Billing System Project

# Description:

This is a sample PLSQL project for reading a file, retrieving all file data, splitting data in a certain format and writes all parsed data into database tables. A workaround (WA) type project that written in PL/SQL language.

**Data format** for reading file is:

*MSISDN|Service\_Name|Start\_Date|End\_Date|Product\_Name|Fee*

For example: “5552550000|Aylik 1 GB paketi|23.08.2017|23.09.2017|DATA|15”

The file that wanted to read/parsed should contain all data in this format. If not, then application gives an error says *“Wrong Data Format!”*

Sample File:

5552550000|Aylık 1 GB Paketi|23.08.2017|23.09.2017|**DATA**|15

5552555555|Yurtdışı 60 dk Aranma|10.09.2017|17.09.2017|**SES**|30

5552550098|Sim Kart Değişim Ücreti|01.09.2017|01.09.2017|**VAS**|10

....

....



# Product Usage Types:

Product types are listed below and stored in EDUMAN.BILLING\_PRODUCT\_TYPES table.

* **DATA**: Used for Data usages. Tax rates: %18 KDV + %5 OIV
* **SES**: Used for Voice usages. Tax rates: %18 KDV + %25 OIV
* **SMS**: Used for Message/SMS usages. Tax rates: %18 KDV + %25 OIV
* **VAS**: Used for Value Added Service usages. Tax rates: %18 KDV + %25 OIV
* **CİHAZ**: Used for Handset product usages. Tax rates: %0 KDV

# Application

## Directory and File Name

File directory created with below script:

CREATE or replace DIRECTORY USER\_DIR AS '\PLSQL\_TRAIN\oracle\_file\_directory'; *-- Full path here*

GRANT READ ON DIRECTORY USER\_DIR TO PUBLIC;

File name format is **invoice\_ddmmyy.txt**

In the day of application run (i.e. 23/09/2017) there should exist a file with **invoice\_23092017.txt** name in created USER\_DIR directory.

Based on application run date file name generated with below script:

to\_char (SYSDATE, 'ddmmyyyy');

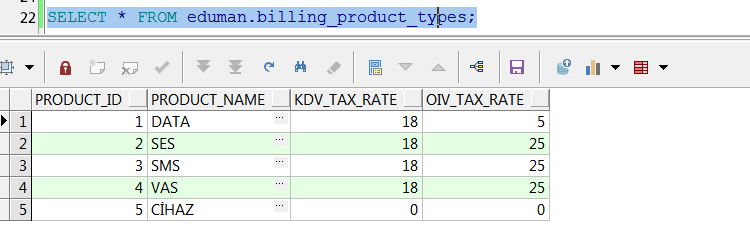
## Tables

All tables are listed below. For table columns’ comments please check table’s creation scripts!

### EDUMAN.BILLING\_PRODUCT\_TYPES

|  |  |  |
| --- | --- | --- |
| Table Name | EDUMAN.BILLING\_PRODUCT\_TYPES | |
| Table Description | Stores all products types such as SES, DATA, SMS etc. | |
| Column Name | **Data Type** | **Column Description** |
| Product\_Id | NUMBER | Unique identifier of products types. |
| Product\_Name | VARCHAR2(20) | Name of products such as DATA, SES, SMS, etc. |
| KDV\_tax\_rate | NUMBER | Defines the tax rate value of products types for KDV (Katma Deger Vergisi). |
| OIV\_tax\_rate | NUMBER | Tax rate value of product types for OIV (Ozel Iletisim Vergisi). |

**Query**: SELECT \* FROM eduman.billing\_product\_types;

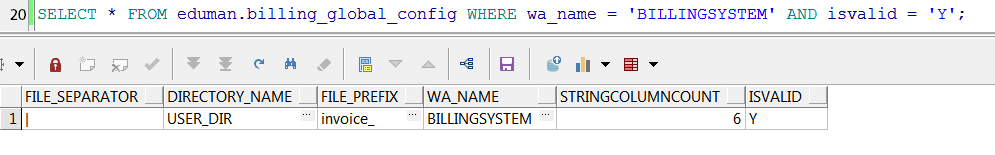


For more please check “1(a) creation\_BILLING\_PRODUCT\_TYPES.sql” file.

### EDUMAN.BILLING\_GLOBAL\_CONFIG

|  |  |  |
| --- | --- | --- |
| Table Name | EDUMAN.BILLING\_GLOBAL\_CONFIG | |
| Table Description | Stores all configurations like file separator, file prefix etc. for package execution. | |
| Column Name | **Data Type** | **Column Description** |
| File\_Separator | VARCHAR2(5) | The characters that put between texts of file data. |
| Directory\_Name | VARCHAR2(20) | Defines the directory of file path. |
| File\_Prefix | VARCHAR2(20) | Defines the extra text which will be added to the beginning of created file  (i.e. invoice\_230917.txt -> “*invoice\_”* is the prefix part). |
| WA\_Name | VARCHAR2(50) | Defines project package name to get corresponding data. |
| StringColumnCount | NUMBER | Defines the number of columns should be in corresponding data. |
| isValid | VARCHAR(1) | Defines the CONSTRAINT of table for insertion only one column with (Y). |

**Query**: SELECT \* FROM eduman.billing\_global\_config WHERE wa\_name = 'BILLINGSYSTEM';

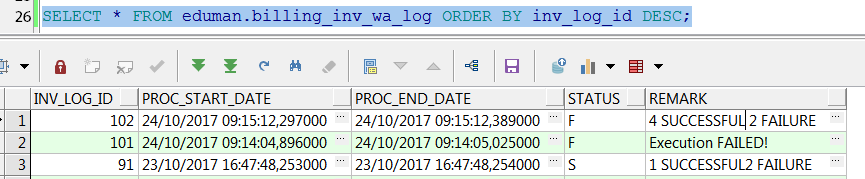


For more please check “3(a) creation\_BILLING\_GLOBAL\_CONFIG.sql” file.

### EDUMAN.BILLING\_INV\_WA\_LOG

|  |  |  |
| --- | --- | --- |
| Table Name | EDUMAN.BILLING\_INV\_WA\_LOG | |
| Table Description | WA log table, stores all execution logs info such as execution start time, end time, status of execution and number of fields that handled. | |
| Column Name | **Data Type** | **Column Description** |
| Inv\_Log\_Id | NUMBER | Defines unique identifier and primary key of BILLING\_INV\_WA\_LOG table. |
| Proc\_Start\_Date | TIMESTAMP | Executions start time (in timestamp). |
| Proc\_End\_Date | TIMESTAMP | Execution end time (in timestamp). |
| Status | VARCHAR2(1) | The status of execution (S) Success (F) Fail. |
| Remark | VARCHAR2(3000) | Contains remark, Status and count of invoices processed. |

**Query**: SELECT \* FROM eduman.billing\_inv\_wa\_log ORDER BY inv\_log\_id DESC;

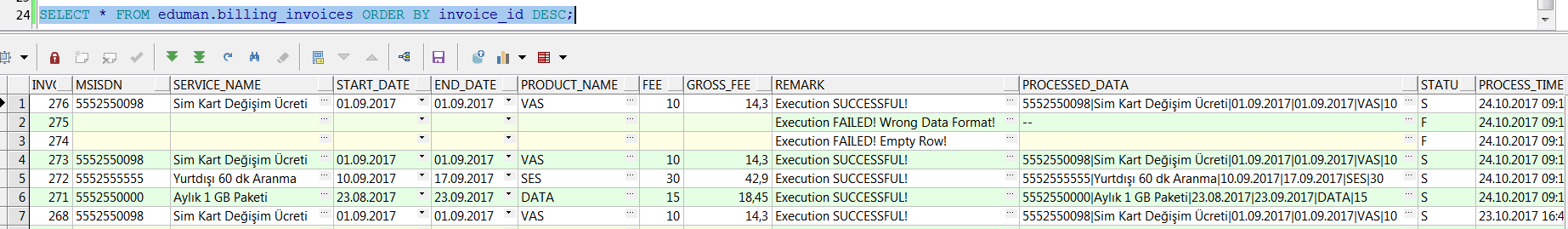


For more please check “4 creation\_BILLING\_INV\_WA\_LOG.sql” file.

### EDUMAN.BILLING\_INVOICES

|  |  |  |
| --- | --- | --- |
| Table Name | EDUMAN.BILLING\_INVOICES | |
| Table Description | Main table that stores all Invoices which are read and parsed from file. Also include gross fee which is calculated based on fee and tax rates. | |
| Column Name | **Data Type** | **Column Description** |
| Invoice\_Id | NUMBER | Unique identifier and primary key of BILLING\_INVOICES table. |
| MSISDN | VARCHAR2(10) | Defines the phone number (MSISDN) that parsed from file data. |
| Service\_Name | VARCHAR2(50) | Defines the name of service/product that parsed from file data. |
| Start\_Date | DATE | The billings start date that parsed from file data. |
| End\_Date | DATE | The billings end date that parsed from file data. |
| Product\_Name | VARCHAR(50) | Name of product which are stored in eduman.billing\_product\_types. |
| Fee | NUMBER | Fee attribute defines the cost of service that parsed from file data. |
| Gross\_Fee | NUMBER | The service price with calculation of kdv+oiv taxes. |
| Remark | VARCHAR (2000) | Defines the remark value of operation, its result can be “Execution SUCCESSFUL!” or “Execution FAILED!” |
| Processed\_Data | VARCHAR (2000) | Defines the whole data that retrieved from a row of file. |
| Status | VARCHAR(1) | Defines the status of execution which can be (S) Success, (F) Fail. |
| Process\_Time | DATE | Defines execution time of each invoice. |

**Query**: SELECT \* FROM eduman.billing\_invoices ORDER BY invoice\_id DESC;



For more please check “2 creation\_BILLING\_INVOICES.sql” file.

# Code Definitions

## Packages

### EDUMAN.BILLINGSYSTEM

This is main package. All codes are available in this package code block.

Created Date : 17.10.2017

Created by : Ercan DUMAN

Purpose : This is a sample PLSQL project for reading a file, retrieving all file data, splitting data in a certain format and writes all parsed data into database tables.

## Procedures

A procedure is a subprogram that performs a specific action which declared and defined inside PL/SQL package.

### GetGlobalConfigurations

Loads all global configurations’ variables for execution of package.

Run Code:

GetGlobalConfigurations ;

### ReadFileData

Reads whole file and retrieve data.

Run Code:

ReadFileData;

### CheckDataFormat

Checking data format for retrieved row data from file.

Run Code:

CheckDataFormat(pis\_FileRowData)

* pis\_FileRowData: Executed whole data from row.

### ParseFileData

To parsing given data, splitting based on exact format and insert in EDUMAN.BILLING\_INVOICES table.

Run Code:

ParseFileData(pis\_FileRowData);

* pis\_FileRowData : Executed whole data from row.

### i\_BillingInvoices

Makes insertion of EDUMAN.BILLING\_INVOICES table with given parameters. There two procedures with same name but different variables. This can be defined as procedure overloading.

Run Code:

i\_BillingInvoices(pis\_Msisdn, pis\_Service, pid\_StartDate, pid\_EndDate, pis\_ProductName, pion\_Fee, pis\_ProcessedData);

* pis\_Msisdn : Phone number of user that parsed from file data.
* pis\_Service : Service name that parsed from file data.
* pid\_StartDate : Service start time that parsed from file data.
* pid\_EndDate : Service end time that parsed from file data.
* pis\_ProductName : Product name which can be SES, DATA, VAS etc.
* pion\_Fee : Fee amount that parsed from file data..
* pis\_ProcessedData : Executed whole data from row of file.

### CalculateGrossFee

To calculate gross fee related to fee amount. Load all global variables for execution of package.

Gets fee amount from EDUMAN.BILLING\_INVOICES table and tax rates from EDUMAN.BILLING\_PRODUCT\_TYPES then do the calculation.

Calculation is based on:

Gross Fee = ((tax\_rates/100) +1) \* Fee Amount

(tax\_rates: KDV\_tax\_rate + OIV\_tax\_rate)

If there is no tax rate, then gross fee is equal to fee amount.

Run Code:

CalculateGrossFee(pin\_InvoiceID, pin\_Fee);

* pin\_InvoiceId : Unique idetifier of current execution.
* pin\_Fee : The price (fee amount) parsed from file data.

### u\_BillingInvoices

To update EDUMAN.BILLING\_INVOICES table’s gross\_fee and remark columns for given invoice id.

Run Code:

u\_BillingInvoices(pin\_InvoiceId, pin\_GrossFee, pis\_Remark);

* pin\_InvoiceId : Unique Identifier of current execution.
* pin\_GrossFee : Calculated fee gross amount.
* pis\_Remark : Output message for each exectuion status.

### i\_BillingInvoicesWALog

Insertion of EDUMAN.BILLING\_INV\_WA\_LOG table.

Run Code:

i\_BillingInvoicesWALog(pit\_ExecutionStartTime )

* pit\_ExecutionStartTime : Start time of package execution

### StartToProcess

The main procedure which apply all configurations and start execution of package.

Run Code:

StartToProcess;

## Try and Run Project

### Installation

For using this project, all scripts should run with given file names order.

1(a) creation\_BILLING\_PRODUCT\_TYPES.sql

1(b) insertion\_BILLING\_PRODUCT\_TYPES.sql

2 creation\_BILLING\_INVOICES.sql

3(a) creation\_BILLING\_GLOBAL\_CONFIG.sql

3(b) insertion\_BILLING\_GLOBAL\_CONFIG.sql

4 creation\_BILLING\_INV\_WA\_LOG.sql

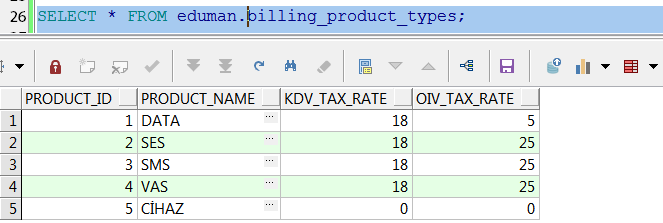
BILLINGSYSTEM.pks

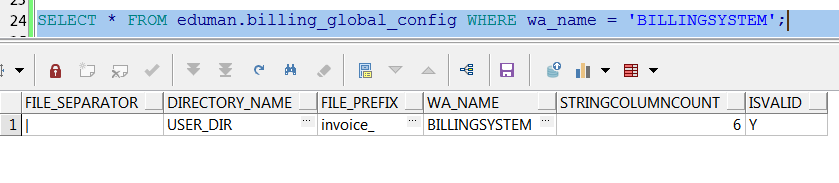
BILLINGSYSTEM.pkb

…

After running all above scripts, select queries in control\_scripts.sql file should run and must see that all tables created and eduman.billing\_global\_config and eduman.billing\_product\_types insertions are successful.

**eduman.billing\_product\_types:**



**eduman.billing\_global\_config:** 

### Run project

Application can simply run as below code.

BEGIN

EDUMAN.BILLINGSYSTEM.StartToProcess;

END;

After running package, insertion of eduman.billing\_invoices and eduman.billing\_inv\_wa\_log tables should be done.

# Notes

* All dynamic variables like file separator, directory name and file prefix stored in **EDUMAN.BILLING\_GLOBAL\_CONFIG** table.
* All product usage types like DATA, SES, SMS, VAS and CİHAZ stored in **EDUMAN.BILLING\_PRODUCT\_TYPES** table.
* Information like the execution start time, end time, status of execution and the number of processes are all stored in **EDUMAN.BILLING\_INV\_WA\_LOG** table.
* Application handle possible errors like,
  + Wrong Data Format!
  + Empty Row!
  + Empty File!
* If a new data type or a new product usage type is requested, only need to insert in corresponding table. (No need to change any code)
* For removing all package instances and tables rollback.sql script created.
* All codes run and tested on Oracle XE local database with EDUMAN schema user.
* PLSQL DEVELOPER 12.0.5.1828 IDE had been used while working on this project.