

Ring Charts

David Corliss, The Polk Company, Southfield, MI

Abstract

Ring Charts are presented as a new, graphical technique for analyzing complex relationships between tables in a relational database. Charts are constructed in the SAS System by creating an Annotate dataset. PROC GSLIDE is then used to produce graphical output. Ring Charts provide the frequency of table joins in a visual format, allowing the system analyst to identify important characteristics. These include hierarchies, inefficient operations, and system constraints.

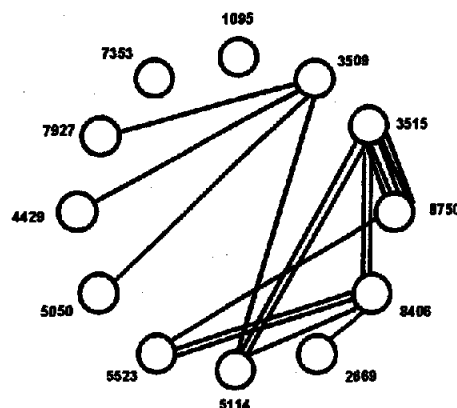
Introduction

Today, more and more information is becoming available at an ever increasing rate. The problem with our data warehouses is no longer how to get the information we need - it has become how to find out what we need to know from a vast supply. Once, we had too little information on how our systems work. Now we have too much, with the useful, interesting, and irrelevant all thrown in together. The ability to measure every conceivable aspect of complex systems has brought a greater need to identify the things that really matter.

Ring Charts are an outgrowth of Chaos Theory. This area of mathematics examines systems that are so complex that they seem completely random, seeking to identify the underlying patterns, causes and explanations. Ring Charts are useful because they allow a vast amount of simple information to be presented in a graphical format. This facilitates the identification of important characteristics, including hierarchies, inefficient operations, and system constraints. One of the first successes of the method was in using telephone records to find hierarchies in crime organizations, which are not known for publishing their organizational charts on the internet.

Two Examples

Figure 1

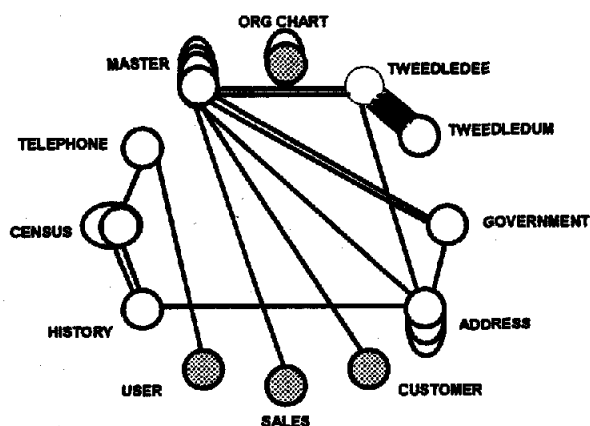


In Figure 1, each small circle represents a person suspected of being a member of an illegal operation. Each person is identified by his or her telephone number. Whenever

telephone records show a call from one person to another, a line is drawn connecting the circles. Over time, as many lines are drawn, some readily identified patterns can emerge. Two persons working together in a close partnership will show many lines connecting them. An example of this can be seen on the right side of the chart. Calls unrelated to organization business will spread out to persons with one or two contacts. These can appear on the chart like the whiskers of a cat (left side). A nexus, where many lines come together from different places, generally indicates higher standing in the organization.

A Ring Chart shows the relationship between a collection of objects related by a shared action. In the organized crime scenario, people were connected by telephone calls. In a relational database (Figure 2), tables are connected by table joins. While persons cannot call themselves on a telephone, it is possible to query data from a single table with no table joins. These actions, called Identity Operations, are represented by curved lines going out from a table and back into itself.

Figure 2



In this chart, as before, nexus tables are at the top of a hierarchy (Master Table). "Whisker" tables at the end of very few queries (gray tables) are under utilized and possibly a waste of time and space. They may contain only one or two essential fields that that could be placed on another table, allowing the query to run with fewer joins. A table may be present whose content is not employed in combination with others in the database and could be stored independently (e.g. Org Chart). Many lines drawn between two tables (Tweedledee and Tweedledum) reflect a partnership that is bad for business: constantly using queries to join the same tables wastes every system resource, especially time. In this example, Tweedledum is never queried without first being joined to Tweedledee. Their fields need to be combined into a single table. Because the intensity of the lines is heaviest between Tweedledee and Tweedledum, this is the *constraint*: the place in the system where usage is maximized and performance is minimized. Constraints are bottlenecks. Allocating more resources or simplifying the system at this point makes the greatest improvement for the least expenditure. However, making additional resources available in other areas will do little to improve operations unless this is resolved first.

Constructing a Ring Chart

The creation of a Ring Chart begins with assembling the data into a Query Dataset. Dynamic data collection will greatly facilitate this process. In a query program, follow the code that runs the query with a data step that appends the name of each table join to a SAS dataset. There will be one line appended to the Query Dataset for each table join in each query. For example:

```
LIBNAME ANDVARI 'C:\SAS\ANDVARI' ;
RUN;

/* Select the desired fields from the database */
/* This example selects fields from Table 04 and Table 07 using a match code */

PROC SQL;
  CREATE TABLE WORK.QUERY
  AS
  SELECT T04VAR_1 T04VAR_2 T04VAR_3 A.MATCH T07VAR_1 T07VAR_2 T7VAR_3;
  FROM ANDVARI.T1 A, ANDVARI.T2 B;
  WHERE A.MATCH = B.MATCH;
QUIT;

RUN;

/* Once the query is completed, write the name of the table joins to a file */

DATA ANDVARI.QUERY;
TABLES='T0407'; OUTPUT;

RUN;
```

Add the code in **bold** for every query run on the database. When enough records have been collected for analysis (at least 30), a Ring Chart can be produced. If the workflow in the system is highly cyclical, with much the same work being repeated for some time period, use only whole time periods in the dataset. If the work cycles each week, four weeks may be used but a month should not.

Once the Query Dataset has been prepared, the following program will produce the Ring Chart. A PROC SUMMARY calculates the number of times each table join appears in the Query Dataset. An Annotate Dataset is created containing one record for every item to be found in the graphical output. In this program, only a single line is drawn for each combination of tables found in the Query Dataset. The width of this line is given by the total from the PROC SUMMARY. Identity operations are displayed as short lines going away from the table queried. A PROC GSLIDE reads the Annotate Dataset and draws the Ring Chart.

```
/******
*
*   Program:  CHART.SAS
*   Platform: SAS v6.12
*   Author:   David Corliss
*
*   Purpose:  This program creates a ring chart for a relational
*             database. The chart has one point on the ring for each
*             table in the database. Data is read from a dataset
*             containing one record for each table join used in
*             queries over some time period. Totals are made for each
*             table join and plotted on the ring chart.
*
*   Program Version and Date: v1.0  January 28, 1999
*   Validated:                v1.0  February 4, 1999
*   Change Control:
*
******/
```

```

*
*****/

/* Assign data library */
LIBNAME ANDVARI 'C:\SAS\ANDVARI';
RUN;

/* Set up the graphics environment */
GOPTIONS RESET=GLOBAL GUNIT=PCT BORDER
          FTEXT=SWISSB HTITLE=6 HTEXT=3 LFACTOR=5;

%ANNOMAC;

/* Calculate the total for each table join */
PROC SORT DATA=ANDVARI.WEEKLY;
BY TABLES;
RUN;

DATA WORK.TEMP;
SET ANDVARI.WEEKLY;
BY TABLES;

TOTAL = 1;
RUN;

PROC SUMMARY DATA=WORK.TEMP;
VAR TOTAL;
OUTPUT OUT=ANDVARI.TOTALS (DROP=_FREQ_ _TYPE_) SUM=;
BY TABLES;
RUN;

/* Create Annotate dataset */
DATA ANDVARI.ANNOTATE;

LENGTH FUNCTION STYLE COLOR $ 8 TEXT $ 25;
RETAIN HSYS XSYS YSYS '3';

/* Draw a circle for each table in the database */
%CIRCLE(60, 80, 3, BLACK);
%CIRCLE(80, 70, 3, BLACK);
%CIRCLE(90, 57, 3, BLACK);
%CIRCLE(90, 42, 3, BLACK);
%CIRCLE(80, 30, 3, BLACK);
%CIRCLE(60, 20, 3, BLACK);
%CIRCLE(40, 20, 3, BLACK);
%CIRCLE(20, 30, 3, BLACK);
%CIRCLE(10, 42, 3, BLACK);
%CIRCLE(10, 57, 3, BLACK);
%CIRCLE(20, 70, 3, BLACK);
%CIRCLE(40, 80, 3, BLACK);

/* Title */
FUNCTION='LABEL'; X=40; Y=95; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=6;
TEXT='RING CHART'; OUTPUT;

/* Label the circle for each table */
FUNCTION='LABEL'; X=62; Y=82; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 1'; OUTPUT;

FUNCTION='LABEL'; X=82; Y=71; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 2'; OUTPUT;

FUNCTION='LABEL'; X=92; Y=58; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 3'; OUTPUT;

FUNCTION='LABEL'; X=92; Y=43; POSITION='6';

```

```

STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 4'; OUTPUT;

FUNCTION='LABEL'; X=82; Y=30; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 5'; OUTPUT;

FUNCTION='LABEL'; X=62; Y=19; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 6'; OUTPUT;

FUNCTION='LABEL'; X=31; Y=19; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 7'; OUTPUT;

FUNCTION='LABEL'; X=11; Y=30; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 8'; OUTPUT;

FUNCTION='LABEL'; X=1; Y=43; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 9'; OUTPUT;

FUNCTION='LABEL'; X=1; Y=58; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 10'; OUTPUT;

FUNCTION='LABEL'; X=11; Y=71; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 11'; OUTPUT;

FUNCTION='LABEL'; X=31; Y=82; POSITION='6';
STYLE='SWISSB'; COLOR='BLACK'; SIZE=3;
TEXT='TABLE 12'; OUTPUT;

RUN;

/* Write a record to the Annotate dataset for each line to be drawn */
DATA WORK.LINES;
SET ANDVARI.TOTALS;

RETAIN HSYS XSYS YSYS '3';

IF TABLES = 'T0101' THEN
DO;
FUNCTION='MOVE'; POSITION='6'; X=60; Y=80; LINE = 1; STYLE='SWISSB';
COLOR='BLACK'; SIZE= TOTAL; TEXT='T0101'; OUTPUT;

FUNCTION='DRAW'; POSITION='6'; X=60; Y=85; LINE = 1; STYLE='SWISSB';
COLOR='BLACK'; SIZE= TOTAL; TEXT='T0101'; OUTPUT;
END;

IF TABLES = 'T0102' THEN
DO;
FUNCTION='MOVE'; POSITION='6'; X=60; Y=80; LINE = 1; STYLE='SWISSB';
COLOR='BLACK'; SIZE= TOTAL; TEXT='T0102'; OUTPUT;

FUNCTION='DRAW'; POSITION='6'; X=80; Y=70; LINE = 1; STYLE='SWISSB';
COLOR='BLACK'; SIZE= TOTAL; TEXT='T0102'; OUTPUT;
END;

ELSE IF TABLES = 'T0103' THEN
DO;
FUNCTION='MOVE'; POSITION='6'; X=60; Y=80; LINE = 1; STYLE='SWISSB';
COLOR='BLACK'; SIZE= TOTAL; TEXT='T0103'; OUTPUT;

FUNCTION='DRAW'; POSITION='6'; X=90; Y=57; LINE = 1; STYLE='SWISSB';
COLOR='BLACK'; SIZE= TOTAL; TEXT='T0103'; OUTPUT;
END;

/*
-
-
-
-

```

Include each possible table join allowed by the database

```

-
-
-

```

```

*/
ELSE IF TABLES = 'T1212' THEN
DO;
  FUNCTION='MOVE'; POSITION='6'; X=40; Y=80; LINE = 1; STYLE='SWISSB';
  COLOR='BLACK'; SIZE= TOTAL; TEXT='T1010'; OUTPUT;

  FUNCTION='DRAW'; POSITION='6'; X=40; Y=85; LINE = 1; STYLE='SWISSB';
  COLOR='BLACK'; SIZE= TOTAL; TEXT='T1010'; OUTPUT;
END;

DROP TABLES TOTAL;

RUN;

DATA ANDVARI.ANNOTATE;
SET ANDVARI.ANNOTATE WORK.LINES;

RUN;

/* Display Ring Chart */
FOOTNOTE J=CENTER 'David Corliss - Midwest SAS Users Group - September 11, 2000 ';
PROC GSLIDE ANNOTATE=ANDVARI.ANNOTATE;

RUN;
QUIT;

/* Re-initialize the dataset for recording table joins */
/*
DATA ANDVARI.ARCHIVE;
SET ANDVARI.WEEKLY;

RUN;

DATA ANDVARI.WEEKLY;
SET ANDVARI.WEEKLY;
IF TABLES NE '' THEN DELETE;

RUN;
*/

```

Scaling

When a data warehouse has a very high number of queries, there can be too many lines on a chart to plot every one. If a single period contains hundred of queries instead of a few dozen, modifications must be made to allow clear presentation of the data. This process, called Scaling, involves dividing the output of the PROC SUMMARY by a scale factor. If a scale factor of 10 is used, ten joins between the same two tables will produce a line only one unit in width.

Conclusion

Ring Charts provide the frequency of table joins in a relational database in a graphical format, allowing the system analyst to identify important characteristics. These include hierarchies, inefficient operations, and system constraints. Tables at the nexus of many queries are at the top of system hierarchy. "Whisker" tables are at the end of very few queries, identifying them as the target of consolidation efforts. The point at which the lines are heaviest marks the constraint of the system. Combining these tables into one may produce a considerable improvement in performance.

A Ring Chart may show that a system uses all of its tables well without carrying any dead weight. The system may be free of any severe constraint and yet performance is still unsatisfactory. Ring Charts show the efficiency of a system's architecture. When a

system is running as well as can be designed but is still too slow, the only way to improve performance is to increase hardware resources. In this event, a Ring Chart can provide the scientific analysis required by company management to justify such an outlay.

References

SAS Institute Inc. (1999), *SAS Language Reference: Concepts, Version 8*, Cary, NC: SAS Institute Inc.

SAS is a registered trademark or trademark of SAS Institute Inc. in the USA and other countries. ® indicates USA registration.

Other brand and product names are registered trademarks or trademarks of their respective companies.

Contact Information

The author greatly values your thoughts and input.

David Corliss

The Polk Company

26955 Northwestern Highway

Southfield, MI 48034

Phone: (248) 728-7863

Fax: (248) 728-7460

Email: David_Corliss@polk.com