Anexo X. Procedimientos ETL del DW

Creación del database link a LUZ

Creación del Database Link para acceder a las tablas de la base de datos origen

```
/*********************************************/
/* DATABASE LINK TO THE OLTP DATABASE LUZ *****/
CREATE DATABASE LINK
  LUZ_OLTP
CONNECT TO system
IDENTIFIED BY asus
USING 'localhost:1521/luz';
```

La creación de un database link requiere disponer de los permisos adecuados

```
GRANT CREATE DATABASE LINK TO dw;
```

Permisos para el acceso a los datos de la tabla LUZ a través del Database Link

```
GRANT SELECT, INSERT, UPDATE DELETE ON COUNTRY@LUZ_OLTP TO DW;
GRANT SELECT, INSERT, UPDATE DELETE ON PROVINCE@LUZ_OLTP TO DW;
GRANT SELECT, INSERT, UPDATE DELETE ON CITY@LUZ_OLTP TO DW;
GRANT SELECT, INSERT, UPDATE DELETE ON STREETTYPE@LUZ_OLTP TO DW;
GRANT SELECT  ON ADDRESS@LUZ_OLTP TO DW;
GRANT SELECT, INSERT, UPDATE DELETE ON BANK@LUZ_OLTP TO DW;
GRANT SELECT, INSERT, UPDATE DELETE ON IDENTITYTYPE@LUZ_OLTP TO DW;
GRANT SELECT  ON CONSUMER@LUZ_OLTP TO DW;
GRANT SELECT, INSERT, UPDATE DELETE ON COMPANY@LUZ_OLTP TO DW;
GRANT SELECT  ON METER@LUZ_OLTP TO DW;
GRANT SELECT  ON CONNECTION@LUZ_OLTP TO DW;
GRANT SELECT  ON PRICE@LUZ_OLTP TO DW;
GRANT SELECT  ON CLIENTS@LUZ_OLTP TO DW;
GRANT SELECT  ON OPERATORS@LUZ_OLTP TO DW;
GRANT SELECT  ON LUZLOGO@LUZ_OLTP TO DW;
```

Creación de la tabla LOG para control de las cargas del DW

Esta tabla registra la fecha y los parámetros de carga de los datos del DW (tanto de las tablas de dimensiones como la tabla de hechos)

```
/***** FOR THE CONTROL OF THE ETL PROCESS ******************/
CREATE TABLE FACT_CONTROL_DATE(
  ID_ETL INTEGER CONSTRAINT PK_FACT_CONTROL_DATE PRIMARY KEY
  , ETL_DATAPROCES TIMESTAMP
  ,DATEINI DATE
  ,DATEFI DATE
  ,NOTES VARCHAR2(200 CHAR)
);

CREATE SEQUENCE seq_FACT_CONTROL_DATE INCREMENT BY 1 START WITH 1;

CREATE OR REPLACE TRIGGER Insert_IDETL
  BEFORE INSERT ON FACT_CONTROL_DATE
  FOR EACH ROW
BEGIN
  SELECT seq_FACT_CONTROL_DATE.NEXTVAL INTO :NEW.ID_ETL FROM DUAL;
END Insert_ID_ETL;
```

Consulta para el proceso ETL de las tablas de dimensiones

Esta consulta es la base para realizar la carga de las tablas de dimensiones. Proporciona todos los datos necesarios de la base de datos operacional. Se ha considerado más óptimo generar una consulta materializada.

```
/* CREATE THE MASTER VIEW FOR THE ETL PROCESS ON DIMENSIONS*********************/
CREATE OR REPLACE VIEW
  STA_DIMENSIONS_QUERY
AS
SELECT
  CONNECTION.READINGDATE, CONNECTION.INSTANTCONSUMPTION
  , METER.ADDRESSCODE, CONNECTION.METERSERIALNUMBER, METER.INSTALLATIONDATE,
METER.METERMODEL
  , ADDRESS.CITYCODE
  , CITY.CITYNAME
  , PROVINCE.PROVINCECODE, PROVINCE.PROVINCENAME
  , COUNTRY.COUNTRYCODE, COUNTRY.COUNTRYNAME
  , CONSUMER.CONSUMERCODE, CONSUMER.CONSUMERNAME, CONSUMER.CONSUMERSURNAME
  , COMPANY.COMPANYTAXCODE, COMPANY.COMPANYNAME
FROM
  CONNECTION@LUZ_OLTP LEFT JOIN METER@LUZ_OLTP ON
CONNECTION.METERSERIALNUMBER=METER.SERIALNUMBER
  LEFT JOIN ADDRESS@LUZ_OLTP ON METER.ADDRESSCODE=ADDRESS.ADDRESSCODE
  LEFT JOIN CITY@LUZ_OLTP ON ADDRESS.CITYCODE=CITY.CITYCODE
  LEFT JOIN PROVINCE@LUZ_OLTP ON CITY.PROVINCECODE=PROVINCE.PROVINCEID
  LEFT JOIN COUNTRY@LUZ_OLTP ON PROVINCE.COUNTRYCODE=COUNTRY.COUNTRYCODE
  LEFT JOIN CONSUMER@LUZ_OLTP ON METER.CONSUMERCODE=CONSUMER.CONSUMERCODE
  LEFT JOIN COMPANY@LUZ_OLTP ON METER.COMPANYCODE=COMPANY.COMPANYTAXCODE;
```

Procedimiento para la inserción/actualización de las tablas de dimensiones

El procedimiento actualiza las tablas de dimensiones y guarda el log en la tabla FACT_CONTROL_DATE

```
CREATE OR REPLACE PROCEDURE ETL_LOAD_DIMENSIONS
/*********************************************
| INSERT DATA IN DIMENSIONS TA TABLES
|
| In Params:  --
| Out Params: --
| Date:
| Author: AJCC
|*********************************************/
IS
BEGIN
  /* DELETE DIMENSIONS TABLES */
  delete from w_address_d;
  delete from w_company_d;
  delete from w_consumer_D;
  delete from w_date_D;
  delete from w_meter_d;
  delete from w_time_d;

  /*********** IMPORT DIMENSIONS ******************/
  /* UPDATE W_ADDRESS_D DIMENSIONS TABLE */
  MERGE INTO W_ADDRESS_D dw USING (SELECT DISTINCT ADDRESSCODE, CITYCODE, CITYNAME,
COUNTRYCODE, COUNTRYNAME FROM STA_DIMENSIONS_QUERY) oltp
  ON (dw.addresscode=oltp.addresscode)
  WHEN MATCHED THEN
    UPDATE SET
      dw.citycode=oltp.citycode
      , dw.CityName=oltp.cityName
```

```
      , dw.countrycode=oltp.countrycode
      , dw.countryname=oltp.countryname
  WHEN NOT MATCHED THEN
    INSERT(dw.addressCode, dw.cityCode, dw.CityName, dw.countrycode, dw.countryname )
    VALUES(oltp.addressCode,     oltp.cityCode,     oltp.CityName,     oltp.countrycode,
oltp.countryname);

  /* UPDATE W_COMPANY_D DIMENSIONS TABLE */
  MERGE INTO W_COMPANY_D dw USING (SELECT DISTINCT COMPANYTAXCODE, COMPANYNAME FROM
STA_DIMENSIONS_QUERY) oltp
  ON (dw.COMPANYCODE=oltp.COMPANYTAXCODE)
  WHEN MATCHED THEN
    UPDATE SET
      dw.COMPANYNAME=oltp.COMPANYNAME
  WHEN NOT MATCHED THEN
    INSERT(dw.COMPANYCODE, dw.COMPANYNAME)
    VALUES(oltp.COMPANYTAXCODE, oltp.COMPANYNAME);


  /* UPDATE W_CONSUMER_D DIMENSION TABLE */
  MERGE INTO W_CONSUMER_D dw USING (SELECT DISTINCT CONSUMERCODE, CONSUMERNAME FROM
STA_DIMENSIONS_QUERY) oltp
  ON (dw.CONSUMERCODE=oltp.CONSUMERCODE)
  WHEN MATCHED THEN
    UPDATE SET
      dw.CONSUMERNAME=oltp.CONSUMERNAME
  WHEN NOT MATCHED THEN
    INSERT(dw.CONSUMERCODE, dw.CONSUMERNAME)
    VALUES(oltp.CONSUMERCODE, oltp.CONSUMERNAME);

  /* UPDATE W_DATE_D DIMENSION TABLE */
  MERGE INTO W_DATE_D dw USING (SELECT DISTINCT TRUNC(READINGDATE) AS NEWDATE FROM
STA_DIMENSIONS_QUERY) oltp
  ON (dw.DATE_WID=OLTP.NEWDATE)
  WHEN MATCHED THEN
    UPDATE SET
      dw.DAYOFWEEK=TO_CHAR(OLTP.NEWDATE, 'D')
      , dw.DAYNUMBERMONTH=TO_CHAR(OLTP.NEWDATE, 'DD')
      , dw.MONTHNUMBER=TO_CHAR(OLTP.NEWDATE, 'MM')
      , dw.MONTHNAME=TO_CHAR(OLTP.NEWDATE, 'MON')
      , dw.YEARNUMBER=TO_CHAR(OLTP.NEWDATE, 'YYYY')
  WHEN NOT MATCHED THEN
    INSERT(
      DATE_WID
      , DAYOFWEEK
      , DAYNUMBERMONTH
      , MONTHNUMBER
      , MONTHNAME
      , YEARNUMBER)
    VALUES(
      OLTP.NEWDATE
    , TO_CHAR(OLTP.NEWDATE, 'D')
    , TO_CHAR(OLTP.NEWDATE, 'DD')
    , TO_CHAR(OLTP.NEWDATE, 'MM')
    , TO_CHAR(OLTP.NEWDATE, 'MON')
    , TO_CHAR(OLTP.NEWDATE, 'YYYY'));


  /* UPDATE W_METER_D DIMENSION TABLE */
```

```
  MERGE INTO W_METER_D dw USING (SELECT DISTINCT METERSERIALNUMBER, INSTALLATIONDATE,
METERMODEL FROM STA_DIMENSIONS_QUERY) oltp
  ON (dw.METERCODE=oltp.METERSERIALNUMBER)
  WHEN MATCHED THEN
    UPDATE SET
      dw.METERINSTALLATIONDATE=oltp.INSTALLATIONDATE
      , dw.MODELNAME=oltp.METERMODEL
  WHEN NOT MATCHED THEN
    INSERT(dw.METERCODE, dw.MODELNAME, dw.METERINSTALLATIONDATE)
    VALUES(oltp.METERSERIALNUMBER, oltp.METERMODEL, oltp.INSTALLATIONDATE);

  /* UPDATE W_TIME_D DIMENSION TABLE */
  MERGE INTO W_TIME_D dw USING (SELECT DISTINCT TO_NUMBER(EXTRACT(hour from READINGDATE))
AS TIMEW FROM STA_DIMENSIONS_QUERY) oltp
  ON (dw.TIME_WID=OLTP.TIMEW)
  WHEN MATCHED THEN
    UPDATE SET
      DW.TIMEHOURNUMBER=(OLTP.TIMEW||':00')
  WHEN NOT MATCHED THEN
    INSERT( TIME_WID, TIMEHOURNUMBER)
    VALUES(OLTP.TIMEW, (OLTP.TIMEW||':00'));

  /* LOG OF ETL IN FACTS TABLE */
  INSERT INTO FACT_CONTROL_DATE(ETL_DATAPROCES, DATEINI, DATEFI, NOTES) VALUES
(Current_TimeStamp, NULL, NULL, 'UPDATE DIMENSION TABLES');

END;
```

Consulta para el proceso ETL de la tabla de hechos

Esta consulta es la base para realizar la carga de la tabla de hechos. Proporciona todos los datos necesarios de la base de datos operacional junto con los valores relacionados de las tablas de dimensiones. Además proporciona el cálculo de la lectura anterior de cada contador y el cálculo del precio de la energía vigente en el momento de la lectura.

```
/* CREATE THE MASTER VIEW FOR THE ETL PROCESS */
CREATE OR REPLACE VIEW STA_FACT_QUERY
AS
SELECT
  W_DATE_D.DATE_WID
  , W_TIME_D.TIME_WID
  , W_METER_D.METER_WID
  , W_ADDRESS_D.ADDRESS_WID
  , W_COMPANY_D.COMPANY_WID
  , W_CONSUMER_D.CONSUMER_WID
  ,(SELECT distinct FIRST_VALUE(B.INSTANTCONSUMPTION) OVER (PARTITION BY
B.METERSERIALNUMBER ORDER BY B.READINGDATE DESC) FROM STA_DIMENSIONS_QUERY B WHERE
B.METERSERIALNUMBER=oltp.METERSERIALNUMBER AND B.READINGDATE<oltp.READINGDATE AND
B.INSTANTCONSUMPTION IS NOT NULL) as PREVIOUSREADING
  , oltp.INSTANTCONSUMPTION AS ACTUALREADING
  ,(SELECT distinct FIRST_VALUE(NEWPRICE) OVER (ORDER BY CHANGEDATA DESC) FROM
PRICE@LUZ_OLTP WHERE COUNTRYCODE=OLTP.COUNTRYCODE AND COMPANYCODE=oltp.COMPANYTAXCODE
AND CHANGEDATA<=OLTP.READINGDATE)  AS PRICE
FROM
  STA_DIMENSIONS_QUERY oltp
  INNER JOIN W_ADDRESS_D ON  oltp.ADDRESSCODE = W_ADDRESS_D.ADDRESSCODE
  inner join W_COMPANY_D ON oltp.COMPANYTAXCODE=W_COMPANY_D.COMPANYCODE
  INNER JOIN W_CONSUMER_D ON oltp.CONSUMERCODE=W_CONSUMER_D.CONSUMERCODE
  INNER JOIN W_METER_D ON OLTP.METERSERIALNUMBER=W_METER_D.METERCODE
  INNER JOIN W_DATE_D ON TRUNC(OLTP.READINGDATE)=W_DATE_D.DATE_WID
  INNER JOIN W_TIME_D ON TO_NUMBER(EXTRACT(hour from READINGDATE))=W_TIME_D.TIME_WID;
```

Procedimiento para la inserción/actualización de la tabla de hechos

El procedimiento actualiza las tablas de dimensiones y guarda el log en la tabla FACT_CONTROL_DATE. Los datos en la tabla de hechos se guardan por periodos por lo tanto es necesario indicar la fecha de inicio y la fecha de final para la inserción de los datos.

```
/*****************************************************/
/*****************************************************/
CREATE OR REPLACE PROCEDURE ETL_LOAD_FACTS(
/*********************************************
| INSERT DATA IN DIMENSIONS TA TABLES
|
| In Params:  --
| Out Params: --
| Date:
| Author: AJCC
|*********************************************/
    DATEINI IN DATE
    ,DATEFI  IN DATE
)
IS
  lException    EXCEPTION;
  lErrorDescrip VARCHAR2(100);

BEGIN

  /*VALIDATION CONTROLS */

  IF DATEINI IS NULL THEN
    lErrorDescrip:=q'[DateINI can't be null]';
    RAISE lException;
  END IF;

  IF DATEFI IS NULL THEN
    lErrorDescrip:=q'[DateFI can't be null]';
    RAISE lException;
  END IF;

  IF DATEINI>DATEFI  THEN
    lErrorDescrip:=q'[DateFI can't be greather than DATAINI]';
    RAISE lException;
  END IF;


  /* DELETE THE FACTS TABLE */
  delete from w_consumption_F;

  /*DELETE SUMMARY TABLES */
  DELETE FROM W_CITY_CONS_SUMMARY_M;
  DELETE FROM W_CITY_METER_SUMMARY_M;
  DELETE FROM W_CITY_SUMMARY_M;
  DELETE FROM W_COMPANY_SUMMARY_M;
  DELETE FROM W_CONSUMER_SUMMARY_M;
  DELETE FROM W_COUNTRY_CITY_SUMMARY_M ;
  DELETE FROM W_COUNTRY_SUMMARY_M;
  DELETE FROM W_COUNTRY_SUMMARY_Y;
  DELETE FROM W_METER_SUMMARY_M;


  INSERT INTO W_CONSUMPTION_F(
      DATE_WID
      , TIME_WID
      , METER_WID
      , ADDRESS_WID
      , COMPANY_WID
      , CONSUMER_WID
      , PREVIOUSREADING
      , ACTUALREADING
      , PRICE
      /* THERE IS A TRIGGER THAT CALCULATES
      , CONSUMPTION
      , ENERGYCOST  */)
    SELECT
       DATE_WID
      , TIME_WID
```

```
      , METER_WID
      , ADDRESS_WID
      , COMPANY_WID
      , CONSUMER_WID
      , PREVIOUSREADING
      , ACTUALREADING
      , PRICE
    FROM STA_FACT_QUERY
    WHERE DATE_WID BETWEEN DATEINI AND DATEFI AND ROWNUM<10000;

  /* LOG OF ETL IN FACTS TABLE */
  INSERT   INTO   FACT_CONTROL_DATE(ETL_DATAPROCES,   DATEINI,   DATEFI,   NOTES)   VALUES
(Current_TimeStamp, DATEINI, DATEFI, 'UPDATE FACTS TABLE');

  EXCEPTION
    WHEN lException THEN
      --Track Store Procedure Result
      DBMS_OUTPUT.PUT_LINE ('ERROR: ' || lErrorDescrip);

    WHEN OTHERS THEN
      DBMS_OUTPUT.PUT_LINE ('ERROR: ' || sqlerrm);
END;
```

Procedimiento para la inicialización y carga del Almacén de datos

Mediante este procedimiento se inicializan todas las tablas de dimensiones, las tablas de resumen, y la tabla de hechos. Acto seguido se procede con la carga de las dimensiones, la tabla de hechos y las tablas resumen necesarias para las consultas.

```
CREATE OR REPLACE PROCEDURE INI_DW(
/*********************************************
| INIT THE DATAWAREHOUSE AND LOAD NEW DATA
|
| In Params:  --
| Out Params: --
| Date:
| Author: AJCC
|*********************************************/
    DATAINI IN DATE
    ,DATAFI  IN DATE
)
IS

BEGIN
  /* DELETE THE FACTS TABLE */
  DBMS_OUTPUT.PUT_LINE ('DELETING FACTS TABLE');
  delete from w_consumption_F;


  /*DELETE SUMMARY TABLES */
  DBMS_OUTPUT.PUT_LINE ('DELETING SUMMARY TABLES');
  DELETE FROM W_CITY_CONS_SUMMARY_M;
  DELETE FROM W_CITY_METER_SUMMARY_M;
  DELETE FROM W_CITY_SUMMARY_M;
  DELETE FROM W_COMPANY_SUMMARY_M;
  DELETE FROM W_CONSUMER_SUMMARY_M;
  DELETE FROM W_COUNTRY_CITY_SUMMARY_M ;
  DELETE FROM W_COUNTRY_SUMMARY_M;
  DELETE FROM W_COUNTRY_SUMMARY_Y;
  DELETE FROM W_METER_SUMMARY_M;
  DELETE FROM W_CITY_METER_SUM_T;


  /* DELETE DIMENSIONS TABLES */
  DBMS_OUTPUT.PUT_LINE ('DELETING DIMENSION TABLES');
  delete from w_address_d;
  delete from w_company_d;
  delete from w_consumer_D;
  delete from w_date_D;
  delete from w_meter_d;
  delete from w_time_d;

  DBMS_OUTPUT.PUT_LINE ('DIMENSION TABLES LOADING');
```

```
  /* LOAD DIMENSIONS */
  ETL_LOAD_DIMENSIONS;
  DBMS_OUTPUT.PUT_LINE ('DIMENSION TABLES LOADED');


  DBMS_OUTPUT.PUT_LINE ('FACTS TABLE LOADING');
  /* LOAD FACT TABLE */
  ETL_LOAD_FACTS(DATaINI, DATaFI);

  DBMS_OUTPUT.PUT_LINE ('FACTS TABLE LOADED');

EXCEPTION
  WHEN OTHERS THEN
    DBMS_OUTPUT.PUT_LINE ('ERROR: ' || sqlerrm);
END;
```