Data Transformation Extras

The methods discussed in the previous section were all that is required to implement a view level converter. However, some extra work is required in order for your new converter to be available as a Data Transformation type conversion option, and is related to consolidating Data Transformation data types with UserInputParameters data types and when they should be available.

Another caveat is that only those converters will be used that support Data Transformation data types, (referred to as ETLDataType in Yellowfin.) Currently, these types are TYPE_TEXT, TYPE_NUMERIC, TYPE_DATE, and TYPE_TIMESTAMP and map directly to each ETLDataType of the same name.

public List<Integer> getAcceptedNativeTypes()

This method should return any data types which are supported by your converter, and should return the same values which would be accepted by acceptsNativeType(int) in the previous section. This is a shorthand for Data Transformation to efficiently select converters based on the data types that they support.

Example:

```
@Override
public List<Integer> getAcceptedNativeTypes() {
   List<Integer> acceptedNativeTypes = new LinkedList<>();
   acceptedNativeTypes.add(UserInpuptParameters.TYPE_TEXT);
   return acceptedNativeTypes;
}
```

Previous topic: Converter implementation
Next topic: Configuration UI