



Hurtownie danych - przegląd technologii

Robert Wrembel
Politechnika Poznańska
Instytut Informatyki
Robert.Wrembel@cs.put.poznan.pl
www.cs.put.poznan.pl/rwrembel



Modelowanie wymiarów

Wymiar

- struktura umożliwiająca grupowanie danych z tabeli faktów
- implementowana jako obiekt bazy danych DIMENSION
- wykorzystanie DIMENSION
 - zaawansowane przepisywanie zapytań (ang. query rewrite)
 - wspomaganie projektowania zbioru zmaterializowanych perspektyw przez Summary Advisor
 - odświeżanie przyrostowe zmaterializowanych perspektyw zawierających agregaty

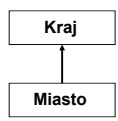
Robert Wrembel
Politechnika Poznańska, Instytut Informatyki

2



CREATE DIMENSION (1)

znormalizowane tabele wymiarów



Nazwa	NULL?	Typ
K_NAZWA	NOT NULL	VARCHAR2(30)
L_MIESZK	NOT NULL	NUMBER(10)

Nazwa	NULL?	Typ
M_NAZWA	NOT NULL	VARCHAR2(30)
L_MIESZK	NOT NULL	NUMBER(10)
K_NAZWA		VARCHAR2(30)

```
create dimension polozenieGEO
level L_KRAJ is kraj.k_nazwa
level L_MIASTO is miasto.m_nazwa
hierarchy h_kraj_miasto
(L_MIASTO child of L_KRAJ
join key miasto.k_nazwa references L_KRAJ);
```

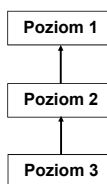
Robert Wrembel
Politechnika Poznańska, Instytut Informatyki

3



CREATE DIMENSION (2)

znormalizowane tabele wymiarów



```
create dimension D
level p3 is tabela3.PK
level p2 is tabela2.PK
level p1 is tabela1.PK
hierarchy H
(p3 child of
p2 child of
p1
join key tabela3.FK references p2
join key tabela2.FK references p1);
```

Robert Wrembel
Politechnika Poznańska, Instytut Informatyki

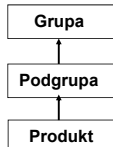
4



CREATE DIMENSION (3)

Name	Null?	Type
ID_PROD	NOT NULL	NUMBER(6)
NAZWA	NOT NULL	VARCHAR2(30)
CENA_DET	NOT NULL	NUMBER(6,2)
CENA_HURT	NOT NULL	NUMBER(6,2)
PODGRUPA	NOT NULL	VARCHAR2(30)
PODGR_INF		VARCHAR2(50)
GRUPA	NOT NULL	VARCHAR2(30)
GRUPA_INF		VARCHAR2(50)

zdenormalizowane tabele wymiarów



```
create dimension dim_produkty
level produkt is produkty.produkt_id
level podgrupa is produkty.podgrupa
level grupa is produkty.grupa
hierarchy hier_produkty
(produkt child of podgrupa
podgrupa child of grupa);
```

Robert Wrembel
Politechnika Poznańska, Instytut Informatyki

5



Uwagi

- W ramach tej samej hierarchii związek 1:n między poziomem nadrzędnym i podrzędnym
- Dla znormalizowanych tabel wymiarów definiować klucze obce
- Cykle w hierarchii są niedozwolone

Robert Wrembel
Politechnika Poznańska, Instytut Informatyki

6



Zależności funkcyjne (1)

- ⇒ zdenormalizowane tabele wymiarów
- ⇒ zależności funkcyjne między atrybutami wymiarów
 - $id_prod \rightarrow \{nazwa, cena_det, cena_hurt\}$
 - $podgrupa \rightarrow \{podgr_inf\}$
 - $grupa \rightarrow \{grupa_inf\}$

Name	Null?	Type
ID_PROD	NOT NULL	NUMBER(6)
NAZWA	NOT NULL	VARCHAR2(30)
CENA_DET	NOT NULL	NUMBER(6,2)
CENA_HURT	NOT NULL	NUMBER(6,2)
PODGRUPA	NOT NULL	VARCHAR2(30)
PODGR_INF		VARCHAR2(50)
GRUPA	NOT NULL	VARCHAR2(30)
GRUPA_INF		VARCHAR2(50)



Zależności funkcyjne (2)

- ⇒ wykorzystanie - przepisywanie zapytań

```
create table sprzedaz
(id_sprzed number(6) not null primary key,
id_prod number(6) not null references produkty(id_prod),
id_sklep number(6) not null references sklepy(id_sklep),
data date not null,
ilosc number(5) not null,
kwota_calk number(9,2) not null);
```

$id_prod \rightarrow nazwa$

```
create materialized view mv_sprzedaz
...
as select id_prod, id_sklep, sum(kwota_calk)
from sprzedaz group by id_prod, id_sklep;
```

```
select pr.nazwa, sk.id_sklep, sum(sp.kwota_calk)
from produkty pr, sklepy sk, sprzedaz sp
where sp.id_prod=pr.id_prod
and sp.id_sklep=sk.id_sklep
group by pr.nazwa, sk.id_sklep;
```



Zależności funkcyjne (3)

- ⇒ jeżeli zdefiniowano
 - $id_prod \rightarrow \{nazwa, ...\}$
- ⇒ wyniki poniższego zapytania można wyznaczyć na podstawie **mv_sprzedaz**

```
create materialized view mv_sprzedaz
...
as select id_prod, id_sklep, sum(kwota_calk)
from sprzedaz group by id_prod, id_sklep;
```

```
select pr.nazwa, sk.id_sklep, sum(sp.kwota_calk)
from produkty pr, sklepy sk, sprzedaz sp
where sp.id_prod=pr.id_prod
and sp.id_sklep=sk.id_sklep
group by pr.nazwa, sk.id_sklep;
```



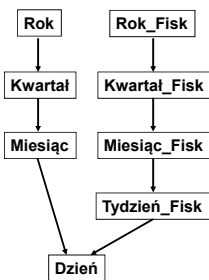
Zależności funkcyjne (4)

- ⇒ definiowanie

```
create dimension dim_produkty
level produkt is produkty.id_prod
level podgrupa is produkty.podgrupa
level grupa is produkty.grupa
hierachy hier_produkty
(produkt child of
podgrupa child of
grupa)
attribute produkt determines
(nazwa, cena_det, cena_hurt)
attribute podgrupa determines
(podgr_inf)
attribute grupa determines
(grupa_inf);
```



Wiele hierarchii

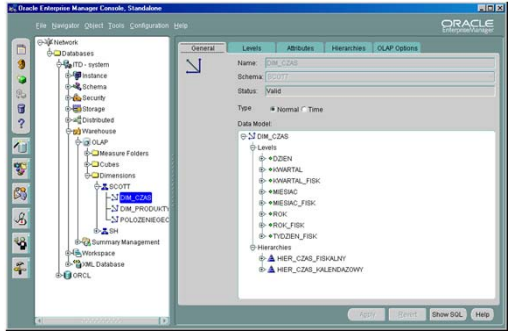


```
create dimension dim_czas
level dzien is czas.id_dzien
level miesiac is czas.miesiac
level kwartal is czas.kwartal
level rok is czas.rok
level tydzień_fisk is czas.tydzien_fisk
level miesiac_fisk is czas.miesiac_fisk
level kwartal_fisk is czas.kwartal_fisk
level rok_fisk is czas.rok_fisk
hierachy hier_czas_kalendazowy
(dzien child of
miesiac child of
kwartal child of
rok)
hierachy hier_czas_fiskalny
(dzien child of
tydzień_fisk child of
miesiac_fisk child of
kwartal_fisk child of
rok_fisk);
```



Słownik bazy danych


- ⇒ USER_DIMENSIONS
- ⇒ USER_DIM_ATTRIBUTES
- ⇒ USER_DIM_CHILD_OF
- ⇒ USER_DIM_HIERARCHIES
- ⇒ USER_DIM_JOIN_KEY
- ⇒ USER_DIM_LEVELS
- ⇒ USER_DIM_LEVEL_KEY
- ⇒ pakiet DEMO_DIM
 - PRINT_DIM('nazwa wymiaru')
 - PRINT_ALLDIMS
 - utworzyć pakiet w schemacie każdego użytkownika (plik %ORACLE_HOME%\rdms\demo\smdim.sql)



Enterprise Manager

Robert Wrembel
Politechnika Poznańska, Instytut Informatyki

13



VALIDATE DIMENSION (1)

- Sprawdzenie poprawności struktury wymiaru określonej w poleceniu **CREATE DIMENSION**
 - sprawdzenie hierarchii wymiarów i zależności funkcyjnych
- Pakiet **DBMS_OLAP.VALIDATE_DIMENSION**

Column	Data Type
ID_PROD	NOT NULL NUMBER(6)
NAZWA	NOT NULL VARCHAR2(30)
CENA_DET	NOT NULL NUMBER(6,2)
CENA_HURT	NOT NULL NUMBER(6,2)
PODGRUPA	VARCHAR2(30)
PODGR_INF	VARCHAR2(50)
GRUPA	VARCHAR2(30)
GRUPA_INF	VARCHAR2(50)


```
create dimension dim_produkty
level produkt is produkty.id_prod
level podgrupa is produkty.podgrupa
level grupa is produkty.grupa
hierachy hier_produkty
(produkt child of
podgrupa child of
grupa);
```

```
insert into produkty values (3,'proszek Persil',35,25,null,null(null,null));
```

```
execute dbms_olap.validate_dimension('dim_produkty', 'SCOTT', -
FALSE, TRUE, :idv)
```

Robert Wrembel
Politechnika Poznańska, Instytut Informatyki

14



VALIDATE DIMENSION (2)

- Wykorzystanie **DBMS_OLAP.VALIDATE_DIMENSION**

```
dbms_olap.validate_dimension ('wymiar', 'uzytkownik', -
TRUE|FALSE, TRUE|FALSE, id_walidacji)
```


TRUE - sprawdzane tylko nowe rekordy
TRUE - sprawdzane czy wszystkie klucze obce wiążące wymiary przyjmują wartości NOT NULL

tworzy identyfikator i przypisuje go do zmiennej idV

```
variable idV number;
execute dbms_olap.create_id(:idV)
execute dbms_olap.validate_dimension('DIM_CZAS', 'SCOTT', -
FALSE, TRUE, :idV)
```

Robert Wrembel
Politechnika Poznańska, Instytut Informatyki

15



VALIDATE DIMENSION (3)

- uzyskanie informacji o walidacji

```
select table_name, dimension_name, relationship, bad_rowid
from system.mview_exceptions;
```

TABLE_NAME	DIMENSION_NAME	RELATIONSHIP	BAD_ROWID
PRODUKTY	DIM_PRODUKTY	FOREIGN KEY	AAA1u1AABAAAMmqAAC
PRODUKTY	DIM_PRODUKTY	NOT NULL	AAA1u1AABAAAMmqAAC


idV pojawia się w kolumnie RUNID tabeli MVIEW_EXCEPTIONS

- uzyskanie informacji o niepoprawnych rekordach tabeli źródłowej

```
select * from produkty where rowid in
(select bad_rowid from system.mview_exceptions);
```

Robert Wrembel
Politechnika Poznańska, Instytut Informatyki

16




VALIDATE DIMENSION (4)

- usunięcie wyników walidacji

```
execute dbms_olap.purge_results(idV)
execute dbms_olap.purge_results(504)
```

Robert Wrembel
Politechnika Poznańska, Instytut Informatyki

17



VALIDATE DIMENSION (5)

- dla Oracle10g

```
EXECUTE DBMS_DIMENSION.DESCRIBE_DIMENSION('produkty')
EXECUTE DBMS_DIMENSION.VALIDATE_DIMENSION('produkty',
FALSE, TRUE, 'id walidacji');
```

- wyniki w tabeli **DIMENSION_EXCEPTIONS** tworzonej w schemacie użytkownika walidującego
 - skrypt UTLDIM.SQL

Robert Wrembel
Politechnika Poznańska, Instytut Informatyki

18



Modyfikowanie wymiarów (1)

```
SQL> exec demo_dim.print_dim('DIM_PRODUKTY')
DIMENSION SCOTT.DIM_PRODUKTY
LEVEL GRUPA IS SCOTT.PRODUKTY.GRUPA
LEVEL PODGRUPA IS SCOTT.PRODUKTY.PODGRUPA
LEVEL PRODUKT IS SCOTT.PRODUKTY.ID_PROD
HIERARCHY HIER_PRODUKTY (
  PRODUKT
  CHILD OF PODGRUPA
  CHILD OF GRUPA
)
ATTRIBUTE GRUPA DETERMINES SCOTT.PRODUKTY.GRUPA_INF
ATTRIBUTE PODGRUPA DETERMINES SCOTT.PRODUKTY.PODGR_INF
ATTRIBUTE PRODUKT DETERMINES SCOTT.PRODUKTY.CENA_DET
ATTRIBUTE PRODUKT DETERMINES SCOTT.PRODUKTY.CENA_HURT
ATTRIBUTE PRODUKT DETERMINES SCOTT.PRODUKTY.NAZWA
```

```
alter dimension dim_produkty drop attribute produkt;
```



Modyfikowanie wymiarów (2)

```
SQL> exec demo_dim.print_dim('DIM_PRODUKTY')
DIMENSION SCOTT.DIM_PRODUKTY
LEVEL GRUPA IS SCOTT.PRODUKTY.GRUPA
LEVEL PODGRUPA IS SCOTT.PRODUKTY.PODGRUPA
LEVEL PRODUKT IS SCOTT.PRODUKTY.ID_PROD
HIERARCHY HIER_PRODUKTY (
  PRODUKT
  CHILD OF PODGRUPA
  CHILD OF GRUPA
)
```

```
alter dimension dim_produkty
drop hierarchy hier_produkty;
```

```
alter dimension dim_produkty drop level produkt;
```



Modyfikowanie wymiarów (3)

```
alter dimension dim_produkty add level
produkt is produkty.id_prod;
```

```
alter dimension dim_produkty add hierarchy
hier_produkty
(produkt child of
 podgrupa child of
 grupa);
```

```
alter dimension dim_produkty add attribute
podgrupa determines produkty.grupa_inf;
```

```
alter dimension dim_produkty compile;
```

```
drop dimension dim_produkty;
```