

|  |
| --- |
| Business Template  **Retailrocket recommender system dataset**  ***Ecommerce data*** |
|  |

Contents

[1 Business Description 3](#_Toc22660618)

[1.1 Business background 3](#_Toc22660619)

[1.2 Problems because of poor data management 3](#_Toc22660620)

[1.3 Benefits from implementing a Data Warehouse 3](#_Toc22660621)

[2 Dimensions of a Business 3](#_Toc22660622)

[3 Logical Scheme 3](#_Toc22660623)

[4 Data Flow 3](#_Toc22660624)

[5 Fact Table Partitioning Strategy 3](#_Toc22660625)

[6 Strategy of Parallel Load 3](#_Toc22660626)

[7 Report Layouts 3](#_Toc22660627)

# 

# Business Description

## Business background

Retail business segment generates significant amount of revenue for each of it’s agents. It is a competitive, well-organized and trend-sensitive environment which uses all kinds of trading channels all across the world, variety of commerce tools and focuses on client needs. To satisfy market demand, organize supply management system and keep strong position among competitors, it is a necessity to obtain a system which will help orient in such environment by managing sales information and provide with an appropriate comprehensive toolkit any analyzing purposes.

## Problems because of poor data management

Possible issues which may occur due to this problem can impact the business in a bad way and if not instantly but in a long run for sure. Among these problems are:

* ineffective warehouse management, when our shelves can be flooded with a non-relevant products, so the supply system is not synchronized with the market demand
* ineffective seasonality handling, when business can’t determine the profitable time interval for a particular product categories
* poor recommender-system performance, so the business doesn’t know who likes it’s products more than others
* etc.

Ignoring such cases and staying without appropriate tools will decrease organization’s revenue, force it to reduce it’s operating power and lose competitive position on the market.

## Benefits from implementing a Data Warehouse

Using of data warehouse can help you with the problems described above and many others. Implementing a data warehouse can answer you the following questions:

* Which product categories generate the largest revenue.
* What are the most profitable trading channels?
* Is there a typical price distribution (e.g., normal) across brands or within specific brands?
* What are the best trading periods for each particular product category through regions?
* How the demographical picture looks like for different product categories?
* Etc.

Further processing data would also let you:

* Correlate specific product features with changes in price.
* If there are any differences between women's brands and men's brands.
* Obtain users’ daily/weekly/etc. activity dynamics to segregate appropriate time intervals.
* Segregate customers into loyalty categories.
* Develop marketing strategies for each operating region.
* And many other.

# Dimensions of a Business

**STEP 1: SELECT THE BUSINESS PROCESS**

Conducting data sets’ structure analysis, the following statement can be formulated: the **business process** which takes place here is **tracking users’ activity** across online and offline organizations’ retail marketplaces. This process matches the following criteria of the considered definition:

* Action is performed frequently. There are more than 2M records of such events in data set.
* Event was selected as primary one for tracking purposes. That means business chose this activity as a most descriptive one for its’ particular purposes.
* Each event is introduced with a specific measurement such as ‘amount’ and ‘total price’
* Each event is complemented with a corresponding timestamp, so it is a good candidate to further decomposition and grain definition.

**STEP 2: DECLARE THE GRAIN**

As was mentioned above, the business process represents the users’ activity tracking routine, but in its initial state it requires decomposition into an atomic events, so the grain could be completely defined as well as fact table configuration could be established.

A unique event which can characterize the grain can be described with the following features:

* **Event date.** The date activity was performed
* **Product.** An item client had interacted with.
* **Store.** Offline marketplace unit where transaction had occurred.
* **Customer.** A person who performed an action.
* **Channel.** Sales channel via which event was handled.
* **Event location.** Location where event took place, for example in case event was occurred in a store, so store location is chosen and if event was performed on the internet platform, customers’ location takes place here.
* **Employee member.** A staff member who served a client in case event took place in a store.
* **Event type.** Type of the event itself that took place.

All these attributes can define an atomic event for the grain, which in this particular case is – **users’ interaction with a unique item in stock.**

**STEP 3: IDENTIFY THE DIMENSIONS**

The dimensions serve as a context complementary information for each fact table row. Their state depends on the chosen dimensional modeling approach and vary in redundancy level (normalization) in star or snowflake schema. Generally speaking of this particular example, the dimensions which will provide each event with context are the following:

* **DIM\_DATE**

This is the calendar date dimension with the granularity of a single day. This dimension must be specified at the beginning of the data warehouse project. Dimension realize several hierarchical structures which can be described in the following way:

* Calendar hierarchy: Day (date) ← Calendar week ← Calendar month ← Calendar year
* Fiscal hierarchy: Day (date) ← Fiscal week ← Fiscal month ← Fiscal year

Attributes of significant meaning:

**PK & Natural key:**

*Date\_ID* – attribute has the value date of the date type.

Notice!

This dimension must have one record that handles the special non applicable date situation where the recorded date is inapplicable, corrupted, or hasn’t happened yet.

* **DIM\_CHANNELS**

Dimension stores information about sales channels through which events could be handled. This is the **SCD Type 1 without hierarchies** dimension.

**Dimension grain:** a single unique channel

**PK:** CHANNEL\_SURR\_ID – surrogate DWH key

**Natural key:** CHANNEL\_ID – natural source id for channel

* **DIM\_STORE\_SCD**

Dimension stores descriptive information about stores in which transactions are handled. This is the **SCD Type 2 with hierarchies** for store location and manager location, this type of hierarchy is described in locations dimension.

**Dimension grain:** a single unique store

**PK:** STORE\_SURR\_ID – surrogate DWH key

**Natural key:** STORE\_ID – natural source id for store

* **DIM\_EVENT**

Dimension stores descriptive information about all kind of events that could be registered and identified by the tracking system. This is a **SCD Type 1 without hierarchies** dimension.

**Dimension grain:** a single unique event type

**PK:** EVENT\_SURR\_ID – surrogate DWH key

**Natural key:** EVENT\_ID – natural source id for store

* **DIM\_CUSTOMER\_SCD**

Dimension stores descriptive information about all the customers who are registered in the system. This is the **SCD Type 2 with hierarchy** for personal customer location, this type of hierarchy is described in locations dimension.

**Dimension grain:** a single unique customer

**PK:** CUSTOMER\_SURR\_ID – surrogate DWH key

**Natural key:** CUSTOMER\_ID – natural source id for store

* **DIM\_EMPLOYEE\_SCD**

Dimension stores descriptive information about all the staff members who are registered in the system. This is the **SCD Type 2 with hierarchy** for personal employee location, this type of hierarchy is described in locations dimension.

**Dimension grain:** a single unique employee

**PK:** EMPLOYEE\_SURR\_ID – surrogate DWH key

**Natural key:** EMPLOYEE\_ID – natural source id for store

* **DIM\_LOCATION**

Dimension that provides with addresses for such entities as stores, customers, employees and events occurrences. This is the **SCD Type 1 with hierarchy**.

* Location hierarchy: full unique address ← district ← city ← country ←country subregion ← country region

**Dimension grain:** unique object/event location

**PK:** LOCATION\_SURR\_ID – surrogate DWH key

**Natural key:** LOCATION\_ID – natural source id for store

* **DIM\_PRODUCT\_SCD**

Dimension contains information about all the products registered in the system. This is the **SCD Type 2 with hierarchy** for product categories.

* Product hierarchy: unique item ← product subcategory ← product category

**Dimension grain:** unique product item

**PK:** PRODUCT\_SURR\_ID – surrogate DWH key

**Natural key:** PRODUCT\_ID – natural source id for store

* **SCD TYPE 1 DIMENSIONS**

For this particular type of dimensions, the following attributes where added:

**INSERT\_DT** – date where a record was introduced in the system

**UPDATE\_DT** – date where a record experienced changes in its attributes

This kind of changing technique doesn’t allow historical tracking, but it is implemented only in dimensions which don’t require such features.

* **SCD TYPE 2 DIMENSIONS**

For this particular type of dimensions, the following attributes where added:

**START\_DT** – date where the attribute value initially took place

**END\_DT** – date where the attribute value validity expires. Initially could be initialized by a non applicable date (e.g. 12/31/9999) when there is no information provided for validity expiration (NULL values prevention)

**IS\_ACTIVE** – flag which indicates the validity of the certain row

**CHANGE\_REASON** – provides with a brief description for changed attributes

**INSERT\_DT** - date where a record was introduced in the system

**UPDATE\_DT** – date where a record experienced changes in its attributes (flag status change in non-valid row situation)

Notes!

1. Each dimension with the hierarchy is performed in its denormalized state to satisfy STAR-schema requirements.
2. Location attributes which take place in dimensions other than DIM\_LOCATION are referenced to this particular objects’ private locations. Transaction location ID in fact table indicates only the event location. E.g. customer can perform an action while being outside of his private location (other city which differs from his private house location), in case purchase was made in store, this attribute value will match the store location ID and so on.

**STEP 4: IDENTIFY THE FACTS**

Each row in a fact table consists of all the dimensional attributes mentioned in step 2 grain declaration plus the following:

**Degenerate attributes:**

* **Transaction\_ID** – a natural key, degenerate dimension. Serves as a parent key for all the records (lines) in a transaction bill. Might be useful for analytics purposes

**Measurements:**

* **Unit\_cost** – cost of unit at the moment of the event performance
* **Amount** – amount of units took place in event
* **Sale\_amount** – overall price customer paid for this particular item/items

**Technical fields:**

* **INSERT\_DT** – insertion date
* **UPDATE\_DT** – date the row experienced an eventual update (can be populated with non-informative date for obscure situations)

This set of attributes will fully define the atomic event which was declared in section 2 grain definition.

A picture containing diagram

Description automatically generated

**3NF modelling**

Normalized layer was created based on the previous STAR dimensional model. The whole process can be described with the following steps:

**1) Manage hierarchy within dimension tables.**

Initially deformalized tables were decomposed into separate ones and each separate table relates to each particular hierarchy level. For example table DIM\_PRODUCTS\_SCD was decomposed into tables DIM\_PRODUCT\_SUBCATEGORY and DIM\_PRODUCTS\_CATEGORY.

Each of the hierarchy level higher then the lowest one will perform **SCD Type 1** versioning pattern.

**2) Manage versioning.**

In case of **SCD Type 1** tables each one was complemented with such technical attributes as **INSERT\_DT** and **UPDATE\_DT**. Purpose of these attributes was clarified in the previous section.

In case of **SCD Type 2** tables each one was complemented with such technical attributes as **INSERT\_DT**, **UPDATE\_DT, START\_DT, END\_DT, CHANGE\_REASON, IS\_ACTIVE**. These attributes were added only to the table with the lowest hierarchy level.

**3) SCD Type 2 referential integrity.**

In case dimensional table implement this versioning logic, Its Primary Key has composed state. Here **composite Primary Key** consists of the two attributes: **PK(<name>\_ID, START\_DT)**. In that case there is not physical references from fact table to dimensional which implements SCD Type 2 versioning, but it implies only ‘logical’ referential integrity which will obtain its physical form in denormalized state via ETL tools further.

The attribute which ‘logically’ refers to such table still persist in fact table, but without physical connection.

**4) Date dimension.**

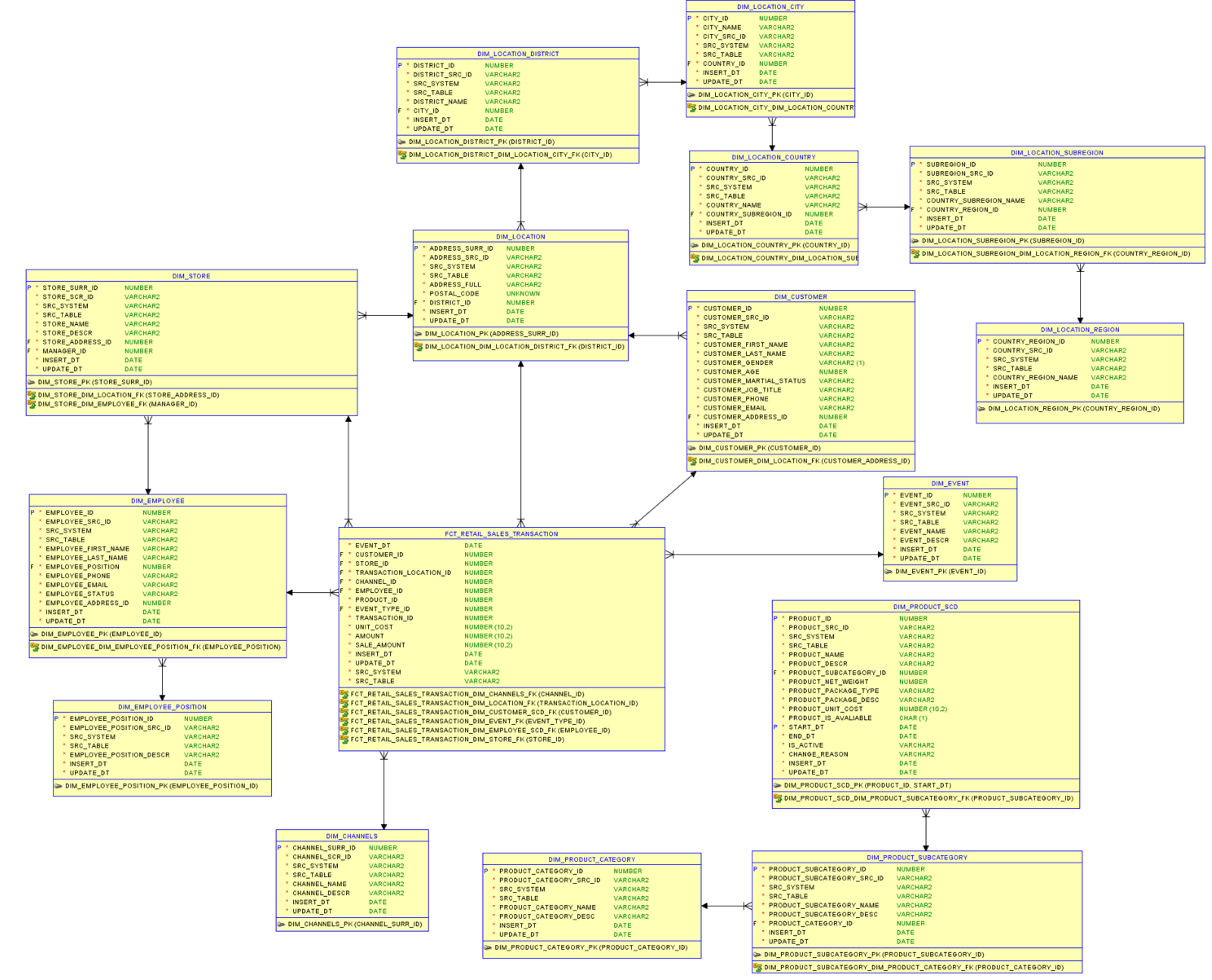
As far as DATE is very special dimension, it would not be populated from the normalized layer. Instead it would be populated once in a lifetime. So as in case of SCD Type 2 its referential persist in fact table, but this dimension will only appear in denormalized layer.

**5) Source systems**

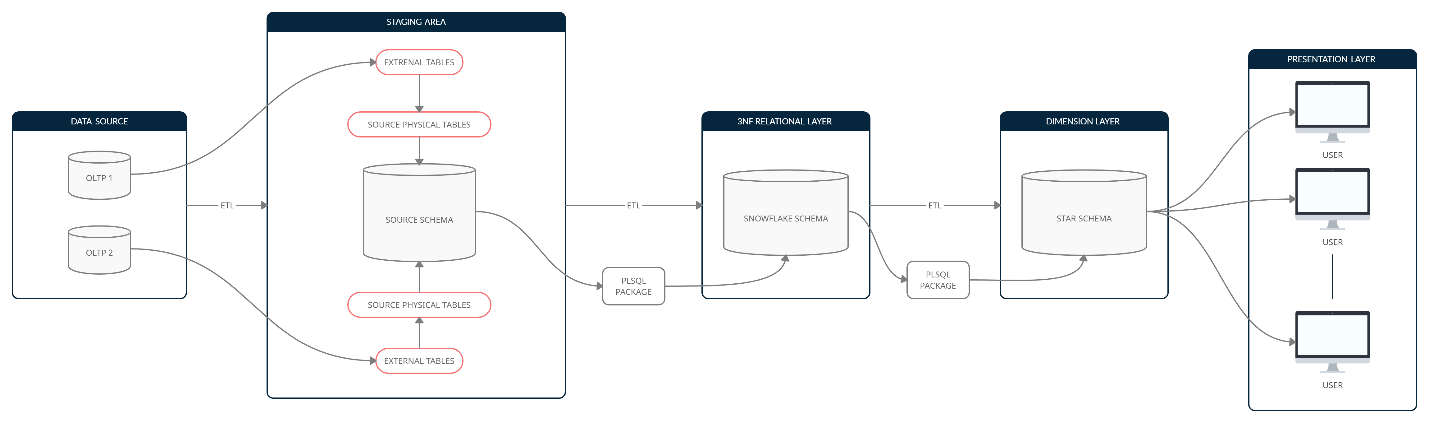
Each table is complemented with the attributes which provide with *source item ID, source system (OLTP), source table* (<name>\_SRC\_ID, SRC\_SYSTEM, SRC\_TABLE correspondingly).

Only Fact table here possesses 2 of these attributes SRC\_SYSTEM, SRC\_TABLE which can be interpreted as the billing system provided with the transaction and concrete table from which transaction has been extracted.

**3nf model**



# Logical Scheme

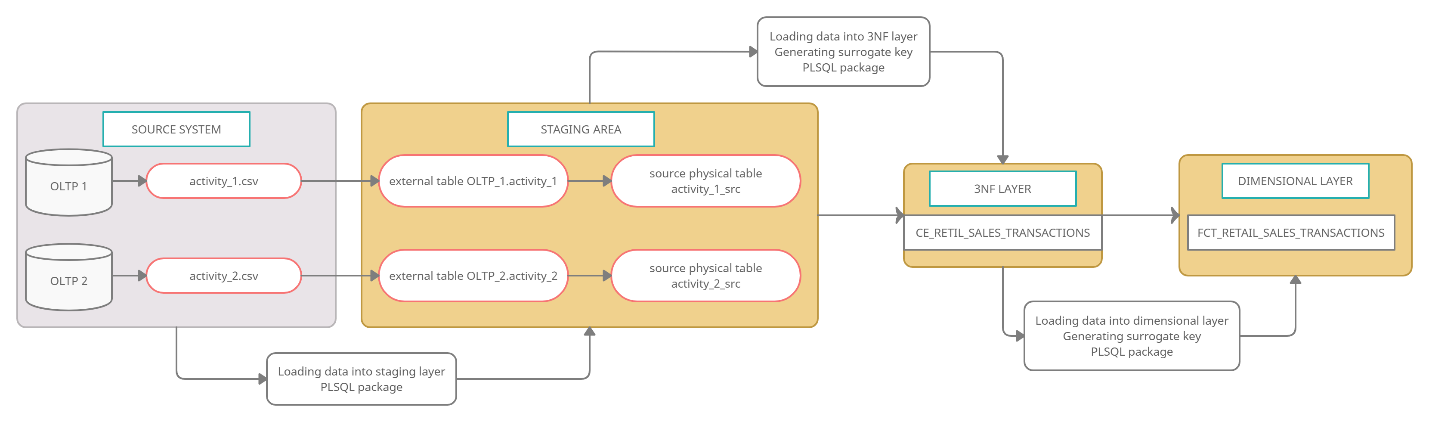


There are two OLTP data sources which provides DWH with the transactional data. The overall process of data load routine is the following:

1. Each OLTP base provides with the flat files of .csv format for each particular business entity. A separate user should be created for each corresponding external source.
2. External tables are created for each flat file from OLTP establishing this way a robust connection DWH and external sources. These tables are assigned to corresponding users (OLTP systems) spaces.
3. Physical tables are created for each external in relation 1:1, forming this way physical source schema with all the necessary data, cleaned and processed with the help of ETL tools.
4. From source schema table space, data flows to 3NF Snowflake Schema. It can be done by using PLSQL Oracle functionality and ETL toolkit.
5. After obtaining a fully developed and populated 3NF model, data is transported to Dimensional Star Schema fully denormalized. It can be done by using PLSQL Oracle functionality and ETL toolkit. This model represents final representative layer which can be accessed and used by 3rd party users for any analytical purposes.

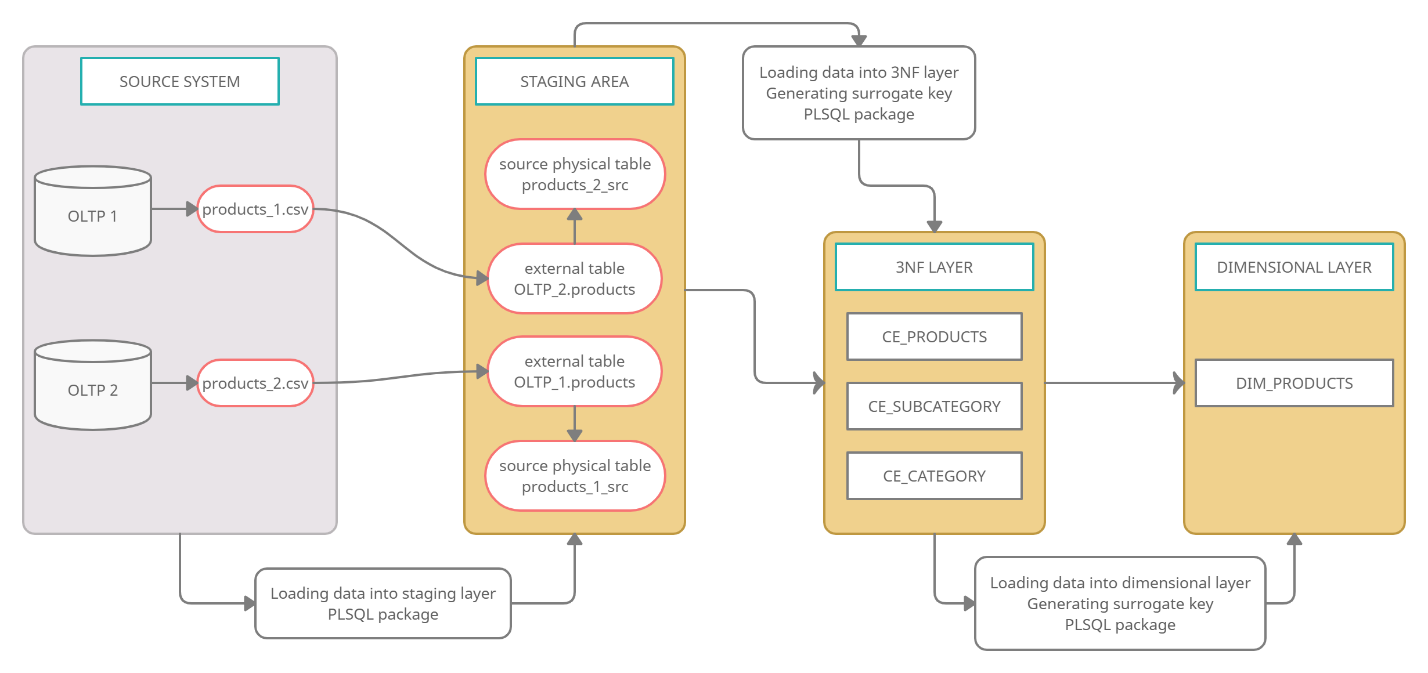
# Data Flow

**Fact table**

****

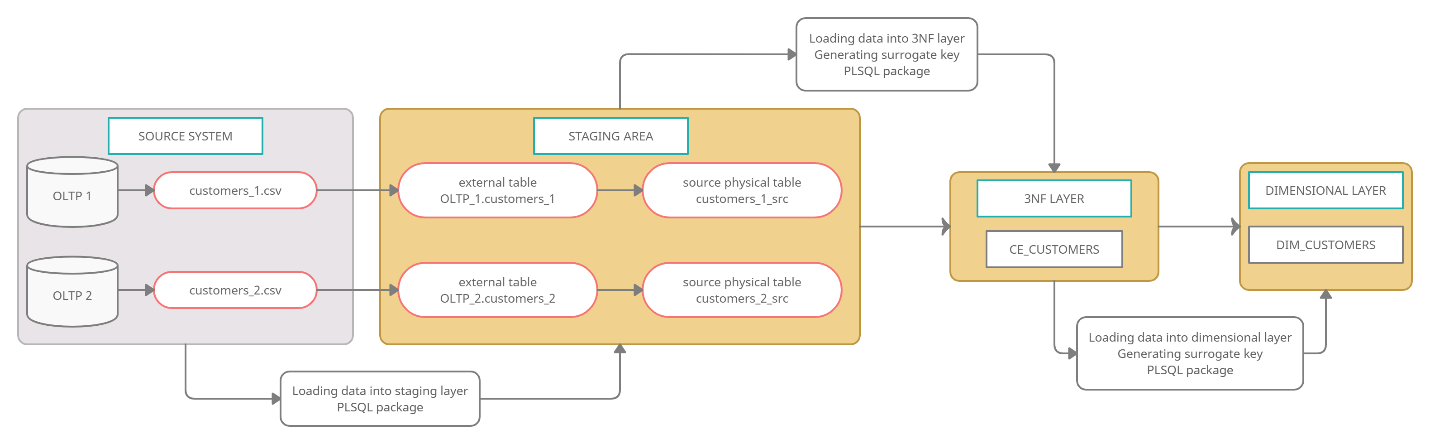
Transactional records come from two separate OLTP systems and are processed by the routine described in *LOGICAL SCHEME* section. Here comes one flat file from each external system and then extraction procedure is applied to each of them as well as migration sequence from source physical schema up to dimensional layer.

**Products**



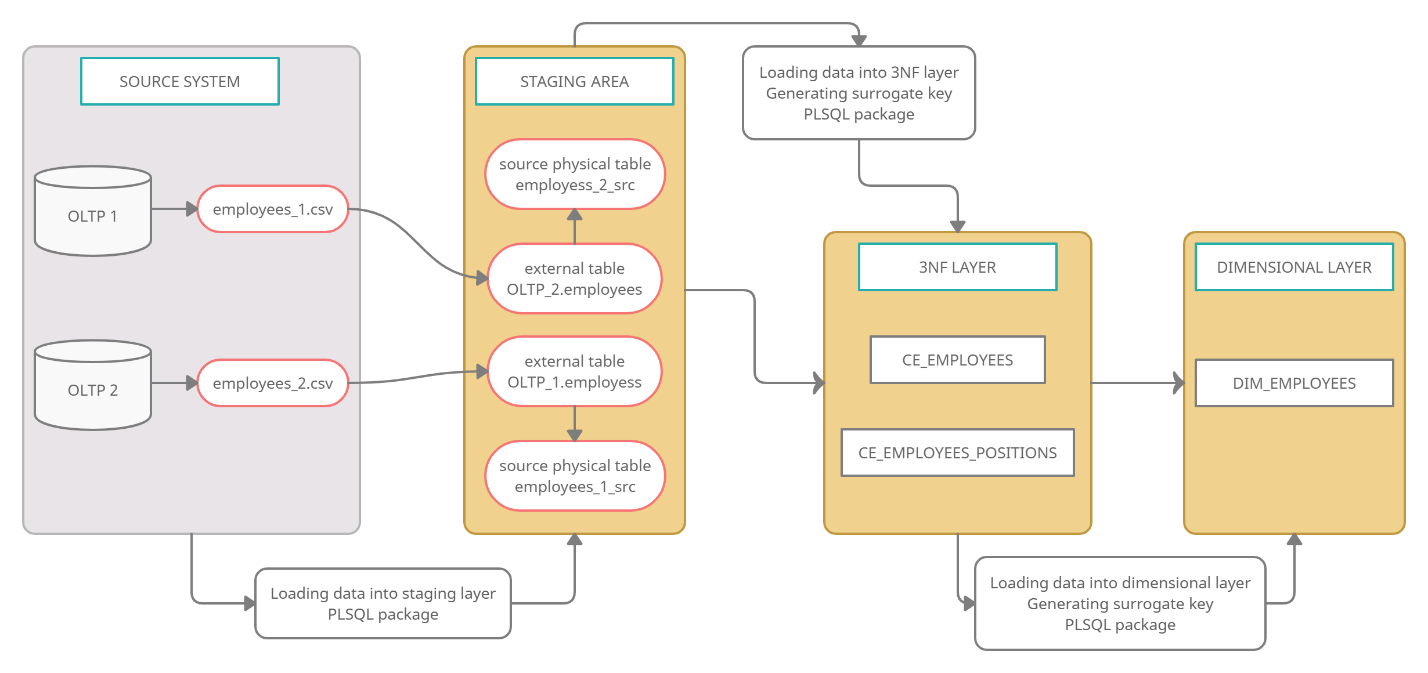
Information about products come from two separate OLTP systems and is processed by the routine described in *LOGICAL SCHEME* section. Here comes one flat file from each external system and then extraction procedure is applied to each of them as well as migration sequence from source physical schema up to dimensional layer. It worth noting that this is the business entity which will experience changes with time (SCD Type 2 table in both 3NF and dimensional layers) so logic here applies frequent flat file updating (by OLTP) and this is the part were external tables play crucial role in establishing robust connections between systems.

**Customers**



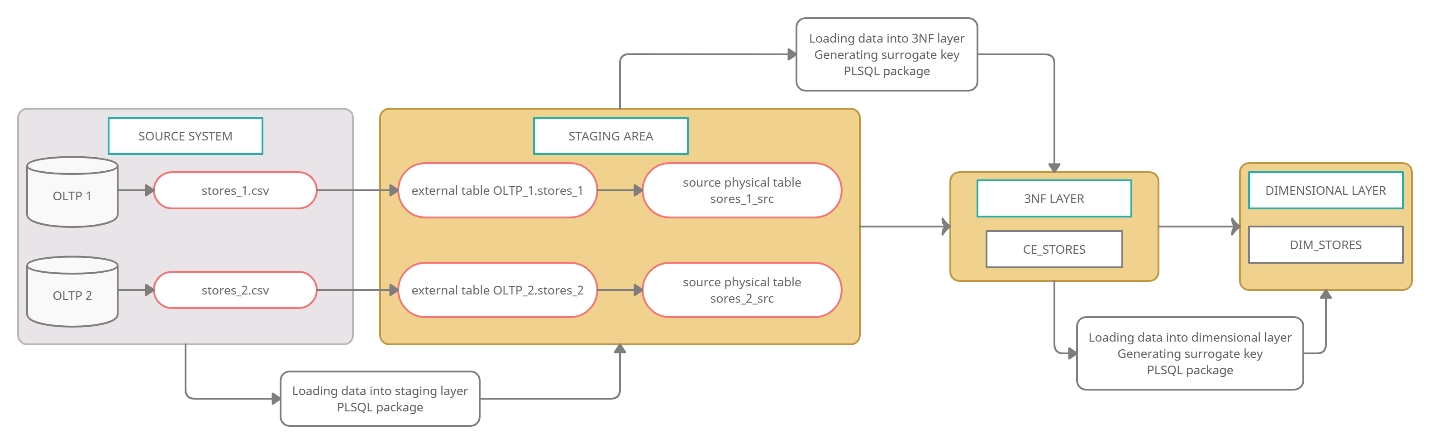
Information about customers come from two separate OLTP systems and is processed by the routine described in *LOGICAL SCHEME* section. Here comes one flat file from each external system and then extraction procedure is applied to each of them as well as migration sequence from source physical schema up to dimensional layer.

**Employees**



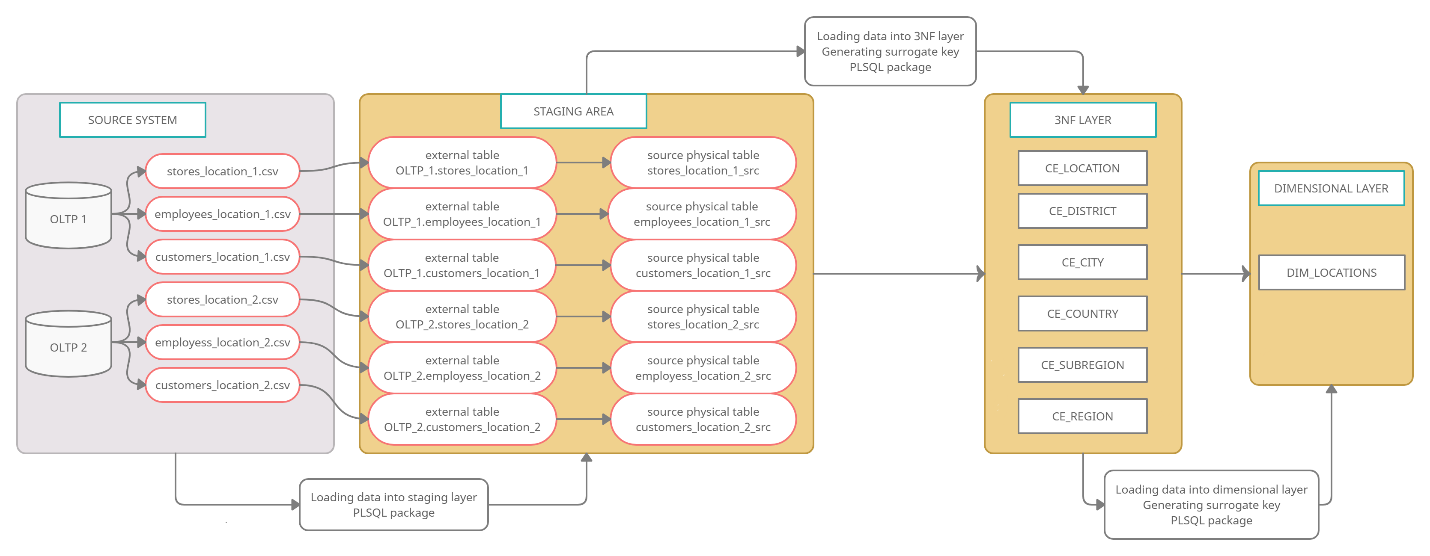
Information about employees come from two separate OLTP systems and is processed by the routine described in *LOGICAL SCHEME* section. Here comes one flat file from each external system and then extraction procedure is applied to each of them as well as migration sequence from source physical schema up to dimensional layer.

**Stores**

****

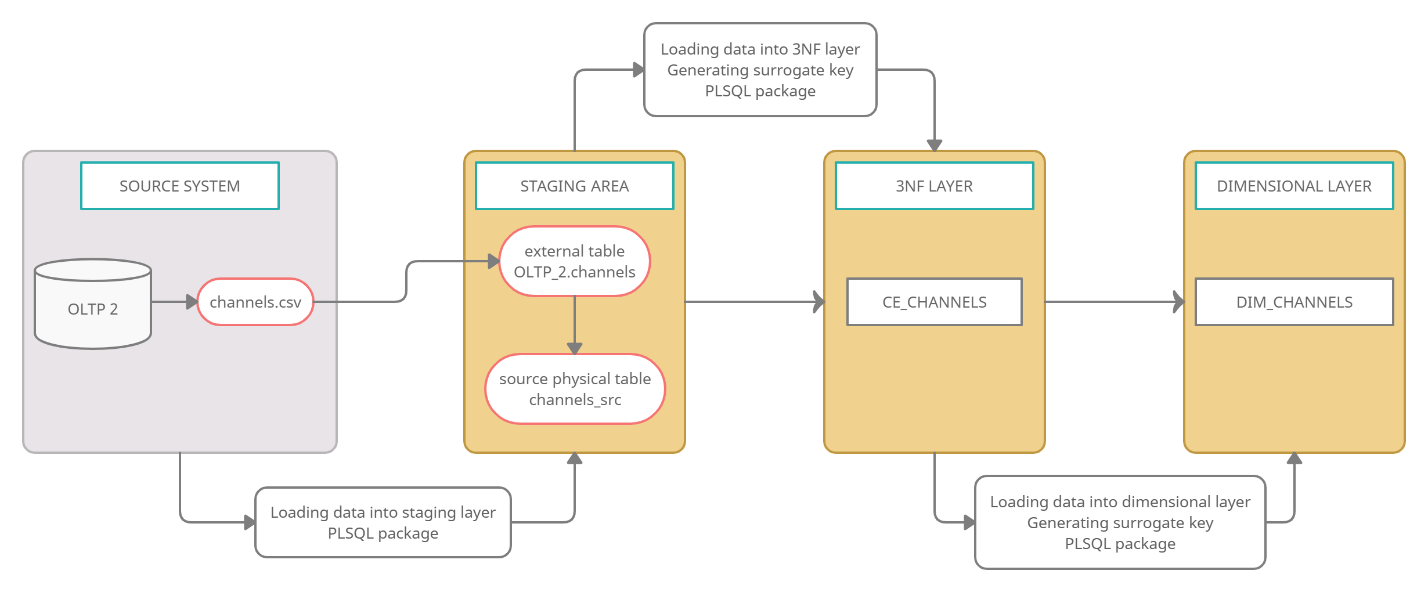
Information about stores come from two separate OLTP systems and is processed by the routine described in *LOGICAL SCHEME* section. Here comes one flat file from each external system and then extraction procedure is applied to each of them as well as migration sequence from source physical schema up to dimensional layer.

**Locations**

****

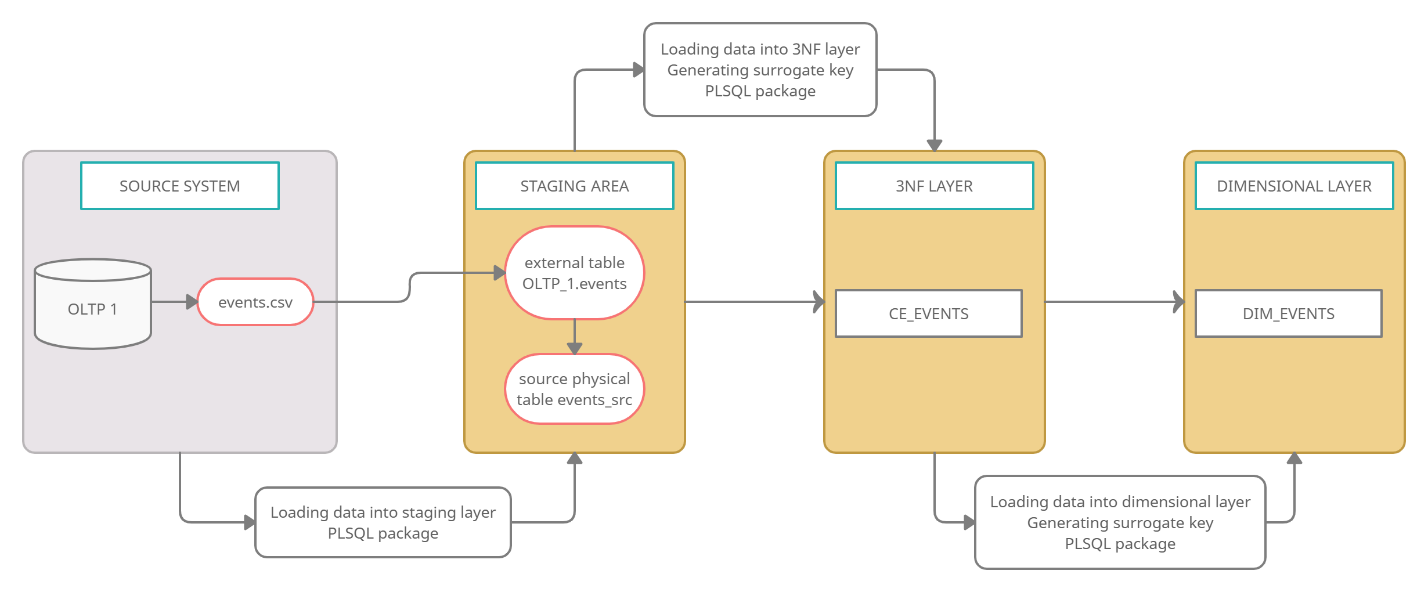
Here, information for locations is obtained the following way: two separate OLTP systems provides with a corresponding flat file with locations for each business inspected entity (each system generates it’s own set of such files) and then this set of files is processed by the routine described in *LOGICAL SCHEME* section. Extraction procedure is applied to each element of this set as well as migration sequence from source physical schema up to dimensional layer.

**Channels**

****

Channels – is a small table which describes each trading channel through which transactions are conducted, come from one OLTP system and is processed by the routine described in *LOGICAL SCHEME* section. Here comes one flat file from one external system and then extraction procedure is applied to it as well as migration sequence from source physical schema up to dimensional layer.

**Events**

****

Events – is a small table which describes each possible event which can be registered and classified, come from one OLTP system and is processed by the routine described in *LOGICAL SCHEME* section. Here comes one flat file from one external system and then extraction procedure is applied to it as well as migration sequence from source physical schema up to dimensional layer.

# Fact Table Partitioning Strategy

# Strategy of Parallel Load

# Report Layouts