**Anton Slizh’s Solution Concept**

**DWH for “Kvas Boosted” Kvas Drinks Retailer Company**

**Overview**

Our company “Kvas Boosted” is specialized on retail of kvas drinks. For more than 5 years our company has been engaged in sales and occupies a leading position in the market. The main geographic region of sales activity is the CIS due to our specific and narrow scope of product range. But, regardless of mentioned limitations we are growing really fast and regularly opening new stores.

The company has contracts with different suppliers from all over the world. The product list is presented both classic kvas types and specific tastes, depending of producer brands. Our customers are ordinary buyers with different income status.

Last year the company’s growth has slowed down. We want to find out the solution how to accelerate the company’s growth. Also, we want to understand the correlation of sales with different factors such as time, location, product, customers. The final solution should help us improve our sales policy and promote the company growth.

**Benefits**

The provided DWH can help business users in different ways:

1. It can help to better planning your business growth. According to well structured and represented data users can make important business decisions more accurately, what minimize risks of failure.
2. The data represented in DWH has high quality and structure. The data is strong standardized and don’t require special skills and knowledges to be accessed by ordinary business users.
3. The variety of gathered and processed data increase the range of business decisions users can make. It helps you to better understand all spheres and details of your business in real time. DWH solutions provide you the ability to prepare high quality reports, which covers all your business needs. Involving DWH system into your business significantly increase the understanding of processes which correlated with company growth and sales policy.

**Business requirements**

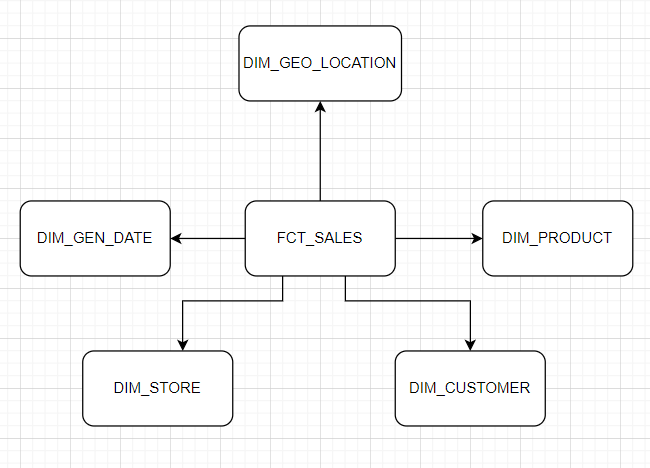
1. Calculating of sales statistics on time, product, location and customers factors.
2. Preparing well organized and documented reports, including diagrams and charts, according to business needs.
3. We want to better understand our customers to expand new markets in different locations.

**Technical Requirements**

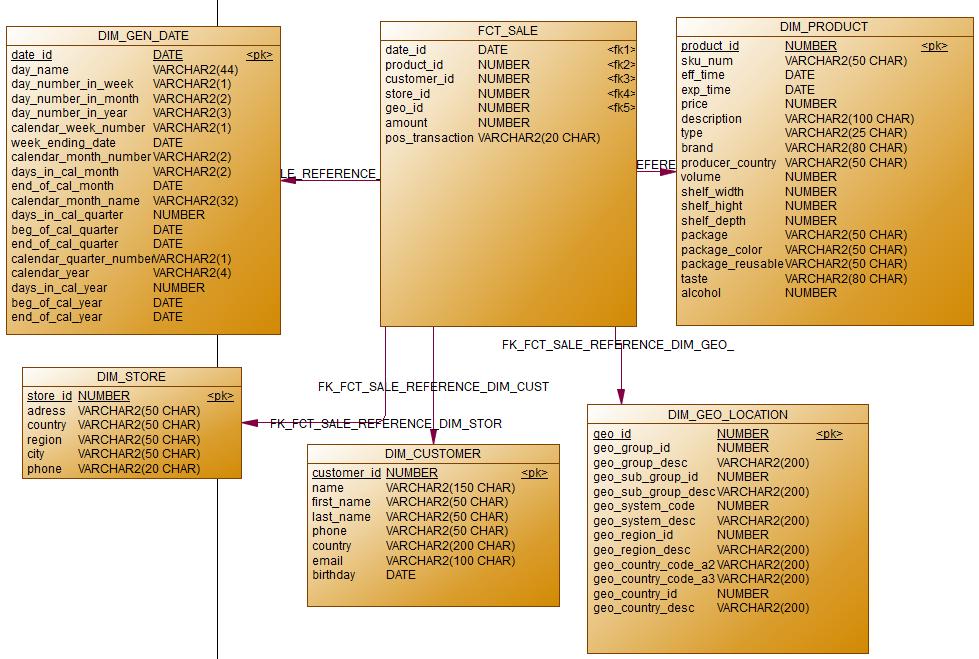
1. The ease of use and access to the system.
2. The system should work at high availability mode.
3. The reports should be prepared on a daily basis.
4. Good integration with different external systems (banking, delivery etc.)
5. High performance and speed of calculating data.

**Solution Sketch**

**Logical Star Schema:**



**Physical Star Schema:**



**Dimensions:**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Size | DW – Merged Dimensions | Descriptions |
| DIM\_GEN\_DATE | SCD1 | BIG | DW.T\_DAYS, DW.T\_WEEKS, DW.T\_MONTHS, DW.T\_QUARTERS,  DW.T\_YEARS | Contains detailed information about dates of sales. |
| DIM\_PRODUCT\_SCD | SCD2 | SMALL | product\_id  sku\_num  eff\_time  exp\_time  price  description  type  brand  producer\_country  volume  shelf\_width  shelf\_hight  shelf\_depth  package  package\_color  package\_reusable  tase  alcohol | This dimension contains the information about kvas drinks products, which are presented in company stores. The dimension has SCD2 type, which allow us to keep history of product changes (for example producer changed the volume or color of package). The *eff\_time* and *exp\_time* attributes represent the time period when record was active. The *sku\_num* attribute represented as natural key for dimension. |
| DIM\_CUSTOMER | SCD1 | SMALL | customer\_id  name  first\_name  last\_name  phone  country  email  birthday | This dimension contains information about ordinary customers. |
| DIM\_STORE | SCD1 | SMALL | store\_id  address  country  region  city  phone | This dimension contains information about company stores, which are located in different countries and cities. |
| DIM\_GEO\_LOCATION | SCD1 | SMALL | Geo\_id  Geo\_group\_id  Geo\_group\_desc  Geo\_sub\_group\_id  Geo\_dub\_group\_desc  Geo\_system\_code  Geo\_system\_desc  Geo\_region\_id  Geo\_region\_desc  Geo\_country\_code\_a2  Geo\_country\_code\_a3  Geo\_country\_id  Geo\_country\_desc | This dimension contains information about countries, regions, groups. This information describes the location where company stores are located. |

**Dimension Hierarchies:**

**DIM\_GEN\_DATE:**

**Hierarchy DAY-WEEK-MONTH-YEAR**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| DAYs | DAY | Store all day at the calendar week | DAY\_ID |
| WEEKs | WEEK | Store all weeks at the calendar month | WEEK\_ID |
| MONTHs | MONTH | Store all months at the calendar year | MONTH\_ID |
| YEARs | YEAR | Store all years | YEAR\_ID |

**Hierarchy DAY-QUARTER-YEAR**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| DAYs | DAY | Store all day at the calendar week | DAY\_ID |
| QUARTERs | QUARTER | Store all quarters at the calendar year | QUARTER\_ID |
| YEARs | YEAR | Store all years | YEAR\_ID |

**DIM\_GEO\_LOCATION:**

**Hierarchy COUNTRY-REGION-PART-SYSTEM**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| COUNTRIEs | COUNTRY | Store all counties in the geographic region | COUNTRY\_ID |
| REGIONs | REGION | Store all regions in the part of the world | REGION\_ID |
| PARTs | PART | Store all parts int the world | PART\_ID |
| SYSTEMs | SYSTEM | Store system world | GEO\_SYSTEM\_ID |

**DIM\_PRODUCT:**

**Hierarchy PRODUCT-BRAND-TYPE**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| PRODUCTs | PRODUCT | Store all products of definite brand | PRODUCT\_ID |
| BRANDs | BRAND | Store all brands of definite type | BRAND\_ID |
| TYPEs | TYPE | Store all types | TYPE\_ID |

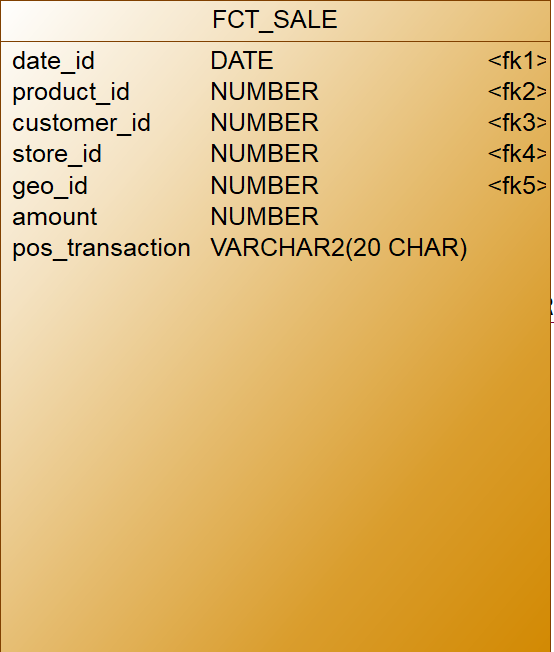
**DIM\_STORE:**

**Hierarchy STORE-CITY-REGION-COUNTRY**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| STOREs | STORE | Store all stores of city | STORE\_ID |
| CITIes | CITY | Store all cities of country region | CITY\_ID |
| REGIONs | REGION | Store all regions of country | REGION\_ID |
| COUNTRIes | COUNTRY | Store all countries where stores are located | COUNTRY\_ID |

**Facts:**

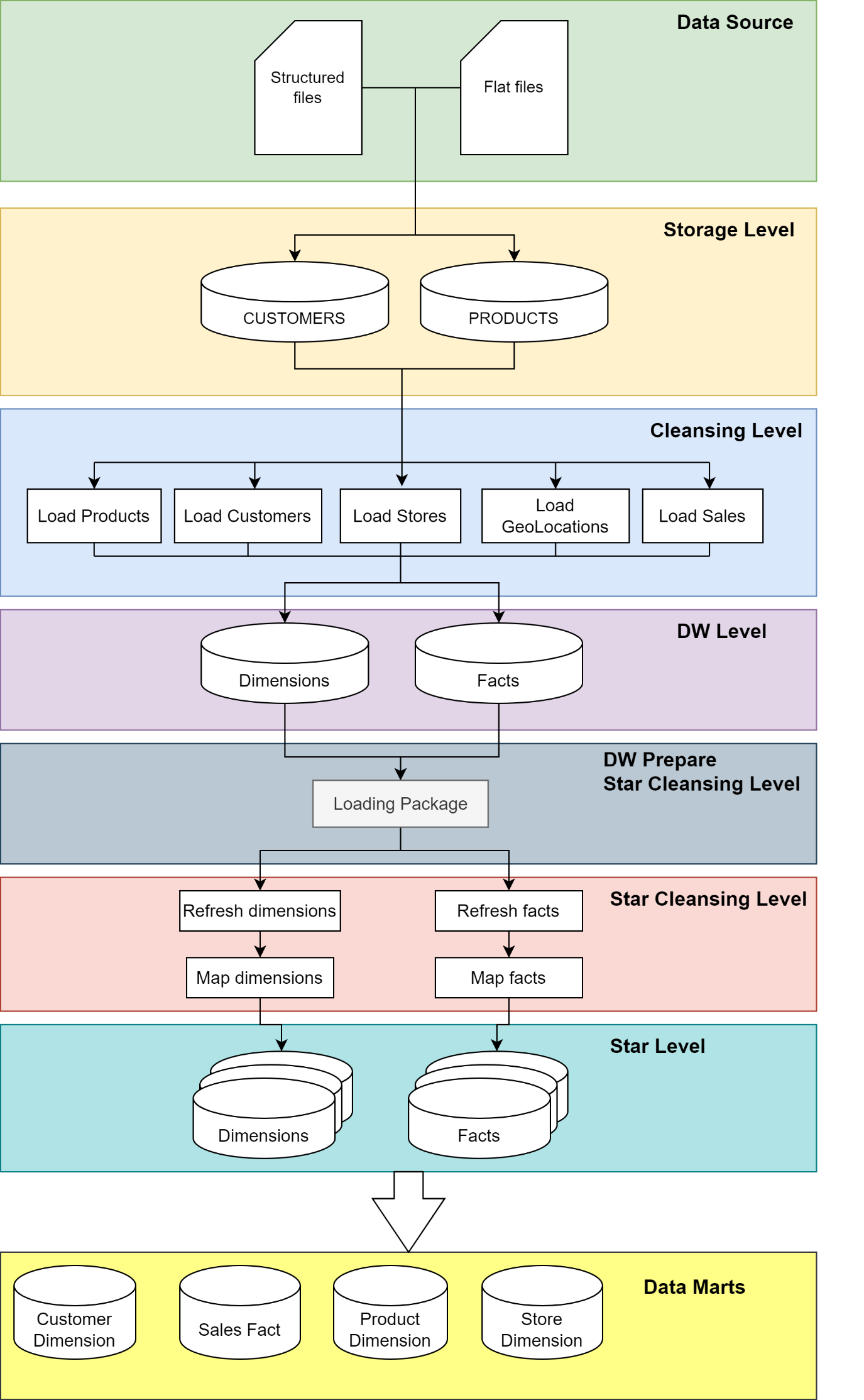
FCT\_SALES



FCT\_SALES represent individual sale transaction. Table contains 2 fact attributes. The *amount* attribute represents the number of individual product units. The *pos\_transaction* represents the code of transaction that provide us to accumulate all products in definite basket.

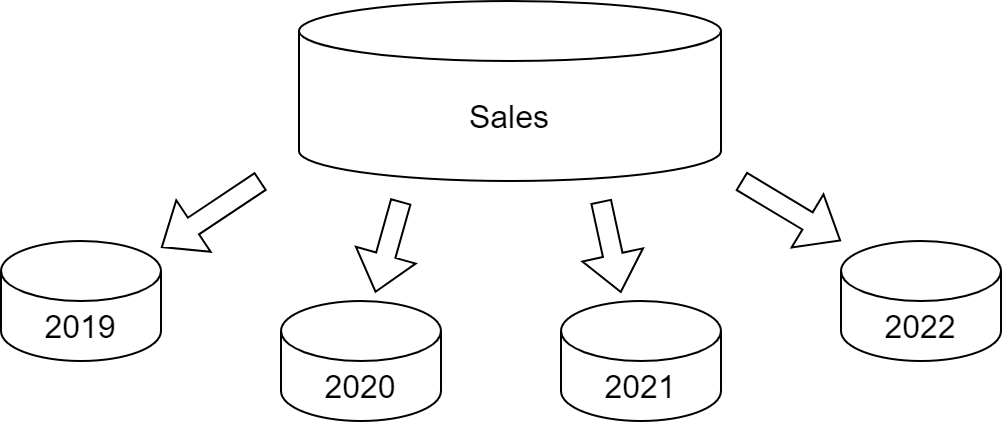
Facts aggregations for the FCT\_SALES table:

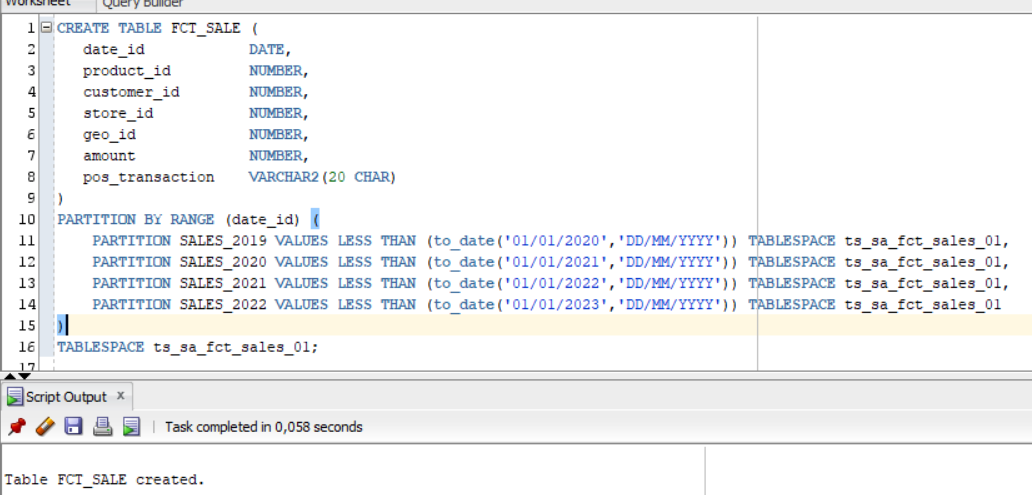
|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Code | Table Name | Additive | Descriptions |
| Total number of products sold | amount | FCT\_SALES | + | Calculates the number of sales according to range of time, product etc. |
| The code of transaction | pos\_transaction | FCT\_SALES | - | Accumulate all products from one basket (with same *pos\_transaction)* |

**DataFlow Diagram:**

**Fact Table Partitioning**

According to our single fact table FCT\_SALES, we can create range partitioning by year. In my opinion, that’s action can optimize queries which include the year conditions.



**

**Parallelization solutions**

Parallel query:

I’m completely sure, that in developing DWH can be used the parallel execution to optimize large analytical queries. The queries for large tables like FCT\_SALES can be performed using many processes to increase performance and reduce the execution time.

Parallel DML:

There are large tables in developing DWH which must be periodically updated. Using parallelization of DML operations we can increase performance while updating large tables with sales, customers, products data.

Parallel DDL:

According to results of first task, we can make conclusion that parallelization on DDL operations has significant impact on execution time. I’m sure that involving this functionality in developing DWH system will allow us to perform managing tasks faster. Also, it significantly reduces the time spend to recovery in case of any failure. The high availability is one of the main customer requirements.

**Extraction solutions**

To choose the extraction method we should consider different factors. The main factor is the possibilities of source system to implement one or another method. Very often, there is no possibility to add additional logic to the source systems to enhance an incremental extraction of data due to the performance or the increased workload of these systems. Also, we should look at the volume and the changeability of data being used.

In our business model we decided to implement the *Full Extraction* method. The main idea of this method is to extract full data completely from the source system. There is no need to keep track all data changes at definite time period. The decision was made because the volume of data system is not significantly large and there are some difficulties in implementing the *Incremental Extraction* method in provided source system.

As the physical extraction method we decided to use *Offline Extraction*. Here the data is not extracted directly from the source, but instead it’s taken from another external area which keeps the copy of source. This help us avoid overloading of the main source system fetching the records from the external source instead of the actual source.

**Transportation solutions**

The transportation method is highly depends on the source system being used. The most common and efficient way to transfer data is to use flat files and mechanism such FTP or other remote files system access protocols. Data is unloaded or exported from the source system into flat files and then transporting to the target platform using FTP or similar mechanism.

Because source systems and data warehouses often use different operating systems and database systems, using flat files is often the simplest way to exchange data between heterogeneous systems with minimal transformations. However, even when transporting data between homogeneous systems, flat files are often the most efficient and most easy-to-manage mechanism for data transfer.

**Transformation solutions**

Oracle gives us the following choices for transforming data inside the database:

* Transforming Data Using SQL
* Transforming Data Using PL/SQL
* Transforming Data Using Table Functions

In the building business model, the best solutions are using the SQL and PL/SQL transformations.

The data transforming and loading processes in the preparing DWH often use the small tables and simple transformations. These actions can be successfully completed by using standard functionality of INSERT, UPDATE, MERGE statements. Also, I should note that I have used very often the ‘UPSERT’ functionality to INSERT new rows into the table and UPDATE existing rows. The MERGE statement is really good choice for this action.

Sometimes for realization more complex transformations the standard SQL functionality is not enough. In preparing DWH for more complex and large tables such as sales or products (with SCD2 implementation) more efficient and logically simpler is to use the PL/SQL statements. For example, a PL/SQL procedure could open multiple cursors and read data from multiple source tables, combine this data using complex business rules, and finally insert the transformed data into one or more target table. It would be difficult or impossible to express the same sequence of operations using standard SQL statements.