ECHO Client Partner User Guide - Chapter 5

ECHO has been replaced by the **Common Metadata Repository (CMR)**, a high-performance, high-quality, continuously evolving metadata system that catalogs all data and service metadata records for the EOSDIS system and will be the authoritative management system for all EOSDIS metadata.

The information contained within this ECHO wiki is now archived for historical reference. Please navigate to the **CMR wiki pages**, or to the **CMR Overview page** on **Earthdata**.

- The ECHO Alternative Query Language
  - Collection/Discovery Search
  - Granule/Inventory Search
  - Common Search Elements
  - Spatial
  - Temporal
  - Search Elements for Collection/Discovery Queries
  - Search Elements for Granule/Inventory Queries
  - Miscellaneous Elements

Guide Navigation

1. Preface
2. Before you begin
3. The basics
4. Logging in, setting up and getting started
5. Querying for Earth Science metadata
6. **The ECHO Alternative Query Language**
7. Ordering data through ECHO
8. Metadata subscriptions
9. Appendices

The ECHO Alternative Query Language

ECHO Alternative Query Language (AQL) is a query language that defines the format for searches on collections and granules in the ECHO system. AQL is an XML-based language. You can find the DTD for AQL at http://api.echo.nasa.gov/echo/dtd/IIMSAQLQueryLanguage.dtd

Code Listing 14: General AQL Structure

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE query PUBLIC "-//ECHO CatalogService (v10)//EN"
"http://api.echo.nasa.gov/echo/dtd/IIMSAQLQueryLanguage.dtd">
<query>
 <for value= "collections"/>
 //"granules" can be substituted for "collections"
  <dataCenterId>
   <all/>
  </dataCenterId>
 <where>
  <list of conditions ... >
 </where>
</query>
```

Note: The <for/> tag shown in this example does not require that you set a value attribute; if you do not explicitly set this attribute, it will default to "collections." Your query will validate correctly, but you may still experience an error if you use granule settings for other attributes within the same query.

There is no discussion regarding the strong suggestion that clients include dataset IDs. Might want to have more high level information regarding the structure of an AQL query and general contructs (list, all, value, etc).
You can specify the data centers that you wish to search by explicitly listing them or by using the empty element **<all/>** to indicate that you want all the data centers searched. This is the default condition, that is, if you do not specify a condition for the **dataCenterId** element, ECHO will search all data centers. *Do not enclose **dataCenterId** values in single-quotes. Is this still needed?*
To search all data providers:

```
<dataCenterId><all/></dataCenterId>
```

To search for collections/granules in the ORNL data provider's metadata only:

```
<dataCenterId>
 <value>ORNL_DAAC</value>
</dataCenterId>
```

To search for collections/granules in the LPDAAC_ECS data providers' metadata only:

```
<dataCenterId>
<list> <value>ORNL_DAAC</value> <value>LPDAAC_ECS</value>
</list>
</dataCenterId>
```

The list of conditions consists of **collectionCondition** elements for Discovery or **granuleCondition** elements for Inventory Search. The fields you can search on within collections occur as children of the **collectionCondition** element. The fields you can search on within granules occur as children of the **granuleCondition** element. ECHO will return only those granules or collections that satisfy all the conditions of the query, that is, ECHO applies the Boolean AND between all conditions of the query.
The **where** tag can have any number of **collectionCondition** or **granuleCondition** elements. Each of these tags can contain any one of the search elements listed in the Inventory Search and Discovery Search tables below. For the query to be valid, each of the search elements may appear only once.

Code Listing 15 shows the general structure of a Discovery query with multiple constraints.

Code Listing 15: Structure of a Discovery Query with Multiple Collection Conditions

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE query PUBLIC "-//ECHO CatalogService (v10)//EN"
"http://api.echo.nasa.gov/echo/dtd/IIMSAQLQueryLanguage.dtd">
<query>
 <for value="collections"/>
  <dataCenterId>
   <value>ORNL_DAAC</value>
  </dataCenterId>
 <where>
  <collectionCondition>… </collectionCondition>
  <collectionCondition>… </collectionCondition>
  …
 </where>
</query>
```

Code Listing 16 shows the general structure of an Inventory query with multiple constraints.

Code Listing 16: Structure of an Inventory Query with Multiple Granule Conditions

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE query PUBLIC "-//ECHO CatalogService (v10)//EN"
"http://api.echo.nasa.gov/echo/dtd/IIMSAQLQueryLanguage.dtd">
<query>
 <for value="granules"/>
  <dataCenterId>
   <value>ORNL_DAAC</value>
  </dataCenterId>
 <where>
  <granuleCondition>… </granuleCondition>
  <granuleCondition>… </granuleCondition>
  …
 </where>
</query>
```

If you specify a list or a single value, then the search looks for an exact match between the string specified and the data stored. For example, if the search value is 'Imagery' and the data stored has the value 'Satellite Imagery', the data will not be returned since there was no exact ma tch. In this instance, it is more useful to use the **textPattern** element for the search.

Collection/Discovery Search

Searches for collections must enclose the search criteria in a **collectionCondition** element. If you want to search the negated criteria, you should set the **negated** attribute in the **collectionCondition** element to **'Y'**.

Code Listing 17 shows how to create a search for all collections except the ones with processing level 1, 1A, 1B, or 2.

Code Listing 17: Using the Negated Attribute on Collection Conditions

```
<collectionCondition negated='Y'>
 <processingLevel>
  <list>
   <value>'1'</value>
   <value caseInsensitive="Y">'1A'</value>
   <value>'1B'</value>
   <value>'2'</value>
  </list>
 </processingLevel>
</collectionCondition>
```

You can set the negated attribute to **'Y'** for a criterion.
The following table lists the criteria you can use to search for collections. For details about each of these criteria, refer to Common Search Elements (same chapter)

Table 4: Collection Search Criteria (Discovery)

Search Criteria	XML Element
Campaign Short Name	<CampaignShortName>...</CampaignShortName>
Dataset ID	<dataSetId>...</dataSetId>
ECHO Insert Date	<ECHOInsertDate>...</ECHOInsertDate>
ECHO Last Update	<ECHOLastUpdate>...</ECHOLastUpdate>
Online Collections Only	<onlineOnly/>
Parameter	<parameter>...</parameter>
Processing Level	<processingLevel>...</processingLevel>
Sensor Name	<sensorName>...</sensorName>
Short Name	<shortName>...</shortName>
Source Name	<sourceName>...</sourceName>
Spatial	<spatial>...</spatial>
Spatial Keywords	<spatialKeywords>...</spatialKeywords>
Temporal	<temporal>...</temporal>
Temporal Keywords	<temporalKeywords>...</temporalKeywords>
Additional Attribute Names	<additionalAttributeNames>...</additionalAttributeNames>
Orderable Items	<orderable/>
Version ID	<versionId>...</versionId>
Archive Center	<archiveCenter>...</archiveCenter>
Additional Attributes	<additionalAttributes>… </additionalAttributes>
Instrument Short Name	<instrumentShortName>…</instrumentShortName>

Granule/Inventory Search

Searches for granules must enclose the search criteria in the **granuleCondition** element. As in the case of **collectionCondition**, if you want

to search the negated criteria, set the **negated** attribute in the **granuleCondition** element to **'Y'**. You can set the negated attribute to **'Y'** for any criterion.
The following table lists the criteria you can use to search for granules.

Table 5: Granule Search Criteria (Inventory)

Search Criteria	XML Element
Only Granules with Browse Data	<browseOnly/>
Campaign Short Name	<CampaignShortName>...</CampaignShortName>
Percentage of cloud Cover	<cloudCover>...</cloudCover>
Dataset ID	<dataSetId>...</dataSetId>
ECHO Insert Date	<ECHOInsertDate>...</ECHOInsertDate>
ECHO Last Update	<ECHOLastUpdate>...</ECHOLastUpdate>
Either Day or Night Granules Only	<dayNightFlag/>
Search on ECHO granule IDs (formerly granuleId)	<ECHOGranuleID>...</ECHOGranuleID>
Search on Granule UR (provider specific)	<GranuleUR>...</GranuleUR>
Online Granules Only	<onlineOnly/>
Two-D Coordinate System	<TwoDCoordinateSystem>...</TwoDCoordinateSystem>
Producer Granule ID	<ProducerGranuleID>...</ ProducerGranuleID>
Additional Attributes	<additionalAttributes>...</additionalAttributes>
Sensor Name	<sensorName>...</sensorName>
Source Name	<sourceName>...</sourceName>
Spatial	<spatial>...</spatial>
Temporal	<temporal>...</temporal>
Orderable Items	<orderable/>
Version ID	<versionId>...</versionId>
PGE Name	<PGEName>...</PGEName>
PGE Version	<PGEVersion>...</PGEVersion>
Measured Parameters	<measuredParameters>...</measuredParameters>
Provider Production Date	<providerProductionDate>…</providerProductionDate>
Provider Insert Date	<providerInsertDate>…</providerInsertDate>
Instrument Short Name	<instrumentShortName>…</instrumentShortName>

Effective with ECHO Release 7.0, the LocalGranuleID search criterion was renamed ProducerGranuleID.

Common Search Elements

You may use the following search criteria for both collection and granule searches.
Case Insensitive Searching
Any **value** or **textPattern** tag has an attribute **caseInsensitive** that you can set to **Y** or **N** to indicate whether the search **value** or **textPattern** is case insensitive. The default is **N**, that is, the default is case sensitive searching. Refer to the dataSet ID example on the next page for an example.

CampaignShortName

CampaignShortName is the name(s) of the campaign or project that gathered data associated with the collection or granule. In **CampaignShortName**, you can use a *single-quoted string* in a **value**, a list of *single-quoted strings* in a **list**, or a **textPattern** element as defined in the section 6.8, Miscellaneous Elements.

This example shows a search for collections/granules where **CampaignShortName** = 'River'.

```
<CampaignShortName>
 <value>'River'</value>
</CampaignShortName>
```

This example shows a search for collections/granules where **CampaignShortName** = 'River' or **CampaignShortName** = 'Lake'.

```
<CampaignShortName>
 <list>
  <value>'River'</value>
  <value>'Lake'</value>
 </list>
</CampaignShortName>
```

This example shows a search for collections/granules where **CampaignShortName** contains the string 'AS_A%D', where the % represents zero or more numbers or letters between the *A* and the *D* in the text string 'AS_A%D'

```
<CampaignShortName>
 <textPattern operator="LIKE">'AS_A%D'</textPattern>
</CampaignShortName>
```

DataSetID

dataSetID specifies the universal name of a collection. You can use this element to restrict the search to a few collections, using the **value** element to specify a single collection, a **list** for a list of collections, or a **textPattern**. For the **value** and **list** elements, you must know the exact name of the collection. For example, you may have performed a Discovery search to locate the collection of interest and now want to perform an Inventory search for granules.

This example shows a search for all collections whose **dataSetId** ends with '1A'.

```
<dataSetId>
 <textPattern caseInsensitive="Y">'%1A'</textPattern>
</dataSetId>
```

This example shows a search for all granules whose **dataSetId** is either 'LEAF CHEMISTRY, 1992-1993 (ACCP)' or 'ASTER DEM Product V002'.

```
<dataSetId>
 <list>
  <value>'LEAF CHEMISTRY, 1992-1993 (ACCP)'</value>
  <value>'ASTER DEM Product V002'</value>
 </list>
</dataSetId>
```

OnlineOnly

Use **OnlineOnly** to search for granules or collections that are available online.

```
<onlineOnly/>
```

ECHO Insert Date
Use **ECHOInsertDate** to specify a search for collections or granules based on their insertion date into ECHO. You can only specify a dateRange.

```
<ECHOInsertDate>
 <dateRange>
  <startDate>
   <Date YYYY="2001" MM="04" DD="05"/> // endDate is not required
  </startDate>
 </dateRange>
</ECHOInsertDate>
```

ECHO Last Update

Use **ECHOLastUpdate** to specify a search for collections or granules based on the date of their last update within ECHO. You can only specify a dateRange.

```
<ECHOLastUpdate>
 <dateRange>
  <startDate>
   <Date YYYY="2001" MM="01" DD="01"/>
  </startDate>
  <stopDate>
   <Date YYYY="2001" MM="06" DD="01"/>
  </stopDate>
 </dateRange>
</ECHOLastUpdate>
```

Sensor Name
Use **sensorName** to search for collections or granules based on the name of the sensor. known value in a **value**, a list of known values in a **li st**, or a text pattern in a **textPattern**. relationship between each value with the Boolean "AND" or "OR".

The "AND" relationship means that results must match every value specified.

You can search for a You can specify the
The "OR" relationship means that results must match either or both of the values specified. "OR" is the default setting for searching with multiple values.

```
<granuleCondition>
 <sensorName operator="AND">
  <list>
   <value>'human observer'</value>
   <value>'rain gauge'</value>
  </list>
 </sensorName>
</granuleCondition>
```

Instrument Short Name
Use **instrumentShortName** to search for collections or granules based on the name of the instrument. You can search for a known value in a **value**, a list of known values in a **list**, or a text pattern in a **textPattern**. You can specify the relationship between each value with the Boolean "AND" or "OR".

The "AND" relationship means that results must match every value specified.

The "OR" relationship means that results must match either or both of the values specified. "OR" is the default setting for searching with multiple values.

```
<granuleCondition>
 <instrumentShortName operator="AND">
  <list>
   <value>'GPS'</value>
   <value>'Telescope'</value>
  </list>
 </instrumentShortName>
</granuleCondition>
```

Source Name
Use **sourceName** to search for collections or granules based on the name of the source. You may specify the name using a **value**, a list of known values in a **list**, or a text pattern in a **textPattern**. You can specify the relationship between each value with the Boolean "AND" or "OR".

The "AND" relationship means that results must match every value specified.

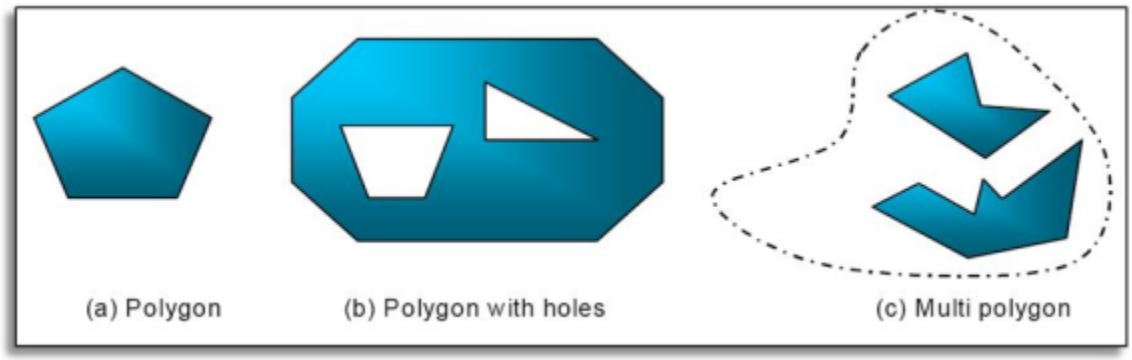
The "OR" relationship means that results must match either or both of the values specified. "OR" is the default setting for searching with multiple values

```
<granuleCondition>
 <sourceName operator="AND">
  <list>
   <value>'CV-580'</value>
   <value>'C-130'</value>
  </list>
 </sourceName>
</granuleCondition>
```

Spatial

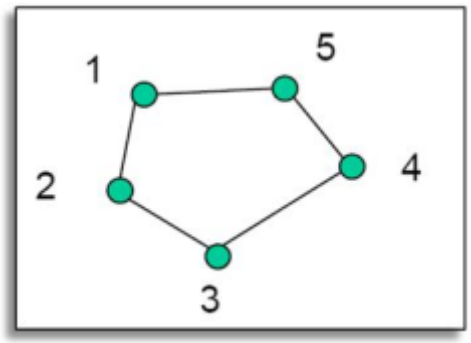
Use spatial constraints to restrict the search within a spatial extent of the earth. Specify the spatial extent using the elements **IIMSPolygon**, **IIMSMultiPolygon**, **IIMSBox** or **IIMSLine**.
You may also specify a **TwoDCoordinate** element along with a spatial element in a spatial constraint. This is useful when you wish to find spatially coincident granules between collections that support **TwoDCoordinate** searches and those that do not. When you use **TwoDCoordinate** together with a spatial element, ECHO will return all granules that satisfy either the **TwoDCoordinate** constraint or the spatial element, or both. You may only use **TwoDCoordinate** in a spatial constraint for granule (inventory) queries, not collection queries.
You may also use **TwoDCoordinate** outside of a spatial constraint. In this case, list **TwoDCoordinate** inside a **granuleCondition**. ECHO will return results that satisfy the **TwoDCoordinate** constraint and all other conditions (including any spatial conditions if they are present).
A **Polygon** consists of one or more rings. Some rings may be embedded within others that define the interior of the polygon. *The ECHO system can handle polygons with only one external ring and multiple internal rings.*

Figure 6: Allowed Geometries for Spatial Data and Query Windows



To define a ring, the last point in the ring must be the same as the first point in the ring. You must specify the points in each polygonal ring in counterclockwise order.

Figure 7: Order of Points in a Polygonal Ring

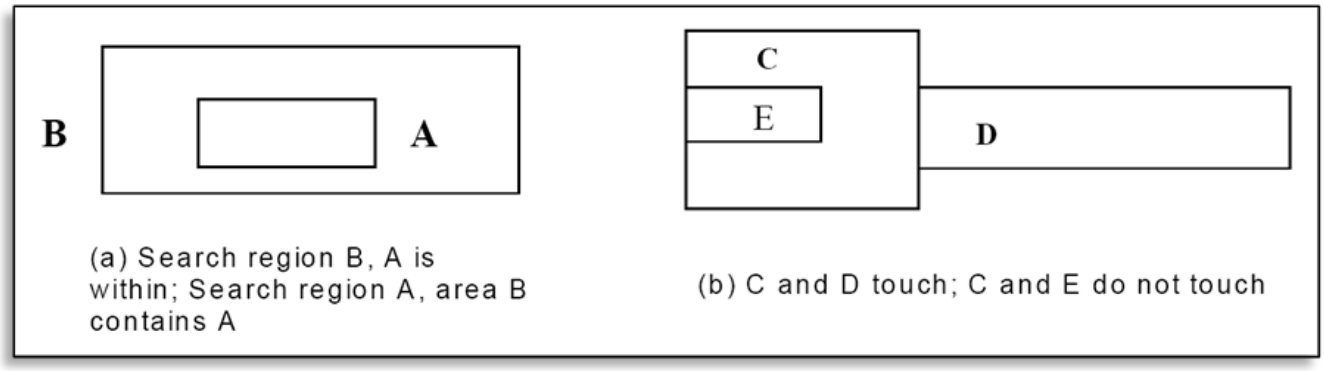


To specify multiple outer rings, use the **MultiPolygon** or **IIMSMultiPolygon** element, with each outer ring in a separate **Polygon**. There should be no overlap between any of the outer rings defined. Refer to Code Listing 19: A Multi-Polygon Spatial Query for examples.
Use **IIMSBox** to specify a bounding box on the Earth. A bounding box contains two points. The first point is the lower left corner of the box. The second point is the upper right corner of the box. The horizontal lines of the box will follow the latitude lines of the earth. This means the points of the box will not connect by lines following great circle arcs but will curve to stay at the same latitude.
Use **IIMSLine** or **LString** to specify a line on the Earth. You must specify the points in the line from one endpoint to the other. You can also use **IIMSPoint** to specify a single point on the Earth. Note that although **IIMSPoint** is used to describe corner points in a polygon or endpoints in a line, it also describes a specific searchable geometry. Refer to Code Listing 20: A Single Point Spatial Query and for examples.
The table below describes Spatial Operators available in the ECHO System, regarding the relationship between two objects.

Table 6: Spatial Operators

Spatial Operator	Description
EQUAL	They have the same boundary and interior.
TOUCH	The boundaries intersect but the interiors do not.
WITHIN	Interior and boundary of the first object are completely contained in the interior of the second.
CONTAINS	The interior and boundary of the second object are completely contained in the interior of the first.
RELATE	The objects are non-disjoint, that is, their bodies and/or boundaries intersect.

Figure 8: Examples of Spatial Regions



Define the query so that the user-specified spatial extent is the second object in the query.

Use **CONTAINS** to retrieve granules/collections that contain the specified polygon.

Use **WITHIN** to retrieve granules/collections that are within the specified spatial extent.

Use **RELATE** to retrieve granules/collections that overlap in some way. **RELATE** is the default operator.
ECHO supports multiple spatial representations (or SpatialTypes). Spatial data of different spatial representations are described differently and hence are stored and queried differently.
The **SpatialType** tag is an optional field in AQL used to specify the spatial type. If the user opts not to specify this field, ECHO performs a default search. By default, ECHO performs the search on **NORMAL** (Cartesian and Geodetic), and **ORBIT**.

Code Listing 18: Spatial Search Element

```
<spatial operator="RELATE">
 <IIMSPolygon>
  <IIMSLRing>
   <IIMSPoint lon="-120" lat="-30" />
   <IIMSPoint lon="-100" lat="-60" />
   <IIMSPoint lon="5" lat="-90" />
   <IIMSPoint lon="-120" lat="-60" />
   <IIMSPoint lon="160" lat="5" />
   <IIMSPoint lon="160" lat="60" />
   <IIMSPoint lon="120" lat="85" />
   <IIMSPoint lon="5" lat="85" />
   <IIMSPoint lon="-120" lat="30" />
   <IIMSPoint lon="-120" lat="-30" />
  </IIMSLRing>
  <IIMSLRing>
   <IIMSPoint lon="80" lat="20" />
   <IIMSPoint lon="80" lat="60" />
   <IIMSPoint lon="20" lat="60" />
   <IIMSPoint lon="20" lat="20" />
   <IIMSPoint lon="80" lat="20" />
  </IIMSLRing>
 </IIMSPolygon>
 <TwoDCoordinateSystem>
  <TwoDCoordinateSystemName>
   <value>'WRS2'</value>
  </TwoDCoordinateSystemName>
  <Coordinate1><range lower='15' upper='20'/></Coordinate1>
<Coordinate2><range lower='33' upper='42'/></Coordinate2>
 </ TwoDCoordinateSystem>
 <SpatialType>
  <list>
   <value>NORMAL</value>
   <value>ORBIT</value>
  </list>
 </SpatialType>
</spatial>
```

Spatial Type: NORMAL

Conventional spatial data are described as shapes such as polygons, multi-polygons, boxes, or lines. NORMAL is the spatial type to use for searching this type of data. Spatial data are stored in the database as Oracle objects to record the shape, spatial locations, and coordinate system used; i.e., Cartesian or Geodetic.

All spatial objects with the exception of Box in AQL, ECHO interprets as Geodetic. Geodetic spatial data has some restrictions. The total area of a single polygon cannot exceed half the earth. Lines, either alone or in a polygon, cannot be longer than half the circumference of the earth.

The same points in Cartesian and Geodetic describe two different areas. Great circle arcs connect points in Geodetic. These follow the shortest paths on the earth between two points. Latitude lines connect points in Cartesian. The image below shows a geodetic area Google Earth below as a red outlined polygon overlayed over a white polygon representing the Cartesian area described with the same points.

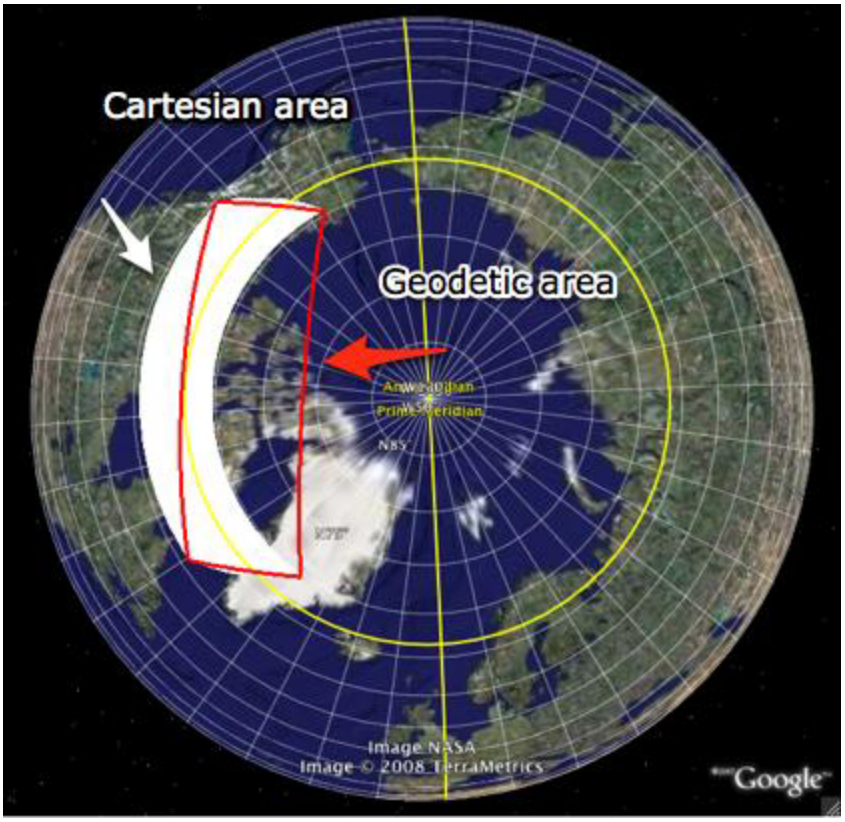


Figure 9: Spatial differences between a Geodetic area and a Cartesian Area

As noted previously, Geodetic and Cartesian represent polygons in different ways. There are other differences though besides how the lines define the shape. These differences stem from the fact that Geodetic and Cartesian are different representations of the earth. Geodetic is an ellipsoidal model of the earth whereas Cartesian describes the surface of the earth as a flat plane.

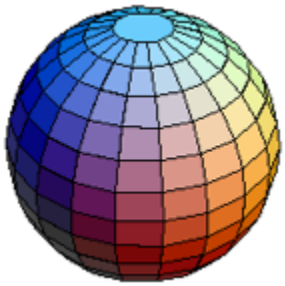


Figure 10: Ellipsoid

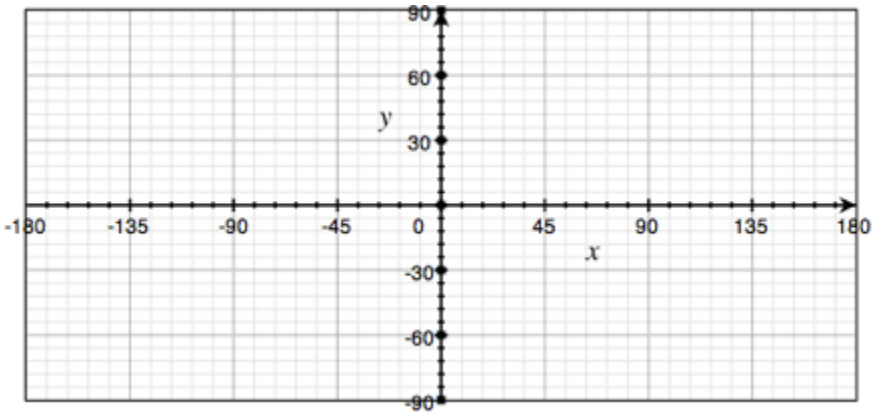


Figure 11: Flat Cartesian plane

These fundamental differences lead to some polygons being valid in Geodetic and not valid in Cartesian. Polygons and lines follow Geodesics so ECHO uses them as-is for searching geodetic data. Boxes, on the other hand, follow the latitude lines of the earth so ECHO uses them as-is for searching Cartesian data. ECHO converts polygons and lines from Geodetic for searching the Cartesian data and converts boxes from Cartesian when searching Geodetic data. The converted areas are valid approximations of the original areas. ECHO performs the conversion first by dividing the spatial area into pieces so that each new area fits within the bounds of sections of the earth. ECHO uses eight octants when converting from Geodetic to Cartesian. ECHO uses four quadrants when converting from Cartesian to Geodetic. The Cartesian graph below shows the octant bounds. ECHO divides the spatial area so that it does not cross the antimeridian, the prime meridian, or either of the poles.

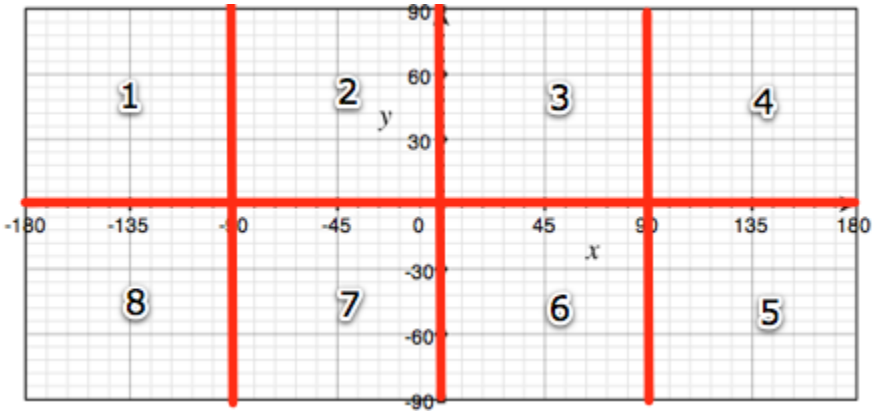
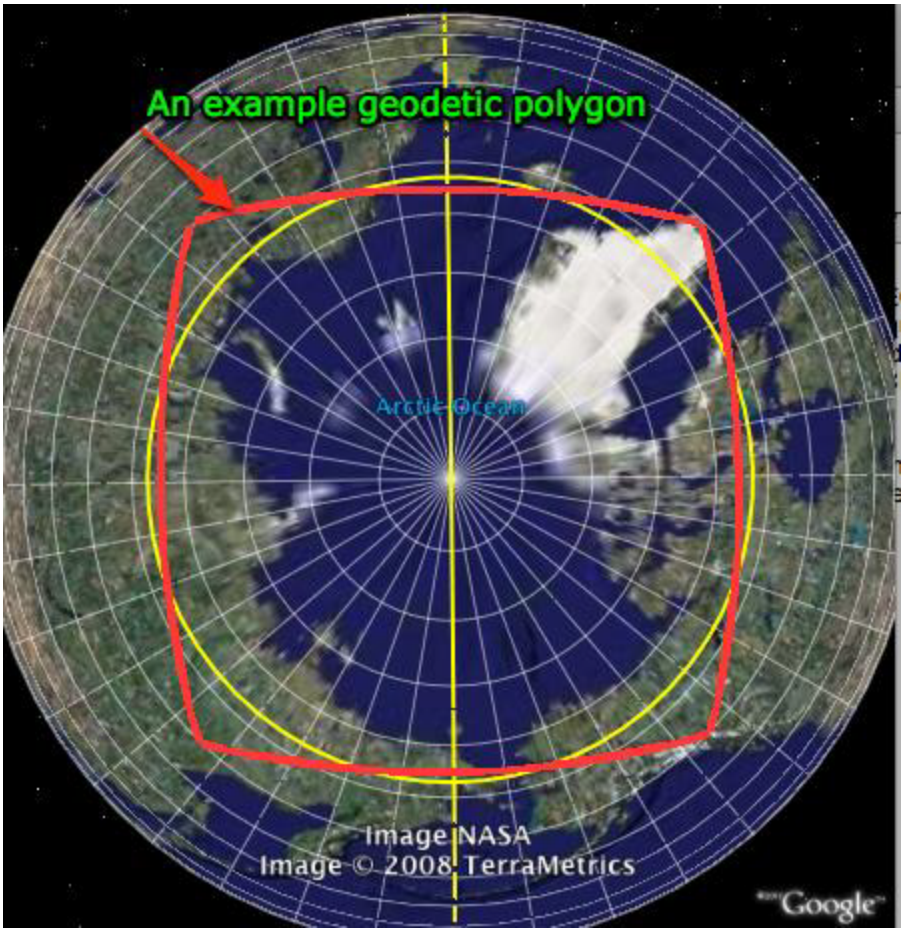


Figure 12: Bounds of the eight octants

After dividing the spatial area, the lines of the spatial area are "densified". Densification means that ECHO adds more points along the lines between the original points. This has the effect of making the lines in Cartesian space approximate the lines in geodetic space and vice versa. ECHO adds points every 10 km during densification. We found this number to be a good compromise between performance and accuracy. It is important that client developers understand ECHO uses an approximation of the area so in a small number of cases there may be some missed or extra results. Images, shown below, illustrate the Geodetic to Cartesian Conversion process.


An example geodetic polygon
Arctic Ocean
Image NASA
Image © 2008 TerraMetrics
Google

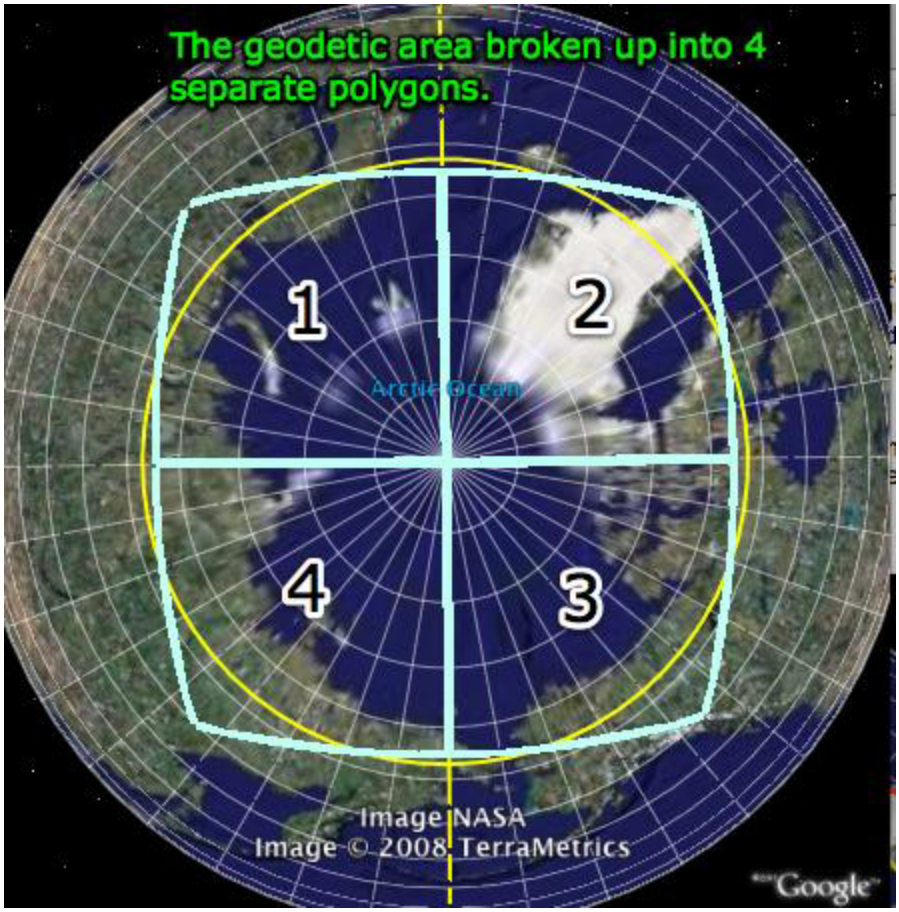
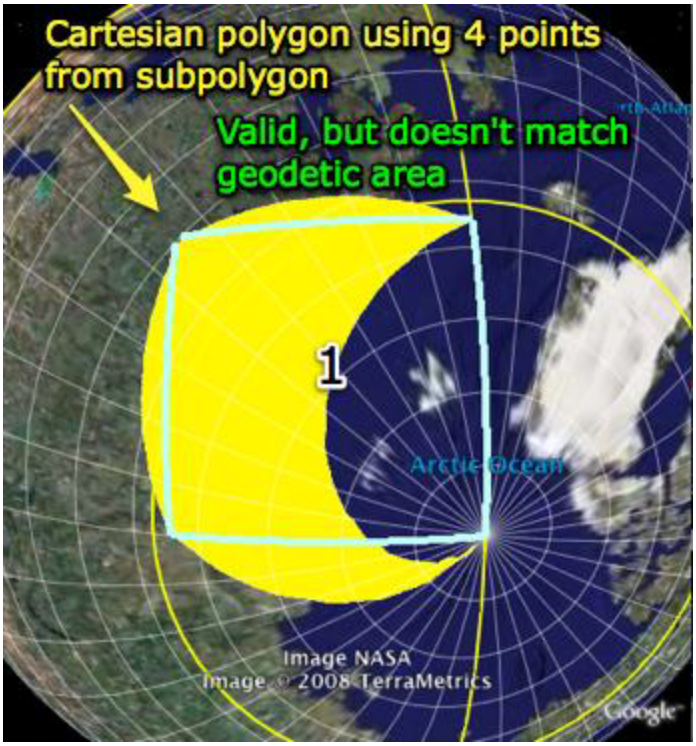


Figure 13: Example geodetic polygon divided by octants

Figure 14: Geodetic polygon



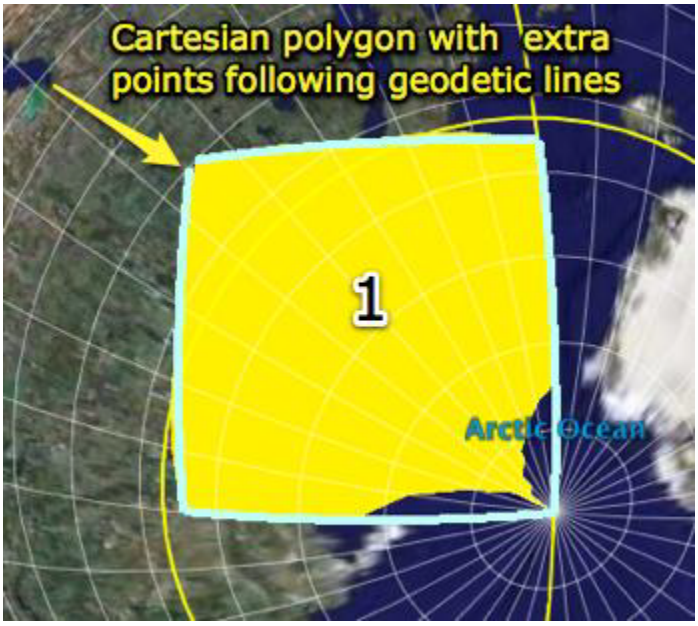


Figure 15: Cartesian polygon before densification

Figure 16: Cartesian polygon after densification

Spatial Type: ORBIT

Spatial data are orbit-based and generated from satellites running along orbits. Because an orbit travels at a fixed direction with a fixed declination angle and the covered area is a stripe of fixed width, the spatial coverage of a granule can be derived from the latitude where the granule starts and ends and the crossing longitude on the equator where its orbit originally starts. Thus, the orbit-based spatial data are not stored as shape objects but as the original data. ECHO uses an algorithm called Backtrack to perform searches on orbit-based data. Spatial coverage can be as large as the entire earth surface including polar area. ECHO does simple validation against the spatial window to make sure it does not exceed the scope of the earth (the latitude is within the range –90 to 90 and the longitude is within the range –180 to 180). There is one limitation in the current implementation: the only supported spatial operator is **RELATE**. With ECHO 10.0, multi-orbit as well as single orbit searches are supported.

This example shows a multi-polygon spatial query.

```
<spatial operator='RELATE'>
 <IIMSMultiPolygon>
  <IIMSPolygon>
   <IIMSLRing>
    <IIMSPoint lat="85" long="-50"/>
    <IIMSPoint lat="70" long="-60"/>
    <IIMSPoint lat="60" long="-50"/>
    <IIMSPoint lat="70" long="-40"/>
    <IIMSPoint lat="85" long="-50"/>
   </IIMSLRing>
  </IIMSPolygon>
  <IIMSPolygon>
   <IIMSLRing>
    <IIMSPoint lat="-85" long="-50"/>
    <IIMSPoint lat="-70" long="-40"/>
    <IIMSPoint lat="-60" long="-50"/>
    <IIMSPoint lat="-70" long="-60"/>
    <IIMSPoint lat="-85" long="-50"/>
   </IIMSLRing>
  </IIMSPolygon>
 </IIMSMultiPolygon>
</spatial>
```

Refer to Figure 6: Allowed Geometries for Spatial Data and Query Windows for an example of an image of a multi-polygon.

This example shows a single point spatial query.

```
<spatial operator='RELATE'>
 <IIMSPoint long='-75.0' lat='35.0'/>
</spatial>
```

Temporal

Use temporal constraints to specify the temporal extent of the data. The dates are stored as Gregorian dates, and the times are stored in GMT. This search allows only dates in YYYY-DD-MM format. You can specify a range of time represented by beginning and ending dates or a periodic temporal range specified by start date and end date. For a periodic temporal range, you can specify the start and end day of each year. You should use integers between 1 and 365 to specify the start and end dates, with start day =< end day. You can specify the end day as 366, *but* if the temporal range contains a non-leap year, then an error will be generated.
This example shows a search for granule/collections whose temporal range overlaps with the time between May 12, 1990, and June 13, 1992.

Code Listing 21: Temporal Search Example

```
<temporal>
 <startDate>
  <Date YYYY="1990" MM="5" DD="12"/>
 </startDate>
 <stopDate>
  <Date YYYY="1992" MM="2" DD="13"/>
 </stopDate>
</temporal>
```

This example shows a search for granules/collections whose temporal range overlaps with all of these time ranges:

- May 16 - July 14, 1990
- January 5 - March 5, 1991
- January 5 - February 10, 1992

Code Listing 22: Temporal Example with Different Day Notations

```
<temporal>
 <startDate>
  <Date YYYY="1990" MM="5" DD="12"/>
 </startDate>
 <stopDate>
  <Date YYYY="1992" MM="2" DD="10"/>
 </stopDate>
 <startDay value="5"/>
 <endDay value="60"/>
</temporal>
```

Version ID
Use **versionId** to search for collections or granules based on the version identifier of the data collection. You can search for a known value in a **value**, a list of known values in a **list**, or a text pattern in a **textPattern**. As of ECHO 10.0, versionId is stored as a string so that it may look like 3,0033,0 ….

Code Listing 23: Version ID in a Collection Condition

```
<versionId>
 <value>'0'</value>
</versionId>
```

Code Listing 24: Version ID in a Granule Condition with a Pattern

```
<versionId>
 <textPattern operator="LIKE">'3.%'</textPattern>
</versionId>
```

Additional Attributes

Use **additionalAttributes** to search for collections or granules based on one or many groups of the additional attributes by specifying the additional attribute name/value pair for an exact match. This function includes searching on multiple additional attributes. You may specify the relationship between groups with either "AND" or "OR".

The "AND" relationship means that result granules must match across every group.

The "OR" relationship means that result granules match with any of the specified groups. "OR" is the default setting.
Within each additional attribute, you must specify an **additionalAttributeName** in *single quotation marks (that is, apostrophes)* for an exact match. You may specify the **additionalAttributeValue** using **value**, **list**, **textPattern**, **range**, or **dateRange**.
This example shows the "AND" relationship between two additional attribute groups. You can use this sample search with either **granuleCondition** or **collectionCondition**.

Code Listing 25: Additional Attributes Example with an AND Relationship

```
<additionalAttributes operator='AND'>
 <additionalAttribute>
  <additionalAttributeName>
   'AveragedFocalPlane1Temperature'
  </additionalAttributeName>
  <additionalAttributeValue>
   <range lower='169' upper='170'/>
  </additionalAttributeValue>
 </additionalAttribute>
 <additionalAttribute>
  <additionalAttributeName>
   'AveragedFocalPlane2Temperature'
  </additionalAttributeName>
  <additionalAttributeValue>
   <range lower='269' upper='270'/>
  </additionalAttributeValue>
 </additionalAttribute>
</additionalAttributes>
```

This example shows the "OR" relationship between two additional attribute groups.

Code Listing 26: Additional Attribute Example with an OR Relationship

```
<additionalAttributes operator='OR'>
 <additionalAttribute>
  <additionalAttributeName>
   'AveragedFocalPlane1Temperature'
  </additionalAttributeName>
  <additionalAttributeValue>
   <range lower='169' upper='170'/>
  </additionalAttributeValue>
 </additionalAttribute>
 <additionalAttribute>
  <additionalAttributeName>
   'AveragedFocalPlane2Temperature'
  </additionalAttributeName>
  <additionalAttributeValue>
   <range lower='269' upper='270'/>
  </additionalAttributeValue>
 </additionalAttribute>
</additionalAttributes>
```

Search Elements for Collection/Discovery Queries

This section describes elements used specifically for collection (Discovery) searches.

Science Keywords

Use **scienceKeywords** to specify the science keywords associate with a collection. Science Keywords are based on the GCMD managed list of science keywords. They have the following hierarchical structure *Category> Topic > Term > Variable Level 1 > Variable Level 2 > Variable Level 3 > Detailed Variable.* A value at any of the specific hierarchy levels can be specified or *anykeyword* can be used to search all the levels. Science Keywords are all stored in upper case. Queries with science keywords will automatically be changed to upper case for searching.

If you specify a list or a single value, then the search looks for an exact match between the string specified and the data stored. For example, if the search value is 'IMAGERY' and the data stored has the value 'SATELLITE IMAGERY', the data will not be returned since there was no exact match. In this instance, it is more useful to use the **textPattern** element for the search.

Code Listing 27: Sample Science Keyword Search

```
<collectionCondition>
 <scienceKeywords operator="AND">
  <scienceKeyword>
   <categoryKeyword>
    <value>'category'</value>
   </categoryKeyword>
   <topicKeyword>
    <value>'topic'</value>
   </topicKeyword>
   <termKeyword>
    <value>'term'</value>
   </termKeyword>
   <variableLevel1Keyword>
    <value>'level1'</value>
   </variableLevel1Keyword>
   <variableLevel2Keyword>
    <value>'level2'</value>
   </variableLevel2Keyword>
   <variableLevel3Keyword>
    <value>'level3'</value>
   </variableLevel3Keyword>
   <detailedVariableKeyword>
    <value>'detailed'</value>
   </detailedVariableKeyword>
  </scienceKeyword>
 </scienceKeywords>
<collectionCondition>
```

Code Listing 28: Sample Science Keyword Search with anyKeyword

```
<collectionCondition>
 <scienceKeywords operator="AND">
  <scienceKeyword>
   <anyKeyword>
    <textPattern>'%WATER%'</textPattern>
   </anyKeyword>
  </scienceKeyword>
 </scienceKeywords>
<collectionCondition>
```

Processing Level

Use **processingLevel** to specify the level at which the data (collection/granule) has been processed, for example, level 0, 1, 1A, 1B, 2, 3, 4. The search can specify one or more processing levels. The values must be single-quoted strings.

This example shows a search for all collections at processing level 1, 1A, or 1B.

Code Listing 29: Processing Level Search Element

```
<collectionCondition>
 <processingLevel>
  <list>
   <value>'1'</value>
   <value>'1A'</value>
   <value>'1B'</value>
  </list>
 </processingLevel>
</collectionCondition>
```

This example shows a search for all collections at processing level 1, 1A, or 1B—any value starting with the digit 1. This also returns results for collections with processing levels 13, 14, etc.

Code Listing 30: Processing Level Element with Text Pattern

```
 <collectionCondition>
 <processingLevel>
  <textPattern operator="LIKE">'1_'</textPattern>
 </processingLevel>
</collectionCondition>
```

Additional Attribute Name

Use additionalAttributeName to search for collections based on the names of the additional attributes in the collections. You can search for a known value in a value, a list of known values in a list, or a text pattern in a textPattern.
This example shows a search for data with an additionalAttributeName value of 1 or 3.

Code Listing 31: Additional Attribute Search Element

```
<collectionCondition>
 <additionalAttributeName>
  <list>
   <value>'value_1'</value>
   <value>'value_3'</value>
  </list>
 </additionalAttributeName>
</collectionCondition>
```

Short Name

Use **shortName** to specify a search for the short name of collections, which may or may not be unique for a particular provider.
This example shows a search for all collections that have shortNames starting with BOREAS.'

Code Listing 32: Short Name Search Element

```
<collectionCondition>
 <shortName>
  <textPattern>'BOREAS%'</textPattern>
 </shortName>
</collectionCondition>
```

Spatial Keywords
Use **spatialKeywords** to specify a word or phrase that summarizes the spatial region covered by the collection. A collection may cover several regions.
This example shows a search for all collections that cover Africa or Bermuda or the Indian Ocean.

Code Listing 33: Spatial Keywords Search Element

```
<collectionCondition>
 <spatialKeywords>
  <list>
   <value>'Africa'</value>
   <value>'Bermuda'</value>
   <value>'Indian Ocean'</value>
  </list>
 </spatialKeywords>
</collectionCondition>
```

This example shows a search for all collections that cover some region(s) of the Americas.

Code Listing 34: Spatial Keywords Element with Text Pattern

```
<collectionCondition>
 <spatialKeywords>
  <textPattern operator="LIKE">'%america%'</textPattern>
 </spatialKeywords>
</collectionCondition>
```

Temporal Keywords
Use **temporalKeywords** to specify a word or phrase that summarizes the temporal characteristics referenced in the collection.
This example shows a search for all collections that have accumulated data during the spring or fall.

Code Listing 35: Temporal Keywords Element

```
<collectionCondition>
 <temporalKeywords>
  <list>
   <value>'spring'</value>
   <value>'fall'</value>
  </list>
 </temporalKeywords>
</collectionCondition>
```

Archive Center
Use **archiveCenter** to search for collections based on the center where a collection is archived. You may search for a known value in a **value** , a list of known values in a **list**, or a text pattern in a **textPattern**.

Code Listing 36: Archive Center Search Element

```
<collectionCondition>
 <archiveCenter>
  <value>'ORNL_DAAC'</value>
 </archiveCenter>
</collectionCondition>
```

Search Elements for Granule/Inventory Queries

This section describes elements used specifically for granule (Inventory) searches.

Granules with Browse Data

Use **browseOnly** to specify that you want only granules that have browse data available.

Code Listing 37: Browse-only Search Element

```
<granuleCondition>
 <browseOnly/>
</granuleCondition>
```

Percentage of Cloud Cover

Use cloudCover to specify a range of percentage of scene cloud coverage in a granule. For the value in each attribute of the range, you should use a floating number between 0 and 100 that specifies the range in percentage of cloud coverage in a granule. *You cannot use the negated condition for this search.*

This example shows a search for only those granules that have cloud cover between 10% and 20%.

Code Listing 38: Cloud Cover Search Element

```
<granuleCondition>
 <cloudCover>
  <range lower="10" upper="20"/>
 </cloudCover>
</granuleCondition>
```

Day, Night, or Both Granules

Use **dayNightFlag** to specify a search for granules that are gathered during daylight only, nighttime only, or both. If you do not specify this criterion, then your results will include granules gathered at all times of day.

This example shows a search for granules gathered at least partly during daylight.

Code Listing 39: Search Element for Granules Gathered During Daylight Only

```
<granuleCondition>
 <dayNightFlag value="DAY"/>
</granuleCondition>
```

This example shows a search for granules gathered at least partly during the night.

Code Listing 40: Search Element for Granules Gathered During the Nighttime Only

```
<granuleCondition>
 <dayNightFlag value="NIGHT"/>
</granuleCondition>
```

This example shows a search for granules gathered partly at night and partly during the daylight.

Code Listing 41: Search Element for Granules Gathered During Daylight, Nighttime, or Both

```
<granuleCondition>
 <dayNightFlag value="Both"/>
</granuleCondition>
```

ECHO Granule IDs

Use **ECHOGranuleID** to specify an inventory search for specific granules. This search does not need to specify any spatial or temporal constraint. You can use this as a second stage query when you know the list of granules you are interested in from a previous query to hone results. The search is then limited to granules in the list. The list may contain granules from different data sets. ECHO performs the search on ECHO_GRANULE_ID, an ECHO-wide unique ID. You must enclose the names of the granules in single quotation marks (that is, apostrophes).

Code Listing 42: Granule ID Search Element

```
<granuleCondition>
 <ECHOGranuleID>
  <list>
   <value>'G1984-ORNL_DAAC'</value>
  </list>
 </ECHOGranuleID>
</granuleCondition>
```

Granule UR

Use **GranuleUR** to specify an inventory search for specific granules. Like granule ID, this criterion can be used as a secondary stage query. Instead of using the ECHO-specific ID for the granules, this criterion allows you to use the provider-given name for the granules. Provider-given names are more meaningful and easier to remember. Since the granule names given by the provider are not guaranteed to be unique within ECHO, if a granule with the same name exists for more than one **dataCenterId** specified in the query, all will be returned in the results. You must enclose the names of the granules in single quotation marks (that is, apostrophes). Note that the GranuleUR can contain value, list, textPattern or patternList children.

Code Listing 43: Granule UR Search Element

```
<granuleCondition>
 <GranuleUR>
  <list>
   <value>'ACCP_CANOPYCHEM.df_can_calc.dat'</value>
   <value>'ANGLIA_10YRCLIMATE.cfrs1120.zip'</value>
  </list>
 </GranuleUR>
</granuleCondition>
```

Two-Dimensional Coordinate System

Use **TwoDCoordinateSystem** as an alternative to specifying a spatial or temporal range. For some collections, granules can be located by searching a range of values on a two dimensional grid, (for example, Path/Row for Landsat, X/Y for MODIS Grid data).

The **TwoDCoordinateSystem** criterion needs three elements:

- the **TwoDCoordinateSystemName**
- the **Coordinate1** range
- the **Coordinate2** range

Coordinate1 and **Coordinate2** may also take a single value. The query returns all granules that match your specified TwoDCoordinateSystemName and overlap with the **Coordinate1** and **Coordinate2** ranges. **Coordinate1** and **Coordinate2** ranges must have both lower and upper attributes specified.
This example shows a search for a range value.

Code Listing 44: TwoDCoordinateSystem Search Element

```
<granuleCondition>
 <TwoDCoordinateSystem>
  <TwoDCoordinateSystemName>
   <value>'WRS2'</value>
  </TwoDCoordinateSystemName>
  <Coordinate1>
   <range lower='15' upper='20'/>
  </Coordinate1>
  <Coordinate2>
   <range lower='33' upper='42'/>
  </Coordinate2>
 </TwoDCoordinateSystem>
</granuleCondition>
```

This example shows a search for a single value.

Code Listing 45: TwoDCoordinateSystem Search Element with a Single Value

```
<granuleCondition>
 <TwoDCoordinateSystem>
  <TwoDCoordinateSystemName>
   <value>'WRS2'</value>
  </TwoDCoordinateSystemName>
  <coordinate1>
   <value>15<value/>
  </coordinate1>
  <coordinate2>
   <range lower='33' upper='42'/>
  </coordinate2>
 </TwoDCoordinateSystem >
</granuleCondition>
```

ProducerGranuleID
Use **ProducerGranuleID** to specify an inventory search for specific granules. **ProducerGranuleID** represents the ID assigned to data by the Science team that produced the data. Instead of using the ECHO-specific IDs for the granules, this criterion allows the user to use the producer's given name for the granules. Granule names are available to the Science teams and are easier to search on. Since the granule names given by the producer are not guaranteed to be unique within ECHO, if a granule with the same name exists for more than one **dataCenterId** specified in the query, all will be returned in the results. You must enclose the names of the granules in single quotation marks (that is, apostrophes). Note that the ProducerGranuleID can contain value, list, textPattern or patternList children.

Code Listing 46: ProducerGranuleID Search Element

```
<granuleCondition>
 <ProducerGranuleID>
  <list>
   <value>'MOD03.A2000107.0930.003.2002056052717.hdf'</value>
   <value>'MOD01.A2000107.0740.003.2002056051321.hdf'</value>
  </list>
 </ProducerGranuleID>
</granuleCondition>
```

PGE Name

Use **PGEName** to search for granules based on the name of the product generation executive (PGE). You can search for a known value in a **value** element, a list of known values in a **list**, or a text pattern in a **textPattern**.

Code Listing 47: PGE Name Search Element

```
<granuleCondition>
 <PGEName>
  <value>'test_PGEName'</value>
 </PGEName>
</granuleCondition>
```

PGE Version

Use **PGEVersion** to search for granules based on the version of PGE that originally produced the granule. You can search for a known value in a **value** element, a list of known values in a **list**, or a text pattern in a **textPattern.**

Code Listing 48: PGE Version Search Element

```
<granuleCondition>
 <PGEVersion>
  <value>'3.0.0'</value>
 </PGEVersion>
</granuleCondition>
```

Collection Short Name

Use **collectionShortName** to search for granules based on the short name of the collection they belong to. You can search for a known value in a **value** element, a list of known values in a **list**, or a text pattern in a **textPattern**.

Code Listing 49: Collection Short Name Search Element

```
<granuleCondition>
 <collectionShortName>
  <value>'MOD03'</value>
 </collectionShortName>
</granuleCondition>
```

Measured Parameters

Use **measuredParameters** to search for granules based on one or more groups of measured science parameters. You can specify the relationship between groups of parameters with either AND or OR.

An AND relationship means that result granules must match across all groups.

An OR relationship means that result granules can match any of the groups. OR is the default setting.
Within each measured parameter, you must specify a **measuredParameterName** in single quotation marks (that is, apostrophes) for exact matching. The detailed measured parameters can be **value**, **list**, and **textPattern** type, or **range** and **dateRange** type.
This example shows a search that specifies an "AND" relationship between two measured parameter groups.

Code Listing 50: Measured Parameter Search Element

```
<granuleCondition>
 <measuredParameters operator="AND">
  <measuredParameter>
   <measuredParameterName>'EV_1KM_RefSB'</measuredParameterName>
   <operationalQualityFlag>
    <value>'Passed'</value>
   </operationalQualityFlag>
   <QAPercentOutOfBoundsData> <range lower='30' upper='40'/>
   </QAPercentOutOfBoundsData>
  </measuredParameter>
  <measuredParameter>
   <measuredParameterName>'EV_1KM_RefSB'</measuredParameterName>
    <operationalQualityFlag>
     <value>'Passed'</value>
    </operationalQualityFlag>
   <QAPercentOutOfBoundsData>
    <range lower='39' upper='50'/>
   </QAPercentOutOfBoundsData>
  </measuredParameter>
 </measuredParameters>
</granuleCondition>
```

This example shows a search that specifies an "OR" relationship between two measured parameter groups.

Code Listing 51: Measured Parameter Search Element

```
<granuleCondition>
 <measuredParameters operator="OR">
  <measuredParameter>
   <measuredParameterName>'EV_1KM_RefSB'</measuredParameterName>
   <operationalQualityFlag>
    <value>'Passed'</value>
   </operationalQualityFlag>
   <QAPercentOutOfBoundsData> <range lower='30' upper='40'/>
   </QAPercentOutOfBoundsData>
  </measuredParameter>
  <measuredParameter>
   <measuredParameterName>'EV_1KM_RefSB'</measuredParameterName>
    <operationalQualityFlag>
     <value>'Passed'</value>
    </operationalQualityFlag>
   <QAPercentOutOfBoundsData>
    <range lower='39' upper='50'/>
   </QAPercentOutOfBoundsData>
  </measuredParameter>
 </measuredParameters>
</granuleCondition>
```

ProviderInsertDate

Use **providerInsertDate** to search for granules that are within a desired date-time range. **StartDate** is a required field, and **StopDate** is optional. The **Date** format is:

```
<Date YYYY="2001" MM="01" DD="01" HH="00" MI="00" SS="00"/>
```

Where YYYY is year; MM is month, DD is day of the month; HH is hour; MI is minute, and SS is second.
This example shows a search using **providerInsertDate**.

Code Listing 52: Provider Insert Date Search Element

```
<granuleCondition>
 <providerInsertDate>
  <dateRange>
   <startDate>
    <Date YYYY="2001" MM="01" DD="01" HH="00" MI="00" SS="00"/>
   </startDate>
   <stopDate>
    <Date YYYY="2003" MM="12" DD="01" HH="23" MI="59" SS="59"/>
   </stopDate>
  </dateRange>
 </providerInsertDate>
</granuleCondition>
```

ProviderProductionDate

Use **ProviderProductionDate** to search for granules that are within a desired date-time range. **StartDate** is a required field, and **StopDate** is optional. The **Date** format is:

```
<Date YYYY="2001" MM="01" DD="01" HH="00" MI="00" SS="00"/>
```

Where YYYY is year; MM is month, DD is day of the month; HH is hour; MI is minute, and SS is second.
This example shows a search of the Data Provider's production date search element.

Code Listing 53: Provider Production Date Search Element

```
<granuleCondition>
 <providerProductionDate>
  <dateRange>
   <startDate>
    <Date YYYY="2001" MM="01" DD="01" HH="00" MI="00" SS="00"/>
   </startDate>
   <stopDate>
    <Date YYYY="2003" MM="12" DD="01" HH="23" MI="59" SS="59"/>
   </stopDate>
  </dateRange>
 </providerProductionDate>
</granuleCondition>
```

Miscellaneous Elements

Use the following elements to search for other elements using AQL.

List

Use **List** to specify a list of values. The values may be numbers, dates, or strings. Each individual value must appear as text of the value element. If you specify a list of strings, you must use a single-quoted string with no wild-card characters. ECHO searches the parent element using Oracle's IN operator for an exact match with any value in the list.
In general, interpretation of the list will be dependent on the parent element in which it appears.

Code Listing 54: The List Element

```
<list>
 <value>'Africa'</value>
 <value>'Bermuda'</value>
 <value>'Indian Ocean'</value>
</list>
```

Pattern List

Use **patternList** to specify a list of text patterns and/or values. See section 6.8.4 for rules associated with values and section 6.8.3 for rules associated with text patterns.
The values may be numbers, dates, or strings. Each individual value must appear as text of the value element. If you specify a list of string values, you must use a single-quoted string with no wild-card characters.
The text patterns must conform to the rules specified in section 6.8.3.
ECHO searches the parent element using Oracle's IN operator for an exact match with any value in the list.
In general, interpretation of the list will be dependent on the parent element in which it appears.

```
<patternList>
 <value>'Africa'</value>
  <textPattern operator="LIKE"
caseInsensitive="Y">'america%'</textPattern>
 <value>'Indian Ocean'</value>
</patternList>
```

Text Pattern
Use **textPattern** to search using the Oracle operator LIKE to perform pattern matching. Because of pattern matching, exact specification of what is desired is not needed. The code listing below shows how to invoke the pattern matching. Note that you must specify the operator, and you must use a single-quoted string with or without wild-card characters.
The wild-card characters supported are '%' and *''. The character %' is a placeholder that represents any number of characters. The character* is a placeholder that represents a single character.

Table 7: AQL Wildcard Use

Wildcard pattern	Interpretation
%JOHN%'	Strings containing JOHN' as a sub-string
MARY%'	Strings beginning with the characters MARY'
%BOB'	Strings ending in BOB'
D_VE'	Strings like DAVE', DOVE', etc.

You can include the actual characters "%" or "*" in the pattern by using the ESCAPE character '\'. If the escape character appears in the pattern before the character "%" or "*", then Oracle interprets this character literally in the pattern, rather than as a special pattern-matching character.

Table 8: Wildcard Examples with ESCAPE Character

Wildcard pattern with ESCAPE character	Interpretation
JOHN%THAN'	String JOHN%THAN'
JOHN%THAN'	Strings JOHNNATHAN', JOHNASDTHAN', etc.
JOHN\\MARY'	String JOHN\MARY'

If the parent element (**collectionCondition** or **granuleCondition**) has the attribute negated with value 'Y', then the search will look for a string NOT like the pattern specified.

Code Listing 55: Text Pattern Element

```
<spatialKeywords>
 <textPattern operator="LIKE"
caseInsensitive="Y">'america%'</textPattern>
</spatialKeywords>
```

Value
Use **value** to specify one value. Refer to each individual parent element for a detailed explanation. In general, if the element represents a string pattern, the search will be case sensitive unless the **caseInsensitive** attribute is set to Y.

```
<value>12.6</value>
```

or

```
<value caseInsensitive="Y">'Asx'</value>
```

Date
Use **Date** to specify a date value.

```
<Date YYYY="2000" MM="05" DD="03"/>
```

or

```
<Date YYYY="2000" MM="10" DD="09" HH="21" MI="04" SS="32"/>
```

Date Range
Use **dateRange** to specify a range of time. The range can include just a **startDate**, just a **stopDate** or both. If you only include a **startDate**, ECHO searches for data from that time forward with no stop date. If you only include a **stopDate**, ECHO searches for data from the current time up until the stop date. If you include both a **startDate** and a **stopDate**, then ECHO searches for data falling between those two dates, for example, between January 1, 2005 and January 31, 2005.

```
<dateRange>
 <startDate><Date YYYY="2005" MM="01" DD="01"/></startDate>
 <stopDate><Date YYYY="2005" MM="01" DD="31"/></stopDate>
</dateRange>
```

or

```
<dateRange>
 <stopDate>
  <Date YYYY="2000" MM="10" DD="09" HH="04" MI="12" SS="34"/>
 </stopDate>
</dateRange>
```

DifEntryId
Use **DifEntryId** to find collections by their GCMD (Global Change Master Directory) DIF (Directory Interchange Format) Entry-ID. You can find the GCMD at the following URL: http://gcmd.nasa.gov.