Discussion 10:
SQL

.....

TA: **Jerry Chen**
Email: **jerry.c@berkeley.edu**
TA Website: **jerryjrchen.com/cs61a**

Agenda

1. Attendance
2. Announcements
3. SQL

Attendance

Sign in at bit.do/jerrydisc

OR

Come to me for check-in

Announcements

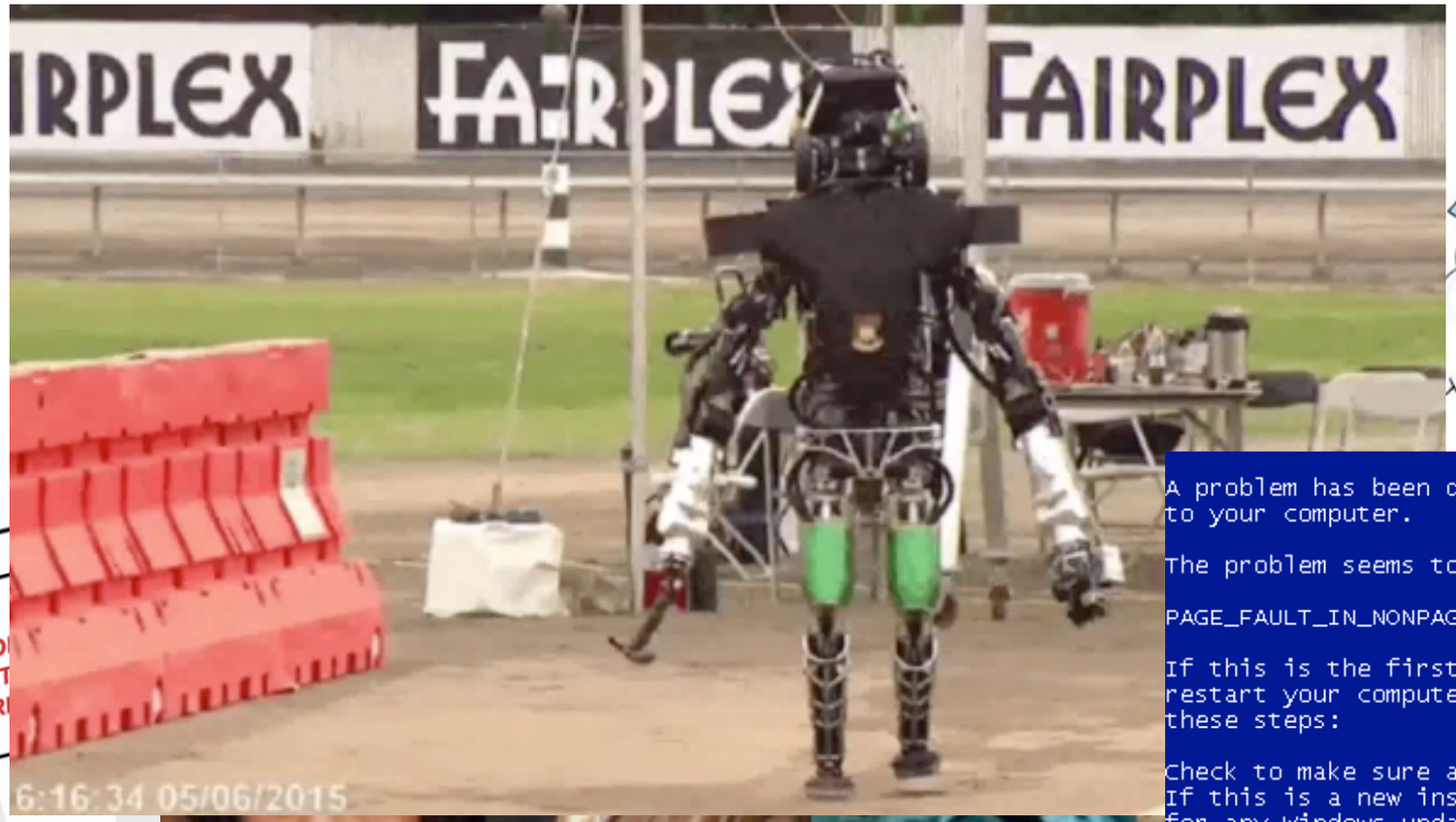
Scheme proj due tonight!

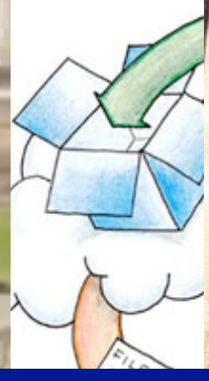
Hw 13 is due next Wed

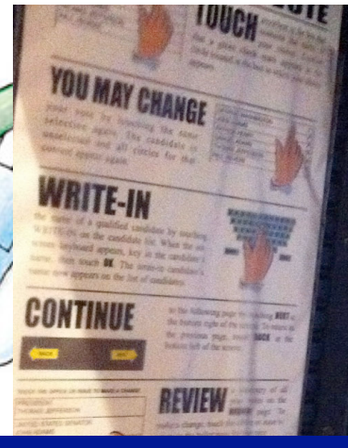
No section, no in-person lab next week

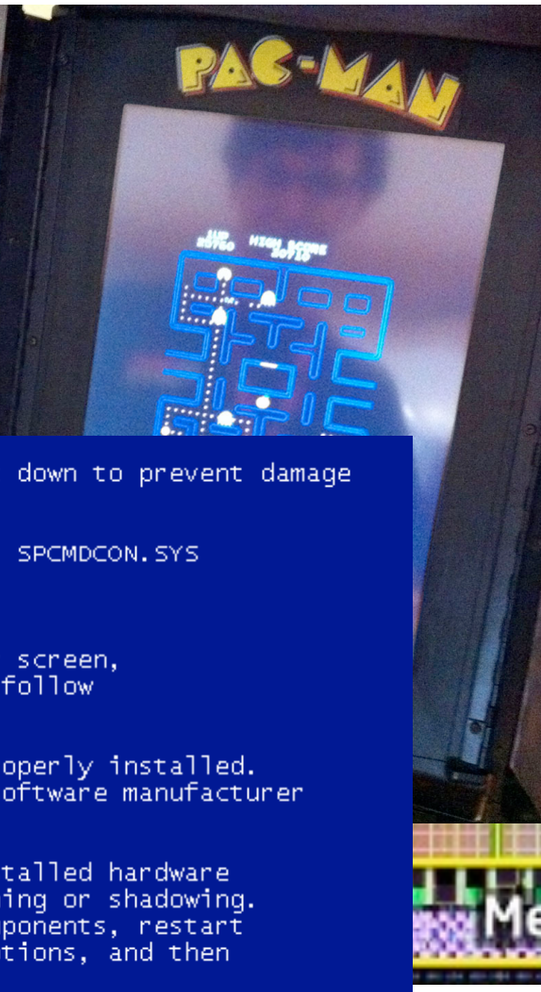
Last section will be a little bit different…

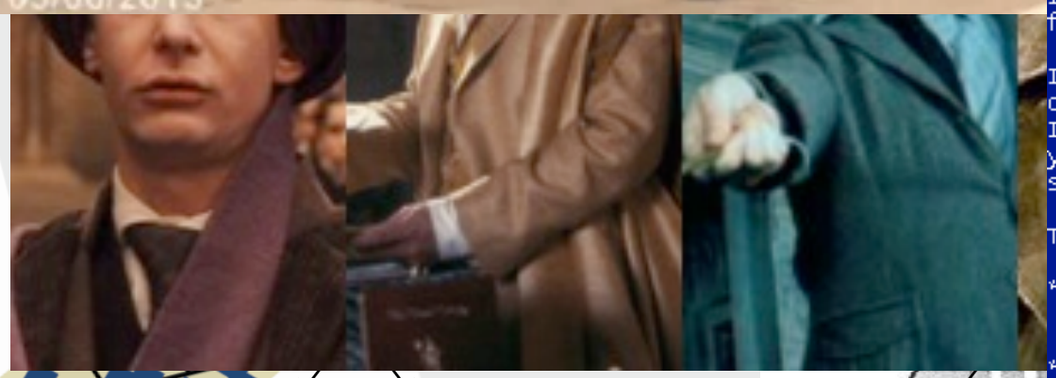
Discussion 11 Preview

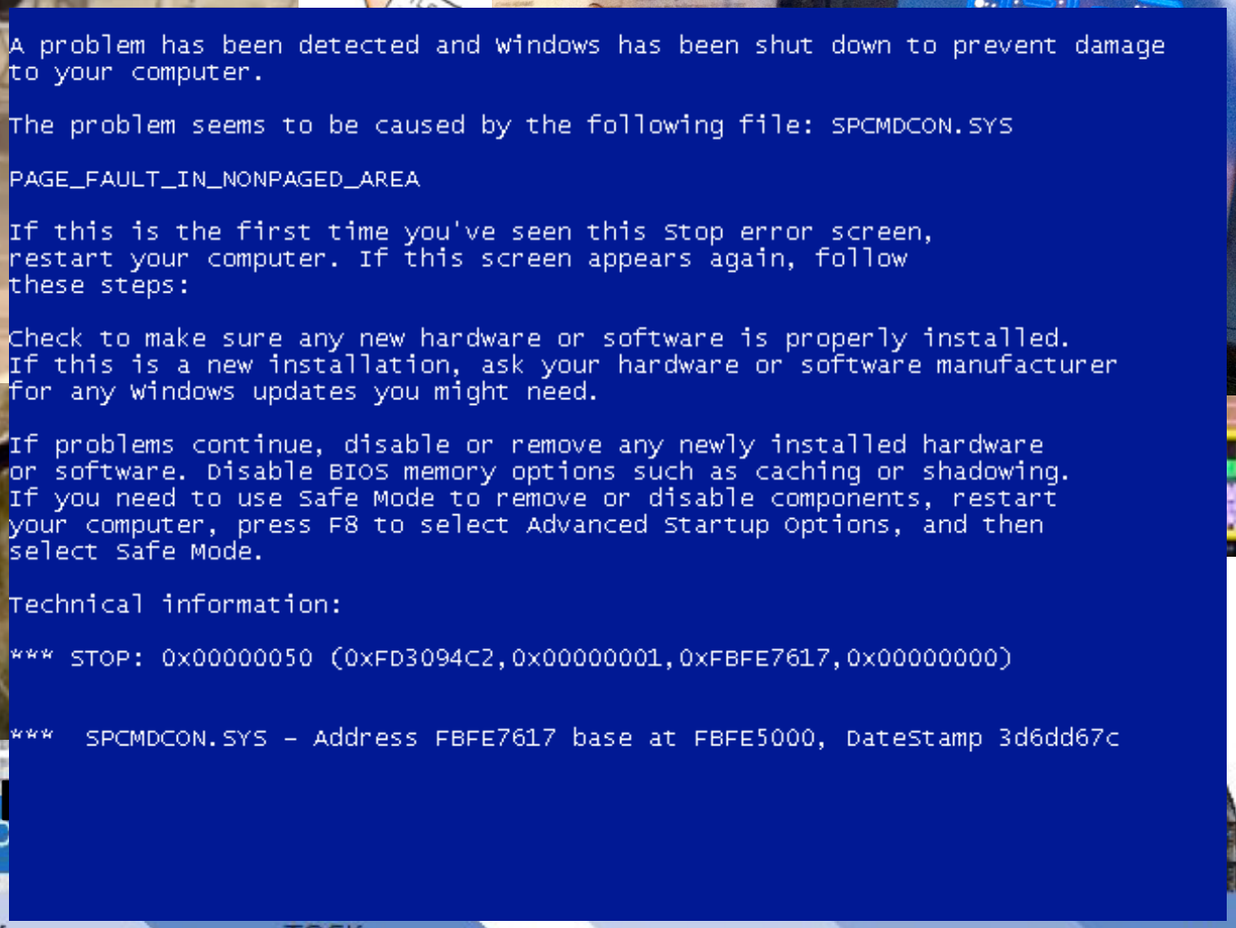












Databases


ORACLE
DATABASE
Microsoft
SQL Server
PostgreSQL
TERADATA
MariaDB
MySQL
IBM
DB2
SYBASE
H2
Access
Apache Derby
HIVE
SQLite
HyperSQL

Databases

Data — information about pretty much anything

A **database** is an ordered collection of data

Use **tables** to organize data

Databases show up everywhere!



SQL

Structured Query Language

(Pronounced "Ess Cue El" or "Sequel")

Used to manage data stored in a database

A **declarative** language — broadly speaking, tell it **what we want**, not how to do it

All "queries" (expressions) end in a semicolon ";"

SQL

The **select** statement create tables

- Use the **union** command to join two select rows

The **create table** expression saves a table for later

Select

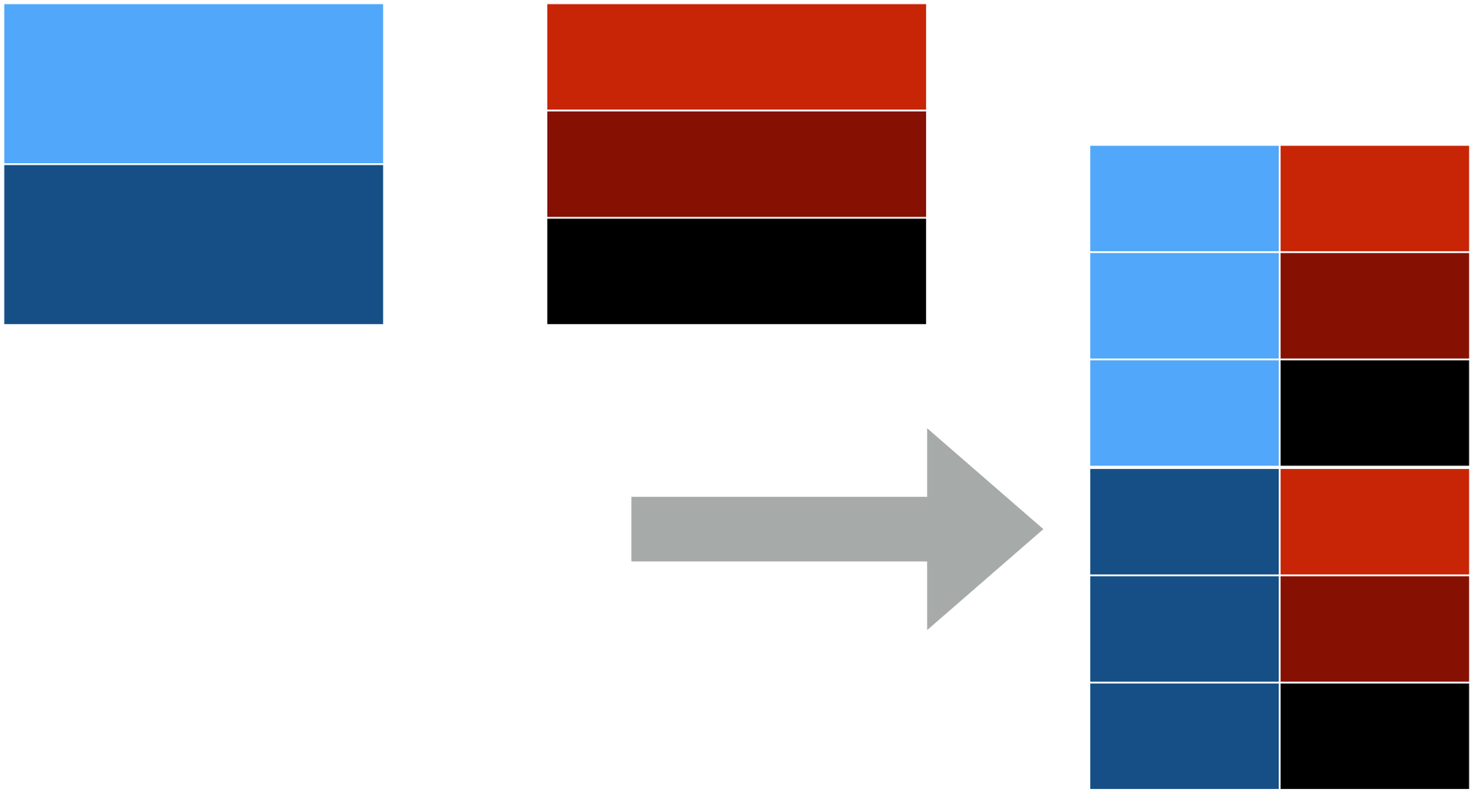
select doesn't have to start from scratch: can select **from** an existing table to create a new one

Specify what columns to keep in your result!

Filter results using boolean expressions in the **where clause**

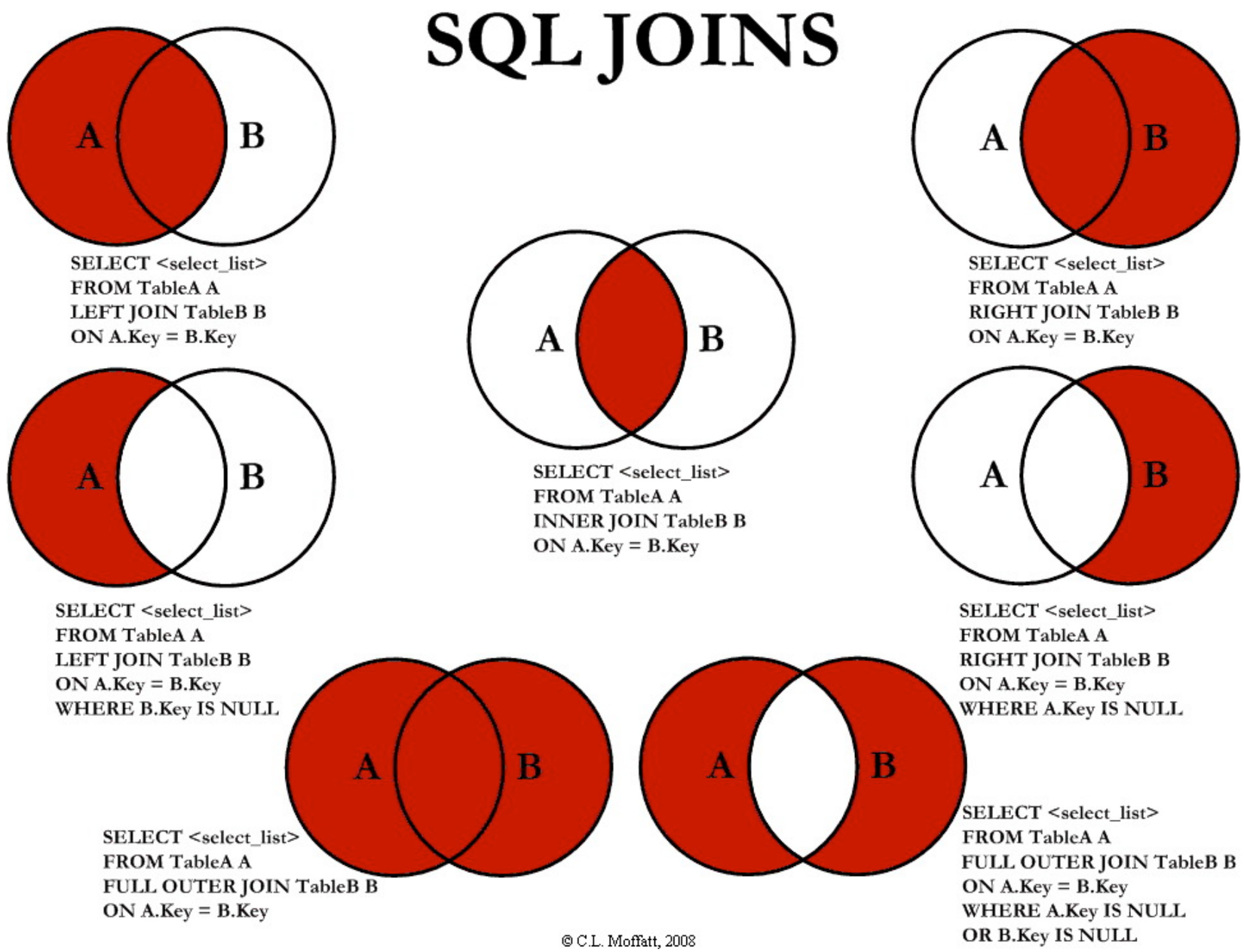
Joins

When we join two tables together, consider all possible pairings:



Joins

Of course, it gets more complicated (out of scope)



Joins

If we're joining with ourself (or a table with the same column names), we may require **aliasing**

Be wary of **duplicates** and **self-joins** (row joined to itself)!

- Solve by **enforcing ordering**

Recursive Select

Like regular recursion start off with a **base row** (base case)

Subsequent rows based off of previous rules (recursive step)

Use filter (where) to determine when to stop

We usually use a local **with** table to create recursive tables