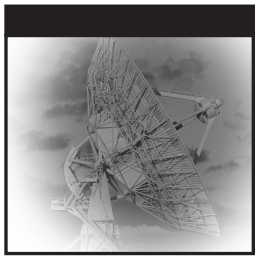


Mastering the SAP® Business Information Warehouse

Kevin McDonald
Andreas Wilmsmeier
David C. Dixon
W.H. Inmon

Wiley Publishing, Inc.

Information Access, Analysis, and Presentation

The old saying that you can lead a horse to water but you can't make it drink is particularly apropos when discussing business intelligence tools. Information access and presentation is a topic that is often the center of great debate and controversy. Even if a tool delivers the right information to the right user at the right time, there is no guarantee that the user will use the tool. No single information access tool can satisfy an organization's or even an individual's requirements. It is not uncommon for each department to insist on using a particular tool because of its unique requirements. SAP BW has and supports a broad range of presentation and access tools that turn data into information and deliver it to the desired consumer.

In Chapter 6 we described how to integrate and transform data so it may be stored in SAP BW storage constructs such as ODS objects, InfoCubes, and master data. This chapter picks up where Chapter 6 left off. Here we'll highlight the main services provided in SAP BW that retrieve data, turn it into meaningful business information, and deliver that information to an information consumer. The chapter has been organized into two main sections: SAP BW information access and analysis services and SAP BW presentation services.

Architecture

SAP BW presentation services layer includes all the components required to present information available on the SAP BW server in the traditional Microsoft Excel-based Business Explorer Analyzer, in the Business Explorer Web Environment, or in third-party applications. Figure 7.1 illustrates the main architectural layers of SAP BW. In this chapter, we'll build on the previous chapters and start with the presentation services layer then. Then we'll move into the information access and analysis services layer. We will also highlight the interfaces that are exposed to third-party reporting and presentation tools.

Query Processing Overview

The query process in SAP BW is a series of requests and responses, including requests for information, database selections, application caching, number crunching, information formatting, and ultimately responding to the requester by presenting results sets. From an end user's perspective this process has been abstracted to the point that the end user's only concern is making a request for information that will lead him or her to solve the particular business problem at hand. However, behind the scenes, SAP BW is busy at work creating the optimal database access plan, locating aggregates, converting currency, applying hierarchies, filtering, and so on.

Let's look at the query request response process in the context of a typical business question. We will use the example of analyzing revenue and contribution margins across multiple geographies and customers to illustrate how the information request response process works in SAP BW. Figure 7.2 depicts the analysis process in SAP BW.

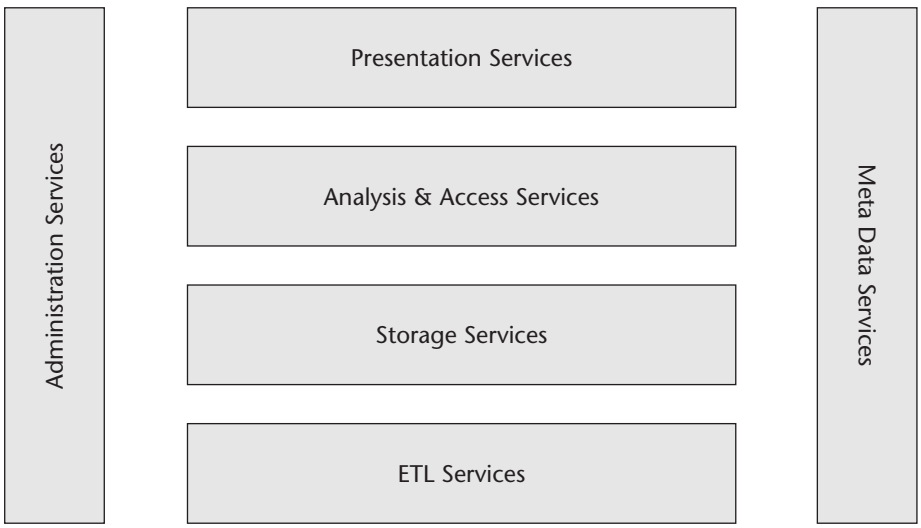


Figure 7.1 Main architectural levels of SAP BW.

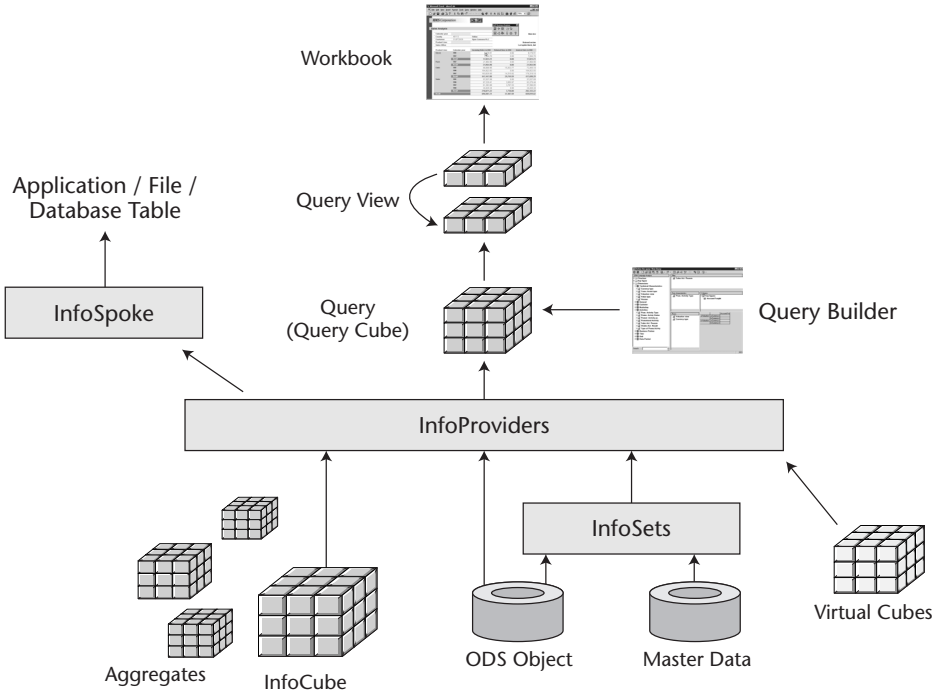


Figure 7.2 Analysis processing.

In our scenario, we'll assume that this is the first time the corporate controller has requested contribution margin and revenue to be analyzed by geography and customer. After launching her favorite Web browser and navigating to the SAP BW reporting homepage on the corporate intranet, the controller selects the *Create New Query* option. After selecting this option, she invokes the meta data services of SAP BW to return a list of potential information providers that may be queried. The controller selects the appropriate InfoProvider, and again invokes the meta data services in SAP BW. The services return a list of dimensions that contain characteristic InfoObjects as well as a list of key figures. The controller selects the *Revenue and Contribution* key figure, as well as the *Geography* and *Customer* characteristics, and assigns them to the rows or columns of the query result sets based on her preference. She then executes the query.

At the center of the information access, analysis, and presentation services in SAP BW is the OLAP engine. An information consumer application requests information from the OLAP engine in the form of a multidimensional expression or similar selection request. In our example the controller requests revenue and contribution margin for the current period and fiscal year for all customers and all geographies. Leading business intelligence tools generate selection expressions so the end users do not need to know the specific syntax. The request is sent from the browser to the SAP BW server. Consumers may be a browser, mobile device, Excel spreadsheet, or as in our example, a third-party client application. The SAP BW server takes the request through one of a

number of standard interface techniques and hands it to the OLAP engine for fulfillment. (We will describe the interfaces in detail later in this chapter.)

The SAP BW server takes the request and determines if there is an aggregate cube in existence that may satisfy the request. In our example an aggregate cube that summarizes customer or geographies may be a candidate for optimal retrieval. If no such aggregate cubes exist for the query, the application server makes a request to the database server.

NOTE SAP BW attempts to limit the number of round trips from presentation server to database server by caching and reusing query results and navigation steps on the application server in a cross-transactional buffer after it has been selected from the database. This caching technique was introduced in version 3.0b with the goal of sharing memory across user sessions.

The database requests and selects records from an InfoProvider, or an aggregate, that is stored in a relational database or multidimensional database for aggregates as a star schema. The records that are selected from the database server may be returned to the application server. Depending on the query's read mode settings, records are cached for the OLAP engine to process and, ultimately, to calculate and return a query navigation state to the presentation server. The caching of the selected data on the application server is sometimes referred to as a *query cube* depending on the read mode that is set for the query. The OLAP engine will use the query cube to calculate and return result sets to the client for subsequent information requests, assuming the query cube contains the necessary data to respond to the request. We will discuss the options for setting a query's read mode and the impact on performance in Chapter 10.

The controller in our example would receive a listing of revenue and contribution margins for all customers in all geographies. She notices that revenue, which is displayed in U.S. dollars (USD), is higher than she had planned and proceeds to investigate which geography is exceeding her expectations. Her request is sent from the client to the application server, where it is once again handled by the OLAP engine. Once the appropriate storage service returns the records, the OLAP engine creates the query data and proceeds as previously described. The controller learns that the United Kingdom is exceeding her sales expectations.

Note that the OLAP engine will take care of any currency translations based on the meta data of the InfoObjects included in the query, as well as the meta data of the query itself. In this case the geographies are countries, and most countries have their own currency. The controller also may wish to analyze the revenue in the group currency of USD. Upon doing so, she notices that Argentina has failed to meet her sales expectation. Not satisfied she has found the cause of the shortfall, she investigates further and requests the currency be converted to local currency for all countries. A request is once again sent to the OLAP engine to be processed. The local currencies are either retrieved

or calculated, depending on the modeling of the InfoMart, and the newly calculated query slice is returned to the client. Now the controller realizes that Argentina did in fact meet sales expectation but experienced a significant foreign exchange impact as a result of the rate dropping.

The process of requesting and responding to queries is the central concept of this chapter. We will first investigate the presentation services and then move to the information access and analysis services. It is not our intent here to re-create SAP documentation, nor is it possible for us to cover every reporting and analysis feature of SAP BW. However, we will look behind the scenes at the central services SAP BW performs and lay out the possible options for putting them to use to solve business problems.

Presentation Services

SAP BW presentation services layer includes all components required to present information available on the SAP BW server, including the traditional Microsoft Excel-based Business Explorer Analyzer, the Business Explorer Web applications, and Business Explorer Formatted Reporting. Mobile device and Enterprise Portal support are rendered from the same set of Business Explorer Web services as the Business Explorer Web applications. Figure 7.3 provides an overview of the components of the presentation services layer.

The line between the BEx Query Designer and BEx Analyzer is often blurred, as the tools are tightly integrated and users of the BEx Analyzer are often also query designers. There are numerous features that are often described as part of the BEx Analyzer when, in fact, they are part of the BEx Query Designer. Rather than split hairs, we have separated the Designer's functions and Analyzer's functions based on where the specific features are used in the process of creating and analyzing.

Business Explorer Components

The Business Explorer (BEx) is much more than an Excel add-in that allows access to the SAP BW server. There are several areas that the Business Explorer in SAP BW now covers—for example, support for formatted reporting, mobile devices, and pure HTML-based Web applications. The Business Explorer's integration with SAP Enterprise Portals enables a single point of entry for a wide spectrum of end-user roles and business-related information packs. This enables collaboration and integration with nonstructured information such as documents that capture users' comments on such things as the explanation of variances, justification for changes in forecast figures, graphics, syndicated information available via the Internet, and the ability to unify the analytics served by SAP BW with transaction processing applications. This functionality allows end users to dynamically create personalized analytic applications.

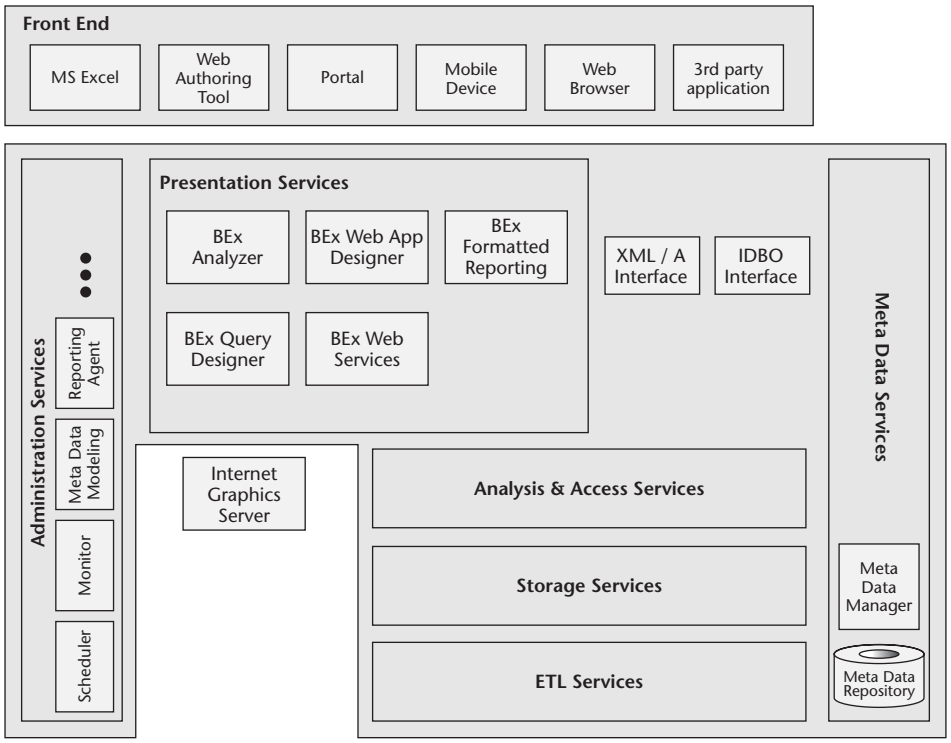


Figure 7.3 Presentation services.

In Figure 7.3 you will notice that the Business Explorer consists of several components: BEx Query Designer, BEx Analyzer, BEx Web Application Designer, BEx Formatted Reporting, and BEx Mobile Device Support. The BEx is designed to meet a vast range of business needs, from simple list reports (e.g., all the customers in a range of zip codes) to the complex (e.g., elimination of intrabusiness volume), while taking into account the local currencies and fiscal year variants of the business entities. The BEx is designed to appeal to a wide range of business users, from hunters and farmer to miners and explorers. Various end users will interact with the BEx tools. Farmers, for example, may request information via a Web browser by entering a URL. Explorers, on the other hand, may use the BEx Analyzer to create and re-create complex queries based on their findings. (For an explanation of the different group types, see Chapter 5.)

In this next section we describe BEx queries from design to execution, along with powerful OLAP functions that may not be obvious to the casual or first-time user. Throughout this chapter we refer to each of the five Business Explorer components by their specific names.

BEx Query Designer

All multidimensional reporting and analysis performed in SAP BW is based on a query definition stored in the Meta Data Repository. Queries provide access to multidimensional information providers (InfoCubes), as well as flat information providers (ODS

objects, master data). The *Business Explorer Query Designer* allows you to define queries in an interactive standalone application by simply dragging and dropping the desired meta data objects into the query results area.

A *query* is a specification of a certain dynamic view on an InfoProvider used for multidimensional navigation. Queries are the basis for all kinds of analysis and reporting functionality available in SAP BW.

The BEx Query Designer is a standalone tool for defining queries. For some readers, *query* may conjure up the image of an SQL generator that creates a simple list of records. The BEx Query Designer is a graphical tool for defining both tabular queries and multidimensional queries that access powerful OLAP functions. The BEx Query Designer, while a standalone client program, interacts with the SAP BW server, more specifically the Meta Data Repository. Meta data about a query's InfoProvider is passed to the Query Designer so the form and function of a query may be defined and ready for ad hoc execution. Note that only one InfoProvider may be assigned to a query. If more than one InfoProvider is needed, a MultiProvider has to be used. We have covered this topic in Chapter 5.

All characteristics, navigational attributes, and key figures available through an InfoProvider are available for use in query definitions. Because queries are multidimensional objects, they effectively define subcubes called *query cubes* on top of the InfoProvider. Query cubes define the degree of freedom available for query navigation in the presentation layer.

To define a query, the Designer is launched in one of five ways:

- As a client application
- From the Web Application Designer
- Via the HTML Query Designer
- Via the traditional BEx Analyzer in Excel
- Via the Crystal Reports Designer

The first option the query designer has is selecting an existing query from a list of favorites or from a list of queries that have been assigned to the specific role the designer has in the organization. If the query does not exist, a new one may be created. Next, an InfoProvider for the new query is selected. The InfoProvider, as we will discuss further later in this chapter, is an abstraction layer that allows the query designer to define both tabular and multidimensional queries with the same tool without regard to how or where the physical data is stored. Upon selecting an InfoProvider, the query designer will see a list of meta data defining the following elements:

Structures. Predefined selection and layout criteria for a row or column that may be reused in all queries for a particular InfoProvider. Structures may contain a combination of key figures, characteristics, and formulas. A *reusable structure* is a particular, commonly used collection of key figures or characteristics stored in the Meta Data Repository for reuse in multiple queries (e.g., a plan/actual variance or a contribution margin schema).

Key figures. A type of InfoObject that is used to record quantitative facts or measures. All of the key figures for a particular InfoProvider are available for queries. A *calculated key figure* is a formula consisting of basic, restricted, or other

calculated key figures available in the InfoProvider stored in the Meta Data Repository for reuse in multiple queries (e.g., an average discount rate). A *restricted key figure* is a key figure with an associated filter on certain characteristic values stored in the Meta Data Repository for reuse in multiple queries (e.g., year-to-date sales of previous year). A query consists of meta data elements arranged in rows, columns, and free characteristics.

Dimensions. The logical grouping of characteristic InfoObjects in InfoCubes.

The query elements assigned to the rows and columns are displayed in the initial query view. Free characteristics are not displayed in the initial query view. Free characteristics are available for navigation in the BEx Analyzer or in analytic Web applications. Each individual navigational step (drill down, drill across, add/remove filters, etc.) in the analysis process provides a different query view, and the steps are controlled by the BEx Analyzer, BEx Web applications, or third-party tools.

Query definitions are created and maintained in the BEx Query Designer by simply dragging the available query elements into the rows, columns, free characteristics, or filter areas and eventually defining additional properties. The Query Designer also integrates all functionality required to define the query elements. Query elements include characteristics, key figures, calculated key figures (formulas), restricted key figures, and reusable structures. Queries may have filters on characteristic values or filters on key figure values (conditions) assigned to select a certain slice of information from the InfoProvider, and they may be parameterized by query variables. Exceptions assigned to a query help identify key figure values regarded exceptional from a business point of view.

The simplest type of query is a tabular query. *Tabular queries* are often used to generate listings of master data, such as a listing of all customers in the state of Illinois. For instance, you could create such a query by locating the InfoObject for the characteristic *State* and dragging and dropping it to the column window. You could then right-click on the state to restrict the query to just the state of Illinois. You complete this simple tabular query by dragging and dropping the characteristic *Customer* to the column window.

NOTE Defining selection values that are valid for all the columns and rows of a query in the filter window will improve query performance.

Multidimensional queries provide more options but still are easy to define and use. Figure 7.4 illustrates the options a query designer is presented when working with multidimensional queries. Along with the rows definition window there is also a Free Characteristics window. In multidimensional mode, the query creator drags and drops key figures and the characteristics desired in the query from a dimension to the desired column or row. As the InfoObjects are placed in a row or column a preview of the query results is displayed. The designer may also place characteristics in the Free Characteristic or *Filter* window. Free characteristics are selected as part of the query's logic but are not displayed in the default view of the query. They may, however, be used for drill-down and drill-across functions by the user of the BEx Analyzer or Web application. You use filters to limit the selection of data to a particular characteristic value or set of values. For example, the query "analyze revenue generated across all product

lines that were sold to customers in Illinois during the month of July in the year 2002" may have three filter values: the state of Illinois, the month of July, and the year 2002. If you are familiar with any of the leading OLAP tools, you will notice a great deal of parity in the basic functions of SAP BW multidimensional queries.

Several good sources of information on SAP Web sites are available that detail the basic steps for building queries, so we will not repeat that information here. The BEx Query Designer is simple enough to learn with a little practice. However, the key success factor is in understanding the information model. With this in mind, instead of the absolute basics, we will focus on the functions that may not be obvious to the novice query designer. We have recognized a plateau that self-taught BEx query designers and users reach in their learning. In this next section we'll help those readers reach the summit.

NOTE **InfoCubes, ODS objects, InfoSets, and master data are available to the query designer for either tabular or multidimensional analysis.**

Designing Queries with Hierarchies

SAP BW hierarchies were introduced in Chapter 3 and expanded on in Chapter 4 as we detailed the impact hierarchies have on information modeling. Hierarchies are used for data modeling, restricting data selection, navigation, and planning. For example, there are hierarchies for cost centers, profit centers, general ledger accounts, customers, and products, to name a few. In most reporting and analysis applications the concept of hierarchies exists. While it is not a new concept, it is an important one, and hierarchies are often the means of defining aggregation levels. SAP BW supports two kinds of hierarchies: internal and external. *Internal hierarchies* are modeled into a specific dimension. *External hierarchies* in SAP BW are called external because they are neither stored in InfoCube dimensions nor as navigational attributes; instead, they are stored in separate hierarchy tables associated to the base InfoObject (e.g., 0CUSTOMER for a customer hierarchy).

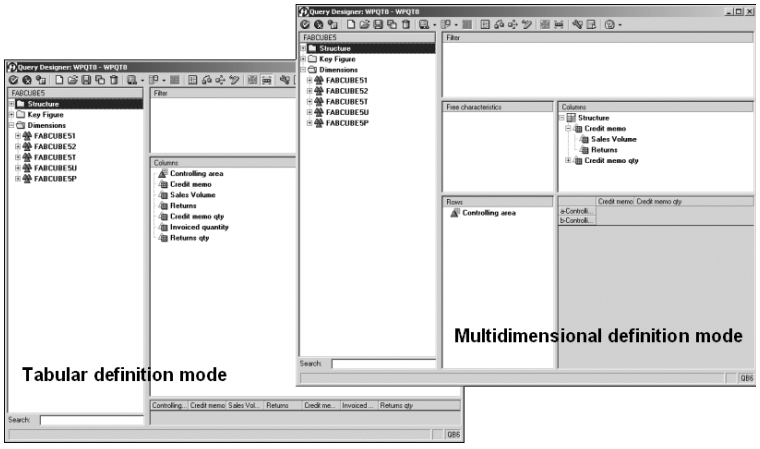


Figure 7.4 BEx Query Designer: tabular and multidimensional modes.

As mentioned in Chapter 4 there are several options for modeling hierarchies, including version and time dependency for the whole hierarchy tree, time dependency for the hierarchy structure, and the use of intervals instead of single values to assign a range of values to a specific hierarchy node. The hierarchy data model provides a means to store balanced and unbalanced hierarchies with different types of hierarchy nodes available at different levels or network structured hierarchies. These elements provide a tremendous amount of flexibility to the information modeler and query designer. Hierarchies may be explicitly assigned to a query at design time or may be entered by the query user at run time when a hierarchy variable is used.

A query designer may set the hierarchy as fixed for a given query, thereby eliminating the possibility that a user of the BEx Analyzer or a BEx Web application may switch to an alternative hierarchy. The query designer may go so far as to fix the query to a given node within a given hierarchy. If the hierarchy that is assigned to a characteristic is time- and/or version-dependent, the query designer may enable the query user to enter three variable properties for the hierarchy. The hierarchy name, version, and key date for data selection may all be prompted at query run time, providing the maximum amount of flexibility.

The query designer sets the display attributes for the hierarchy. When a query is executed in the BEx Analyzer the hierarchy is in either the rows or columns depending on the placement of the underlying characteristic. The hierarchy may be expanded and collapsed at its node levels. Each node level may contain an optionally displayed results line. The result line, if calculated, is defined by the query designer, but it may be changed in an ad hoc manner in the BEx Analyzer. The manner in which the result line is calculated may be set using the *Calculate* function on the right mouse context menu. The result line may be displayed as the min, max, first, last, variance, count, average, or standard deviation of the underlying data in the query cube. Values that are selected as part of the query view that do not have a corresponding external hierarchy node assignment will be displayed as *Not Assigned*. Note that each node level in a hierarchy can have its own settings for how aggregation should be performed, displayed, and used in calculating results rows. This allows for one node level in a hierarchy to be used as a negative value. For example, if a hierarchy is created for general ledger accounts, and all of the revenue accounts summarize to a node level, that node level may have a results line that is a negative number. This may be the case if your data model is tracking both debits and credits, since the revenue account is a credit in double-entry accounting. The hierarchy node may be set so the value is shown as a positive number.

A significant difference between SAP BW and other leading business intelligence tools is in the area of hierarchy handling. The Business Explorer toolset has a restriction that the hierarchy must already exist prior to the query being executed. Unlike many business intelligence tools on the market, the query user may not define hierarchies and nodes in an ad hoc manner on the fly while analyzing information. End users may, however, restrict the selection conditions of a predefined hierarchy at run time or select a new hierarchy for the query. Selecting the local query navigation window, selecting the characteristic that has the hierarchy assignment, and then right-clicking on the desired properties accomplishes this. The hierarchy properties window will appear and a new hierarchy may be selected by name, as illustrated in Figure 7.5. This is, of course, assuming the query designer has not set any restrictions on the hierarchy or node within a hierarchy that is to be used in a given query.

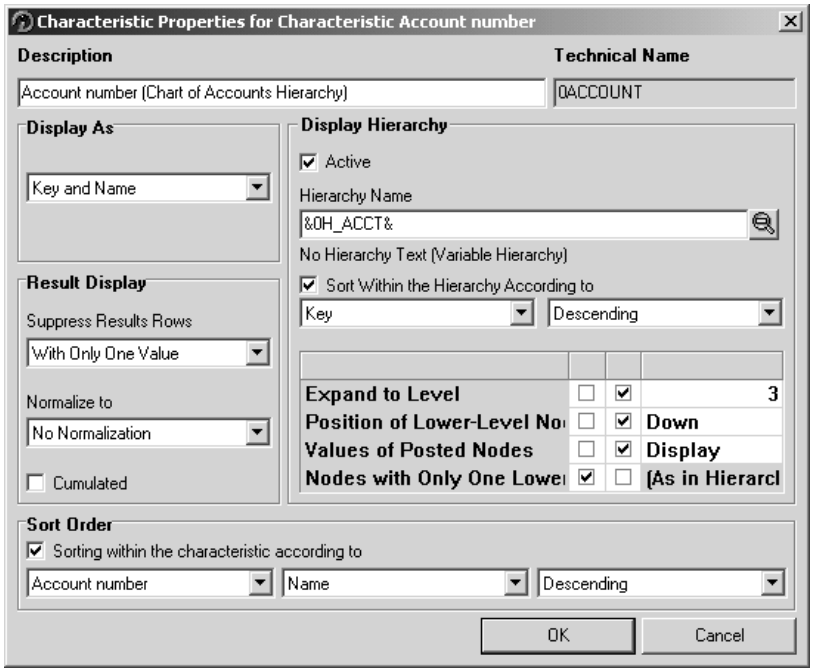


Figure 7.5 Hierarchy properties.

Variables

Variables are placeholders in a query definition that have their value defined at query run time, thus enabling the parameterization of queries. There are five types of variables: characteristic, text, formula, hierarchy, and hierarchy nodes. A classic example of the use of hierarchies is found in a common financial query to display a profit-and-loss statement. The variables enable the query designer to create the profit-and-loss statement once and allow the end user to determine at query run time the financial period for which the query should be run. In this example the characteristic fiscal period is entered by the end user or defaulted based on his or her personalization preferences.

Several processing types are available for the five variable types. The processing type sets the behavior of the variable. For example, if a variable has a processing type of "manual entry," a pop-up window will prompt the user to enter a value at the time of query execution. The five processing types are listed in Table 7.1. We will briefly describe the replacement path processing type, since it may not be self-evident. This will be done as we describe the five variable types, specifically the text variable. Note that variables need to be defined prior to being used in the BEx Query Designer and that the variables have a global scope. This means that once their properties are defined, those variable properties are inherited by every query that utilizes the variable and all variables are available for all queries. Once variables are input, personalization options may be set to prevent the user from entering the variable values more than one time.

Table 7.1 Variable Processing Types

PROCESSING TYPE	DESCRIPTION
Manual/default entry	Prompts the end user to enter a value at query run time.
Replacement path	Indicates that the value for the variable is to be found in the data of the query.
Authorization	Indicates that the value for the variable is stored with the user authorization.
Customer Exit	ABAP code that may be written by an SAP customer to fill a variable.
SAP Exit	ABAP code written by SAP to fill a variable value.

Characteristic variables are the most common type of variable and are used when an end user or group of end users would like to run the same query with a different set of parameter values, as described in the preceding example with profit-and-loss statement. The characteristic value is commonly defined with a processing type of manual/default with the settings *ready for input* so at query run time the characteristic value may be selected. The entry for a characteristic variable is not necessarily limited to one value. For example, say an IT director is responsible for three cost centers. The director may run the same query definition for displaying direct costs as a sales manager that is responsible for one cost center. The sales manager will be prompted for her cost center numbers and the IT director for his. The processing type authorization is an alternative to the manual entry processing type. The authorization processing type looks to the end users' authorization settings as defined with transaction RSMM and uses the value found there as input for the variable.

Text variables are commonly used in conjunction with a characteristic variable. In our profit-and-loss example, imagine there are two columns on the report, one column for the base period and another column for a forecasted period. When the end user of the report is prompted to enter a number for the base period to be analyzed, the value entered will populate the characteristic variable for the query and the base period column on the report will be populated with the profit-and-loss values for that period. A text variable may be used in this situation to change the column header text for the base period, so when "0012002" is entered for the characteristic value, the column header may display the text "Period 1 2002". The text variable is commonly defined with a processing type of *replacement path data*, so at query run time the characteristic value in the text variable's replacement path is used for the column header. Text variables using a replacement path have the option to use the characteristic value or the accompanying text value for a characteristic value. In our example, if the characteristic value is defined the column header text would be set to "0012002". Text variables do not support the processing type authorization or exit. The replacement path is most frequently used for text variables.

Formula variables allow numbers to be passed to the query at run time. This is especially useful for entering exchange, inflationary, or growth rates in a query at run time. This may be used for simulations where the formula variable value is passed into the calculations of a column, row, or cell. For example, the forecasted period in the previous example may take the base period and multiply the revenue for the base period by a factor that the user enters at query run time to calculate the projected revenue in the forecasted period. In this scenario, the end user would enter a value (e.g., 5 percent), and the formula in the forecasted period column would select the values in the base period column and multiply the values in the base period column by 1.05 to determine the forecasted period. Formula variables do not support the processing type authorization.

Hierarchy variables and *hierarchy node variables* behave in the same manner as the characteristic variables. The hierarchy variable represents an entire hierarchy tree for a given characteristic. The hierarchy node represents a given substructure within a hierarchy. A difference between the hierarchy and hierarchy node variables is the support of the authorization processing type. The hierarchy node is able to look to the settings in transaction RSMM for the default value for a given user. The hierarchy variable allows the query user to select entirely new hierarchies versus simply selecting a different node within the same hierarchy.

With the release of SAP BW version 3.0 there is a wizard for creating variables that guides query designers through the process of defining variables of all four types.

Conditional Analysis

Conditional result set processing in the BEx allows the end user to limit the information that is returned to the query view to only the records that meet specific conditions—for example, a condition may be set to only display the top 10 customers from a revenue perspective or the bottom 15 percent of all employees based on their latest performance review. Conditions are an efficient way to zoom in on a specific subset of information.

A single query may have one or more conditions assigned to it, and a condition may have multiple preconditions assigned to it. There are six condition types that are applied to levels of navigational states:

Top N

Top percentage

Top sum

Bottom N

Bottom percentage

Bottom sum

When applied, the *top percentage* and *bottom percentage* and *absolute count* condition types act as their names imply. They reduce the list to display an absolute number, say the top 10 customers, from our revenue example and display the results line for the amount of revenue for all customers or a percentage. The percentage option works in a similar manner. In our performance review example, the scores would be looked at as a percentage of total, and those with scores in the bottom 15 percent would be listed individually.

The *top sum* and *bottom sum* are a bit different from the percentage and absolute count. The top sum, for example, has a threshold set for a particular key figure. If we use the example of revenue as we did for the top 10 customers, this time we would set the condition not to the number of customers we are looking to analyze but to the amount of revenue for which we are looking; let's assume 100,000EUR. All of the customers would be sorted according to their revenues, in descending order. Then moving down the list, customers would be selected and listed in the condition up until the point that the 100,000EUR total revenue threshold was broken. The customer that breaks the threshold is included in the list.

Along with the ability to create ranking lists, query designers and analyzers can create absolute lists that set thresholds that will either include or exclude individual rows of a query based on whether or not they meet the condition. For example, to support a physical inventory, a query may be set with a condition that displays all inventory positions that have negative stock quantities. Note that conditions may also be defined for combinations of characteristics—for instance, "show me the top 10 product/customer combinations."

Exception Analysis

Exception reporting is a bit different than conditional filtering. Exceptions allow for predefined or ad hoc rules to be applied to a query that indicate how the information should be highlighted to show that a threshold has been achieved. Up to nine different colors may be assigned to values that cross an exception threshold. These colors, in hues of red, yellow, and green, may be used to identify deviations.

There are three main areas of exception reporting that you should be aware of: the setting of exceptions, the online evaluation of exceptions, and the background processing of exceptions. The primary difference between the online exceptions and the background exception is that online exceptions highlight deviations within a query output, whether in the BEx Analyzer or in a Web application created with the BEx Web Applications Designer, but they do not allow for the automatic handling of notifications.

The background scheduling of exception reporting is done via the *reporting agent*. In the reporting agent the exception definitions that have been set in the BEx Query Designer are scheduled and follow-up actions are defined. Once the reporting agent performs the exception evaluation in the background, it updates a so-called exception log for the *Alert Monitor* to access. The Alert Monitor displays the exceptions collected in the log and provides a single place to evaluate all of the deviations that have been triggered by the background process run by the reporting agent. The Alert Monitor may be viewed in the BEx Analyzer, in Web applications, or in the Administrator's Workbench. The reporting agent allows settings to be set that indicate the follow-up action required to take corrective action. The follow-up actions may, for example, include sending an email to the responsible person. Background exception reporting—that is, the reporting agent settings—are organized by application component, InfoProvider, and query. We will discuss the administration options of the reporting agent in Chapter 9 and concentrate now on defining exceptions and online evaluations.

Exceptions are defined in the BEx Query Designer while defining the global query properties. The Alert Monitor icon allows for two menu selections: one for creating new exceptions and one for changing exceptions. Once the exception definition window is opened, the query designer must make three primary settings. The first setting is the key figure that the exception is to be applied to. The second setting is the threshold values and the alert level that should be used to highlight the exception should the threshold be reached. (As mentioned, nine unique alert levels may be set.) The third setting is the aggregation level and combination of characteristic values by which the threshold should be evaluated. Figure 7.6 depicts the characteristic combinations, or *cell restrictions*.

After the designer selects the appropriate characteristics, he or she must assign an operator for each characteristic. There are five options:

Totals Only. The exception is only evaluated when a characteristic is aggregated.

Everything. Exceptions are evaluated regardless of the level of aggregation for the characteristic.

Everything but Totals. Exceptions are evaluated for nonaggregated values for a characteristic.

Fixed Values. The exception is only evaluated for a preset characteristic value.

Hierarchy Level. Exceptions are only evaluated for specific levels of a hierarchy.

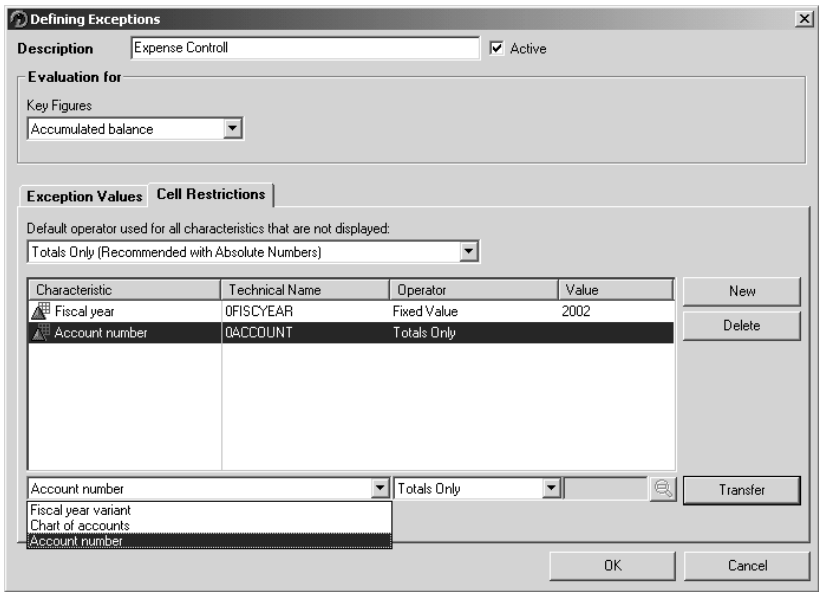


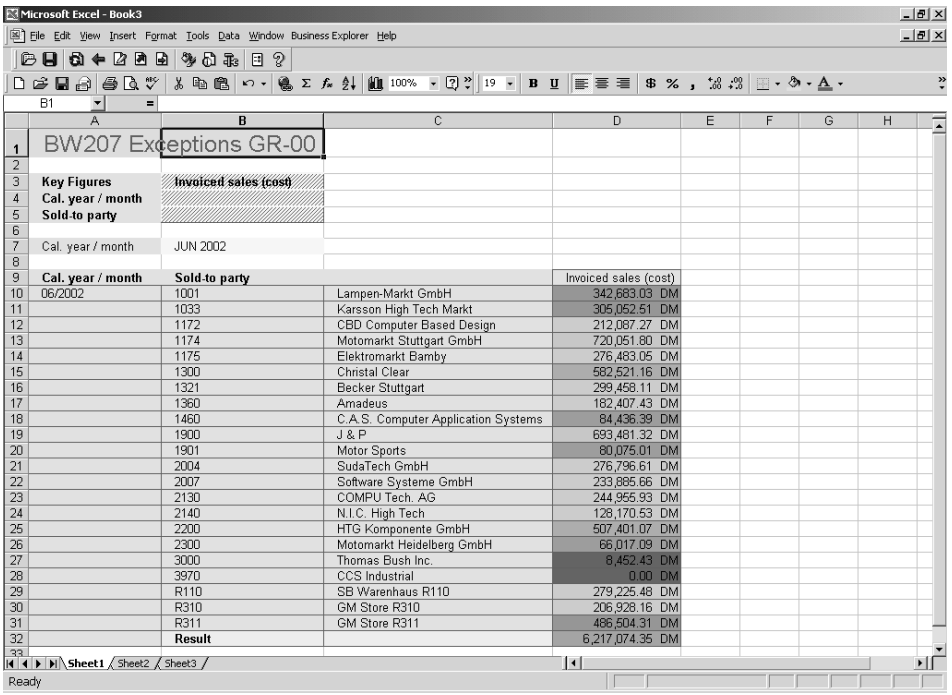
Figure 7.6 Exception definition.

A setting that query designers should be aware of is the default operator for all of the characteristics that do not have explicit cell restrictions defined. There are two options for characteristics that are not explicitly restricted: Totals Only or All. The *Totals Only* option is recommended for when the key figure for the exception is an absolute number. The *All* option is recommended when the key figure for the exception is a relative number. The *All* setting is typically used for key figures that are percentages or ratios. The default operator is applied regardless of the drill-down level of the characteristics that are not explicitly defined in the exception.

The ability to set the operators by characteristic and to combine multiple characteristics enables query designers to create a powerful set of business rules that may be evaluated and subsequently trigger alerts. Figure 7.7 illustrates online exception highlighting in the BEx Analyzer.

Restricting and Calculating Key Figures

Queries are designed for InfoProviders in the sense that only the characteristics and their attributes defined in the InfoProvider are available for analysis. Key figures are a bit different in that they may be used in one of three ways. In the first way, often referred to as *basic*, the key figure is dragged from the InfoProvider area of the BEx Designer to a column or row. The basic key figure is calculated by the OLAP engine based on the query's aggregation settings and the key figures attributes that were set when it was defined as an InfoObject in the Meta Data Repository. The other two options are restricted key figures and calculated key figures.


Microsoft Excel - Book3

File Edit View Insert Format Tools Data Window Business Explorer Help

B1 =

	A	B	C	D	E	F	G	H
1	BW207 Exceptions GR-00							
2								
3	**Key Figures**	**Invoiced sales (cost)**						
4	**Cal. year / month**							
5	**Sold-to party**							
6								
7	Cal. year / month	JUN 2002						
8								
9	**Cal. year / month**	**Sold-to party**		Invoiced sales (cost)				
10	06/2002	1001	Lampen-Markt GmbH	342,683.03 DM				
11		1033	Karsson High Tech Markt	305,052.51 DM				
12		1172	CBD Computer Based Design	212,087.27 DM				
13		1174	Motomarkt Stuttgart GmbH	720,051.80 DM				
14		1175	Elektromarkt Bamby	276,483.05 DM				
15		1300	Christal Clear	582,521.16 DM				
16		1321	Becker Stuttgart	299,458.11 DM				
17		1360	Amadeus	182,407.43 DM				
18		1460	C.A.S. Computer Application Systems	84,436.39 DM				
19		1900	J & P	693,481.32 DM				
20		1901	Motor Sports	80,075.01 DM				
21		2004	SudaTech GmbH	276,796.61 DM				
22		2007	Software Systeme GmbH	233,885.66 DM				
23		2130	COMPU Tech. AG	244,955.93 DM				
24		2140	N.I.C. High Tech	128,170.53 DM				
25		2200	HTG Komponente GmbH	507,401.07 DM				
26		2300	Motomarkt Heidelberg GmbH	66,017.09 DM				
27		3000	Thomas Bush Inc.	8,452.43 DM				
28		3970	CCS Industrial	0.00 DM				
29		R110	SB Warenhaus R110	279,225.48 DM				
30		R310	GM Store R310	206,928.16 DM				
31		R311	GM Store R311	486,504.31 DM				
32		**Result**		6,217,074.35 DM				
33								

Sheet1 Sheet2 Sheet3

Ready

Figure 7.7 Displaying exception in the BEx Analyzer.

Restricted key figures are the combination of a key figure with a filter value for a characteristic or set of characteristics. For example, in a report, a query designer wants to display in one column the net sales revenue key figure for the northeast region and in the next column net sales for all regions. The designer drags and drops the net sales key figure to the column, then selects the *Edit* context menu option. Once done, a dialog box appears, enabling the query designer to select the characteristic value or variable to filter the key figure selection with.

Calculated key figures allow arithmetic formulas to be defined using one or more basic key figures or formula variables or calculated key figures. For example, a query designer wants to display in the column of a report the net sales revenue key figure and in the next column the cost of sales key figure. The reporting requirement is that the gross margin be calculated. If the gross margin were available in the InfoProvider, the designer can select it and drag it into a third column. If not, the designer can calculate it by using the basic key figures for net sales and cost of sales. To do this, after dragging and dropping the net sales key figure to the column and the cost of sales key figure, the designer selects the *New Formula* context menu option. A dialog box appears, and the query designer can set the formula to calculate gross margin—in this case by subtracting cost of sales from net sales using the Formula Editor. The Formula Editor contains a mathematically complete set of functions as well as data functions and boolean operators.

BEx Analyzer

The traditional SAP BW tool for actually performing multidimensional reporting and analysis in SAP BW is the *Business Explorer, or BEx, Analyzer*. It is implemented as an add-on to Microsoft Excel combining the power of SAP BW OLAP analysis with all the features (e.g., charting) and the VBA (Visual Basic for Applications) development environment of Microsoft Excel. Storing query results in Microsoft Excel workbooks, for example, allows you to use information in offline mode or to add comments and send information to other users.

The BEx Analyzer is the tool for the power users and analysts. It provides the typical functions of selecting a query, saving changes, refreshing data, formatting the query layout, and of course, navigating through the results of the OLAP engine. You can launch the BEx Analyzer from the Administrator Workbench, Microsoft Excel, or the BEx Browser.

The most common method for starting the BEx Analyzer is to open Excel, then select the BEx Analyzer add-in. This causes the Business Explorer toolbar and menu to appear. Four files may be found in the BW subdirectory of the SAPGUI installation on a client workstation. A typical path, although this is customizable, is C:\Program Files\SAPpc\BW. When installing the SAPGUI, you must select the SAP BW options in order to install the BEx front-end tools. Following is a list of the add-in files for Microsoft Excel and their usage:

sapbex.xla. Effectively the BEx Analyzer. This file is used as a Microsoft Excel add-in and is launched when the *Start* | *Programs* | *Business Explorer* | *Analyzer* menu path is selected in Windows environments.

sapbex0.xla. This file is used by SAP BW when the BEx Analyzer is launched via the BEx Browser or the Administrator Workbench.

sapbexc.xla. This file provides a front-end installation check. It should be used after the SAPGUI is installed on a client workstation or for troubleshooting errors. The utility performs a check on the workstation to make certain the correct DLL, OCX, and EXE files are installed to ensure the Business Explorer toolset, as well as OLE DB for OLAP tools, will be able to execute queries. Pressing the start button in this spreadsheet initiates the check and suggestions for corrective action are provided.

sapbexs.xla. This file is a sample style sheet that provides documentation for the query elements.

Once you have launched Excel with the BEx Analyzer add-in, you will see a Business Explorer menu has been added to the standard Microsoft menu, as well as a BEx toolbar. The toolbar consists of icons to open, save, refresh, format, change, and navigate queries. As we described earlier in this chapter, queries are defined in the BEx Query Designer; however, the toolbar has an option for launching the BEx Designer. Once a query is defined, it may be analyzed in Excel. You will need to select the folder icon in order to launch a query, a workbook, a query view, or to set exceptions conditions.

A *workbook* is a standard Microsoft Excel workbook with embedded references to query views and optional application elements built using Microsoft Excel functionality. For example, you can add charts, push buttons, and list boxes to a query workbook, alongside the result area and filter cell area of a query. You can also set the workbook template as a corporate standard so that all queries that are inserted into workbooks have the same look and feel. Remember that a query is independent of the workbook it is attached to and that a workbook may have several queries attached to it.

Workbooks allow query results and any applications built within Microsoft Excel to be saved and viewed offline and online. This is ideal for distributing the query workbooks to a larger audience, say, via email, while still requiring proper authorization to review the workbook. One of the nice features in the BEx Analyzer is the ability to protect the workbook from changes in addition to standard Excel sheet protection. Excel sheet protection limits the majority of the functions available to the end user. The SAP sheet protection password protects not only the query areas from changes but the entire active worksheet, yet it still allows for continued ad hoc navigation through the query cube.

The query designer determines the initial view of a query when the query is first defined. The properties of a query are considered global. That is, the designer of the query predetermines the rows, columns, filter values, variables, hierarchies, and free characteristics for all users who execute the query. This does not prohibit a specific user from customizing the query. The changes that are made to a query by an end user may be done in the local view or in the global definition, depending on authorization. Changes in the local view only impact the current user's session while analyzing a given query. A common example of a local change is to swap the rows or a query with the columns. The local view is also ideal for performing multiple navigation steps at one time. Hierarchies may also be added to a query in the local view, assuming the hierarchy definition exists for a characteristic in the query. Changes made to the global

definition will impact all users of the query. The BEx Analyzer launches the BEx Query Designer for global definition changes.

Once a query is executed via the BEx Analyzer, the end user will see the initial view of the query. There are four parts to every query: title, filter, results, and text. The initial view of a newly created query will by default have the query name (title) and free characteristics (also known as dynamic filters) at the top left portion of the Excel spreadsheet. This is referred to as the *filter cell area*. Below the filter cell area is the query grid or results area. The query grid will be filled according to the selection criteria set forth by the query designer in the BEx Query Designer. The results area in the spreadsheet is under SAP BW's control. While an end user may type a value into a cell in the results area, a refresh or navigation step will cause the values to be overwritten by the newly created query view that is placed into the results area. The title, filter, results, and text may each be moved any place on a worksheet independent of each other.

The query is executed and the results area is filled. The BEx Analyzer toolbar, Business Explorer menu bar, or the context-sensitive right mouse button may be used to navigate through the query cube. The end user may easily filter, drill down and across, translate currencies, apply condition and exceptions, expand and collapse a hierarchy while navigating through the data. This navigation is often referred to as an *analysis path*.

The BEx Analyzer can display a variety of technical meta data as well as business meta data for a query. You display the technical meta data by selecting *Business Explorer | Settings | Display BW Server Information* from within Excel or by selecting the *Settings* icon in the BEx toolbar. In doing so, you will be presented with technical information pertaining to the system ID, host, database, hardware, and IP address. In addition, by switching the Trace option on, you can create a more powerful set of technical meta data. The trace option records all of the steps in an analysis path from connection to close. This utility is very helpful should you wish to extend workbooks by adding Visual Basic code to create custom pull-down lists, buttons, and the like. You can activate the trace log by selecting *Business Explorer | Settings | Trace*. To then view the trace log, you select *Business Explorer | Settings | Display Trace*. The SAP BW functions that are called from the Excel add-in are logged in the trace file. The log file helps programmers debug extensions to the front end that utilize the BEx Analyzer user exits.

NOTE Customer exits allow installation specific code to be added to an SAP software component without upgrade repercussions.

Visualizing Queries Results

There are two powerful visualization features in the BEx Analyzer. The features are relatively simple to use and apply to query results sets. Once a query is executed, you select the *Layout* button in the BEx toolbar. There you will find the *Attach Chart* and *Attach Map* options.

When the *Attach Chart* option is selected, a chart will appear in the current Excel worksheet. The chart may be moved to any place on the existing worksheet or to any worksheet in the current workbook. The chart is automatically attached to the query results area and updated as the query results area is updated. For example, if you

attach a chart to a query that has a drill-down displaying countries and the revenue sold in those geographies and swap the country characteristic with the sold-to party, the chart would automatically adjust to the new characteristic. You may select any of the chart types available in standard Microsoft Excel to visualize the query results area. You simply select the chart, press the right mouse button, and select the chart type to display the charting options. A key point to remember here is that the navigation is conducted on the query results and then chart display is altered—not the other way around.

When the *Attach Map* option is selected, a map will appear in a new Excel worksheet named BEx Map. Unlike the chart, the map may not be moved any place on the existing worksheet or to another worksheet in the current workbook. The BEx Map is attached to the query results area in a similar manner as the chart is. However, the BEx map and its settings take precedence and control over the results area. The prerequisite is that the query consists of at least one geo-relevant characteristic. In the InfoObject definition of a characteristic, the geography-specific meta data is configured. There are two primary types of geographical settings for characteristics: static geo-types and dynamic geo-types. *Static geo-types* are used for characteristics that represent objects that do not change—for example, country or state borders. *Dynamic geo-types* are used to identify objects that may not be in the same locations—for example, vendors or store locations. Both the static and the dynamic types may derive their points from values stored as attributes for the characteristic.

The BEx Map may have up to three layers of data displayed visually for any query. Each layer may contain separate images or color shading. For example, for revenue sold to all customers in Europe, each country may be shaded according to the total sales value, with the darkest states representing the highest sales value. Then a second layer may be defined to show a pie chart with a breakdown of profit by product type, where the size of the pie chart indicates the amount of profit relative to profit in the other countries. See Figure 7.8.

NOTE The BEx Analyzer does not use the Internet Graphics Server (IGS) to render its images, while the BEx Web applications use the IGS.

BEx Formatted Reporting

Another component in the Business Explorer suite is *BEx Formatted Reporting*. SAP has partnered with Crystal Decisions to deliver a pixel based reporting tool to extend the Business Explorer and meet the formatted output requirements that many organizations have. Figure 7.9 is an example of the type of report that may be created with the BEx Formatted Reporting option. At this writing, SAP has entered into an original equipment manufacturing (OEM) and reseller agreement with Crystal Decisions, which may cause some confusion, since some of the options in BEx Formatted Reporting are included with the purchaser's SAP BW license and others may need to be purchased as an add-on to the SAP BW license. We will explain the functionality and options available to you, but advise that you contact SAP to determine the appropriate software licensing.

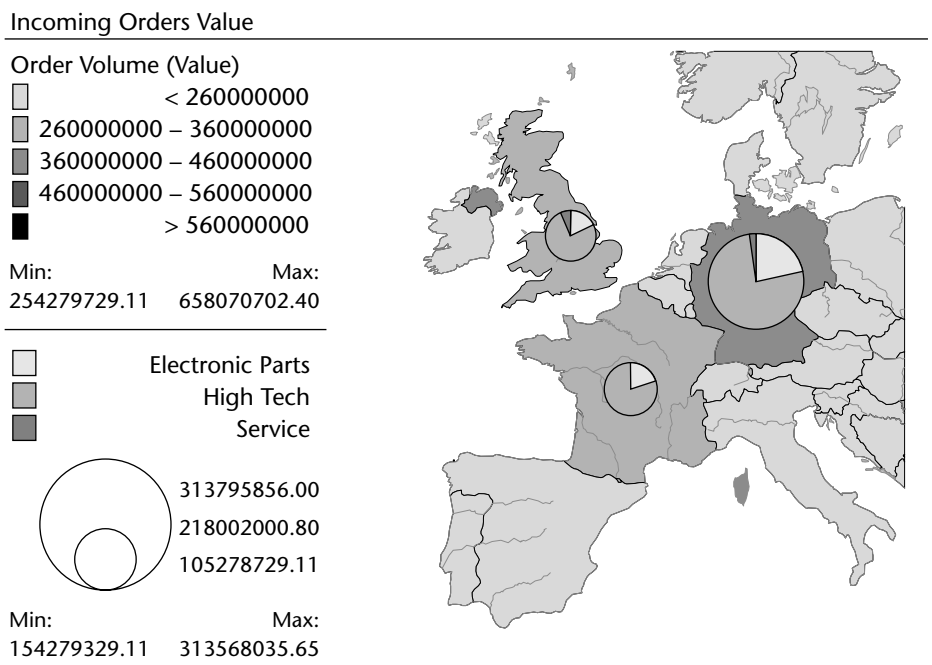


Figure 7.8 BEx Map.

There are many reports for the public sector, in particular, that must be into a specific format specified by different acts or amendments in government. These so-called legal reports need to conform to specific formatting requirements that are entered into law. Figure 7.9 shows an example from the Brazilian government called a *Mapa de Reintigrações*, or "Map of Reintegration." This report describes the reintegration of funds into a program. The data in the third column of this report is sourced from a BEx query that was created for an InfoProvider that is abstracting an ODS object. The data in the other columns is being sourced from another ODS object.

Of course, custom development using SAP BW interfaces, described later in this chapter, will accomplish the same results; however, the maintenance involved in supporting the solution would be too great. With the BEx Formatted Reporting option, the report designer does not require programming skills. The process of utilizing the Formatted Reporting option is reasonably straightforward once the additional software components are installed. There are five steps to creating, publishing and viewing a formatted report:

1. Create a tabular/single structure query in the BEx Query Designer.
2. Log in to the Crystal Reports Designer client software and select the tabular/single structure query from SAP BW based on your SAP BW authorizations.

Firma _____

MAPA DE REINTEGRAÇÕES

METODO DAS QUOTAS DEGRESSIVAS

(a) _____

Activade principal _____

(1)	(2)	(3)	(4)	(5)	(6)	(7)	(8)	(9)	(10)	(11)	(12)	(13)	(14)	(15)	(16)
01	30000000000	KRL3	001	2000	100.00	90.25	0.01		0.75	9.00	91.00				0.00
01	30000000000	KRL3	002	2000	100.00	89.50	0.01		0.75	9.75	90.25				0.00
01	30000000000	KRL3	003	2000	100.00	88.75	0.01		0.75	10.50	89.50				0.00
01	30000000000	KRL3	004	2000	100.00	88.00	0.01		0.75	11.25	88.75				0.00
01	30000000000	KRL3	005	2000	100.00	87.25	0.01		0.75	12.00	88.00				0.00
01	30000000000	KRL3	006	2000	100.00	86.50	0.01		0.75	12.75	87.25				0.00
01	30000000000	KRL3	007	2000	100.00	85.75	0.01		0.75	13.50	86.50				0.00
01	30000000000	KRL3	008	2000	100.00	85.00	0.01		0.75	14.25	85.75				0.00
01	30000000000	KRL3	009	2000	100.00	84.25	0.01		0.75	15.00	85.00				0.00
01	30000000000	KRL3	010	2000	100.00	83.50	0.01		0.75	15.75	84.25				0.00
01	30000000000	KRL3	011	2000	100.00	82.75	0.01		0.75	16.50	83.50				0.00
01	30000000000	KRL3	012	2000	100.00	82.00	0.01		0.75	17.25	82.75				0.00
02	30000000000	KRL5	001	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
02	30000000000	KRL5	002	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
02	30000000000	KRL5	003	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
02	30000000000	KRL5	004	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
02	30000000000	KRL5	005	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
02	30000000000	KRL5	006	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
02	30000000000	KRL5	007	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
02	30000000000	KRL5	008	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
02	30000000000	KRL5	009	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
02	30000000000	KRL5	010	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
02	30000000000	KRL5	011	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
02	30000000000	KRL5	012	2000	100.00	74.11	0.01		0.00	25.89	74.11				0.00
03	30000000000	KRL4	001	2000	100.00	91.00	0.01		0.00	9.00	91.00				0.00
03	30000000000	KRL4	002	2000	100.00	91.00	0.01		0.00	9.00	91.00				0.00
03	30000000000	KRL4	003	2000	100.00	91.00	0.01		0.00	9.00	91.00				0.00
					2,700.00	2,286.82			9.00	504.18	2,295.82				0.00

Figure 7.9 BEx Formatted Reporting.

3. Design the report in Crystal Reports Designer and save the formatted report definition back to SAP BW.
4. Publish the report from SAP BW to the Crystal Enterprise Server.
5. View the report via an Internet browser.

First, you develop your reports by choosing your query, designing your report, and storing it within the SAP BW Meta Data Repository. You then publish this report to the Crystal Enterprise Server. The integration between SAP BW and the Crystal Reports Designer includes a so-called connector. This connector makes it possible to launch the BEx Query Designer from within the Crystal Reports Designer, assuming both software components are installed on the same client machine. Information is passed from SAP BW to Crystal Enterprise after SAP BW receives a multidimensional expression from the Crystal Enterprise Server.

One of the current limitations with the integration, although we anticipate it will be resolved in a future release, is the restriction to one structure in the SAP BW query. This reduces the opportunity to use complex structures and single cell calculations in the published formatted report. From a query and report design perspective, this may be its biggest limitation and at that it is not very big. Query designers should look to see if creating more than one query and having them populate the formatted report will work around the limitation.

From a publishing and viewing perspective, the integration of the components still leaves a bit to be desired as illustrated in Figure 7.10. A potentially large drawback is that the Crystal Enterprise Server is serving the formatted reports to a Web server that is not integrated with the SAP Web Application Server. This means organizations will need to administer multiple Web servers and Web security profiles in order to view formatted reports from SAP BW. While there is some security synchronization between SAP BW and the Enterprise server, this is a manual process that must be administered to keep in sync. Crystal Enterprise Server has its own security scheme that is quite different from the security scheme found in SAP BW.

Business Explorer Web Application Designer

In this section we will describe the components of a BEx Web application and how it is accessed, as well as integrated, in the SAP Enterprise Portal. BEx Web applications may be built in the *Business Explorer Web Application Designer* or in HTML editors like Macromedia's Dreamweaver or Microsoft's FrontPage. The BEx Web Application Designer is a visual development environment that allows Web designers to create tailored Web templates and cockpits that include traditional query elements like structures, calculated key figures, variables, filters, and free characteristics, as well as documents and Web content sources. Business charts, maps, and other graphical items may also be integrated into the Web applications. These graphical items and the traditional query elements are placed on a Web template that is the basis for BEx Web applications.

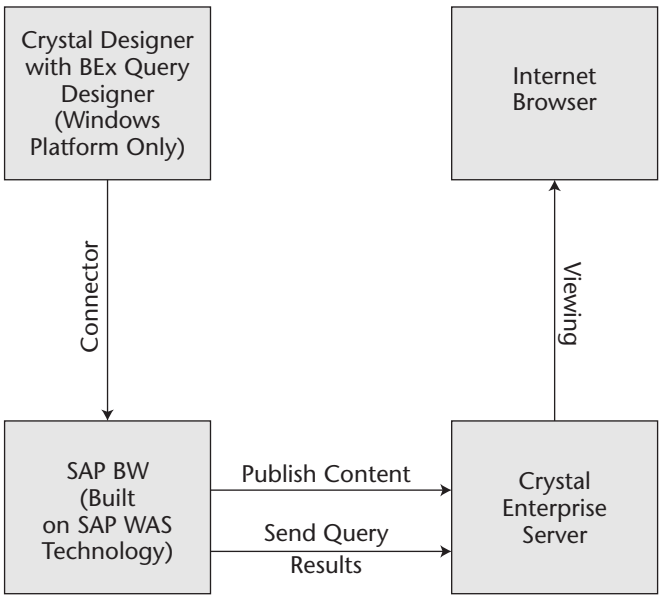


Figure 7.10 BEx Formatted Reporting components.

Figure 7.11 illustrates the Web Services object model supported by SAP BW. A Web template is composed of *data providers* and *Web items*. Data providers are objects, either BEx Queries, BEx Query view, or the entries in the Alert Monitor that Web items use to

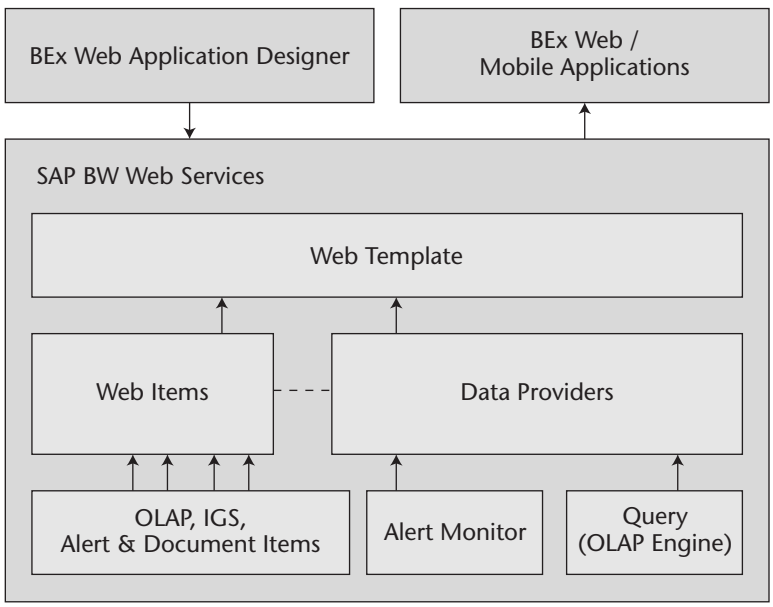


Figure 7.11 Web Services object model.

retrieve information. Web items are objects that make data available in HTML. It is the Web template that is invoked by a URL. When the template is filled with, or bound to, data, a Web application is instantiated. Prior to the binding it is simply a Web page with SAP BW-specific object tags—in other words, a Web template. The template's Web items are instantiated and are bound to a specific data provider.

Note that the query-based data providers assigned to Web items and the data providers will eventually call an InfoProvider, since SAP BW Queries are defined for a given InfoProvider. The use of the terms *data* and *information* may cause confusion for some readers, since information is generally considered data within a useful or relevant context and data usually is thought of as facts that still need to be processed to be useful. To this we say, what is information to one service may be data to another.

Figure 7.12 illustrates the three main components of the BEx Web Application Designer. As shown from left to right, they are the Web items, template layout with overview and HTML tabs options, and the Properties windows. On the left side of the figure is a list of all of the SAP delivered Web items. Each of the standard Web items has a list of general and specific properties that control the behavior and appearance of the item. When a Web application designer drags the Web item to the template layout window, an icon representing a table appears in the layout window. At the same time the general Properties for the Table are displayed. The designer is asked to name the data provider and to select a Query or Query View for the newly created data provider. Each Web item has a list of general and specific properties, as shown for the Table Web item on the right side of Figure 7.12.

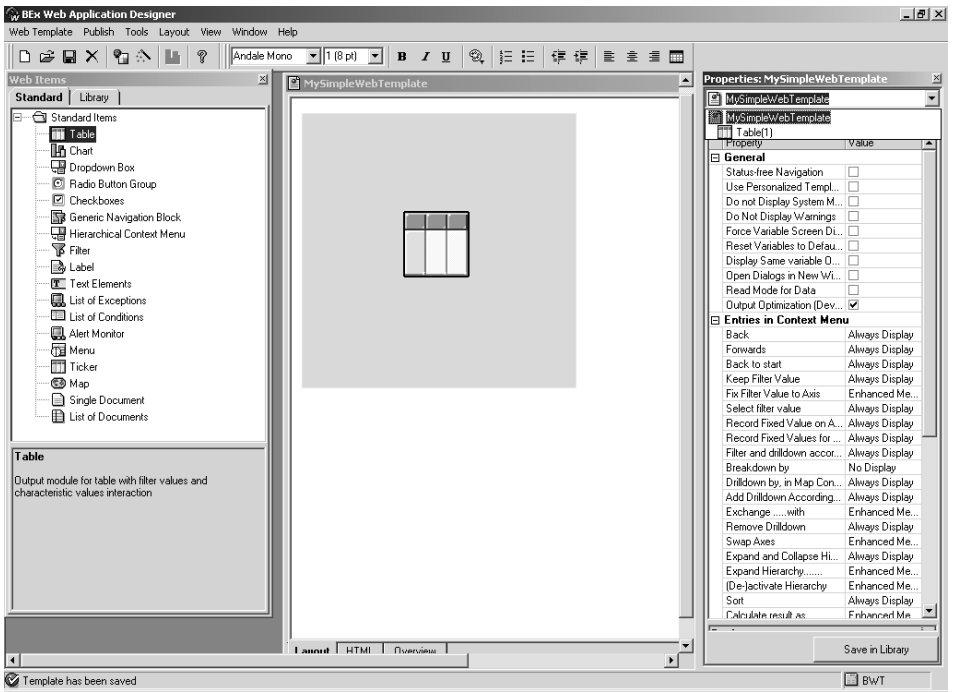


Figure 7.12 BEx Web Application Designer.

The general settings of a Web item control such things as whether or not the title is to be displayed for a Web item, whether or not a border should be displayed around the Web item, if the item should have links, should the Web item's initial view be closed or open, and what should the initial size of the Web item be set to. The general properties are the same for all Web items, with the exception of an additional sizing parameter for the Web items that represent graphical items.

There are four categories of Web items: OLAP, document, alert, and graphic. Every Web item is assigned to a data source, or, as is the case of the Alert Monitor item, the data source is implied. In Chapter 9 we will discuss the Alert Monitor in greater detail. Each of these Web items also has a set of specific properties. Again, there are exceptions, as the so-called list of exceptions and list of conditions Web items do not have specific object properties. The specific properties for all other Web items allow the Web designer to control how each item appears and the options each item represents to the end user. These properties may be set or replaced by the Web designer or at run time by the Web application user by setting or replacing the parameter in the URL or object tag.

The process of defining a new Web template starts when a Web application designer logs in to an SAP BW system from the BEx Web Application Designer. A Web item dragged to the template layout window in the BEx Web Application Designer HTML is created.

Before we dive into the Web Services object model, let's first make certain we are clear on what SAP means when it uses certain terms. BEx Web application is reasonably simple. A BEx Web application is an HTML page that contains one or more Web items and data providers that have rendered business information from SAP BW. The key words here are *have rendered*. It is with the BEx Web applications that the consumer of information interacts. SAP BW Web applications are based on Web templates. The Web template is an HTML page that has at least one SAP BW Web item and data provider defined as objects of the HTML page. SAP uses the term *Web applications* generically to represent everything from a static HTML page that renders precalculated information from SAP BW to complex interactive Web-based cockpits that retrieve information from various SAP components.

The Web template is the top level of the Web Services object model. The Web template controls the structure, data sources, and properties of the Web application. Following is sample HTML of a Web template:

```
<HTML>

!-- BW web template object tags -->
<object>
         <param name="OWNER" value="SAP_BW">
         <param name="CMD" value="SET_PROPERTIES">
         <param name="TEMPLATE_ID" value="MYSIMPLEWEBTEMPLATE">
         <param name="SUPPRESS_SYSTEM_MESSAGES" value="X">
         <param name="MENU_DISPLAY_DOCUMENTS" value="">

         TEMPLATE PROPERTIES
</object>
!-- BW data provider object tags -->
<object>
```

```
          <param name="OWNER" value="SAP_BW">
          <param name="CMD" value="SET_DATA_PROVIDER">
          <param name="NAME" value="DataProvider(1)">
          <param name="QUERY" value="0D_SD_C03_Q004">
          <param name="INFOCUBE" value="0D_SD_C03">
          DATA_PROVIDER:             DataProvider(1)
</object>
<HEAD>
<META NAME="GENERATOR" Content="Microsoft DHTML Editing Control">
<TITLE>MySimpleWebTemplate</TITLE>
      <link href= "MIME/BEx/StyleSheets/BWReports.css" type="text/css"
rel="stylesheet">
</HEAD>
<BODY>
<P>
!-- BW table web item object tags -->
<object>
          <param name="OWNER" value="SAP_BW">
          <param name="CMD" value="GET_ITEM">
          <param name="NAME" value="Table(1)">
          <param name="ITEM_CLASS" value="CL_RSR_WWW_ITEM_GRID">
          <param name="DATA_PROVIDER" value="DataProvider(1)">
          ITEM:            Table(1)
</object>
</P>
</BODY>
</HTML>
```

SAP BW Web Services

SAP BW version 3.0 has made significant advancements in its support of Internet protocols. SAP BW Web Services are the greatest example of this. They take care of requests from Web browsers or mobile devices that send requests via the Hypertext Transfer Protocol (HTTP). The services route the requests to the OLAP engine, along with the Internet Graphics Server (IGS) if necessary, and retrieves and assembles the requested Web components for delivery back to the requesting device (browsers, mobile phones, personal digital assistants, and so on).

SAP BW Web Services work with a request handler that receives requests from devices, validates those requests, and routes them to the SAP BW Web Services component. Once the SAP BW Web Services component receives a request, it disassembles it into its component parts, thereby separating content from format. The data requests are routed to the OLAP engine for processing. All requests for information from SAP BW are routed to the OLAP engine or the Alert Monitor. When the OLAP engine returns the result set to SAP BW Web Services, formatting and construction of the Web components are assembled. For example, say the OLAP engine retrieves a list of the top 10 customers for a given period. The requesting Web-based application asks for the results to be displayed as a bar chart. SAP BW Web Services takes care of rendering the results in this format.

Since SAP BW is built on top of the SAP Web Application Server (SAP WAS), SAP BW can take advantage of technology like the HTTP server, the Multipurpose Internet Mail Extensions (MIME) repositories, and version control. When a Web template is requested by a browser, the HTTP handler will identify the object tags within the template that is requested to find the objects in the template that are owned by SAP. The handler looks to the `OWNER` parameter, and if the value is set to `SAP_BW`, the object is routed to SAP BW Web Service. The other standard HTML tags are handled by the SAP WAS in the same manner a traditional Web server would.

Let's walk through a simple example of how a SAP BW Web application would access information from SAP BW, assuming the following URL was sent from a Web browser: http://yourcompany.com:1080/SAP/BW/BEx?CMD=LDOC&TEMPLATE_ID=MYSIMPLEWEBTEMPLATE. The URL starts off with the HTTP protocol and the domain and port number assigned to the SAP BW Web Application Server. /SAP/BW/BEx? represents the path and program for SAP BW Web Services. Next, you will find the command that is being sent to the Web service, in this case CMD=LDOC, which is requesting that a Web template be loaded. The & symbol marks the beginning of a parameter. The value assigned to the parameter follows the equal sign. In the example URL only one parameter, `TEMPLATE_ID`, is passed to SAP BW. The `TEMPLATE_ID` is set equal to `MYSIMPLEWEBTEMPLATE`.

Once the SAP WAS hands the request to the SAP BW Web Service, the template `MYSIMPLEWEBTEMPLATE` is retrieved and interpreted. The Web service looks for object tags that are owned by `SAP_BW` and determines if the objects are data provider objects or Web item objects. The data provider object in this case has set the parameter `CMD` with the value `SET_DATA_PROVIDER` and named the data provider, appropriately enough, `DataProvider(1)`. The `CMD` command must be sent to SAP BW in order to instantiate a query. The other parameters of the object prove more interesting, because they set the value for the InfoCube and query. In the code sample you will see that the query and InfoCube parameter values are set using the technical names for the desired query and InfoCube. In this example standard Business Content for a Sales and Distribution InfoCube and query have been assigned to the data provider object.

There are two types of data provider objects. The first type selects data from an SAP BW query, as defined in the preceding code. The second selects information for the Alert Monitor. You explicitly assign the data provider object for OLAP items in the BEx Web Application Designer by selecting a predefined query or query view. The behavior of a data provider may be changed by sending a URL to SAP BW containing the `CMD` parameter. For example, a Web template that has been designed and assigned to retrieve information about open sales orders from a query based on a sales and distribution query may reused to select data about sales deliveries from a different query and InfoCube. This is accomplished by calling the command `CMD=RESET_DATA_PROVIDER` and assigning the parameters `INFOCUBE` and `QUERY` with the values of a deliveries InfoCube and query.

The data provider is a parameter of the Web items and is assigned in our example to the Table Web item. This essentially binds the table presentation item to a query that is assigned to the Data Provider object. The Data Provider object will route its request for information to the OLAP request handler and subsequently the OLAP engine. The Table Web item object is instantiated and takes care of the presentation of the results returned from the OLAP engine and displays the results in a Table Web item.

Data providers may be manipulated by sending URLs that include commands and parameters. Web application designers may control such behavior as the filtering values, sorting, drill-down level, switching of characteristics, currency conversion, and exceptions. The lines between information access, analysis, and presentation are a bit challenging to separate. We will explain the topic of manipulating the behavior of data providers as we describe the presentation services later in the next section.

The second type of data provider selects data from the Alert Monitor. The data provider is automatically assigned when an Alert Monitor Web item is dragged onto the Web template layout. Like the Table object, the Alert Monitor is instantiated by setting the command `CMD=GET_ITEM`, assigning a name to the monitor and setting the `ITEM_CLASS` parameter as shown in the object code sample that follows. The difference between the Alert Monitor Web item and the Table Web item is that there is no explicit assignment of a data provider, since the data provider is the Alert Monitor and is controlled in the Administration Workbench with the reporting agent technology.

```
<object>
         <param name="OWNER" value="SAP_BW">
         <param name="CMD" value="GET_ITEM">
         <param name="NAME" value="Alert Monitor(1)">
         <param name="ITEM_CLASS" value="CL_RSR_WWW_ITEM_ALERT_MONITOR">
         ITEM:            Alert Monitor(1)
</object>
```

Web application designers may assign one or more data providers as well as multiple Web items to the same data provider. This enables the designer to create complex Web-based cockpits that source information from various SAP BW-based queries.

The data provider and associated Web items may be set in a Web template or they may be set and manipulated via an SAP BW URL. Together these objects create the so-called SAP BW Web API. The Web API is published and documented by SAP. The most comprehensive documentation is the Web API Reference (SAP AG SAP BW 3.0a Documentation 2001) that can be found on the SAP Web site

Enterprise Portal Integration

The BEx Web applications may be integrated into an Enterprise Portal environment, as the Web applications are accessible through a URL. The BEx Web Application Designer provides two options for easily integrating the BEx Web applications with the SAP Enterprise Portal: *Publish as iView* and *Save as iView File*. Both options accomplish the same result, which is to effectively place an iView wrapper around a BEx Web application so the iView Studio may manage the placement of and access to the BEx Web application as it would any information or application within its framework.

The Save as iView File option will create an IVU file. This may then be imported into the portal, which then creates an iView. The following code is an example of an IVU file:

```
<iView name=<![CDATA["BEx Web Template Name"]]> type="Java">
        <channel>BI Web applications</channel>
        <title><![CDATA[BEx Web Template Name]]></title>
        <params>
```

```
                <PhysicalService>
                <System  type="SimplePropertyType">
                        <Value><![CDATA[BWTCLNT800]]></Value>
                        ....
                </System>
                <SAP_BWReport  type="SimplePropertyType">

<Value><![CDATA[cmd=ldoc&TEMPLATE_ID=BWTIVUEXAMPLE]]></Value>
                        <PropertyAttribute  name="DispPropTitle"
final="false">
                        <Value>SAP_BWReport</Value>
                        </PropertyAttribute>
                        ....
                </SAP_BWReport>
                        <Description  type="SimplePropertyType">
                        <Value><![CDATA[BEx Web Template Name]]></Value>
                        ....
                        </Description>
                <MasterLink  type="SimplePropertyType">
                        .....
                </MasterLink>
                </PhysicalService>
        </params>
</iView>
```

For this to work, the portal must be set to recognize the SAP BW system. This is defined in the *Systems.xml* file within the portal. The XML in this document contains the information about the component systems that are integrated in the portal. It is located under the *System Landscape* tab inside the portal, which is assigned to all portal administrators. It may also be accessed directly in the filesystem. The administrator must enter the server name, client number, and system name into this document for each SAP BW system that the portal is to recognize. A unique name will need to be assigned for each SAP BW system. In the seventh line of code in our example, `BWTCLNT800` is the unique system name.

The iViews inside the portal are grouped into channels. Channels primarily make the job of the portal administrator easier. This does not have anything to do with the role concept, but rather it is a way to organize the copious amounts of iViews that will invariably be created. In our example, the iView is being assigned to the `BI Web applications` channel. If this channel does not exist in the portal, it will be created upon importation. If it does exist, the iView will be added to the existing channel. The iView will be given the name "BEx Web Template Name" as a reference in the portal and, in this case, will have the same as its title when rendered for the end user.

From within the portal framework an administrator may import the iView with a few clicks. First, the iView type must be set. In our example this is Java; however, this may be of .NET type. Next, the administrator navigates to the IVU file in the local filesystem and uploads it. Once the iView is inside the portal, it will appear under the appropriate channel. An iView may be created from within the portal with similar ease. When you are creating an iView for an SAP BW Web applications the two key

elements are the component system and the query. The results are the same. The iView appears under the specified channel, regardless of the creation method. There is a preview button that may be used for testing the iView, but bear in mind that a valid SAP BW user-name will be needed unless the Single Sign-On or User Mapping has been configured.

For a user to access the iView within the portal, the iView should be assigned to an appropriate page in the portal. This page is assigned to a workset, which is in turn assigned to a role. The roles are then assigned to the users. This does not necessarily enable the user to enter the SAP BW system without a SAP BW username and password. If the user has a SAP BW username, the administrator can set up a mapping between the portal user and the corresponding BW user. This is quite easy to do, but it can be cumbersome if there are a lot of users. The portal addresses this issue by allowing users to set up their own mapping for the systems that appear in the Systems.xml. For more information on the Enterprise Portal, refer to SAP's Web site.

BEx Mobile Intelligence

One of the newest additions to the Business Explorer toolset is BEx Mobile Intelligence. The BEx Mobile Intelligence is less of a product than it is support for mobile data access protocols. There are no specific mobile devices that need to be purchased from SAP to use the mobile features. Any mobile phone or wireless PDA supporting the Wireless Access Protocol (WAP) or HTTP should be sufficient to access the Web applications from SAP BW. There is also support for Short Message Service (SMS) from the reporting agent.

There are two primary approaches to mobile access: online and offline. The offline approach involves prestaging a BEx Web application by scheduling the Web application page to precalculate the results of its Web items by selecting from the data providers in a background job. We will discuss the options for this in Chapter 9, as the settings in the reporting agent are abstracted from the presentation services. That is, the reporting agent doesn't care if the Web page is to be sent to a Web browser or a mobile device. The mobile device will then need to download the Web pages using Web Folders, WEBDAV, or other synchronization utilities.

WAP Device Support

Mobile phones that support the WAP send requests for information to a WAP gateway that is typically hosted by the mobile service provider. The request is translated by the gateway into an HTTP request and routed to the SAP BW server. The request from the gateway is little different than a request from a traditional Internet browser. The main difference is that the request is marked coming in from a specific WAP device type. The request uses the same SAP BW URL mechanism that the Internet browser would, and the data selection process in the SAP BW Web Application Server is the same regardless of the requesting device. SAP BW uses its so-called request handler to determine the appropriate output format for the Web application output. For a WAP device this would be the Wireless Markup Language (WML) instead of HTML for the Internet browser. The WML would be routed back to the gateway, where it would be translated into device-specific byte code and sent to the device for final rendering.

PDA Support

PDAs that support the HTTP send requests for information to an Internet service provider (ISP). Unlike the WAP request, there is no need to translate the request prior to routing to the SAP BW Web Application Server. The request uses the SAP BW URL mechanism the same as any Internet browser would. The data selection process in the SAP BW server is the same regardless of the requesting device. This time SAP BW uses its so-called request handler to determine the appropriate output format for the Web application and generates the appropriate HTML.

Design Restrictions

There are a few restrictions of which designers of mobile Web applications should be cognizant. They include but are not limited to the lack of support for HTML style sheets; limitations on the usage of Java Script; and the lack of support for the ticker, map, and hierarchy Web items. Otherwise, the Web application design process is the same regardless of the output device. There are also device display and color restrictions that should be considered prior to designing mobile Web applications.

Information Analysis and Access Services

In Chapter 3 we introduced the information analysis and access services aspect of the SAP BW architecture. In this section we will further explore these services. The information access services layer—as the name already says—provides access to structured and unstructured information stored in the SAP Business Information Warehouse. Structured information is retrieved through SAP BW InfoProviders, unstructured information is retrieved from a content management service. The integration of unstructured documents and structured documents has become an essential feature for business intelligence tools. Figure 7.13 highlights the components of the information access services layer.

The main services in this layer are the InfoProvider interface, the OLAP engine, the data mining engine, and the InfoSpoke Manager. We categorize these layers into request-handling services, processing services, and retrieval services. We will discuss each of these in detail with focused attention on the request-handling interfaces, since they are a primary integration point for third-party applications, specifically the interfaces that expose the service of the OLAP engine.

Information Provider Interface

SAP has isolated and abstracted functionality and complexity throughout the SAP BW product. Nowhere is this more evident than in the retrieval services and specifically the InfoProvider interface. The *Information Provider interface* has been introduced to standardize access to the structured data available to SAP BW. The Information Provider interface allows access to information that is stored in SAP BW, as well as data that is not stored in SAP BW. The interface enables the OLAP engine to treat all requests for information in the same manner regardless of where and how the data is physically stored. There are two generic types of InfoProviders: physical and virtual.

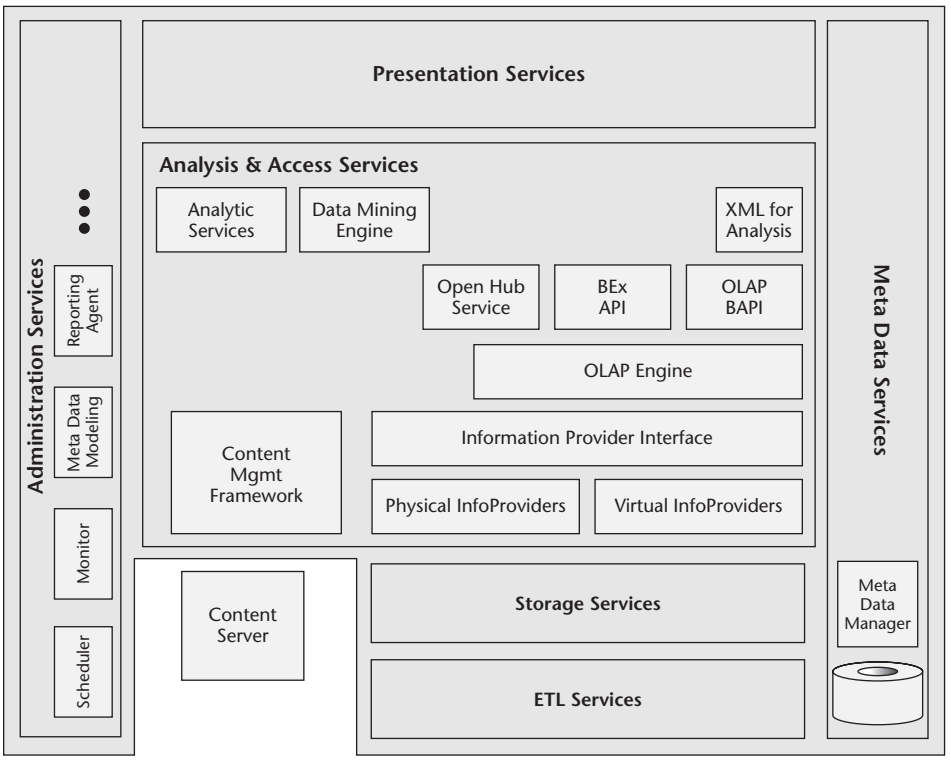


Figure 7.13 Information access services architecture.

Table 7.2 provides a list of the physical and virtual InfoProviders available in SAP BW. Both physical and virtual InfoProviders are accessed by the same set of OLAP services. The InfoProvider may be thought of as an abstraction layer for all data flowing through SAP BW either to end users, third-party business intelligence tools, other SAP components, or custom-built analytic applications. The ultimate consumer of the information requested from SAP BW need not be concerned about how or where the data is physically stored. While the OLAP engine accesses all different types of InfoProviders in the same way the InfoProvider in turn accesses the data storage services in very specific ways.

Table 7.2 The Foundation for InfoMarts: Types of InfoProviders

PHYSICAL INFOPROVIDERS	VIRTUAL INFOPROVIDERS
BasicCube (standard)	SAP remote cubes
BasicCube (transactional)	Generic remote Cubes
ODS objects	Virtual cube with services
Master data	MultiProviders

Physical InfoProviders

All queries that are passed to the OLAP engine send either a multidimensional expression or, in the case of the BEx Analyzer, proprietary command codes. As we will discuss later in this chapter, there are two primary OLAP interfaces, so the language and syntax of the expression may differ, but the essence of the requests is the same. The OLAP engine handles the request the same way regardless of the interface the request was received from and passes it to the InfoProvider interface. The InfoProvider interface, in taking care to abstract the data source, handles all requests from the OLAP engine in the same manner and determines the InfoProvider type for the specific request.

Once the InfoProvider type is identified, the query requests are handled in a specific manner depending on how and where the data is physically stored. For example, if the InfoProvider is abstracting an ODS object, the InfoProvider will flatten the multidimensional request for the data. Remember from Chapter 4, the ODS object is nothing more than a relational table. The flattened request will be passed to the InfoSet interface, which will, in turn, send a request to execute an actual InfoSet. The InfoSet will then hand the request to a storage service, which will, in turn, determine the optimal access plan. The same holds true for a master data InfoProvider.

The result set of the InfoSet will be retrieved and handed back to the OLAP engine, where (pardon the redundancy) analytical processing services are applied prior to handing the specified query view back to the original requester. The OLAP engine's ability to generate both multidimensional query cubes and tabular result sets has by and large enabled the InfoProvider interface to abstract both multidimensional and flat data sources. Note that both transactional ODS objects and standard ODS objects are available as InfoProviders.

In the case of a physical InfoProvider for basic InfoCubes (transactional and standard), the requests are routed to the storage services. The so-called Data Manager uses its Aggregate Manager to identify aggregate cubes and ultimately find the optimal database access path. The InfoProvider abstracts the storage and access details from the OLAP engine.

Virtual InfoProviders

The information requests flow into the InfoProvider layer in similar ways for both physical and virtual InfoProviders. The InfoProvider layer abstracts OLAP access from the Data Managers, and the Data Manager takes care to access the physical data storage if indeed the data is stored in SAP BW; if it isn't the Data Manager hands off the request to a remote system. The differences occur in how the requests are routed, and how/where the actual data access takes place. It is important to understand that the end user requesting information from SAP BW need not know or care where or how the information is physically stored.

We categorize virtual InfoProviders into two categories: remote and local. Remote InfoProviders are useful whenever data residing on a remote system cannot or should not be copied to the local database. Sometimes external data may not be copied to a

local system for licensing reasons, and other times there are functional or nonfunctional requirements that prohibit data from being copied from an operational system into SAP BW. This may be the case when real-time data from an operational system is needed but access to the data must be tightly controlled. Virtual InfoProviders may be local if, for example, the InfoProvider does not directly interface with the SAP BW Data Manager. Such is the case when MultiProviders are used as InfoProviders.

Local Virtual InfoProviders

MultiProviders are typically used whenever there is information from different business processes or parts of a business process that needs to be displayed and available for navigation within a single query. MultiProviders are an alternative to modeling overly complex InfoCubes or ODS objects. It is important to keep in mind that MultiProviders are unions of InfoProviders and do not allow defining joins. The MultiProvider takes the retrieval request it receives from the OLAP engine and determines, based on the meta data for the MultiProvider, the number of physical or virtual InfoProviders that comprise the MultiProvider. For more information on InfoProviders, refer to Chapter 4.

Remote InfoProviders

Remote InfoProviders are easily defined for data sources available on SAP R/3 and other SAP BW systems. In the SAP BW system requesting the data, an SAP remote InfoCube is defined with an InfoSource that points to another SAP system. The BW Service API is used to access the remote data on the SAP systems. For non-SAP systems there is a generic remote InfoCube that may be defined. This generic interface enables custom-developed programs to act like an SAP remote InfoCube and provide information to the requesting SAP BW system.

NOTE The only outbound SAP BW interface that does not utilize the InfoProvider layer is the ODS BAPI.

Virtual InfoProviders are a powerful way to satisfy reporting and analysis requirements without the need to physically reconstruct InfoCubes or create new application-layer ODS objects. Virtual InfoProviders are InfoProviders that do not route data selection requests directly to the SAP BW Data Manager. For example, remote InfoCubes are physically stored on a remote system supporting the remote InfoCube interface definition; these can be flat database tables, R/3 DataSources, and actual InfoCubes stored in a remote SAP BW system. The virtual InfoProvider merely hands the request off to another retrieval service, which takes care of the data selection.

The SAP remote InfoCube interface allows direct access to data stored in SAP R/3 systems (through the BW Service API) and external systems supporting this interface like the ACNielsen workstation, providing access to market information. Using the Remote Cube interface, SAP BW users are able to define remote cubes for any type of data stored on any kind of system. There is also an option that allows for the master data to be selected locally from SAP BW while the data records are selected from a remote system.

Note that SAP BW does not currently support the Open Database Connectivity (ODBC) or OLE DB protocols for access to non-SAP data sources. It does not appear that SAP would need to apply very much engineering effort to accomplish such openness. This openness would enable SAP to effectively decouple the presentation services and the analysis and access services from the storage and ETL services and provide SAP with a standalone product for the business intelligence space.

OLAP Engine

Regardless of whether or not an organization decides to standardize on the Business Explorer front end that is delivered as part of the SAP BW product suite, the SAP BW OLAP engine provides all analysis and navigational functions. While there are several integration possibilities for third-party, so-called alternative front-end packages, the OLAP engine is the analysis brain for SAP BW. Later in this section we will discuss each option and the possibilities for certifying third-party software to the interfaces. For now, let's focus on functions supported by the engine itself.

The Meta Data Repository stores a list of all queries that have been defined in SAP BW regardless of the tool that will ultimately present the results delivered from the OLAP engine. When a query is requested by a presentation service, whether the Business Explorer or a third-party program, the query definition is retrieved from the Meta Data Repository. The OLAP engine takes the query definition and generates, updates, and executes the queries by running the generated program. Once the selection request is handed to the InfoProvider interface, the InfoProvider may determine the type and invoke the appropriate service be that a storage, analysis, or remote system service. The InfoProvider then returns the selected data records to the OLAP engine for runtime calculations, currency conversions, and authorization checks.

The OLAP engine checks the query definition of the so-called query read mode. There are three types of query read modes, each type controlling the behavior of the OLAP engine. The read modes are used to accelerate performance and are set based on a number of factors. We will cover the read mode and other performance optimization options in greater detail in Chapter 10.

Analytical Services

A handful of analytical services are delivered with SAP BW that are performed outside of the OLAP engine. It is a common misconception—most likely promoted by competing vendors—that SAP BW only supports OLAP. This is not the case. As we discussed, tabular reporting formatted reporting and what we categorize as generic analytical services are all supported. Examples of the so-call generic analytical services are customer lifetime value analysis (CLTV), recency, frequency, and monetary value (RFMV) analysis for campaign optimization, and ABC analysis for customer classification. SAP has delivered additional Business Content, including cubes, ODS, and queries, that utilize these services. In version 3.0, CLTV analysis may be accessed via the command code `RSAN_CLTV`. Entering this in the OK command box takes you to the CLTV Modeling functionality. Once logged into the SAP BW Administration Workbench, you'll find the `OK` command on the upper left-hand side of the GUI. A prerequisite for this functionality is the availability and activation of ODS objects 0CRM_OLVM and 0CRM_OLVF.

In the context of analytical business applications, we discuss the analytic services in greater detail throughout Chapter 8.

Data Mining Services

Data mining is the term commonly used to describe a class of database applications that uncover previously unknown patterns from data. The goal of data mining is to identify golden nuggets of information in large volumes of data. Many terms have been used somewhat synonymously with data mining, for example, knowledge discovery, data archeology, information harvesting, or predictive mining. Some of these terms represent types or components of data mining.Regardless of the terminology, data mining algorithms are finding their way out of university research papers and into enterprise software applications. Our goal here is not to provide a discourse on data mining but to sketch the methods that SAP delivers in SAP BW to support the data mining processes.

SAP has delivered data mining methods and Business Content to help organizations identify potentially significant patterns, association, and trends that otherwise would be too time-consuming to uncover through conventional human analysis or that was missed, since analysts tend to see the information they are expecting or hoping to discover and may miss valid information that lies outside their expectation zone. SAP BW also supports third-party data mining engines like IBM's Intelligent Miner and emerging interface standards for data mining like the data mining extensions to OLE-DB and the Predictive Markup Language (PMML) for exchanging data models and result sets between different data mining engines.

SAP BW is delivered with the following three types of data mining methods:

- Classification
- Association analysis
- Clustering

An important data mining transaction code to remember is `RSDMWB`. This transaction will take you to the data mining workbench. For those readers with investments in IBM's Intelligent Miner, transaction code `MINING_IBM` will take you to the customization options for the integration with SAP BW. As SAP BW evolves and more analytic applications are built on the platform, there is little doubt that new data mining functionality will be created or integrated by SAP. In Chapter 8 we will describe the SAP BW data mining methods and how they are applied, for example, to customer relationship analytics and other business applications delivered by SAP and we will investigate these services in greater detail as we discuss analytic applications that utilize the services.

Other Core Services

In this section we focus on two core aspects of SAP BW. The first is a unique feature that allows for an analysis path to jump to various receiver objects. These receiver objects may be other SAP BW queries, Web applications, SAP transactions, ABAP reports, or Web URLs. This functionality is called the Report-to-Report Interface. The

second feature is the ability for an end user to personalize his or her experience. Personalization may be used to set commonly used variables, Web templates, and favorites lists.

Report-to-Report Interface

The *Report-to-Report Interface* (RRI) in SAP BW is a feature that enables information surfing. The RRI works in a similar way as Internet surfing, where the user jumps from one page to another following a path of predefined links from one Web page to the next. End users of queries may jump from one query in SAP BW to another query. The RRI enables linking together information based on the context of an end user's analysis path. For example, a sales manager may be analyzing her most profitable customers in a given year and from the profitability query jump to an accounts receivable query to analyze payment behavior. In doing so she may find that her most profitable customer is indeed not as profitable as first thought because of the drain on cash flow and increase in interest expense. The unique feature with the RRI is the ability to pass the context of the sending query—in this case, the specific customer number the sales manager analyzed the profitability of—to the receiving query as an input parameter. The RRI is configured in the Administration Workbench, which is not technically a part of the BEx Query Designer. The BEx Analyzer and the Web application's context menu use the *Go To* options in the context menu or toolbar to initiate the RRI.

The RRI is possible because of the common meta data definition in SAP BW. The meta data objects in the sending query will be matched to the meta data objects in the receiver. When a characteristic is common in both the sender and receiver, the value from the sender will be passed into the receiver if the receiver has a variable or input parameter. The RRI may help to avoid the problem of having particularly large or long-running reports, since information modelers may use a summarized report and still provide drill-down functionality into a more detailed query SAP BW or transaction in R/3. For example, you may have modeled your master data attributes in SAP BW in such a way that they are not necessarily representing the current value (refer to Chapter 4 for information modeling options), but the query users want to view or even maintain the master data for a certain characteristic. This may be accomplished by defining a receiver object for a query or InfoProvider that calls the appropriate maintenance transaction in R/3.

SAP BW's RRI allows you to call from a query (Sender) to another object (Receiver) by maintaining a Sender/Receiver Link. The Receiver may be any one or several of the follow objects:

- SAP BW query
- SAP BW Web application
- SAP BW formatted report
- InfoSet
- Transaction
- ABAP report
- Web address

The receiver objects may be located on remote systems or in the same logical system as the sending object. The technical settings on the remote system must be established with transaction SM59 so that the RRI will recognize the system as one with potential receiver objects. This functionality helps to support the InfoMart scenario we discussed in Chapter 4, where, for example, you have an InfoMart for the finance department that contains highly summarized data and a central data warehouse layer with detailed transactional level data. A query built on an InfoCube in the finance InfoMart may contain data at the month/period level of detail. The central data warehouse layer may contain the daily detail data. The receiver object in this case may be called when a specific analysis path requires details of the transactions that occurred within a month. This type of drill-through to the daily records (query built on ODS object) from a summarized report (query built on InfoCube) is not uncommon.

We mentioned that SAP R/3 transactions and ABAP reports are valid receiver objects. However, jumping back to R/3 from SAP BW might cause a few challenges. As we discussed in Chapter 5, the meta data and Business Content in SAP BW has been transformed from the meta data that exists in R/3. This poses a bit of a challenge for drilling through to R/3, but in most cases, it is achievable without coding a customer exit or routine. One of the preconditions for jumping back to an R/3 receiver object is that the data that was loaded into SAP BW was loaded through an InfoSource. The process of drilling back through to the R/3 system uses a kind of reverse meta data and data transformation in order to determine the appropriate field mapping to pass into and the appropriate characteristic values. Only trivial data transformations can automatically be reversed. Complex transformation rules may have to be reversed by implementing customer exits. There are two customer exits that should be used in the case where SAP BW cannot by itself derive all the information from the InfoSource definition: EXIT_SAPLRSBBS_001 and EXIT_SAPLRSBBS_002. The first exit is for the meta data transformation and the second for the data transformation.

Personalization

Personalization is the customization of information based on previously known or real-time preferences set by the consumers of the information. Personalization may take on many forms but has the goal of efficiently serving or anticipating the consumers' needs. While the personalization features in SAP BW are not only used by the BEx Analyzer but also in the BEx Web applications and Administrator Workbench, we will explain the concept and its underlying components here.

Prior to BW release 3.0, there was no useful server-side personalization for the Business Explorer Analyzer and very limited personalization available for Web reporting. The personalization options were configured on the client within the BEx Analyzer plug-in for Microsoft Excel. The limitations have created a burden on the data warehouse administrator, since the best way to achieve personalization was to create permanent variable assignments and limited access to queries and views through the use of authorizations and workbooks that were saved on the client. In typical SAP fashion the 3.0 release has not only overcome the personalization shortcoming of the previous release but lays a foundation for extension and integration with analytic applications.

There are three main areas of personalization in SAP BW:

- The automatic recording of previously accessed objects. This *history* is accessible to users of the BEx Analyzer, BEx Query Designer, and Web Application Designer. This is referred to as "open dialog" personalization.
- *Variables* may be automatically filled with personalized default values.
- Web applications and their underlying queries and query views may be bookmarked or added to a favorites menu. This is referred to as *Web template* personalization. Web templates and their usage will be discussed further in the next section as we discuss the BEx Web Application Designer.

Personalization provides information consumers with an experience that is customized to their preferences. The ease of navigation to specific Web queries from a favorites list or a history list are not difficult to imagine and are analogous to bookmarking your favorite Web site via your Web browser or adding a buddy to your buddy list in an instant messaging application. However, behind-the-scene personalization also impacts data warehouse administrators' jobs. The bad news is they have another data set to monitor and manage. The good news is the impact activating the personalization data is stored in specific ODS objects and may be monitored and managed via the Administrator's Workbench like any other ODS object.

Before using the three areas of personalization in SAP BW (variables, open dialog, and Web template), you must activate them. The activation process is straightforward and can be found in the implementation guide (IMG). Enter the transaction code SPRO and in the reference IMG under *SAP BW reporting relevant settings*. The process of activating a personalization option creates an underlying ODS objects that may be found in the Administrator's Workbench as part of the technical Business Content.

Personalizing History (Open Dialog)

When designing a query in the BEx Query Designer, the end user is prompted with a dialog window where he or she may select the InfoArea an existing query is stored in or create a new query. There are now selection buttons for history and favorites. These two options are similar in that they provide a customized list of queries to a query designer. They are different in that the history is automatically recorded based on the query user's most recent interaction with the Business Explorer, whereas the favorites list contains queries that the user has explicitly designated as his or her favorite. The history is stored in the ODS object 0PERS_BOD. The ODS object contains the history of all accessed queries, workbooks, and views for all users in the system. A user's favorites are associated with the role concept in SAP BW and not stored in the ODS object.

Personalizing Web Templates

When Web queries are accessed, a shortcut is automatically created in the user's favorites menu. This shortcut stores not only information about the Web query but also the information about specific viewing parameters selected by the end user. For example, say a user is conducting a drill-down analysis path while analyzing global sales revenue. The analysis ends with a detailed drill-down of the Americas region and the country of Argentina. This information is stored in the bookmark. The bookmark is an SAP BW URL that contains the parameters and values to re-create the ending place

of an analysis path. The ODS object 0PERS_WTE contains the data for Web template personalization.

Variable Personalization

Variable personalization occurs automatically after the feature has been activated in the IMG. Once a query is executed and a value for a variable is entered, the value is stored in the ODS object for the user/query/variable combination. When the query is subsequently executed, the values for the variables are retrieved from the ODS object and automatically entered as the default values for the query. The ODS object 0PERS_VAR contains the variable information. Since it also contains the variable history from cost and profit center planning, update rules automatically update the ODS object with this new information.

Currency Conversion

Support for foreign currency translation has always been one of the strengths of SAP and SAP BW is no exception. There are two primary places where currency translation is possible in SAP BW: during the update of a data target via the update rules as described in Chapter 6 and during reporting. Currency translation during reporting also has two options: during query definition and ad hoc while analyzing a query. Readers familiar with the currency translation options and configuration found in SAP R/3 will recognize the terminology, configuration tables, and functionality.

Currency translation is the seemingly simple process of taking a specific value that is represented in a specified source currency, retrieving an exchange rate, and calculating the new value in a target currency. For example, revenue recorded in EUR may be translated into USD by looking up the exchange rate for the two currencies at a given point in time and multiplying the source currency by the exchange rate to determine the target currency. Figure 7.14 shows the four parameters that must be defined or derived in order to perform a currency conversion: source currency, target currency, translation date, and exchange rate type.

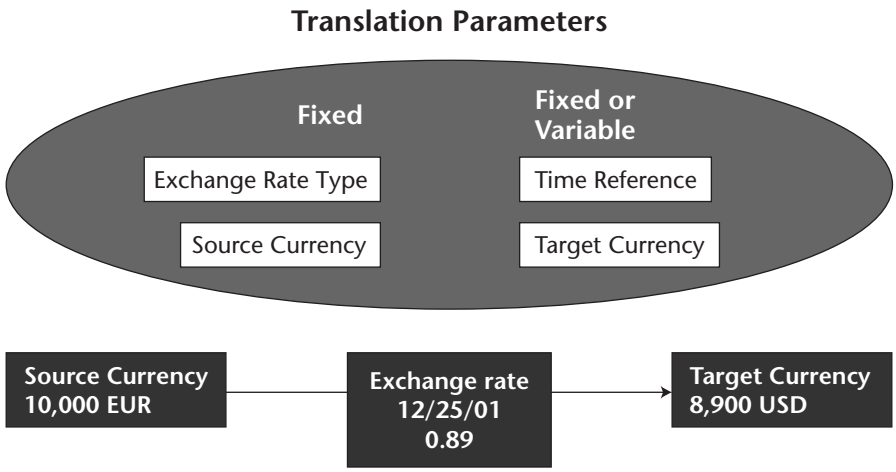


Figure 7.14 Currency translation.

Several currency translations may be defined or applied to the same query. During query design time a translation may be defined for each structure or key figure (restricted or calculated). During query execution a business analyst may right-click and from the context menu select the currency translation option. From there the analyst can select the currency translation type and target currency.

As is the case with SAP R/3, SAP BW has a configuration table where exchange rates from one currency to another are stored with an effective date. This is referred to as an *exchange rate type*. Exchange rate types are fixed values. The next parameter is the source currency. The *source currency* is the currency that is assigned to a specific key figure InfoObject in an InfoProvider. The currency of a key figure InfoObject may be fixed or assigned for each unique data record that is loaded into a data target, depending on the InfoObject definition in the Meta Data Repository. Regardless of the key figure InfoObject definition, the InfoProvider will send records to the OLAP engine for each key figure of type Amount. Query users will see this referred to as the database currency in the BEx Analyzer.

Currency translation may be established explicitly during query design time or ad hoc during query analysis. The next two parameters may have their values dynamically set. The time reference and target currency may vary depending on the values present in the context of a query navigation state. Table 7.3 highlights the options available for defining or deriving all four of the parameters.

While the currency conversion functionality is quite powerful because of its flexibility, we have seen that many organizations opt to design currency translation into the information model. In other words, they chose to store the data pretranslated as of a given time in a specific key figure.

Table 7.3 BEx Currency Translation Options

TRANSLATION PARAMETERS	PLACE DEFINED OR DERIVED	QUERY DESIGN OPTION	AD HOC OPTION
Source currency	Key figure in data record	X	X
Target currency	Fixed in trans. type	X	X
	InfoObject attribute	X	
	Variable entry	X	X
Exchange rate	Fixed in rate tables	X	X
Time reference	Fixed date in trans. type	X	X
	Current date	X	X
	Time characteristic value	X	

Content Management Framework

Integrating unstructured data with the so-called traditional data found in a data warehouse enables powerful collaborative business intelligence scenarios. SAP BW supports this requirement, since it includes a *content management framework* (CMF). The CMF is a collection of services that are the cornerstone for collaborative business intelligence and the knowledge management component of the Enterprise Portal. The SAP BW product uses these services to retrieve documents that are context-sensitive to an analysis path.

The CMF enables you to link documents stored in the SAP DB, or an HTTP content server, with the SAP BW meta data, transaction data, and master data objects. The documents may be used to provide administration advice or to supplement the documentation of the objects or to relate unstructured information to structured information. Common meta data objects that documents may be attached to include queries, InfoCubes, InfoObjects, and InfoProviders. Access to the documents is possible from the Administrator Workbench, from the BEx Analyzer, and from the BEx Web applications.

There are three main objects in SAP BW that may have documents attached to them: the meta data objects, master data values, and navigation state objects. Documents may be attached to enhance the documentation, to provide explanation of the data genealogy, or in the case of the documents linked to a query state, to explain variances from plan or assumptions used when forecasting. A *navigation state* is the set of attributes and measures that are being displayed to the end user at a specific stage in an analysis path. For example, a query may enable the analysis of the profitability of customers that purchased PopTarts, in February across all geographies. The combination of customer, profit, month, and geography and their specific values would be a query state. Documents may be linked to this state, as well as to the specific master data for the listed customers and to the geography meta data object, without regard to any specific geography.

Attaching Documents

You may attach documents to the objects two ways. The first way is by entering the Administrator Workbench and selecting a meta data object or master data value. The second way is to attach documents to a query state at analysis time either in the BEx Analyzer or in a BEx Web application.

Once in the Administrator Workbench, the documents for master data InfoObjects and meta data may be maintained. In the manual maintenance transaction there is an option for adding documents to a master data value. More than one document may be attached to a specific master data value. This allows for several file types to be attached. For example, you may wish to attach to a master data value for the InfoObject Personnel ID both a bitmap picture of the employee as well as a Microsoft Word document of their original resumé. During query analysis the type of document that is displayed may be controlled. More specifically, you need to set a parameter for the document

Web item to indicate which document type should be displayed. You may set the parameter at design time or analysis time. In both the BEx Analyzer and the BEx Web applications, you can view documents by selecting the context menu and the *Go to|Documents* or *Go to|Document Navigation State* option.

NOTE InfoObjects must be defined as a characteristic that supports document attributes before documents may be attached.

Attaching documents to a query state is a powerful way to capture in context comments from business analysts and use them to collaboratively solve business problems. Documents may be added at the time of analysis via a Web browser. This is a common method for both the BEx Analyzer and the BEx Web applications. A separate window in the Internet browser will be opened and the so-called document browser will be launched. In the document browser the characteristics and key figure will be listed for the query that launched the document browser. In the document browser window will be a listing of all the documents with their specific navigation state attributes.

Only documents with attributes that match the filter values for the selected navigation state are displayed. Characteristics that are aggregated (that is not implicitly selected) will not be displayed. By selecting the *More Functions* button, as shown in Figure 7.15, you will find the function for adding documents via the Web browser.

Internet Graphics Server

The Internet Graphics Server (IGS) represents the SAP attempt to standardize all graphics rendering for all their software components. The IGS takes data from SAP BW and other SAP components, as well as non-SAP components such as those from software vendor ESRI, and generates graphical output. The server is flexible enough to create graphics in many different formats to suit the application requiring the graphics processing. Available formats include BMP, JPEG, GIF, PNG, VML, and SVG. The SAP Web Application Server (WAS) is tightly integrated with the IGS even though the IGS is installed as a separate server. The two work together and communicate via an RFC connection that is configured both in the WAS system and in the IGS. When the IGS is being used with SAP BW, the server may be installed on the same physical machine as an SAP BW application server. The IGS is constructed in such a way that it will function in a heterogeneous environment, although in early 3.0 versions of SAP BW, the IGS must run on a Windows platform. The IGS is the foundation for Web items that are rendered in BEx Web applications as charts, graphics, and maps.

Information Access Interfaces

SAP has come a long way in overcoming criticism that its software is only extensible by programming with its proprietary ABAP language and that it has closed or inaccessible APIs. Figure 7.16 depicts the interfaces currently exposed for presentation services to invoke. We will examine the interfaces that are available to third-party independent software vendors and Web application developers, as well as the interfaces used by the Business Explorer.

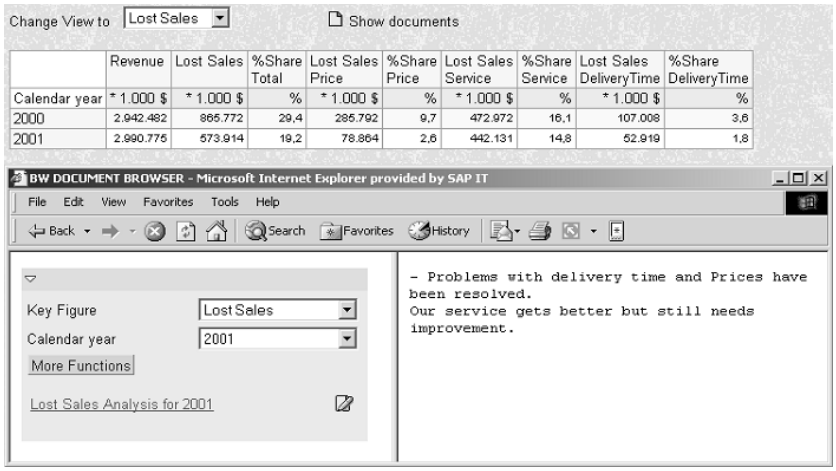


Figure 7.15 Documents for a query navigation state.

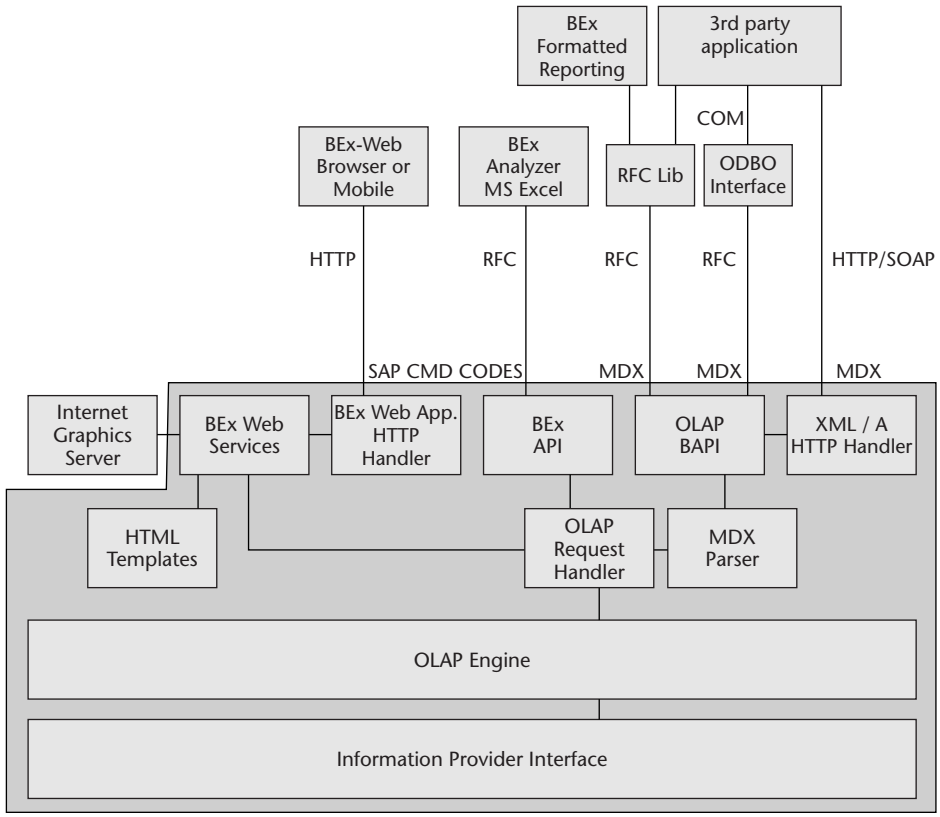


Figure 7.16 Interfaces for presentation services.

Interface Options for Third-Party Presentation Tools

In addition to the BEx Web API discussed earlier in this chapter, there are three recommend methods for integrating so-called front-end presentation tools with SAP BW. The first method supported by SAP was the OLE DB for OLAP interface. This interface was published with the 1.2 release of SAP BW and gained wide acceptance by the business intelligence tools vendors. However, this interface has several drawbacks, so a platform-independent business application programming interface (BAPI) was published with release SAP BW 2.0. While this interface provided a greater flexibility in the choice of operating systems and platforms that the third-party tools could run on, it did not support access through HTTP. The BEx Web API provides a command framework for accessing SAP BW information from a URL.

Note that regardless of the interface option chosen by the third-party software vendor, the OLAP engine performs the analytical processing and requests the data from the data provider as previously described in this chapter. The actual path that a request for data takes prior to arriving at the OLAP engine, however, is quite different.

NOTE ODBO is the acronym for the acronym OLE DB for OLAP or Object Link Embedding for Databases extended for Online Analytical Processing.

The connection protocols supported by SAP for third-party vendors to select from are COM, RFC, or HTTP. Most business intelligence tool vendors have chosen to support at least one of the protocols in order to meet the evolving demands of their customers. The leading business intelligence tool vendors support one or more of the interface protocols available for accessing SAP BW. For an up-to-date listing of certified vendors refer to www.sap.com/partners/software/directory. Because of the relatively recent support by SAP of the XML for Analysis (XML/A) and the Simple Object Access Protocol (SOAP), you should refer to SAP's Web site for the most up-to-date list of vendors.

OLAP BAPI

There has been a bit of confusion about the number and type of OLAP engine interfaces that SAP BW supports. The confusion has been primarily caused by the term *OLAP BAPI.* When SAP first exposed the analytic functions of the OLAP engine, there was a recommended approach to integration: the OLE DB for OLAP interface (ODBO). Application programmers would write to SAP's ODBO driver, and SAP would take care of the connection back to SAP BW and call the proper function via the OLAP API. The criticisms of the OLE DB of OLAP interfaces were varied, but the main issues were that the driver only supported Microsoft platforms and SAP's interpretation and that eventual extension of the ODBO protocol was different from other vendors' interpretations. To alleviate platform concern, SAP decided to document and publish the OLAP API, thus turning it into a BAPI that would be supported through release changes. Up until SAP BW version 2.0 (when SAP added the B in front of API) vendors could write to the OLAP API, but they had no guarantee that SAP would not change the interface in subsequent releases. In Figure 7.16 we have not distinguished between the OLAP API and OLAP BAPI, as the latter is the more current terminology.

You may notice that all three supported connection protocols eventually access the OLAP BAPI. There are two main groups of functions that are supported by the OLAP BAPI. The first group is the browsing of meta data and master data. The second group is the fetching of data and execution of multidimensional expressions. The naming of the interfaces may be the cause of the confusion, since the ODBO, OLAP BAPI, and XML/A interfaces all access the OLAP BAPI. While we believe that XML for Analysis will become the interface of choice, ODBO is the most common method for third-party tools to integrate with SAP BW. Nearly all of the leading business intelligence tools on the market today have integrated their products with SAP BW via the ODBO interface.

Note that the ODBO interface calls the same underlying BAPIs that application programmers may call via the RFC libraries delivered by SAP. The primary difference is the amount of knowledge a programmer needs to know about the connectivity calls. With ODBO the SAP BW-supplied driver abstracts many of the details, whereas the RFC library approach does not. While the interfaces have changed, the use of multidimensional expression has remained constant.

OLE DB for OLAP

To understand ODBO you must first understand its foundation. OLE DB is a set of interfaces that Common Object Model (COM) objects expose to provide applications with uniform access to data regardless of the data's type or location (www.microsoft.com/data/oledb/olap/faq.htm 2001). The OLE DB protocol has consumers that request information and providers that provide information. The providers do so by exposing the interfaces of their objects to consumers. OLE DB for OLAP is the addition of two interfaces to the OLE DB protocol. These interfaces provide access to multidimensional data sources. The SAP BW ODBO driver is a provider in this protocol.

ODBO PROVIDER

There are two types of providers: data providers and provider services. The data provider owns data and exposes it to the consumer as a row set or table. The provider services make certain the data exposed by the BW server is returned to the third-party consumer, somewhat like a router sitting between a consumer and the ultimate data provider. The provider service in the case of BW is the mdrmsap.dll.

There are actually several libraries and files that are needed to support calls back to the SAP BW application server and to invoke an OLAP BAPI:

- **mdrmsap.dll—The SAP BW OLE DB for OLAP Provider**
- **librfc32.dll—SAP's main RFC library**
- **wdtlog.ocx—Dialog box for login parameters**
- **mdrmdlg.dll—Manager for the connection to an SAP system**
- **mdxpars.dll—Parser for multidimensional expressions so they may be sent to BW**
- **scerrlklp.dll—Error-handling routines**

Source: **SAP AG ASAP for BW**

This major drawback in using OLE DB, or its predecessor the Open Database Connectivity Protocol (ODBC), is that they require a client component to be programmed to interface with a specific provider component. The provider component for SAP BW is delivered and installed on the client machine with the SAPGUI and SAP BW front-end add-on. The installation is necessary in order for the consumer application to access an SAP BW provider. This causes a number of client and server dependencies, such as version, platform, and in some cases, programming languages.

Many of the third-party BI tools certified to this interface ran into significant challenges that prevent enterprises from fully deploying an alternative the BEx front end. The challenges vary widely based on the architecture of the BI tool. The protocol definition left consumers and provider to interpret how certain functions should be supported. There were a few common areas where consumers and SAP, as the provider, had differing interpretations. For example, the support variable intervals, hierarchies, and calculated members were all different enough that either SAP or the BI vendors would need to rework their products or isolated portions of their products. These challenges limited the rollout of alternative front ends to SAP BW.

Recent enhancements to the level of protocol support by SAP are encouraging signs that third-party BI vendors will be able to satisfy their clients and large installed bases with meaningful integration to SAP BW. However, this tight coupling of client and server is not suitable for Web applications that are inherently stateless—not to mention that the protocol is dependent on the Microsoft Windows platform.

XML for Analysis

One of the newer developments in the BI community's adoption of XML-based standards. *XML for Analysis* (XML/A) was defined by Microsoft and Hyperion Solutions and is said to advance the concepts of OLE DB that utilizes SOAP, XML, and HTTP. Since OLE DB is based on the Common Object Model and XML/A is based on the Simple Object Access Protocol, programmers will need to adapt their existing ODBO application to support XML/A. The communication API is designed to standardize access to an analytical data provider (OLAP and data mining) over the Web. It attempts to overcome the shortfalls of ODBO by decoupling the consumer and the provider. While ODBO was designed to standardize data access across data sources, and to some extent it did so, the technique required a client component to be deployed in order to expose COM or DCOM interfaces. XML/A is designed to eliminate this constraint.

The OLAP BAPI serves as a basis for the XML/A interface. The Web application that is requesting data from the SAP BW Application Server first passes through a so-called XML/A request handler. The request handler validates the inbound request, parses it, and passes the request to the OLAP BAPI.

In Figure 7.17 the provider Web service and data source represent SAP BW. Since SAP BW software components are built on top of the SAP Web Application Server, all necessary technology underpinnings are in place for organizations to expose SAP BW based Web services. Organizations would only need to describe the Web services they wish to expose to their trading partners by creating an XML document called a Web Services Description Language (WSDL) and register the document with a Universal Discovery Description and Integration (UDDI) directory. Loosely coupled, stateless applications may run across platforms in a language independent environment.

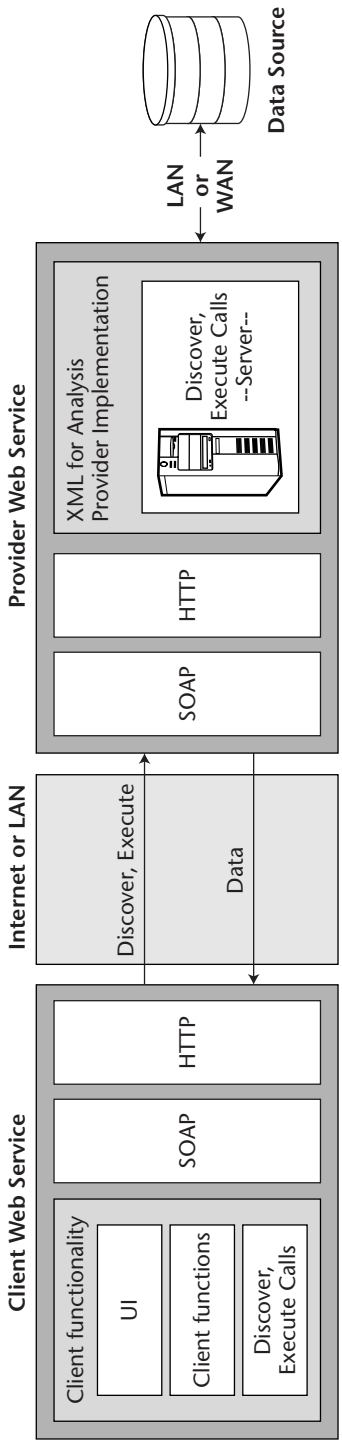


Figure 7.17 XML for Analysis.
Source: Microsoft Corporation

Both XML/A and ODBO send Multidimensional Expressions MDX commands to the OLAP BAPI and an MDX parser. The parser takes care to transform the expression into a format that the OLAP engine understands (i.e., ABAP). Once parsed, the request is sent to the OLAP request handler, which in turn invokes the OLAP engine. We find it somewhat ironic that the MDX is translated into ABAP prior to being passed to the Data Manager because if the MOLAP aggregate option is implemented, the request will be retranslated back into MDX for MOLAP aggregate selection. Nevertheless, the ABAP is generated and the appropriate InfoProvider is called.

Preparing for Third-Party Access

When a third-party application requests data from SAP BW, an MDX is used, unless data is being requested from an SAP BW Web template. If it is being requested from a Web template, an SAP BW URL is used. In this section we briefly discuss the steps taken to expose SAP BW queries to third-party applications that use the OLAP BAPI.

One of the challenges application programmers familiar with MDX may have when working with SAP BW is terminology. SAP is notorious for creating new names for terms that are generally accepted in the industry. SAP BW is no different. Table 7.4 contains a list of terminology differences between the Microsoft's OLE DB objects and SAP BW's objects.

SAP has included a tool to assist programmers in translating object definitions from one set paradigm to the next. Transaction code `MDXTEST` enables developers to test the result of their expression within the SAP BW Administration Workbench. This transaction and other query related administrational tools are briefly discussed in Chapter 9. If you are interested in more detailed information about how to write programs that access the ODBO interface, refer to the ASAP for BW Accelerator OLE DB for OLAP, which may be found on SAP's public Web site at www.sap.com/partners/software in the Business Intelligence section.

Table 7.4 OLE DB Objects and SAP BW Objects

OLE DB	SAP BW
Catalogs	InfoCubes
Cubes	Query cubes
Dimensions	Characteristics
Hierarchies	Hierarchies
Levels	Hierarchy nodes
Members	Characteristics
Measures	Key Figures

If you would simply like to execute SAP BW queries from a certified third-party tool, you need keep the following in mind:

- Queries must first be defined in the BEx Query Designer. To date, it is not possible to create the query definition in the third party too.
- The BEx query must be made available for OLE DB for OLAP. This is done during query design time by selecting the *Query Properties* icon and checking the *ODBO* radio button, as shown in Figure 7.18.
- In SAP BW version 3.0 it is now possible to directly access the InfoProvider without first creating a query. A so-called virtual query that contains all characteristics and key figures of an InfoProvider is accessed in this case.

In the third-party tool the end user would typical start by creating a query and selecting a data source. Selecting the SAP BW data source results in a list of BEx Queries (after the user has to be authenticated via the SAP login dialog). In the third-party tool the end user will see the BEx Queries listed as cubes (the OLE DB terminology) with the naming convention <*InfoCube name*>/<*Query name*>. These cubes may

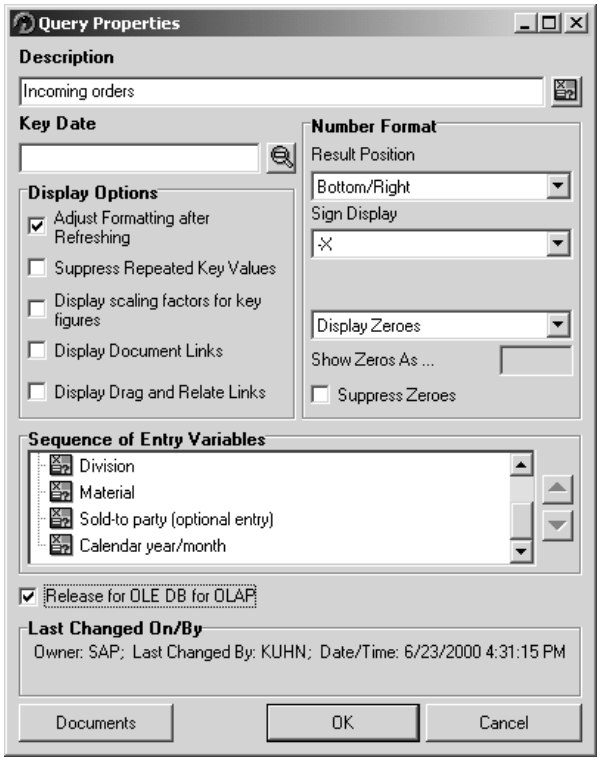


Figure 7.18 Activating OLE DB for OLAP.

be customized in the third-party tool by the end user but only within the framework of the originally defined BEx Query—that is, members and measures in the underlying InfoCube may not be added to the query cube by the third-party tool. Note that not all of the OLAP functions that are available in the BEx Analyzer are available via the ODBO protocol and not all the ODBO functions are available in the BEx Analyzer.

NOTE If the BEx Query has not been released for OLE-DB for OLAP, it will not be accessible via the OLAP BAPI, XML/A, or ODBO interfaces.

Third-party Web application developers and independent software vendors (ISVs) are now confronted with an interesting option to use the BEx Web API or the OLAP BAPI to encapsulate the functionality of the SAP BW OLAP engine in their products without creating a maintenance and upgrade nightmare for themselves and their customers. At this writing, there was no statement from SAP on its intent to support the BEx Web API and its commands and parameters through release changes as it does with its BAPIs. Until such a statement is released, there is a risk that applications built to the API may need to be changed should SAP change the interface in a future release. We believe SAP will commit to not changing the specification in time. We also believe that SAP will not ask vendors to certify their products to the Web API. However, until an official statement of direction is made by SAP, ISVs that seek to certify their products with SAP BW will be making BAPI calls to the OLAP engine and sending multidimensional expressions.

The Business Explorer API

Like the third-party interface options, the SAP front-end tools have also evolved and undergone naming changes. The *Business Explorer API* connects the Business Explorer Analyzer and the Business Explorer Web Services (the SAP BW reporting and analysis front-end tools) to the OLAP engine, allowing access to all available queries. The Business Explorer API is not an open interface available to be used by other applications. As it is not a BAPI, the Business Explorer API, is subject to change from one SAP BW release to the next. SAP has come under some criticism from leading business intelligence tool vendors for not exposing all of the functionality that is found in the Business Explorer API in the OLAP BAPI; however, this was primarily a result of the loose interface definition by Microsoft. In fact, the OLAP BAPI only recently supports certain hierarchy, variable, and time functions. The BEx API, unlike the OLAP BAPI, supports a proprietary set of command codes in place of the multidimensional expressions used in the OLAP BAPI.

NOTE Business Application Programming Interfaces (BAPIs) are documented and supported methods of business objects that exist in SAP applications. BAPIs are the recommended method for integrating applications from independent software vendors with SAP applications.

There are two primary services that the Business Explore front-end components utilize: the BEx API and the BEx Web API. The BEx API is nearly analogous to the OLAP BAPI; however, the interface is closed and utilized only by the BEx Analyzer, running as a Microsoft Excel add-in, and the BEx Query Designer. We believe that over time SAP will work to standardize access to the OLAP engine through two methods: via the OLAP BAPI with its three options (native, ODBO, or XML/A) and via the Web component framework and its BEx Web API (see BEx Web Services in Figure 7.16). The development philosophy appears to be first developing additional functionality for the Business Explorer components and then expose those interfaces, if possible, through the OLAP BAPI.

The BEx API enables the BEx Query Designer to access meta data about InfoProviders and, as its name implies, the creation of queries. The BEx API also services the runtime requests generated by the BEx Analyzer as end users requests query views and analytical functions. The BEx API, like the OLAP BAPI, interacts with the OLAP request handler so it may send and receive commands to and from the OLAP engine. The most significant characteristic of the BEx API is that it enables the BEx Analyzer to expose functions that may not be exposed through the OLAP BAPI.

It is a much simpler process for SAP to extend a proprietary protocol than it is to extend and test an industry-standard protocol. For example, the use of variables is a powerful concept that has been pervasive across all SAP software components. The variable concept enables the filtering of a request by single values, a list of values, an interval, or list of intervals while allowing the variables to have default values set based on an individual's personalized settings. This concept has been a part of SAP BW since the 1.0 version; however, it was not until 2.0b that SAP enhanced the ODBO protocol to expose the feature to third party applications.

Summary

In this chapter we detailed the services for presenting, accessing, and analyzing information and touched on how these services interact with the storage services. We described the five main components to the Business Explorer. The BEx Analyzer, BEx Query Designer, BEx Formatted Reporting, BEx Mobile, and the BEx Web Application Designer together are the suite of tools to access, analyze, and present information from SAP BW. Information is retrieved from SAP BW via the InfoProvider interface and returned to the information consumer after it is processed either in the OLAP engine, analytic services, or data mining services. The information consumer will receive a view of the query cube for each step in the analysis navigation. The integration options for third-party software applications send multidimensional expressions to SAP BW. SAP BW interfaces take multidimensional expressions from the client application and process them accordingly. The adoption of the MDX and XML for Analytics standards has created an easily extensible platform for analytics. Furthermore,

SAP has published a BEx Web API that enables SAP BW Web applications to be incorporated in any HTML-based application by simply added a valid SAP BW URL and integrate it into the Enterprise Portal. SAP BW also supports the integration of unstructured documents to be attached to meta data objects and query navigational states.

Now that we have completed the core architecture of SAP BW, in Chapter 8 we will look into how it may be used as a platform to support analytic applications. The final two chapters will focus on the administration of the system and the performance optimization options.