PPDC

Manipulating Data with Style in SQL

An introduction to SQL, the interface language to most of the world's structured data, and practices for readable and reusable SQL code

Ryan B. Harvey

October 14, 2014

Polyglot Programming DC

Relational Data

- Relational data is organized in tables consisting of columns and rows
- Fields (columns) consist of a column name and data type constraint
- Records (rows) in a table have a common field (column) structure and order
- Records (rows) are linked across tables by key fields

Relational Data Model: Codd, Edgar F. "A Relational Model of Data for Large Shared Data Banks" (1970)

Intro to SQL

- SQL ("Structured Query Language") is a declarative data definition and query language for relational data
- SQL is an ISO/IEC standard with many implementations in common database management systems (a few below)

Structured Query Language: ISO/IEC 9075 (standard), first appeared 1974, current version SQL:2011

Which database system should I use?

1. Use the one your data is in
2. Unless you need specific things (performance, functions, etc.), use the one you know best
3. If you need other stuff or you've never used a database before:

   A. SQLite: FOSS, one file DB, easy/limited

   B. PostgreSQL: FOSS, Enterprise-ready

The above are my opinions based on experience. Others may disagree, and that's OK.

SQL: Working with Objects

- Data Definition Language (DB Objects)
  - CREATE (TABLE, INDEX, VIEW, FUNCTION, ...)
  - ALTER (TABLE, INDEX, VIEW, FUNCTION, ...)
  - DROP (TABLE, INDEX, VIEW, FUNCTION, ...)

SQL: Working with Rows

- Data Manipulation Language (Records) aka Query Language
  - SELECT … FROM …
  - INSERT INTO …
  - UPDATE … SET …
  - DELETE FROM …

SQL: SELECT Statement

- SELECT <col_list> FROM <table> ...
  - Merging/Column Binding: JOIN clause
  - Row binding: UNION clause
  - Filtering: WHERE clause
  - Aggregation: GROUP BY clause
  - Aggregated filtering: HAVING clause
  - Sorting: ORDER BY clause

Intro to Relational Algebra

- Basic operators

SELECT	σ	WHERE, HAVING
PROJECT	Π	<COL_LIST>
RENAME	ρ	AS

- Join operators: inner/outer, cartesian
- Set operators: union, intersect, set minus, and, or, etc.
- SELECT name, id FROM t1 WHERE id<3 AND dob<DATE '2004-01-01'

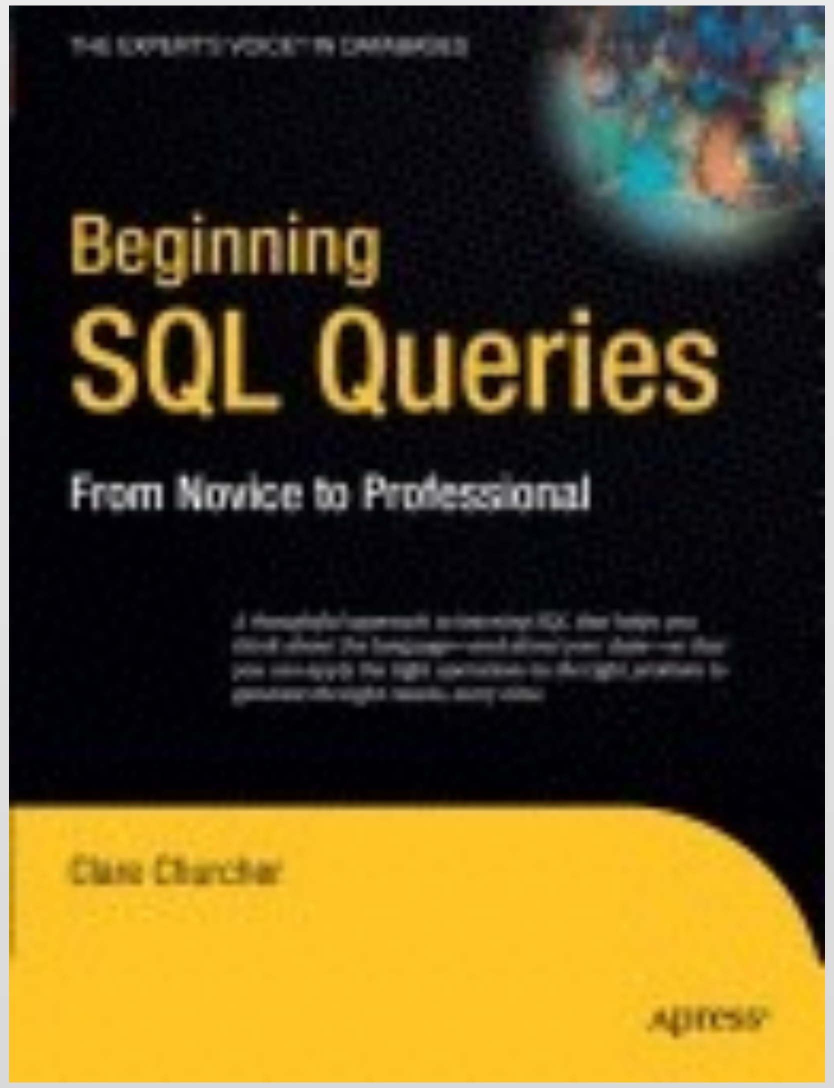
$$\Pi_{NAME,ID}\ \sigma_{ID<3\ \wedge\ DOB<(1/1/2004)}\ (T1)$$

For a very detailed Intro to Relational Algebra, see lecture notes from 2005 databases course, IT U of Copenhagen

PPDC

SQL Beginner Resources

- Basic SQL Commands Reference: http://www.cs.utexas.edu/~mitra/csFall2013/cs329/lectures/sql.html



THE EXPERT'S VOICE IN DATABASE

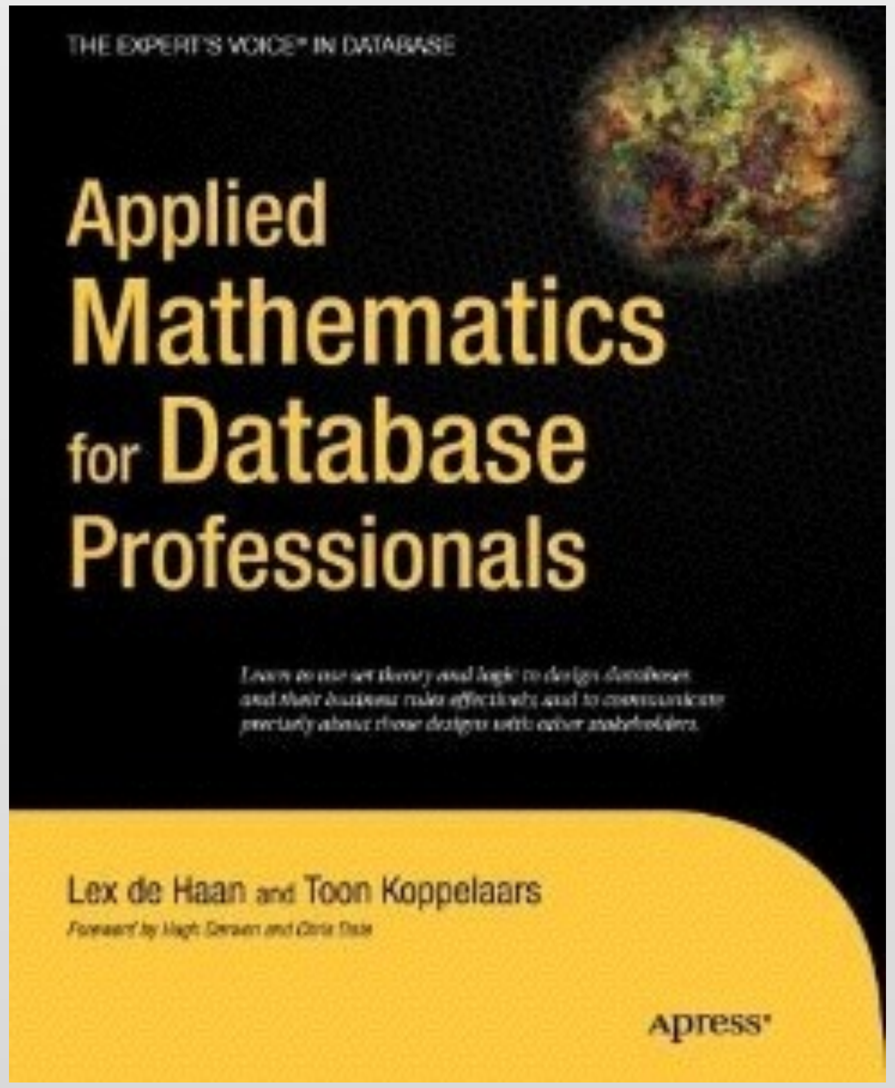
Beginning SQL Queries

From Novice to Professional

[illegible]

Clare Churcher

Apress



THE EXPERT'S VOICE IN DATABASE

Applied Mathematics for Database Professionals

Learn to use set theory and logic to design databases and their business rules effectively, and to communicate precisely about those designs with other stakeholders.

Lex de Haan and Toon Koppelaars

Foreword by Hugh Darwen and Chris Date

Apress

FEATURE COMPARISON

SQL: Common Table Expressions (CTEs)

P
P
D
C

- WITH <name> [(<col_list>)] AS (SELECT ...)
- SELECT <col_list> FROM <table or CTE> ...
  - Merging/Column Binding: JOIN clause
  - Row binding: UNION clause
  - Filtering: WHERE clause
  - Aggregation: GROUP BY clause
  - Aggregated filtering: HAVING clause
  - Sorting: ORDER BY clause

Same as before!

SQL: Views from SELECTs

- CREATE VIEW <name> AS ...
- SELECT <col_list> FROM <table> ...
  - Merging/Column Binding: JOIN clause
  - Row binding: UNION clause
  - Filtering: WHERE clause
  - Aggregation: GROUP BY clause
  - Aggregated filtering: HAVING clause
  - Sorting: ORDER BY clause

SQL: Functions from Views

- CREATE FUNCTION <name> (<params>) AS ...
- SELECT ... <params> ...
  - Merging/Column Binding: JOIN clause
  - Row binding: UNION clause
  - Filtering: WHERE clause
  - Aggregation: GROUP BY clause
  - Aggregated filtering: HAVING clause
  - Sorting: ORDER BY clause

PPDC

SQL: Tuning with EXPLAIN

- EXPLAIN <options> SELECT … ← Same as before!
  - rows scanned: COST option
  - wordy response: VERBOSE option
  - output formatting: FORMAT option
  - actually run it: ANALYZE option
  - runtime (only with ANALYZE): TIMING option
- (EXPLAIN is not part of the SQL standard, but most implementations support it)

SQL: Tuning using Indexes

- CREATE INDEX <name> ON <table> (<col_list|expression>) ...
  - UNIQUE indices for key fields
  - Use functions in expressions: LOWER(<text_col>), INT(<num_col>)
  - Specify ordering (ASC, DESC, NULLS FIRST, etc.) and method (BTREE, HASH, GIST, etc.)
  - Partial indexes via WHERE clause

What's in your WHERE clause?

SQL in other languages

(or, accessing data in databases via SQL in other languages)

- R with libraries
  - RPostgreSQL, dplyr
- Python with modules
  - psycopg2, SQLAlchemy

SQL in other languages

(or, operating on other languages' data structures via SQL)

- R with libraries
  - RSQLite, sqldf
- Python with modules
  - Pandas, PandaSQL

Mostly, Data Frames.

Now, let's look at <u>some code</u>!

Ryan B. Harvey

http://datascientist.guru

ryan.b.harvey@gmail.com

@nihonjinrxs

+ryan.b.harvey

Employment & Affiliations*

IT Project Manager
Office of Management and Budget
Executive Office of the President

Data Scientist & Software Architect
Kitchology Inc.

Research Affiliate
Norbert Wiener Center for Harmonic Analysis & Applications
College of Computer, Mathematical & Natural Sciences
University of Maryland at College Park

Thank you!

Questions?

* My remarks, presentation and prepared materials are my own, and do not represent the views of my employers.