# Building OLAP Catalog

Conf.dr. Mihaela MUNTEAN
Catedra de Informatică Economică, A.S.E. Bucureşti

*The Oracle OLAP option of Oracle Database 10g Enterprise Edition provides advanced analytic features to help us to summarize, analyze and calculate data faster than standard SQL. In this article I discussed about the CWM2, a set of PL/SQL packages which are using to create the OLAP Catalog metadata for an analytic workspace. BI Beans applications query OLAP Catalog metadata.*
***Keywords:*** *star schema, dimensions, cubes, hierarchies, levels, attributes, OLAP catalog, SQL.*

Catalogul de metadate OLAP include:
- *un set de tabele* (*metadata model tables*) ce sunt utilizate pentru a defini componentele catalogului OLAP (cuburi, măsuri, dimensiuni, ierarhii, nivele, atribute). Aceste componente sunt mapate la sursele de date (coloanele unei tabele sau tabele virtuale dintr-o schemă stea/fulg de zăpadă).
- *o colecție de pachete PL/SQL* pentru crearea şi editarea metadatelor OLAP. Aceste pachete conțin proceduri pentru actualizarea tabelelor asociate modelului multidimensional logic (figura 1);
- *un set de viziuni (tabele virtuale)* ce oferă informații despre metadatele înregistrate în tabelele asociate modelului multidimensional logic.

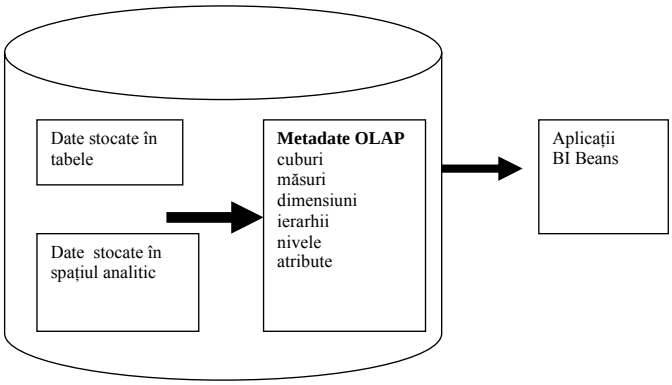


**Fig. 1.** Modelul multidimensional logic utilizat de catalogul de metadate OLAP

Pentru a crea catalogul de metadate se pot utiliza următoarele instrumente: *Oracle Enterprise Manager*, *Oracle Warehouse Builder, Analytical Workspace Manager* şi *colecția de pachete PL/SQL -CWM2.*

*Colecția de pachete CWM2* este utilizată pentru a defini catalogul de metadate OLAP utilizat de un spațiu analitic (figura 2). Pentru fiecare componentă a modelului de metadate există un pachet PL/SQL separat. Fiecare componentă a catalogului este unic identificată de proprietarul ei şi de numele ei. Atunci când se creează o componentă a catalogului de metadate, de fapt se adaugă un tuplu în tabela asociată acelui tip de componentă.

Pentru exemplificare, se va utiliza instrumentul *SQL*Plus* şi se face conexiunea la baza de date ca utilizator *SYSTEM.*

- Se va crea un spațiu tabel *GLOBALX* şi un spațiu tabel temporar *GLOBTEMPX*

```
SQL> create tablespace globalx logging
datafile 'd:\oracle\product\10.1.0\oradata\oracle10\globalx.dbf'
size 5M reuse autoextend on next 5M maxsize unlimited
extent management local segment space management auto;
SQL> create temporary tablespace glotempx
tempfile 'd:\oracle\product\10.1.0\oradata\oracle10\glotempx.tmp'
size 5M reuse autoextend on next 5M maxsize unlimited;
```

➢ Se va crea utilizatorul *GLOBALX*, proprietarul surselor de date. Acest utilizator primeşte o serie de privilegii: *CONNECT, OLAP_USER, CREATE ANY TYPE, CREATE ANY DIRECTORY*. De asemenea, se va crea un pseudonim *GX* pentru un director al sistemului de fişiere al serverului bazei de date şi se vor acorda drepturi tuturor utilizatorilor pe acest director.

```
SQL> create user globalx identified by "globalx" default tablespace globalx
temporary tablespace glotempx quota unlimited on globalx
quota unlimited on glotempx account unlock;
SQL> grant connect to globalx;
SQL> grant olap_user to globalx;
SQL> grant create any type to globalx;
SQL> grant create any directory to globalx;
SQL> create or replace directory gx as 'd:\oracle\product\10.1.0\oradata\oracle10';
SQL> grant all on directory gx to public;
```

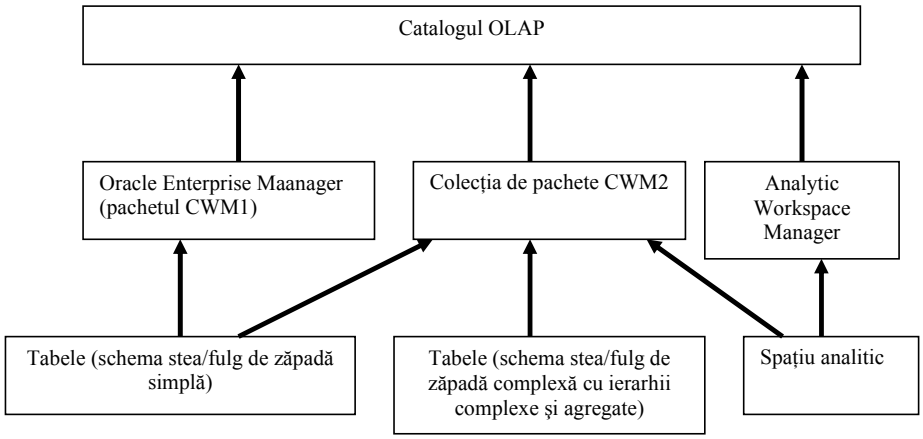


**Fig. 2.** Utilizarea colecției de pachete CWM2

➢ Utilizatorul *GLOBALX* va crea o schemă stea utilizată ca sursă de date cu următoarele tabele de dimensiuni (cheile primare sunt subliniate, pentru fiecare cheie externă este prezentată tabela referită):

```
Canale (canal_id number(5), canal_dsc varchar2(15), total_canal_id number(5), to-
tal_canal_dsc varchar2(15))
Clienti (client_id number(5), client_dsc varchar2(30), oras_id number(5), oras_dsc
varchar2(30), judet_id number(5), judet_dsc varchar2(15), total_clienti_id number(5),
total_clienti_dsc varchar2(15))
Produse (produs_id number(5), produs_dsc varchar2(31), categorie_id number(5), catego-
rie_dsc varchar2(20), total_produs_id number(5), total_produs_dsc varchar2(15))
Timp (luna_id number(5), luna_dsc varchar2(10), trimestru_id number(5),
trimestru_dsc varchar2(5), an_id number(5), an_dsc varchar2(5), luna_nrzile number(5),
trimestru_nrzile number(5), an_nrzile number(5), data_sfarsit_luna date, da-
ta_sfarsit_trimestru date, data_sfarsit_an date)
```

Schema stea include două tabele de fapte ce conțin informații despre o serie de indicatori (preț unitar, cost unitar, cantitatea vândută) utilizați în analiza vânzărilor unei firme

```
Pret_cost (produs_id number(5):Produse, luna_id number(5) :Timp, pret_unitar number,
cost_unitar number)
Unit (canal_id number(5): Canale, produs_id number(5): Produse, client_id number(5) :
Clienti, luna_id number(5) : Timp, cantitatea number)
```

➢ Se va crea un spațiu tabel *GLOBAL* şi un spațiu tabel temporar *OLAPTEMP*. Spațiul analitic se va crea într-un spațiu tabel diferit de cel în care se găsesc sursele de date.

```
SQL>   create tablespace global logging
datafile 'd:\oracle\product\10.1.0\oradata\oracle10\global.dbf'
size 5M reuse autoextend on next 5M maxsize unlimited
extent management local segment space management auto;
SQL>   create temporary tablespace olaptemp
tempfile 'd:\oracle\product\10.1.0\oradata\oracle10\olaptemp.tmp'
size 5M reuse autoextend on next 5M maxsize unlimited;
```

➢ Se va crea un utilizator *utilizator_aw*

```
SQL> create user utilizator_aw profile default identified by utilizator_aw
default tablespace global temporary tablespace olaptemp
quota unlimited on global quota unlimited on olaptemp account unlock;
```

➢ Utilizatorul *utilizator_aw* va primi dreptul de a se conecta la baza de date, va fi ataşat rolului *OLAP_USER* (va fi proprietarul spaţiului analitic creat), va primi dreptul de a crea directoare şi tipuri de obiecte. Acest utilizator va putea crea un spaţiu de lucru analitic cu instrumentul *Analytic Workspace Manager* utilizând catalogul de metadate generat cu colecţia de pachete CWM2.

```
SQL> grant connect to utilizator_aw;
SQL> grant olap_user to utilizator_aw;
SQL> grant create any type to utilizator_aw;
SQL> grant create any directory to utilizator_aw;
```

➢ Utilizatorul *utilizator_aw* va primi şi dreptul de *SELECT* pe tabelele sursă (schemă stea) care se găsesc în spaţiul tabel *GLOBALX*.

```
SQL> grant select on globalx.canale to utilizator_aw;
SQL> grant select on globalx.produse to utilizator_aw;
SQL> grant select on globalx.clienti to utilizator_aw;
SQL> grant select on globalx.timp to utilizator_aw;
SQL> grant select on globalx.unit to utilizator_aw;
SQL> grant select on globalx.pret_cost to utilizator_aw;
```

Pentru a crea catalogul de metadate OLAP se parcurg următorii paşi (conexiunea la baza de date se face ca utilizator g*lobalx*):

1. Se vor crea dimensiunile logice utilizând următoarele proceduri:

-*CWM2_OLAP_DIMENSION. CREATE_DIMENSION* pentru a crea o dimensiune;

-*CWM2_OLAP_DIMENSION_ATTRIBUTE. CREATE_DIMENSION_ATTRIBUTE* pentru a crea atributele unei dimensiuni. Se vor defini atribute pentru a descrie în detaliu dimensiunea (atributele *long description şi short description);*

-*CWM2_OLAP_HIERARCHY* pentru a defini relaţii ierarhice între nivele unei dimensiuni;

-*CWM2_OLAP_LEVEL.CREATE_LEVEL* pentru a crea niveluri ;

-*CWM2_OLAP_LEVEL_ATTRIBUTE.CREATE_LEVEL_ATTRIBUTE* pentru a crea atribute ale nivelurilor;

-*CWM2_OLAP_LEVEL.ADD_LEVEL_TO_HIERARCHY* pentru a asocia nivelurile la ierarhii;

-*CWM2_OLAP_TABLE_MAP* pentru a mapa nivelele şi atributelor lor la coloanele corespunzătoare din tabelele de dimensiuni ;

De exemplu, se va crea dimensiunea *CANALE* ce utilizează ca sursă tabela de dimensiuni *Canale*. Această dimensiune are o singură ierarhie *CANAL_ROLLUP* formată din două nivele *TOTAL_CANAL* şi *CANAL*. Se vor executa următoarele proceduri:

```
execute cwm2_olap_dimension.drop_dimension('GLOBALX', 'CANALE');
execute cwm2_olap_dimension.create_dimension('GLOBALX',
 'CANALE', 'Canale', 'CANALE', 'Canale', 'Canale', NULL);
execute cwm2_olap_dimension_attribute.create_dimension_attribute_2
('GLOBALX', 'CANALE', 'Long Description', 'Long Description', 'Long Description',
'Long Description', 1);
execute cwm2_olap_dimension_attribute.create_dimension_attribute_2('GLOBALX', 'CANA-
LE','Short Description', 'Short Description', 'Short Description', 'Short
Description', 1);
execute cwm2_olap_hierarchy.create_hierarchy
('GLOBALX', 'CANALE','CANAL_ROLLUP', 'Canal Rollup', 'Canal Rollup','Canal
Rollup','UNSOLVED LEVEL-BASED');
execute cwm2_olap_dimension.set_default_display_hierarchy('GLOBALX', 'CANALE', 'CA-
NAL_ROLLUP');
execute cwm2_olap_level.create_level('GLOBALX', 'CANALE', 'TOTAL_CANAL', 'Total Ca-
nal', 'Total Canal','Total Canal','Total Canal');
execute cwm2_olap_level.create_level
('GLOBALX', 'CANALE', 'CANAL', 'Canal','Canal','Canal','Canal');
execute cwm2_olap_level.add_level_to_hierarchy
('GLOBALX', 'CANALE', 'CANAL_ROLLUP','TOTAL_CANAL', NULL);
execute cwm2_olap_level.add_level_to_hierarchy
('GLOBALX', 'CANALE', 'CANAL_ROLLUP','CANAL', 'TOTAL_CANAL');
execute cwm2_olap_level_attribute.create_level_attribute_2('GLOBALX', 'CANALE', 'Long
Description', 'TOTAL_CANAL', 'Long Description','Long Description', 'Long
Description', 'Long Description', 1);
execute cwm2_olap_level_attribute.create_level_attribute_2('GLOBALX', 'CANALE', 'Long
Description', 'CANAL', 'Long Description','Long Description','Long Description', 'Long
Description', 1);
execute cwm2_olap_level_attribute.create_level_attribute_2
('GLOBALX', 'CANALE','Short Description', 'TOTAL_CANAL', 'Short Description','Short
Description','Short Description', 'Short Description', 1);
execute cwm2_olap_level_attribute.create_level_attribute_2
```

```
('GLOBALX', 'CANALE','Short Description', 'CANAL', 'Short Description','Short
Description','Short Description', 'Short Description', 1);
execute cwm2_olap_table_map.map_dimtbl_hierlevel
('GLOBALX', 'CANALE', 'CANAL_ROLLUP', 'TOTAL_CANAL', 'GLOBALX', 'CANALE', 'TO-
TAL_CANAL_ID', null);
execute cwm2_olap_table_map.map_dimtbl_hierlevel
('GLOBALX', 'CANALE', 'CANAL_ROLLUP', 'CANAL', 'GLOBALX', 'CANALE', 'CANAL_ID', 'TO-
TAL_CANAL_ID');
execute cwm2_olap_table_map.map_dimtbl_hierlevelattr('GLOBALX', 'CANALE', 'Long
Description', 'CANAL_ROLLUP', 'CANAL','Long Description','GLOBALX', 'CANALE', 'CA-
NAL_DSC');
execute cwm2_olap_table_map.map_dimtbl_hierlevelattr('GLOBALX', 'CANALE', 'Long
Description', 'CANAL_ROLLUP', 'TOTAL_CANAL','Long Description','GLOBALX', 'CANALE',
'TOTAL_CANAL_DSC');
execute cwm2_olap_table_map.map_dimtbl_hierlevelattr('GLOBALX', 'CANALE', 'Short
Description', 'CANAL_ROLLUP', 'CANAL','Short Description','GLOBALX', 'CANALE', 'CA-
NAL_DSC');
execute cwm2_olap_table_map.map_dimtbl_hierlevelattr
('GLOBALX', 'CANALE','Short Description', 'CANAL_ROLLUP',
'TOTAL_CANAL','Short Description','GLOBALX', 'CANALE', 'TOTAL_CANAL_DSC');
execute cwm2_olap_validate.validate_dimension('GLOBALX', 'CANALE');
```

2. Se vor crea cuburile logice şi se vor specifica dimensiunile cuburilor. Fiecare cub trebuie să aibă cel puțin o dimensiune şi cel puțin o măsură. Se vor utiliza următoarele proceduri:

-*CWM2_OLAP_CUBE.CREATE_CUBE* pentru a crea cubul;

-*CWM2_OLAP_CUBE.ADD_DIMENSION_TO_CUBE* pentru a identifica dimensiunile cubului;

-*CWM2_OLAP_MEASURE.CREATE_MEASURE* pentru a crea măsurile cubului;

-*CWM2_OLAP_TABLE_MAP* pentru a mapa măsurile cubului la coloanele corespunzătoare din tabela de fapte şi coloanele cheie externă din tabela de fapte la coloanele cheie primară din tabelele de dimensiuni.

De exemplu, se va crea cubul *UNIT_CUB* format din patru dimensiuni *Canale, Clienti, Produse* şi *Timp.* Cubul are o singură măsură *Cantitatea.* Se vor executa următoarele proceduri:

```
execute cwm2_olap_cube.drop_cube('GLOBALX', 'UNIT_CUB');
execute cwm2_olap_cube.create_cube
('GLOBALX', 'UNIT_CUB', 'UNIT CUB', 'UNIT CUB','UNIT CUB');
execute cwm2_olap_cube.add_dimension_to_cube
('GLOBALX', 'UNIT_CUB', 'GLOBALX', 'CANALE');
execute cwm2_olap_cube.add_dimension_to_cube
('GLOBALX', 'UNIT_CUB', 'GLOBALX', 'CLIENTI');
execute cwm2_olap_cube.add_dimension_to_cube
('GLOBALX', 'UNIT_CUB', 'GLOBALX', 'PRODUSE');
execute cwm2_olap_cube.add_dimension_to_cube
('GLOBALX', 'UNIT_CUB', 'GLOBALX', 'TIMP');
execute cwm2_olap_measure.create_measure
('GLOBALX', 'UNIT_CUB', 'CANTITATE', 'CANTITATE', 'cantitatea vanduta', 'cantitatea
vanduta');
execute cwm2_olap_table_map.map_facttbl_levelkey
('GLOBALX', 'UNIT_CUB', 'GLOBALX', 'UNIT',
'LOWESTLEVEL','DIM:GLOBALX.CANALE/HIER:CANAL_ROLLUP/LVL:CANAL/COL:CANAL_ID;DIM:GLOBALX
.CLIENTI/HIER:CLIENT_ROLLUP/LVL:CLIENT/COL:CLIENT_ID;DIM:GLOBALX.PRODUSE/HIER:PRODUS_R
OLLUP/LVL:PRODUS/COL:PRODUS_ID;DIM:GLOBALX.TIMP/HIER:CALENDAR/LVL:LUNA/COL:LUNA_ID;');
execute cwm2_olap_table_map.map_facttbl_measure
('GLOBALX', 'UNIT_CUB', 'CANTITATE', 'GLOBALX', 'UNIT',
'CANTITA-
TEA','DIM:GLOBALX.CANALE/HIER:CANAL_ROLLUP/LVL:CANAL/COL:CANAL_ID;DIM:GLOBALX.CLIENTI/
HIER:CLIENT_ROLLUP/LVL:CLIENT/COL:CLIENT_ID;DIM:GLOBALX.PRODUSE/HIER:PRODUS_ROLLUP/LVL
:PRODUS/COL:PRODUS_ID;DIM:GLOBALX.TIMP/HIER:CALENDAR/LVL:LUNA/COL:LUNA_ID;');
execute cwm2_olap_validate.validate_cube('GLOBALX', 'UNIT_CUB');
```

Procedura *MAP_FACTTBL_LEVELKEY* are ca parametrii de intrare: *numele cubului, numele tabelei de fapte asociată, un şir de mapare şi un indicator ce specifică cum sunt stocate datele în tabela de fapte.* Valoarea acestui indicator poate fi, de exemplu, *LOWESTLEVEL* ceea ce presupune că există o singură tabelă de fapte în care se vor stoca datele pentru toate măsurile unui cub. Dacă una din dimensiunile cubului are mai multe ierarhii, aceste ierarhii trebuie să aibă acelaşi nivel de granulație.

3. *Validarea metadatelor OLAP***.** Testul de validare a metadatelor utilizează pachetul *CWM2_OLAP_VALIDATE*. Procedurile de validare verifică integritatea structurală a

metadatelor şi dacă sunt corect mapate la coloanele corespunzătoare din tabelele de dimensiuni şi tabelele de fapte.

4. Nici una din procedurile *CWM2* nu include un COMMIT ceea ce presupune actualizarea tabelelor asociate catalogului de metadate OLAP.

Instrumentul *Analytical Workspace Manager* permite vizualizarea catalogului de metadate OLAP creat, dacă se selectează opţiunea *OLAP Catalog View* din meniul *View* (figura 3).

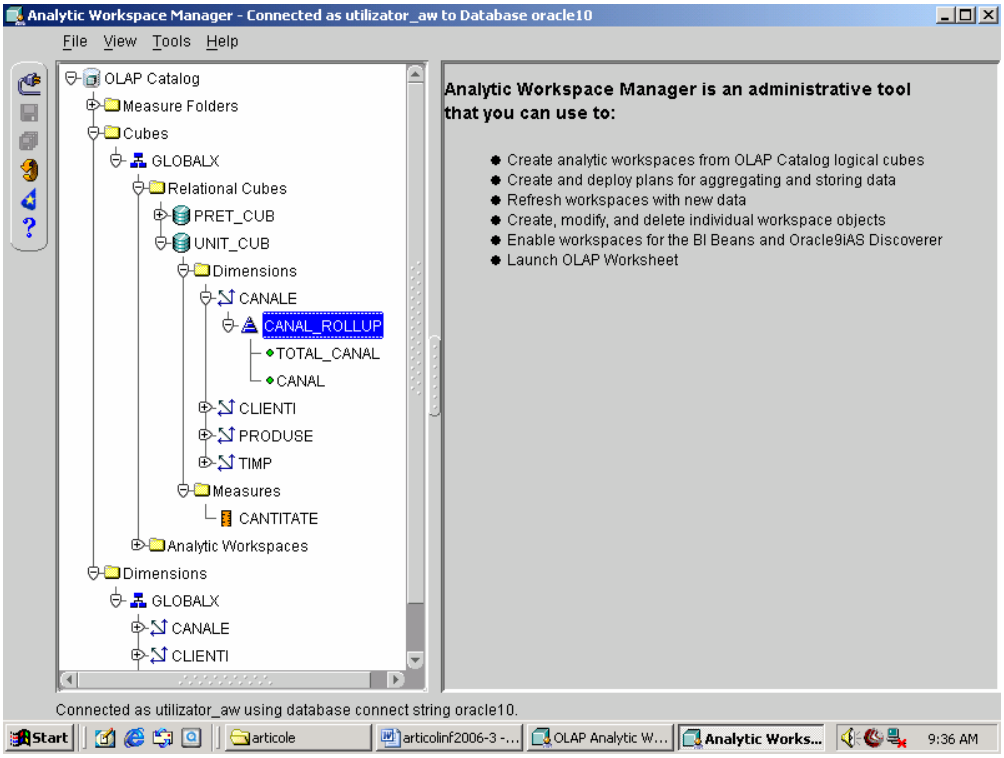


**Fig. 3.** Vizualizarea catalogului de metadate OLAP

**Bibliografie**

[1] Mihaela Muntean**,** *Iniţiere în tehnologia OLAP. Teorie şi practică***,** editura ASE, Bucureşti, 2004

[***] Oracle Corporation, *Oracle OLAP Application Developer's Guide, 2004*

[***] Oracle Corporation, *Oracle OLAP Reference 10g Release 1, 2004*