

**Ekstrakcja, transformacja i
ładowanie danych (ETL).
Wspomaganie operacji ETL przez
Oracle Warehouse Builder**

Plan rozdziału

- Wprowadzenie do procesów ETL
- Ekstrakcja danych
- Transformacja danych
- Ładowanie danych
- ETL w bazie danych Oracle
- ETL z wykorzystaniem Oracle Warehouse Builder

ETL

- Podstawowy czynnik wpływający na wynik wdrożenia hurtowni danych
- Może pochłoniąć ponad 70% czasu projektu
- **Ekstrakcja**
 - pobieranie danych z operacyjnych baz danych i innych systemów
- **Transformacja**
 - integracja, weryfikacja, walidacja, czyszczenie, etykietowanie danych i tworzenie spójnego obrazu danych, poprawianie błędnych danych
- **Ładowanie**
 - proces przesyłania danych ze składnicy pośredniej do docelowej hurtowni danych

Procesy ETL

- Ekstrakcja danych źródłowych
- Transformacja i czyszczenie danych źródłowych
- Indeksowanie i podsumowywanie danych
- Ładowanie danych do hurtowni danych
- Wykrywanie zmian w danych źródłowych
- Odświeżanie danych

- **Efekt:** dane odpowiednie, użyteczne, wysokiej jakości, dokładne, dostępne

Źródła danych

- **Produkcyjne**
 - systemy operacyjne, operacyjne bazy danych (IMS, DB2, Oracle, Sybase, Informix), systemy plików, dedykowane aplikacje (SAP, PeopleSoft, Oracle Financials)
- **Zarchiwizowane**
 - dane historyczne, potrzebne do inicjalizacji hurtowni, mogą wymagać unikalnej transformacji
- **Zewnętrzne**
 - komercyjne bazy danych, Internet, problemy związane z formatem, częstotliwością odświeżania, przewidywalnością
- **Wewnętrzne**
 - wewnętrzne bazy danych, dokumenty, arkusze kalkulacyjne

Techniki ekstrakcji

- Programy
 - C, C++, COBOL, Java, PL/SQL
- Bramy (ang. gateways)
 - Oracle Transparent Gateways (MS SQL Server, DB2, Informix, Sybase, Ingres, Teradata, AS/400, MQSeries)
- Narzędzia do ekstrakcji
 - <http://www.dbmsmag.com/9706d16.html>
- Własne rozwiązania

Metadane

- Lokalizacja, typ i struktura źródła danych
- Metoda dostępu do danych źródłowych
- Informacje o prawach dostępu do danych źródłowych
- Procedury obsługi błędów ekstrakcji
- Metody sprawdzania poprawności danych
- Metody obsługi brakujących wartości

Obsługa błędów ekstrakcji

- Najczęstsze przyczyny błędów
 - czasowa niedostępność źródeł danych
 - błędne mapowanie wartości
 - nieaktualne metadane
 - niewłaściwe planowanie zajętości urządzeń
 - zmiany strukturalne w danych źródłowych
- Jakość procesu ekstrakcji
 - testowanie
 - dokumentowanie
 - monitorowanie

Transformacja

- Najtrudniejszy element ETL, zazwyczaj wymaga istnienia obszaru tymczasowego (ang. staging area), w którym następuje konsolidacja, czyszczenie, restrukturyzacja
- Lokalizacja obszaru tymczasowego
 - obszar zdalny (poza źródłem danych)
 - obszar lokalny (w ramach źródła danych)
- Techniki
 - czyszczenie
 - eliminacja niespójności
 - dodawanie i łączenie danych
 - integracja danych

Anomalie danych

- Nazewnictwo i kodowanie
 - ASCII, UTF-8, EBCDIC
- Semantyka danych
 - temperatura podawana w skali Celsjusza i Fahrenheita
- Błędy literowe
 - „Uniwersyteta Poznańska”
- Brak unikalnych kluczy globalnych

Najczęstsze problemy

- Klucze złożone
- Różne formaty danych wejściowych
- Wiele sprzecznych źródeł danych
- Brakujące wartości
- Duplikaty wartości
- Identyfikacja wspólnych encji
- Brakujące więzy referencyjne

Jeszcze o transformacji

- Zarządzanie **jakością** danych
- Generowanie podsumowań
- Obsługa wyjątków i błędów transformacji
- Metadane o transformacji
 - zasady transformacji
 - algorytmy i procedury
- Transformacja potokowa (Oracle)
 - tabele zewnętrzne
 - funkcje tablicowe
 - łączenie z hurtownią danych (operacja **MERGE**)

Ładowanie danych

- Proces przesyłania danych z obszaru tymczasowego do docelowej hurtowni danych
 - inicjalizacja hurtowni: bardzo duża ilość danych, unikalne procesy transformacji, znaczące przetwarzanie po załadowaniu
 - odświeżanie hurtowni: wykonywane wg. cykliów biznesowych, prostsze procesy transformacji i przetwarzania, mniejsza objętość danych
- Konstruowanie procesu ładowania
 - dostępne narzędzia, metody przesyłania, okno ładowania i odświeżania, objętość inicjalizacji i odświeżania, częstotliwość odświeżania, ziarnistość danych

Metody ładowania danych

- Specjalizowane narzędzia
- Własne programy 3GL
- Bramy (ang. gateways)
- Replikacja synchroniczna i asynchroniczna
- FTP
- Ręczne ładowanie

Przetwarzanie po załadowaniu

- Indeksowanie danych
 - możliwości: przed ładowaniem, podczas ładowania, po ładowaniu
 - indeksy unikalne: wyłączanie ograniczeń integralnościowych
- Odtwarzanie kluczy
- Wyliczanie agregatów
 - perspektywy materializowane
 - programy narzędziowe (Summary Advisor/SQLAccess Advisor)
- Sprawdzanie spójności danych

ETL w środowisku Oracle

- Podstawowe narzędzie: Oracle Warehouse Builder
- Programy narzędziowe
 - Export/Import
 - Data Pump
 - SQL*Loader
- Zaawansowane elementy SZBD
 - mechanizm Change Data Capture (pakiet `DBMS_LOGMNR_CDC_PUBLISH` i `DBMS_LOGMNR_CDC_SUBSCRIBE`)
 - przenaszalne przestrzenie tabel (transportable tablespaces)
 - mechanizm strumieni (Oracle Streams)
 - tabele zewnętrzne
 - zaawansowane elementy SQL (wstawianie wielotablicowe, instrukcja **MERGE**, funkcje tablicowe, równoległy DML, perspektywy materializowane)

Przenaszalne przestrzenie tabel

- Przenaszalne przestrzenie tabel (ang. transportable tablespaces) umożliwiają kopiowanie danych między różnymi bazami danych działającymi na różnych platformach
 - ułatwiają transport danych między hurtownią danych i tematycznymi hurtowniami danych
 - umożliwiają współdzielenie danych „tylko do odczytu” w środowisku heterogenicznym
 - ułatwiają migrację bazy danych między platformami
- Przeniesienie przestrzeni tabel polega na fizycznym przekopiowaniu plików danych i wyeksportowaniu metadanych ze słownika bazy danych

Procedura przeniesienia przestrzeni tabel

- Przestrzeń tabel `USER_DATA`, `udata01.dbf`, system operacyjny: Linux (IA 32b), przeniesienie do systemu operacyjnego Windows (IA 64b)

1. Przełączenie przestrzeni tabel w tryb READ ONLY

```
ALTER TABLESPACE USER_DATA READ ONLY;
```

2. Eksport danych przestrzeni

```
$ exp tablespaces=USER_DATA transport_tablespace=Y file=umetadata01.dbf
```

3. Binarne przekopiowanie plików `udata01.dbf` i `umetadata01.dbf`

4. „Podpięcie” przestrzeni tabel do nowej bazy danych

```
$ imp tablespaces=USER_DATA transport_tablespace=Y  
file=umetadata01.dbf datafiles='udata01.dbf'
```

Oracle Streams

- Technologia umożliwiająca replikację, kolejkowanie wiadomości, ładowanie hurtowni danych, informowanie o zdarzeniach. Umożliwia integrację heterogenicznych baz danych
- Elementy składowe
 - **Capture**: dane z plików dziennika powtórzeń zapisywane w postaci rekordów logicznych (Logical Change Records)
 - **Staging**: rekordy LCR przechowywane w obszarze tymczasowym
 - **Consumption**: rekordy są przetwarzane i konsumowane
- Implementacja
 - pakiet `DBMS_STREAMS_ADM`

Export/Import

- Programy narzędziowe `exp` i `imp` umożliwiające przenoszenie danych między różnymi instancjami bazy danych

```
C:\>EXP.EXE USERID=scott/tiger FILE=exp.dmp GRANTS=y  
INDEXES=y ROWS=y TABLES=(sklepy,produkty) CONSTRAINTS=n
```

```
Export: Release 10.1.0.3.0 - Production on Sat Feb 12 00:21:18 2005
```

```
Copyright (c) 1982, 2004, Oracle. All rights reserved.
```

```
Connected to: Oracle Database 10g Enterprise Edition Release 10.1.0.3.0 - Production  
With the Partitioning, OLAP and Data Mining options
```

```
Export done in US7ASCII character set and AL16UTF16 NCHAR character set  
server uses EE8ISO8859P2 character set (possible charset conversion)
```

```
Note: constraints on tables will not be exported
```

```
About to export specified tables via Conventional Path ...
```

```
. . exporting table                SKLEPY                20 rows exported  
. . exporting table                PRODUKTY            141 rows exported
```

```
Export terminated successfully without warnings.
```

Oracle Data Pump

- Programy narzędziowe stanowiące ulepszenie „starych” narzędzi do eksportu i importu danych. Umożliwiają:
 - restart przerwanych eksportu/importu danych
 - okresowe monitorowanie zadania eksportu/importu
 - eksport/import bezpośrednio między bazami danych, z pominięciem plików systemu operacyjnego
 - większą elastyczność (zmiana nazwy pliku, zmiana docelowej przestrzeni tabel, filtrowanie danych i metadanych)
- Dostępne w postaci programów `expdp` `impdp` oraz pakietów `DBMS_DATAPUMP` i `DBMS_METADATA`
- Pliki zrzutów nie są kompatybilne z `exp` i `imp` (!)

Oracle Data Pump - przykład

```

declare
  h1 NUMBER;
begin
  h1 := dbms_datapump.open(operation=>'EXPORT',job_mode=>'TABLE',job_name=>'EXPORT000004',
                           version=>'COMPATIBLE');

  dbms_datapump.set_parallel(handle=>h1,degree=>1);
  dbms_datapump.add_file(handle=>h1,filename=>'EXPDAT.LOG',directory=>'DANE',filetype=>3);
  dbms_datapump.set_parameter(handle=>h1,name=>'KEEP_MASTER',value=>0);
  dbms_datapump.metadata_filter(handle=>h1,name=>'SCHEMA_EXPR',value=>'IN(''SCOTT'')');
  dbms_datapump.metadata_filter(handle=>h1,name=>'NAME_EXPR',value=>'IN(''SKLEPY'')');
  dbms_datapump.set_parameter(handle=>h1,name=>'ESTIMATE',value=>'BLOCKS');
  dbms_datapump.add_file(handle=>h1,filename=>'EXPDAT%U.DMP',directory=>'DANE',filetype=>1);
  dbms_datapump.set_parameter(handle=>h1,name=>'INCLUDE_METADATA',value=>1);
  dbms_datapump.set_parameter(handle=>h1,name=>'DATA_ACCESS_METHOD',value=>'AUTOMATIC');
  dbms_datapump.start_job(handle=>h1,skip_current=>0,abort_step=>0);
  dbms_datapump.detach(handle=>h1);
end;
/

```

Oracle Enterprise Manager

ORACLE Enterprise Manager 10g Database Control

Export Options

Database: miner10g

Maximum Number of Threads in Export Job: 1

Estimate Disk Space

Optional File

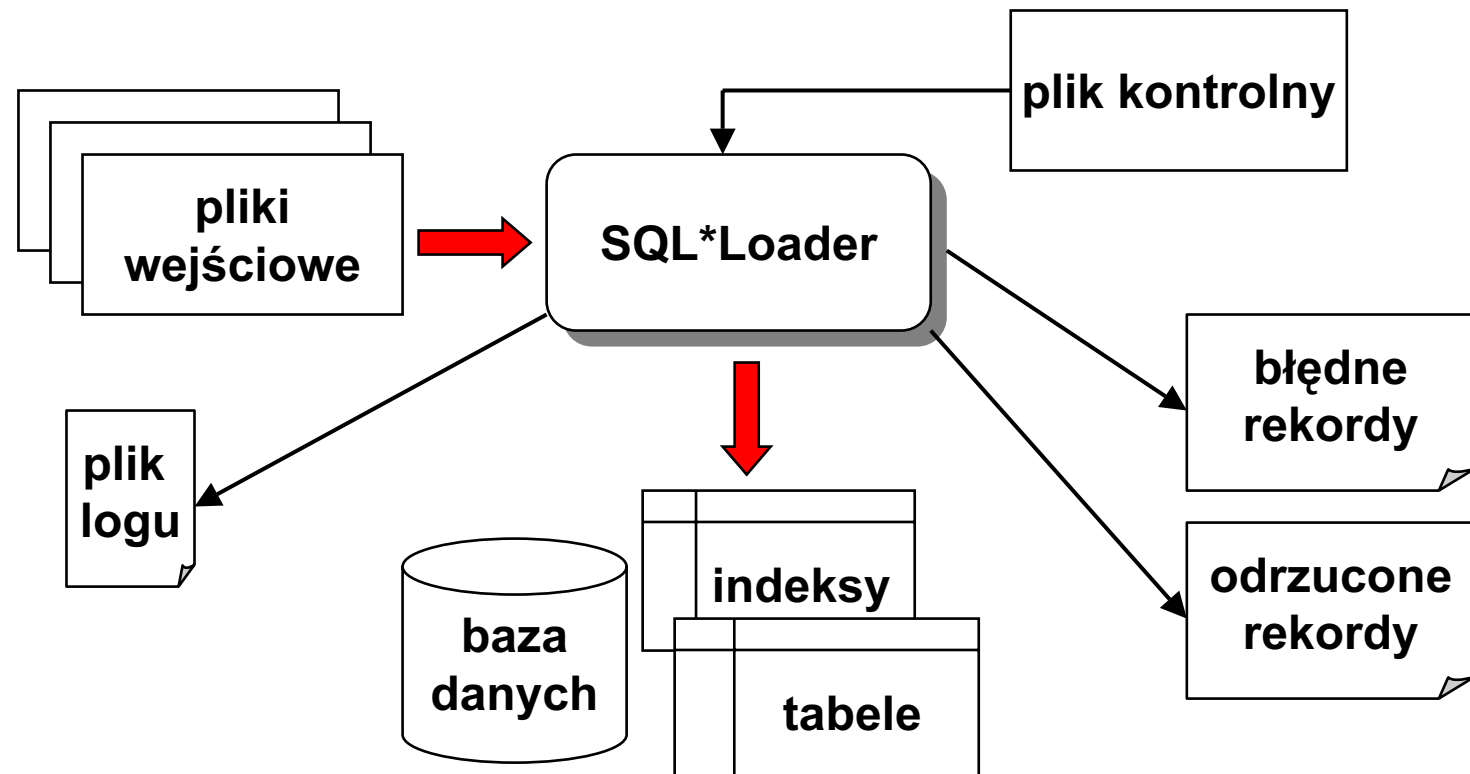
Generate Log File:

Directory Object: DANE

Log File: EXPDAT.LOG

SQL*Loader (1)

- Podstawowe narzędzie do ładowania bazy danych
- Posiada mechanizmy wstępnego przetwarzania i weryfikacji danych
- Ładuje dane do wielu tabel jednocześnie



SQL*Loader (2)

- Formaty danych wejściowych:
 - fixed record format : każdy rekord ma tę samą długość w bajtach
 - variable record format : długość każdego rekordu jest zapisana w pierwszym polu rekordu
 - stream record format : długość każdego rekordu jest dowolna

```
C:\> SQLLDR CONTROL=dane.ctl, LOG=dane.log, BAD=dane.bad,  
DATA=dane.dat USERID=scott/tiger, ERRORS=999,  
LOAD=2000, DISCARD=dane.dis
```

- CONTROL – plik sterujący
- LOG – plik dziennika ładowania
- BAD – plik ze złymi rekordami
- DISCARD – plik z odrzuconymi rekordami
- DATA – plik z danymi
- USERID – dane użytkownika
- ERRORS – maksymalna liczba błędów
- LOAD – liczba logicznych rekordów do załadowania

Metody ładowania

- Ścieżka konwencjonalna
 - wykorzystuje bufor danych
 - zapisuje dane w dzienniku powtórzeń
 - uruchamia wyzwalacze i sprawdza ograniczenia integralnościowe
 - nie ogranicza współbieżności
- Ścieżka bezpośrednia
 - zapisuje bezpośrednio do plików danych, z pominięciem bufora danych i dziennika powtórzeń (opcjonalnie)
 - sprawdza tylko wybrane ograniczenia (klucz podstawowy, unikalny, atrybut niepusty) i nie uruchamia wyzwalaczy
 - ogranicza współbieżność

Plik kontrolny (1)

```

LOAD DATA
INFILE * BADFILE 'dane.bad' DISCARDFILE 'dane.dis'
INTO TABLE produkty INSERT [ APPEND, REPLACE ]
  WHEN (3) != 'MOVIE'
FIELDS TERMINATED BY WHITESPACE [ ' ,' ]
OPTIONALLY ENCLOSED BY ' " '
( id_produktu NUMBER,
  nazwa_produktu CHAR "UPPER(:nazwa_produktu)",
  typ CHAR "DECODE(:typ, 'G', 'GAME', 'M', 'MOVIE', 'VIDEO')",
  kategoria_produktu CHAR,
  departament CHAR,
  global_id_produktu SEQUENCE(MAX,1) )
BEGINDATA
106,Destroyer,G,"Video Game","Game Rental"
107,Aliens,G,"Video Game","Game Rental"
92,Soda,F,"Non-alcohol","Beverage"
.....

```

- Dane mogą być w pliku zewnętrznym jak i w pliku kontrolnym. SQL*Loader umożliwia wstępne przetwarzanie ładowanych danych.

Plik kontrolny (2)

```

LOAD DATA
INFILE 'sprzedaz.dat' BADFILE 'dane.bad' DISCARDFILE 'dane.dis'
INTO TABLE produkty
( id_sklepu POSITION(1:4) INTEGER EXTERNAL NULLIF id_sklepu=BLANKS,
  id_czasu  POSITION(5:7) INTEGER EXTERNAL NULLIF id_sklepu=BLANKS,
  id_produktu      POSITION(8:12) INTEGER EXTERNAL,
  suma_sprzedazy  POSITION(13:19) DECIMAL EXTERNAL,
  suma_zysku      POSITION(20:26) DECIMAL EXTERNAL,
  liczba_klientow POSITION(27:29) INTEGER EXTERNAL,
  liczba_towarow  POSITION(30:32) INTEGER EXTERNAL,
  nazwa_sklepu    POSITION(33:46) CHAR,
  miejscowosc     POSITION(47:59) CHAR,
  region          POSITION(60:66) CHAR)

```

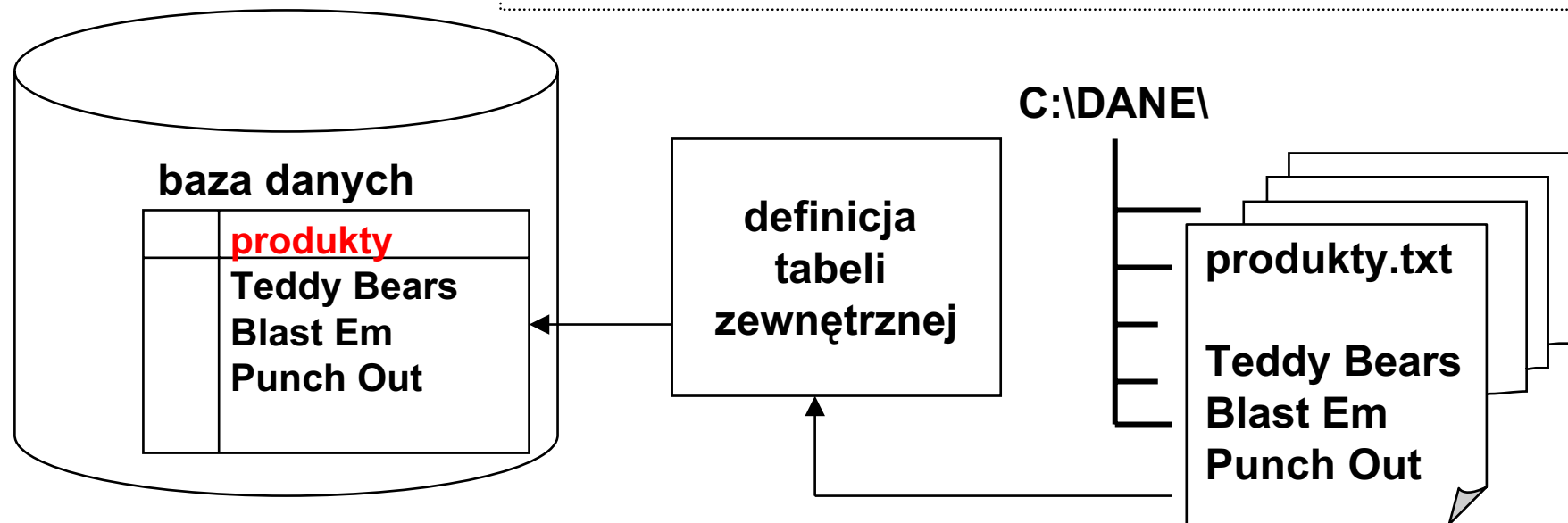
```
123456789012345678901234567890123456789012345678901234567890123456
```

19	4	109	22.66	22.66	7	7	Store No. 19	Boston	East	
9	4	61	2.35	1.41	1	1	Store No. 9	Seattle	West	
12	4	106	9.71	9.71	3	3	Store No. 12	Minneapolis	Central	
11				15	10.58	9	9	Store No. 11	Cincinnati	Central

sprzedaz.dat

Tabela zewnętrzna (1)

```
CREATE OR REPLACE DIRECTORY dane AS 'c:\dane';  
GRANT read, write ON DIRECTORY DANE TO PUBLIC;
```



- brak indeksów i ograniczeń integralnościowych
- tabele tylko do odczytu
- źródła danych w plikach w systemie operacyjnym

Tabela zewnętrzna (2)

sklepy.txt

```
Store No. 1;New York;East
Store No. 3;Atlanta;East
Store No. 4;Los Angeles;West
Store No. 5;San Francisco;West
Store No. 7;Pittsburgh;East
Store No. 8;New Orleans;East
Store No. 9;Seattle;West
Store No. 10;Dallas;Central
```

```
oracle_loader /
oracle_datapump
```

```
CREATE TABLE sklepy (
  nazwa_sklepu VARCHAR2(30),
  miejscowosc VARCHAR2(30),
  region VARCHAR2(20) )
ORGANIZATION EXTERNAL (
  TYPE oracle_loader
  DEFAULT DIRECTORY dane
  ACCESS PARAMETERS (
    RECORDS DELIMITED BY NEWLINE
    LOAD WHEN (nazwa_sklepu != BLANKS)
    BADFILE 'bledne.log'
    LOGFILE 'dziennik.log'
    FIELDS TERMINATED BY ','
    MISSING FIELD VALUES ARE NULL (
      nazwa_sklepu CHAR(30),
      miejscowosc CHAR(30),
      region CHAR(20) ))
  LOCATION (dane:'sklepy.txt'))
REJECT LIMIT UNLIMITED;
```

Funkcje tablicowe

```
CREATE TYPE array AS TABLE OF NUMBER;
/
```


```
CREATE OR REPLACE FUNCTION get_kwoty (p_typ IN VARCHAR2)
RETURN array PIPELINED AS
BEGIN
  FOR i IN (SELECT kwota FROM transakcje WHERE typ = p_typ)
  LOOP
    PIPE ROW (i.kwota);
  END LOOP;
  RETURN;
END get_kwoty;
/
```

```
SELECT * FROM
TABLE ( get_kwoty('WYPŁATA') );
```

COLUMN_VALUE
-100
-150
-120,5
-300
-150
-600
-150
...

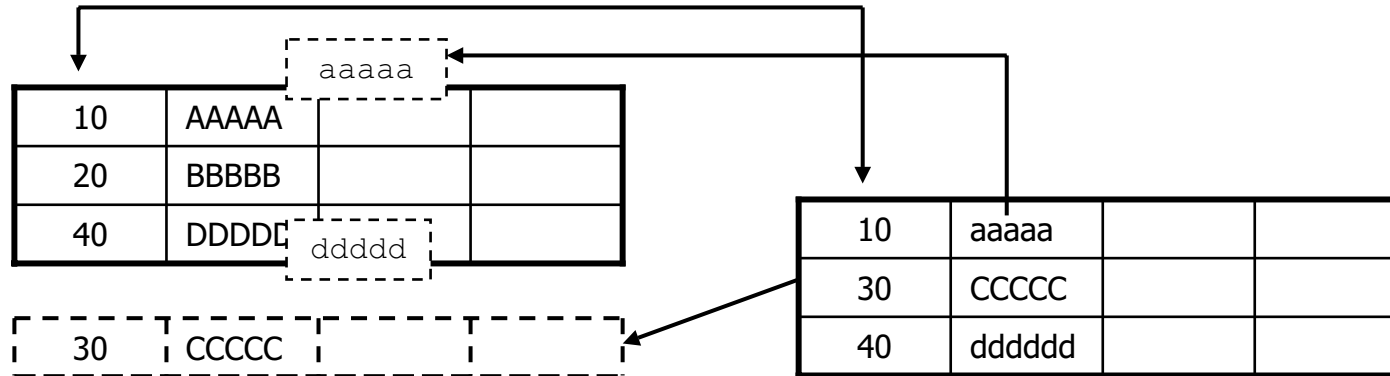
Wstawianie do wielu tabel

FIRST / ALL



```
INSERT FIRST
WHEN typ_produkty = 'GAME' THEN
    INTO gry(id_gry,nazwa_gry,liczba_sprzedanych_gier)
    VALUES (id_produkty,nazwa_produkty,liczba_sprzedanych_sztuk)
WHEN typ_produkty = 'MOVIE' THEN
    INTO filmy(id_filmu,tytul,liczba_sprzedanych_kaset)
    VALUES (id_produkty,nazwa_produkty,liczba_sprzedanych_sztuk)
WHEN typ_produkty = 'FOOD' THEN
    INTO towary(id_towaru,nazwa,liczba_sprzedanych_towarow)
    VALUES (id_produkty,nazwa_produkty,liczba_sprzedanych_sztuk)
ELSE
    INTO reszta(id_produkty,nazwa_produkty,liczba_sprzedanych_sztuk)
    VALUES (id_produkty,nazwa_produkty,liczba_sprzedanych_sztuk)
SELECT id_produkty, nazwa_produkty, typ_produkty, SUM(liczba_towarow)
FROM produkty NATURAL JOIN sprzedaz;
```

Operacja MERGE



```

MERGE INTO produkty_arch pa
USING ( SELECT id_produktu, nazwa_produktu, typ_produktu
          FROM produkty ) p
ON ( pa.id_produktu = p.id_produktu )
WHEN MATCHED THEN
    UPDATE SET pa.nazwa_produktu = p.nazwa_produktu,
              pa.typ_produktu = p.typ_produktu
WHEN NOT MATCHED THEN
    INSERT (pa.id_produktu, pa.nazwa_produktu, pa.typ_produktu)
    VALUES (p.id_produktu, p.nazwa_produktu, p.typ_produktu);
  
```

Równoległy DML (1)

- **Możliwość ręcznego „zrównoleglania” operacji**
 - wstawianie do wielu instancji RAC
 - modyfikacja i usuwanie dla różnych zakresów ROWID
- **Automatyczne wykonywanie operacji DML równolegle**
 - odświeżanie hurtowni danych
 - tworzenie tymczasowych tabel agregujących
 - uaktualnianie danych historycznych
 - wykonywanie zadań wsadowych
- **Włączanie równoległych operacji DML**

```
ALTER SESSION ENABLE PARALLEL DML;
```

Równoległy DML (2)

- Ograniczenia przetwarzania transakcyjnego
 - zatwierdzanie dwufazowe transakcji równoległej
 - zmiany wprowadzone przez równoległe polecenie nie mogą być widziane w ramach transakcji (brak możliwości ponownego odczytu lub zapisu tabeli)
- Wpływ na konsumpcję zasobów (dysk, blokady)
- Ograniczenia równoległego DML
 - przed wersją 9.2 tylko dla tabel partycjonowanych
 - nie może być wykonany na tabeli z aktywnymi wyzwalaczami
 - nie może być wykonany jeśli na niepartycjonowanej tabeli istnieją założone indeksy bitmapowe
 - nie może być wykonany na tabeli z ograniczeniami (referencyjnym zwrotnym, DELETE CASCADE, integralnościowymi opóźnionymi)
 - nie może być wykonany dla tabel w klastrze
 - nie może być wykonany dla atrybutów obiektowych
 - nie może być wykonany w ramach transakcji rozproszonej

Przykłady równoległego DML

```
INSERT /*+ PARALLEL(sprzedaz,3) */
INTO sprzedaz
SELECT * FROM sprzedaz_stg;
```

```
SELECT /*+ PARALLEL(produkty,3) */
* FROM produkty;
```

```
ALTER TABLE sprzedaz PARALLEL (DEGREE 3);
```

Plan wykonywania

```
-----
0  SELECT STATEMENT Optimizer=ALL_ROWS (Cost=1 Card=141 Bytes=6 204)
1 0  PX COORDINATOR
2 1    PX SEND* (QC (RANDOM)) OF 'TQ10 000' (Cost=1 Card=141 Bytes=6204)           :Q1000
3 2    PX BLOCK* (ITERATOR) (Cost=1 Card=141 Bytes=6204)                       :Q1000
4 3    TABLE ACCESS* (FULL) OF 'PRODUKTY' (TABLE) (Cost=1 Card=141 Bytes=6204) :Q1000
```

```
2 PARALLEL_TO_SERIAL
3 PARALLEL_COMBINED_WITH_CHILD
4 PARALLEL_COMBINED_WITH_PARENT
```

Odczytywanie danych i innej bazy danych

- Do połączenia z inną instancją bazy danych służą łącza bazodanowe

```
CREATE DATABASE LINK dblab
CONNECT TO scott IDENTIFIED BY tiger
USING 'dblab.cs.put.poznan.pl'
```

tnsnames.ora



```
INSERT /*+ APPEND PARALLEL(sprzedaz,3) */
INTO sprzedaz
SELECT * FROM fakty_sprzedazy@dblab;
```

Perspektywy materializowane w ETL

- Mechanizm replikacji z wykorzystaniem perspektyw materializowanych można wykorzystać do asynchronicznego przesyłania danych do hurtowni danych
- Cechy perspektywy materializowanej
 - sposób odświeżenia: przyrostowe, pełne
 - częstość odświeżania: definiowana przez START WITH i NEXT
 - konieczny dziennik perspektywy materializowanej (oparty na kluczu lub adresie ROWID)

Przykład użycia perspektyw materializowanych w ETL

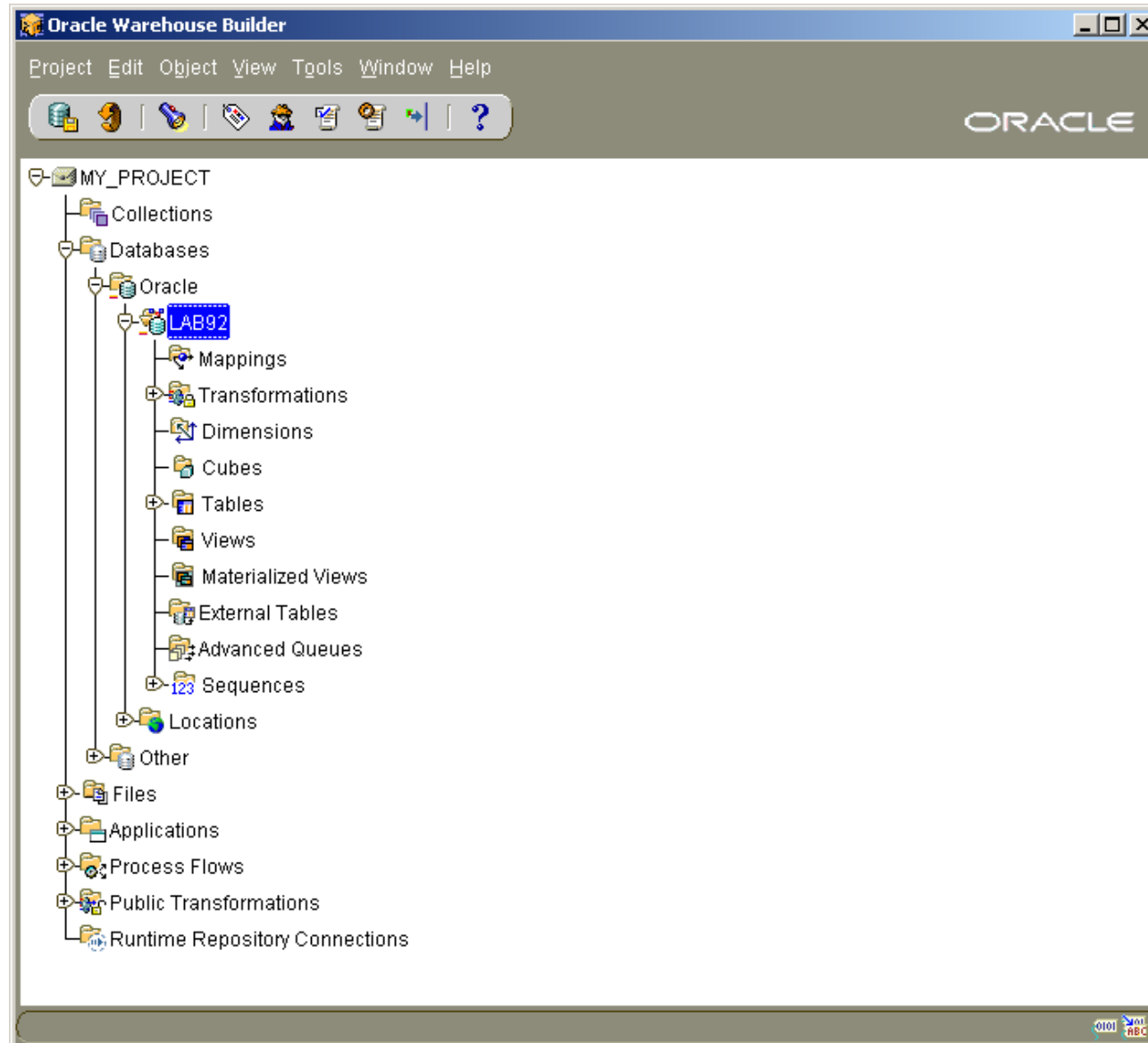
baza danych B1

```
CREATE MATERIALIZED VIEW LOG ON fakty_sprzedazy  
WITH PRIMARY KEY INCLUDING NEW VALUES;
```

baza danych B2

```
CREATE MATERIALIZED VIEW B1_sprzedaz  
BUILD IMMEDIATE  
REFRESH FORCE WITH PRIMARY KEY  
START WITH SYSDATE NEXT SYSDATE+(1/24)  
AS  
SELECT miejscowosc, typ_produkту, dzien_tygodnia,  
       SUM(liczba_klientow), AVG(liczba_towarow)  
FROM sklepy@b1 NATURAL JOIN produkty@b1  
              NATURAL JOIN czas@b1  
              NATURAL JOIN fakty_sprzedazy  
GROUP BY miejscowosc, typ_produkту, dzien_tygodnia;
```

Oracle Warehouse Builder - ETL



Dodanie pliku zewnętrznego (2)

lista atrybutów

Flat File Sample Wizard: Field Properties

Specify the field names and the field properties.

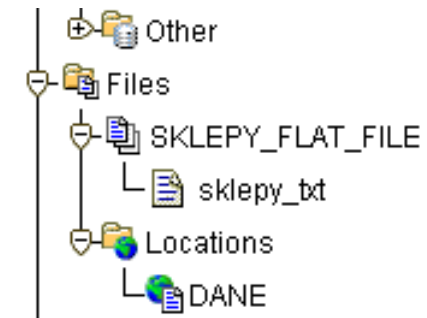
Name	Type	Mask	NULLIF	DEFAULTIF	Length	Precision	Scale	SQL Type	SQL I
SKLEP	CHAR				12			Default (VARCHAR2)	12
MIASTO	CHAR				13			Default (VARCHAR2)	13
REGION	CHAR				7			Default (VARCHAR2)	7

File: C:\Documents and Settings\nikolaj\Moje dokumenty\Dydaktyka\Szkola PLOUG\2005\sklepy.txt

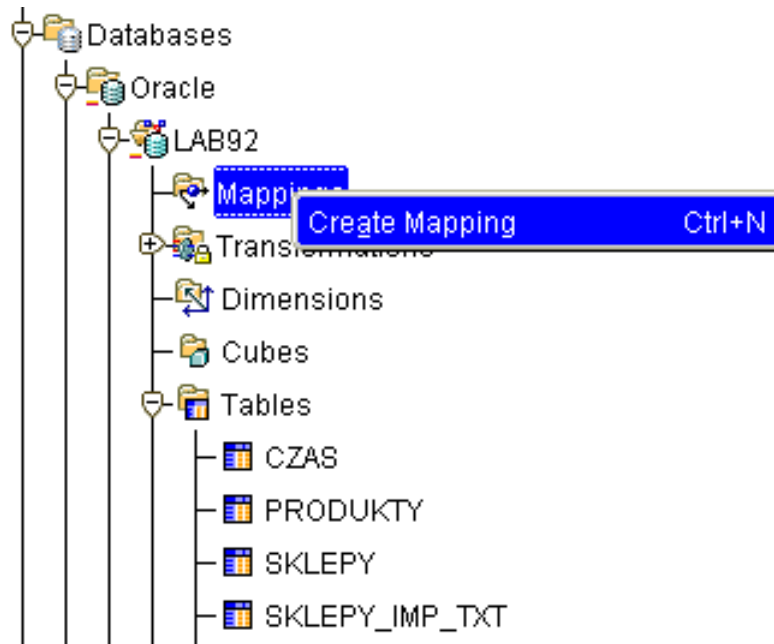
Use the first record as the field names

SKLEP	MIASTO	REGION
Store No. 1	New York	East
Store No. 3	Atlanta	East
Store No. 4	Los Angeles	West
Store No. 5	San Francisco	West
Store No. 7	Pittsburgh	East

Anuluj Pomoc < Wstecz Dalej >

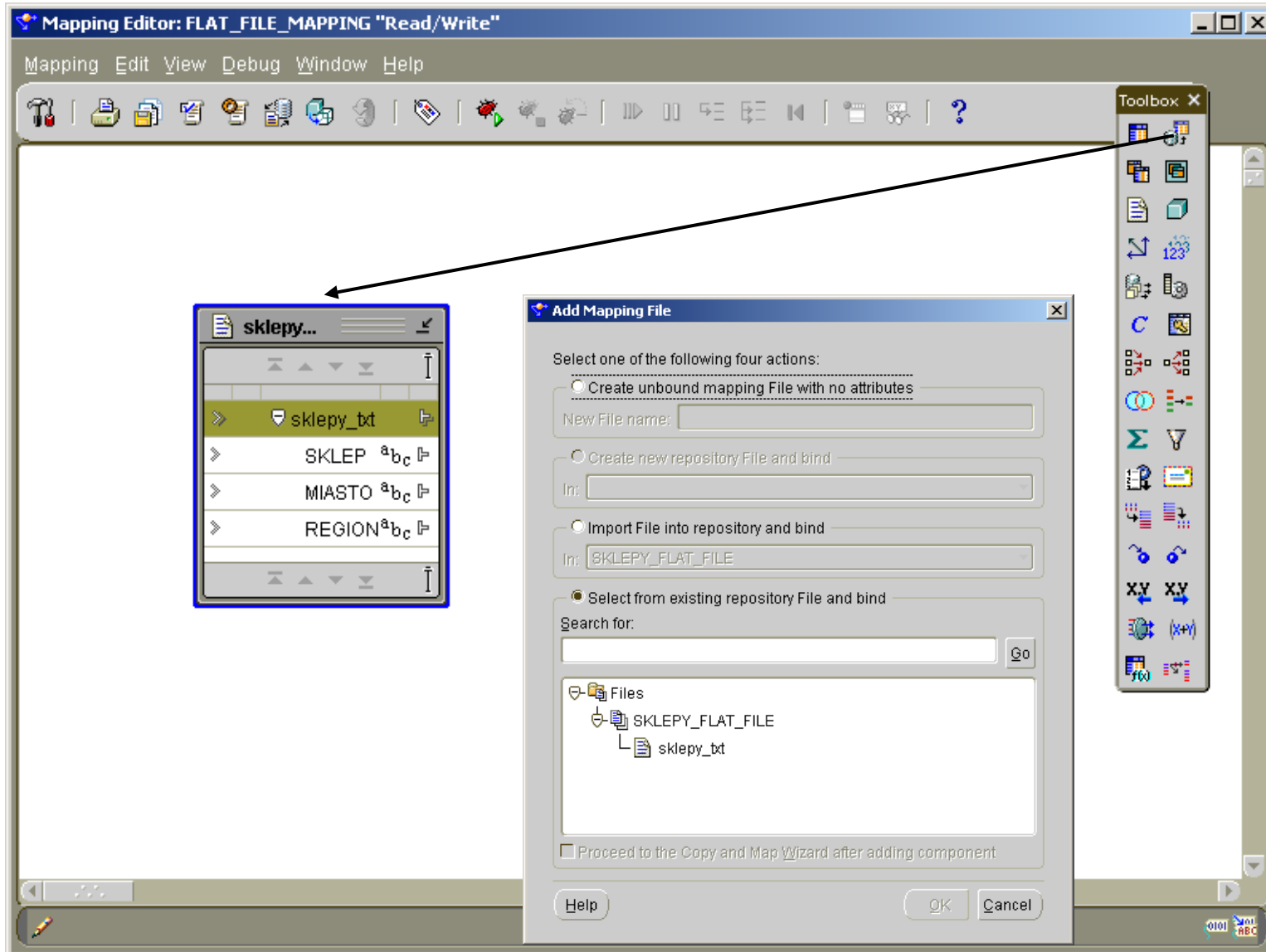


Utworzenie mapowania

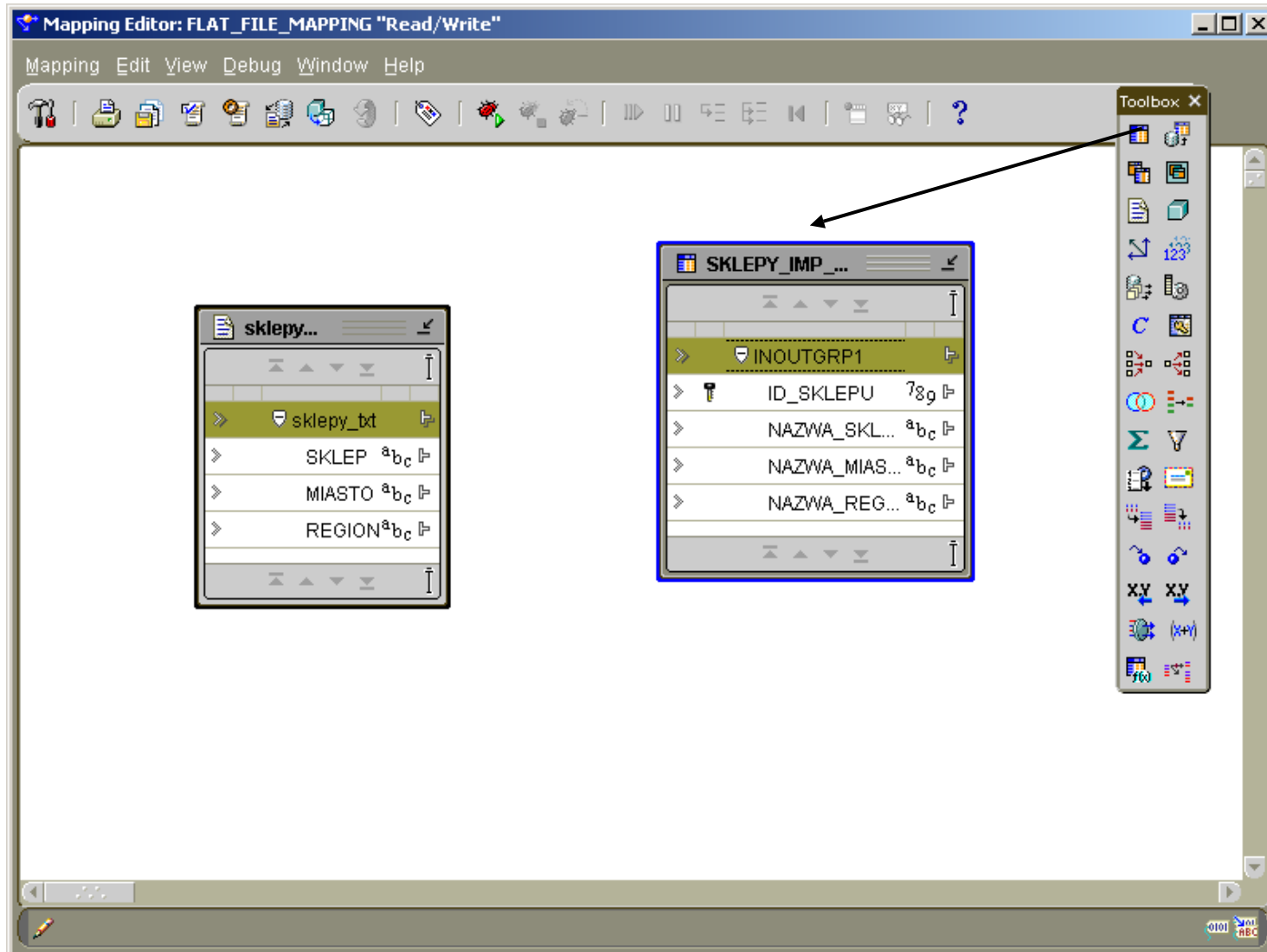


- Mapowanie jest obiektem OWB zawierającym wszystkie informacje potrzebne do zdefiniowania procesu ETL (specyfikację źródeł danych i obiektów docelowych)

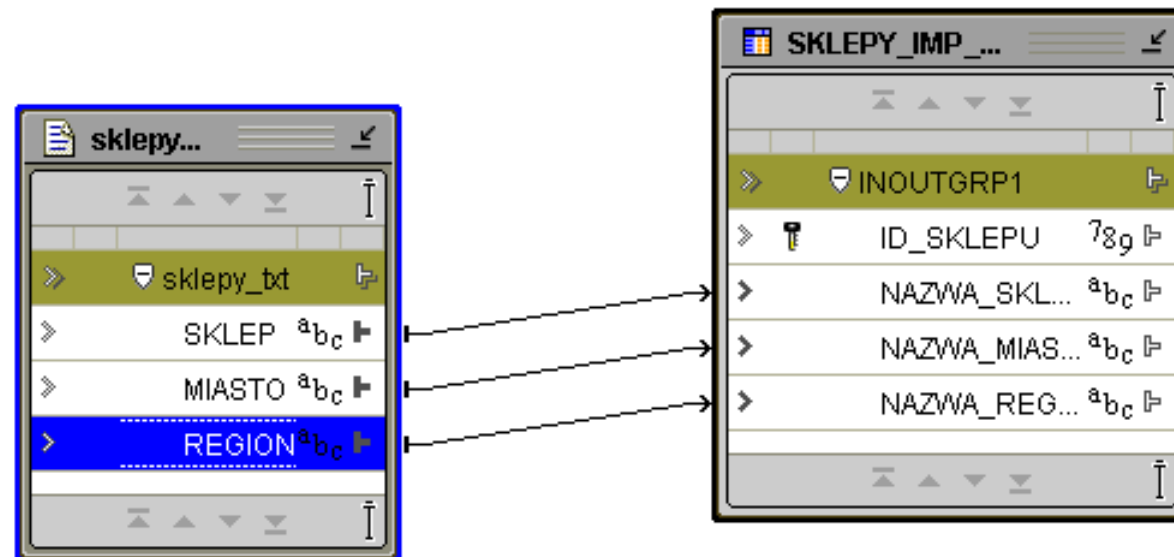
Dodanie źródła mapowania



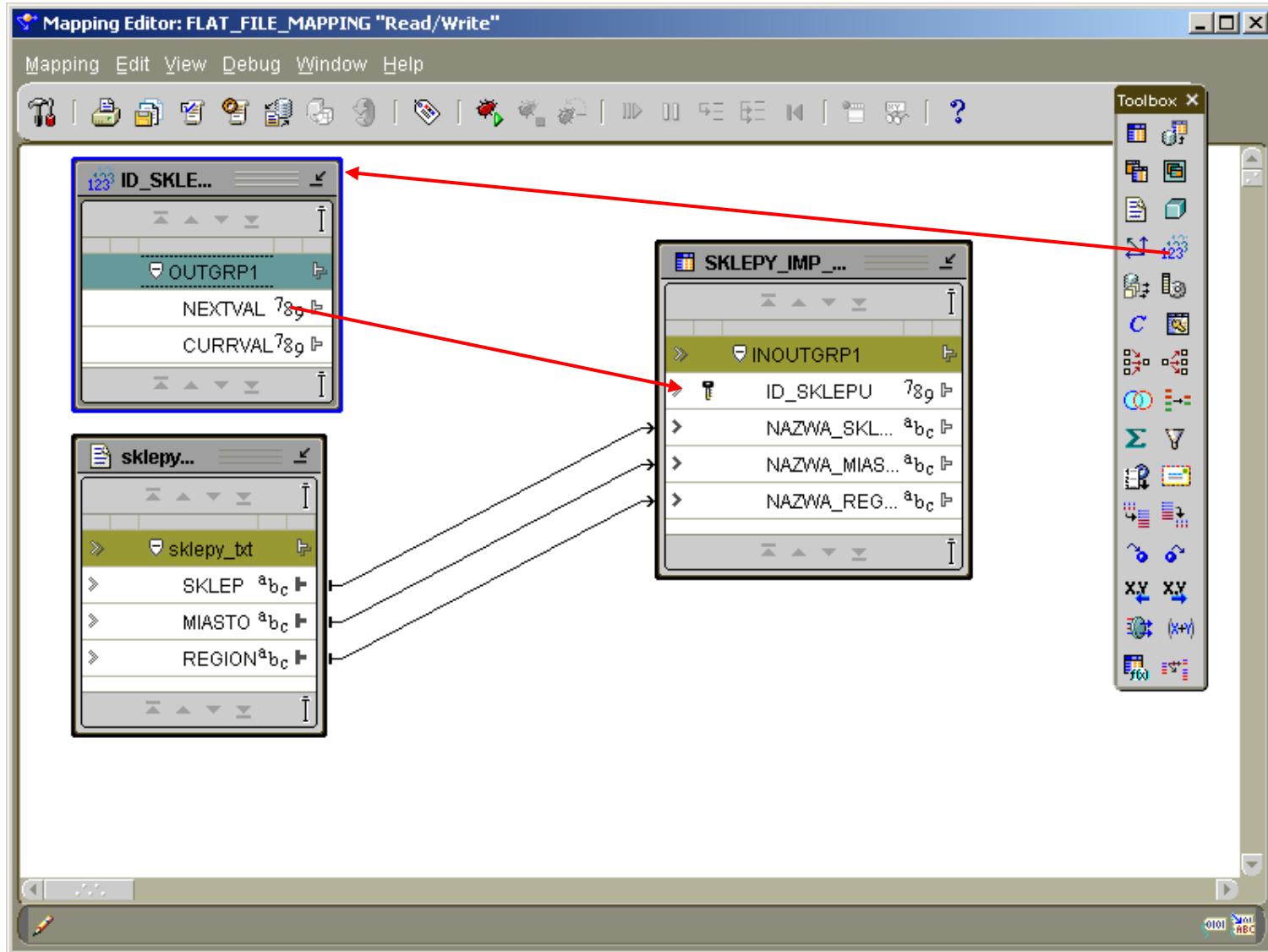
Dodanie mapowanej tabeli



Mapowanie kolumn



Dodanie sekwencera



Dodanie filtru

The screenshot illustrates the process of adding a filter to a data stream in a software application. The main interface shows a data stream named "SKLEPY_IMP..." with a group "INOUTGRP1" containing fields: "ID_SKLEPU" (789), "NAZWA_SKLEPU" (abc), "NAZWA_MIASTA" (abc), and "NAZWA_REGIONU" (abc). A filter component "FLTR" is connected to this stream, also containing the same fields. A red arrow points from the "FLTR" component to the "Filter Condition" dialog box.

The "Filter Properties: FLTR 'Read/Write'" dialog box shows the "Filter Condition" field. The "Filter Condition" dialog box is open, showing the "Inputs" tab with the following fields:

- INOUTGRP1
- 789 ID_SKLEPU
- abc NAZWA_SKLEPU
- abc NAZWA_MIASTA
- abc NAZWA_REGIONU

The "Filter Condition" dialog box also shows the "Filter Condition for FLTR" field with the following condition:

```
1 INOUTGRP1.NAZWA_REGIONU LIKE 'Cent%'
```

The dialog box includes a "Validation results" section and buttons for "Pomoć", "OK", and "Anuluj".

Automatyczna generacja kodu

The image displays two windows from the Oracle SQL Developer environment, illustrating the process of automatic code generation for a data integration task.

Code Viewer: FLAT_FILE_MAPPING [0 error(s), 4 warning(s)]

```

6 CREATE OR REPLACE PACKAGE "FLAT_FILE_MAPPING" AS
7   sql_stmt VARCHAR2(32767);
8   get_abort BOOLEAN := FALSE;
9   get_trigger_success BOOLEAN := TRUE;
10  get_errors NUMBER(22) := 0;
11  get_status NUMBER(22) := 0;
12  -- Status variable for Batch cursors
13  "SKLEPY_IMP_TXT_St" BOOLEAN := FALSE;
14  "FILTERED_SKLEPY_VIEW_St" BOOLEAN;
15
16  -- Function Main -- Entry point in package "FLAT_FL
17  FUNCTION Main RETURN NUMBER;
18
19  END "FLAT_FILE_MAPPING";
20
21 /
22
23 CREATE OR REPLACE PACKAGE BODY "FLAT_FILE_MAPPING"
24
25 -----
26 -- Function "SKLEPY_IMP_TXT_Bat"
27 -- performs batch extraction
28 -- Returns TRUE on success
29 -- Returns FALSE on failure
30 -----
31 FUNCTION "SKLEPY_IMP_TXT_Bat" RETURN BOOLEAN IS
32
33 BEGIN
34   EXECUTE IMMEDIATE 'ALTER SESSION ENABLE PARALLEL
35
36   BEGIN
37
38     INSERT
39     /*+ APPEND PARALLEL(SKLEPY_IMP_TXT, DEFAULT, DE
40     INTO
41     "SKLEPY_IMP_TXT"
42     (/*+ SKLEPY
  
```

Mapping Editor: FLAT_FILE_MAPPING "Read/Write"

The Mapping Editor window shows a data flow diagram with the following components and connections:

- Source:** ID_SKLEPU_SEQ (table icon) and SKLEPY_EX... (table icon).
- Target:** SKLEPY_IM... (table icon).
- Transformation:** FLTR (funnel icon).
- Target:** FILTERED_... (table icon).
- Target:** DATAGENERATOR (table icon).

Arrows indicate the flow of data from the sources to the SKLEPY_IM... table, then through the FLTR transformation to the FILTERED_... table, and finally to the DATAGENERATOR. A bidirectional arrow connects SKLEPY_IM... and FLTR.

The status bar at the bottom of the Code Viewer shows: Line 1 Column 1 | Read Only | [PL/SQL] Set based | Windows: CR/LF