Fun with SQL – Again ☺

Paul Kent

BigData @ SAS

http://about.me/paul.kent

lmgtfy “paul kent SAS”

paulmkent

@hornpolish

New Hill, North Carolina
Y’all

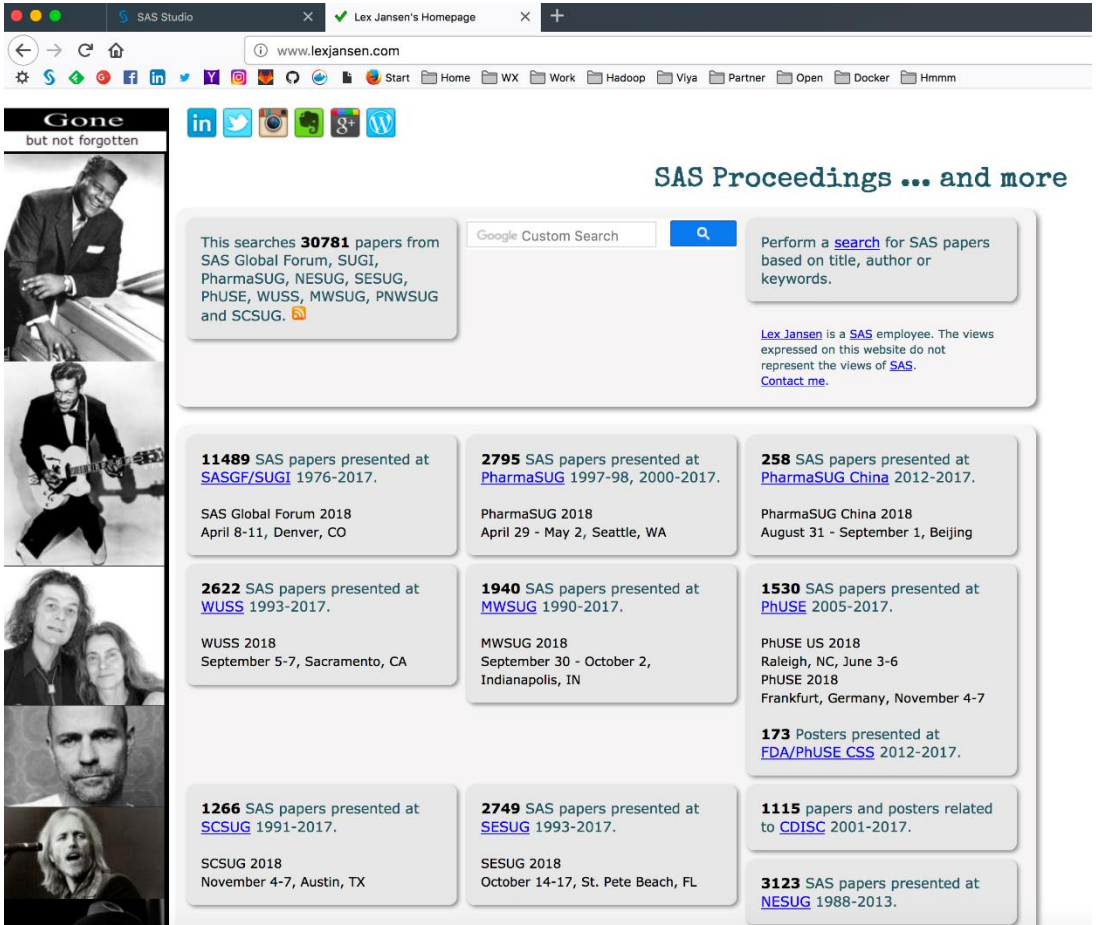
Johannesburg, South Africa
Julle

Fareham, England
?

Agenda

- What is SQL
- Skip this session? See laxjansen.com/search ☺
- Joining Stuff. What could go wrong?
- Huh? never thought of that. Cool SQL tricks

http://www.lexjansen.com



What is SQL?

SQL – Structured Query Language

- https://en.wikipedia.org/wiki/SQL
- ESS-kew-EL? SEE-kwel?
- Domain Specific Language. take that hipsters!
- Based on Relational Algebra and Tuple Relational Calculus
- Declarative… Say What You Want. don't say How to get it.
- The SQL procedure in SAS
  - an implementation of ANSI Standard Structured Query Language (SQL)
  - facilitates extracting data from multiple sources
  - with simple and concise queries.

Joins

- Match this pile of “stuff” with that pile of “stuff”
- How many different ways can this go wrong

SQL sentence structure

```
Select <columns>
  From <tables>
 Where <they match like this>
   and <they look like this>
   and <this value> is in <some other list>

 group by <them like this>
 order by <this>, <that>
     ;
```

Evil Online Retailer
(hypothetical of course)

People:

GUID	Name	Sex
99	Paul	M
78	Denise	F

Purch:

GUID	Date	Item	Qty
99	11/02	PhoneX	1
99	11/08	PhCaseX	1

Click:

GUID	Date	Page
99	10/28	Phone8
101	10/29	PhoneX
78	11/01	PhCases

Who bought stuff?

```
Select *
  From People, Purch
    ;
```

guid	name	sex
99	Paul	M
78	Denise	F

guid	item	qty	date
99	PhoneX	1	02NOV17
99	PhCaseX	1	08NOV17

guid	name	sex	guid	item	qty	date
99	Paul	M	99	PhoneX	1	02NOV17
99	Paul	M	99	PhCaseX	1	08NOV17
78	Denise	F	99	PhoneX	1	02NOV17
78	Denise	F	99	PhCaseX	1	08NOV17

Who bought stuff?

```
Select *
  From People P, Purch PR
 Where P.guid = PR.guid
     ;
```

guid	name	sex	guid	item	qty	date
99	Paul	M	99	PhoneX	1	02NOV17
99	Paul	M	99	PhCaseX	1	08NOV17

Who bought stuff?

```
Select P.*,
       PR.date, PR.item, PR.qty
  From People P, Purch PR
 Where P.guid = PR.guid
     ;
```

guid	name	sex	date	item	qty
99	Paul	M	02NOV17	PhoneX	1
99	Paul	M	08NOV17	PhCaseX	1

Data Model Counts

- SQL is pretty good when your data is “sound”
- Context Matters – SQL grew up in an era when IT spent hugely on requirements analysis and entity-relationship diagrams for the data they were about to gather.
  - Computers were very expensive!
  - People? We got lots of those.
- But what about harder Questions..
  - “Who does, but also ***who could*** buy from our evil empire?”

SQL and SAS came from different places

- SQL: Master Detail is predominant.
- In fact the database that came before us (IMS, hierarchical database) did not even allow for detail record under “no” parent
- SAS: really? you’ve never worked in real like have you?
- SQL: missing values are kinda formal. NULL doesnt even match NULL
- SAS: missing values happen all the time. they are “just like“ measured values

Who did and did-not buy stuff?

```
Title "Candidate Pool";
Select P.*,
       PR.date, PR.item, PR.qty
  From People P LEFT JOIN Purch PR
    ON P.guid = PR.guid
     ;
```

Candidate Pool

guid	name	sex	date	item	qty
78	Denise	F	.		.
99	Paul	M	02NOV17	PhoneX	1
99	Paul	M	08NOV17	PhCaseX	1

Evil Online Retailer
(hypothetical of course)

People:

GUID	Name	Sex
99	Paul	M
78	Denise	F

Purch:

GUID	Date	Item	Qty
99	11/02	PhoneX	1
99	11/08	PhCaseX	1

Click:

GUID	Date	Page
99	10/28	Phone8
101	10/29	PhoneX
78	11/01	PhCases

Who clicked looking to buy stuff?

```
Title "Just Looking?";
Select P.*,
       C.date, C.page
  From People P LEFT JOIN Click C
    ON P.guid = C.guid
     ;
```

Just Looking?

guid	name	sex	date	page
78	Denise	F	04NOV17	http://evil.com/store/PhCases.htm
99	Paul	M	28OCT17	http://evil.com/store/Phone8.htm

Who clicked looking to buy stuff?

```
Title "Just Looking?";
Select P.*,
       C.date, C.page
  From People P FULL JOIN Click C
    ON P.guid = C.guid
     ;
```

guid	name	sex	date	page
78	Denise	F	04NOV17	http://evil.com/store/PhCases.htm
99	Paul	M	28OCT17	http://evil.com/store/Phone8.htm
.			29OCT17	http://evil.com/store/PhoneX.htm

Who clicked looking to buy stuff?

```
Select COALESCE(P.guid, C.guid) as guid,
       P.name, P.sex,
       C.date, C.page
  From People P FULL JOIN Click C
    ON P.guid = C.guid
     ;
```

guid	name	sex	date	page
78	Denise	F	04NOV17	http://evil.com/store/PhCases.htm
99	Paul	M	28OCT17	http://evil.com/store/Phone8.htm
101			29OCT17	http://evil.com/store/PhoneX.htm

JOINS :: Harder that they first appear

- Pay attention to the structure of the data (especially the matching keys)
  - “normal”, LEFT, RIGHT or FULL join?
  - FULL is computationally more expensive

- T.* is good for ad-hoc
- T.VAR1, T.VAR2, T.VAR3 is good for production

- COALESCE(your, keys)

JOINS? Is that all you've got?

- SQL SET OPERATORS: SO HANDY VENN YOU NEED THEM
- Howard Schreier, Howles Informatics
- http://www2.sas.com/proceedings/sugi31/242-31.pdf

PAPER 242-31

SQL SET OPERATORS: SO HANDY VENN YOU NEED THEM

Howard Schreier, Howles Informatics

ABSTRACT

When it comes to combining data from multiple tables in the SQL Procedure, joins get most of the attention and subqueries are probably second. Often overlooked are the set operators (OUTER UNION, UNION, INTERSECT, and EXCEPT). This tutorial begins by relating OUTER UNION to similar functionality provided by the DATA step's SET statement, then explains and demonstrates the full repertoire of SET operators.

Back to our Evil Online Retailer

(hypothetical of course)

People:

GUID	Name	Sex
99	Paul	M
78	Denise	F

Purch:

GUID	Date	Item	Qty
99	11/02	PhoneX	1
99	11/03	PhCaseX	1

Click:

GUID	Date	Page
99	10/28	Phone8
101	10/29	PhoneX
78	11/01	PhCases

Back to our Evil Online Retailer

(hypothetical of course)

People:

GUID	Name	Sex
99	Pau	
78	Der	

Purch:

GUID	Date		
99	11/02		
99	11/03	PhCaseX	1

age
hone8
honeX
hCases

Who are the People
How do they Click
And when do they Buy

Because Maybe we can find/use their friends to suggest they buy a whole lot more

SQL does SETS
formally!

- Matches columns by Position, not by Name?
  - yikes.
  - OPTIONS DWIM=YES;

```
57              Proc SQL;
58          Select * from people
59          union all
60          Select * from click
61          union all
62          Select * from purch
63          order by date
64          ;
WARNING: A table has been extended with null columns to perform the UNION ALL set operation.
ERROR: Column 2 from the first contributor of UNION ALL is not the same type as its counterpart from
ERROR: Column 2 from the first contributor of UNION ALL is not the same type as its counterpart from
ERROR: Column 3 from the first contributor of UNION ALL is not the same type as its counterpart from
NOTE: PROC SQL set option NOEXEC and will continue to check the syntax of statements.
65
```

Who clicked looking to buy stuff?

```
Proc SQL;
Select *
from people
union all CORRESPONDING
Select * from click
union all CORRESPONDING
Select * from purch;
```

guid
99
78
99
101
78
99
99

Who clicked looking to buy stuff?

```
Proc SQL;
Select *
from people
OUTER union CORRESPONDING
Select * from click
OUTER union CORRESPONDING
Select * from purch;
```

guid	name	sex	date	page	item	qty
99	Paul	M	.			.
78	Denise	F	.			.
99			28OCT17	http://evil.com/store/Phone8.htm		.
101			29OCT17	http://evil.com/store/PhoneX.htm		.
78			04NOV17	http://evil.com/store/PhCases.htm		.
99			02NOV17		PhoneX	1
99			08NOV17		PhCaseX	1

Who clicked looking to buy stuff?

```
Proc SQL;
Select *
from people
OUTER union CORRESPONDING
Select * from click
OUTER union CORRESPONDING
Select * from purch
order by guid, date;
```

guid	name	sex	date	page	item	qty
78	Denise	F	.			.
78			04NOV17	http://evil.com/store/PhCases.htm		.
99	Paul	M	.			.
99			28OCT17	http://evil.com/store/Phone8.htm		.
99			02NOV17		PhoneX	1
99			08NOV17		PhCaseX	1
101			29OCT17	http://evil.com/store/PhoneX.htm		.

Who clicked looking to buy stuff?

```
Proc SQL;
Select COALESCE(P.guid, PC.guid) as guid,
       P.name, P.sex,
       PC.date, PC.what, PC.page, PC.item, PC.qty
  from people P
    FULL JOIN
          (
       Select *, "Click" as what from click
       OUTER union CORRESPONDING
       Select *, "Buy" as what from purch
          ) PC
    on P.guid = PC.guid
 order by date, guid    ;
```

guid	name	sex	date	what	page	item	qty
99	Paul	M	28OCT17	Click	http://evil.com/store/Phone8.htm		.
101			29OCT17	Click	http://evil.com/store/PhoneX.htm		.
99	Paul	M	02NOV17	Buy		PhoneX	1
78	Denise	F	04NOV17	Click	http://evil.com/store/PhCases.htm		.
99	Paul	M	08NOV17	Buy		PhCaseX	1

Cool SQL tricks

- Fuzzy Joins
- Bounding Box Matches
- &&&say&what ?
  - really! SQL is good with together with macros

Fuzzy Join

```
select *
  from file1 a, file2 b
 where a.sex = b.sex
   and ( ( (a.day = b.day) * 2 )
       + ( (a.mon = b.mon) * 2 )
       + ( (a.yr  = b.yr ) * 2 )
       + ( (a.fnm = b.fnm) * 1 )
       + ( (a.lnm = b.lnm) * 1 )
       ) >= 6;
```

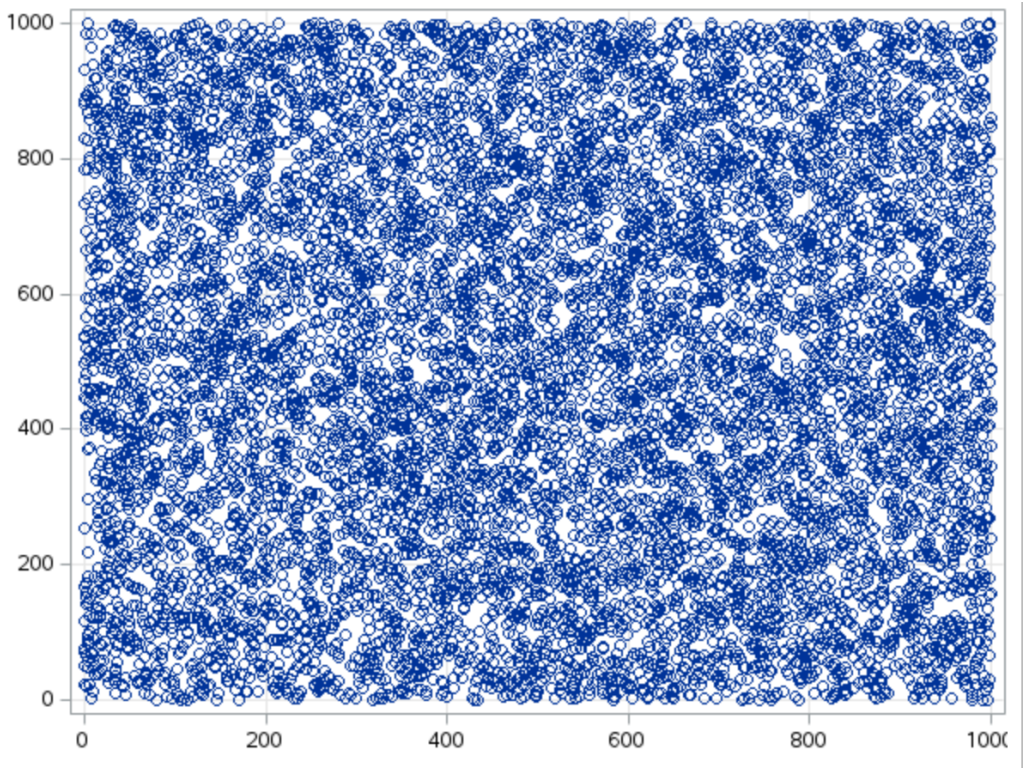
Bounding Box Match

- Paradox
- Doing more work to do less work
- 10000 points where:
  - X ranges from 0 to 1000
  - Y ranges from 0 to 1000
- Find those points “within 1” away from each other

The Points

```
57
58          data points;
59            do id = 1 to 10000;
60              x = uniform(12345) * 1000;
61              y = uniform(56789) * 1000;
62              output;
63              end;
64            run;

NOTE: The data set WORK.POINTS has 10000 observations and 3 variables.
NOTE: DATA statement used (Total process time):
      real time           0.01 seconds
      cpu time            0.01 seconds
```

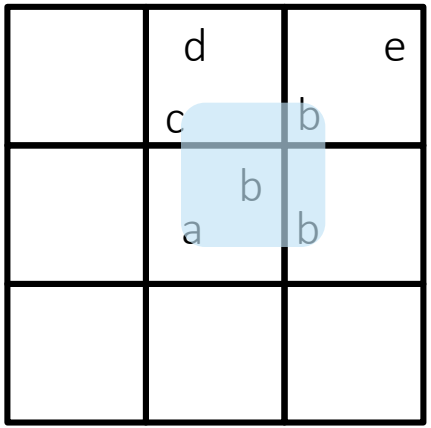
The Points



Points Nearby – Simple

```
65
66          proc sql stimer _method;
NOTE: SQL Statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.00 seconds

67            create table t1 as
68            select a.id as id1, a.x as x1, a.y as y1,
69                   b.id as id2, b.x as x2, b.y as y2
70              from points a, points b
71           where b.x > a.x and b.x - a.x < 1
72             and b.y > a.y and b.y - a.y < 1;
NOTE: The execution of this query involves performing one or more Cartesian product

NOTE: SQL execution methods chosen are:

      sqxcrta
          sqxjsl
              sqxsrc( WORK.POINTS(alias = A) )
              sqxsrc( WORK.POINTS(alias = B) )
NOTE: Table WORK.T1 created, with 99 rows and 6 columns.

NOTE: SQL Statement used (Total process time):
      real time           3.00 seconds
      cpu time            2.99 seconds
```

Think outside the Box



- Compute the bottom left of the grid for each point
- Only 4 of the many grid-boxes will have a point at most 1-unit away

- Query becomes match the box of this point with 4 possible boxes of the partner.
- (some overspray, so deal with that)

Points Nearby – Gridded Prep

```
74          data offset;

75            xoff = 0; yoff = 0; output;
76            xoff = 0; yoff = 1; output;
77            xoff = 1; yoff = 0; output;
78            xoff = 1; yoff = 1; output;
79            run;

NOTE: The data set WORK.OFFSET has 4 observations and 2 variables.
NOTE: DATA statement used (Total process time):
      real time           0.00 seconds
      cpu time            0.01 seconds
```

Points Nearby – Gridded Solution

```
82          create table t2 as
83          select a.id as id1, a.x as x1, a.y as y1,
84                 b.id as id2, b.x as x2, b.y as y2
85            from points a, points b, offset
86         where int(a.x) + xoff = int(b.x)
87           and int(a.y) + yoff = int(b.y)
88           and b.x > a.x and b.x - a.x < 1
89           and b.y > a.y and b.y - a.y < 1
90           ;
NOTE: The execution of this query involves performing one or more Cartes

NOTE: SQL execution methods chosen are:

      sqxcrta
          sqxjhsh
              sqxjsl
                  sqxsrc( WORK.OFFSET )
                  sqxsrc( WORK.POINTS(alias = A) )
              sqxfil
                  sqxsrc( WORK.POINTS(alias = B) )
NOTE: Table WORK.T2 created, with 99 rows and 6 columns.

NOTE: SQL Statement used (Total process time):
      real time           0.02 seconds
      cpu time            0.03 seconds
```

&&&say&what

- Find all the SSN’s
  - in all SAS datasets
  - in all directories
- UNIX “find“ command can find all the datasets
- Now we “just“ need a libname for each directory
- And a test to see if SSN variable exists

find /Users/kent -name "*.sas7bdat"

```
/Users/kent/SyncUX/docker/sas-viya-programming/data/census.sas7bdat
/Users/kent/SyncUX/docker/sas-viya-programming/data/crime.sas7bdat
/Users/kent/SyncUX/public_html/xml/tumor.sas7bdat
/Users/kent/SyncUX/public_html/RDS/sasdata/rdsprojects.sas7bdat
/Users/kent/SyncUX/sgf2014/mm2013/sasuser/out1.sas7bdat
/Users/kent/SyncUX/sgf2014/mm2013/sasuser/out2.sas7bdat
/Users/kent/SyncUX/sgf2014/mm2013/out1.sas7bdat
/Users/kent/SyncUX/sgf2014/mm2013/out2.sas7bdat
/Users/kent/SyncUX/viya/sas-viya-programming/data/census.sas7bdat
/Users/kent/SyncUX/viya/sas-viya-programming/data/crime.sas7bdat
/Users/kent/SyncUX/viya/sas-viya-machine-learning/stacking/data/test_preds.sas7bdat
/Users/kent/SyncUX/viya/sas-viya-machine-learning/stacking/data/adult_train.sas7bdat
```

```
data files;
  infile “find.txt”;
  length filename dirname $100;
  input filename $;
  rev = reverse(filename);
  slash = index(rev, '/’);
  filename = reverse(substr(rev,1,slash-1));
  dirname  = reverse(substr(rev,slash+1));
  *put slash= dirname= filename=;
  run;
```

```
%macro findSSN;
Proc SQL;
  select distinct dirname
    into :dir1 thru :dir9999
    from files;

  %let ndir=&SQLOBS;
  %do i = 1 %to &ndir;

    libname lib "&&&dir&i";
    select "&&&dir&i" as dir, memname from dictionary.tables
       where libname="LIB" and name="SSN";
    libname lib clear;

    %end;
%mend;
%findSSN;
```

SQL can be fun!

paul.kent@sas.com

paulmkent

@hornpolish

sas.com