

Add a dispose method to DataStore API

Motivation:	Add a close method to DataStore API
Contact:	Andrea Aime
Tracker:	http://jira.codehaus.org/browse/GEOT-1520
Tagline:	

Description

The DataStore API is missing a dispose() method that would allow the various datastores holding some kind of resource reference to release it. Notable examples are:

- ArcSDE and JDBC datastores, that hold onto a connection pool
- caching datastore wrappers, that hold onto a in memory or disk based cache

Status

This proposal is ready for discussion.

Voting:

- [Andrea Aime](#) +1
- [Ian Turton](#) +1
- [Justin Deoliveira](#) +1
- [Jody Garnett](#) +1
- [Martin Desruisseaux](#) +0
- [Simone Giannecchini](#) +1

Tasks

	no progress		done		impeded		lack mandate /funds/time		volunteer needed
--	-------------	---	------	---	---------	---	--------------------------	---	------------------

	2.4.0-RC1	
	Andrea Aime	API changed based on BEFORE / AFTER
	Andrea Aime	Provide an empty stub in datastore base classes so that most implementations aren't influenced by the change

	Andrea Aime	Update JDBC data stores so that they close their connection pool
	Saul Farber	Update ArcSDE data store so that it closes its own connection pool
	2.5.0-M1	
	Andrea Aime	Forward port above changes
	Saul Farber	Forward port ArcSDE changes
	Andrea Aime	Update demos/example and user guide

API Changes

BEFORE

Before datastore users could only release the connection and hope the datastore would clean up after itself somehow (usually letting the garbage collector tear down whatever resource was held into memory):

```
DataStore myDataStore = ...;  
    // use datastore  
    ...  
    myDataStore = null;  
    System.gc(); System.gc(); System.gc(); //  
clean up pretty please!!!
```

AFTER

The proposed change is:

```
public interface DataStore {
    ...
    /**
     * Releases all resources eventually held
     by this DataStore. The DataStore and all
     objects
     * generated out of it are not supposed to
     be working anymore after this method is
     called.
     * This call provides no thread safety
     guarantees, making sure nothing else goes on
     while
     * closing the datastore is a
     responsibility of the client code.
     * Subsequent calls to this method will be
     treated as no-ops.
     */
    public void dispose();
}
```

Sample use:

```
DataStore myDataStore = null;
try {
    myDataStore = ...;
    // use data store
    ...
} finally {
    if(myDataStore != null)
        myDataStore.dispose();
}
```

Documentation Changes

Please list the pages effected by this proposal.

Website:

- Update [Upgrade to 2.4](#) instructions

User Guide:

- Update examples to reflect changes to the DataStore API

User Manual:

- Update examples to reflect changes to the DataStore API

Issue Tracker:

- check related issues to see of problems are affected