

SQL Processing with SAS

**Brent Burlingham –
Institutional Analysis,
U. of S.**

UNIVERSITY OF SASKATCHEWAN

Saskatoon, Saskatchewan, Canada.

Institutional Analysis www.usask.ca/ia

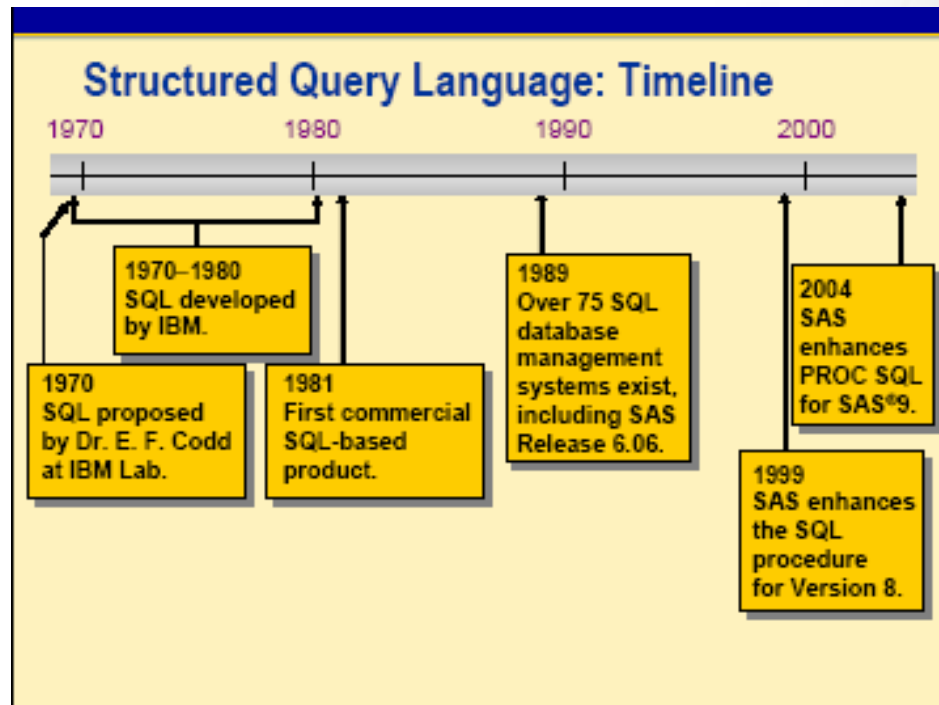


SQL – Structured Query Language

- *Standardized language widely used to retrieve and update data in tables and in views based on those tables.*
- *Originally designed as a query tool for relational databases, but is now used by many software products.*

Content used with permission from Live Web course SQL Processing with SAS®.

SQL – Structured Query Language



Content used with permission from Live Web course SQL Processing with SAS®.

Proc SQL

The SQL procedure uses SQL to:

- query SAS data sets*
- generate reports from SAS data sets*
- combine SAS data sets in many ways*
- create and delete SAS data files, views and indexes*
- update existing SAS data sets*

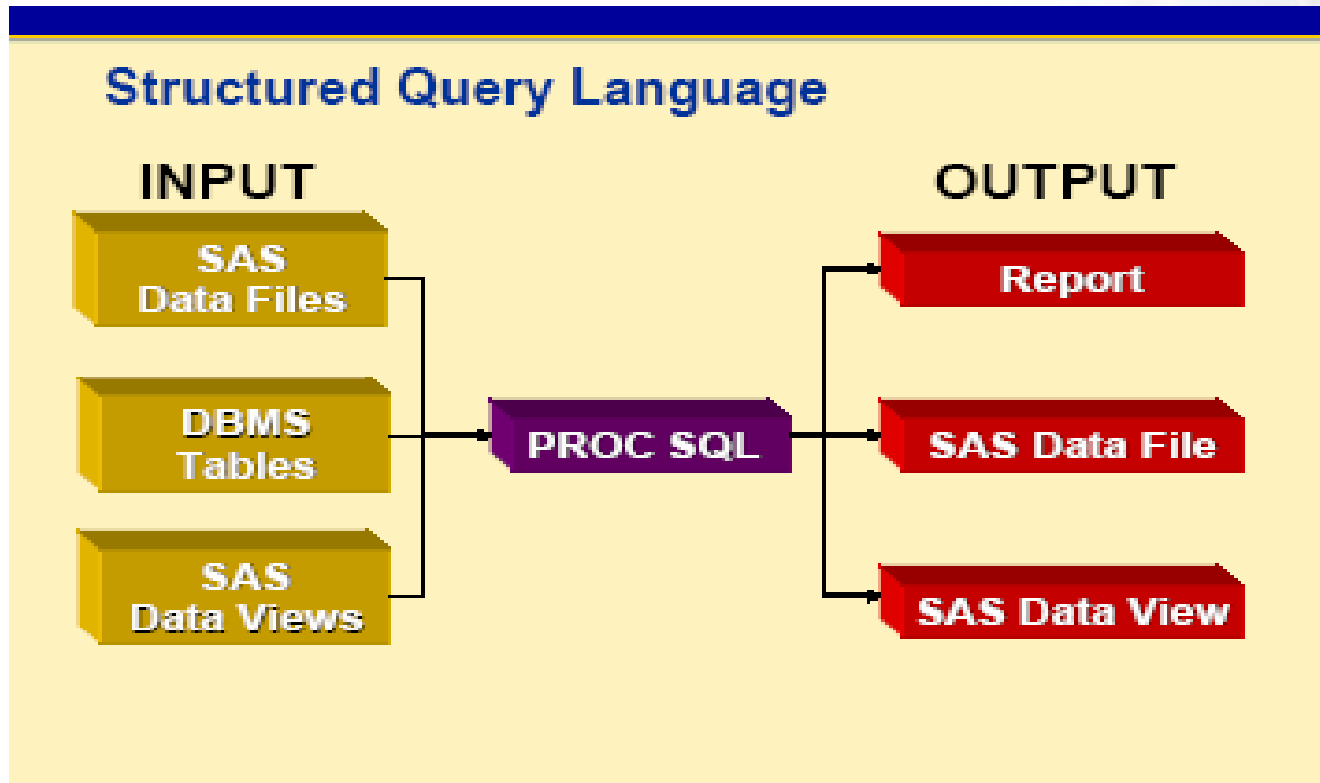
Content used with permission from Live Web course SQL Processing with SAS®.

Proc SQL - continued

- *enables you to use SQL within the SAS System*
- *follows guidelines set by the American National Standards Institute (ANSI)*
- *includes enhancements for compatibility with SAS software*
- *is part of Base SAS software*
- *can replace the need for multiple DATA and PROC steps with one query.*

Content used with permission from Live Web course SQL Processing with SAS®.

Proc SQL



Content used with permission from Live Web course SQL Processing with SAS®.

Proc SQL

IS NOT:

- *a replacement for the DATA step*
- *a custom reporting tool*

IS:

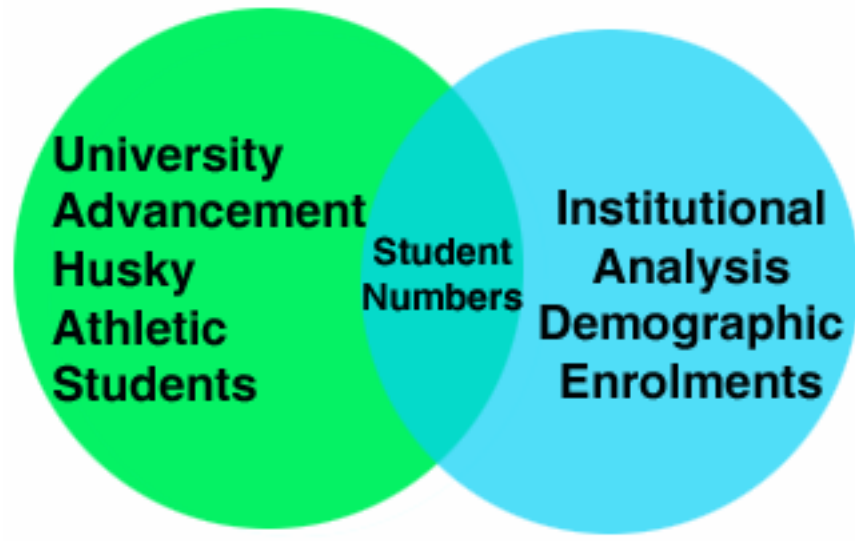
- *a tool for queries*
- *for data manipulation*
- *an augmentation to the DATA step*

Content used with permission from Live Web course SQL Processing with SAS®.

An Example of Proc SQL

- *University Advancement had identified a population of students involved with Husky Athletics, and wanted to provide demographic details*
- *Institutional Analysis had student demographics stored in iDat (an ongoing project to create a data warehouse of U. of S. activity for reporting)*

An Example of Proc SQL



University
Advancement –
Excel File

Institutional
Analysis –
MS SQL Server
Tables

An Example of Proc SQL

/* Import Advancement's Husky student file with UID, SNO and BID.
Only SNO is used to select data from iDAT's snapshot */

PROC IMPORT OUT= work.husksno

DATAFILE= "I:\Projects\AdHoc_Queries\Advancement\2005_2006_H
uskie_Athetics_Review\student ids for sports in ufriend.xls"

DBMS=EXCEL REPLACE;

SHEET="student id for ufriend sports\$";

GETNAMES=YES;

MIXED=NO;

SCANTEXT=YES;

USEDATE=YES;

SCANTIME=YES;

RUN;

An Example of Proc SQL

```
data work.husksno;  
  set work.husksno;  
    length sno2 $6;  
    length citizenship $30;  
    length country $30;  
    length highschool_country_prov $30;  
    length highschool_municipality $30;  
    length fsess $5;  
    length lsess $5;  
    length count_colleges_enrolled 8.;  
    length numsessi 8.;  
    length numsessso 8.;  
    length numsesss 8.;  
    length numsesssw 8.;  
/* Text SNO version required for DEMOGRAPHIC_ENROLMENTS (source version  
SNO is numeric) */  
  sno2 = SNO;  
run;
```

An Example of Proc SQL

/* Select required fields and counts from work.DEMOGRAPHIC_ENROLMENTS based on SNO (converted to sno2) in provided data file */

proc sql;

update work.husksno as t

set fsess = (select min(distinct first_enrol_acad_sess_code) from work.DEMOGRAPHIC_ENROLMENTS
where student_number = t.sno2
and academic_session_code = (select max(academic_session_code) from

work.DEMOGRAPHIC_ENROLMENTS

where student_number = t.sno2

)

)

,
lssess = (select max(distinct last_enrol_acad_sess_code) from work.DEMOGRAPHIC_ENROLMENTS

where student_number = t.sno2
and academic_session_code = (select max(academic_session_code) from

work.DEMOGRAPHIC_ENROLMENTS

where student_number = t.sno2

)

)

An Example of Proc SQL

```
numsessi = (select count(distinct academic_session_code) from work.DEMOGRAPHIC_ENROLMENTS  
           where (student_number = t.sno2) and (Academic_session_char = 'I')  
           )
```

```
numsesso = (select count(distinct academic_session_code) from work.DEMOGRAPHIC_ENROLMENTS  
           where (student_number = t.sno2) and (Academic_session_char = 'O')  
           )
```

```
numsesss = (select count(distinct academic_session_code) from work.DEMOGRAPHIC_ENROLMENTS  
           where (student_number = t.sno2) and (Academic_session_char = 'S')  
           )
```

```
numsessw = (select count(distinct academic_session_code) from work.DEMOGRAPHIC_ENROLMENTS  
           where (student_number = t.sno2) and (Academic_session_char = 'W')  
           )
```

An Example of Proc SQL

```
,  
citizenship = (select distinct Citizenship_descr from work.DEMOGRAPHIC_ENROLMENTS  
               where (student_number = t.sno2 and Run_month = 'Yen')  
               and academic_session_code = (select min(academic_session_code) from  
work.DEMOGRAPHIC_ENROLMENTS
```

```
where student_number = t.sno2
```

```
)
```

```
)
```

```
,  
country = (select distinct Citizenship_country_name from work.DEMOGRAPHIC_ENROLMENTS  
           where (student_number = t.sno2 and Run_month = 'Yen')  
           and academic_session_code = (select min(academic_session_code) from  
work.DEMOGRAPHIC_ENROLMENTS
```

```
where student_number = t.sno2
```

```
)
```

```
)
```

An Example of Proc SQL

```
highschool_country_prov = (select distinct Matric_province from work.DEMOGRAPHIC_ENROLMENTS  
where (student_number = t.sno2 and Run_month = 'Yen')  
and academic_session_code = (select min(academic_session_code) from
```

```
work.DEMOGRAPHIC_ENROLMENTS
```

```
where student_number = t.sno2
```

```
)
```

```
)
```

```
highschool_municipality = (select distinct Matric_highschool_city from  
work.DEMOGRAPHIC_ENROLMENTS
```

```
where (student_number = t.sno2 and Run_month = 'Yen')
```

```
and academic_session_code = (select min(academic_session_code) from
```

```
work.DEMOGRAPHIC_ENROLMENTS
```

```
where student_number = t.sno2
```

```
)
```

```
)
```

```
count_colleges_enrolled = (select count(distinct college_name) from work.DEMOGRAPHIC_ENROLMENTS  
where (student_number = t.sno2) and (Run_month = 'Yen')
```

```
);
```

An Example of Proc SQL

/* Write output data set to an Excel file. */

PROC EXPORT DATA= WORK.Husksno

OUTFILE= "I:\Projects\AdHoc_Queries\Advancement\2005_2006_Huskie_Athetics_Review\husky_demog.xls"

DBMS=EXCEL REPLACE;

SHEET="husky_demog";

RUN;

Resultant data file captures student numbers for Husky Athletic students with Citizenship, Country of Citizenship, Province or Country of Matriculation, First Session Enrolled, Last Session Enrolled, Number of Colleges Enrolled In, Counts of I, O, S and W Sessions Enrolled In.

Comparison of SQL with Traditional SAS Programming

Handling a Complex Query

You can also solve this problem by using a multiway join.

```
select distinct e.FirstName, e.LastName
  from airline.flightschedule as a,
       airline.staffmaster as b,
       airline.payrollmaster as c,
       airline.supervisors as d,
       airline.staffmaster as e
 where a.Date='04mar2000'd and
       a.Destination='CPH' and
       a.EmpID=b.EmpID and
       a.EmpID=c.EmpID and
       d.JobCategory=substr(c.JobCode,1,2)
 and d.State=b.State and
       d.EmpID=e.EmpID;
```

Content used with permission from Live Web course SQL Processing with SAS®.

Comparison of SQL with Traditional SAS Programming

Perform the same task using traditional SAS programming.

```
/* Find the crew for the flight. */  
  
proc sort data=airline.flightschedule (drop=flightnumber)  
    out=crew (keep=empid);  
    where destination='CPH' and date='04MAR2000'd;  
    by empid;  
run;  
  
/* Find the State and job code for the crew. */  
  
proc sort data=airline.payrollmaster (keep=empid jobcode)  
    out=payroll;  
    by empid;  
run;  
  
proc sort data=airline.staffmaster  
    (keep=empid state firstname lastname)  
    out=staff;  
    by empid;  
run;  
  
data st cat (keep=state jobcategory);  
    merge crew (in=c)  
        staff  
        payroll;  
    by empid;  
    if c;  
    jobcategory=substr(jobcode,1,2);  
run;  
  
/* Find the supervisor IDs. */  
  
proc sort data=st cat;  
    by jobcategory state;  
run;  
  
proc sort data=airline.supervisors  
    out=superv;  
    by jobcategory state;  
run;
```

Content used with permission from Live Web course SQL Processing with SAS®.

Comparison of SQL with Traditional SAS Programming

```
data super (keep=empid);
  merge st cat(in=s)
        superv;
  by jobcategory state;
  if s;
run;

/* Find the names of the supervisors. */

proc sort data=super;
  by empid;
run;

data names(drop=empid);
  merge super (in=super)
        staff (keep=empid firstname lastname);
  by empid;
  if super;
run;

proc print data=names noobs uniform;
run;
```

Content used with permission from Live Web course SQL Processing with SAS®.

Choosing Between SQL Joins and DATA Step Merges

- *For ad hoc queries, select the method that you can code in the shortest time.*
- *For production jobs, experiment with different coding techniques and evaluate performance statistics.*

Content used with permission from Live Web course SQL Processing with SAS®.

SQL Processing with SAS

- *Highly recommend SAS Live Web Course SQL Processing with SAS[®].*
- *4 x 3.5 hour sessions May 2, June 26.*
- *For information see:
<http://support.sas.com/lw>*

SQL Processing with SAS

- *Many books available through SAS Publishing Bookstore:*
- *<http://www.sas.com/apps/pubscat/welcome.jsp>*

SQL Processing with SAS

→ *Questions?*