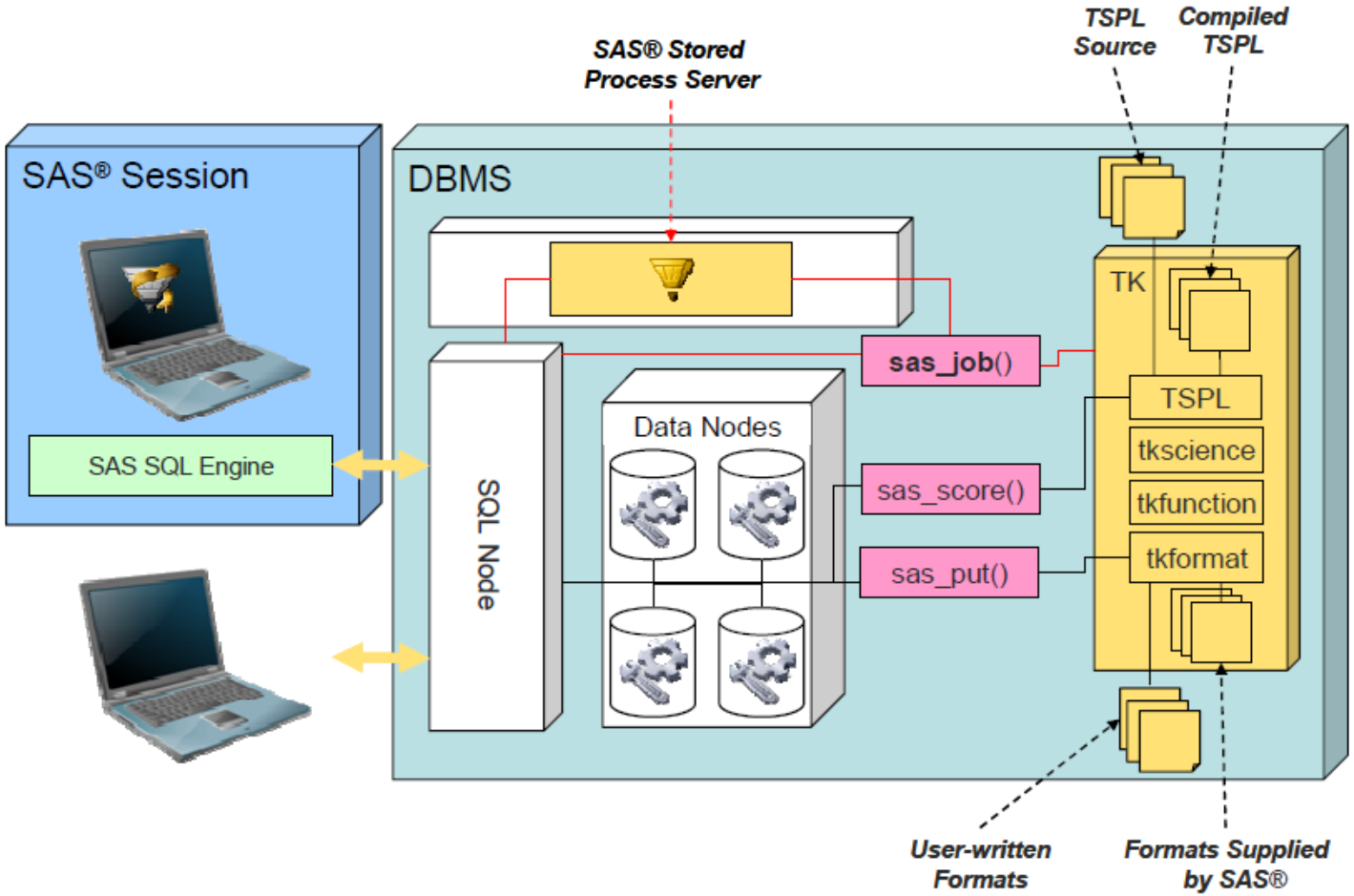
Integrating SAS with DBMSs



The Purpose

To analyze the data, especially “Big Data,” that are stored in and managed by a standard DBMS with powerful SAS statistical analysis tools by making use of the power of the two for improving analysis capabilities and computing performance.

The DBMS

ORACLE

The Approach

Using the SAS/ACCESS software to interact with databases from within SAS. It provides three methods:

- Use the LIBNAME statement to assign SAS librefs to Oracle DBMS objects. You can use a SAS two-level name to specify any table or view in the database. You can then work with Oracle tables or views as you would with SAS datasets. (“Libname Oracle”)
- Use the SQL pass-through facility to interact with a data source using its native SQL syntax (here, Oracle SQL*PLUS) without leaving your SAS session. SQL statements are passed directly to the data source for processing.
- Use the ACCESS and DBLOAD procedures to indirectly access DBMS data. Although SAS still supports these procedures for database systems and environments on which they were available in SAS 6, they are no longer the recommended methods for accessing DBMS data.

LIBNAME Statement for Relational Databases

- In SAS/ACCESS, the LIBNAME statement extends the SAS global LIBNAME statement to enable you to assign a libref to a relational DBMS.
- This feature lets you reference a DBMS (Oracle) object directly in a DATA step or SAS procedure. You can use it to
  - read from or write to a DBMS object as if it were a SAS dataset;
  - associate a SAS libref with a relational database server, a database, a schema, or a group of tables (a table cluster) and views.

The LIBNAME Statement General Syntaxes

Form 1: **LIBNAME** *libref engine-name*
<SAS/ACCESS-connection-options>
<SAS/ACCESS-LIBNAME-options> ;

Form 2: **LIBNAME** *libref* CLEAR | _ALL_ CLEAR;

Form 3: **LIBNAME** *libref* LIST | _ALL_ LIST;

Required Arguments

libref is any SAS name that serves as an **alias** to associate SAS with a DBMS. The SAS/ACCESS LIBNAME statement creates pointers (or nicknames), pointing to the location where your tables and views are stored in a DBMS.

engine-name is the SAS/ACCESS engine name for your DBMS, such as **oracle** or **db2**. The engine name is DBMS-specific.

CLEAR disassociates one or more currently assigned librefs. Specify *libref* to disassociate a single libref. Specify _ALL_ to disassociate all currently assigned librefs.

ALL specifies that the CLEAR or LIST argument applies to all currently assigned librefs.

LIST writes the attributes (libref name, scope, engine, path, schema/user, etc.) of one or more SAS/ACCESS libraries to the SAS log. Specify *libref* to list the attributes of a single SAS/ACCESS library. Specify _ALL_ to list the attributes of all libraries that have librefs in your current session.

The LIBNAME Statement General Syntaxes (con't)

Form 1: **LIBNAME** *libref engine-name*
<*SAS/ACCESS-connection-options*>
<*SAS/ACCESS-LIBNAME-options*> ;
Form 2: **LIBNAME** *libref* CLEAR | _ALL_ CLEAR;
Form 3: **LIBNAME** *libref* LIST | _ALL_ LIST;

Optional Arguments

SAS/ACCESS-connection-options provide connection information and control how SAS manages the connection to the DBMS, e.g., user name and password. These arguments are different for each database.

SAS/ACCESS-LIBNAME-options define how DBMS objects are processed by SAS. Some LIBNAME options can enhance performance; others determine locking or naming behavior. These options are also DBMS-specific.

The Oracle Engine in the LIBNAME Statement (a review)

The Syntax:

libname ***libref*** **oracle**
user=<'> *User_ID* <'>
password= <'> *Password* <'>
path= <'> *Oracle_Connection* <'> <*other_options*> **;**

Example: Oracle Engine in the LIBNAME Statement

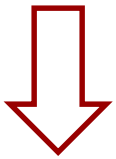
```
libname myora oracle
user='XY' password='XXXXXXX' path='xe';
run;

proc print data=myora.product_t;
title "The Product Table form the Oracle Database";
run;
```

The Product Table form the Oracle Database

Obs	PRODUCTID	PRODUCTLINEID	PRODUCTDESCRIPTION	PRODUCTFINISH	PRODUCTSTANDARDPRICE
1	1	1	End Table	Cherry	175.00
2	2	2	Coffee Table	Natural Ash	200.00
3	3	2	Computer Desk	Natural Ash	375.00
4	4	3	Entertainment Center	Natural Maple	650.00
5	5	1	Writers Desk	Cherry	325.00
6	6	2	8-Drawer Desk	White Ash	750.00
7	7	2	Dining Table	Natural Ash	800.00
8	8	3	Computer Desk	Walnut	250.00

Use the New Libref in a PROC SQL Procedure

```
proc sql;
      select productid, productfinish, productstandardprice
           from myora.product_t
           where productstandardprice>250
           order by 3;
quit;
```



PRODUCTID	PRODUCTFINISH	PRODUCTSTANDARDPRICE
5	Cherry	325.00
3	Natural Ash	375.00
4	Natural Maple	650.00
6	White Ash	750.00
7	Natural Ash	800.00

Update a DBMS Table From Within SAS

```
proc sql;
        select * from myora.product_t;
        update myora.product_t
           set productstandardprice=500
               where productfinish ='Cherry';
        select * from myora.product_t;
quit;
```

Oracle Procuct_t table **before** update

PRODUCTID	PRODUCTLINEID	PRODUCTDESCRIPTION	PRODUCTFINISH	PRODUCTSTANDARDPRICE
1	1	End Table	**Cherry**	**175.00**
2	2	Coffee Table	Natural Ash	200.00
3	2	Computer Desk	Natural Ash	375.00
4	3	Entertainment Center	Natural Maple	650.00
5	1	Writers Desk	**Cherry**	**325.00**
6	2	8-Drawer Desk	White Ash	750.00
7	2	Dining Table	Natural Ash	800.00
8	3	Computer Desk	Walnut	250.00

Oracle Procuct_t table **after** update

PRODUCTID	PRODUCTLINEID	PRODUCTDESCRIPTION	PRODUCTFINISH	PRODUCTSTANDARDPRICE
1	1	End Table	**Cherry**	**500.00**
2	2	Coffee Table	Natural Ash	200.00
3	2	Computer Desk	Natural Ash	375.00
4	3	Entertainment Center	Natural Maple	650.00
5	1	Writers Desk	**Cherry**	**500.00**
6	2	8-Drawer Desk	White Ash	750.00
7	2	Dining Table	Natural Ash	800.00
8	3	Computer Desk	Walnut	250.00

Run a SAS Procedure Against an Oracle Table

```
proc means data=myora.employee_payroll_t
      min mean max;
      var salary;
      class employee_gender;
run;
```

Analysis Variable : SALARY				
EMPLOYEE_GENDER	N Obs	Minimum	Mean	Maximum
F	191	24015.00	35591.31	207885.00
M	233	22710.00	40050.04	433800.00

Do a Simple t-test in Proc Means

```
proc means data=myora.employee_payroll_t  n
mean stderr t prt;
     var salary;
     class employee_gender;
run;
```

Analysis Variable : SALARY						
EMPLOYEE_GENDER	N Obs	N	Mean	Std Error	t Value	Pr > |t|
F	191	191	35591.31	1298.42	27.41	<.0001
M	233	233	40050.04	2591.04	15.46	<.0001

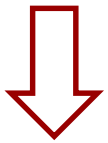
To Disassociate the myora Libref

```
LIBNAME myora CLEAR;
```

(You disassociate all the librefs if you issue "**LIBNAME** _ALL_ CLEAR")

To check the effect, run

```
proc print data=myora.product_t;
title "Can I still get data from the Oracle Database?";
run;
```



ERROR: Libname MYORA is not assigned.

The SQL Pass-Through Facility

It connects SAS to a DBMS (Oracle) and to send native SQL statements directly to the DBMS for execution. You can

- establish and terminate connections with a DBMS using its **CONNECT TO** and **DISCONNECT FROM** statements;
- use the SQL syntax of your DBMS (Oracle);
- send dynamic, non-query, DBMS-specific SQL statements to a DBMS using the **EXECUTE (…) BY** statement;
- retrieve data directly from a DBMS using the CONNECTION TO component in the FROM clause of a PROC SQL SELECT statement.

The SQL Pass-Through Facility Syntaxes for Oracle

- **CONNECT TO** ORACLE <**AS** *alias*>
  <**(**<*database-connection-arguments*><*connect-statement-arguments*>**)**>**;**
- **DISCONNECT FROM** ORACLE | *alias***;**
- **EXECUTE (***DBMS-specific-SQL-statements***) BY** ORACLE | *alias***;**
- **SELECT** *column-list* **FROM CONNECTION TO** ORACLE | *alias (DBMS-query);*

Connect to and Disconnect From Oracle with the Pass-Through Facility

```
proc sql;
    CONNECT TO oracle AS dbcon
       (USER='xy' PASSWORD='xxxxx' PATH='xe');
       SELECT *
       FROM CONNECTION TO dbcon
          (SELECT * from product_t);
    DISCONNECT FROM dbcon;
quit;
```

PRODUCTID	PRODUCTLINEID	PRODUCTDESCRIPTION	PRODUCTFINISH	PRODUCTSTANDARDPRICE
1	1	End Table	Cherry	500.00
2	2	Coffee Table	Natural Ash	200.00
3	2	Computer Desk	Natural Ash	375.00
4	3	Entertainment Center	Natural Maple	650.00
5	1	Writers Desk	Cherry	500.00
6	2	8-Drawer Desk	White Ash	750.00
7	2	Dining Table	Natural Ash	800.00
8	3	Computer Desk	Walnut	250.00

Using the EXECUTE…BY Statement

Find the customers who have placed an order:

```
proc sql;
    connect to oracle (user='xy' password='xxxxxxx');
        execute (create view whoordered_v as
          select customer_t.customerid, customer_t.customername
            from customer_t, order_t
                    where order_t.customerid=customer_t.customerid)
                    by oracle;
        execute (grant select on whoordered_v to xy) by oracle;
        select *
          from connection to oracle
           (select * from whoordered_v);
    disconnect from oracle;
quit;
```

CUSTOMERID	CUSTOMERNAME
1	Contemporary Casuals
1	Contemporary Casuals
2	Value Furniture
3	Home Furnishings
4	Eastern Furniture
5	Impressions
8	California Classics
11	American Euro Lifestyles
12	Battle Creek Furniture
15	Mountain Scenes

Practice

- Use the LIBNAME statement to create a libref to associate with the Oracle database you created before (use your own user name and password). From the Customer_t table find all available information of the customers who live in New York and Pennsylvania; sort your result first by state and then by name. Then you update the Customer_t table by adding a new customer with the following data: customerid=100, customername= Cool Furture, customeraddress=123 Lake Side Drive, customercity=Ithaca, customerstate=NY, and customerpostalcode=14850. Display the whole tale to confirm the update, sorting by customerid in descending order. Disassociate the libref you created.
- Use the SQL Pass-through facility to connect to the Oracle database you created earlier by assigning an alias called "myconn." Within Proc SQL, use the EXECUTE statement to create a table (called PTF_t) in the SASUSER library, which is the inner join of the customer_t and order_t, containing the following columns: customerid, customername, customerstate and customerpostalcode. Disconnect from the Oracle database. Display the new PTF_t table in SASUSER.

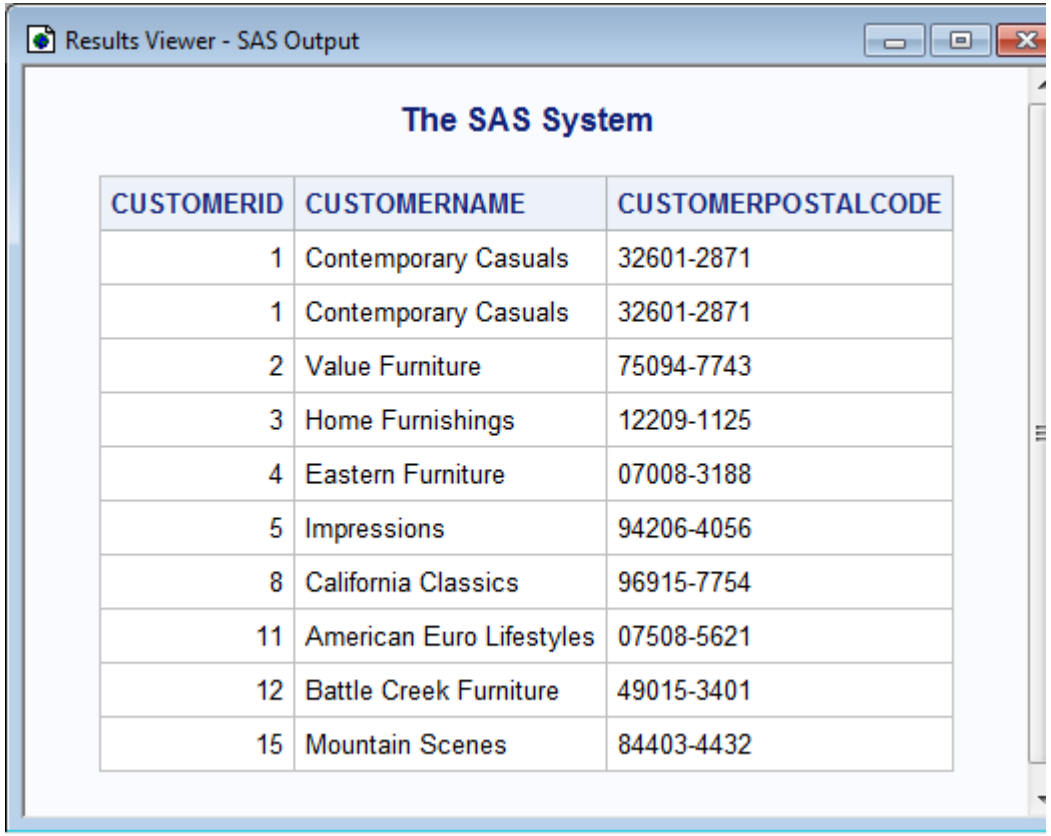
Solution to bullet 1, part 1

CUSTOMERID	CUSTOMERNAME	CUSTOMERADDRESS	CUSTOMERCITY	CUSTOMERSTATE	CUSTOMERPOSTALCODE
3	Home Furnishings	1900 Allard Ave.	Albany	NY	12209-1125
16	Lab Fixtures	11 Downtown Rd.	Ithaca	NY	14850
18	Office Furniture	100 Tower Rd.	Ithaca	NY	14853
13	Heritage Furnishings	66789 College Ave.	Carlisle	PA	17013-8834
20	Office Plus Furniture	160 Bigler Rd.	University Park	PA	16802

Solution to bullet 1, part 2

CUSTOMERID	CUSTOMERNAME	CUSTOMERADDRESS	CUSTOMERCITY	CUSTOMERSTATE	CUSTOMERPOSTALCODE
100	Cool Furture	123 Lake Side Drive	Ithaca	NY	14850
20	Office Plus Furniture	160 Bigler Rd.	University Park	PA	16802
18	Office Furniture	100 Tower Rd.	Ithaca	NY	14853
16	Lab Fixtures	11 Downtown Rd.	Ithaca	NY	14850
15	Mountain Scenes	4132 Main Street	Ogden	UT	84403-4432
14	Kaneohe Homes	112 Kiowai St.	Kaneohe	HI	96744-2537
13	Heritage Furnishings	66789 College Ave.	Carlisle	PA	17013-8834
…	…	…	…	…	…

Solution to bullet 2



CUSTOMERID	CUSTOMERNAME	CUSTOMERPOSTALCODE
1	Contemporary Casuals	32601-2871
1	Contemporary Casuals	32601-2871
2	Value Furniture	75094-7743
3	Home Furnishings	12209-1125
4	Eastern Furniture	07008-3188
5	Impressions	94206-4056
8	California Classics	96915-7754
11	American Euro Lifestyles	07508-5621
12	Battle Creek Furniture	49015-3401
15	Mountain Scenes	84403-4432