Data warehousing a ETL

Firemní semináře 2013/2014

MFF UK Praha, 21.5.2014

Filip Moudrý, filip.moudry@javlin.eu

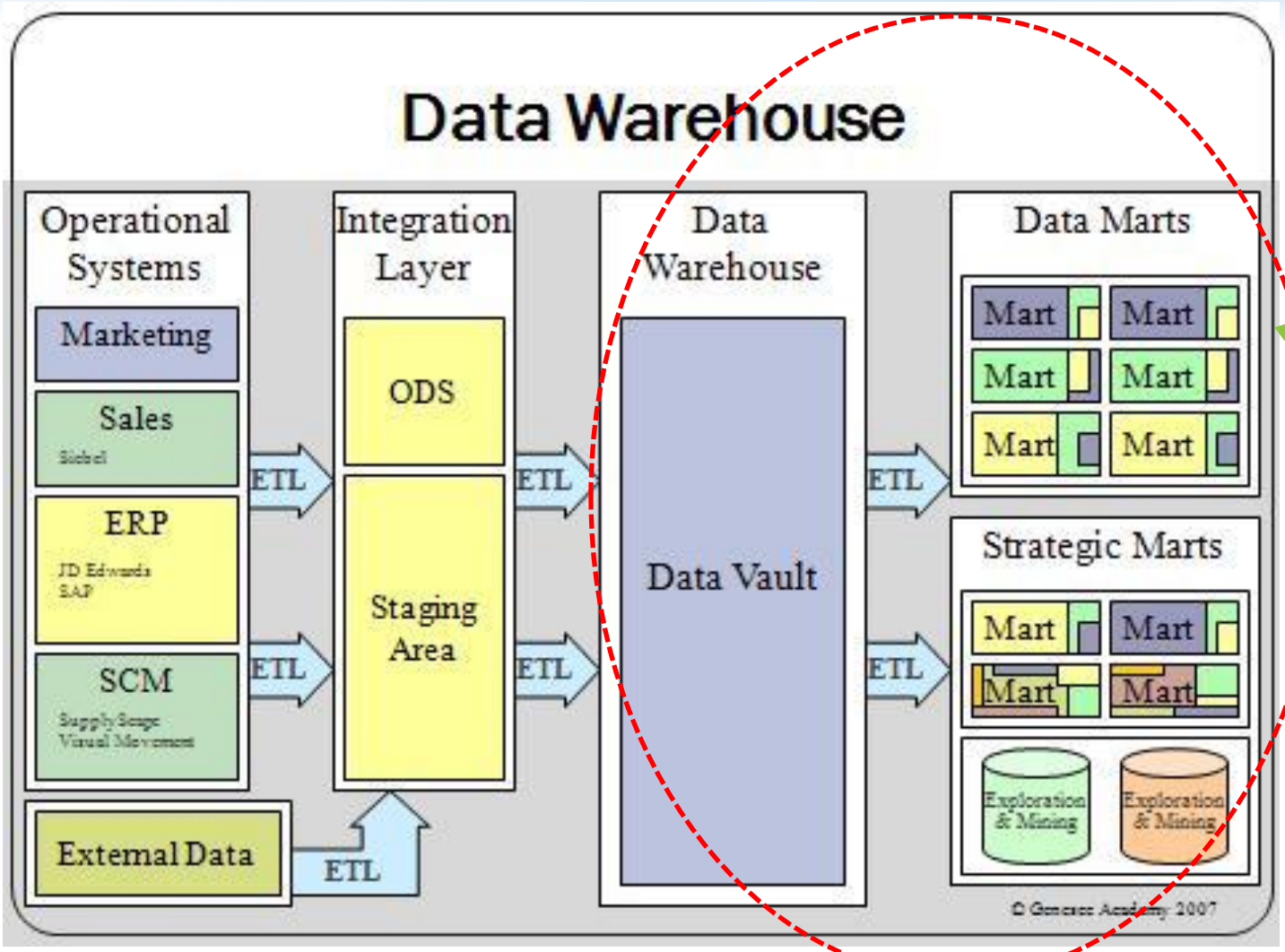
Program

- Javlin
- data warehouse
- ETL
- praktické ukázky
- diskuze

Když se řekne:

- data warehouse
- ETL
- data warehouse / ETL specialista

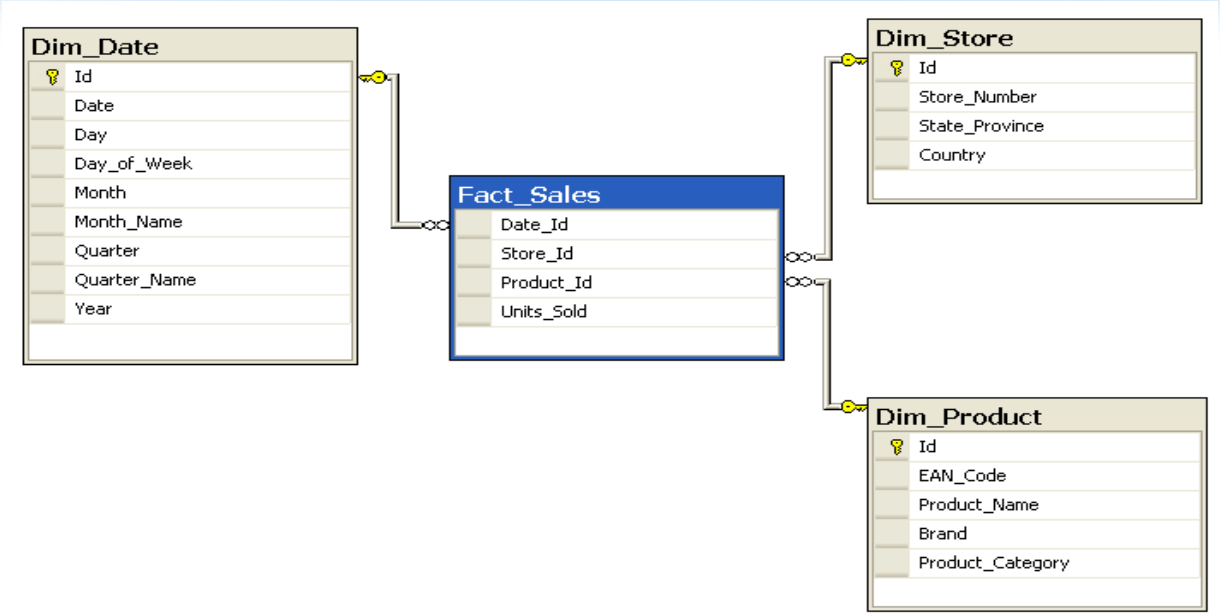
Datový sklad (DWH)



Datový sklad (DWH)

- centrální uložiště dat pro reporting a datovou analýzu
- zdroj pro BI nástroje
- integrována data z různých operativních zdrojů
- historická a aktuální data
- velký objem dat
- odlišné požadavky než u primárních systémů

Dimenzionální modelování



SELECT P.Brand, S.Country **AS** Countries, **SUM**(F.Units_Sold) **FROM** Fact_Sales F
INNER JOIN Dim_Date D **ON** F.Date_Id = D.Id
INNER JOIN Dim_Store S **ON** F.Store_Id = S.Id
INNER JOIN Dim_Product P **ON** F.Product_Id = P.Id
WHERE D.**YEAR** = 2013 **AND** P.Product_Category = ‘LED television'
GROUP BY P.Brand, S.Country

Dimenzionální modelování

- jedna nebo více faktových tabulek, které referencují libovolný počet dimenzionálních tabulek
- faktová tabulka obsahuje pouze cizí klíče (surrogate keys) dimenzionálních tabulek, atomická nebo sumarizovaná fakta
- faktová tabulka může obsahovat i degenerované dimenze
- dimenzionání tabulky o více řádů menší než faktové
- dimenzionální tabulky jsou **denormalizované**

Denormalizace dimenzionálních tabulek

- DWH je určen pro potřeby reportingu a business uživatelů
- datový model je snadno pochopitelný
- názvy atributů odpovídají jménům sloupců reportu, hodnoty jsou přímo vypisované
- odstranění zbytečných inner joinů pro číselníky
- není třeba užívat CASE…WHEN…OTHERWISE konstrukce v SQL dotazech

Denormalizace dimenzionálních tabulek

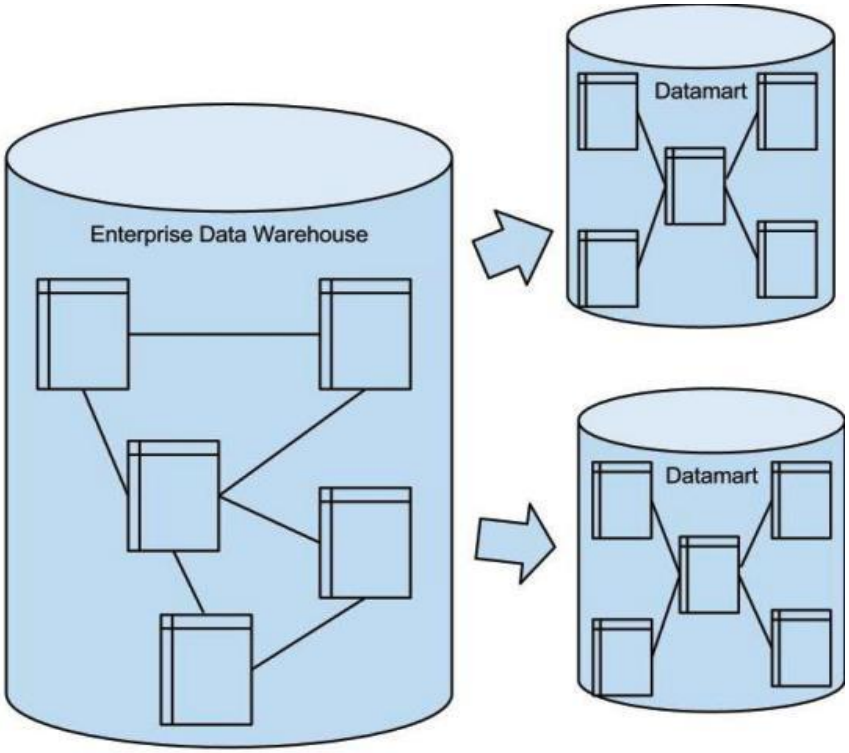
- dimenzionální tabulky jsou o více řádů menší než faktové tabulky
- v řádcích faktové tabulky zastoupeny pouze svým cizím klíčem
- typicky se dimenzionální tabulky příliš často nemění

	DateKey	FullDateUK	FullDateUSA	DayOfMonth	DayName	MMYYYY	MonthName	MONTH	YEAR	MonthYear	FirstDayOfMonth	LastDayOfMonth	FiscalDayOfYear	FiscalWeekOfYear	FiscalMMYYYY	FiscalMonth	FiscalQuarter	FiscalYear	FiscalYearName
1	20130101	01/01/2013	01/01/2013	1	Tuesday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	360	52	122012	12	4	2012	FY 2012
2	20130102	02/01/2013	01/02/2013	2	Wednesday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	361	52	122012	12	4	2012	FY 2012
3	20130103	03/01/2013	01/03/2013	3	Thursday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	362	52	122012	12	4	2012	FY 2012
4	20130104	04/01/2013	01/04/2013	4	Friday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	363	52	122012	12	4	2012	FY 2012
5	20130105	05/01/2013	01/05/2013	5	Saturday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	364	52	122012	12	4	2012	FY 2012
6	20130106	06/01/2013	01/06/2013	6	Sunday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	1	1	012013	1	1	2013	FY 2013
7	20130107	07/01/2013	01/07/2013	7	Monday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	2	1	012013	1	1	2013	FY 2013
8	20130108	08/01/2013	01/08/2013	8	Tuesday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	3	1	012013	1	1	2013	FY 2013
9	20130109	09/01/2013	01/09/2013	9	Wednesday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	4	1	012013	1	1	2013	FY 2013
10	20130110	10/01/2013	01/10/2013	10	Thursday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	5	1	012013	1	1	2013	FY 2013
11	20130111	11/01/2013	01/11/2013	11	Friday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	6	1	012013	1	1	2013	FY 2013
12	20130112	12/01/2013	01/12/2013	12	Saturday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	7	1	012013	1	1	2013	FY 2013
13	20130113	13/01/2013	01/13/2013	13	Sunday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	8	2	012013	1	1	2013	FY 2013
14	20130114	14/01/2013	01/14/2013	14	Monday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	9	2	012013	1	1	2013	FY 2013
15	20130115	15/01/2013	01/15/2013	15	Tuesday	012013	January	1	2013	Jan-2013	2013-01-01	2013-01-31	10	2	012013	1	1	2013	FY 2013

Základní přístupy k tvorbě DWH

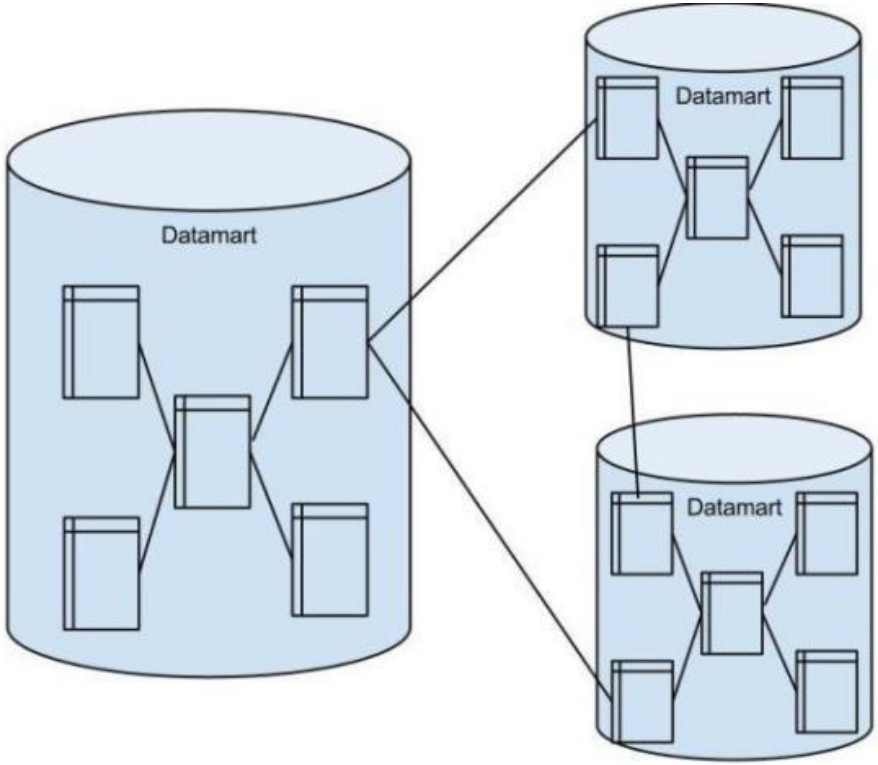
Bill Inmon (top-down)

- centralizovaný DWH dle ER modelování



Ralph Kimball (bottom-up)

- malé datamarty spojené přes konformní dimenze



Porovnání přístupů tvorby DWH

- vytvoření enterprise-wide DWH je časově a finančně nákladné
- top-down je robustní vzhledem ke změnám business požadavků
- z enterprise-wide DWH se snadno generují nové data marty
- bottom-up model zaručuje rychlé dodání business hodnoty
- bottom-up model lze inkrementálně rozvíjet

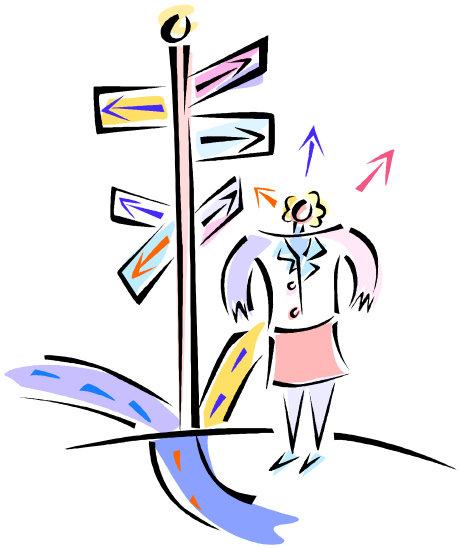
Postup návrhu bottom-up DWH

- určení business procesu
- určení granularity faktu
- definice dimenzí
- definice faktů
- *případné doplnění informací o pokrytí (nefaktové faktové tabulky),*
- validace zda vyhovuje požadovaným reportům

Typy modelů faktů

volba modelu závisí na business procesu a požadavcích reportingu

- transakční model
- pravidelný snapshot
- kumulovaný snapshot

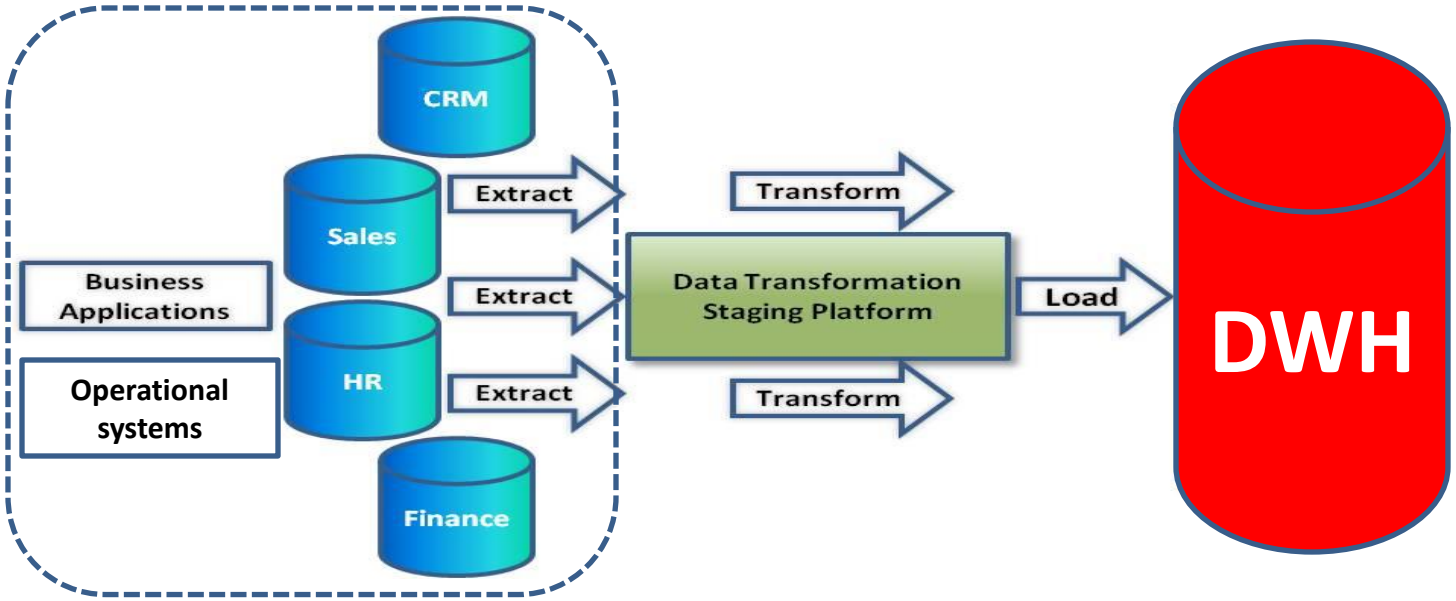


ETL

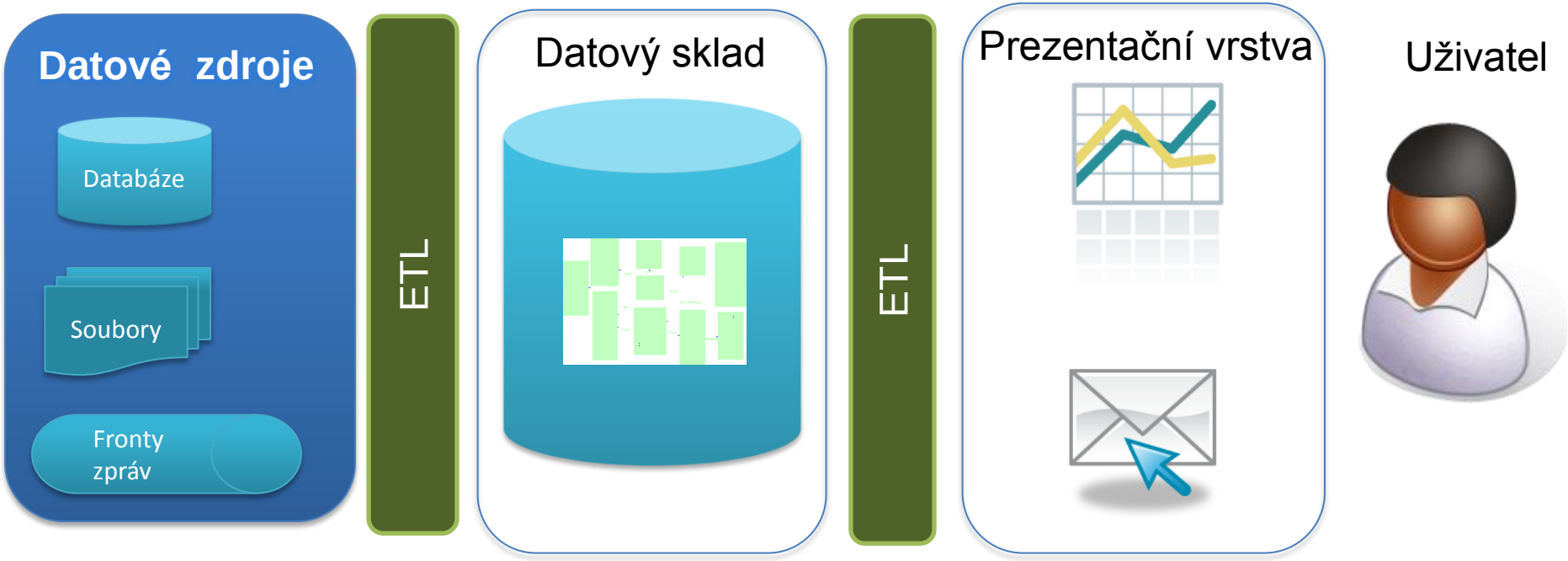
Extract the data from source systems

Transform the data

Load the data into target systems



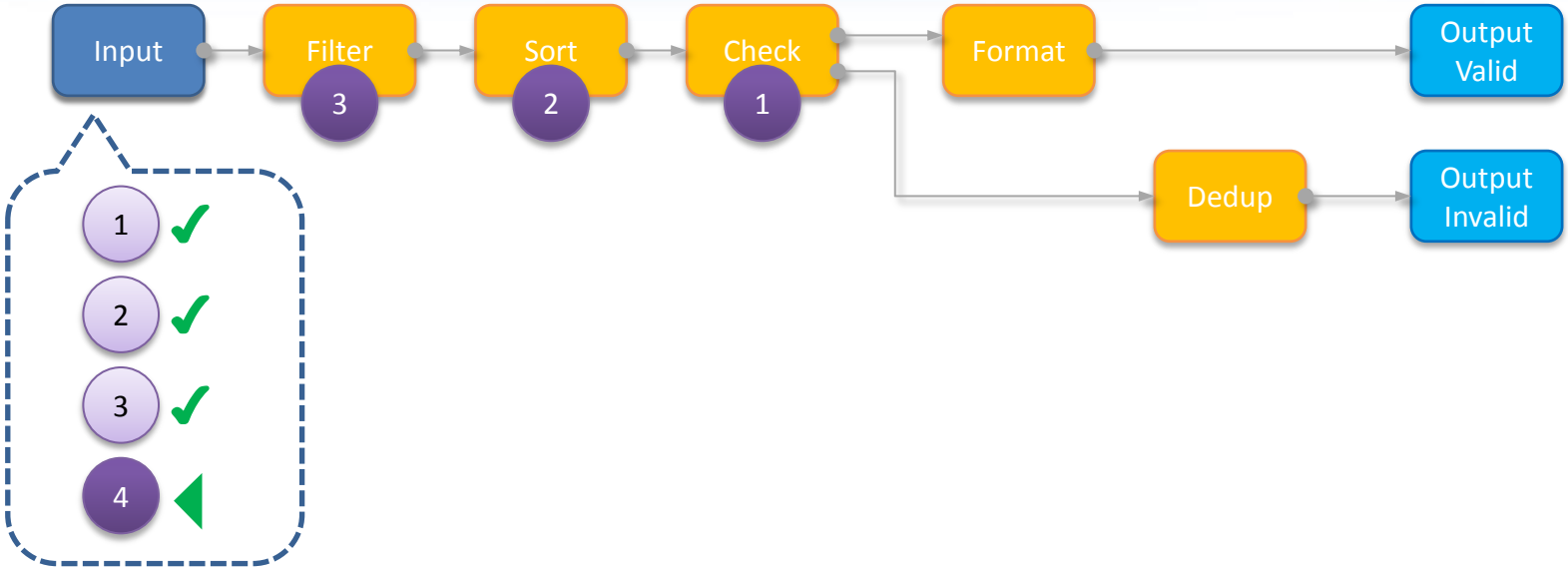
Datový sklad a ETL



Plánování, Extrakce, Transformace a zápis

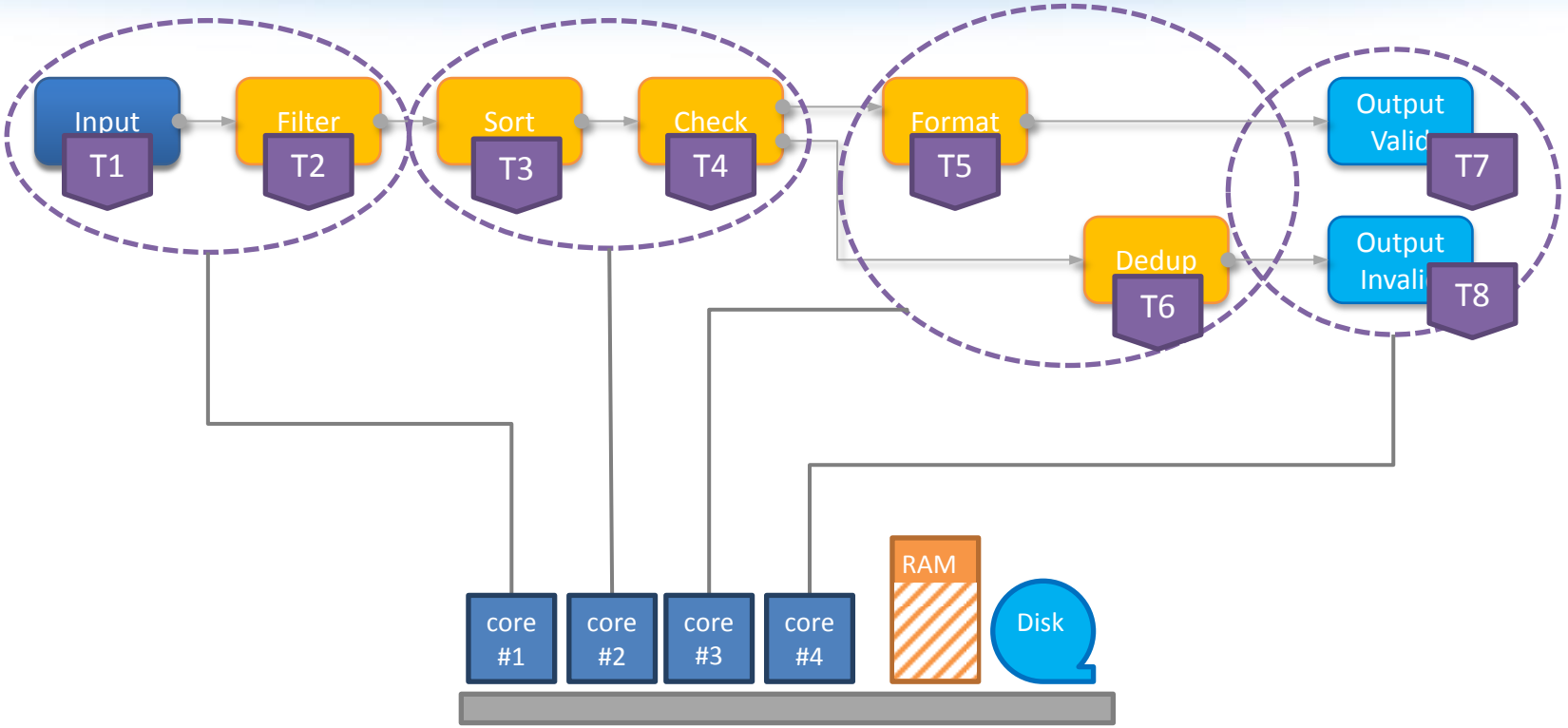
Extrakce

ETL – grafický model procesu



- graf lze chápat jako **produktovod**
- jednotlivé záznamy jsou sekvenčně čteny z datového zdroje a průběžně protékají transformačními komponentami až k výstupu

Výpočetní model CloverETL



- komponenty se vykonávají jako samostatná Java vlákna
- více vláken může sdílet stejný procesor
- v případě použití hardwarového clusteru lze paralelizovat zpracování mezi více stroji a rovněž explicitně určit alokaci komponent na uzly clusteru

Příklad ETL transformace

Data: Objednávky zákazníků

CUSTOMER_NAME	ORDER_ID	TOTAL_AMT	ADDR	CITY	STATE
Gallagher, Evangeline A.	292	3301.00	9812 Pharetra Avenue	Madison	AZ
Mike Cheong	150	12.35	154 Milles Dr	Boston	MA
Arnetta Ferrero	155	55.12	23 Cactus Rd	Phoenix	AZ
James D. Barber	314	120.49	113/2 Mule St	Chicago	IL
Frederic Gables	110	59.70	153 SW Rodeo Dr	Phoenix	AZ

```
CUSTOMER_NAME|ORDER_ID|TOTAL_AMT|ADDR|CITY|STATE
Gallagher, Evangeline A.|292|3301.00|9812 Pharetra Avenue|Madison|AZ
Mike Cheong|150|12.35|154 Milles Dr|Boston|MA
Arnetta Ferrero|155|55.12|23 Cactus Rd|Phoenix|AZ
James D. Barber|314|120.49|113/2 Mule St|Chicago|IL
Frederic Gables|110|59.70|153 SW Rodeo Dr|Phoenix|AZ
```

Úloha: Seznam 3 největších objednávek se jménem zákazníka, které byly doručené do Arizony

Řešení UNIX skriptem

```
cat customer_orders.txt |
grep "AZ$" |
sort -t"|" -k3 -n -r |
cut –d"|" –f1,2,3 |
tail -n 3
```

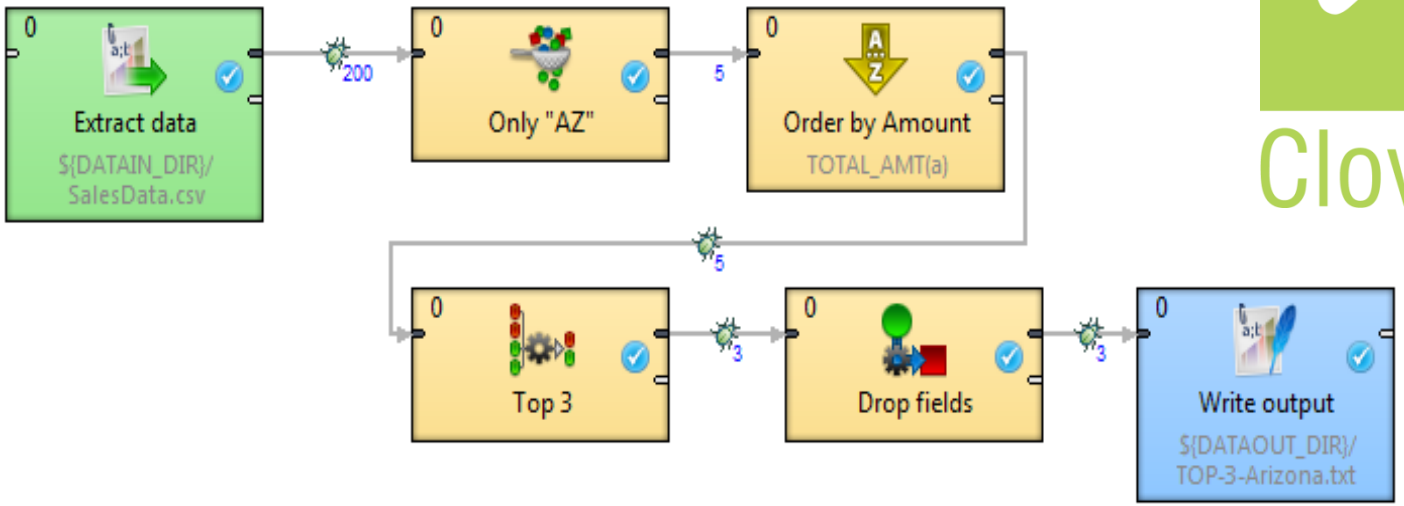
- co když chci také sčítat objednávky?

Řešení pomocí SQL

```
SELECT
customer_name,order_id,total_amt
FROM customer_orders
WHERE state = 'AZ'
ORDER BY total_amt DESC
LIMIT 3
```

- jak dostanu data do databáze?

Řešení pomocí CloverETL



CloverETL

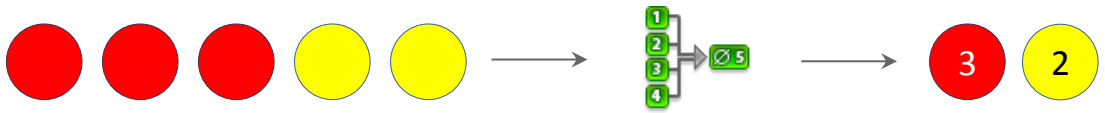
Příklad ETL agregace

- agregace nesetříděných dat (klíč je barva, agregace je počet)

color	count
●	3
●	2

Všechna data jsou držena v paměti, dokud není zpracován celý vstup.

- agregace setříděných dat (klíč je barva, agregace je počet)



Příklad ETL denormalizace

Original data
Multiple records grouped based on the key.

Denormalize

Denormalized data
Single record containing values determined by processing the whole input group.

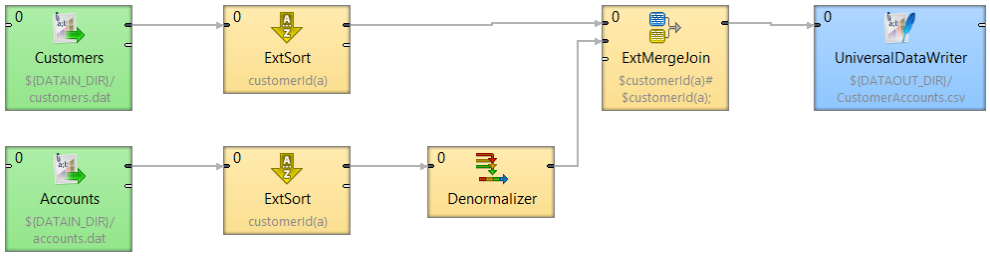
Account

accountId	customerId	balance	created	closed
9804568699	27345	2300.56	2011-11-14	
1108193472	27345	-1739.05	2005-07-22	
6054951154	27345	4500.60	2009-09-01	2010-04-30
9459175447	27345	3200.80	2011-03-08	

CustomerAccounts

customerId	totalBalance	accounts
27345	8262.91	[9804568699, 1108193472, 6054951154, 9459175447]

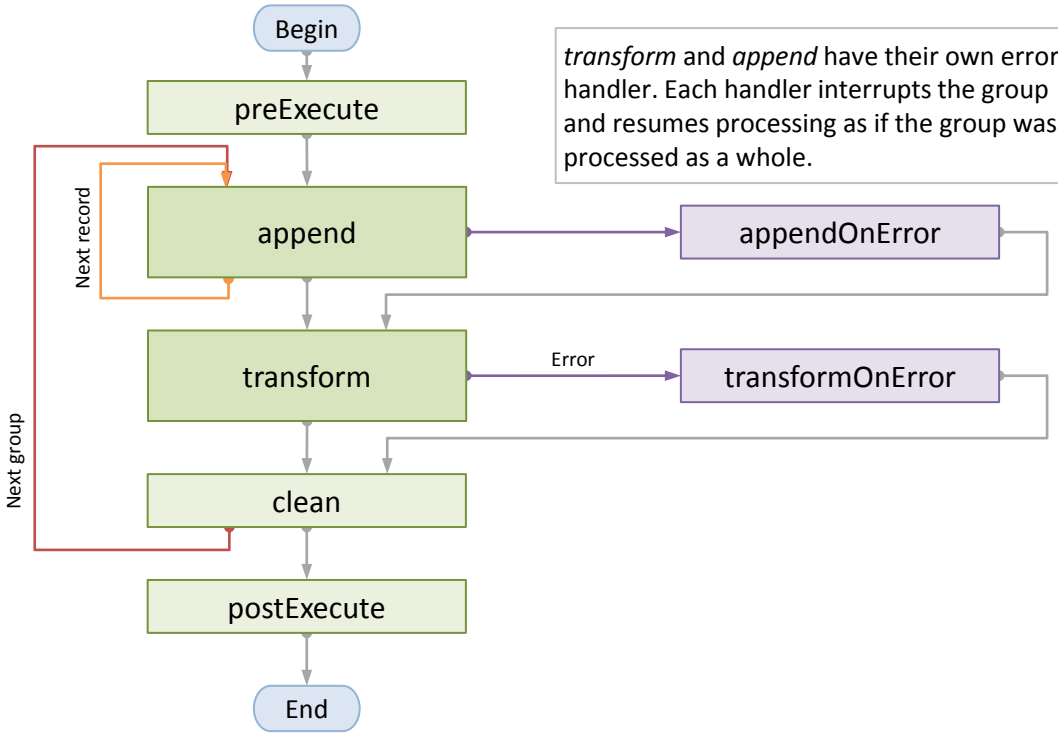
Příklad ETL denormalizace



append is called once for each record in a group. It is typically used to update global variables which are then used in *transform* function.

transform is called once per group and is the only function which generates output records.

clean is called after each transform and can be used to clean-up internal variables.



transform and *append* have their own error handler. Each handler interrupts the group and resumes processing as if the group was processed as a whole.

Příklad ETL normalizace

Original data

One complex record.

Normalize

Normalized data

Multiple (usually simple) records belonging to a group which shares a key with the source record.

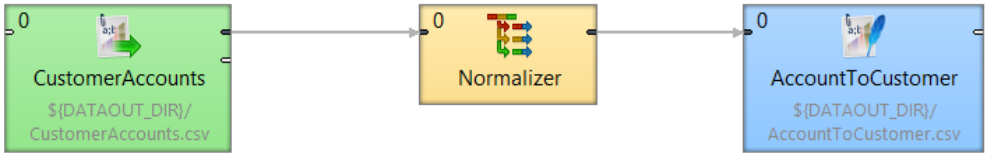
CustomerAccounts

customerId	totalBalance	accounts
27345	8262.91	[9804568699, 1108193472, 6054951154, 9459175447]

AccountToCustomer

accountId	customerId
9804568699	27345
1108193472	27345
6054951154	27345
9459175447	27345

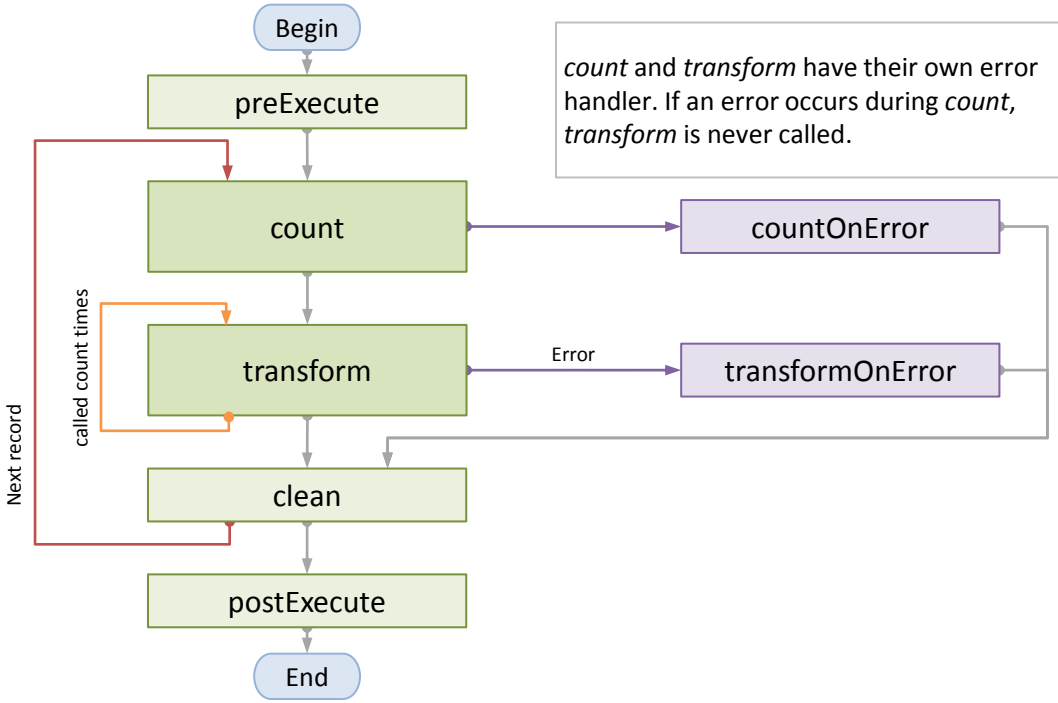
Příklad ETL normalizace



count is called once for each input record.

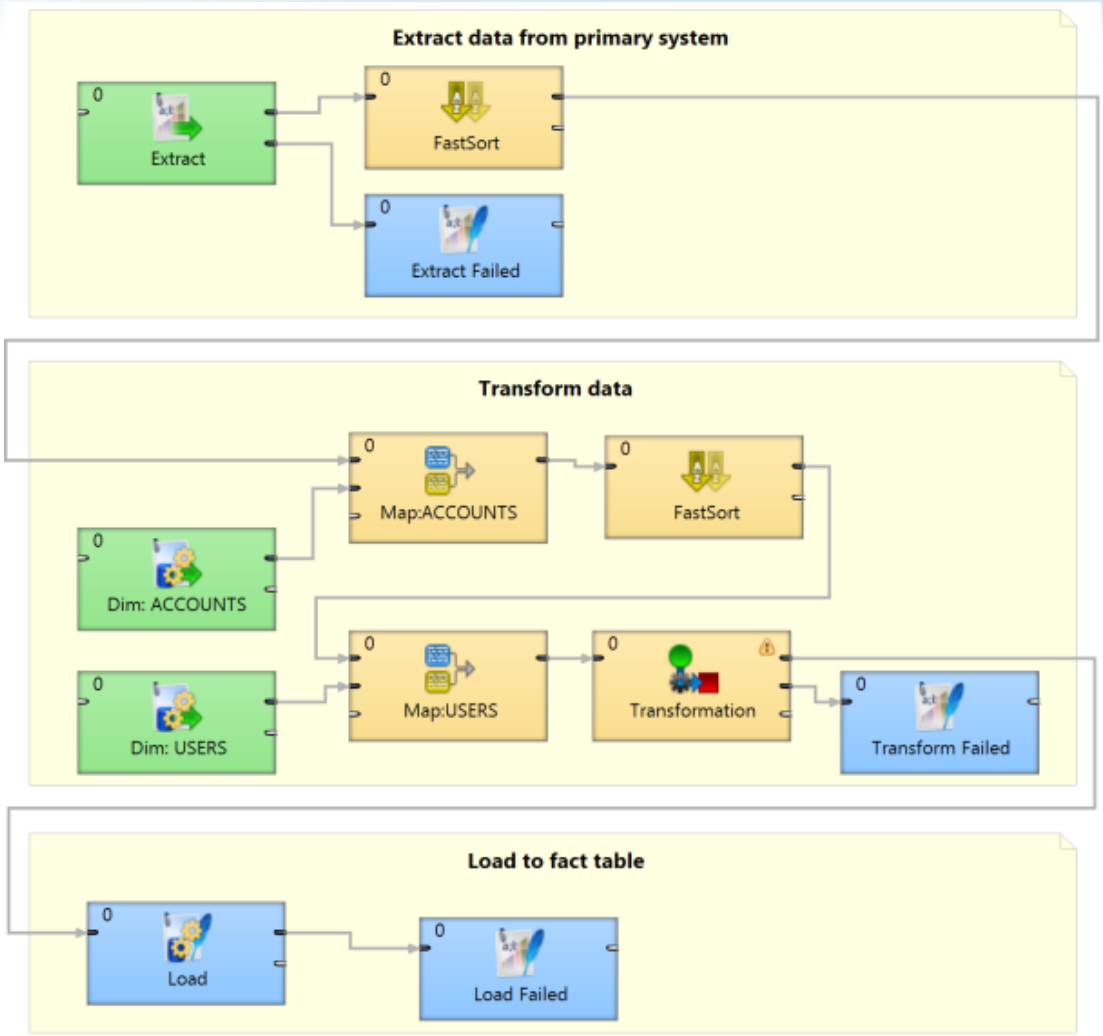
transform is called as many times as is the value returned by count function. It is therefore possible to generate up-to count records in transform.

clean is called after each transform and can be used to clean-up internal variables.



count and *transform* have their own error handler. If an error occurs during *count*, *transform* is never called.

Načtení dat do faktové tabulky

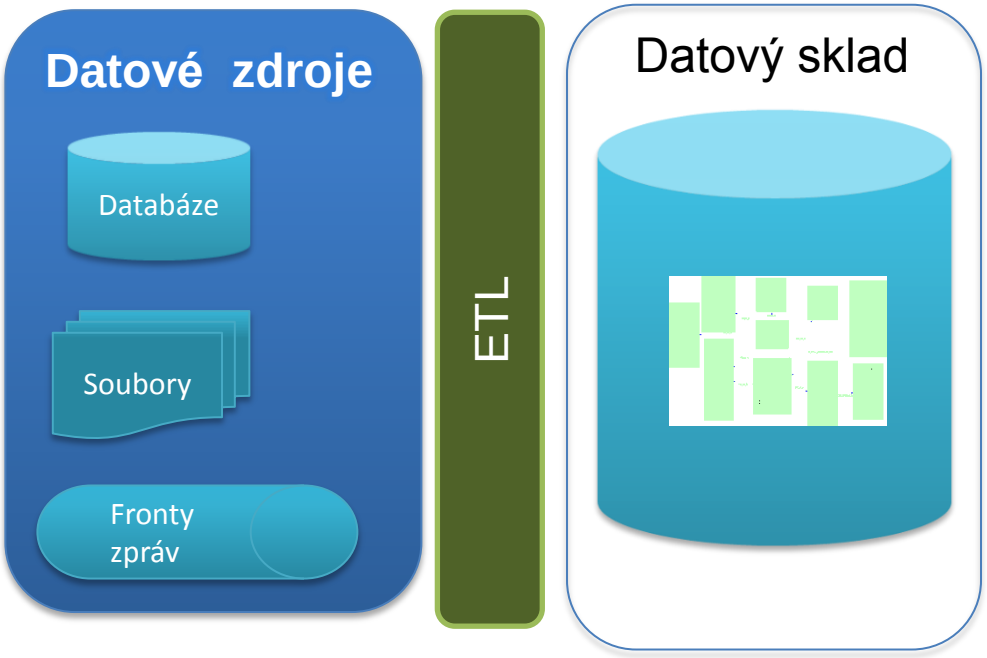


- logika patrná z grafu
- nutná orchestrace s náčtem dimenzionálních tabulek
- řešení chyby v jednotlivých fázích

ETL v praxi

Čtení z primárních zdrojů

- DB
- flatfile
- binary data file
- spreadsheet
- XML/JSON
- web service

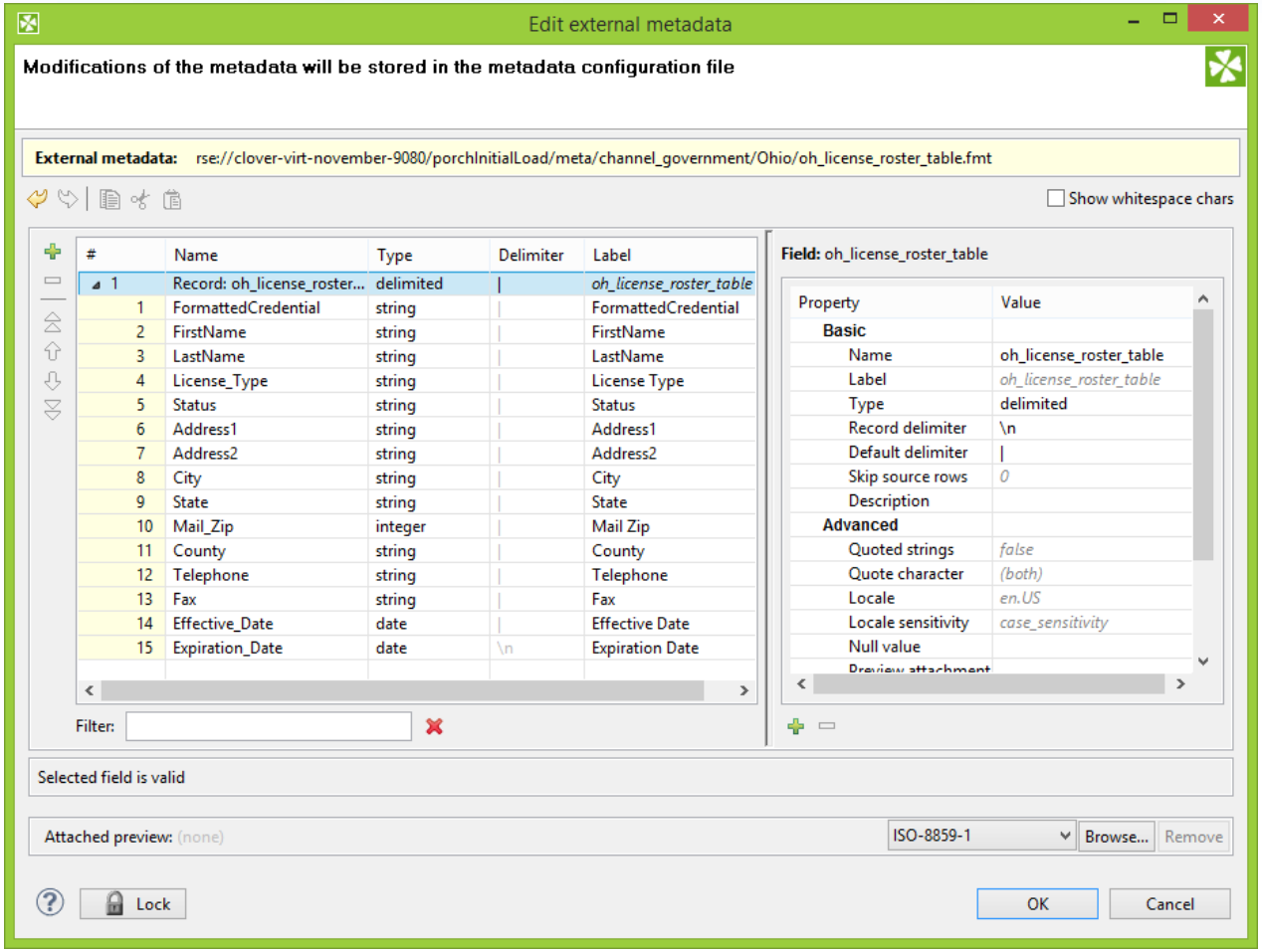


Zápis do DWH

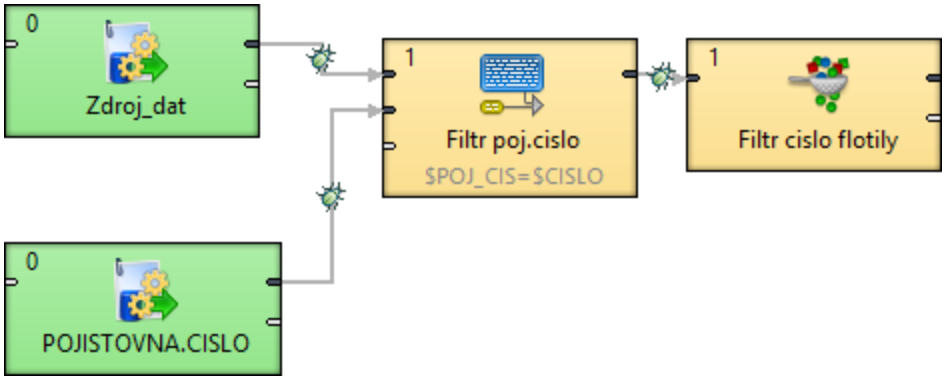
- DB
- web service

Plánování, Extrakce, Transformace a zápis

Metadata

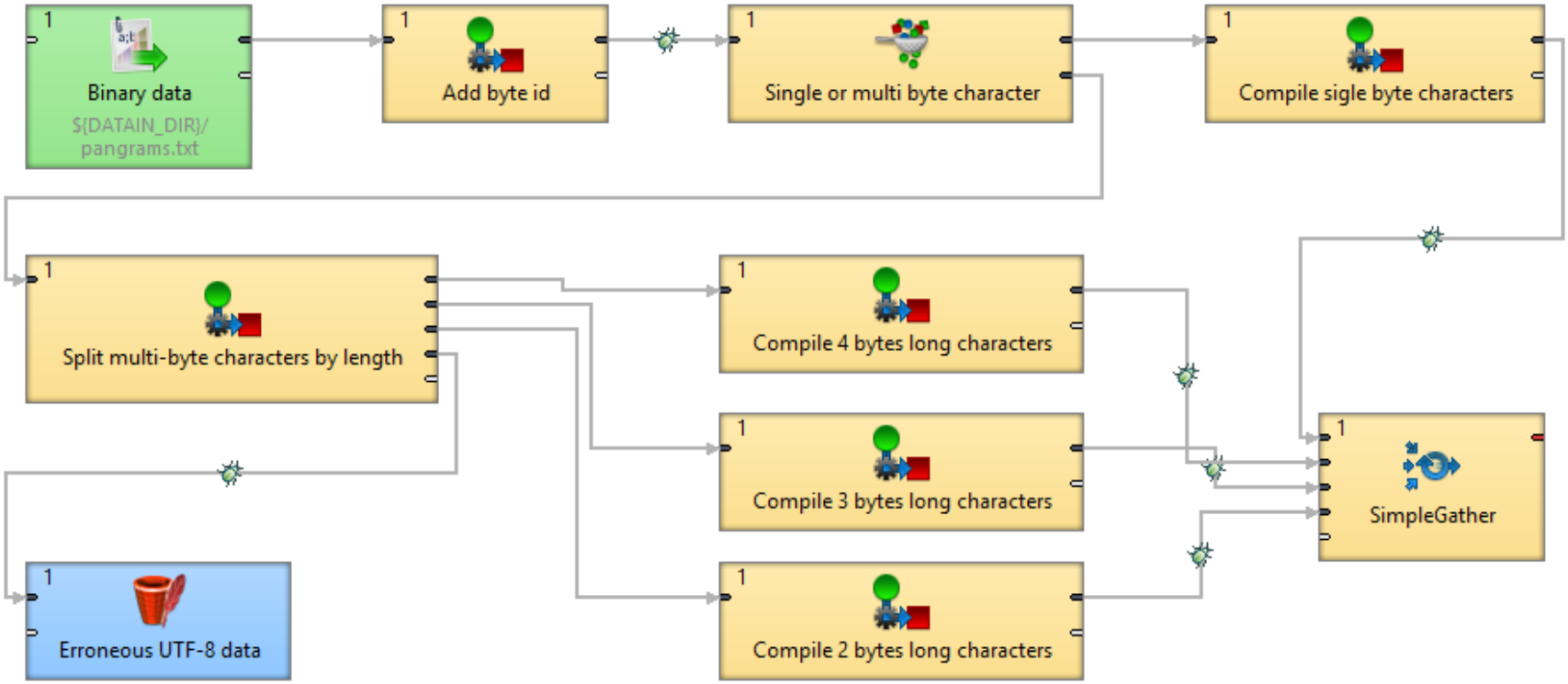

Edit external metadata
Modifications of the metadata will be stored in the metadata configuration file
External metadata: rse://clover-virt-november-9080/porchInitialLoad/meta/channel_government/Ohio/oh_license_roster_table.fmt
Show whitespace chars
Name Type Delimiter Label
1 Record: oh_license_roster... delimited | oh_license_roster_table
1 FormattedCredential string | FormattedCredential
2 FirstName string | FirstName
3 LastName string | LastName
4 License_Type string | License Type
5 Status string | Status
6 Address1 string | Address1
7 Address2 string | Address2
8 City string | City
9 State string | State
10 Mail_Zip integer | Mail Zip
11 County string | County
12 Telephone string | Telephone
13 Fax string | Fax
14 Effective_Date date | Effective Date
15 Expiration_Date date \n Expiration Date
Filter:
Field: oh_license_roster_table
Property Value
Basic
Name oh_license_roster_table
Label oh_license_roster_table
Type delimited
Record delimiter \n
Default delimiter |
Skip source rows 0
Description
Advanced
Quoted strings false
Quote character (both)
Locale en.US
Locale sensitivity case_sensitivity
Null value
Selected field is valid
Attached preview: (none)
ISO-8859-1
Browse...
Remove
Lock
OK
Cancel

Čtení dat z databáze

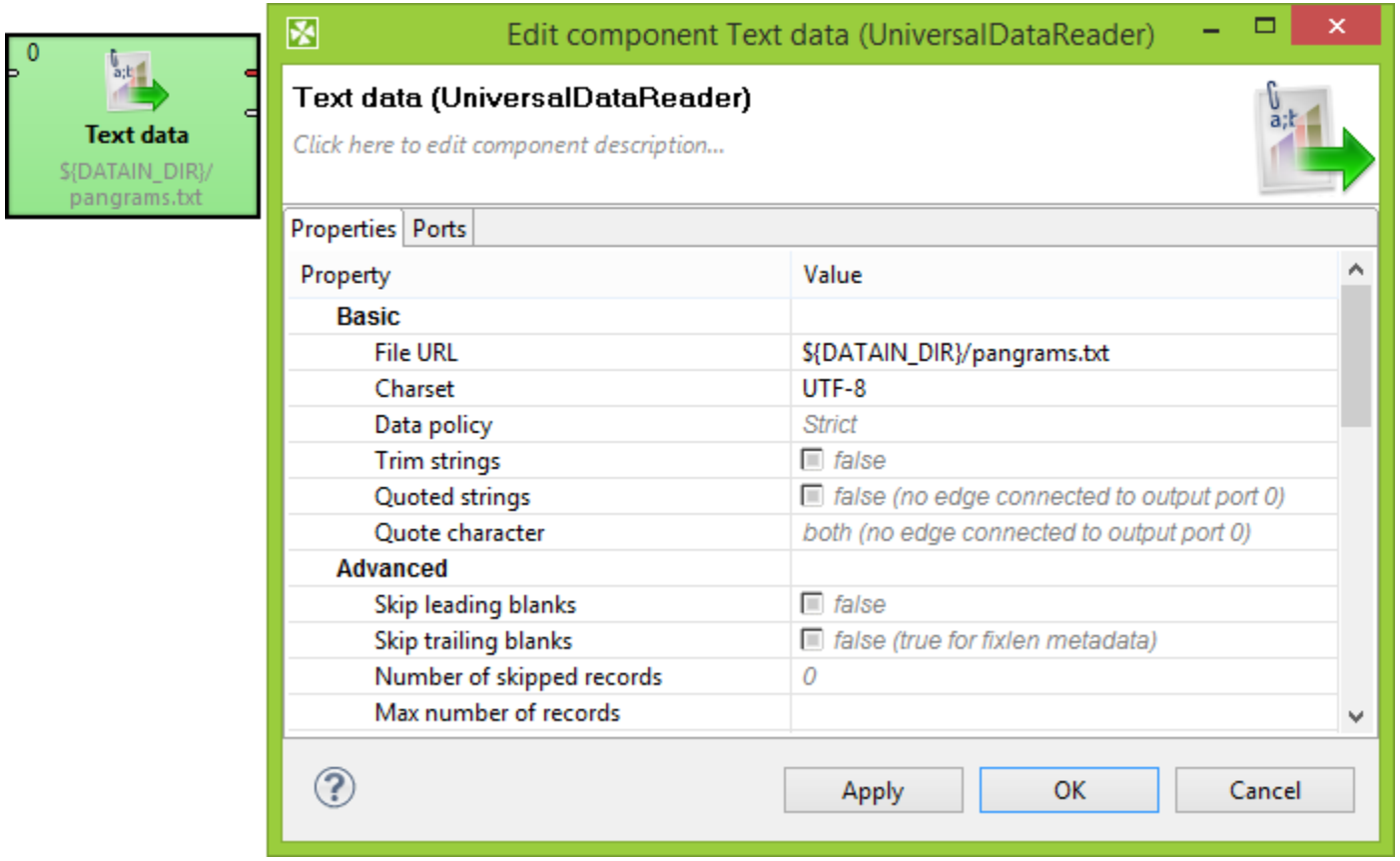



```
SELECT DISTINCT
ACLASS, ADESC,
CAST(TRIM(CONCAT(CONCAT(TRIM(VCVLVL),' '),TRIM(VCDSVC)))
                                        AS CHAR(30) CCSID 870) AS BUSSCAT,
CSADD1, CSADD2, CSADD3, CSADD4, CSCID1, CSCONT, CSCRGN, XEMAIL, CSHFAX,
CSHNME, CSHTEL, CSCHRS,  CSNAM, CSPCDE, CSSTYP, CSWFAX, CSWTEL,
IBATCH AS DOBA_SML,
ECCOLR, ECCTYP, ECDOFR, ECDTIM, ECENID, ECWGHT, ECYOFR,
CASE WHEN SJTYPE IS NULL
  THEN CAST(' ' AS CHAR(3) CCSID 870)
  ELSE  SJTYPE
END AS GARANCE,
HADD4, HDLID1, HNAME,
IASETG, ICASHV, IDEAL, ILSTYP, IMKREF, IPROD, IRESVL, ISERL, IVARSV,
JACCT, JNAME, JSORT,
CASE WHEN (RGCTYP = 'UFO') OR (RGCTYP = '') OR (RGCTYP IS NULL)
  THEN ECCLTP
  ELSE RGCTYP
END AS KLIENTTYP,  KMDESC,
IEXDAT AS KONSML,
IDFUNC AS POC_SML, 'N' AS PREDNOST,
ONAME AS PRODEJCE,
READD1, READD2, READD3, READD4, REPOST,
ECVATP AS VATPAYER,
VCAREQ, VCFUEL, VCHSNO, VNECAP, VNMOD, VNPWR, VPSTA3, VPTDOC, VREGNO,
VYRCON
FROM &LIB/ICMSTB AS MS
LEFT JOIN &LIB/ICSRVA AS SX ON ISERL = SXSERL
LEFT JOIN &LIB/ICPROD AS AP ON IPROD = APROD
LEFT JOIN &LIB/ICCUST AS CS ON ICUST = CSNUM
LEFT JOIN &LIB/ICVLCD AS VL ON CSBCAT = VCVLVL AND VCFDNM = 'AXBCAT'
LEFT JOIN &LIB/ICSAP3 AS EC ON ISERL = ECSERL
LEFT JOIN &LIB/ICDEAL AS H  ON IDEAL = HDEAL
LEFT JOIN &LIB/ICBANK AS J  ON ISERL = JSERL
LEFT JOIN &LIB/ICCREF AS RG ON ISERL = RGSERL
LEFT JOIN &LIB/ICSLMN AS O  ON ISALMN= OSALNO
LEFT JOIN &LIB/ICRESI AS RE ON ICUST = RECUST
LEFT JOIN &LIB/ICPRVL AS VP ON ISERL = VPSERL
LEFT JOIN &LIB/ICMRKT AS K  ON IMKREF= KMREF
LEFT JOIN (SELECT SJSERL, SJTYPE, SJROLE
           FROM &LIB/ICSUBJ AS SJT
```

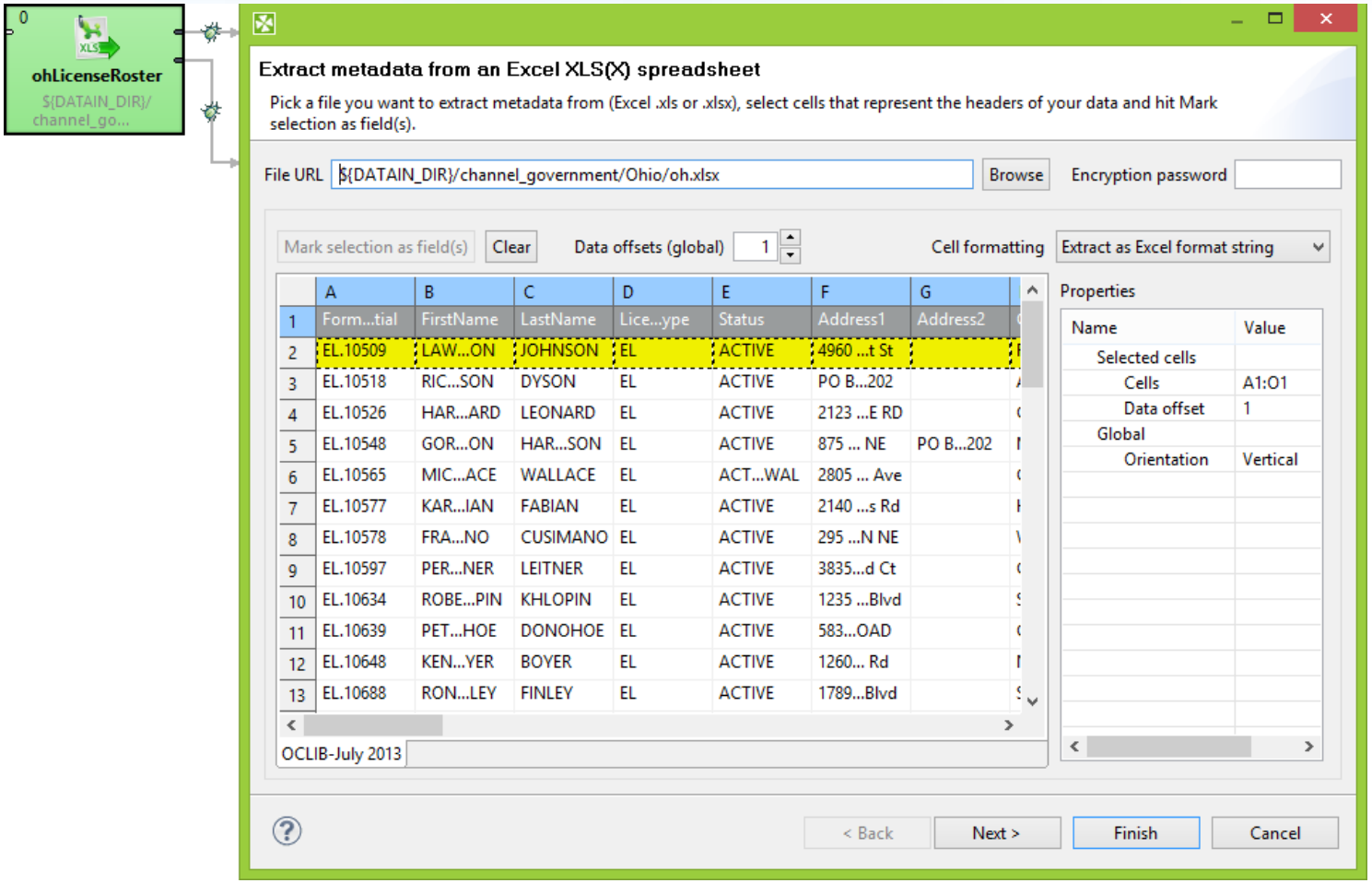
Čtení dat z binárních souborů



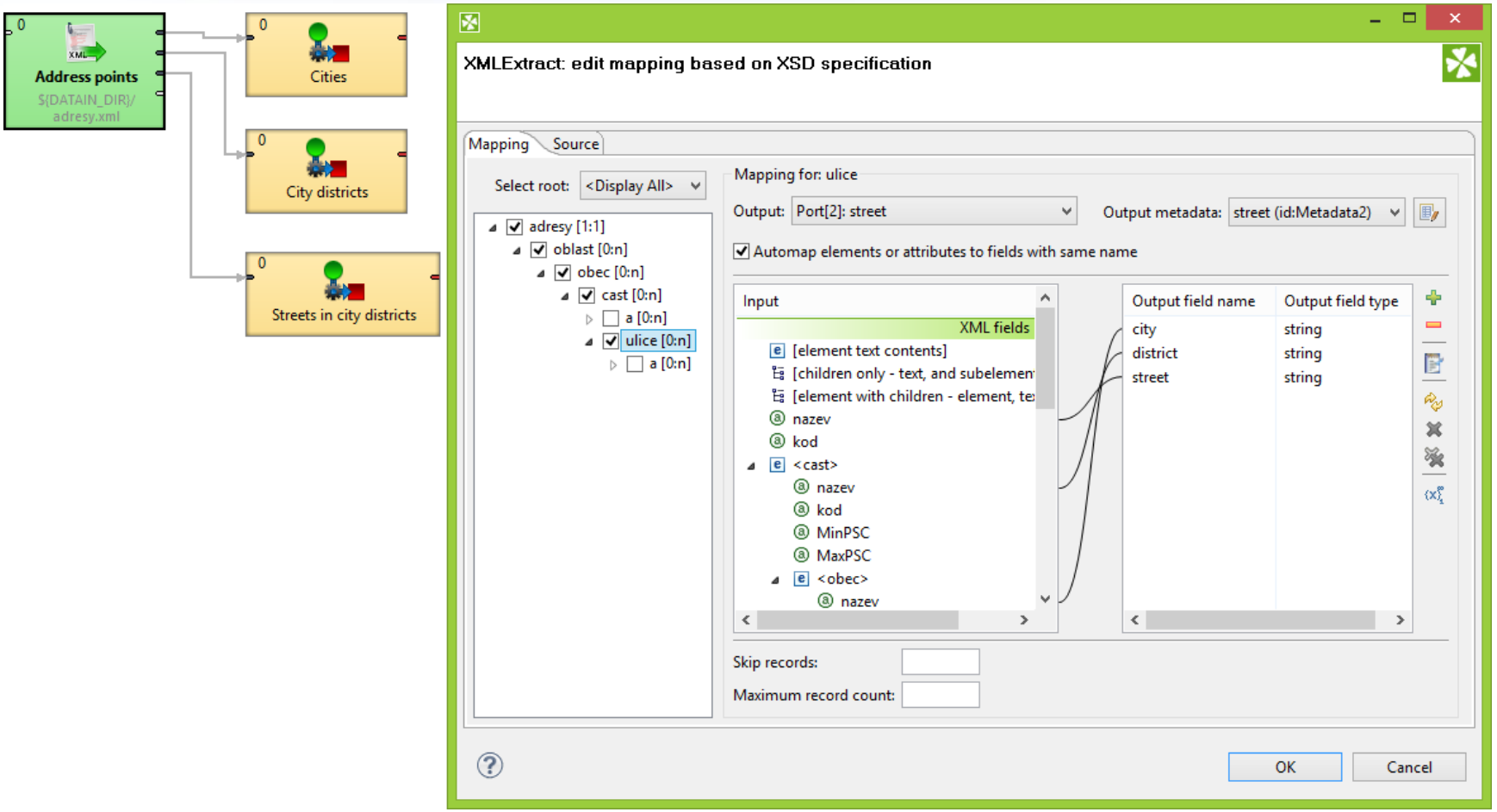
Čtení dat z binárních souborů



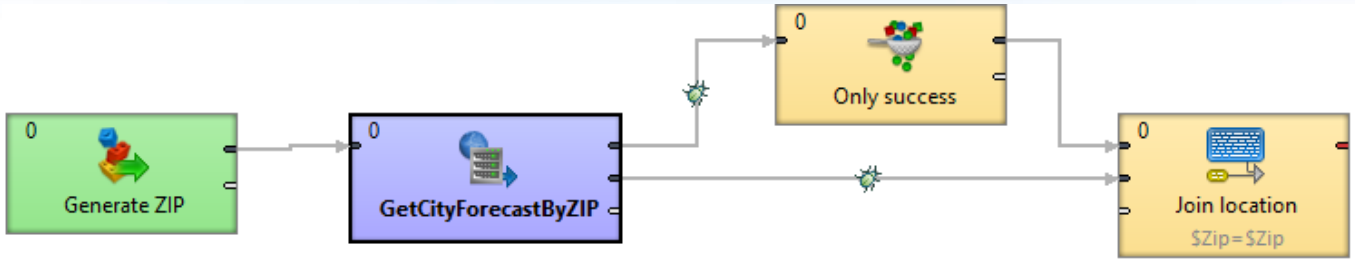
Čtení dat ze spreadsheet souborů



Čtení dat z XML struktur



Čtení dat z web service



Edit component GetCityForecastByZIP (WebServiceClient)

GetCityForecastByZIP (WebServiceClient)

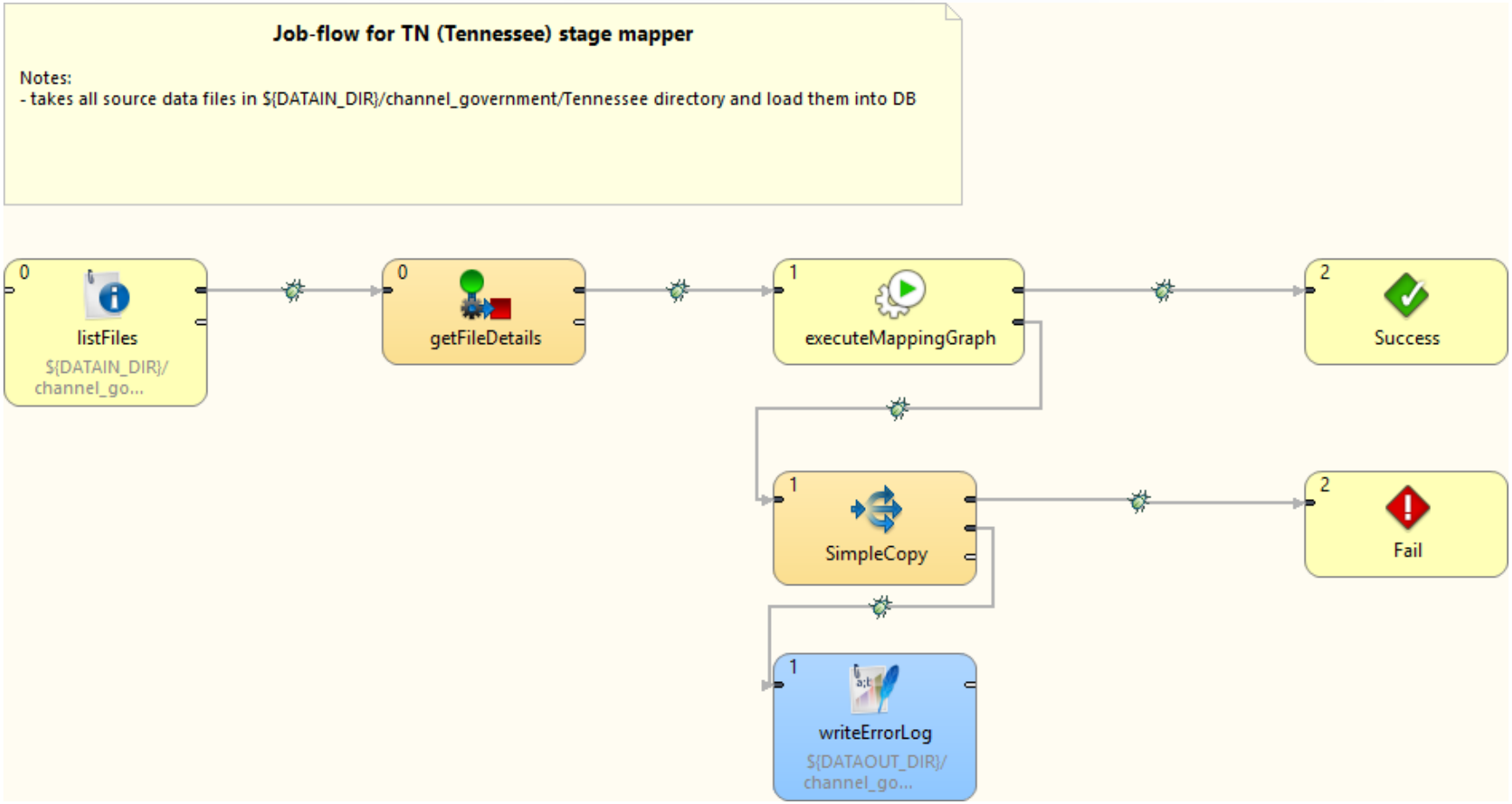
Click here to edit component description...

Properties | Ports

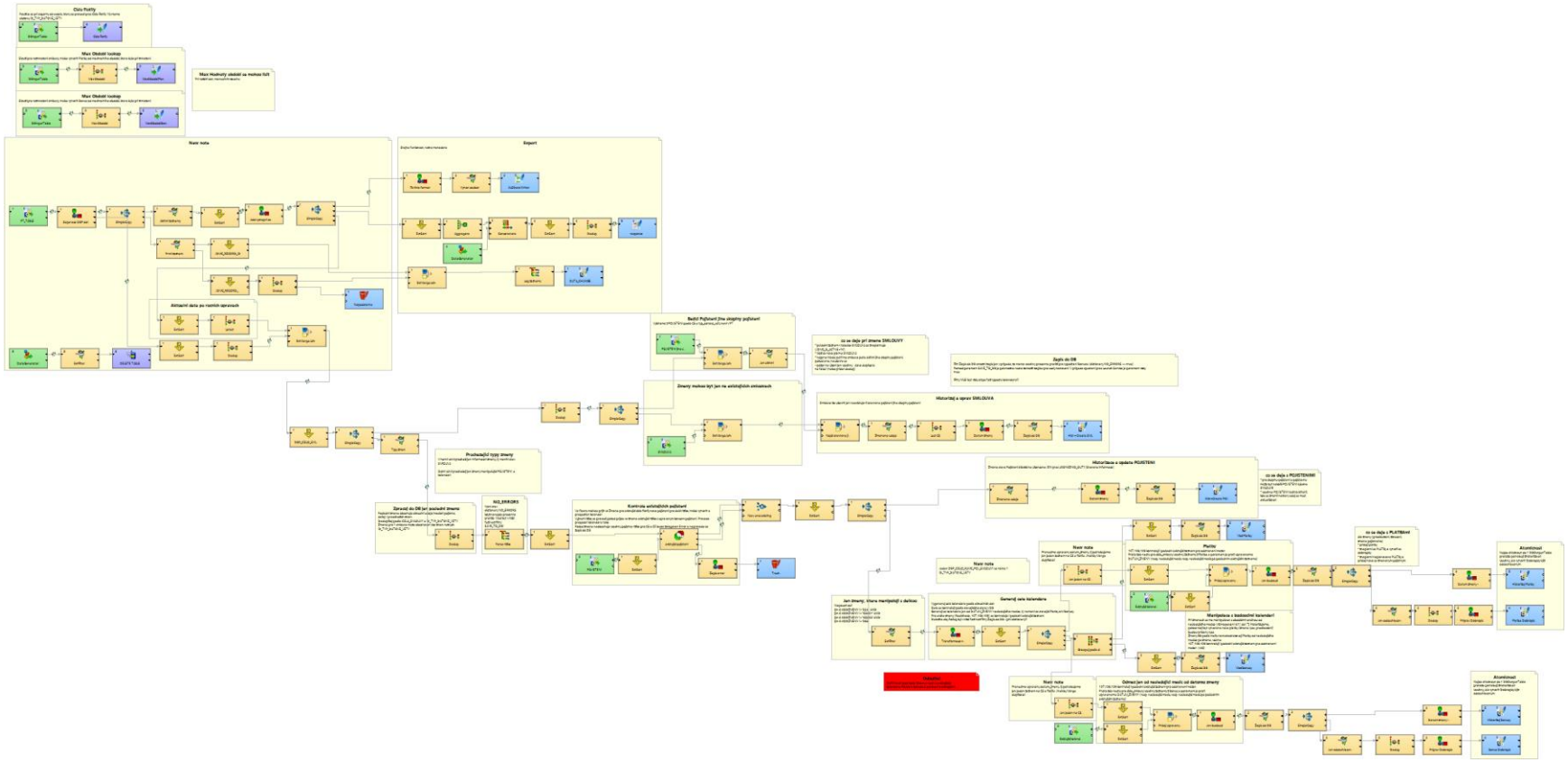
Property	Value
Basic	
WSDL URL	http://wsf.cdyne.com/WeatherWS/Weather.asmx?wsdl
Operation name	{http://ws.cdyne.com/WeatherWS/}Weather#WeatherSoap12#GetCityForecastByZIP
Request Body structure	<weat:GetCityForecastByZIP xmlns:weat="http://ws.cdyne.com/WeatherWS/"> <weat:ZIP>$zipCo...
Request Header structure	
Response mapping	<Mappings><Mapping element="Weat:GetCityForecastByZIPResponse"><Mapping element="We...
Fault mapping	
Namespace Bindings	{XMLS=http://www.w3.org/2001/XMLSchema, incl=http://www.w3.org/2004/08/xop/include, Weat...
Use nested nodes	*true*

Apply | OK | Cancel

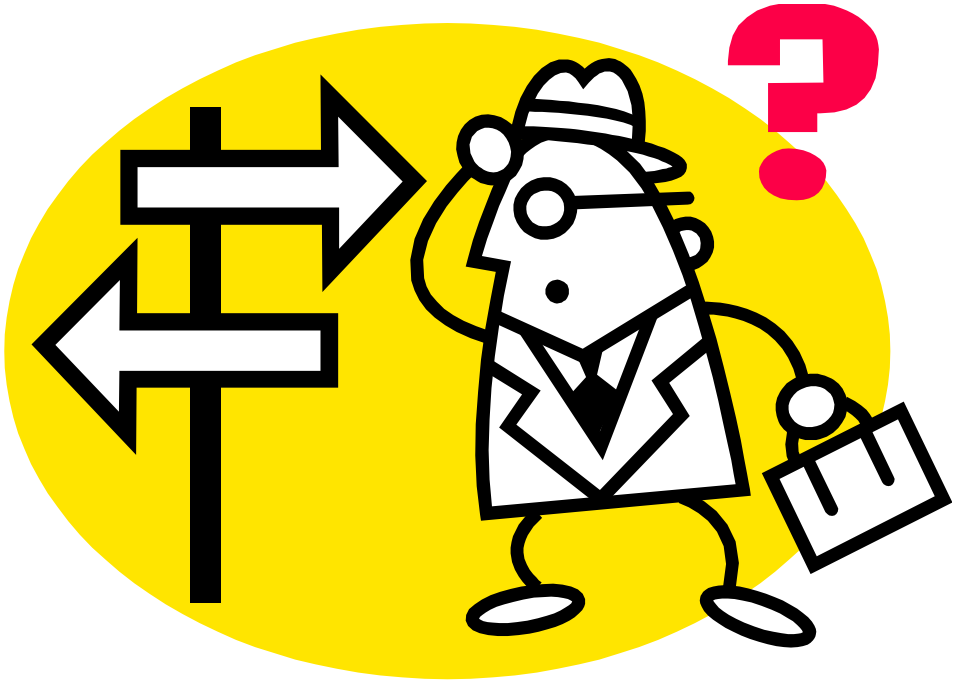
Závěrem...


Job-flow for TN (Tennessee) stage mapper
Notes:
- takes all source data files in ${DATAIN_DIR}/channel_government/Tennessee directory and load them into DB
0
listFiles
${DATAIN_DIR}/channel_go...
0
getFileDetails
1
executeMappingGraph
2
Success
1
SimpleCopy
2
Fail
1
writeErrorLog
${DATAOUT_DIR}/channel_go...

Závěrem...



Diskuze



Děkujeme za pozornost