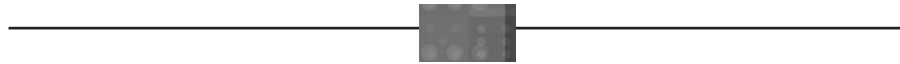


JOE CELKO'S ANALYTICS & OLAP IN SQL

J O E C E L K O ' S

Analytics and OLAP in SQL



The Morgan Kaufmann Series in Data Management Systems

Series Editor: Jim Gray, Microsoft Research

Joe Celko's Analytics and OLAP in SQL

Joe Celko

Data Preparation for Data Mining Using SAS

Mamdouh Refaat

Querying XML: XQuery, XPath, and SQL/XML in Context

Jim Melton and Stephen Buxton

Data Mining: Concepts and Techniques, Second Edition

Jiawei Han and Micheline Kamber

Database Modeling and Design: Logical Design, Fourth Edition

Toby J. Teorey, Sam S. Lightstone and Thomas P. Nadeau

Foundations of Multidimensional and Metric Data Structures

Hanan Samet

Joe Celko's SQL for Smarties: Advanced SQL Programming, Third Edition

Joe Celko

Moving Objects Databases

Ralf Hartmut Güting and Markus Schneider

Joe Celko's SQL Programming Style

Joe Celko

Data Mining, Second Edition: Concepts and Techniques

Ian Witten and Eibe Frank

Fuzzy Modeling and Genetic Algorithms for Data Mining and Exploration

Earl Cox

Data Modeling Essentials, Third Edition

Graeme C. Simsion and Graham C. Witt

Location-Based Services

Jochen Schiller and Agnès Voisard

Database Modeling with Microsoft Visio for Enterprise Architects

Terry Halpin, Ken Evans, Patrick Hallock, Bill Maclean

Designing Data-Intensive Web Applications

Stephano Ceri, Piero Fraternali, Aldo Bongio, Marco Brambilla, Sara Comai, and Maristella Matera

Mining the Web: Discovering Knowledge from Hypertext Data

Soumen Chakrabarti

Advanced SQL: 1999—Understanding Object-Relational and Other Advanced Features
Jim Melton

Database Tuning: Principles, Experiments, and Troubleshooting Techniques
Dennis Shasha and Philippe Bonnet

SQL:1999—Understanding Relational Language Components
Jim Melton and Alan R. Simon

Information Visualization in Data Mining and Knowledge Discovery
Edited by Usama Fayyad, Georges G. Grinstein, and Andreas Wierse

Transactional Information Systems: Theory, Algorithms, and Practice of Concurrency Control and Recovery
Gerhard Weikum and Gottfried Vossen

Spatial Databases: With Application to GIS
Philippe Rigaux, Michel Scholl, and Agnes Voisard

Information Modeling and Relational Databases: From Conceptual Analysis to Logical Design
Terry Halpin

Component Database Systems
Edited by Klaus R. Dittrich and Andreas Geppert

Managing Reference Data in Enterprise Databases: Binding Corporate Data to the Wider World
Malcolm Chisholm

Understanding SQL and Java Together: A Guide to SQLJ, JDBC, and Related Technologies
Jim Melton and Andrew Eisenberg

Database: Principles, Programming, and Performance, Second Edition
Patrick and Elizabeth O'Neil

The Object Data Standard: ODMG 3.0
Edited by R. G. G. Cattell and Douglas K. Barry

Data on the Web: From Relations to Semistructured Data and XML
Serge Abiteboul, Peter Buneman, and Dan Suciu

Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations
Ian Witten and Eibe Frank

Joe Celko's SQL for Smarties: Advanced SQL Programming, Second Edition
Joe Celko

Joe Celko's Data and Databases: Concepts in Practice
Joe Celko

Developing Time-Oriented Database Applications in SQL
Richard T. Snodgrass

Web Farming for the Data Warehouse
Richard D. Hackathorn

Management of Heterogeneous and Autonomous Database Systems
Edited by Ahmed Elmagarmid, Marek Rusinkiewicz, and Amit Sheth

Object-Relational DBMSs: Tracking the Next Great Wave, Second Edition
Michael Stonebraker and Paul Brown, with Dorothy Moore

A Complete Guide to DB2 Universal Database
Don Chamberlin

Universal Database Management: A Guide to Object/Relational Technology
Cynthia Maro Saracco

Readings in Database Systems, Third Edition
Edited by Michael Stonebraker and Joseph M. Hellerstein

Understanding SQL's Stored Procedures: A Complete Guide to SQL/PSM
Jim Melton

Principles of Multimedia Database Systems
V. S. Subrahmanian

Principles of Database Query Processing for Advanced Applications
Clement T. Yu and Weiyi Meng

Advanced Database Systems
Carlo Zaniolo, Stefano Ceri, Christos Faloutsos, Richard T. Snodgrass, V. S. Subrahmanian, and Roberto Zicari

Principles of Transaction Processing
Philip A. Bernstein and Eric Newcomer

Using the New DB2: IBMs Object-Relational Database System
Don Chamberlin

Distributed Algorithms
Nancy A. Lynch

Active Database Systems: Triggers and Rules For Advanced Database Processing
Edited by Jennifer Widom and Stefano Ceri

Migrating Legacy Systems: Gateways, Interfaces, & the Incremental Approach
Michael L. Brodie and Michael Stonebraker

Atomic Transactions

Nancy Lynch, Michael Merritt, William Weihl, and Alan Fekete

Query Processing for Advanced Database Systems

Edited by Johann Christoph Freytag, David Maier, and Gottfried Vossen

Transaction Processing: Concepts and Techniques

Jim Gray and Andreas Reuter

Building an Object-Oriented Database System: The Story of O2

Edited by François Bancilhon, Claude Delobel, and Paris Kanellakis

Database Transaction Models for Advanced Applications

Edited by Ahmed K. Elmagarmid

A Guide to Developing Client/Server SQL Applications

Setrag Khoshafian, Arvola Chan, Anna Wong, and Harry K. T. Wong

The Benchmark Handbook for Database and Transaction Processing Systems, Second Edition

Edited by Jim Gray

Camelot and Avalon: A Distributed Transaction Facility

Edited by Jeffrey L. Eppinger, Lily B. Mummert, and Alfred Z. Spector

Readings in Object-Oriented Database Systems

Edited by Stanley B. Zdonik and David Maier

This Page Intentionally Left Blank

JOE CELKO'S

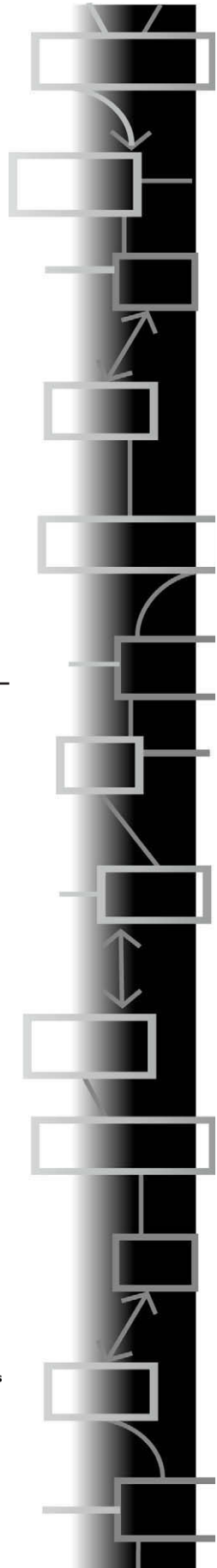
Analytics and OLAP in SQL

Joe Celko



AMSTERDAM • BOSTON • HEIDELBERG • LONDON
NEW YORK • OXFORD • PARIS • SAN DIEGO
SAN FRANCISCO • SINGAPORE • SYDNEY • TOKYO
MORGAN KAUFMANN PUBLISHERS IS AN IMPRINT OF ELSEVIER

MK[®]
MORGAN KAUFMANN PUBLISHERS



Publisher	Diane Cerra
Publishing Services Manager	George Morrison
Editorial Assistant	Asma Palmeiro
Cover Design	Side by Side Studios
Cover Image	Side by Side Studios
Cover Designer	Eric DeCicco
Composition	Multiscience Press, Inc.
Copyeditor	Multiscience Press, Inc.
Proofreader	Multiscience Press, Inc.
Indexer	Multiscience Press, Inc.
Interior printer	The Maple-Vail Book Manufacturing Group
Cover printer	Phoenix Color Corp.

Morgan Kaufmann Publishers is an imprint of Elsevier.
500 Sansome Street, Suite 400, San Francisco, CA 94111

This book is printed on acid-free paper.

© 2006 by Elsevier Inc. All rights reserved.

Designations used by companies to distinguish their products are often claimed as trademarks or registered trademarks. In all instances in which Morgan Kaufmann Publishers is aware of a claim, the product names appear in initial capital or all capital letters. Readers, however, should contact the appropriate companies for more complete information regarding trademarks and registration.

No part of this publication may be reproduced, stored in a retrieval system, or transmitted in any form or by any means-electronic, mechanical, photocopying, scanning, or otherwise-without prior written permission of the publisher.

Permissions may be sought directly from Elsevier's Science & Technology Rights Department in Oxford, UK: phone: (+44) 1865 843830, fax: (+44) 1865 853333, e-mail: permissions@elsevier.com.uk. You may also complete your request on-line via the Elsevier homepage (<http://elsevier.com>) by selecting "Customer Support" and then "Obtaining Permissions."

Library of Congress Cataloging-in-Publication Data

Application submitted.

ISBN-10 : 0123695120
ISBN-13: 978-0-12-369512-3

For information on all Morgan Kaufmann publications,
visit our Web site at www.mkp.com or www.books.elsevier.com

Printed in the United States of America
06 07 08 09 5 4 3 2 1



For Charlotte and Tess, who should have gotten a dedication at least a decade ago for all the work they put into school.

This Page Intentionally Left Blank

C O N T E N T S

Introduction	xvii
Beyond Queries	xvii
Some of the Differences between OLAP and OLTP	xviii
Corrections and Additions	xx
 I Basic Reports and History	 1
1.1 Cases	2
1.2 Control-Break Reports	3
1.3 Cross-Tabulation Reports	3
1.4 Presentation Graphics	4
1.5 Local Databases	4
 2 Cross-Tabulations	 7
2.1 Crosstabs by Cross-Join	11
2.2 Crosstabs by Outer Joins	12
2.3 Crosstabs by Subquery	13
2.4 Crosstabs by CASE Expression	14
2.5 Crosstabs with Row and Column Summaries	14
 3 Dimension Tables	 17
3.1 Star and Snowflake Schemas	17
3.2 Kinds of Dimensions	22
3.2.1 Slowly Changing Dimensions	23
3.2.2 Hierarchical Dimensions	23
3.2.3 Categorical Dimensions	26
3.3 Calendars and Temporal Data	26
3.3.1 Report Range Tables	28
3.4 Helper Tables	29
3.5 Surrogate Keys	30
3.6 Degenerate Dimensions	32

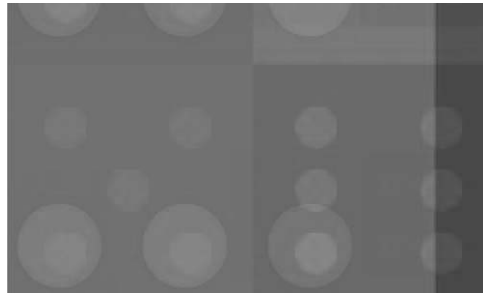
4	Data Migration and Scrubbing	33
4.1	Pumping Data	34
4.2	Verification and Validation	35
4.3	Extract, Transform, and Load (ETL)	36
4.4	Databases Also Evolved	36
4.5	Data Warehouses	37
4.6	Extract, Load, and then Transform (E-L-T)	38
4.6.1	Does It Work?	39
4.7	Scrubbing Data with Non-First-Normal-Form (1NF) Tables	40
4.7.1	Setting up a Scrubbing Table	40
4.7.2	Designing a Target Table	46
4.7.3	Adding Constraints and Validations	48
5	MERGE Statement	51
5.1	Simple MERGE Statement	52
5.2	Merging without the MERGE Statement	53
5.3	TRIGGERS and MERGE	54
5.4	Self-Referencing MERGE	55
6	OLAP Basics	57
6.1	Cubes	57
6.2	Dr. Codd's OLAP Rules	58
6.2.1	Basic Features	59
6.2.2	Special Features	61
6.2.3	Reporting Features	61
6.2.4	Dimension Control	62
6.3	MOLAP	62
6.4	ROLAP	63
6.5	HOLAP	63
6.6	OLAP Query Languages	63
7	GROUPING Operators	65
7.1	GROUP BY GROUPING SET	65
7.2	ROLLUP	66

7.3	CUBES	68
7.4	Notes about Usage	69
8	OLAP Operators in SQL	71
8.1	OLAP Functionality	72
8.1.1	Row Numbering	72
8.1.2	RANK and DENSE_RANK	74
8.1.3	The Window Clause	75
8.2	NTILE()	78
8.3	Nesting OLAP functions	80
8.4	Sample Queries	80
9	Sparseness in Cubes	83
9.1	Hypercube	84
9.2	Dimensional Hierarchies	84
9.3	Drilling and Slicing	86
10	Data Quality	89
10.1	Checking Columns for Value Counts	90
10.2	Finding Rules in a Schema	91
10.2.1	Mathematical Rules	93
10.2.2	Logical Rules	94
10.2.3	Pattern Rules	96
10.3	Feedback for Data Quality	97
10.4	Further Reading	97
11	Correlation	99
11.1	Causes and Correlation	101
11.2	Linear Correlation	102
11.2.1	Pearson's r	102
11.2.2	The Rashomon Effect	105
11.3	Nesting Functions	105
11.4	Further Reading	106

12	Data Distributions	107
12.1	Flat Distribution	107
12.2	Zipfian Distribution	108
12.3	Gaussian, Normal, or Bell Curve	109
12.4	Poisson Distribution	111
12.5	Logistic or “S” Distribution	112
12.6	Pareto Distribution	112
12.7	Distribution Discovery	113
12.7.1	Missing Data Discovery	114
12.8	References	115
13	Market-Basket Analysis	117
13.1	Simple Example of a Market Basket	117
13.2	Relational Division	119
13.2.1	Division with a Remainder	120
13.2.2	Exact Division	121
13.2.3	Todd’s Division	122
13.2.4	Division with Set Operators	124
13.3	Romney’s Division	124
13.4	How to Use Relational Divisions	127
14	Decision, Classification, and Regression Trees	129
14.1	Casual Caldistics	130
14.2	Decision and Correlation Trees	132
14.3	Entropy	134
14.4	Other Algorithms and Software	135
15	Computer-Intensive Analysis	139
15.1	Bootstraps	140
15.2	Subgroups	140
15.3	Bayesian Analysis	141
15.4	Clustering	143
15.4.1	Uses for Clustering	144
15.4.2	Problems with Clustering	144

16	Relationship Analytics	147
16.1	Adjacency List Model for General Graphs	148
16.2	Covering Paths Model for General Graphs	152
16.3	Conclusion and Solution	155
16.4	Further Reading	158
17	Database Architectures	159
17.1	Parallelism	160
17.1.1	Parallelism Failures	160
17.2	Hashing	162
17.3	Bit Vector Indexes	163
17.4	Streaming Databases	163
17.5	Further Reading	163
18	MDX from a SQL Viewpoint	165
18.1	MDX SELECT Statement	166
18.2	Hierarchical Navigation	168
18.3	Set Operations	169
18.4	GENERATE Function	169
18.5	Time Series Functions	169
18.6	Filtering	170
18.7	ORDER ()	171
18.8	TOP Functions	171
18.9	Numeric Functions	171
18.10	Conclusions	172
	Index	173

This Page Intentionally Left Blank



Introduction

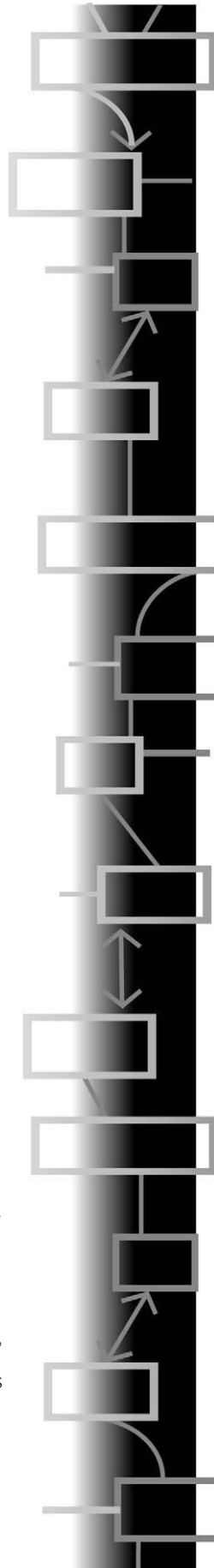
Where is the wisdom?
Lost in the knowledge.
Where is the knowledge?
Lost is the information.
—*T. S. Elliot (1888–1965)*

Where is the information?
Lost in the data.
Where is the data?
Lost is the @#\$\$"& database!!
—*Joe Celko (1947–)*

Many years ago, I did a blank verse follow-up to a T. S. Eliot poem, which you see at the start of this introduction. What I want to do in this book is to move from data to information via analytics of various kinds. I am not so arrogant to assume that I can go from information to wisdom for the book after this—I leave wisdom and enlightenment to my wife, the Soto Zen monk.

Beyond Queries

The goal of this book is to go beyond my usual “How to write a query” book and get into online analytical processing (OLAP), or analytics, as





well as data quality, data transformations, and the other tools outside the usual scope of a working programmer—I'd like to look at a bigger picture. I will not cover the topics in detail, but instead try to give you an overview of these techniques.

OLAP and data warehousing in SQL are a part of this, but they are not the only tools. I will try to stick to standard SQL and SQL/PSM in any code. However, much of this kind of work has to be done with third-party tools, such as report generators, statistical packages, data transformation tools, and so forth. I will break my usual pattern of using purely standard SQL and give overviews of commercial tools, but not in great detail. Any description of a particular release of a particular product would be dated in a few years at most, anyway.

Some of the Differences between OLAP and OLTP

Size

The first obvious difference between analytics and online transaction processing (OLTP) databases is the size of the data involved. Analytics wants to have as much data as possible from which to draw information. OLTP wants to be fast, which means running lean.

My first column in the information technology (IT) trade press was entitled “Program Tests Basic ‘Rhino Algorithm’” (*Information Systems News*, September 8, 1980). The column talked about how Tarzan avoided being run over by rhinoceroses in the old movie. His algorithm was to jump to one side at the last minute so as to avoid becoming jungle road kill. It worked beautifully in every black and white Tarzan movie I ever saw as a kid. The problem with his algorithm was that it did not scale well to an entire herd of stampeding rhinoceroses.

Data Sources

The next difference is that analytics have multiple data sources. Data is spread out in many places within an enterprise, and these “local islands of data” are not going to be on the same platforms. It would not make sense to put a mainframe in every department or on every desktop; nor would it make sense to put a major corporate accounting system on a PC.

But you also want to use external data as well. For example, census data lets you plan inventory choices to match the people who will be shopping at each store branch.

Transactions versus Computations

The third major difference is that analytics do more computations than OLTP. Most of these computations are in the aggregate, rather than on single entities. You do not care that John Smith bought a candy bar from your store on Thursday. Your cash register system will capture that data and print out a receipt for Mr. Smith's transaction. In analytics, we want to find the purchase patterns for all candy bars in all our stores. The information we want from analytics often has a strong temporal component, rather than being a single event. We try to predict the future by using what we have learned about the past behavior of our system.

Volume and intended use make a big difference in the representation of data, the type of queries that are useful, and even how the information is physically stored.

Data Users

The OLTP system should be available to lots of users, even on the Internet. These users are customers, suppliers, and employees. They can be skilled workers, but they can also be an unspecified population. The interface is very important to them.

The OLAP system works with a small set of skilled users. They are more likely to be statisticians who know which algorithms to use. While nice displays are always appreciated, the OLAP user can handle specialized, often complicated, interfaces.

Queries

The OLTP system works with fast responses to relatively simple queries on particular entities. A typical query might be looking up your bank account balance and recent transactions.

The OLAP system can wait to receive detailed answers to complex queries. Speed is nice, but it is not as important as the depth of precision of the results. These queries are for groupings and categories of entities.

An online user submitting an order will not wait 30 seconds to see if the item is in stock, while an analyst will gladly wait 30 minutes to get a complete report of sales trends for all items for the last year.

Normalization

The OLTP system must be normalized so that as it changes, it preserves data integrity. Denormalizing for speed seldom is worth the loss in data integrity. You need declarative referential integrity (DRI) actions, triggers, assertions, and so forth to be sure that your data is valid. The



schema should be at least third normal form (3NF), though we'd really like fifth normal form (5NF) and domain-key normal forms.

The OLAP system is static, so its data must have integrity before it goes into the schema. It is not going to change while it is in use. The OLAP system works with a star schema or snowflake schema, which are highly denormalized but have data integrity because they are static.

Also, normalization prevents redundancy, and redundancy destroys data integrity in OLTP. Redundancy can speed up queries in OLAP.

Query Tools and Computers

The OLTP system favors a highly portable SQL. It needs to be deployed quickly and available to a wide range of users if it is successful.

The OLAP system favors SQL, but it needs statistical extensions and will also use statistical packages (SAS, SPSS, and specialized products). Once a data warehouse is in place, it does not change.

While the SQL-99 standard OLAP features are being supported in DB2, Oracle, Mimer, and other SQL products, there comes a point where you need other tools.

SQL can run on a general computer, but you will find the OLAP system does much better with a very large database (VLDB) product whose architecture is built to handle massive amounts of data with totally different access methods.

The poor SQL programmer who has grown up in the OLTP world is suddenly as helpless in the OLAP world as the procedural programmer who was dropped into a nonprocedural SQL world. It does not feel good to begin the work day with “Everything I know is wrong” written on your t-shirt.

Corrections and Additions

Please send any corrections, additions, suggestions, improvements, or alternative solutions to me or to the publisher.

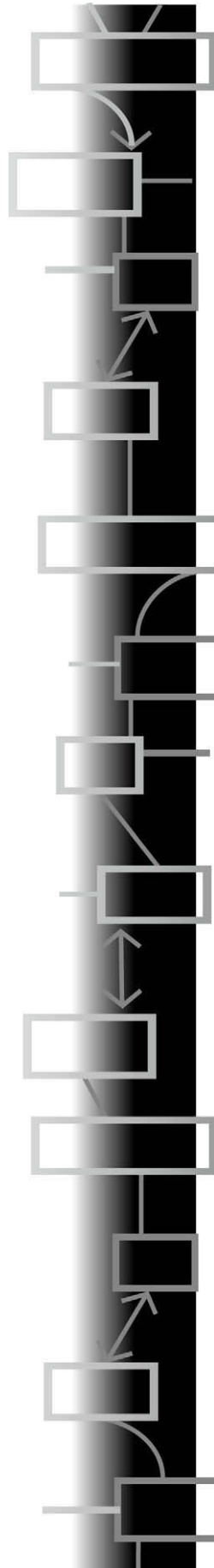
Morgan-Kaufmann Publishers
500 Sansome Street #400
San Francisco, CA 94111-3211

CHAPTER 1

Basic Reports and History

Prisoner: Where am I?
Number Two: In the Village.
Prisoner: What do you want?
Number Two: Information.
Prisoner: Which side are you on?
Number Two: That would be telling. We want information,
information, information . . .
Prisoner: You won't get it.
Number Two: By hook or by crook we will.
Prisoner: Who are you?
Number Two: The new Number Two.
Prisoner: Who is Number One?
Number Two: You are Number Six.
Prisoner: I am not a number. I am a free man.
Number Two: Ha, ha, ha, ha . . .
—*The Prisoner* (1967)

ONCE UPON A time, long, long ago, we had punch cards and magnetic tapes for data processing. We printed reports onto paper. There were no databases or cheap video terminals. Instead we produced tons of continuous-form paper outputs, and I do mean tons.





The problem was that nobody read these reports. In fact, there was a standard trick in software engineering consulting in the 1980s to find out about the usage. You took one of these huge stacks of paper and inserted an envelope with a \$10 bill in the middle of the report. The note told the person who opened the envelope to keep the money and return the note, which had the name of the report on it, to the consultant. Only about one out of 10 or 12 envelopes were discovered.

Data was failing to be turned into information. People simply got a sense of security and importance from looking at a large stack of paper. When hardcopy reports were used, it was usually just the summary-level data. The whole report would sit around in a binder until the next time it was run. The new report would replace the old in the binder, and a few summary pages would be put into a second binder.

If the report did not give you the data in a format you could use, you had to reformat it by hand with a calculator. Later on, you would use a spreadsheet, but the idea was the same.

The reason for all of this was that computers and computer time were very costly, and display terminals were years away. Running a paper report once and letting people share it was the economical sound decision.

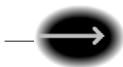
1.1 Cases

The simplest possible report is a printout of each record on a separate page or in a few lines on a case-by-case basis. In the old days, however, you could not afford to run a program to print a few pages on demand. We would print out a set of these cases with a date on them, do updates manually, and then key in the changes in batches. The real “database” was a vertical filing cabinet.

This meant that you had to have a sorting key that defined the order of the output. This usually meant alphabetical or numerical order on one or more fields. The classic example is a printed telephone book. If you know a name and alphabetical order, it is quite easy. However, if you wanted to find all the people who live on Elm Street, you would want to get the same data in a very different order.

People do not appreciate the power of alphabetical ordering for data. Hong Kong telephone operators commit thousands of names and numbers to memory because Chinese does not have a way to map sounds to characters.

Much of the computer time in the early days of data processing was spent sorting data to get reports in the desired order. Since magnetic tapes are sequential by their physical nature, we got very good at



merging and sorting algorithms. This also meant that we assumed that all data had to be in sequential order and that we needed to have a physical sort key.

This assumption became the *primary key* in SQL. By the time we realized that a relational key has nothing to do with the physical ordering of the data's storage, and that all keys are equally keys, the language was defined.

Today, cases can be called up on a screen and updated instantly. But, even more significantly, you do not have to use one sort key to get to the data.

1.2 Control-Break Reports

The next type of report goes by several names: control-break or banded reports are the most common. They consist of lines of detailed information, followed by summary information at various levels of aggregation. The assumption is that your data falls into a hierarchy. Today, a dimension hierarchy is just another type of control break. Like the alphabet, hierarchies are a useful way to organize data. Some things don't change, they simply get mechanized.

Its layout was determined by the hardware that we originally used to produce these reports. The electronic accounting machines had a limited number of accumulators and almost no internal data storage. A record was read from a deck of cards (later from a magnetic tape) and printed on the output paper, and the relevant fields were added to the accumulators as the data passed through the hardware. When the record that started the next group was read, the accumulator contents was output and reset. The number of levels of aggregation was determined by the hardware.

The way that people actually used these reports was to ignore the details and flip the stack of paper to the final summary pages. The grand totals were on the last pages, so people would put them into a binder and throw everything else out.

The obvious way to design such a report would have been to start with the grand totals, then move to subtotals, then sub-subtotals, and finally to the details. Today, this is what is done with terminals and drill-down displays.

1.3 Cross-Tabulation Reports

The third most common report is a cross-tabulation report, also known as "crosstabs" in statistics. This has become more important, because it is



the basis for cubes and other OLAP storage methods, which we will discuss later.

A simple crosstabs is an array of cells on at least two attributes. The values of one attribute go on the horizontal axis, and the values of the other attribute go on the vertical axis. Each cell has a statistic about the cases for the entities that have both values. For example, a sex by race crosstabs would have a total count for white females, white males, black females, black males, and so forth.

Most such reports also include row and column totals, such as males and females, and a total for each racial category. While simple counts are the most common statistic used, you can also compute averages, extrema, and more powerful statistics such as chi-squares.

Interactive packages, such as Brio, allow the user to arrange and trim the grid with interactive tools.

1.4 Presentation Graphics

The fourth kind of report was not very popular in the old days, because it had to be drawn by hand. These were the graphic presentations, such as pie charts, bar charts, and line charts.

The early computers did not have the output devices to do these things for us. I am not going to bother with a discussion of this type of report for two reasons. The first reason is that I am more concerned with data that is represented in a purely computerized format and ways to manipulate it within a computer.

The second reason is that the best material about this type of report has been presented by Edward Tufte in his books and courses (<http://www.edwardtufte.com>). His book, *The Visual Display of Quantitative Information*, is a classic that anyone who works with data should own. Stephen Few (www.perceptualedge.com) is also writing good stuff on information presentation.

1.5 Local Databases

The final kind of report is not really a traditional report at all. Personal computers and mass storage are cheap. We can send the user a small, almost personal database of his or her own to use on a local machine as he or she sees fit. This is not the same thing as sending a report to the user that cannot be altered. We download raw data from a trusted source, usually a central company database.

The user can be a subscriber who gets new data on a regular basis from a central data source. This is handy for things that have a regular



schedule, such as monthly inventory changes or sales leads. One of the best examples of this kind of reporting is continuous stock market downloads. The raw data comes from a service and gets formatted and analyzed locally by the subscriber's program.

The other model is to download data on request, but the idea of local report generation is still the same. The most common example of this is the Internet. You probably do not think of it as a reporting service.

In recent years, companies have also set up servers that do nothing but provide reports.

This Page Intentionally Left Blank

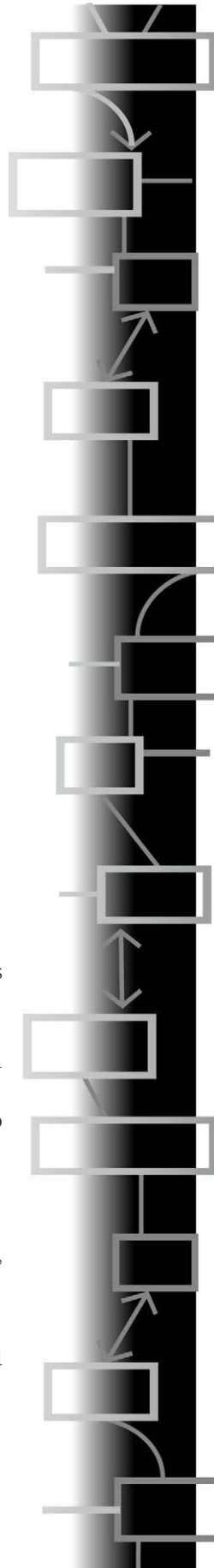
CHAPTER 2

Cross-Tabulations

A CROSS-TABULATION, or crosstabs for short, is a common statistical report. It can be done in IBM's QMF tool, using the ACROSS summary option, and in many other SQL-based reporting packages. SPSS, SAS, and other statistical packages have library procedures or language constructs for crosstabs. Many spreadsheets can load the results of SQL queries and perform a crosstabs within the spreadsheet. You can even do it with Microsoft Access (the poor man's analytics) crosstab queries!

If you can use a reporting package on the server in a client/server system instead of the following method, do so. It will run faster and in less space than the method discussed here. However, if you have to use the reporting package on the client side, the extra time required to transfer data will make these methods on the server side much faster.

A crosstabs takes the values in a column, turns them into columns in the report. For example, if you have a table with Sales data, a crosstabs would have columns for each year that appears in the sales date "year" axis and the product codes as rows on the other "Products" axis. The following SQL code is based on a trick that mathematicians call an identity matrix. When you do a CROSS JOIN with the CrossTabs table, you multiply a value by one if you want to keep it and multiply by 0 if you want to drop the value.





```
CREATE TABLE Sales
(product_name CHAR(15) NOT NULL,
 product_price DECIMAL(5,2) NOT NULL,
 qty INTEGER NOT NULL,
 sales_year INTEGER NOT NULL);
```

```
CREATE TABLE Crosstabs
(year INTEGER NOT NULL,
 year1 INTEGER NOT NULL,
 year2 INTEGER NOT NULL,
 year3 INTEGER NOT NULL,
 year4 INTEGER NOT NULL,
 year5 INTEGER NOT NULL,
 row_total INTEGER NOT NULL);
```

The table would be populated as follows:

Sales_year	year1	year2	year3	year4	year5	row_total
1990	1	0	0	0	0	1
1991	0	1	0	0	0	1
1992	0	0	1	0	0	1
1993	0	0	0	1	0	1
1994	0	0	0	0	1	1

The query to produce the report table is:

```
SELECT S1.product_name,
       SUM(S1.qty * S1.product_price * C1.year1),
       SUM(S1.qty * S1.product_price * C1.year2),
       SUM(S1.qty * S1.product_price * C1.year3),
       SUM(S1.qty * S1.product_price * C1.year4),
       SUM(S1.qty * S1.product_price * C1.year5),
       SUM(S1.qty * S1.product_price * C1.row_total)
FROM Sales AS S1, Crosstabs AS C1
WHERE S1.year = C1.year
GROUP BY S1.product_name;
```

Obviously, (S1.product_price * S1.qty) is the total dollar amount of each product in each year. The yearN column will be either a 1 or a 0. If

it is a 0, the total dollar amount in the SUM() is 0; if it is a 1, the total dollar amount in the SUM() is unchanged.

This solution lets you adjust the time frame being shown in the report by replacing the values in the year column to whichever consecutive years you wish. A two-way crosstabs takes two variables and produces a spreadsheet with all values of one variable in the rows and all values of the other represented by the columns. Each cell in the table holds the COUNT of entities that have those values for the two variables. NULLs will not fit into a crosstabs very well, unless you decide to make them a group of their own.

Another trick is to use the POSITION() function to convert a string into a 1 or a 0. For example, assume we have a “day of the week” function that returns a three-letter abbreviation, and we want to report the sales of items by day of the week in a horizontal list.

```
CREATE TABLE Weekdays
(day_name CHAR(3) NOT NULL PRIMARY KEY,
mon INTEGER NOT NULL,
tue INTEGER NOT NULL,
wed INTEGER NOT NULL,
thu INTEGER NOT NULL,
fri INTEGER NOT NULL,
sat INTEGER NOT NULL,
sun INTEGER NOT NULL);

INSERT INTO WeekDays
VALUES ('MON', 1, 0, 0, 0, 0, 0, 0),
      ('TUE', 0, 1, 0, 0, 0, 0, 0),
      ('WED', 0, 0, 1, 0, 0, 0, 0),
      ('THU', 0, 0, 0, 1, 0, 0, 0),
      ('FRI', 0, 0, 0, 0, 1, 0, 0),
      ('SAT', 0, 0, 0, 0, 0, 1, 0),
      ('SUN', 0, 0, 0, 0, 0, 0, 1);

SELECT item,
       SUM(amt * qty * mon
           * POSITION('MON' IN DOW(sales_date))) AS mon_tot,
       SUM(amt * qty * tue
           * POSITION('TUE' IN DOW(sales_date))) AS tue_tot,
       SUM(amt * qty * wed
           * POSITION('WED' IN DOW(sales_date))) AS wed_tot,
```