# Exit Task

During execution of Exit task you should fill all chapters below according any Business process for you choice.

Create next separate documents:

* BI Solution Proposal
* BI Solution Concept

Create scripts for Start Scheme and store it on next structure:

Instance name

Schema name

Tables (Object group)

T\_DIM\_.... (Object name)

-scripts

T\_DIM\_.... (Object name)

-scripts

Views (Object group)

Chapter’s structures of all documents below:

# Overview

## Business Background

Lag.Net is the fast expanding internet provider. Which provide fast internet by different channels like mobile or fiber. Lag.Net provides internet access. Company has some offices in some countries. Also Lag.Net has a lot of clients in different countries , it’s clients is a person users and organization as well.

Company works in conditions of hard competition. Then it has to improve business process and service quality.

So Lag.Net needs system to watch clients traffic and payments. System should consider location, service office, providing channel. Also system should provide to clients information about their spending and tariffs by web-based interface.

Lag.Net –internet without any lag!

## Benefits

Lag.Net needs BI solution system for accurate and deep analysis trends, services sale, hardware using. Main task trace clients spending traffic and money.

System should provide managers and employees easy search and access to data about clients, offices, servers and e.t.c.

CEO can test his iPad on the interactive BI report.

# Requirements

## Business Requirements

Lag.Net has some business requirements for the solution.

* Simple and fast search of spending money and traffic.
* Company has a lot of clients. It stores information about their tariff plan, amount of consumed services and amount of payments.
* Lag.Net has some offices in different countries. Every office has its own geographical location. Warehouse must consider this fact.
* View hardware load and features.
* Storing and editing tariff plans.
* Divide information by provided internet channels.

## Technical Requirements

Solution has some important technical requirements.

* Working 24\*7
* Implement localization
* System should be compatible with https data sending
* Time granularity – day
* Statistic updated daily
* Store statistic by last 12 month

# Solution Sketch

## Source Tables structure

Clients

Info stored in local OLTP systems, tables have same format

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Comment |
| First Name | NUMBER | Basic info about client gained by the |
| Last Name | VARCHAR(10) | contract conclusion |
| Address | VARCHAR(250) |  |
| Age | NUMBER |  |
| Gender | VARCHAR(10) |  |

Server Info

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Comment |
| Channel Desc | VARCHAR(10) | selfcommenting |
| CPU MHZ | NUMBER |  |
| CPU MHZ | NUMBER |  |
| RAM | NUMBER |  |

Offices

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Comment |
| Country | VARCHAR(25) | Fields is selfcommenting |
| City | VARCHAR(25) |  |
| Address | VARCHAR(250) |  |
| Employee Count | NUMBER |  |
| Description | VARCHAR(1000) |  |

Tariffs

|  |  |  |
| --- | --- | --- |
| Name | Data Type | Comment |
| TARIFF\_ID | NUMBER | TARIFF\_ID |
| TARIFF\_DESC | VARCHAR(25) | TARIFF\_DESC |
| IN\_COST | NUMBER | IN\_COST |
| OUT\_COST | NUMBER | OUT\_COST |
| FEE | NUMBER | FEE |
|  |  |  |

## Summarize Data Plan



# DWH Solution Concept

## Logical Diagram



## Physical diagram



## Dimensions

### Dimension Types

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Type | Size | DW – Merged Dimensions | Descriptions |
| DIM\_TIMES | TYPE 1 | BIG | DW.T\_DAYS, DW.T\_WEEKS, DW.T\_MONTHS, DW.T\_QUARTERS,  DW.T\_YEARS | It is a specific type of dimension. Appears at the DWH and contains all the time since the introduction of the project until its completion. |
| DIM\_GEO\_LOCATIONS | TYPE 1 | SMALL | DW.T\_COUNTRIES  DW.T\_CNTR\_GROUPS  DW.T\_CNTR\_SUB\_GROUPS  DW.LC\_CNTR\_GROUPS  DW.T\_GEO\_TYPES  DW.T\_GEO\_SYSTEMS  DW.LC\_GEO\_SYSTEMS  DW.T\_GEO\_PARTS  DW.T\_GEO\_REGIONS  DW.T\_GEO\_OBJECTS  DW.T\_CNTR\_GROUP\_SYSTEMS  DW. LC\_CNTR\_GROUP\_SYSTEMS  DW.LC\_CNTR\_SUB\_GROUPS  DW.LC\_ GEO\_PARTS  DW.LC\_COUNTRIES  DW.LC\_ GEO\_REGIONS | This kind of dimension contains information about all countries, subregions, regions of the world. And also enters information on the types of economic development and unions according to the international classification. |
| DIM\_CLIENT\_SCD | SCD TYPE 2 | BIG | DW.T\_CUSTOMERS  DW.T\_TARIFFS | This dimension contains information about clients, their persons data, and tariff plan which they use(used). |
| DIM\_SERVERS | TYPE 1 | BIG | DW.T\_SERVERS  DW.T\_LOCATIONS  DW.T\_CHANNELS | This dimension contains information about our servers, their technical properties. |
| DIM\_TARIFFS | TYPE 1 | SMALL | DW.T\_TARIFFS | Provides information about channels of sales (description and class) |
| DIM\_GEN\_PERIODS | TYPE 1 | SMALL | DW.T\_PERIOD\_DESC | Dimension specific type, which allows grouping of facts on the basis of logic (clients tariff in our case) |

### Dimension Hierarchies

**Hierarchy DAY / MONTH / QUARTER / YEAR**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| DAY | DAY | Store all day at the calendar year | DAY\_ID |
| MONTH | MONTH | Store all months at the calendar year | WEEK\_ID |
| QUAR | QUARTER | Store all quarters at the calendar year | QUAR\_ID |
| YEAR | YEAR | Store all years at the calendar year | YEAR\_ID |

**Hierarchy WEEK / MONTH / QUARTER / YEAR**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| WEEK | DAY | Store all weeks at the calendar year | WEEK\_ID |
| MONTH | MONTH | Store all months at the calendar year | MONTH\_ID |
| QUAR | QUARTER | Store all quarters at the calendar year | QUAR\_ID |
| YEAR | YEAR | Store all years at the calendar year | YEAR\_ID |

**DIM\_SERVERS:**

**Hierarchy PRODUCTS / SUBCATEGORY / CATEGORY**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| SERVER | PROD\_NAME | Store all servers of ever y type | SERVER\_ID |
| TYPE | PROD\_SUBCATEGORY | Store all type of internet we provide | TYPE\_ID |

**DIM\_GEO LOCATIONS:**

**Hierarchy COUNTRY / SUBREGION / REGION**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| COUNTRIES | COUNTRY\_NAME | Store all countries for each region. | COUNTRY\_ID |
| SUBREGIONS | COUNTRY\_SUBREGION | Store all subregions for each region . | COUNTRY\_SUBREGION\_ID |
| REGIONS | COUNTRY\_REGION | Store all regions of the world. | COUNTRY\_REGION\_ID |

**Hierarchy COUNTRY / REGION / CONTINENT**

|  |  |  |  |
| --- | --- | --- | --- |
| Name | LEVEL\_CODE | LEVEL\_DESC | LEVEL\_NATURAL\_KEY |
| COUNTRIES | COUNTRY\_NAME | Store all countries for each region. | COUNTRY\_ID |
| SUBREGIONS | COUNTRY\_SUBREGION | Store all regions for each region . | COUNTRY\_REGION\_ID |
| REGIONS | COUNTRY\_CONTINENT | Store all continents of the world. | COUNTRY\_CONTINENT\_ID |

## Facts

FCT\_SPENDINGS\_D

### Facts Aggregations

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Name | Code | Table Name | Additive | Descriptions |
| TRAFFIC\_IN | TRAFFIC\_IN | FCT\_SPENDINGS\_D | YES | Calculate sum incoming traffic by day. |
| TRAFFIC\_OUT | TRAFFIC\_OUT | FCT\_SPENDINGS\_D | YES | Calculate sum outgoing traffic by day. |
| SPENDING | SPENDING | FCT\_SPENDINGS\_D | YES | Calculate sum of money spent on incoming and outgoing traffic |
| INOUT\_PERCENT | INOUT\_PERCENT | FCT\_SPENDINGS\_D | NO | Percent incoming traffic to outgoing by counted by day |

## Dataflow Diagram



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Level Type | Object Name | Tablespace | Description | Privileges |
| SA – Stage Area | u\_sa\_loader | ts\_sa\_data\_01 | Loading from Flat file storage system. Contains information about our business process | Create tables and views |
| DW - Cleansing Level | u\_dw\_cl | ts\_dw\_cl\_01 | Extract data from flat files, cleans it and prepare data to insertion into DW 3NF tables. | Create and drop tables,  Create and drop views  Select from SA views  Merge with SA views |
| DW – Level | u\_dw | ts\_dw\_data\_01 | Store clean information in the 3-rd normal form.  Basic DWH level | Create and drop tables,  Create and drop views  Select from DW-CL views  Merge with DW-CL views |
| DW– Prepare Star Cleansing Level | u\_sal\_dw\_cl | ts\_sal\_dw\_cl\_data\_01 | Create views consisting of merged tables from DW level. | Create and drop tables,  Create and drop views  Select from DW views |
| STAR - Cleansing | u\_sal\_cl | ts\_sal\_cl\_data\_01 | Loads information from previous level. Doesn’t load info which doesn’t matter our business analysis | Create and drop tables,  Create and drop views  Select from SAL-DW views  Merge with SAL-DW views |
| STAR – Level | u\_sa\_fct | ts\_sa\_fct \_data\_01 | Store information about facts | Create and drop tables,  Create and drop views  Select from SA-CL views |
| u\_sa\_dim | ts\_sa\_dim \_data\_01 | Store information about dimensions | Create and drop tables,  Create and drop views  Select from SA-CL views |

* For all tables : autoextend on
* For cleansing tables “no logging” cause this data doesn’t matter
* For DW and Star level tables “logging” to improve safety of the information
* “segment space management auto” for all tables

( may be for DW and Star better use manual settings for PCTUSED/PCTFREE and e.t.c for better performance)

Grants given: higher level can select and merge data he needs with the views of the lower level.

## Partitioning rules

There are two tables that can be large and should be partitioned: DWH. FCT\_SPENDINGS\_D and DIM\_CLIENTS.

We will use partition by range by event date and make some subpartitions by hash. It’s most obvious decision cause, time of storing and analysis data in our DW is 1 year. Then we can move our partitions on the volumes with different performance faster to actual data, slower to another.

Sample code example:

PARTITION BY RANGE (event\_dt)

subpartition by hash(to\_number(to\_char(event\_dt))) subpartitions 4

(

PARTITION part\_1 VALUES LESS THAN(to\_date('01/01/2008','dd/mm/yyyy'))

(

subpartition part\_1\_sub\_1,

subpartition part\_1\_sub\_2,

subpartition part\_1\_sub\_3,

subpartition part\_1\_sub\_4

),

PARTITION part\_2 VALUES LESS THAN(to\_date('01/01/2012','dd/mm/yyyy'))

(

subpartition part\_2\_sub\_1,

subpartition part\_2\_sub\_2,

subpartition part\_2\_sub\_3,

subpartition part\_2\_sub\_4

)

);

## Strategy of Parallel execution

Parallel execution is useful for many types of operations that access significant amounts of data. Parallel execution improves processing for.

* Large table scans and joins
* Aggregations and copying

We have partition already then have sense to use partition granules parallelism. When partition granules are used, a query server process works on an entire partition or subpartition of a table or index. Because partition granules are statically determined by the structure of the table or index when a table or index is created, partition granules do not give you the flexibility in parallelizing an operation that block granules do. The maximum allowable DOP is the number of partitions. This might limit the utilization of the system and the load balancing across parallel execution servers.

When partition granules are used for parallel access to a table or index, you should use a relatively large number of partitions (ideally, three times the DOP), so that Oracle can effectively balance work across the query server processes.

Partition granules are the basic unit of parallel index range scans and of parallel operations that modify multiple partitions of a partitioned table or index. These operations include parallel creation of partitioned indexes, and parallel creation of partitioned tables.