Tutorial on SQL and Data Federation for Genomics Researchers

a presentation by

Kirk Paul Lafler
SAS® Consultant, Author, and Trainer
E-mail: KirkLafler@cs.com

sas Certified Professional Version 6

Objectives

- What is data federation?
- Characteristics associated with data federation
- What is a join?
- Why join?
- SAS® DATA step merge versus a Join
- Cartesian product joins
- Two table joins
- Table aliases to reference tables
- Three table joins
- Left and Right outer joins
- What happens during a join?
- Available join algorithms

Tables Used in Examples

	Title	Length	Category	Year	Studio	Rating
1	Brave Heart	177	Action Adventure	1995	Paramount Pictures	R
2	Casablanca	103	Drama	1942	MGM / UA	PG
3	Christmas Vacation	97	Comedy	1989	Warner Brothers	PG-13
4	Coming to America	116	Comedy	1988	Paramount Pictures	R
5	Dracula	130	Horror	1993	Columbia TriStar	R
6	Dressed to Kill	105	Drama Mysteries	1980	Filmways Pictures	R
7	Forrest Gump	142	Drama	1994	Paramount Pictures	PG-13
8	Ghost	127	Drama Romance	1990	Paramount Pictures	PG-13
9	Jaws	125	Action Adventure	1975	Universal Studios	PG
10	Jurassic Park	127	Action	1993	Universal Pictures	PG-13
11	Lethal Weapon	110	Action Cops & Robber	1987	Warner Brothers	R
12	Michael	106	Drama	1997	Warner Brothers	PG-13
13	National Lampoon's Vacation	98	Comedy	1983	Warner Brothers	PG-13
14	Poltergeist	115	Horror	1982	MGM / UA	PG
15	Rocky	120	Action Adventure	1976	MGM / UA	PG
16	Scarface	170	Action Cops & Robber	1983	Universal Studios	R
17	Silence of the Lambs	118	Drama Suspense	1991	Orion	R
18	Star Wars	124	Action Sci-Fi	1977	Lucas Film Ltd	PG
19	The Hunt for Red October	135	Action Adventure	1989	Paramount Pictures	PG
20	The Terminator	108	Action Sci-Fi	1984	Live Entertainment	R
21	The Wizard of Oz	101	Adventure	1939	MGM / UA	G
22	Titanic	194	Drama Romance	1997	Paramount Pictures	PG-13

	Title	Actor_Leading	Actor_Supporting
1	Brave Heart	Mel Gibson	Sophie Marceau
2	Christmas Vacation	Chevy Chase	Beverly D'Angelo
3	Coming to America	Eddie Murphy	Arsenio Hall
4	Forrest Gump	Tom Hanks	Sally Field
5	Ghost	Patrick Swayze	Demi Moore
6	Lethal Weapon	Mel Gibson	Danny Glover
7	Michael	John Travolta	Andie MacDowell
8	National Lampoon's Vacation	Chevy Chase	Beverly D'Angelo
9	Rocky	Sylvester Stallone	Talia Shire
10	Silence of the Lambs	Anthony Hopkins	Jodie Foster
11	The Hunt for Red October	Sean Connery	Alec Baldwin
12	The Terminator	Arnold Schwarzenegge	Michael Biehn
13	Titanic	Leonardo DiCaprio	Kate Winslet

What is Data Federation?

- Process of integrating data from many different sources
- Leaves data in place without using resources to copy data
- Makes access to data sources as easy as possible
- Provides a degree of reusability
- A data federation approach often replaces a data warehouse

Data Federation Characteristics

- It represents an integration approach
- Ability to aggregate data from many different sources
- Data sources can be in any location
- It can be implemented within any lifecycle methodology
- Provides flexibility
- It is user-centric
- Federated data does not copy and store data like a data warehouse
- It contains metadata (information about the actual data and its location)
- Is NOT always designed with optimality in mind

What is a Join?

- Process of combining tables side-by-side (horizontally)
- Consists of a matching process between rows in tables
- Some or all of the tables' contents are brought together
- Gather and manipulate data from across tables

Visually, it would look something like this:

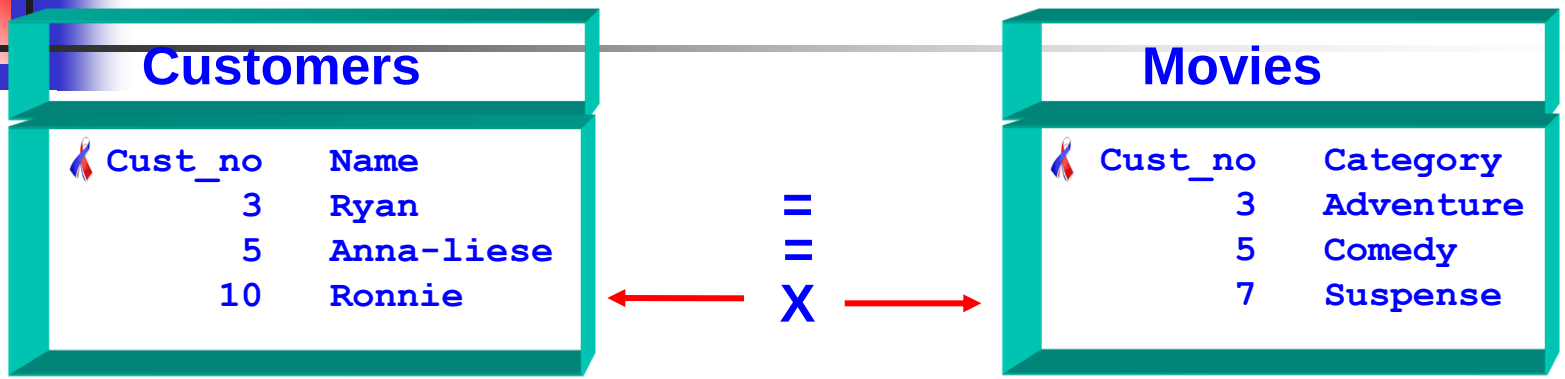
Why Join?

- Data in a database is often stored in separate tables
- Joins allow data to be combined as if it were stored in one huge file
- Provide exciting insights into data relationships
- Types of joins:
  - ✓ Inner joins – a maximum of 256 tables can be joined
  - ✓ Outer joins – a maximum of 2 tables can be joined

DATA Step Merge versus a Join

- Merges process data differently than a standard join
- The merge process overlays the duplicate by-column
- Joins adhere to ANSI guidelines
- Joins do not automatically overlay the duplicate matching column

DATA Step Merge Process

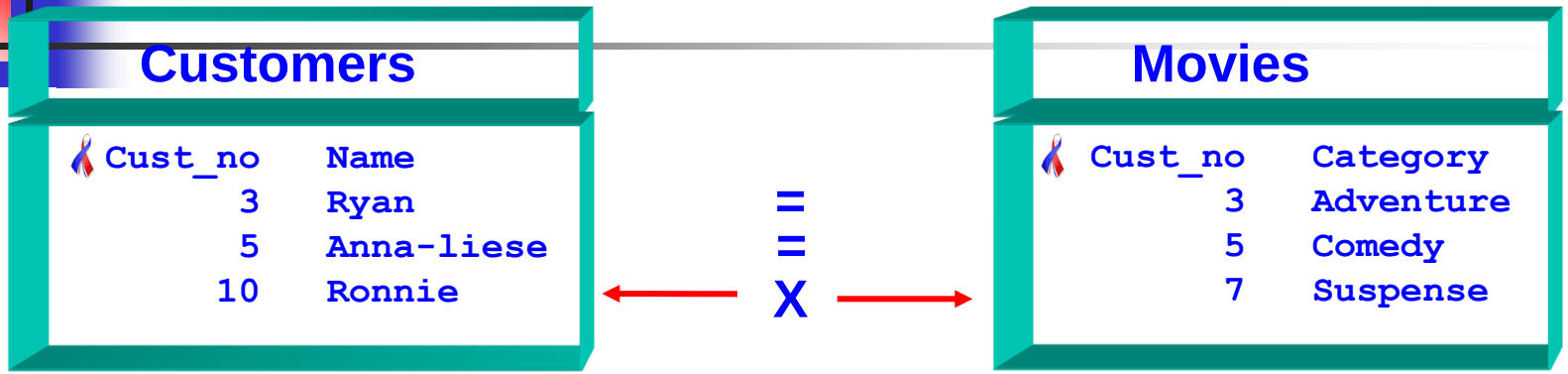



```
DATA merged;
  MERGE customers (IN=c)
        movies (IN=m);
  BY cust_no;
  IF c AND m;
RUN;
```

Merged

Cust_no	Name	Category
3	Ryan	Adventure
5	Anna-liese	Comedy

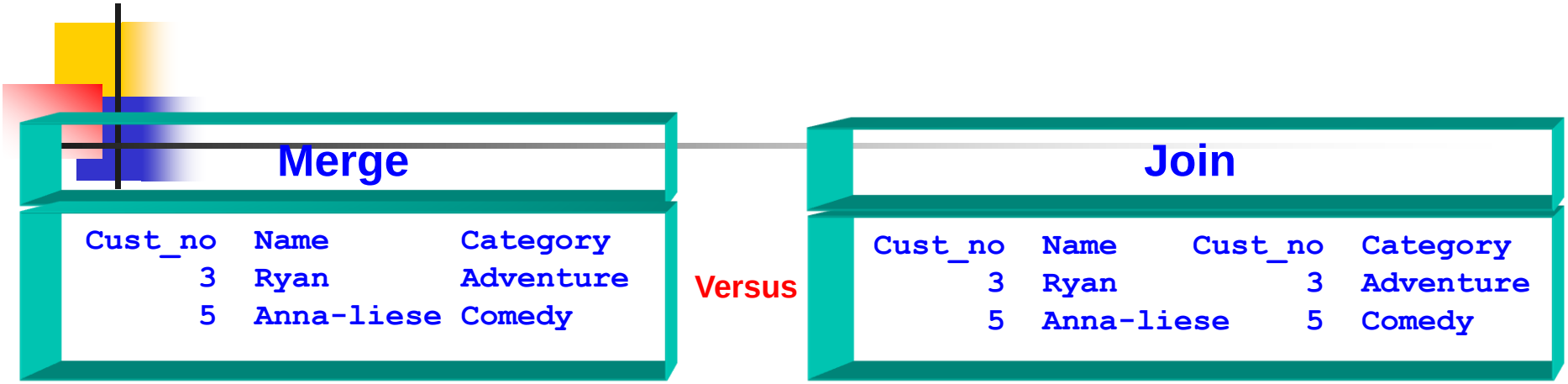
Join Process




```
PROC SQL;
  SELECT *
  FROM customers,
       movies
  WHERE
     customers.cust_no =
        movies.cust_no;
QUIT;
```

Cust_no	Name	Cust_no	Category
3	Ryan	3	Adventure
5	Anna-liese	5	Comedy

Merge versus Join Results



Features

1. Data must be sorted using by-value.
2. Requires variable names to be same.
3. Duplicate matching column is overlaid.
4. Results are not automatically printed.

Features

1. Data does not have to be sorted using by-value.
2. Does not require variable names to be same.
3. Duplicate matching column is not overlaid.
4. Results are automatically printed unless NOPRINT option is specified.

Cartesian Product Join (Cross Join)

Customers

Cust_no	Name
3	Ryan
5	Anna-liese
10	Ronnie

No WHERE or Key Used

Movies

Cust_no	Category
3	Adventure
5	Comedy
7	Suspense

Result represents all possible combinations of rows and columns

```
PROC SQL;
  SELECT *
  FROM customers,
       movies;
QUIT;
```

Cust_no	Name	Cust_no	Movie_no	Category
3	Ryan	3	1011	Adventure
3	Ryan	5	3090	Comedy
3	Ryan	7	4456	Suspense
5	Anna-liese	3	1011	Adventure
5	Anna-liese	5	3090	Comedy
5	Anna-liese	7	4456	Suspense
10	Ronnie	3	1011	Adventure
10	Ronnie	5	3090	Comedy
10	Ronnie	7	4456	Suspense

Absence of WHERE clause produces a Cartesian product

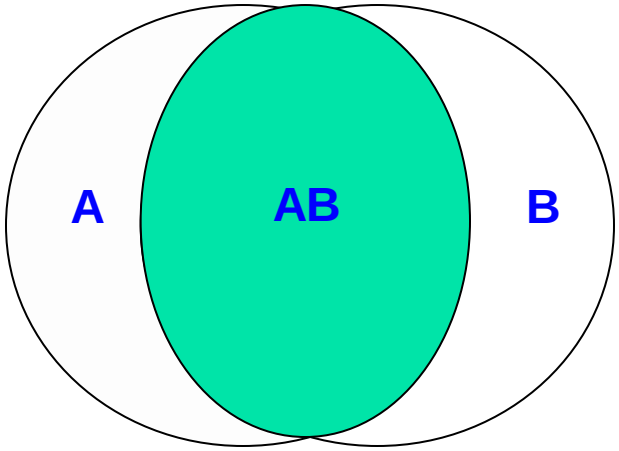
Example – Cartesian Product

*A **Cartesian product join**, sometimes referred to as a cross join, can be very large because it represents all the possible combinations of rows and columns.*

```
PROC SQL;
  SELECT  *
     FROM MOVIES,
          ACTORS;
QUIT;
```

Equi-Join with Two Tables

*The result of an **Equi-join** is illustrated by the shaded area (AB) in the Venn diagram.*



Equi-Join with Two Tables

Customers

Cust_no	Name
3	Ryan
5	Anna-liese
10	Ronnie

=

Equi Join

Movies

Cust_no	Category
3	Adventure
5	Comedy
7	Suspense

```
PROC SQL;
 SELECT *
  FROM customers,
       movies
   WHERE
    customers.cust_no =
       movies.cust_no;
QUIT;
```

Cust_no	Name	Cust_no	Movie_no	Category
3	Ryan	3	1011	Adventure
5	Anna-liese	5	3090	Comedy

Example – WHERE-clause

*The most reliable way to join two tables together, and to avoid creating a Cartesian product, is to use a **WHERE** clause with common columns or keys.*

```
PROC SQL;
 SELECT  MOVIES.TITLE, RATING, ACTOR_LEADING
  FROM MOVIES,
       ACTORS
   WHERE MOVIES.TITLE = ACTORS.TITLE;
QUIT;
```

Table Aliases

Customers

Cust_no	Name
3	Ryan
5	Anna-liese
10	Ronnie

=

Equi Join

Movies

Cust_no	Category
3	Adventure
5	Comedy
7	Suspense

```
PROC SQL;
 SELECT *
  FROM customers c,
       movies m
   WHERE c.cust_no =
         m.cust_no;
QUIT;
```

Aliases

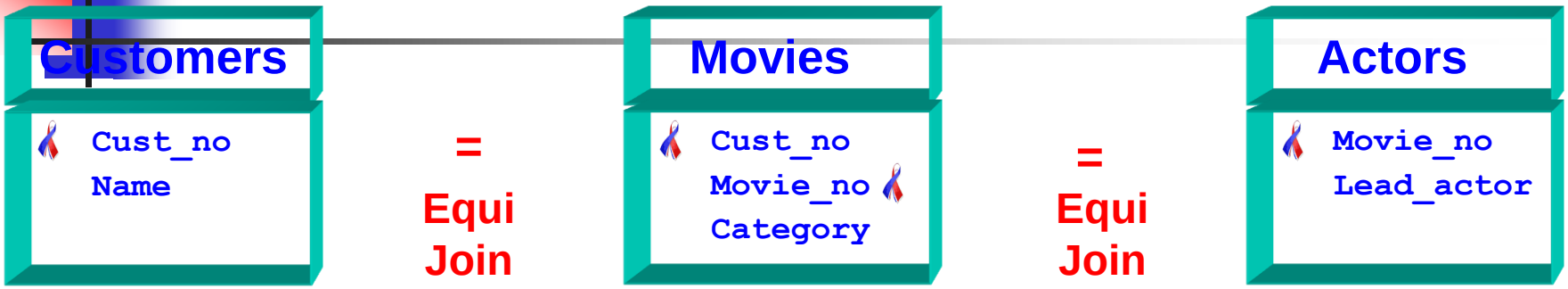
Cust_no	Name	Cust_no	Movie_no	Category
3	Ryan	3	1011	Adventure
5	Anna-liese	5	3090	Comedy

Example – Table Alias

*Assigning a **table alias** is a not only a useful way to reference a table, but can reduce the number of keystrokes typed.*

```
PROC SQL;
 SELECT M.TITLE, RATING, ACTOR_LEADING
  FROM MOVIES  M,
       ACTORS  A
   WHERE M.TITLE = A.TITLE;
QUIT;
```

Joining Three Tables

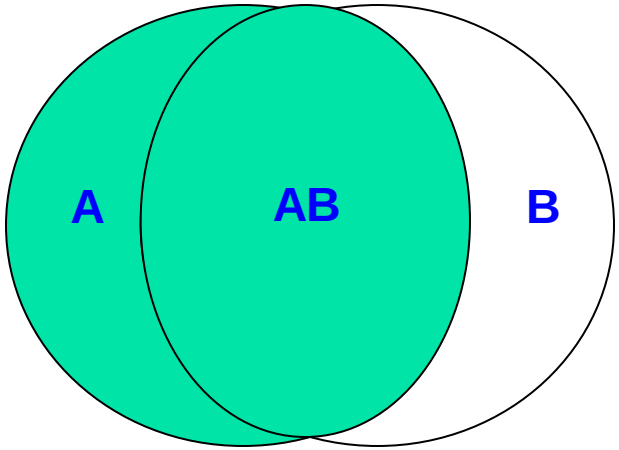



```
PROC SQL;
  SELECT c.cust_no, c.name,
     m.movie_no, c.category,
     a.lead_actor
  FROM customers c,
       movies m,
       actors a
  WHERE
    c.cust_no  = m.cust_no AND
    m.movie_no = a.movie_no;
QUIT;
```

Cust_no	Name	Movie_no	Category	Lead_actor
3	Ryan	1011	Adventure	Mel Gibson

Left Outer Joins

*The result of a **Left Outer join** is illustrated by the shaded areas (A and AB) in the Venn diagram.*



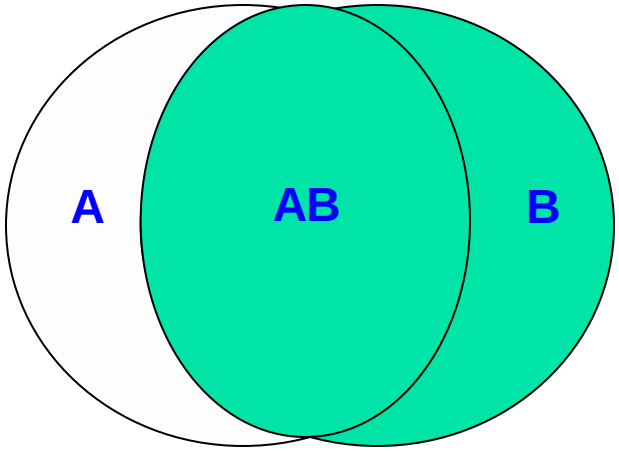
Example – Left Outer Join

*The result of a **Left Outer join** produces both matched rows from both tables plus any unmatched rows from the left table.*

```
PROC SQL;
 SELECT MOVIES.TITLE, RATING, ACTOR_LEADING
  FROM MOVIES LEFT JOIN
       ACTORS
   ON MOVIES.TITLE = ACTORS.TITLE;
QUIT;
```

Right Outer Joins

*The results of a **Right Outer join** is illustrated by the shaded areas (B and AB) in the Venn diagram.*



Example – Right Outer Join

*The result of a **Right Outer join** produces matched rows from both tables while preserving unmatched rows from the right table.*

```
PROC SQL;
 SELECT MOVIES.TITLE, RATING, ACTOR_LEADING
  FROM MOVIES RIGHT JOIN
       ACTORS
   ON MOVIES.TITLE = ACTORS.TITLE;
QUIT;
```

What Happens during a Join?

When joining two tables:

- An intermediate Cartesian product is built from the two tables
- Rows are selected that match the WHERE clause, if present

When joining more than two tables:

- SQL query optimizer evaluates the available methods for retrieving the data and attempts to use the most efficient method
- The join is reconstructed into several two-way joins
- Removes unwanted rows and columns from the intermediate tables
- Determines the order of processing to reduce the size of the intermediate Cartesian product

Join Algorithms

Users supply the list of tables for joining along with the join conditions, and the PROC SQL optimizer determines which join algorithm to use for performing the join. The algorithms include:

Join Algorithms

Users supply the list of tables for joining along with the join conditions, and the PROC SQL optimizer determines which of the join algorithms to use for performing the join. The algorithms include:

- ✓ **Nested Loop Join (brute-force join)** – When an equality condition is not specified, a read of the complete contents of the right table is processed for each row in the left table.

Nested Loop Join - Features

- Used with join relations of two tables
- One or both of the tables is relatively small
- I/O intensive
- This join generally performs fairly well with smaller tables, but generally performs poorly with larger join relations

Join Algorithms

Users supply the list of tables for joining along with the join conditions, and the PROC SQL optimizer determines which of the join algorithms to use for performing the join. The algorithms include:

✓ Nested Loop Join– When an equality condition is not specified, a read of the complete contents of the right table is processed for each row in the left table.

✓ **Sort-Merge Join** – When the specified tables are already in the desired sort order, resources are not expended for resorting.

Sort-Merge Join - Features

- Used with joins of two tables
- Works best when one or both of the join relations are in the desired order
- One or both of the tables are of moderate size
- If the optimizer determines a sort is not needed – no sort will be performed, otherwise sort resources are expended:
  - using an explicit sort operation <or>
  - by taking advantage of pre-existing ordering
- Generally performs well, particularly when the majority of the rows are being joined

Join Algorithms

Users supply the list of tables for joining along with the join conditions, and the PROC SQL optimizer determines which of the join algorithms to use for performing the join. The algorithms include:

✓ Nested Loop Join– When an equality condition is not specified, a read of the complete contents of the right table is processed for each row in the left table.

✓ Sort-Merge Join – When the specified tables are already in the desired sort order, resources are not expended for resorting.

✓ **Indexed Join – When an index exists on >=1 variable(s) to represent a key, matching rows may be accessed using the index.**

Indexed Join - Features

- Used with joins of two tables
- An index must be defined that produces a small subset of the total number of rows in a table
- Matching rows are accessed directly using the index
- One or both of the tables are of moderate to large size
- Generally performs well, particularly when a small number of rows are being joined

Join Algorithms

Users supply the list of tables for joining along with the join conditions, and the PROC SQL optimizer determines which of the join algorithms to use for performing the join. The algorithms include:

✓ Nested Loop Join– When an equality condition is not specified, a read of the complete contents of the right table is processed for each row in the left table.

✓ Sort-Merge Join – When the specified tables are already in the desired sort order, resources are not expended for resorting.

✓ Indexed Join – When an index exists on >=1 variable(s) to represent a key, matching rows may be accessed using the index.

✓ **Hash Join – When an equality relationship exists and the smaller table is able to fit in memory, set-matching operations generally perform well.**

Hash Join - Features

- Used with joins of two tables
- The SQL optimizer attempts to estimate the amount memory required to build a hash table in memory
- A hash table data structure associates keys with values
- The optimizer tries to use the smaller of the tables as the hash table
- Its purpose is to perform more efficient lookups
- Requires an equi-join predicate
- This algorithm generally performs well with small to medium join relations

Options MSGLEVEL=I

By specifying the MSGLEVEL=I option, helpful notes describing index usage, sort utilities, and merge processing are displayed on the SAS Log.

```
OPTIONS MSGLEVEL=I;
PROC SQL;
  SELECT  MOVIES.TITLE, RATING, LENGTH, ACTOR_LEADING
    FROM MOVIES,
         ACTORS
      WHERE MOVIES.TITLE = ACTORS.TITLE AND
            RATING = 'PG';
QUIT;
```

MSGLEVEL=I Log

SAS Log Results

```
OPTIONS MSGLEVEL=I;
PROC SQL;
   SELECT  MOVIES.TITLE, RATING, LENGTH, ACTOR_LEADING
      FROM MOVIES,
              ACTORS
         WHERE MOVIES.TITLE = ACTORS.TITLE AND
                  RATING = 'PG';
```

INFO: Index Rating selected for WHERE clause optimization.

```
QUIT;
```

_METHOD Option

A **_METHOD** *option can be specified on the PROC SQL statement to display the hierarchy of processing that takes place. Results are displayed on the Log.*

Codes	Description
sqxcrta	Create table as Select
sqxslct	Select
sqxjsl	Step loop join (Cartesian)
sqxjm	Merge join
sqxjndx	Index join
sqxjhsh	Hash join
sqxsort	Sort
sqxsrc	Source rows from table
sqxfil	Filter rows
sqxsumg	Summary statistics with GROUP BY
sqxsumn	Summary statistics with no GROUP BY

Program Example using _METHOD

```
PROC SQL _METHOD;
  TITLE '2-Way Equi Join';
  SELECT MOVIES.TITLE, RATING, ACTOR_LEADING
    FROM MOVIES,
         ACTORS
      WHERE MOVIES.TITLE = ACTORS.TITLE;
QUIT;
```

SAS Log Results

```
NOTE: SQL execution methods chosen are:
    sqxslct
      sqxjhsh
        sqxsrc( MOVIES )
        sqxsrc( ACTORS )
```

Conclusion

- Data federation characteristics
- A merge and join do not process data the same way
- A join combines tables side-by-side (horizontally)
- Joins adhere to ANSI guidelines
- Cartesian product represents all possible combinations of rows from the underlying tables
- 256 tables can be joined using an inner join construct
- 2 tables can be joined using an outer join construct
- Four types of join algorithms:
  - ✓ Nested loop join
  - ✓ Sort-Merge join
  - ✓ Indexed join
  - ✓ Hash join

PROC SQL

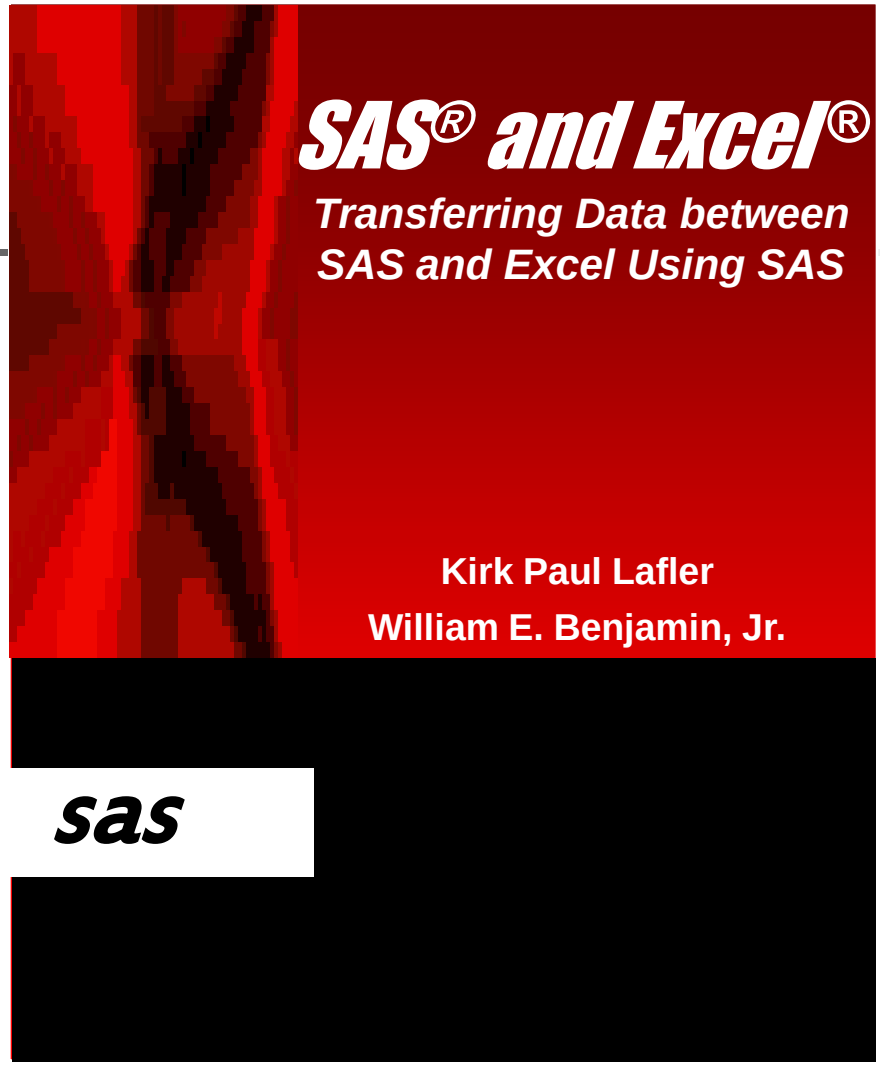
Beyond the Basics Using SAS

Kirk Paul Lafler

sas

PROC SQL Examples Book

Available at www.sas.com!



A Book of Data Transfer Methods with Examples

Coming in September 2010!

Thank you for attending!

Questions ?

Joining
Worlds of Data
is exciting and fun!

Kirk Paul Lafler
SAS® Consultant, Author, and Trainer
Software Intelligence Corporation
E-mail: KirkLafler@cs.com

sas Certified Professional Version 6

Advances in Bioinformatics and Genomics Symposium

We kindly thank the WUSS Leadership and the following organizations for their assistance and support for this symposium:

Genentech
IN BUSINESS FOR LIFE

COMSYS

sas THE POWER TO KNOW.

ADVANCED CLINICAL SERVICES LLC
Clinical Research Staffing Experts

jmp™
Statistical Discovery.™ From SAS.

genomic® health
Turning the promise of genomics into the practice of medicine™

Western Users of SAS Software

BASAS

TMC Software, Inc. WE MAKE YOUR DATA USEFUL.

Mark your calendars for WUSS 2010!

Three days of **classes**, **workshops** and **presentations** introducing the latest in SAS® technology and applications.

November 3-5

Hyatt Regency Mission Bay

San Diego, CA

More information will be available at **www.wuss.org**