAL ENVIRONMENT RESEARCH COUNCIL

Scope of talk

- History of Oracle in BGS
- Different data and different applications
- The Spatial Story
- Parallel Paths
- Can Oracle Spatial Help unify RDBMS and GIS?

- Why Oracle spatial
- The role of spatial database technology
- Example Data
- Methods used to spatially enable Oracle tables
- Example code for the various methods
- Mechanisms used to update geometry
- Best of the RDBMS & GIS worlds

History of Oracle in BGS

- Oracle first introduced to BGS in 1983 running under VMS 4.2 on a VAX 8600 computer with 12Mb memory, 2.7 Gb disk space. User Interface was Oracle Forms and SQLPlus

- Initial relational designs were poor. No more than flat lists taken from paper systems (some still exist today)

- A menu driven interface was developed by BGS in the late 80's using ProFortran (MARS).

- The widespread introduction of PC's and Windows in the early 90's led to the use of Microsoft Access as the major user interface used to query Oracle data in BGS

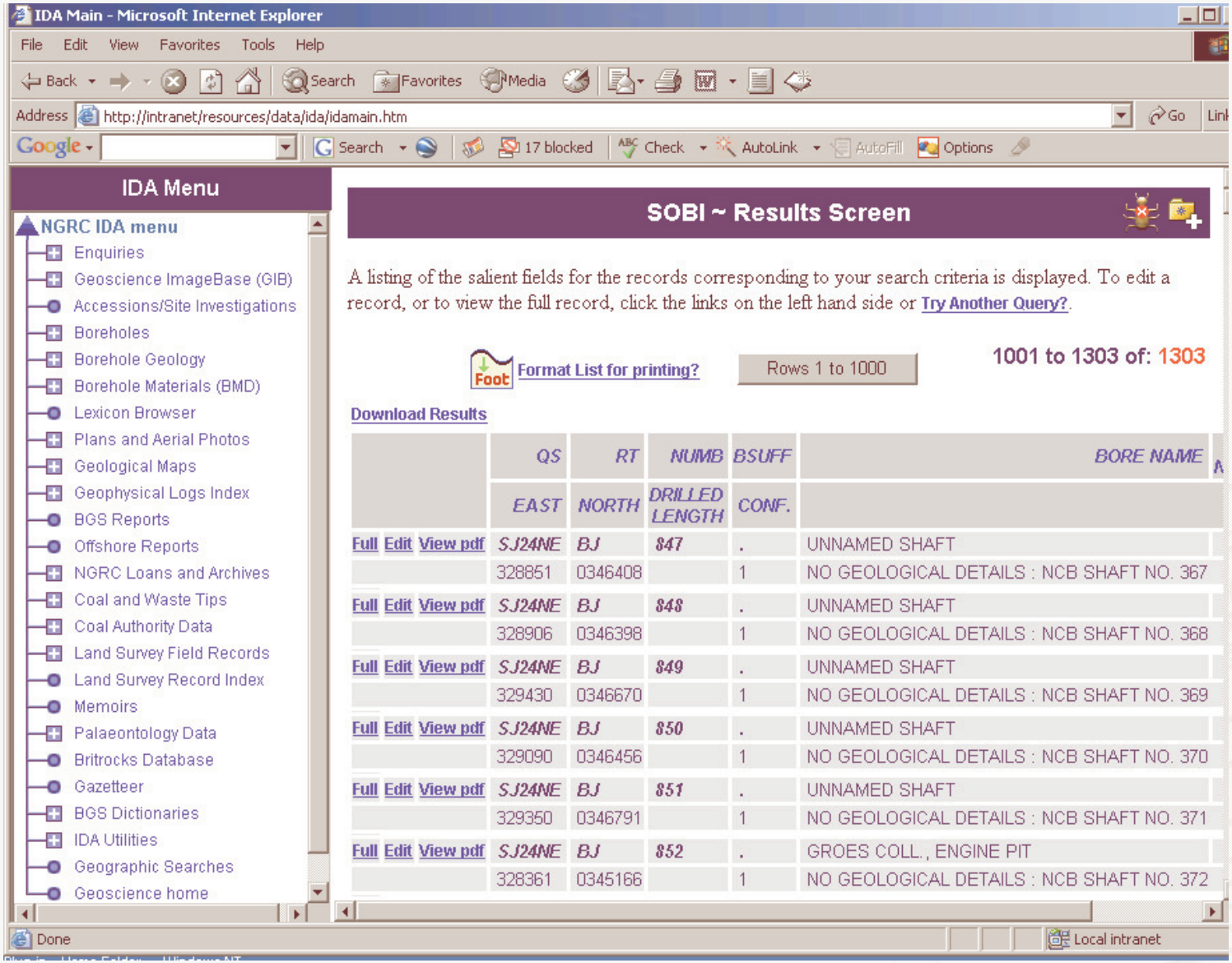
History of Oracle in BGS

- 1999 saw the development of a BGS web based interface to Oracle called Intranet Data Access (IDA). This is still in use today and relies heavily on Cold Fusion to provide Intranet access to our Oracle databases

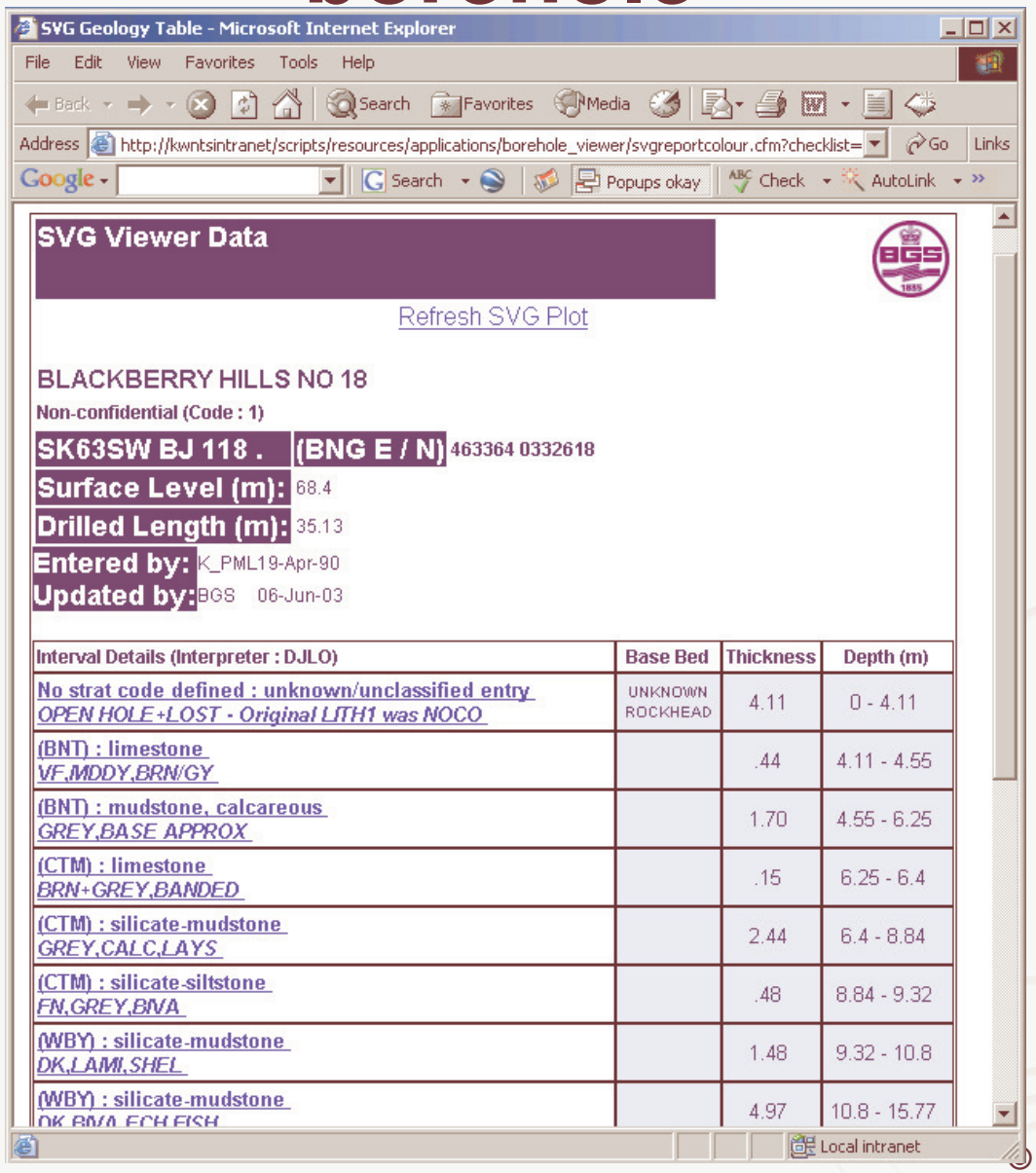
Different data and different applications

- Relational
- GIS (spatial vector data as points,lines and polygons)
- GIS (spatial raster data with or without attributes)
- Surfaces (TINS and GRIDS)
- Images (e.g. borehole, map & photograph scans)
- Time series measurements
- Downhole measurements (depth series)
- 3D models
- etc

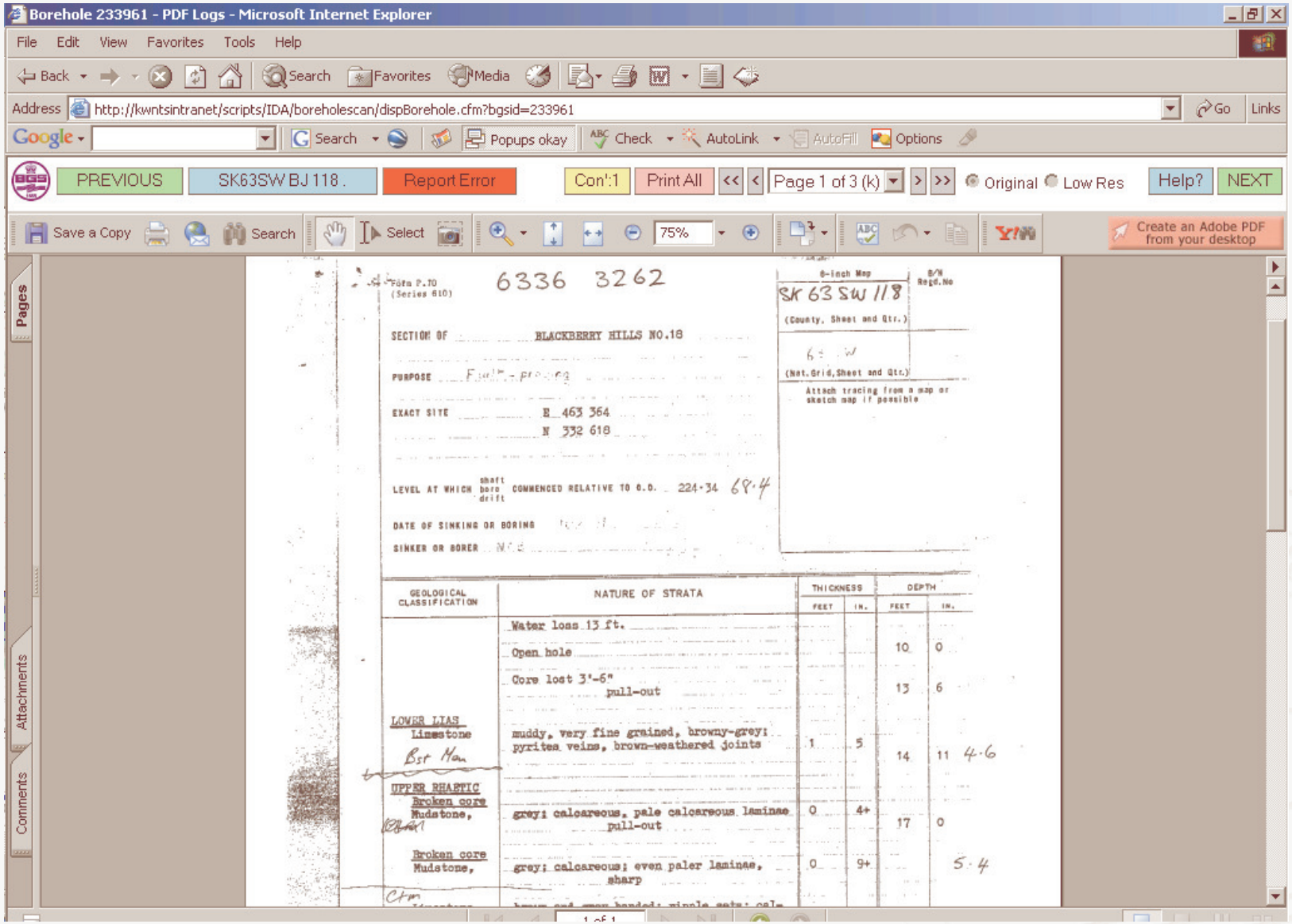
Relational data on a web page (IDA)



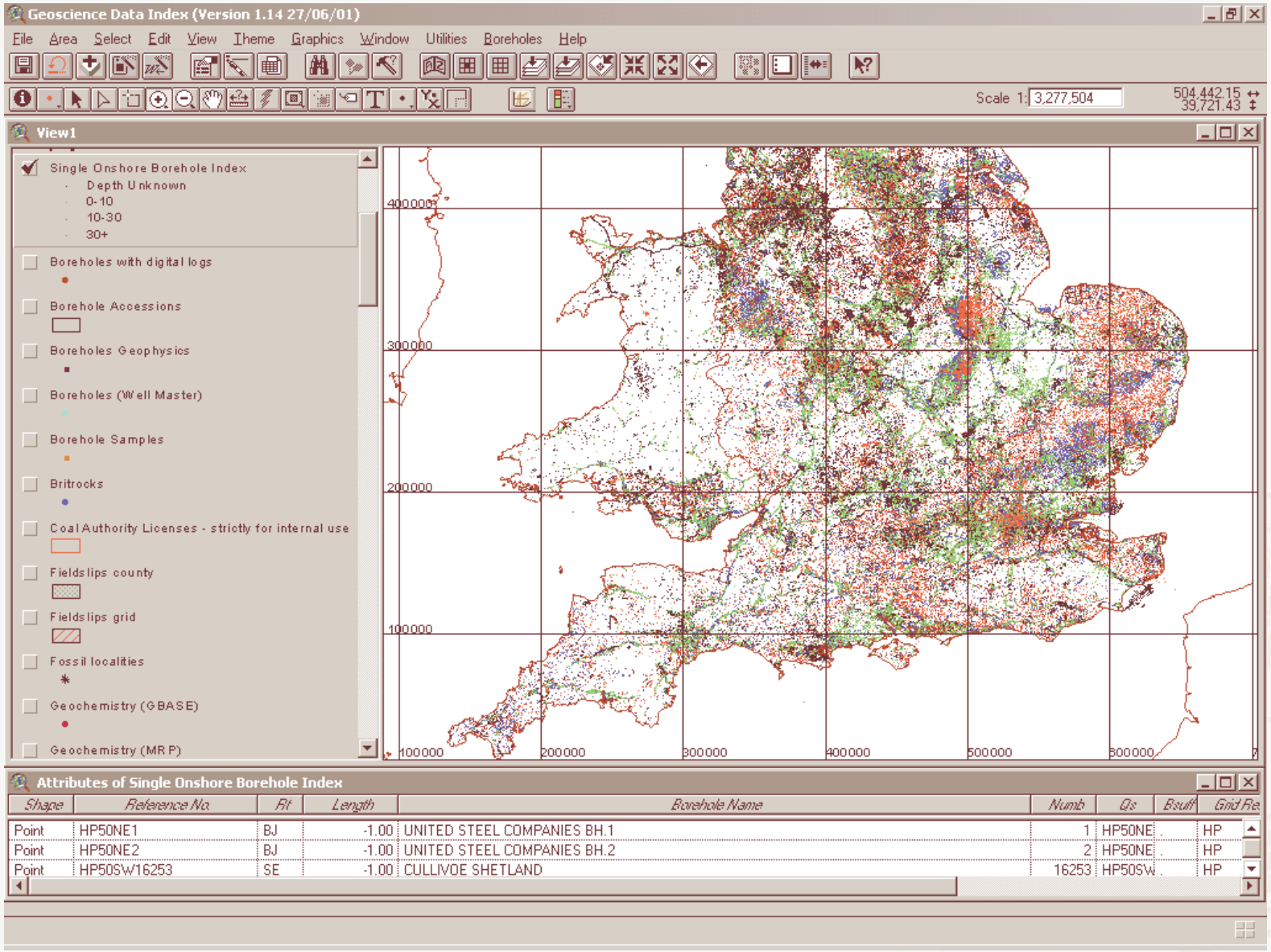
Depth series information from borehole



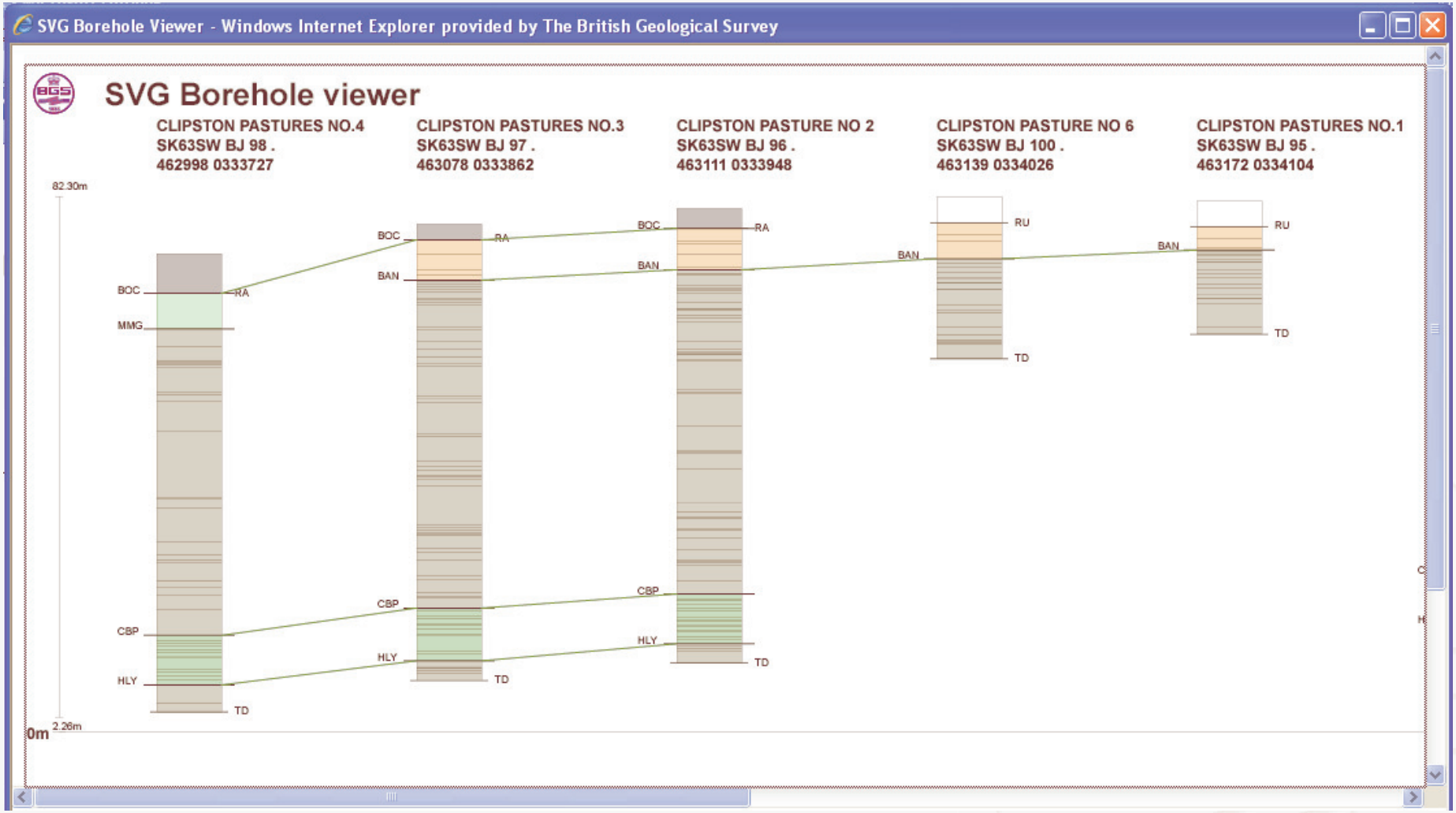
Scan of borehole log



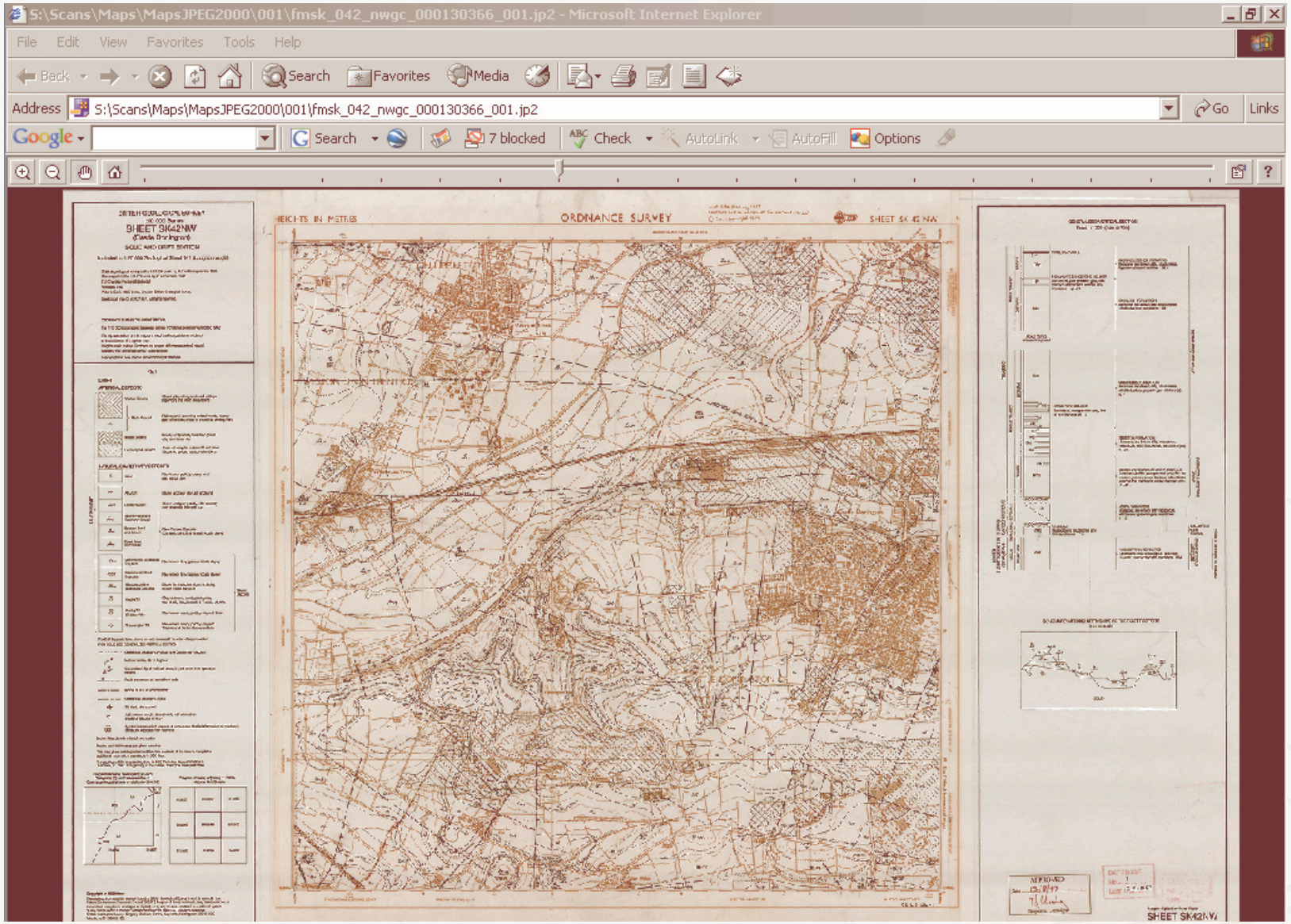
GIS point data in ArcView 3



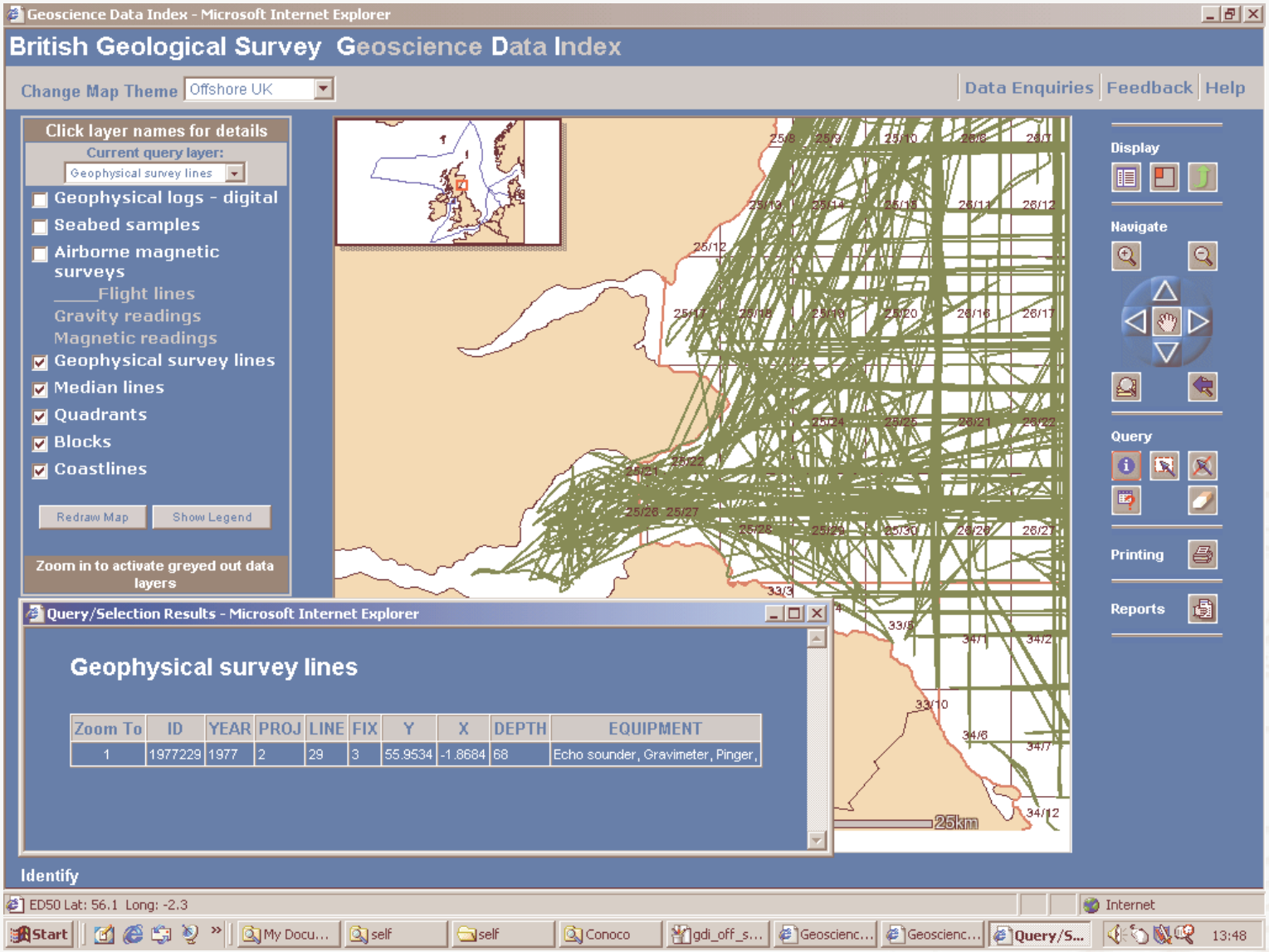
SVG Borehole viewer



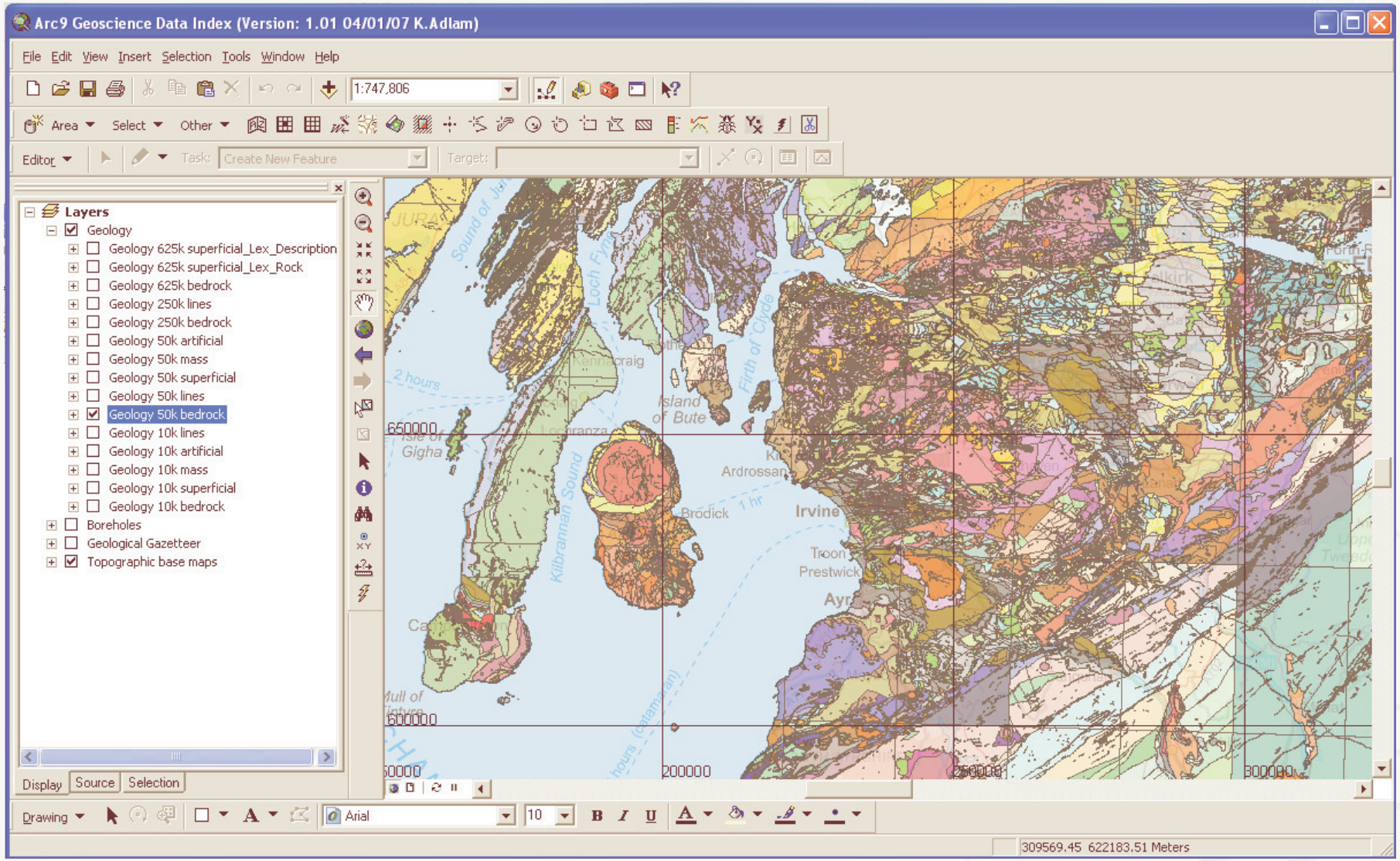
Scanned map viewed in IE



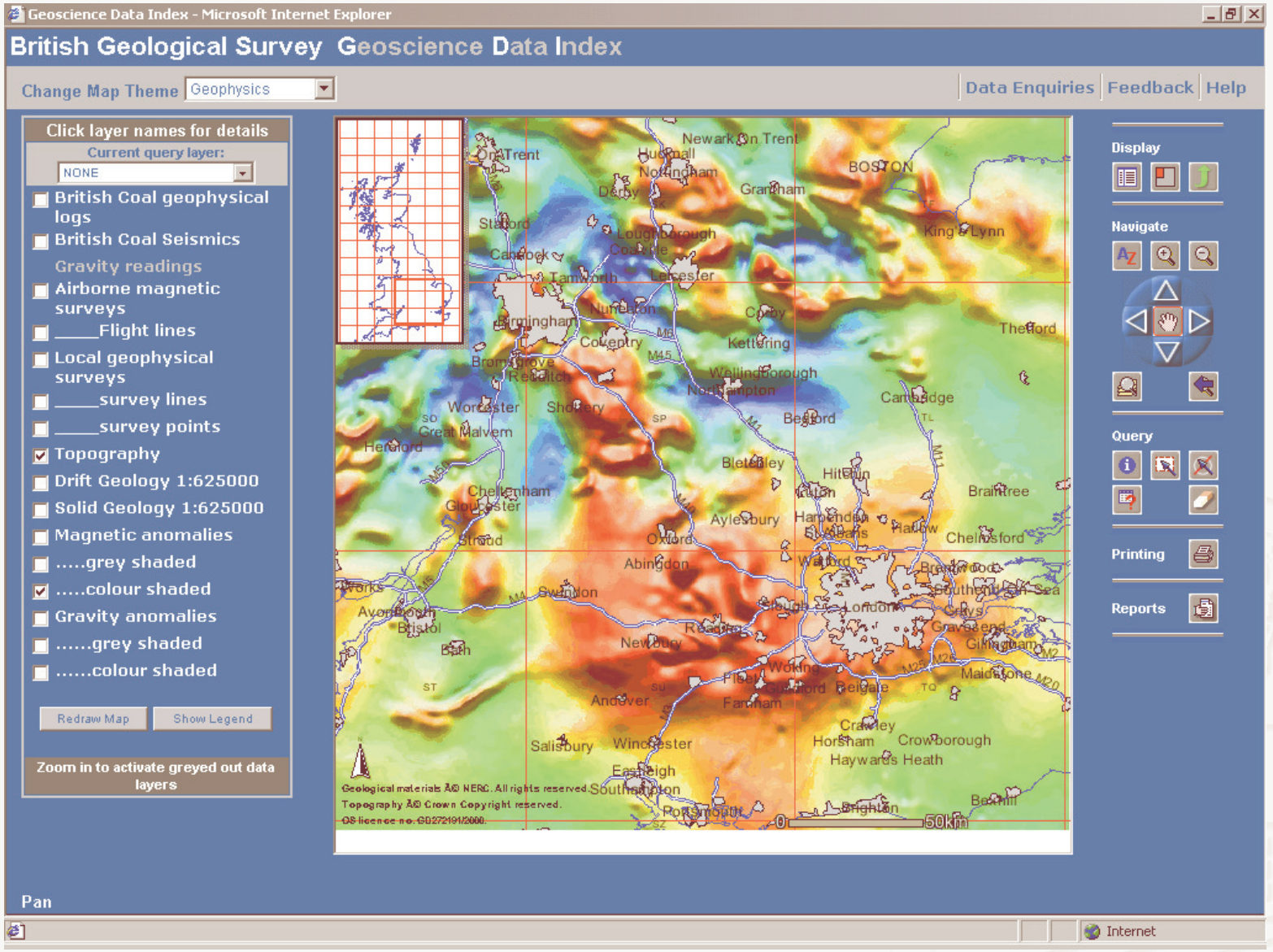
GIS line data displayed in GeoIndex



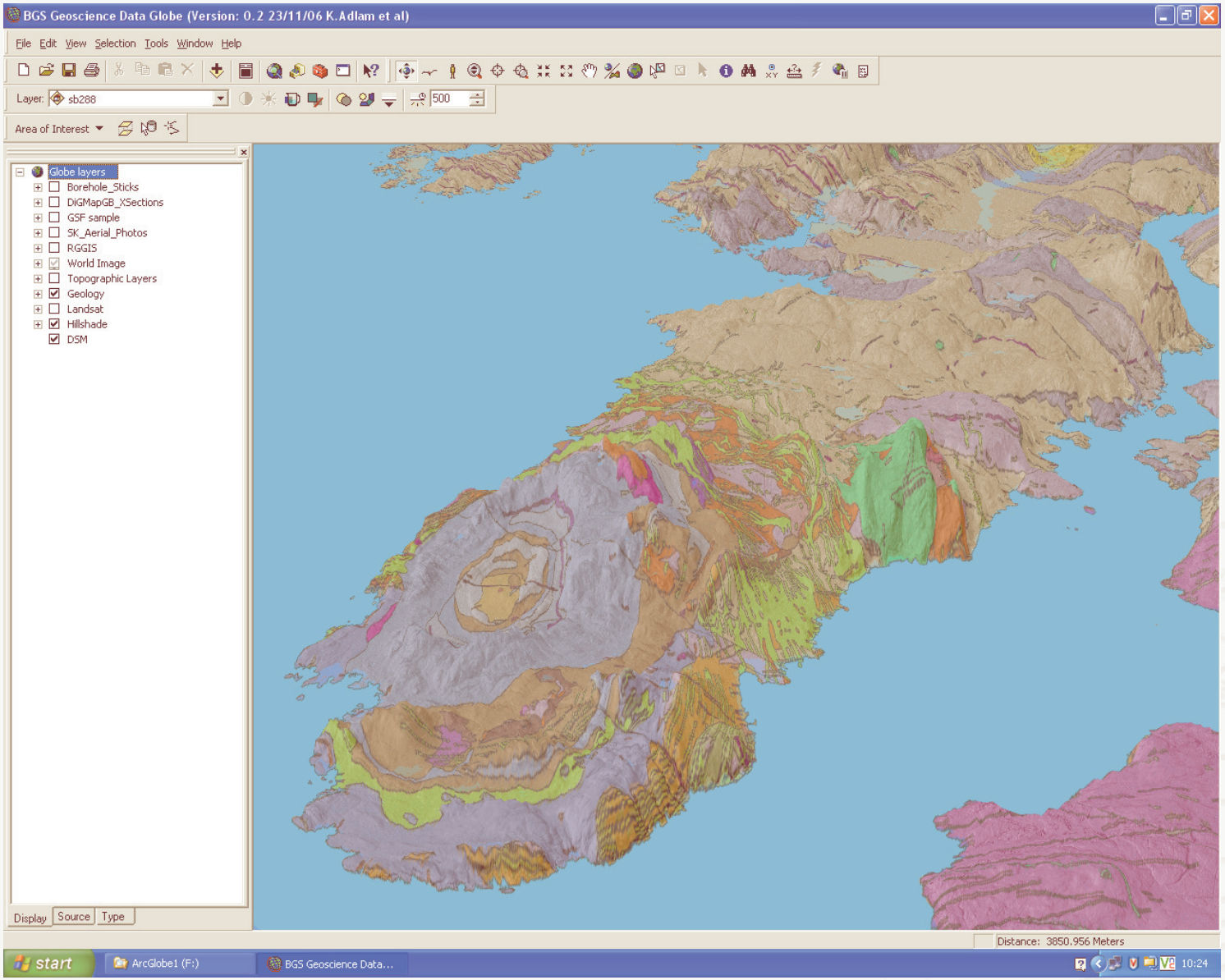
GIS polygon data in ArcMap



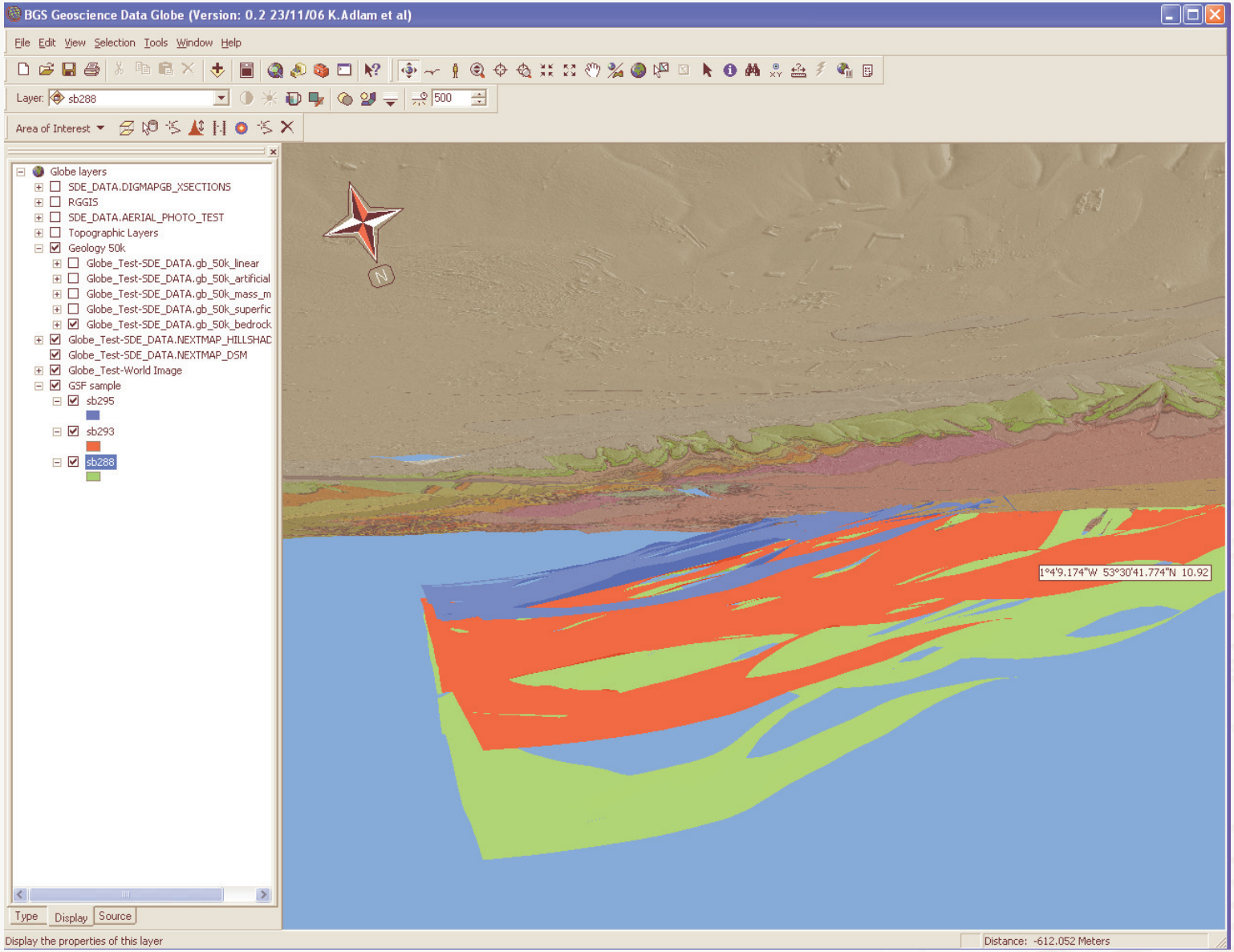
Raster data displayed in GeoIndex



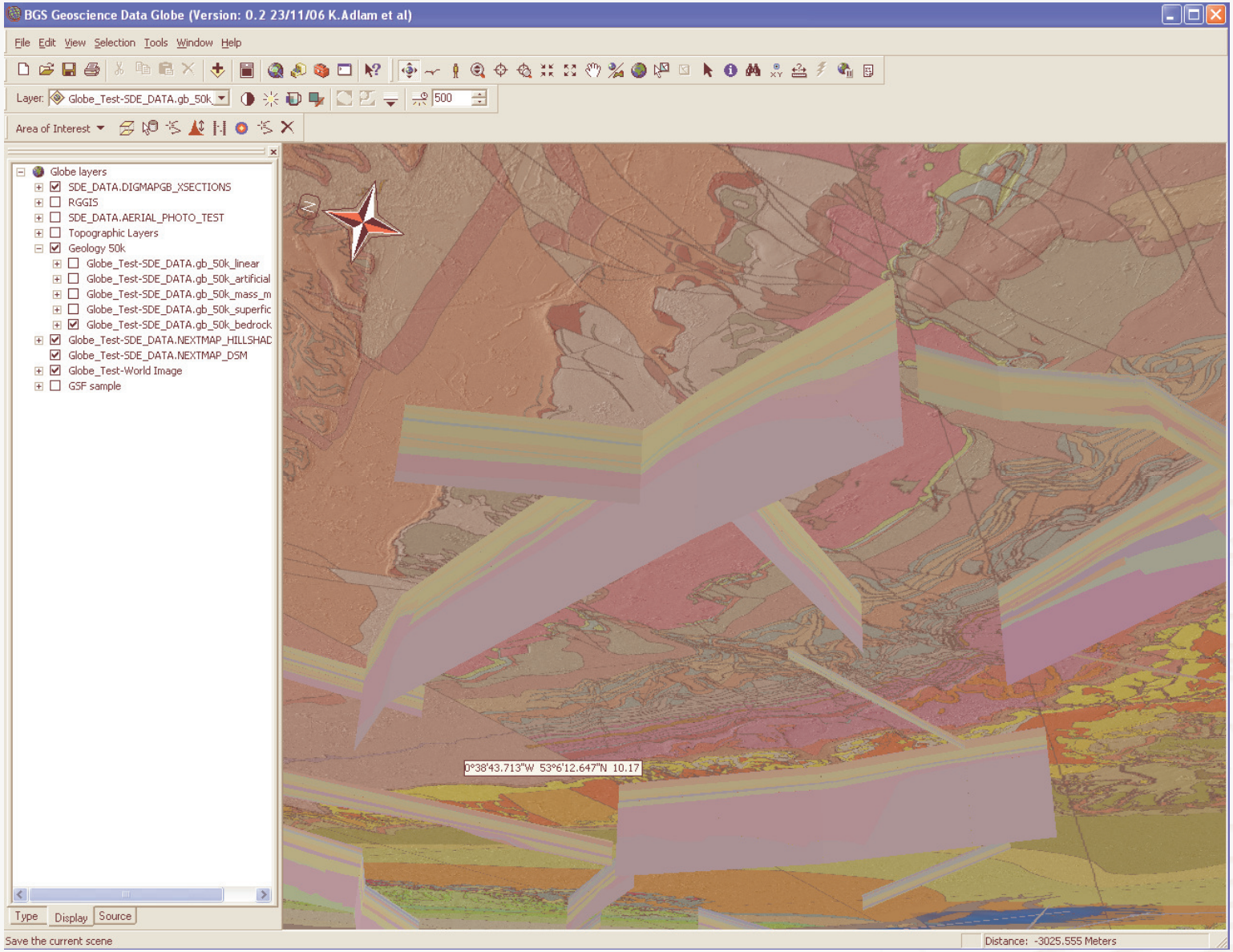
Visualisation in ArcGlobe



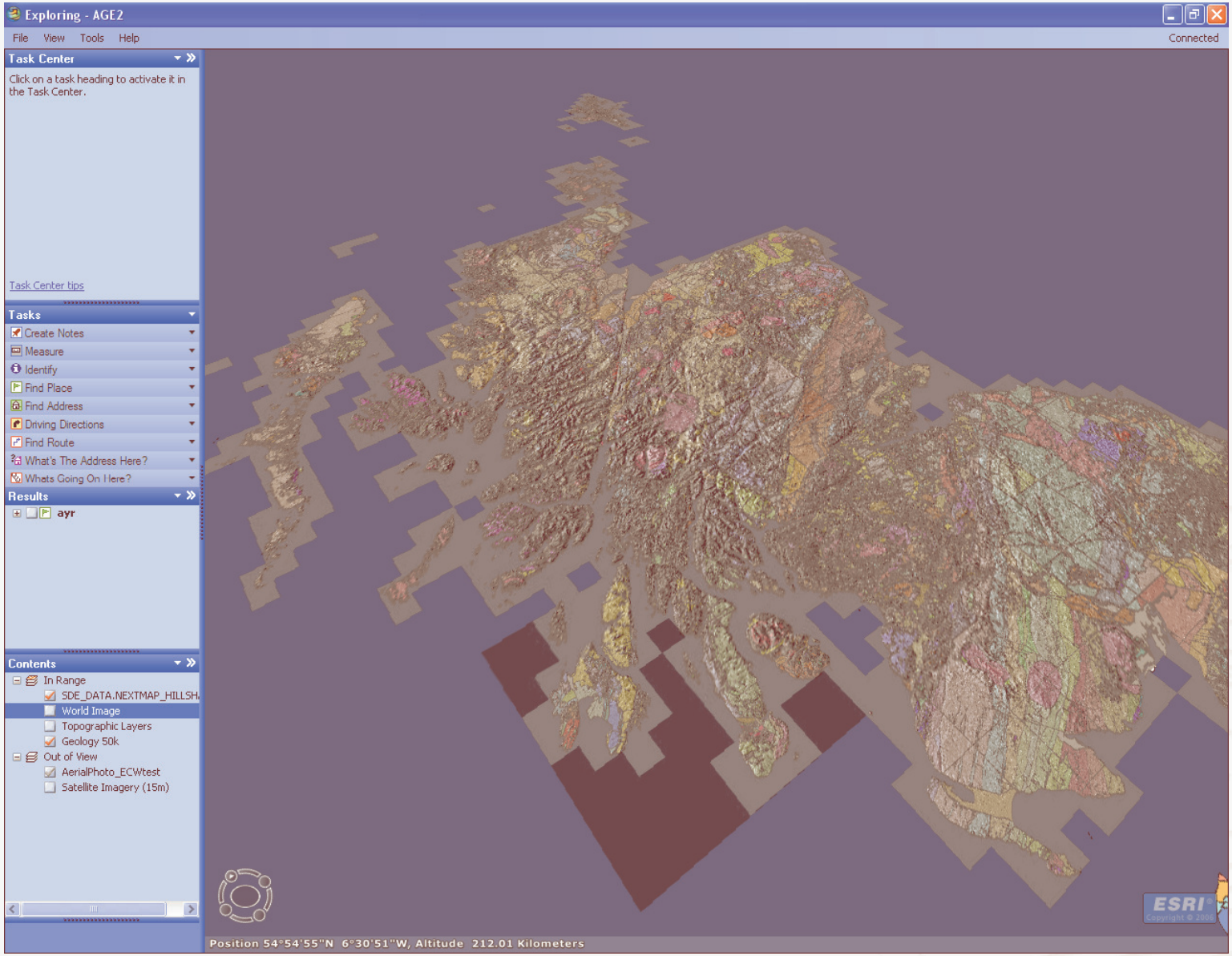
Surfaces displayed in ArcGlobe



Visualisation of cross-sections



Visualisation in ArcGIS Explorer



The Spatial Story

- During the time period that Oracle has been used within BGS for storing relational data, ESRI products have been used to store, manage, analyse and display GIS data

- In 1985 Arc/Info was first used to support the BGS Geoscience Data Index (GDI) This allowed visualisation, spatial query and analysis of many of our important datasets. Seats were limited due to the cost of the Unix workstations at the time

- Introduction of PC's and ArcView 3 in the early 90's led to widespread use of GIS in BGS

The Spatial Story

- In 2000 BGS launched GeoIndex a web based version of the Geoscience Data Index. This is based on ESRI's ARCIMS and allows our customers to visualise and query, at the index level, some of our major dataset over the web.

- In addition to 2D GIS data BGS also holds 2.5D and 3D data in the form of cross-sections borehole data seismic section, stratum contours etc. Much of this data has been modelled using specialist modelling packages

Parallel Paths

- The previous slides have summarised the parallel developments in the use of GIS and RDBMS within BGS
- The two environments have developed side by side for over 20 years yet very little real integration has taken place between these two environments
- Yes, data has been shared between the environments but this has involved translations and copies of data

Why Parallel Paths?

- GIS started life using proprietary file based formats designed to get the best performance possible
- Until the introduction of Spatial Data Option (SDO) Oracle did not support spatial data Only standard data types existed and indexes were limited to 2D B-Trees
- There was little opportunity for GIS and RDBMS developers to come together and create a unified data store to support both relational and spatial data
- The results was and still is two parallel environments, two parallel developments, two parallel sets of rules two parallel sets of data and two separate cultures

Can Oracle Spatial Help unify RDBMS and GIS?

- Oracle Spatial is seen as a way to start to bring the two parallel developments closer together
- To reduce movement of data between systems
- To have one central data repository that can be used to support both GIS and Relational environments
- During the initial phase we are concentrating on spatially enabling existing data held in Oracle excluding GIS data directly managed by ESRI's ArcSDE
- Many of our Oracle tables have a spatial component but this takes a number of different forms

Types of location data in existing Oracle tables

- X-Y coordinates (205936,387654)
- National grid reference (TQ38762768)
- Ordnance Survey tiles (TQ, TQ24NE etc.)
- Two Corner coordinates of rectangular areas
- Four corner coordinates of quadrilateral areas
- Line and polygon data held as ordered rows with x-y coordinates

  ID,SEQ,X,Y

  1,1,365709,456384

  1,2,456890,468345

  1,3,469670,476758

Spatially enabling relational data current production situation

- Important relational datasets are translated to ESRI shapefile format on a monthly basis and held on a central fileserver
- This means that GIS users do not see the current state of the active database
- Small point datasets held in Oracle can be accessed directly from ArcGIS by creating an event theme but this is too slow for larger datasets as no spatial index is used.

Some issues encountered

- When using views that contain a geometry field we have found that they are extremely slow when accessed from other user schemas. It appears that the spatial index is only used if the user is the owner of the view or has MERGE ANY VIEW privilege

- It is possible to register an Oracle view with SDE but not possible to register a view with an ESRI Geodatabase. This means than ESRI domains, sub-types etc cannot be used against views.

- Editing using ESRI clients via views is not possible

- Editing related tables using ArcMap causes problems when constraints are present

ESRI/Oracle Interoperability
What is easy?

- Interoperability at the simple level
- Can easily register Oracle Spatial tables with SDE (unique number, geometry).
- Can copy data held in ESRI formats to Oracle as SDE binary or Oracle Spatial formats
- Easily output the result of ESRI Geoprocessing operations directly to SDE or Oracle Spatial
- Managing data held in Oracle using Oracle clients and viewing the data using ESRI clients is easy
- Managing data held in Oracle using ESRI clients and viewing the data using Oracle clients is easy

ESRI/Oracle Interoperability
What is difficult?

- True integration
- ESRI and Oracle use different mechanisms to constrain data (Tables vs ESRI Domains)
- Editing of geometries held in Oracle Spatial was a problem due to ESRI versioning architecture. At 9.2 it is now possible to do non–versioned editing on layers
- Views can be registered with SDE but not with the Geodatabase this means that views cannot be used with ESRI domains, subtypes, relationship classes etc.
- Managing and editing data using Oracle clients and ESRI clients at the same time

ESRI/Oracle Interoperability
What is difficult?

- There are structural differences between the way GIS and relational systems are designed
- GIS developers tend to have a relatively ‘FLAT’ object based view and consider layers containing features (a homogeneous collection of objects that have a geometry and a set of attributes). Rarely would the attributes be split into a set of related tables. Separate tables would of course be used when 1:M or M:M relationships are needed
- RDBMS developers not surprisingly take a relational view and split data up into many more tables to create a more normalised structure

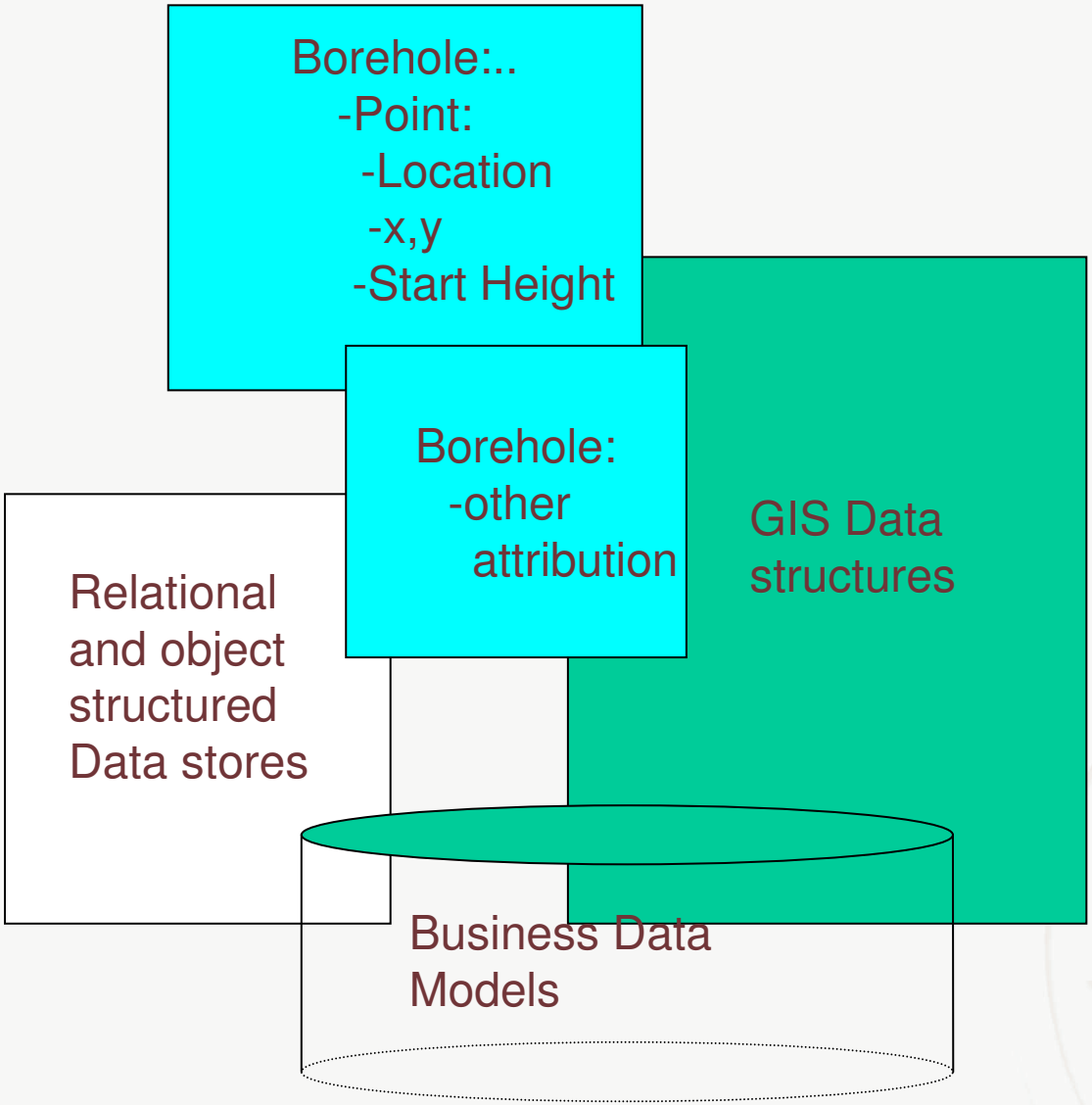
Why Oracle spatial

- The advantages of a database approach
- Spatial component is inside the database and not middleware
- Oracle Spatial is considered mature, reliable and a functionality-rich product
- Few data modelling restrictions
- Seamless integration with other Oracle products
- To standardise formats to make sharing possible
- Centralise for easy access by all parties
- Provide in generic form also for non-specialist GIS staff

Role of Spatial Database Technology

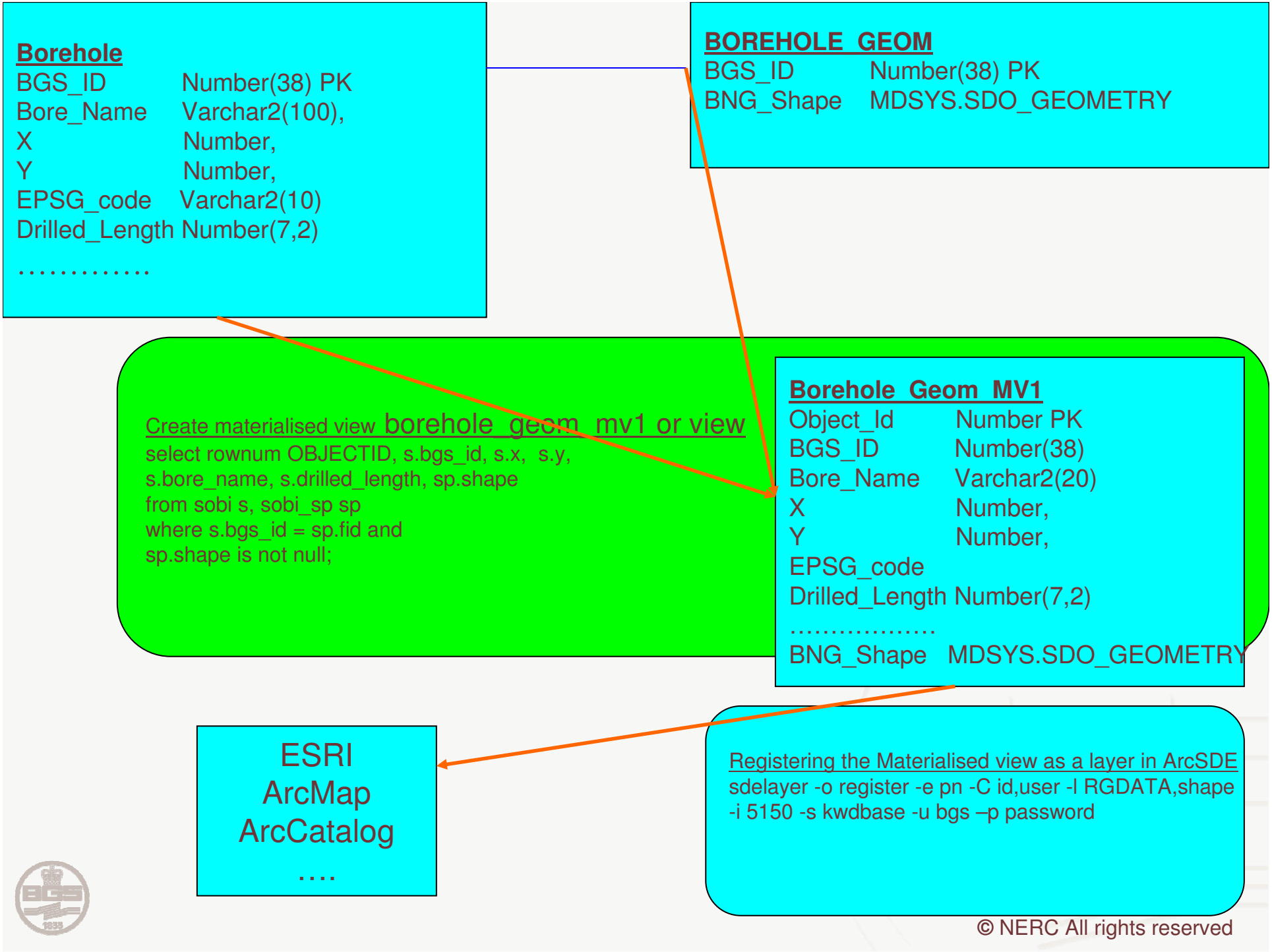
- Spatially enabled databases should improve (that is the expectation):
  - Uniqueness and security
  - Speed
  - Spatial queries on structured data
  - Structured queries on spatial data
  - Effective data management
  - Enable alternate access views

Borehole Data - Example



Methods used to spatially enable Oracle tables

- Adding a Geometry field directly to existing tables where this would not break any existing applications. Geometry column maintained by trigger
- Creating a new table (geometry table) consisting of a geometry field and a unique identifier that can be related to an existing table via a view or materialised view. Geometry maintained by a trigger or a batch process
- Using geometry function and function based index to create a view where read only access is required. No need for a trigger to maintain a separate geometry field as this is created on the fly using a function



Spatially enabling an existing table
Adding a geometry column


```
-- add geometry column to existing table
ALTER TABLE RGDATA ADD SHAPE MDSYS.SDO_GEOMETRY;

-- insert metadata for the table with new column
INSERT INTO USER_SDO_GEOM_METADATA VALUES (
'RGDATA', 'SHAPE', MDSYS.SDO_DIM_ARRAY(
MDSYS.SDO_DIM_ELEMENT('X', -9, 2, 10),
MDSYS.SDO_DIM_ELEMENT('Y', 49, 61, 10)), 4277);

-- Update table to populate geometry column
UPDATE RGDATA SET SHAPE =
MDSYS.SDO_GEOMETRY(2001,4277,MDSYS.SDO_POINT_TYPE(LONGIT
UDE,LATITUDE,NULL),NULL,NULL);

-- create spatial index
CREATE INDEX RGDATA_SI ON RGDATA(SHAPE) INDEXTYPE IS
MDSYS.SPATIAL_INDEX;

-- registering the table as a layer in ArcSDE
sdelayer -o register -e pn -C id,user -l RGDATA,shape -i 5150 -s kwdbase -u
bgs –p password
```

Spatially enabling a table using a function and function based index

---- create function

```
create or replace function get_long_lat_pt(longitude in number,
latitude in number)
return MDSYS.SDO_GEOMETRY deterministic is
begin
return mdsys.sdo_geometry(2001, 8307,
mdsys.sdo_point_type(longitude, latitude, NULL),NULL, NULL);
end;
```

--- Insert metadata for the function

```
insert into user_sdo_geom_metadata values('RGDATA',
'BGS.get_long_lat_pt(longitude,latitude)',
mdsys.sdo_dim_array(mdsys.sdo_dim_element('Longitude', -9, 2,
10), mdsys.sdo_dim_element('Latitude', 49, 61, 10)), 8307);
```

--- create function based index

```
create index RGDATA_SI on
RGDATA(get_long_lat_pt(longitude,latitude)) indextype is
mdsys.spatial_index;
```

Spatially enabling a table using a function and function based index

-- create a view that uses the index

```
create view RGVIEW as select id, latitude, longitude,
get_long_lat_pt(longitude,latitude) SHAPE from RGDATA;
```

-- insert metadata for the view or other object

```
insert into user_sdo_geom_metadata values
('RGVIEW','SHAPE',
mdsys.sdo_dim_array(mdsys.sdo_dim_element('Longitude'
, -9, 2, 10), mdsys.sdo_dim_element('Latitude', 49, 61, 10)),
8307);
```

-- Register the view as a layer in ArcSDE

```
sdelayer -o register -e pn -C id,user -l RGVIEW,shape -i
5150 -s kwdbase -u BGS –p password
```

Spatially enabling a table using a separate geometry table

-- create geometry table

```
create table SOBI_SP (bgs_id number(38) not null, bng_shape
MDSYS.sdo_geometry);
```

-- insert data into table using X, Y from source table

```
insert into SOBI_SP
select bgsid,mdsys.sdo_geometry(2001, 27700,
MDSYS.SDO_POINT_TYPE(bng_easting, bng_northing, NULL),
NULL, NULL) from sobi
where bng_easting is not null and bng_northing is not null;
```

-- create view or materialised view

Create view or materialised view of the base & geometry tables;

-- insert metadata for table & view or materialised view

```
insert into user_sdo_geom_metadata values ('SOBI_SP','SHAPE',
mdsys.sdo_dim_array(mdsys.sdo_dim_element('bng_easting', -100000, 800000, 0.5),
mdsys.sdo_dim_element('bng_northing', -100000, 1400000, 0.5)), 27700);
```

-- Register objects as layers in ArcSDE

```
sdelayer -o register -e pn -C bgs_id,user -l SOBI_SP, shape -i 5150 –s
kwdbase -u BGS –p password
```

Mechanisms used for updating geometry

- Use a trigger to maintain the geometry based on the location attributes entered (e.g. Eastings and Northings)

PL/SQL Triggers & Procedures
to translate classic data
into spatial format.

Triggers, procedures in oracle jobs,
materialised views keep the spatial data in-sync
with the classic data, in real-time and/or on call.

- Use a batch process to update the geometry at a specified interval e.g. every night

- No mechanism required when functions and function based indexes are used

Best of the RDBMS and GIS Worlds

- Unique and secure master location of all data
- Structured query on spatial data
- Spatial query on structured data
- Enable alternative access views
- Server spatial processing (rather than client side)
- Effective data management

Can Oracle Spatial Help unify RDBMS and GIS?

- Yes to an extent, but there are as still technological and cultural hurdles to overcome before we can reach the goal of deep integration between GIS and RDBMS systems and between GIS and RDBMS developers
- Oracle Spatial can certainly be effectively used to spatially enable our data that is currently managed by Oracle and Oracle clients
- In the future we would like to be able to have one central data store and a set of diverse applications capable of directly using that data store with a deep level of integration