In order to use Analysis Services as a regular data source, you must perform the following steps:

1. Add a new OLE DB data source transformation to your data flow.
2. Ensure that your OLE DB data source uses the Microsoft OLE DB Provider for Analysis Services 10.0 option (located under the .Net Providers for OleDb" collection of providers).
3. Set the Data Access Mode property of your Data Source to SQL Command, which allows you to type in your query.
4. Type in an Analysis Services query (such as the DMX statement in Listing 14-6).
5. Click the Parse Query button to ensure the validity of the statement.

Now you can use the new OLE DB data source just like any other data source and apply any data flow transformations on top of the results returned by Analysis Services.

Text Mining Transformations

This section examines two particularly interesting data flow transformations that facilitate text mining: Term Extraction and Term Lookup.

SQL Server Data Mining supports the TEXT data type, but that data type is not enough to perform meaningful text analysis. From the algorithm’s perspective, columns having the TEXT data type are treated just like discrete columns that have the LONG data type — as a collection of various distinct states, without any way to directly access the content of a text value.

To perform text mining with SQL Server Data Mining, you must first bring the text to some form that can be consumed by the algorithms. The solution included in the product is to represent each piece of text as a collection of words and phrases, and perform data mining based on the occurrence of certain key words and phrases inside a certain document (and possibly some frequency-related scores). Therefore, a document is modeled very similarly to a shopping basket that contains (or does not contain) certain items (which happen to be key words and phrases).

After each document is represented as a collection of key phrases, you can perform data mining using one of the following model types:

- Classification models that use the key words and phrases nested table as input to predict the class of a document
- Clustering models that find similar documents based on common occurrences
- Association models that detect cross-correlations between key words and phrases
The process of text mining usually consists of at least the following three phases:

1. Build a dictionary of key words and phrases over a collection of representative documents. This task is usually accomplished using the Term Extraction transformation.

2. Based on the dictionary, extract the list of significant key words and phrases for each document to be analyzed. This task is usually accomplished using the Term Lookup transformation.

3. Train mining models on top of the transformed data.

**NOTE** If you intend to perform predictions on the mining models built in phase 3, you must convert any new document to the same representation (that is, run the Term Lookup transformation on new documents using the same dictionary used in training the mining model or models).

**Term Extraction Transformation**

You use the Term Extraction transformation to build a glossary of key words and phrases for a specific domain. This is usually the first step of a text mining project. The transformation applies to a pipeline that already contains one column with text data of type ntext or nvarchar. The purpose of the transformation is to analyze this column and build a dictionary of key terms based on its content. The output of the transformation is a table with a single column. Extracting key terms is not a trivial task because it involves sophisticated techniques, such as word stemming and grammar parsing. The transformation extracts nouns and noun phrases, such as *data mining*.

You can download the data used for the examples in this section from www.infoplease.com/t/hist/state-of-the-union/. This download contains the text for 219 State of the Union addresses of U.S. Presidents between 1790 and 2006.

The user interface of the Term Extraction transformation is quite simple. In the first tab of the Term Extraction Transformation Editor shown in Figure 14-16 (Term Extraction), you must specify the textual column. You can also name the output columns for key terms and their associated scores. The score is based on Term Frequency and Inverse Document Frequency (TFIDF). This is a statistical technique used to evaluate how important a word is to a document. The importance increases proportionally to the number of times a word appears in the document, but it’s offset by how common the word is in all of the documents in the collection.

The Exclusion tab shown in Figure 14-17 provides the option to specify inclusion and exclusion terms. You may already have a list of predefined terms that must be included, as well as a list of terms that you don’t want extracted. You can specify these two term lists in this tab.
The Advanced tab of the Term Extraction Transformation Editor (shown in Figure 14-18) also provides options for the terms. For example, you can specify that terms must be single words or noun phrases. In the case of a noun phrase, you can specify the maximum length. You can also choose the type of score to be computed (Frequency or TFIDF), as well as a minimum frequency threshold and the maximum length (in words) of any single term (phrase). In addition, you can choose whether the term extraction should be case-sensitive or not (the default setting is case-insensitive).

After you have configured the term extraction transformation, you should bind its output (in the pipeline) to a relational table destination that will store the dictionary of terms. The table will contain two columns: one containing the terms and the other containing the score associated with each term. Figure 14-19 shows the full transformation used to extract the terms from the State of the Union addresses. The Data Conversion transformation that appears in the pipeline before the Term Extraction transformation converts the text from the original database format (ASCII text) to one of the formats supported by Term Extraction (UNICODE text).
After a dictionary is built using the Term Extraction transformation, each document to be analyzed must be transformed to a collection of terms, based on that dictionary. The Term Lookup transformation is used to search for key terms from the input textual column, based on a dictionary. The dictionary is usually generated by the Term Extraction transformation. Because the dictionary is just a table, you can write SQL queries to modify the list by adding or removing terms when necessary.

Just like the Term Extraction transformation, the Term Lookup transformation requires the input text column in the pipeline to have the ntext and/or nchar data types. The editor for the Term Lookup transformation is quite simple. The first tab, Reference Table, is used for specifying the reference table (that is, a dictionary produced by the Term Extraction transformation), as shown in Figure 14-20.

The Term Lookup tab shown in Figure 14-21 is used to specify the column mapping (for mapping the input text column to the Term column dictionary). You can also pass through some input columns, such as President and Year in this example. Pass-through columns are accessible to the rest of the data flow. Note that the actual input text column (TextBody ASNTEXT in this example) does not need to be passed through.
The Advanced tab of the Term Lookup Transformation Editor allows you to specify whether the lookup should be case-sensitive or not. Just like the Term Extraction transformation, Term Lookup is case-insensitive by default.

The Term Lookup transformation produces two new columns as output: Term and Frequency. You can think of the output of this transformation as a fact table with a large number of rows containing the document ID, key terms, and associated frequency. Figure 14-22 shows the full transformation used to look up terms from the State of the Union addresses.

**NOTE** If your data flow destination is a relational table that holds the terms associated with each document, then you must ensure that a document key appears in that table (such as a document ID column or the Year column in the current example). This key will be used later in modeling the nested table relationship between the documents table and the terms table.
Figure 14-19 Complete term extraction data flow

Figure 14-20 Reference Table tab in the Term Lookup Transformation Editor
More Details on the Text Mining Process

Let’s review the steps performed using the Term Extraction and Term Lookup transformations. You started with a relational table containing one large text column, the `TextBody` column, plus some additional columns, as shown in Table 14-4.

The Term Extraction transformation selected a set of significant terms from the `TextBody` column of the table. The dictionary contains the most significant terms and a score associated with each term, as shown in Table 14-5.

In the next step, the Term Lookup transformation detected the significant terms in each document (based on the dictionary in Table 14-5) and placed them in a new table, together with the document identifier (in this case, the `Year` column was used as a document key). Table 14-6 shows a fragment of the resulting table.
Figure 14-22 Complete term lookup data flow

Table 14-4 Original StateOfTheUnion Table

<table>
<thead>
<tr>
<th>PRESIDENT</th>
<th>YEAR</th>
<th>TEXTBODY</th>
<th>POLITICAL PARTY</th>
</tr>
</thead>
<tbody>
<tr>
<td>George Washington</td>
<td>1790</td>
<td>I embrace with great satisfaction . . .</td>
<td>No party</td>
</tr>
<tr>
<td>George Washington</td>
<td>1790 Dec</td>
<td>In meeting you again I feel . . .</td>
<td>No party</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Jimmy Carter</td>
<td>1981</td>
<td>To the Congress of the United States: . . .</td>
<td>Democratic</td>
</tr>
<tr>
<td>Ronald Reagan</td>
<td>1982</td>
<td>Mr. Speaker, Mr. President, Distinguished . . .</td>
<td>Republican</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

The StateOfTheUnionTerms table can now be used as a nested table in a model that analyzes the documents in the original StateOfTheUnion table. Figure 14-23 shows an example of a mining structure designed to perform text mining on this data set. The structure uses the StateOfTheUnion table as a case table (using Year as the key column and Political Party as a case level column) and the StateOfTheUnionTerms table generated by the Term Lookup transform as a nested table. For simplicity, the frequency of a term in a State of the Union address is not included in the model.
In the mining structure, you can now add multiple models that apply different analysis techniques to the documents. Figure 14-24 shows the output of an Association Rules mining model that analyzes the cross-correlations of terms in State of the Union addresses.

**Summary**

In this chapter, you learned the basic concepts of another important SQL Server component: SQL Server Integration Services (SSIS). You were introduced to control flow and data flow, and learned about a few important tasks and transformations. SSIS is an important tool for data cleaning and transformation, which is a time-consuming step for any data mining project.
Figure 14-23 A mining structure designed to text mine the State of the Union data set

Figure 14-24 Output of an Association Rules mining model
The second half of the chapter focused on the data mining–specific features in the SSIS environment. You learned about each of the data mining–related tasks and transformations and saw some examples of control flow and data flow using these data mining tasks and transformations. Finally, you learned about the two text mining–related transformations and saw a typical text mining project example based on SQL Server 2008.

Data mining and SSIS are mutually beneficial. SSIS provides a data processing environment for data mining, and data mining techniques can be used as part of a data transformation process. This makes SSIS smarter, placing it ahead of other classic ETL products.

By now you should have a clear understanding of the relationship between SSIS and data mining, as well as the type of data mining projects that you can complete in the SSIS environment.

Stay tuned, as Chapter 15 introduces you to some architectural details for SQL Server Data Mining.