

TECHNOLOGY TRANSFER PRESENTS

CHRIS DATE

HOW TO WRITE CORRECT SQL AND KNOW IT

A RELATIONAL APPROACH TO SQL

OCTOBER 19-21, 2009

RESIDENZA DI RIPETTA - VIA DI RIPETTA, 231
ROME (ITALY)



info@technologytransfer.it
www.technologytransfer.it

ABOUT THIS SEMINAR

SQL is ubiquitous. But SQL is complicated, difficult, and error prone (much more so than SQL advocates would have you believe), and testing can never be exhaustive. So to have any hope of writing correct SQL, you must follow some discipline. What discipline? Answer: The discipline of using SQL relationally. But what does this mean? Isn't SQL relational anyway?

Well, of course SQL is the standard language for use with relational databases but that doesn't make it relational! The sad truth is, SQL departs from relational theory in all too many ways; duplicate rows and nulls provide two obvious examples, but they're not the only ones. Thus, systems based on SQL give you rope to hang yourself, as it were. So if you don't want to hang yourself, you need to understand relational theory (what it is and why); you need to know about SQL's departures from that theory; and you need to know how to avoid the problems they can cause. In a word, you need to use SQL relationally. Then you can behave as if SQL truly were relational, and you can enjoy the benefits of working with what is, in effect, a truly relational system. Of course, a seminar like this wouldn't be needed if everyone already used SQL relationally but they don't. On the contrary, there's a huge amount of bad practice to be observed in current SQL usage. Such practice is even recommended in textbooks and other publications, by writers who really ought to know better; in fact, a review of the literature in this regard is a pretty dispiriting exercise. The relational model first saw the light of day in 1969 yet here we are, almost 40 years later, and it still doesn't seem to be very well understood by the database community at large. Partly for such reasons, this seminar uses the relational model itself as an organizing principle; it discusses various features of the model in depth, and shows in every case how best to use SQL to implement the feature in question. Note: Classroom exercises are an integral part of the seminar, and attendee discussion and interaction are encouraged.

On completion of this seminar, attendees will:

- Appreciate how relational principles provide SQL's logical underpinnings
- Understand the breadth and depth of those principles
- Know how to formulate complex SQL code with confidence that it's correct
- Generally, be able to use SQL relationally

WHO SHOULD ATTEND

- Database application Designers and Implementers
- Information Modelers and database Designers
- Data and database Administrators
- Computer science Professors specializing in database matters
- DBMS Designers, Implementers, and other vendor personnel
- Database Consultants
- People responsible for DBMS product evaluation and acquisition

The seminar is *not* meant for beginners: Attendees will be expected to have at least an elementary familiarity with database concepts in general and the SQL language in particular. Attendees will also be expected to attempt a number of pencil and paper exercises in class. Solutions to those exercises will be discussed in class as well.

<p>1. Setting the scene</p> <ul style="list-style-type: none"> • Codd's relational model • Model vs. implementation • Properties of relations • Base vs. derived relations • Relations vs. relvars • Values vs. variables <p>2. Types and domains</p> <ul style="list-style-type: none"> • Domains are types • Types and operators • System vs. user defined types • Scalar vs. nonscalar types • Scalar types in SQL • SQL type checking and coercion • SQL collations • SQL row and table types <p>3. Tuples and relations, rows and tables</p> <ul style="list-style-type: none"> • What's a tuple? • Rows in SQL • What's a relation? • Relations are n-dimensional • Relational comparisons • TABLE_DUM and TABLE_DEE • Tables in SQL • SQL column naming <p>4. No duplicates, no nulls</p> <ul style="list-style-type: none"> • What's wrong with duplicates? • Avoiding duplicates in SQL • What's wrong with nulls? • Avoiding nulls in SQL • A remark on outer join <p>5. Base relvars, base tables</p> <ul style="list-style-type: none"> • Data definition • INSERT, DELETE, UPDATE • Updating is set level • Relational assignment • Candidate and foreign keys • Tables and predicates • Tables vs. types 	<p>6. SQL and Relational Algebra I: The original operators</p> <ul style="list-style-type: none"> • Result types • Restriction, projection, join • Union, intersection, difference • WITH and complex expressions • What expressions mean • Evaluating SQL expressions • Optimization • Importance of column names <p>7. SQL and Relational Algebra II: Additional operators</p> <ul style="list-style-type: none"> • Semijoin and semidifference • Extend • Image relations • Divide • Aggregation and summarization • Grouping and ungrouping • "What if" queries • What about ORDER BY? <p>8. SQL and constraints</p> <ul style="list-style-type: none"> • Type constraints • Type constraints in SQL • Database constraints • Database constraints in SQL • The role of transactions • Immediate vs. deferred checking • Multiple assignment • Constraints vs. predicates • The Golden Rule • Correctness vs. consistency <p>9. SQL and views</p> <ul style="list-style-type: none"> • Views are relvars • <i>The Principle of Interchangeability</i> • Views and predicates • Retrieval operations • Views and constraints • Updating operations • What are views really for? • Views and snapshots 	<p>10. SQL and logic I: Relational calculus</p> <ul style="list-style-type: none"> • Propositions and predicates • Quantification: EXISTS, FORALL, UNIQUE • Range variables and correlation • Calculus expressions • SQL support • Transforming expressions <p>11. SQL and logic II: Using logic to write SQL code</p> <ul style="list-style-type: none"> • Important identities • SQL and implication • SQL and FORALL • Correlated subqueries • Naming subexpressions • Dealing with ambiguity • Using COUNT • ALL or ANY comparisons • GROUP BY and HAVING <p>12. Further SQL topics</p> <ul style="list-style-type: none"> • Explicit tables • Range variables • Table, row, and scalar subqueries • "Possibly nondeterministic" expressions • Cursor operations • Empty set issues • A BNF grammar <p>13. The Relational Model</p> <ul style="list-style-type: none"> • The relational model vs. others • The relational model defined • What remains to be done? • The future of SQL <p>14. Database Design Theory</p> <ul style="list-style-type: none"> • The place of design theory • FDs and BCNF • JDs and 5NF • 6NF • Normalization is not a panacea • But don't denormalize! • <i>The Principle of Orthogonal Design</i> • Remarks on physical design
--	---	---

INFORMATION

<p>PARTICIPATION FEE</p> <p>€ 1500</p> <p>The fee includes all seminar documentation, luncheon and coffee breaks.</p> <p>VENUE</p> <p>Residenza di Ripetta Via di Ripetta, 231 Rome (Italy)</p> <p>SEMINAR TIMETABLE</p> <p>9.30 am - 1.00 pm 2.00 pm - 5.00 pm</p>	<p>HOW TO REGISTER</p> <p>You must send the registration form with the receipt of the payment to: TECHNOLOGY TRANSFER S.r.l. Piazza Cavour, 3 - 00193 Rome (Italy) Fax +39-06-6871102</p> <p>within October 5, 2009</p> <p>PAYMENT</p> <p>Wire transfer to: Technology Transfer S.r.l. Banca Intesa Sanpaolo S.p.A. Agenzia 6787 di Roma Iban Code: IT 34 Y 03069 05039 048890270110</p>	<p>GENERAL CONDITIONS</p> <p>GROUP DISCOUNT</p> <p>If a company registers 5 participants to the same seminar, it will pay only for 4. Those who benefit of this discount are not entitled to other discounts for the same seminar.</p> <p>EARLY REGISTRATION</p> <p>The participants who will register 30 days before the seminar are entitled to a 5% discount.</p> <p>CANCELLATION POLICY</p> <p>A full refund is given for any cancellation received more than 15 days before the seminar starts. Cancellations less than 15 days prior the event are liable for 50% of the fee. Cancellations less than one week prior to the event date will be liable for the full fee.</p> <p>CANCELLATION LIABILITY</p> <p>In the case of cancellation of an event for any reason, Technology Transfer's liability is limited to the return of the registration fee only.</p>
--	--	--

CHRIS DATE HOW TO WRITE CORRECT SQL AND KNOW IT

October 19-21, 2009
Residenza di Ripetta
Via di Ripetta, 231
Rome (Italy)

Registration fee:
€ 1500

If registered participants are unable to attend, or in case of cancellation of the seminar, the general conditions mentioned before are applicable.

first name

surname

job title

organisation

address

postcode

city

country

telephone

fax

e-mail



Stamp and signature

Send your registration form with the receipt of the payment to:
Technology Transfer S.r.l.
Piazza Cavour, 3 - 00193 Rome (Italy)
Tel. +39-06-6832227 - Fax +39-06-6871102
info@technologytransfer.it
www.technologytransfer.it



Chris Date is an independent author, lecturer, researcher, and consultant, specializing in relational database technology. He is best known for his book **An Introduction to Database Systems** (eighth edition, Addison-Wesley, 2004), which has sold over 780,000 copies and is used by several hundred colleges and universities worldwide. Mr. Date was inducted into the Computing Industry Hall of Fame in 2004. He enjoys a reputation that is second to none for his ability to communicate complex technical subjects in a clear and understandable fashion. He is also the author of many other books on database management, including most recently:

- From Addison-Wesley: **Databases, Types, and the Relational Model: The Third Manifesto** (coauthored with Hugh Darwen, 2006)
- From Apress: **Date on Database: Writings 2000-2006** (2006)
- From Trafford: **Logic and Databases: The Roots of Relational Theory** (2007)
- From Apress: **The Relational Database Dictionary, Extended Edition** (to appear 2008)
- From O'Reilly: **A Relational Approach to SQL: How to Write Correct SQL and Know It** (to appear 2008)