

# SQL: THE SEQUEL

MORE SQL IN THE DATABASE, AND  
USING SQL IN DATA SCIENCE CONTEXTS



RYAN B. HARVEY

AUGUST 6, 2014



# REVIEW: 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

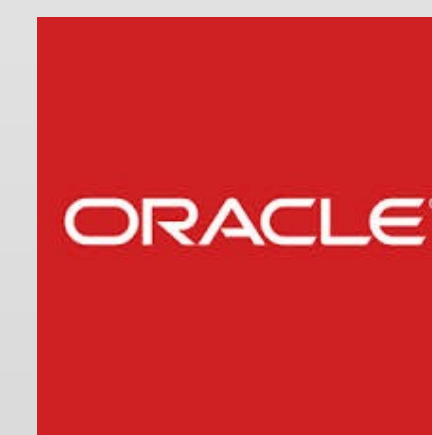
# REVIEW: WHY SHOULD I USE A DATABASE SYSTEM?

1. YOU CARE ABOUT STRONG DATA TYPES, TYPE VALIDATION AND DATA ACCESS CONTROLS
2. YOU NEED TO RELATE MULTIPLE TABLES TOGETHER VIA COMMON FIELDS
3. YOUR DATA IS LARGER THAN A FEW 10s TO 100 MB, MAKING FILE PARSING ONEROUS
4. YOU NEED TO SUBSET OR AGGREGATE YOUR DATA OFTEN BASED ON FIELD VALUES

THE ABOVE ARE MY OPINIONS BASED ON EXPERIENCE. OTHERS MAY DISAGREE, AND THAT'S OK.

# REVIEW: 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)



# REVIEW: 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

- QUERY LANGUAGE (RECORDS)
  - SELECT ... FROM ...
  - INSERT INTO ...
  - UPDATE ... SET ...
  - DELETE FROM ...

# SQL: SELECT STATEMENT

- `SELECT <COL_LIST> FROM <TABLE> ...`
- MERGING: `JOIN` CLAUSE
- ROW BINDING: `UNION` CLAUSE
- FILTERING: `WHERE` CLAUSE
- AGGREGATION: `GROUP BY` CLAUSE
- AGGREGATED FILTERING: `HAVING` CLAUSE
- SORTING: `ORDER BY` CLAUSE

You'll  
remember  
this from  
last time





# SQL: VIEWS FROM SELECTs

- CREATE VIEW <NAME> AS ...
- SELECT <COL\_LIST> FROM <TABLE> ...
- MERGING: JOIN CLAUSE
- ROW BINDING: UNION CLAUSE
- FILTERING: WHERE CLAUSE
- AGGREGATION: GROUP BY CLAUSE
- AGGREGATED FILTERING: HAVING CLAUSE
- SORTING: ORDER BY CLAUSE

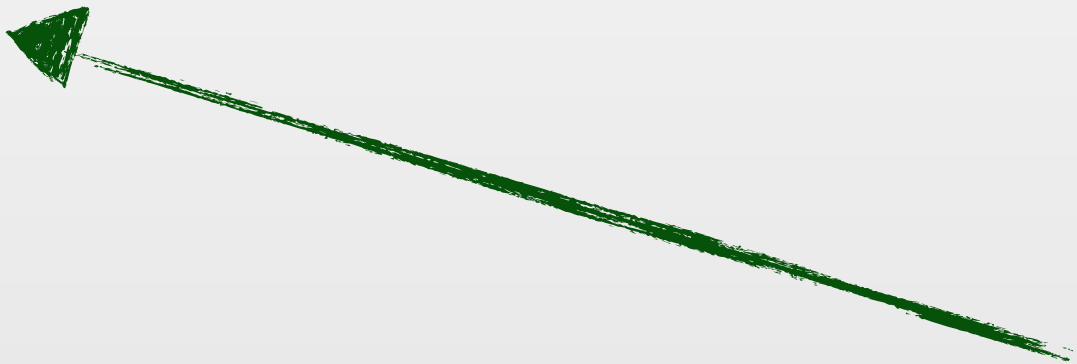
Same  
as  
before!



# SQL: FUNCTIONS FROM VIEWS

- CREATE FUNCTION <NAME> (<PARAMS>) AS ...
- SELECT ... <PARAMS> ...
- MERGING: JOIN CLAUSE
- ROW BINDING: UNION CLAUSE
- FILTERING: WHERE CLAUSE
- AGGREGATION: GROUP BY CLAUSE
- AGGREGATED FILTERING: HAVING CLAUSE
- SORTING: ORDER BY CLAUSE

Almost  
same  
as  
before!



# 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)

# SQL: TUNING USING INDEXES

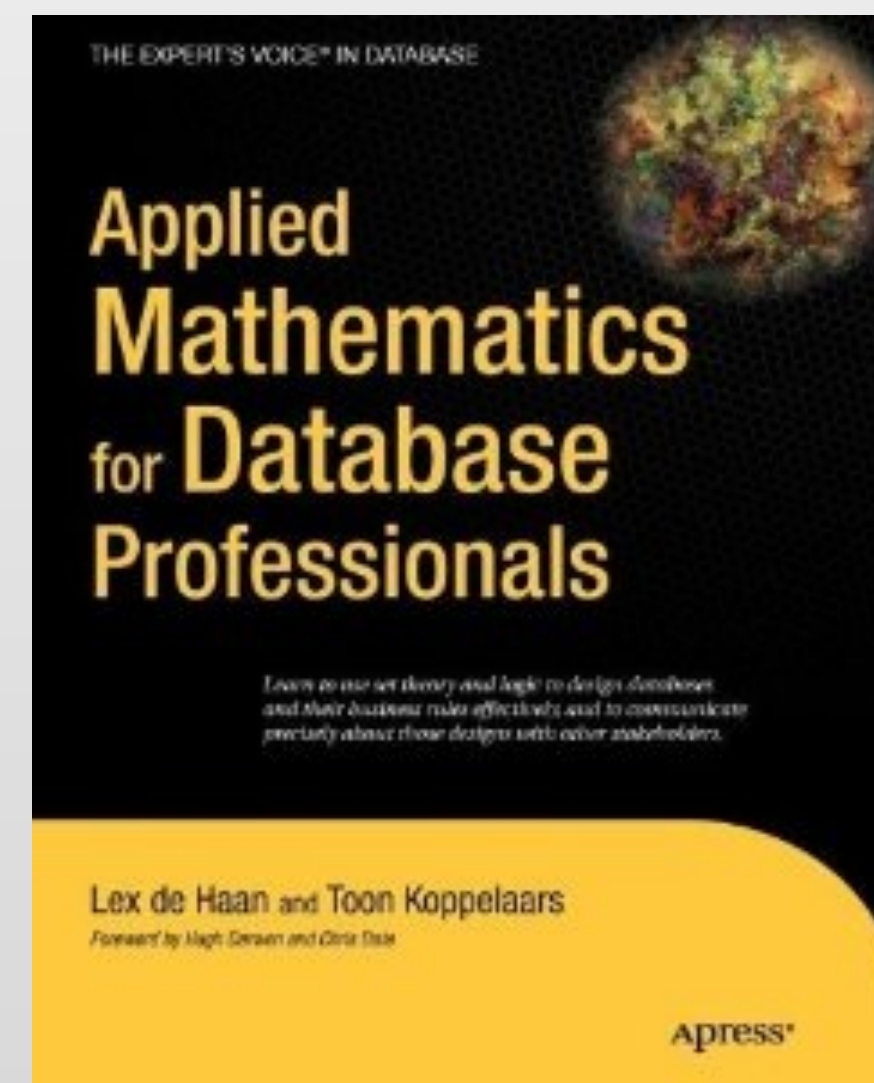
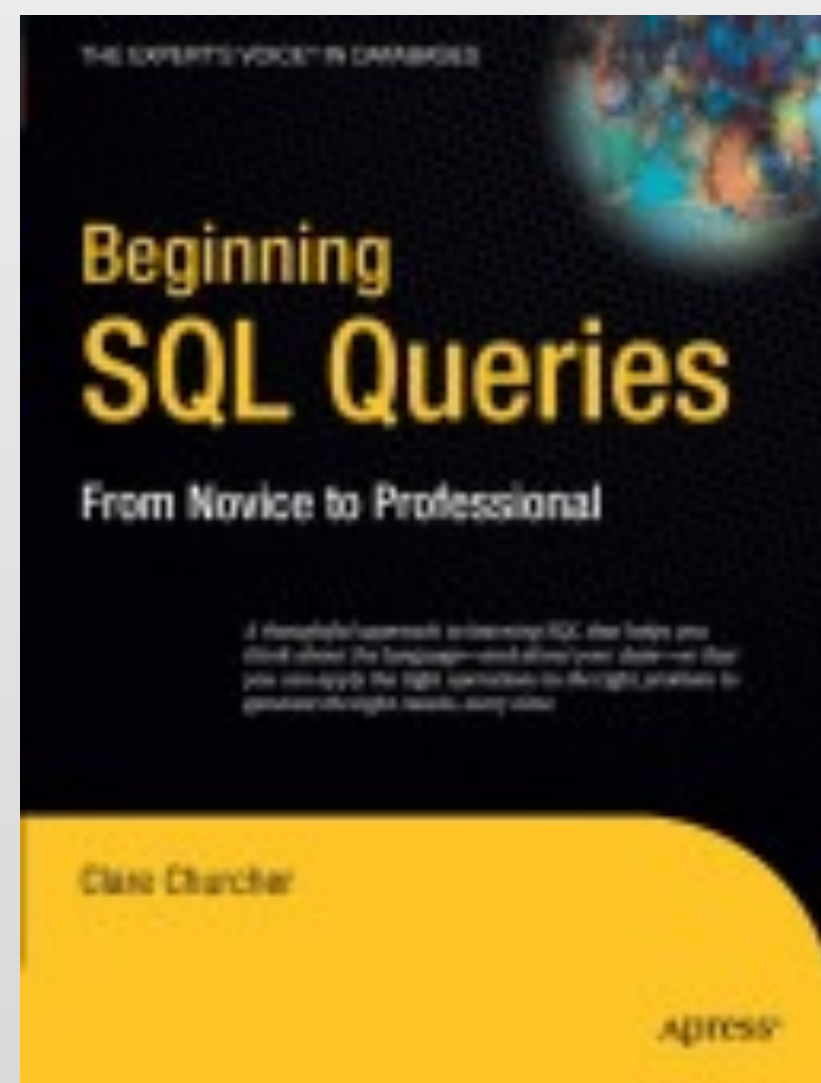
- `CREATE INDEX <NAME> ON <TABLE>`  
`(<COL_LIST|EXPRESSION>) ...`

What's in  
your  
WHERE  
clause?

- `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

# SQL BEGINNER RESOURCES

- BASIC SQL COMMANDS REFERENCE:  
[HTTP://WWW.CS.UTEXAS.EDU/~MITRA/  
CSFALL2013/CS329/LECTURES/SQL.HTML](http://www.cs.utexas.edu/~MITRA/csfall2013/cs329/lectures/sql.html)



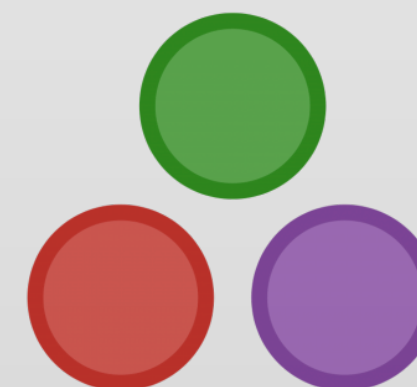
Same  
as  
before!

Still  
useful!

# SQL IN OTHER LANGUAGES

(OR, ACCESSING DATA IN DATABASES VIA SQL IN OTHER LANGUAGES)

- R WITH LIBRARIES
  - RPOSTGRESQL, DPLYR
- PYTHON WITH MODULES
  - PSYCOPG2, SQLALCHEMY
- JULIA WITH PACKAGES (IN DEV)
  - POSTGRESQL, DBI

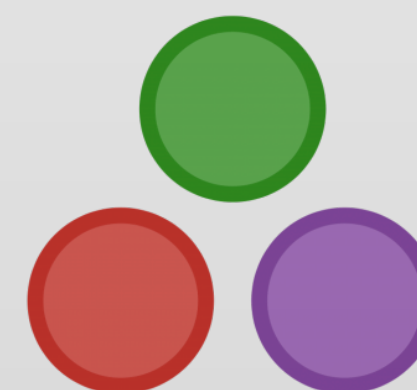


You'll  
remember  
this from  
last time

# SQL IN OTHER LANGUAGES

(OR, OPERATING ON OTHER LANGUAGES' DATA STRUCTURES VIA SQL)

- R WITH LIBRARIES
  - RSQLite, SQLDF
- PYTHON WITH MODULES
  - PANDAS, PANDASSQL
- JULIA WITH PACKAGES (IN DEV)
  - SQLITE, DBI



Mostly,  
Data  
Frames.



NOW, LET'S LOOK AT  
SOME CODE!





**RYAN B. HARVEY**

[HTTP://DATASCIENTIST.GURU](http://datascientist.guru)  
[RYAN.B.HARVEY@GMAIL.COM](mailto:RYAN.B.HARVEY@GMAIL.COM)  
[@NIHONJINRXS](#)  
[+RYAN.B.HARVEY](#)

D  
W  
D  
C

**DAY JOB**

**IT PROJECT MANAGER**

**OFFICE OF MANAGEMENT AND BUDGET**

**EXECUTIVE OFFICE OF THE PRESIDENT**

**SIDE JOB**

**DATA SCIENTIST & SOFTWARE ARCHITECT**

**KITCHOLOGY INC.**