OLE DB for OLAP

BUSINESS INFORMATION WAREHOUSE

A description of the OLE DB for OLAP support of SAP´s BW Release 2.0B

Document Version 1.0

Table of Contents

1 Business Information Warehouse OLE DB for OLAP

1.1 Overview

OLE DB for OLAP (ODBO) is implemented in SAP's Business Information Warehouse, to provide the ability to query data from an OLAP server. ODBO is a set of COM (Component Object Model) objects and interfaces that extends OLE DB to provide access to multi-dimensional data structures. It is designed to facilitate communication between providers and consumers of multi-dimensional data. In other words, this interface provides third-party vendors or customer developers an industry standard method of accessing BW data via their front-end presentation tool.

The rational of selecting OLE DB for OLAP as BW's front-end interface are as follows:

- A broad industry standard interface to support all flavors OLAP architectures and vendors
- Leverage OLE DB architecture for reusable and parallel components
- Common API drives creation of generic OLAP tools
- ADO/MD simplifies task of application development
- OLAP services on top of any provider, hence no need to move data.

Limitations that should be taken into account when selecting ODBO as BW´s frontend:

- Though there is a coordinated development effort between 3rd party vendors and BW which is followed by a test and certification procedure, the integration will not reach the degree of integration that is given with the SAP front-end tool provided with BW
- BW comes with preconfigured information models, predefined InfoSources, InfoCubes and reports. These models reflect the SAP business experience and accelerate the BW implementation. The use of this information model from 3rd party vendors is limited to the InfoCubes. Standard reports are not provided. Decision makers have to wait for standard reports to be modeled.

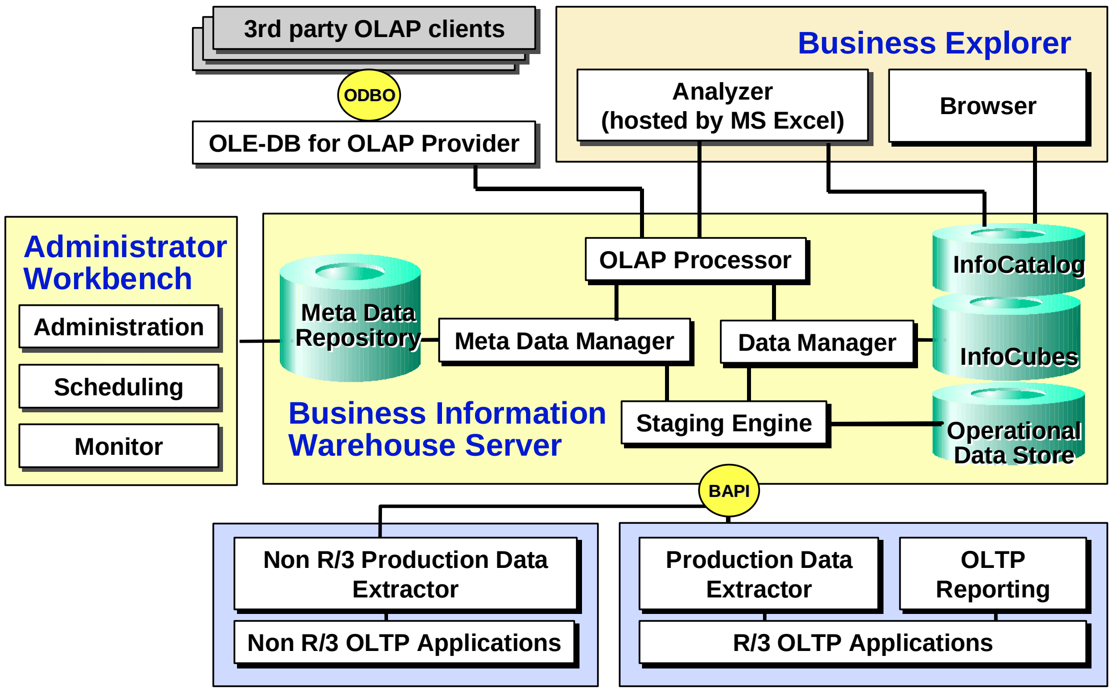
Software vendors who are developing products that consume, expose, and use OLE DB for OLAP to provide Universal Data Access and functionality to users involved in online data analysis break down into data providers and data consumers.

1. A *consumer* is any piece of system or application code that needs access to a broad range of data and consumes an OLE DB interface. OLE DB compliant data consumers will be able to efficiently access all relational databases through existing ODBC drivers.
2. OLE DB providers can be classified into two classes. A *data provider* is any OLE DB provider that owns data and exposes its data in a tabular form as a rowset. A *service component* is any OLE DB component that does not own its own data, but encapsulates some service by producing and consuming data through OLE DB interfaces.

1.2 BW Architecture

Figure 1 shows the architecture of SAP's Business Information warehouse with its three layers:

- The layer of R/3 and non-R/3 OLTP systems with the Data Extractors
- The Business Information Warehouse Server with its central OLAP Processor and Meta Data Manager
- The layer of frontend applications with SAP BW Business Explorer Analyzer hosted by Excel

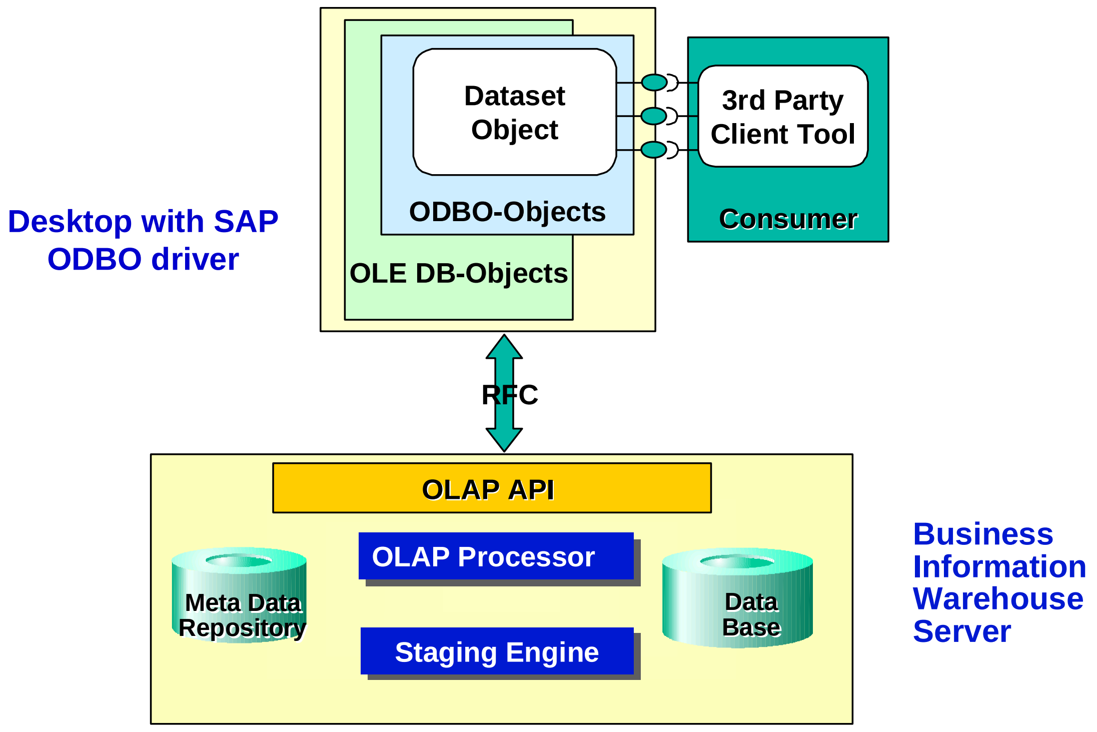


Client applications may consume directly the low level ODBO interfaces. This gives the client application maximum control and flexibility. Another scenario is that ODBO clients make use of a high level programming interface: the ActiveX Data Objects MultiDimensional (abbreviated ADO MD) which can be easily used in Visual Basic (VB) or VB scipt programming environments.

The BW ODBO implementation is based on the release 2.1 of Microsoft's OLE DB for OLAP Programmer's Reference.

Figure 2 shows in more detail the components of the BW ODBO provider implementation, which basically consists of two components:

One component is the **BW ODBO provider** that sits between the data provider and consumer and manages/modifies the data exposed by the data provider. The SAP BW ODBO setup installs a DLL on the desktop, where the client application is installed too. The BW provider is registered in the systems registry under the name MdrmSap. The provider dll on the desktop connects via RFC to A set of **function module APIs** on the BW server which closely resembles the interfaces of the ODBO objects.



Multidimensional result sets or datasets are queried in ODBO by means of a Multidimensional Expression (MDX) statement. The mdx grammar is documented in Micosoft´s ODBO programmers reference.

In the following table, service component developers of BW and ODBO client applications find some hints concerning relationships between the BW objects and the objects that appear on the ODBO interface.

`ODBO terminology`	BW terminology	Comment
`Dimension`	Characteristic	In BW the term Dimension is used for a group of related Characteristics (e.g. the time dimension groups related characteristics as CalendarYear or FiscalPeriod) In BW terminology, characteristics are classifications such as division, region, customer group and organizational unit on which the key figures such as revenue, sales quantity, and number of employees are based
`Measures`	Key figures	Key figures/ `Measures` are treated in ODBO as members of a special dimension (the measures dimension). Key figures are quantifiable values such as revenue.
`Members`	Characteristic Values	Characteristic Values may carry additional information.

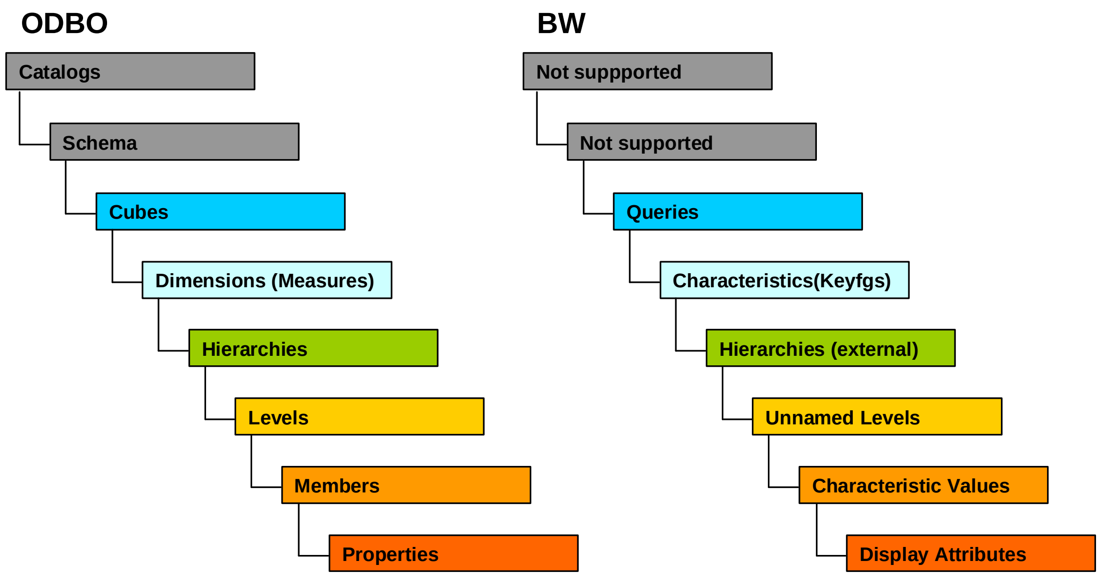
`Dimension Properties`	Attributes	The additional information of characteristic values is expressed by Attributes. E.g. a material has attributes like colour, weight ..
`Dataset`	Query	A dataset represents a multidimensional result set. A dataset can be defined in ODBO with an mdx statement

BW InfoCubes are similar to ODBO cubes – both are the central metadata objects and containers for data used for reporting. InfoCubes contain two types of data: key figures and characteristics. On the implementation level, the InfoCubes are sets of relational tables arranged according to a star schema. The concepts of catalogs and schemas are not currently supported by the BW.

The current version of SAP´s OLE DB for OLAP implementation gives access to the information held in the InfoCubes through queries which are created with the Business Explorer Analyzer.

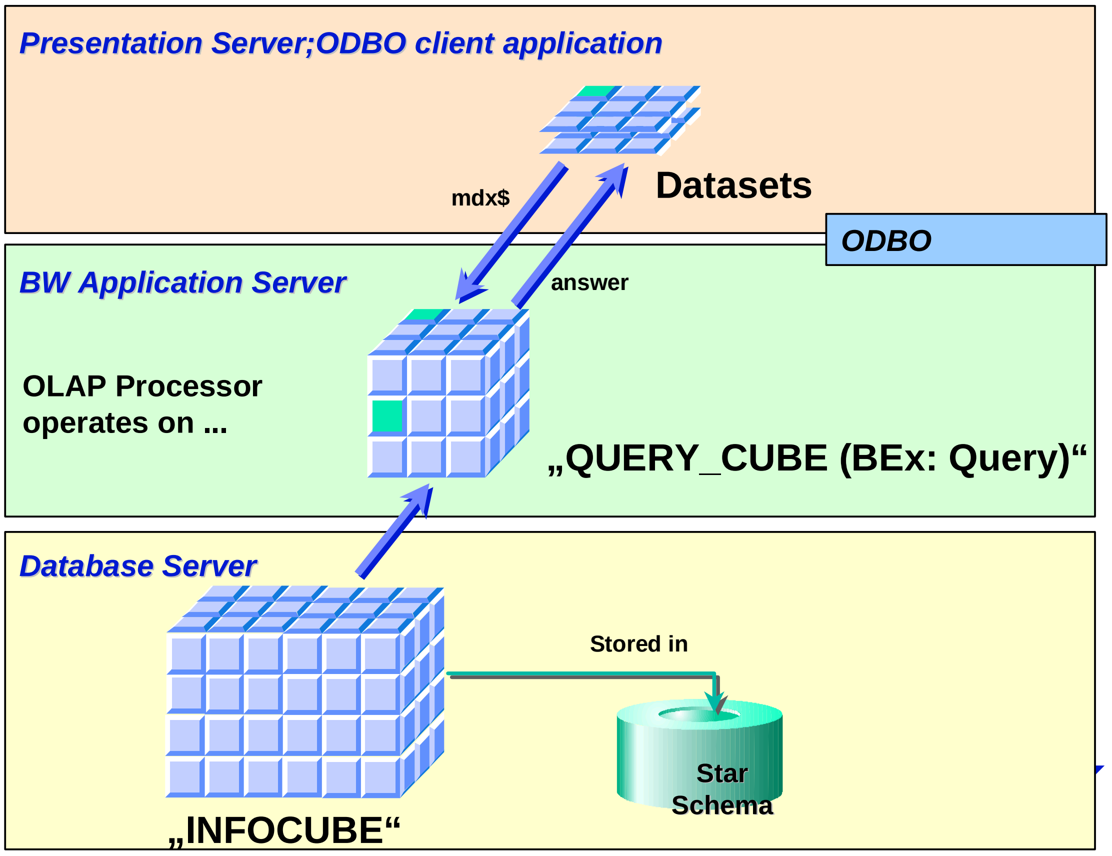
The mapping can be done, since Queries have essentially the same structure as `Cubes`. The queries select characteristics, key figures and derived key figures from the underlying InfoCube or the meta data repository respectively. The multidimensional dataset represented by this entity is termed a QUERY_CUBE. `Query_Cubes` are enabled for use in the OLAP API within the BEx Analyzer by setting a check box "OLE DB for OLAP release" in the query settings dialog. The only restriction for the selection of queries is, that the queries shall not use any variables with processing by user input.

Fig 3 shows the mapping of BW objects to the objects of the Multidimensional Schema defined by ODBO.



Note: There is no replication of data for building `Query_Cubes`. The only overhead data, that is stored persistently on the database is the definition of the query itself. `Query_Cubes` are populated only during the runtime of a query, and only to the extent, which is necessary to create the dataset, which has been specified by the mdx statement being processed.

Fig.4 shows the flow of data.

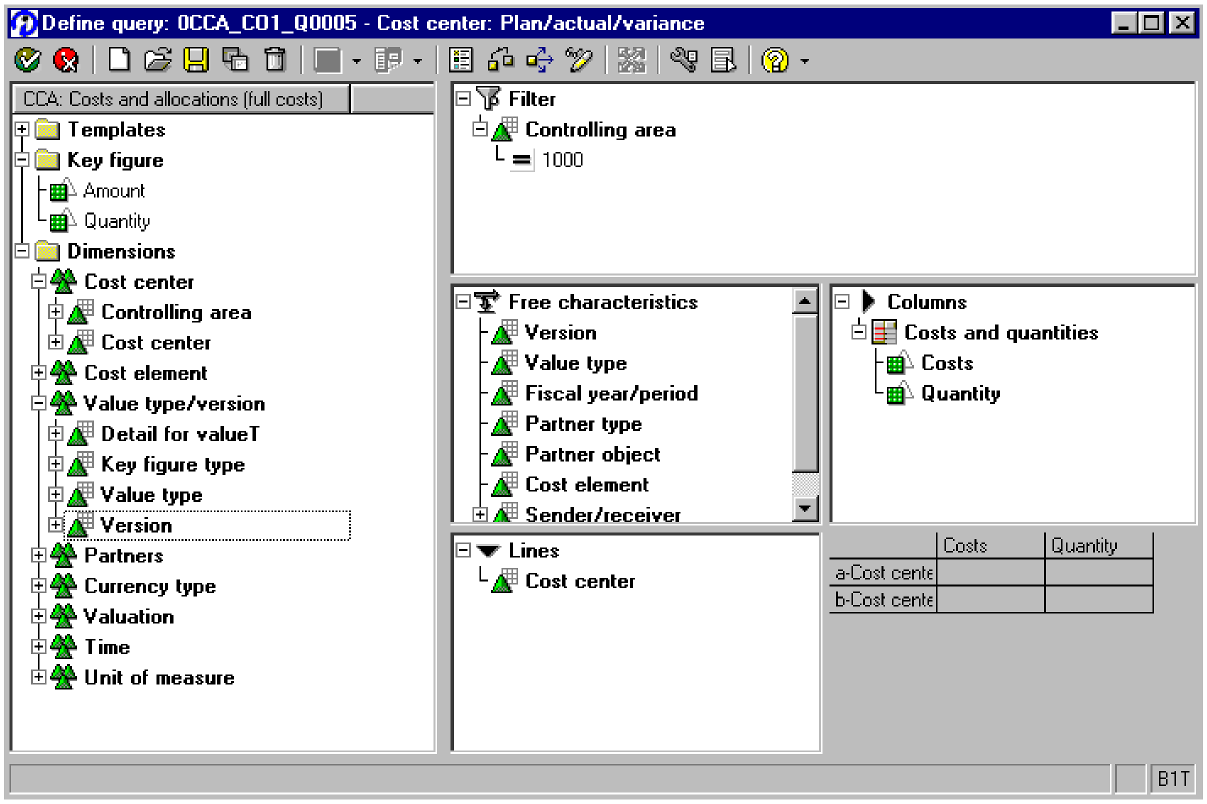

Presentation Server;ODBO client application
Datasets
mdx$
answer
ODBO
BW Application Server
OLAP Processor operates on ...
„QUERY_CUBE (BEx: Query)“
Database Server
Stored in
Star Schema
„INFOCUBE“

Derived key figures give ODBO consumers access to complex key figure definitions. In later releases, these will also be accessible by the CALCULATED MEMBERS feature of the MDX Grammar. Besides performance advantages and a reduced complexity for the consumer, one apparent advantage of defining derived key figures or - in terms of ODBO calculated members - on the providers side of the interface, is to ensure a consistent interpretation and use of derived key figures for different application areas.

1.3 Sample Definition of a Query Cube

The figure below shows an elementary example of a `Query_Cube` definition with the BEx Analyzer.

The most straightforward use of the Query Editor for the definition of a `Query_Cube` is to define a query with a single structure named "measures" and drop all required key figures from the key figures-folder of the tree with available objects to this structure. For clarity, it is recommended that this structure be named "measures" as done in the example, although any other name will also work. If necessary, derived key figures can be added to the key figures available in the selected InfoCube. The members created in this "measures" structure will be mapped on the ODBO interface to members of the Measures dimension.

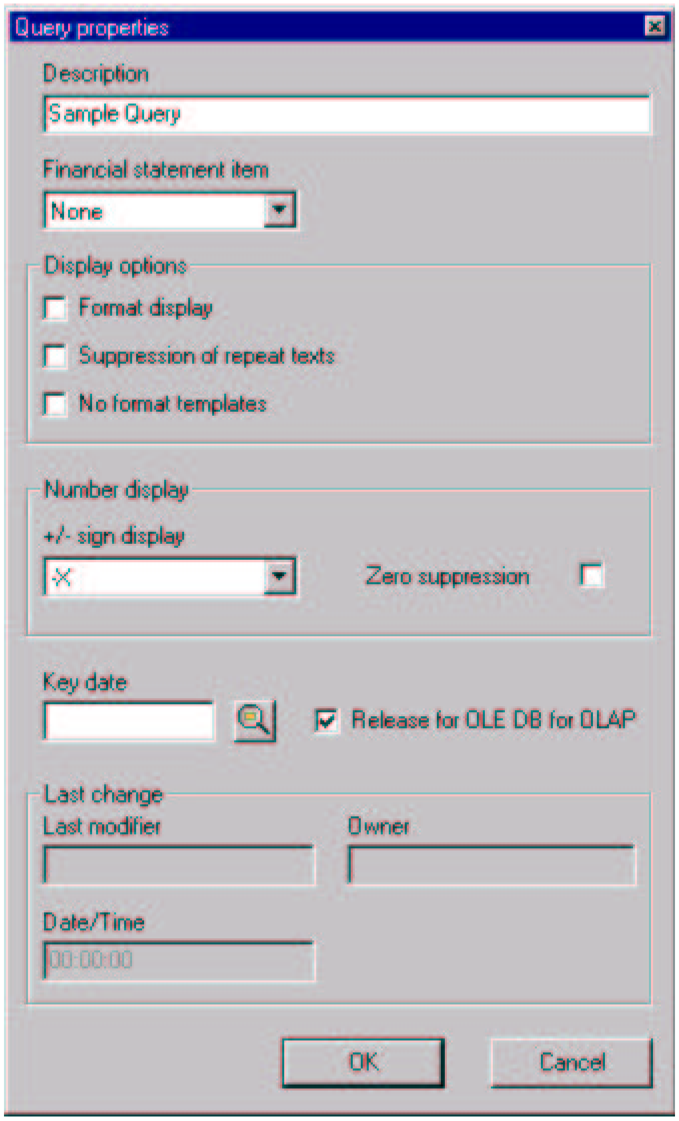


	Costs	Quantity
a-Cost cente		
b-Cost cente		

The selection of further dimensions for the QUERY_CUBE is done by simply dragging characteristics from the tree with available objects to the folder with aggregated characteristics of the query definition or to the columns and rows folder respectively. It does not matter, in which folder you drop the characteristics, since this information is significant only for the display of the query with Business Explorer Analyzer. In the figure the characteristics Version, Value type, Fiscal year/period, Partner type, Partner object, Cost element, Sender/receiver and Cost center have been selected. The selected characteristics will show up in the DIMENSIONS schema rowset as dimensions.

It is possible to set filters within the Query definition to restrict the dataset to a defined subset of data of the underlying InfoCube.

Before saving the query, set the check box "OLE DB for OLAP release" in the query properties dialogue to on. This enables the query to be displayed as a QUERY_CUBE in the CUBES schema rowset. The description in this dialogue will be mapped to the description of the CUBES schema rowset. The other properties in the dialogue have no relevance for the ODBO interface.



Description: Sample Query

Financial statement item: None

Display options:
- [ ] Format display
- [ ] Suppression of repeat texts
- [ ] No format templates

+/- sign display: -X

- [ ] Zero suppression
- [x] Release for OLE DB for OLAP

Date/Time: 00:00:00

For in depth information on more advanced features of the BEx Analyzer the reader is referred to the online help.

1.4 OLE DB Leveling

OLE DB defines a set of interfaces for accessing a diverse range of data types, located in a variety of data stores. OLE DB data providers only expose the interfaces that reflect the natural functionality of their data store. Providers all support a base level of functionality. Above that base level, common service components provide interoperability through generic implementations of extensions such as scrolling, or predicate-based positioning if the provider doesn't support them.

The leveling rules that are described in the ODBO programmers reference, define the bandwidth in which providers may vary within their implementation and still fulfill the ODBO compliance criteria. How the leveling rules apply for the BW ODBO implementation is documented within this document. Client applications should check the properties given in the DBPROPSET_DATASOURCEINFO of the BW provider, to get accurate information about the actual variations of the SAP provider among other provider.

1.5 Application of Leveling Rules

This chapter gives a detailed description of the features and leveling rules the actual implementation of the BW OLAP API and ODBO wrapper object will support. The first section is a commented list of leveling rules which apply to the BW ODBO provider. Notes added by SAP are indicated with the prefix "NOTE by SAP:".

The second section gives a list of the ODBO specific properties of DBPROPSET_DATASOURCEINFO and their respective values.

1.5.1 DataSource type

DBPROP_DATASOURCE_TYPE

- BW 2.0A: Multi-dimensional (DBPROPVAL_DST_MDP)
- BW 2.0B: Multi-dimensional and tabular (DBPROPVAL_DST_TDPANDMDP)

1.5.2 Flattening

MDPROP_FLATTENING_SUPPORT

The OLE DB for OLAP Provider supports complete flattening.

(MDPROPVAL_FS_FULL_SUPPORT)

1.5.3 Named levels

MDPROP_NAMED_LEVEL

OLE DB for OLAP Provider supports Named Levels (MDPROPVAL_NL_NAMEDLEVELS)

1.5.4 Join Cubes

MDPROP_MDX_JOINCUBE

The OLE DB for OLAP Provider supported only a cube in the FROM condition (MDPROPVAL_MJC_SINGLECUBE)

1.5.5 Number of axes

MDPROP_AXES

10 axes are currently supported. Furthermore, there is a restriction, that a maximum of 18 dimensions plus the measures dimension may be used in an MDX command at the same time. For each dimension, a maximum of one hierarchy may be used in an MDX command at the same time.

1.5.6 Update of cell data

MDPROP_AGGREGATECELL_UPDATE

Updating cell data is not supported (MDPROPVAL_AU_UNSUPPORTED)

1.5.7 RangeRowsets

MDPROP_RANGEROWSET

- BW 2.0A: Not supported (MDPROPVAL_RR_NORANGEROWSET)
- BW 2.0B: Read only (MDPROPVAL_RR_READONLY)

1.5.8 Filters / Slicer axis

MDPROP_MDX_SLICER

BW 2.0A and BW 2.0B support several tuples in the filter (MDPROPVAL_MS_MULTIPLETUPLES) providing all of these tuples belong to exactly one dimension, OR can be created as CROSSJOIN(s) via sets for exactly one dimension each.

1.5.9 Outer Reference

MDPROP_MDX_OUTERREFERENCE

Not supported

1.5.10 Use of attributes / Properties in the MDX

MDPROP_MDX_QUERYBYPROPERTY

- BW 2.0A: Not supported
- BW 2.0B: Supported in the commands ORDER and FILTER

1.5.11 Case statements

MDPROP_MDX_CASESUPPORT

Not supported

1.5.12 String compare

MDPROP_MDX_STRING_COMPOP

Only equal and not-equal

1.5.13 Descendants flags

MDPROP_MDX_DESCFLAGS

SELF, BEFORE and AFTER are supported in all combinations.

(MDPROPVAL_MD_SELF & MDPROPVAL_MD_BEFORE & MDPROPVAL_MD_AFTER)

1.5.14 Set functions

MDPROP_MDX_SET_FUNCTIONS

In addition to the standard functions, the following functions are supported:

TOPPERCENT, BOTTOMPERCENT, TOPSUM, BOTTOMSUM, DRILLDOWNMEMBER, DRILLDOWNLEVEL, DRILLDOWNMEMBERTOP (not recursive), DRILLDOWNMEMBERBOTTOM (not recursive), DRILLDOWNLEVELTOP, DRILLDOWNLEVELBOTTOM, DRILLUPMEMBER, DRILLUPLEVEL

1.5.15 Member functions

MDPROP_MDX_MEMBER_FUNCTIONS

No further members functions are supported in addition to the standard functions. For the list of the built-in functions you can refer to the documentation of the Microsoft data Access SDK.

1.5.16 Numeric functions

MDPROP_MDX_NUMERIC_FUNCTIONS

Not to be confused with Calculated members. In addition to the built-in functions, the following functions are also supported:

MEDIAN, VAR, STDDEV, RANK; AGGREGATE (only BW 2.0B), COVARIANCE, CORRELATION, LINREGSLOPE, LINREGVARIANCE, LIGREGR2, LINREGPOINT

1.5.17 Formulas

MDPROP_MDX_FORMULAS

WITH SET and CREATE SET are supported in the SESSION context.

(MDPROPVAL_MF_WITH_NAMEDSETS & MDPROPVAL_MF_CREATE_NAMEDSETS & MDPROPVAL_MF_SCOPE_SESSION)

1.5.18 Numeric functions in SET expressions

MDPROP_MDX_NONMEASURE_EXPRESSONS

Only expressions on the Measure dimension are supported.

(MDPROPVAL_MNE_MEASURESONLY)

2 BW Specific Extensions of OLE DB for OLAP

On the ODBO interface a couple of additions have been made to provide the use of variables (in the scope of ODBO termed in the following SAP VARIABLES), namely the new SCHEMA row set and a BW specific sap_variables-clause in the mdx syntax.

Variables are an extremely powerful concept in BW to parameterize queries for supporting a maximum of reuse and flexibility.

Variables act as placeholders for the following:

BW Terminology	**ODBO Terminology**
Characteristic values Hierarchy nodes	MEMBER_UNIQUE_NAME
Hierarchies	HIERARCHY_UNIQUE_NAME
Formula Variables	Numeric Values
Text variables	Strings

Variables may represent:

- a single value like a MEMBER_UNIQUE_NAME
- an interval of values
- a complex selection of values (i.e. a list of single values or intervals connected by OR)

The list of SAP VARIABLES and their properties for a specific QUERY_CUBE is available in the new SCHEMA rowset: SAP VARIABLES.

GUID	**Number of restrictions**	**Restriction columns**
MDSCHEMA_SAP_VARIABLES	1	CUBE_NAME

DEFINE_GUID(MDSCHEMA_SAP_VARIABLES, 0xcaff2c30, 0x30d, 0x11d3, 0x87, 0x1, 0x0, 0x10, 0x5a, 0x18, 0x50, 0x2e);

Column name	**Type indicator**	**Description**
CATALOG_NAME	DBTYPE_WSTR	The name of the catalog to which this cube belongs
SCHEMA_NAME	DBTYPE_WSTR	not supported
CUBE_NAME	DBTYPE_WSTR	The name of the query-cube to which this variable belongs
VARIABLE_NAME	DBTYPE_WSTR	Name of the variable. Since variable names may

		contain leading numbers they are always returned by the provider with delimiters
VARIABLE_CAPTION	DBTYPE_WSTR	A label or caption associated with the variable used primarily for display purposes
VARIABLE_UID	DBTYPE_GUID	variable GUID
VARIABLE_ORDINAL	DBTYPE_I2	Ordinal number of the variable, among the group of variables of a cube
VARIABLE_TYPE	DBTYPE_I2	The variable type. Can be one of the following values: • SAP_VAR_TYPE_MEMBER, which indicates that the variable is a placeholder for a selection of MEMBER_UNIQUE_NAMES • SAP_VAR_TYPE_HIERARCHY, which indicates that the variable is a placeholder for a HIERARCHY_UNIQUE_NAME • SAP_ VAR _TYPE_NUMERIC, which indicates that the variable is a placeholder for a numeric value in formulas
DATA_TYPE	DBTYPE_UI2	Data type of the variable values. Can be any of the types listed in the Appendix A of the *OLE DB Programmer's Reference*
CHARACTER_MAXIMUM_ LENGTH	DBTYPE_UI4	The maximum possible length of variable values if the data type is character
VARIABLE_PROCESSING_ TYPE	DBTYPE_UI2	Type of processing for the variable. Can be one of the following: • SAP_ VAR _PROC_TYPE_USER_INPUT; this indicates that values for this variable can be entered manually before query execution. In this version only variables of this type are returned on the ODBO interface
VARIABLE_SELECTION_TY PE	DBTYPE_UI2	Type of selection of the variable. Variables of type SAP_ VAR _TYPE_MEMBER allow selection not only of single members but also intervals of members or complex select options with just a single variable. The type of selection can be one of the following: • SAP_ VAR _SEL_TYPE_VALUE, which indicates that the variable is replaced with a single member. For variables of the type NUMERIC this is the only possible selection type • SAP_ VAR _SEL_TYPE_INTERVAL, which indicates that the variable is a placeholder for an interval of members ▪ SAP_ VAR _SEL_TYPE_COMPLEX, which indicates that the variable is a placeholder for a complex selection of members.
VARIABLE_ENTRY_TYPE	DBTYPE_UI2	Type that indicates whether a replacement of the

		variable is mandatory or optional: Input type is one of the following: • SAP_ VAR _INPUT_TYPE_OPTIONAL, which indicates that a specification of a value for this variable is optional • SAP_ VAR _INPUT_TYPE_MANDATORY, this indicates that a specification of this variable is mandatory. If no value for the variable is specified, the execution of the statement fails with an error • SAP_VAR_INPUT_TYPE_MANDATORY_NOT_INITIAL, this indicates that a specification of this variable is mandatory. If no value for the variable is specified, the execution of the statement fails with an error. Furthermore the initial value is not a valid entry
REFERENCE_DIMENSION	DBTYPE_WSTR	This column contains a DIMENSION_UNIQUE_NAME for parameter type SAP_ VAR _TYPE_HIERARCHY. Note that the dimension must not be contained in the query cube. A value for the hierarchy must be selected from this dimension
REFERENCE_HIERARCHY	DBTYPE_WSTR	This column contains a HIERARCHIE_UNIQUE_NAME for variable type SAP_ VAR _TYPE_MEMBER. Note that the hierarchy must not be contained in the query cube. If for hierarchy node variable the hierarchy itself is variable this column is equal to the REFERENCE_DIMENSION
DEFAULT_LOW	Variant	Contains a default value for the variable or is NULL. The actual data type of this column is as the data type indicated by the column DATA_TYPE
DEFAULT_LOW_CAP	DBTYPE_WSTR	A label or caption associated with the default member
DEFAULT_HIGH	Variant	Contains a default value for the variable or is NULL. This is only relevant for variables with selection type: • SAP_VAR _SEL_TYPE_INTERVAL • SAP_ VAR_SEL_TYPE_SELECTION
DEFAULT_HIGH_CAP	DBTYPE_WSTR	A label or caption associated with the default member
DESCRIPTION	DBTYPE_WSTR	A human readable description of the variable

To execute a mdx statement that selects data from a parameterized cube it is necessary to specify variable values in the mdx statement for all mandatory variables. The mdx syntax has been extended with a BW specific sap_variables-clause which contains basically a list of variable name – value pairs.

SAP BW extension of the MDX grammar for cube-variables

This section shows the extensions, which have been defined for supporting a variable value specification in an mdx select statement. Variable support consists of a new clause where BW variables and their values are listed.

```
<select_statement> ::= [WITH <formula_specification>]
                       SELECT [<axis_specification>[,<axis_specification>]…]
                       FROM [<cube_specification>]
                       WHERE [<slicer_specification>]
                       [<cell_props>]
                       [<sap_variables>]
;
<sap_variables> :
                        |SAP VARIABLES  <sap_variables_list>
;
<sap_variables_list> :    <sap_variable>
                       |<sap_variables_list> ',' <sap_variable>
;
<sap_variable> :        <variable_value_specification>
                       |<variable_value_list><variable_value_specification>
;
<variable_value_specification> :        <var_single_value_specification>
                        |<var_interval_value_specification>
;
<var_single_value_specification> : <variable_name> <sign> <variable_value>
;
<var_interval_value_specification> : <variable_name> <sign> <variable_value >':' <variable_value>
;
<sign> :                                INCLUDING | EXCLUDING
;
<variable_value> :      <member> | <unsigned_numeric_literal>
;
```

Examples:

Specifying a value for a single value variable:

select [DUYZ7E3E5GH2F0W4D7OGO6RKD].members on columns, non empty [ODB_CUST].members on rows from [ODBOSCEN1/MKTBRANCH] **SAP VARIABLES [ODBBRANC] INCLUDING [ODB_BRANC].[CHEM]**

Excluding a value for a single value variable:

select [DUYZ7E3E5GH2F0W4D7OGO6RKD].members on columns, non empty [ODB_CUST].members on rows from [ODBOSCEN1/MKTBRANCH] **SAP VARIABLES [ODBBRANC] EXCLUDING [ODB_BRANC].[CHEM]**

Specifying an interval of values for an interval variable:

select [2M19UOW3BTEMXNZMHSSSEP631].members on columns, non empty [0CALYEAR].members on rows from [ODBOSCEN1/MKTCUST] **SAP VARIABLES [ODBCUST] INCLUDING [ODB_CUST].[B01] : [ODB_CUST].[C05]**

Specifying a list of values and intervals for a selection variable:

select [4G3SFZOHUNDFDZN137S0IQ425].members on columns, non empty [ODB_BRANC].members on rows from [ODBOSCEN1/MKTPROD] **SAP VARIABLES [ODBPROD] INCLUDING [ODB_PROD].[CP01230111] [ODBPROD] INCLUDING [ODB_PROD].[LP10101190] [ODBPROD] INCLUDING [ODB_PROD].[PP98010102]:[ODB_PROD].[PR48666001]**

Constants

Use the following defines for the constants defined for the VARIABLES schema rowset

```
#define SAP_VAR_TYPE_MEMBER                      1
#define SAP _ VAR _TYPE_NUMERIC                  2
#define SAP_VAR_TYPE_HIERARCHY                   3

#define SAP _ VAR _PROC_TYPE_USER_INPUT          1

#define SAP _ VAR _SEL_TYPE_VALUE                1
#define SAP _ VAR _SEL_TYPE_INTERVAL             2
#define SAP _ VAR _SEL_TYPE_COMPLEX              3

#define SAP _ VAR _INPUT_TYPE_OPTIONAL                           0
#define SAP _ VAR _INPUT_TYPE_MANDATORY                          1
#define SAP _ VAR _INPUT_TYPE_MANDATORY_NOT_INITIAL              2
```

A typical scenario for usage of parameterized query cubes

OLE DB for OLAP clients typically start the creation of a report by browsing some of the DataSource Metadata, essentially the list of cubes. Let us suppose the end user now selects a cube, the client program can easily check whether there are any variables defined for this cube by inspecting the SAP_VARIABLES schema rowset.

The SAP_VARIABLES schema rowset requires a restriction on CUBE_NAME. If CUBE_NAME is not specified, this results in an error. If the cube does not have any variables, the provider returns an empty schema rowset.

In one of the next steps before executing an mdx statement for this parameterized cube, the client application probably wants to allow the user to select variable values.

Depending on the VARIABLE_TYPE the application will follow different strategies to get a list of possible variable values.

- SAP_VAR_TYPE_MEMBER. In this case the client application can take the HIERARCHY_UNIQUE_NAME from the column REFERENCE_HIERARCHY and use it for getting a list of members which might replace the variable . In case of an interval variable the client app may provide two input fields. One for the low value and one for the high value. The client app may take the default values to fill its input fields initially. In case of a selection variable the client app may provide a mechanism for entering a list of single values.
- Each Variable may provide a default value and a caption for each default value.

Before executing an mdx for a parameterized cube the client app has to assure that it has values for all variables with INPUT_TYPE “SAP_VAR_INPUT_TYPE_MNDTRY”. If this is the case the client app could create a mdx statement with the additional cube_variable clause and execute it as any other non parameterized query.

3 Installation of BW ODBO frontend components

.. and Prerequisites for connecting to the BW server through ODBO.

To access BW through OLE DB for OLAP the installation of a BW 2.0B front-end (alternative BW 2.0A front-end) is required. The following components should be installed on the desktop computer:

Mdrmsap.dll	the BW OLE DB provider itself
Mdrmdlg.dll	service dll to connect to the SAP server
Scerrlkp.dll	error handling dll
Mdxpars.dl	mdx parser dll
Librfc32.dll	SAP RFC library
Wdtlog.ocx	SAP RFC logon dialog
Saplogon.ini	SAP connection parameter file

The SAP BW OLE DB for OLAP provider is registered under the ProgID MDrmSAP.

Note: the SAP BW front-end setup covers only the components required by the SAP OLE DB for OLAP provider. Any components that live above the driver and are required by the client application must be installed by the 3rd party application itself. (In general the installation of the Microsoft OLE DB redistributables is necessary. If theses are not shipped with the specific client software you can find these components (**Microsoft Data Access Components 2.5 Redistribution Typical Setup)** at http://www.microsoft.com/data/download2.htm.

For Windows 95 it is furthermore necessary to install DCOM95 prior to the installation of the OLE DB redistributables. The corresponding executable can also be found on the BW presentation CD under \msredist\DCOM95.EXE.

Note also that the installation of these components requires administration rights on the computer.

The following parameters are mandatory for establishing a connection to the BW server through the ODBO driver:

Password	
UserId	
DataSource	name of the system as it appears in the ***saplogon*** dialog
SFC_CLIENT	the production client number of the BW server system
SFC_LANGUAGE	the logon language

A typical example for a provider string is as follows:

```
DataSource=J02;UserID=SCHROEDER;Password=MyPassword;Provider=MDrmSAP;SFC_CLIENT=005;SFC_LANGUAGE=DE
```

Installation Step- by- Step

Steps 1 to 3 are necessary only if you want to use the BW ODBO provider with a custom application that uses ADO MD and you have not installed the required components yet.

If you are using the ODBO provider with 3rd party applications like BusinessObjects, Arcplan inSight or Cognos Powerplay follow only the steps beginning with step 4. The client applications are responsible to install any OLE DB components which are necessary to run the application and which live above the SAP provider. The SAP installation ensures only the presence of msdadc.dll.

1. If you are installing on Windows 95 ensure that you have installed DCOM95.If this is not the case run at first the installation of DCOM95. The corresponding Microsoft redistributables can be found in the folder \msredist\DCOM95.EXE
2. If you want to use the ODBO provider with a custom application that uses ADO MD or other OLE DB components ensure that the necessary Microsoft components have been installed on your machine.If this is not the case install the Microsoft OLE DB redistributables from the file mdac_olp.exe. This executable is also available under \msredist. If you are not running a complete installation you need only the following components from the custom installation: ODBC components; OLE DB components; Microsoft ActiveX Data Objects; Microsoft ADO Multi-Dimensional; Microsoft Remote Data Services.
3. reboot your machine
4. run sapsetup.exe for the OLE DB provider from the BW presentation CD

3.1 Important Notes

- The BW provider supports named levels only as dummy levels, i.e. for each level number within a hierarchy a dummy level name is generated.
- It is very common in BW client systems that dimensions may have multiple hierarchies. Thus the provider may return multiple hierarchies for a dimension.
- The actually supported functionality is documented in the section Application of leveling rules.
- The BW OLE DB for OLAP provider supports user defined dimension properties. Thus user defined dimension properties can be specified with the DIMENSION PROPERTIES clause in an mdx statement and are returned in the corresponding axis rowsets. One limitation is currently that dimension properties are not returned with the MEMBERS schema rowset.
- There is a limited support of the concept of CATALOGs with the BW ODBO provider. There is support for a CATALOG schema rowset. And the CATALOG column is filled in all other schema rowsets. The CATALOG column returns the name of the InfoCube underlying a QUERY_CUBE.

3.2 Using the BW OLE DB for OLAP provider step by step

To enable third-party applications to query through the ODBO interface, the following steps must be followed:

1. Install the BW ODBO frontend components as described beforehand.
2. As an administrator, define a `Query_Cube` using Business Explorer Analyzer. Refer to the Business Explorer Analyzer online documentation for detailed procedures.
3. Select the characteristics dimension and drag and drop into either the column or row folder. This will be mapped to dimensions of the ODBO `Query_Cube`.
4. Define a single structure with the desired key figures and drag key figures from the InfoCube to this structure. The name of the structure will be mapped to the ODBO interface to members of the measures dimension. The members of the structure are the members of the measures dimension.
5. Prior to saving the query, you must set the Query Property dialog checkbox with “Release for OLE DB for OLAP”. You may access the Query Property dialog through the icon with a hand holding an envelope on the tool bar.
6. Save the `Query_Cube`. Be aware the name of `Query_Cube` which appears in the client application will be in the form “InfoCube Name/Query Name”.

7. The client application will request for you to logon to the BW server. Provide your system name, user ID, password and language. (You may also want to run the query with the BEx Analyzer for test purposes).
8. Follow the client application instructions to design your front-end presentation. The created `Query_Cube` should appear in the list of Cubes the application offers you.

4 FAQs

1. How is the BW ODBO interface related to the interface of the Business Explorer Analyzer?

   Both interfaces represent a relatively thin layer on top of the BW OLAP processor in which most of the processing is done. Thus the performance behavior will be very similar for both interfaces.
2. Does the concept of the QUERY_CUBE mean that even for a very simple mdx statement all data selected in the QUERY_CUBE is loaded into the Application servers memory?

   No. If you define a QUERY_CUBE with a large number of dimensions and measures, the BW OLAP processor will load only the amount of data into the application servers memory which is necessary to build a specific dataset, as specified on the ODBO interface with the mdx statement. The concept of QUERY_CUBEs does also not mean that any data of the underlying InfoCube is replicated. QUERY_CUBEs are populated only during the runtime of a query.
3. The BW ODBO provider does not support Calculated Members. Is there a corresponding feature available with BW?

   Since the BW ODBO provider gives access to QUERY_CUBEs the basic key figures of the underlying InfoCube can be easily extended with derived key figures which cover basically the same functionality as Calculated members. One advantage of this approach, is that there is a consistent definition of key figures on the BW server.
4. Can we use the functionality of Variables with the ODBO interface?

   QUERY_CUBEs may be parameterized with Variables. However Variables with user input can be processed with the ODBO interface only by some 3rd party applications. Chapter 3 (BW Specific Extensions of OLE DB for OLAP) describes a couple of additions that have been made to provide the use of variables.
5. What is ADO MD?

   ADO MD stands for ActiveX Data Objects MultiDimensional. ADO MD is a high level programming interface which gives access to a multidimensional data source through an easy to use object hierarchy. ADO MD can be used for example with VB or VB script.
6. Is there a SAP certification process for 3rd party client application vendors?

   To ensure a smooth operation of partner products with the BW ODBO implementation, SAP offers interested partners a certification of their product integration with BW. The certification will also ensure the rollout of important information to CSPs after the certification. Besides some consulting Partners with a signed certification contract will also have access to a BW test system, prior to the certification.

4.1 References

For a detailed technical description of OLE DB for OLAP, please refer to:

- Microsoft OLE DB for OLAP Programmer's Reference, available from the Microsoft web site at http://www.microsoft.com/data/oledb/olap

- BW Online Documentation

5 Definitions and Abbreviations

API - Application Programming Interface
A specification and a set of tools to allow programmatic communication with another system.

BW - Business Information Warehouse
A component of the SAP business framework, specifically designed to meet the data warehousing and reporting needs of R/3 customers.

BNF - Bacchus-Normal-Form
A system of notation used to decompose and describe a grammar. ODBO Microsoft's OLE DB for OLAP

Characteristic
A SAP BW dimension field. A key to facts.

InfoCube
An independent warehouse of information in the BW. Each InfoCube is comprised of one fact table, and one or many dimension tables.

Key Figure
A SAP BW fact, normally a quantitative field.

MDX Multidimensional Expression Language
The grammar used with the OLE DB protocols.

OLAP - Online Analytical Processing
In contrast to OLTP, on-line transaction processor. R/3 is both an OLTP and an OLAP. The BW is specifically an OLAP.

OLE - Object Link Enabling
A Microsoft proprietary set of protocols which specifies how desktop applications may communicate with and manipulate members of other applications.

OLE DB - OLE Database
One of the protocols specified as part of OLE. This protocol enables data consumer applications to access data provider applications.

Query_Cube
A source of datasets formed by applying a pre-built query on a BW InfoCube through the OLAP processor. The object which provides data in the BW OLAP API.

RFC - Remote Function Calls
A type of R/3 ABAP/4 programming unit. An RFC is a function which has been enabled for remote calling. It executes in a separate task space when called externally. The term RFC is used for both the calls and the functions.