AN SOFTWARE INTERFACE BETWEEN SAS® AND ORACLE®

Wei-le Qian, State Information Center, Beijing, China
Yue-jin Chen, State Information Center, Beijing, China

ABSTRACT

This paper introduces a SAS interface software developed under IBM MVS operating system. It provides SAS users with a way to access and analyze their ORACLE relational data base using SAS software.

This software consists of three SAS procedures, all of them can execute in full-screen mode or in batch.

1. PROC ORAEXT ---- In full-screen mode, you can make choice to ORACLE tables and views, and the columns you want to extract. A SAS variables can be the combination of several columns (include function operation). It also allows you to operate on SAS variable name and its format, finally establishes the SAS data sets you expected.

2. PROC ORALOAD ---- Creates and loads ORACLE tables, or insert data to an existing table, using SAS data set input.

3. PROC ORASQL you can perform any SQL statements in interactive mode or in batch.

Above three procedures were written with PL/I language. We use ORACLE's PRO·PLI product to operate ORACLE resources, use SAS program interface to operate SAS data set, and use IBM's ISPF product to implement the full-screen user interface. We also provide a chinese version for the interface software.

INTRODUCTION

If you want to use SAS software to analyze and process data, then the structure of the data must be presented in SAS data sets. You may use DATA steps provided by SAS BASICS to create SAS data set directly, probably you may exchange data from OS data sets or VSAM files to SAS data sets. For those SAS users, however, who might be the relational data base users, such as ORACLE, expect some other ways. Since ORACLE provides very powerful and convenient functions in data management, inquiry, and update etc., it is very significant for those users to develop such an interface software, which can conveniently transform ORACLE table and views to SAS data sets, or vice versa.

Following will introduce three SAS procedures to satisfy these users' requirement.

1. PROC ORAEXT : This procedure is used to extract the data in ORACLE data base to create SAS data set. To invoke the procedure, type

   PROC ORAEXT;
   RUN;

You then see the DATA ACCESS PANEL (Screen 1). Three fields will be input, the USERNAME field and the PASSWORD field are the identification to enter into ORACLE relational data base, it provides you privileges to access data base. The OUTPUT SAS DATA SET field allows you to specify the name of the SAS data set where you want to store the extracted data, the SAS data set can either be permanent data set or a temporary data set. The default name of data set is WORK.DATAn. After press PF3, you will enter TABLE SELECTION PANEL (Screen 2), which lists all the tables and views that users have authority to access. Press PF7 and PF8, you can control screen scrolling forward and backward. You can select one of the tables or views, or more. You only need pressing S in FUNC
field, and then press ENTER, the • sign will appear in the SEL field to indicate that these tables or views are selected. When you have made all the choices you expected, press PF3 to enter DATA EXTRACTION PANEL (Screen 3), which lists the column names and data types for each table and view you have supplied on the prior screen, the format of the column in SAS format. Notice that, at this time SINGLE is present in the TYPE field. You may type any of these commands in the TYPE field, they are: SINGLE, COMBINE, WHERE, LIST, SUBMIT. Each indicates different functions.

Let's see the case of type SINGLE in the TYPE field. For each column you want to extract, type an S in the FUNC field, then press ENTER, that column will be included in the SAS data set, and fill in the SAS NAME field with a valid SAS name.

The default SAS name supplied by system is the first eight characters of the column name. Notice that, do not make same SAS name as could as possible, if so, however this procedure can also manage it automatically.

You can change SAS format of each column, which will determine the output format of the variables in SAS data set. In the FUNC field, you can also press D, then the selected column will be canceled. In addition, if you press A in the FUNC field, you will select all the columns on the Screen 3. In this case, • sign will be displayed in each SEL field, indicate that all the columns will be extracted. The key PF7, and PF8 control screen scrolling. Once you have made all the choice, press the key PF3, the data will be extracted, and the SAS data set will be created. The extraction status will be displayed on Screen 1. If you want to do some other operations, like COMBINE, or WHERE, do not press PF3, just type correspondent command with your expected operation in the TYPE field.

If you type COMBINE in the TYPE field, another screen will be displayed (Screen 4). COMBINE means that a SAS variable corresponds not only a column in the table, but also the multi-column's combining operation. For instance, on several columns, you can make addition, subtraction, multiplication and division. You also can perform the function operation, such as SUM(Z), ABS(T) etc. on the column provided by ORACLE. All of the column names and formats are listed for you on the screen as reference on forming a combined item.

In the SAS NAME field in COMBINE AREA, you must to fill a SAS variable name you expected, this variable will be included in the SAS data set where you want to store the extracted data. The
combined item is input in the IT EXPRESSION field, of course it is valid for ORACLE. The FORMAT field specify the variable's SAS format. You can perform the combining operation on SAS variables many times.

Sometimes when you extract data and deal with multi-table operations, you expect to input a inquiry condition. Then, you only need to type WHERE in the TYPE field to enter Screen 5. Please write the condition statement in the WHERE CLAUSE portion of the screen. All of the column names, SAS names and the formats are listed on screen as your reference.

On the screen with the TYPE field described above, you can input SINGLE, COMBINE, WHERE or LIST separately, and you may input SUB or SUBMIT at any time to start extracting. In the case of Screen 3, pressing the key PF3 is equal to input SUBMIT.

For those users who want to use this procedure in batch, you can invoke the procedure, type

```
PROC ORAEXTB U=SCOTT P=TIGER OUT=C.EMP;
SEL STMT='SELECT * FROM EMP WHERE DNO > 100 AND Z < 200';
RUN;
```

Any of the valid SQL inquiry statements can be written in the SEL STMT session.

2. PROC ORALOAD. This procedure is opposite to procedure ORAEXT, it create and load an ORACLE table according to user's SAS data set.

For existing tables, excuse data insert operation. To invoke this procedure in full-screen mode, type

```
PROC ORALOAD;
RUN;
```

You then see DATA ACCESS PANEL(Screen 6). In the INPUT DATA SET field, you must to fill a name of SAS data set, which contains the data you want to use as input for the table creation. USERNAME and PASSWORD are used to identify your ORACLE privileges. You specify the name of the table you want to create in the TABLE NAME field. If the table is new, type NEW in the TYPE field, the procedure will execute creating and loading. If the table is existed, type INSERT in the TYPE field, the procedure will execute data insert operation. Press PF3 to enter the screen specified by the TYPE field.

If it is a new table, then enter into TABLE
CREATE/LOAD PANEL (Screen 7). This displays the list of variables in the SAS data set and the information that will be used to create the ORACLE table. In the FUNC field of the column you expect, type S and ENTER, the * sign will be displayed in the SEL field, it indicates that this column will be included in the new table. The default for SAS NAME and FORMAT are COLNAME and COLFORMAT, which can also be changed if necessary. The NULLS field denote the status of the column. Press PF3 to start executing, and the execution status will be displayed on Screen 6.

If you want to use the data in a SAS data set to execute an insert operation on an existing table, you then get Screen 8, all of the column names and formats are listed on the left half part of the screen, and the SAS variable names and the SAS formats of the SAS data set are displayed on the right half part of the screen. The key of the selection is pointing out, what are the columns you want to insert, and which of the SAS variables is mapped to the column in the table. Please pay attention to CKEY and SKEY fields.

The CKEY is identification code of the column names in the table, which has displayed on the screen. The SKEY is used to identify the SAS variable name corresponding to the column name in the selected table, it is empty before execution. When you fill a code in the SKEY field, which must be one of the code in the CKEY field, the mapping relations between SAS variables and columns in the table have been defined. You can see the mapping relations in the example of Screen 8. If you want to insert the data of SAS variable YMD to the column Y in the table, you may fill 6 to left position of YMD in the SKEY field, and press ENTER, so the mapping relation is built.

When you have defined the mapping relation, Press PF3 to start data insert operation.

In batch mode, only new tables are considered. To invoke the procedure, type

```
PROC ORALOADB U=SCOTT P=TIGER TNAME=EMP
DATA=C.EMP;
RUN;
```

The table EMP will be created, the data will be loaded into the table EMP using SAS data set C.EMP as input.

3. PROC ORASQL: In this procedure, you can execute any SQL statements in SAS environment. To invoke the procedure in interactive mode, type

```
PROC ORASQL U=SCOTT P=TIGER;
RUN;
```

When submitted, you enter into an environment in which you can execute any SQL statements. It is similar to ORACLE'S SQL PLUS, but only SQL statements can be executed. The screen shows as follows:

```
SQL> SELECT * FROM COL WHERE
SQL> TNAME=EMP;
```

When you press ENTER after the semicolon, the execution will start. If you type

```
SQL>EXIT;
```

You can return to SAS state.

It is probably more significant for users that this procedure can operate in batch. First let's see its statement form:

```
PROC ORASQLB U=SCOTT P=TIGER N=5,
SQL STMT1=' ......SQL STATEMENT ......' ,
SQL STMT5=' ......SQL STATEMENT ......' ,
RUN;
```

From above statement, you may find its attraction, which means that in a SAS job, you can operate on ORACLE resources conveniently under SAS environment.

If you are familiar with SAS/AF product, and use the MACRO variable, it becomes possible to execute SQL statement (built with SAS product) in your application system.

You will find the important points that under SAS
environment, whatever in interactive mode or in
batch, to execute any one of SQL statement only
means to execute a SAS procedure.

IMPLEMENTATION AND SOFTWARE SUPPORT

This interface software between SAS and ORACLE
described above has developed several SAS
procedures in substance.

SAS system provides a SAS user program interface
to users. The SAS procedures can be written in
PL/I statement. With the help of such tools,
provided by SAS system, as GRAMMAR PROCESSOR, X-
ROUTINES, SUBLIB ROUTINES, we can implement
procedure's grammar writing, I/O operation on SAS
data set, and processing some special problems
under SAS environment.

For these procedures certainly deal with the
operation on tables and views in ORACLE relational
data base, we also use an ORACLE's product
PRO-PL/I, which allows you to process ORACLE
resources in language level. So we can write such
a statement in our PL/I program, like EXEL SQL ...
to implement dynamic operation on ORACLE.

In order to provide a friendly interface to users,
we use IBM's product PL/I to implement user
interface.

For this interface software in Chinese version, we
have provided Chinese user interface, the Chinese
support in variable name, and the Chinese support
for character's values.

Running Environment :
 IBM MVS Version 3.8
 ORACLE Version 5.1
 SAS Version 5.1

REFERENCES

* ORACLE SQL·PLUSE USER'S GUIDE *, VERSION 2.0,
 ORACLE CORPORATION .

* ORACLE PRO-PL/I USER'S GUIDE *, VERSION 1.1,
 ORACLE CORPORATION.

* ORACLE ERROR MESSAGES AND CODE MANUAL, VERSION
  5.1, ORACLE CORPORATION.

* SAS PROGRAMMER'S GUIDE FOR PL/I *, VERSION 5
  EDITION, SAS INSTITUTE INC, CARY, NC.

CONTACT AUTHOR:

Weili Qian
Department of Data Base and Programme Bank
State Information Centre
No.58 Sanlihe Road
Muxidi, Beijing, China
Postal Code: 100045