# JUST PASSING THROUGH… *OR ARE YOU?*

## DETERMINE WHEN SQL PASS-THROUGH OCCURS TO OPTIMIZE YOUR QUERIES

Misty Johnson
Wisconsin Department of Health Services, Madison, WI

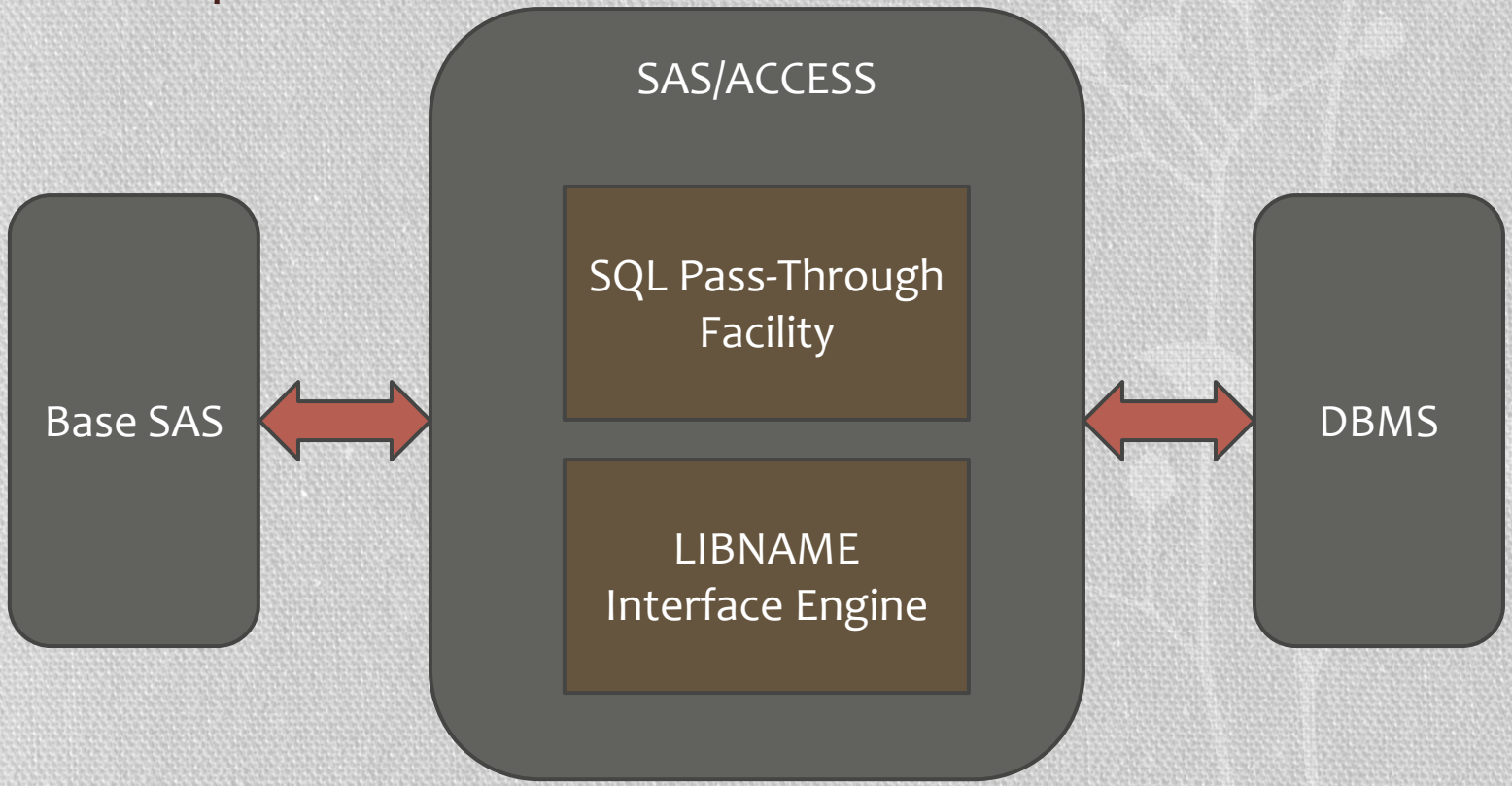
# Outline

- SAS/ACCESS
- SQL Pass-Through Facility
- LIBNAME Interface Engine
  - Triggers of Implicit SQL Pass-Through
  - Helpful system options
- Query Comparisons
- Effect of Processing Location on Run time
- Summary

# SAS/ACCESS

# SAS/ACCESS

- Separate licensed product
- Communicate with a Data Base Management System (DBMS)
- Two components:



Component 1:

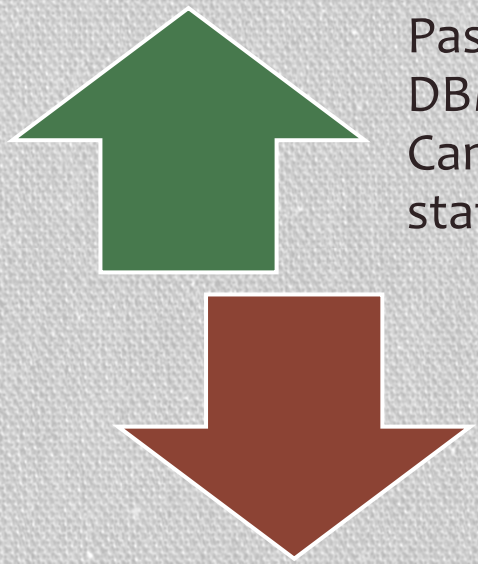
# SQL PASS-THROUGH FACILITY

# SQL Pass-Through Facility

- Send queries directly to DBMS
  - Faster processing
- PROC SQL coding structure
  - SAS SQL
  - Native SQL

```
PROC SQL;
   CONNECT TO ODBC (DATASRC=FMS_PROD USER=&SYSUSERID. DBPROMPT=YES);
        SELECT *
        FROM CONNECTION TO ODBC
            /* THIS CODE PROCESSED ON DBMS */
            (SELECT DISTINCT Contract_Year AS year, Profile_ID AS profid,
            Appn AS app, Proj AS prj, Resp AS ra
            FROM DBO.VOUCHER_CODING
            WHERE CONTRACT_YEAR IN ('14','15','16','17'));
            /* END CODE PROCESSED ON DBMS */
   DISCONNECT FROM ODBC;
QUIT;
```

# SQL Pass-Through Facility



Passes query directly to the DBMS
Can use native SQL statements/functions

Larger code structure

Must use PROC SQL

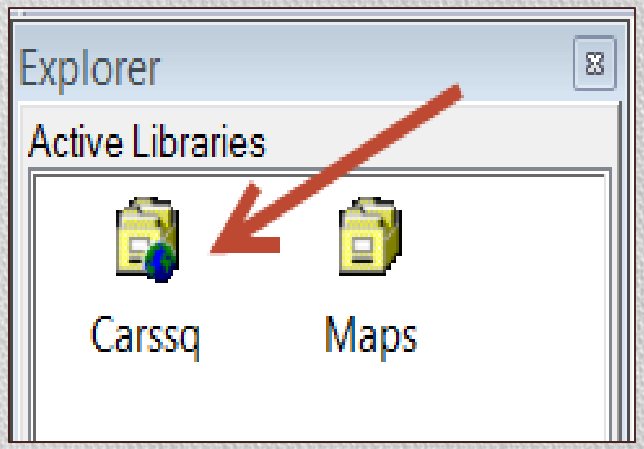
Component 2:

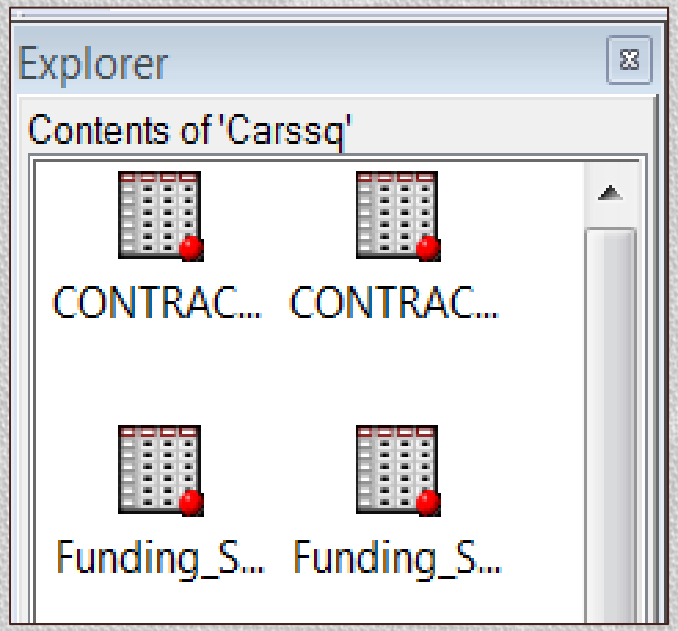
# LIBNAME INTERFACE ENGINE

# LIBNAME Interface Engine

- Link to DBMS like a SAS Library

```
LIBNAME CARSSQ ODBC DATASRC=… SCHEMA=… USER=…;
```



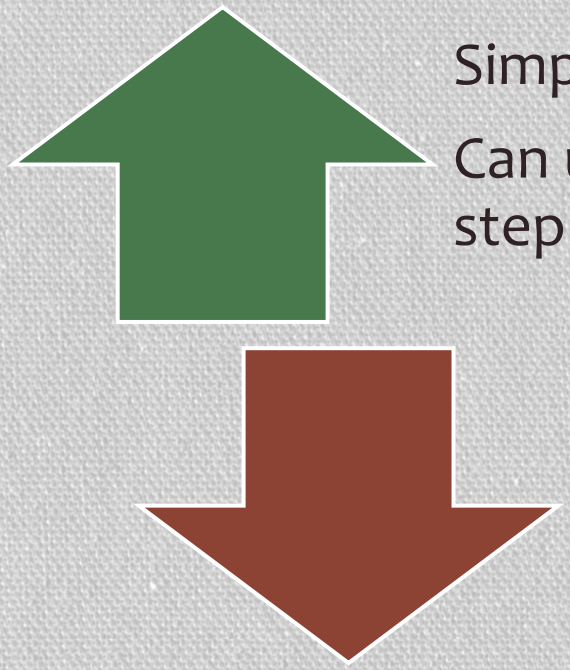
# LIBNAME Interface Engine

- Communicate with DBMS
- Engine optimizes queries
  - Specific to DBMS
- PROC SQL or DATA step coding structure
  - SAS only

```
LIBNAME CARSSQ ODBC
DATASRC=FMS_PROD SCHEMA=DBO USER=&SYSUSERID. DBPROMPT=YES;

PROC SQL;
        SELECT …
        FROM CARSSQ.VOUCHER_CODING
        WHERE …;
QUIT;
```

# LIBNAME Interface Engine



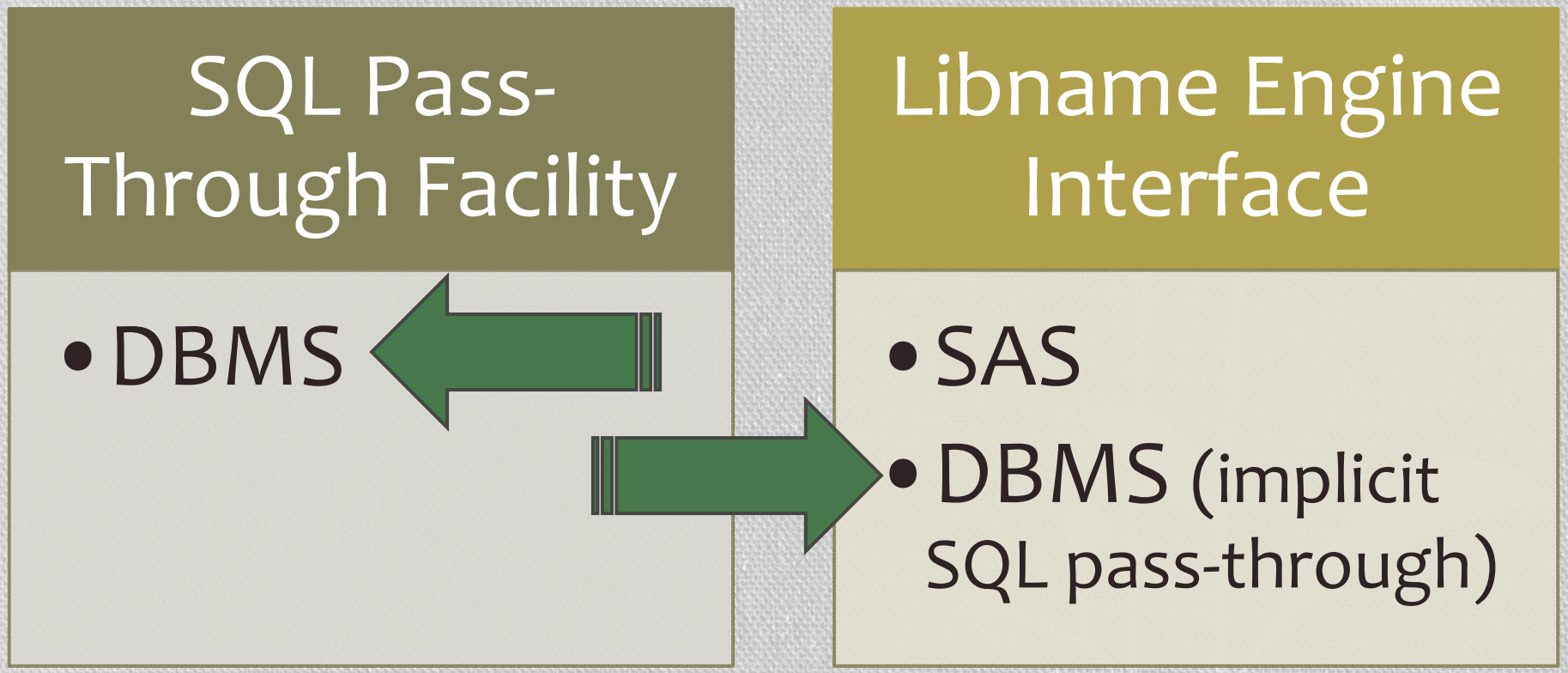
Simple coding

Can use PROC SQL or DATA step

Cannot use native SQL

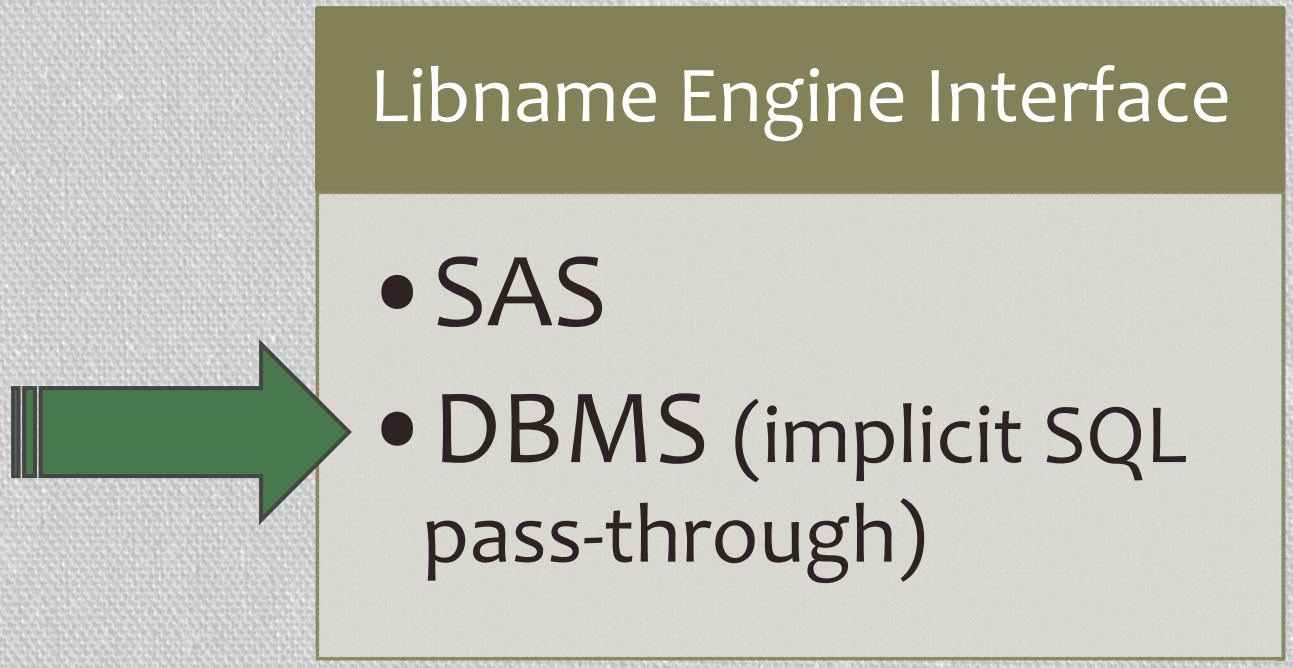
Must understand triggers of implicit SQL pass-through

# Where does Processing Occur?



# Best of both worlds??

- Simple coding structure
- Pass-through to DBMS
  - Triggers of implicit SQL pass-through



# WHAT TRIGGERS IMPLICIT SQL PASS-THROUGH?

# Implicit SQL Pass-Through Triggers

PROC SQL

- DISTINCT
- JOIN
- UNION
- COMPUTED
- Aggregate functions (SUM, MAX, etc)

DATA step

- WHERE clause

# Conditions that Prohibit Pass-Through

- Multiple, disparate librefs
- Most DATA set options
- SAS functions in the SELECT clause
- Use of DIRECT SQL= in LIBNAME option
- Any generated SQL syntax not accepted by the DBMS

# Handy Options for Implicit Pass-Through

**DEBUG=DBMS_SELECT**

- Displays SELECT clause generated by the libname engine
- Notification if implicit pass-through occurred

**SASTRACE='',,,d'**

- Displays *all* the SQL statements generated by the LIBNAME engine
- Notification if implicit pass-through occurred

# QUERY PROCESSED ON SAS

# Query Processed on SAS (script)

```
OPTIONS DEBUG=DBMS_SELECT SASTRACE=',,,d' SASTRACELOC=SASLOG
NOSTSUFFIX;
LIBNAME OLD 'L:\BFSPROC\CARSVOUCHERS';

LIBNAME CARSSQ ODBC DATASRC=FMS_PROD SCHEMA=DBO
USER=&SYSUSERID. DBPROMPT=YES;

PROC SQL;
     CREATE TABLE old.TABLE2 AS
     SELECT Contract_Year AS year, Profile_ID AS profid,
Appn AS app,
     Proj AS prj, Resp AS ra, Pcnt AS perct
     FROM CARSSQ.VOUCHER_CODING as Codetable
     WHERE CONTRACT_YEAR IN ('14','15','16','17');
QUIT;
```

# Query Processed on SAS (log)

```
9    LIBNAME CARSSQ ODBC DATASRC=FMS_PROD SCHEMA=DBO USER=&SYSUSERID. DBPROMPT=YES;
NOTE: Libref CARSSQ was successfully assigned as follows:
      Engine:        ODBC

11   <PROC SQL statements here>
ODBC: AUTOCOMMIT is NO for connection 0
ODBC: AUTOCOMMIT turned ON for connection id 0
DBMS_SELECT: SELECT * FROM "DBO"."VOUCHER_CODING"

ODBC_1: Prepared: on connection 0
SELECT * FROM "DBO"."VOUCHER_CODING"

DBMS_SELECT: SELECT  "Contract_Year", "Profile_ID", "Appn", "Proj", "Resp", "Pcnt  FROM
"DBO"."VOUCHER_CODING"  WHERE  (("Contract_Year" IN  ('14','15','16','17')))

ODBC_2: Prepared: on connection 0
SELECT  "Contract_Year", "Profile_ID", "Appn", "Proj", "Resp", "Pcnt"  FROM
"DBO"."VOUCHER_CODING"  WHERE  (("Contract_Year" IN  ('14','15','16','17')))

ODBC_3: Executed: on connection 0
Prepared statement ODBC_2
NOTE: Table OLD.TABLE2 created, with 2406 rows and 13 columns

17   QUIT;
NOTE: PROCEDURE SQL used (Total process time):
      real time           11.12 seconds
      cpu time            0.18 seconds
```

SASTRACE

DEBUG=
DBMS_SELECT

Trigger = DISTINCT

# QUERY PROCESSED ON THE DBMS

# Query Processed on the DBMS (script)

```
OPTIONS DEBUG=DBMS_SELECT SASTRACE=',,,d' SASTRACELOC=SASLOG
NOSTSUFFIX;
LIBNAME OLD 'L:\BFSPROC\CARSVOUCHERS';

LIBNAME CARSSQ ODBC DATASRC=FMS_PROD SCHEMA=DBO
USER=&SYSUSERID. DBPROMPT=YES;

PROC SQL;
     CREATE TABLE old.TABLE2 AS
     SELECT DISTINCT Contract_Year AS year, Profile_ID AS
profid, Appn AS app,
     Proj AS prj, Resp AS ra, Pcnt AS perct
     FROM CARSSQ.VOUCHER_CODING as Codetable
     WHERE CONTRACT_YEAR IN ('14','15','16','17');
QUIT;
```

# Query Processed on the DBMS (log)

```
9    LIBNAME CARSSQ ODBC DATASRC=FMS_PROD SCHEMA=DBO USER=&SYSUSERID. DBPROMPT=YES;
NOTE: Libref CARSSQ was successfully assigned as follows:
      Engine:        ODBC

11   <PROC SQL statements here>
ODBC: AUTOCOMMIT is NO for connection 0
ODBC: AUTOCOMMIT turned ON for connection id 0
DBMS_SELECT: SELECT * FROM "DBO"."VOUCHER_CODING"

ODBC_1: Prepared: on connection 0
SELECT * FROM "DBO"."VOUCHER_CODING"

ODBC: AUTOCOMMIT is NO for connection 1
ODBC: AUTOCOMMIT turned ON for connection id 1
DBMS_SELECT:  select distinct Codetable."year", Codetable."profid", Codetable."app",
Codetable."prj", Codetable."ra", Codetable."perct" from "DBO"."VOUCHER_CODING" Codetable where (Codetable."Contract_Year"
in ('14','15','16','17'))) Codetable

ODBC_2: Prepared: on connection 1
 select distinct Codetable."year", Codetable."profid", Codetable."app", Codetable."prj",
Codetable."ra", Codetable."perct" from ( select Codetable."Contract_Year" as "year", Codetable."Profile_ID" as "profid",
Codetable."Appn" as "app", Codetable."Proj" as "prj", Codetable."Resp" as "ra", Codetable."Pcnt" as "perct" from
"DBO"."VOUCHER_CODING" Codetable where (Codetable."Contract_Year" in ('14','15','16','17'))) Codetable

DEBUG: SQL Implicit Passthru stmt has been prepared successfully.

ODBC_3: Executed: on connection 1
Prepared statement ODBC_2

DEBUG: SQL Implicit Passthru stmt used for fetching data.

ACCESS ENGINE:  SQL statement was passed to the DBMS for fetching data.
NOTE: Table OLD.TABLE2 created, with 2406 rows and 13 columns.

17   QUIT;
NOTE: PROCEDURE SQL used (Total process t
      real time           5.80 seconds
      cpu time            0.21 seconds
```

SASTRACE

DEBUG= DBMS_SELECT

Same query…

# QUERY PROCESSED ON SAS

# Query Processed on SAS (script)

```
OPTIONS DEBUG=DBMS_SELECT SASTRACE=',,,d' SASTRACELOC=SASLOG
NOSTSUFFIX;
LIBNAME OLD 'L:\BFSPROC\CARSVOUCHERS';

LIBNAME CARSSQ ODBC DATASRC=FMS_PROD SCHEMA=DBO
USER=&SYSUSERID. DBPROMPT=YES;

PROC SQL NOIPASSTHRU;
     CREATE TABLE old.TABLE2 AS
     SELECT DISTINCT Contract_Year AS year, Profile_ID AS
profid, Appn AS app,
     Proj AS prj, Resp AS ra, Pcnt AS perct
     FROM CARSSQ.VOUCHER_CODING as Codetable
     WHERE CONTRACT_YEAR IN ('14','15','16','17');
QUIT;
```

# Query Processed on the DBMS (log)

```
9    LIBNAME CARSSQ ODBC DATASRC=FMS_PROD SCHEMA=DBO USER=&SYSUSERID. DBPROMPT=YES;
NOTE: Libref CARSSQ was successfully assigned as follows:
      Engine:        ODBC

11   <PROC SQL statements here>
ODBC: AUTOCOMMIT is NO for connection 0
ODBC: AUTOCOMMIT turned ON for connection id 0
DBMS_SELECT: SELECT * FROM "DBO"."VOUCHER_CODING"

ODBC_1: Prepared: on connection 0
SELECT * FROM "DBO"."VOUCHER_CODING"

DBMS_SELECT: SELECT  "Contract_Year", "Profile_ID", "Appn", "Proj", "Resp", "Pcnt  FROM
"DBO"."VOUCHER_CODING"  WHERE  (("Contract_Year" IN  ('14','15','16','17')))

ODBC_2: Prepared: on connection 0
SELECT  "Contract_Year", "Profile_ID", "Appn", "Proj", "Resp", "Pcnt"  FROM
"DBO"."VOUCHER_CODING"  WHERE  (("Contract_Year" IN  ('14','15','16','17')))

ODBC_3: Executed: on connection 0
Prepared statement ODBC_2
NOTE: Table OLD.TABLE2 created, with 2406 rows and 13 columns

17   QUIT;
NOTE: PROCEDURE SQL used (Total process time):
   real time           9.28 seconds
   cpu time            0.18 seconds
```

SASTRACE

DEBUG=
DBMS_SELECT

# Effect of Processing Location on Run Time

| Method | Where Processing Occurred | Real Time (seconds) |
|---|---|---|
| No distinct, no implicit pass-through | SAS | 11.1 |
| Distinct, no implicit pass-through | SAS | 9.3 |
| Distinct, implicit pass-through | DBMS | 5.8 |

# Summary

- The SAS/ACCESS Libname engine interface
  - Easy way to communicate with a DBMS
  - Relatively easy coding
  - Engine optimizes queries
  - Must know triggers to achieve SQL pass-through
    - Edit queries to allow them to pass-through to DBMS
  - Use System Options DEBUG and/or SASTRACE

- Where you process data matters!
  - Large DBMS may be more efficient than SAS
  - Large datasets = significant time/resource savings

# THANK YOU

Misty Johnson

Misty.Johnson@wi.gov