Databases, continued.

SELECT Statement

- SELECT ra, dec, psfMag_r FROM sources
- SELECT ra, dec, psfMag_r FROM sources WHERE psfMag_r < 21.5
- SELECT ra, dec, psfMag_r FROM sources WHERE psfMag_r < 21.5 LIMIT 5
- SELECT COUNT(psfMag_r), AVG(psfMag_r) FROM sources WHERE psfMag_r < 21.5
- SELECT COUNT(*), run FROM sources GROUP BY run
- SELECT COUNT(*), run FROM sources GROUP BY run ORDER BY run
- SELECT COUNT(*) as ct , run FROM sources GROUP BY run ORDER BY ct

NULL

- How do we mark missing data?
  - Typical way to do this is to designate a value as "magic"
  - E.g.,: -9999 in our example database
- Relational databases provide us with a special constant, a "NULL"
  - The meaning is always clear (i.e. – no data)
  - Plays well with aggregate functions
    - I.e., AVG(), COUNT() ignore null values

UPDATE

- UPDATE sources

  SET psfMag_r = NULL

  WHERE psfMag_r = -9999.0

The table to update

Columns to update (and the values to use)

Selecting the subset of rows to update

JOIN: Joining tables

- Example:
  - Each row in the 'sources' table has a `run' entry – the ID of the SDSS run where this object was observed
  - Each entry in the 'runs' table has a 'mjdstart' entry, indicating the time when the observing for this run started
  - How can we find the mjdstart for each object? An algorithm for doing it by hand:
    - For each row in the sources table:
      - Read off the value of 'run'
      - Find the corresponding row in the 'runs' table
      - Read off the value of mjdstart

JOIN: Joining tables

The columns we're interested in.

Those appearing in more than one table need to be prefixed by the table name.

- SELECT

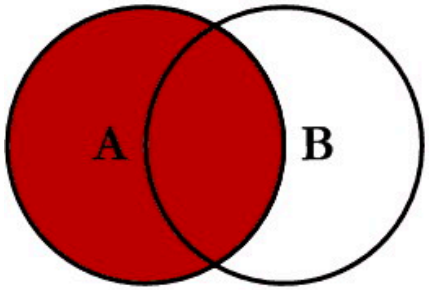
  sources.ra, sources.dec, sources.run, mjdstart

  FROM

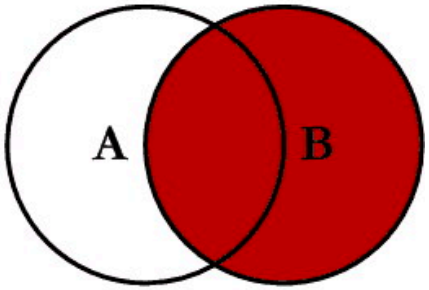
  sources JOIN runs ON sources.run = runs.run

Instructions how to "join" (how to match together) the rows in sources and runs tables

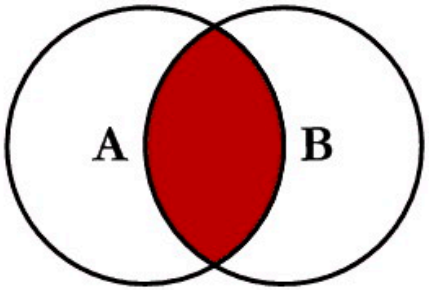
SQL JOINS



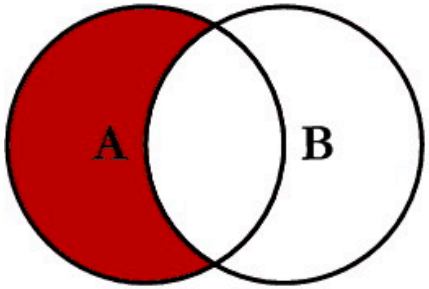

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
```



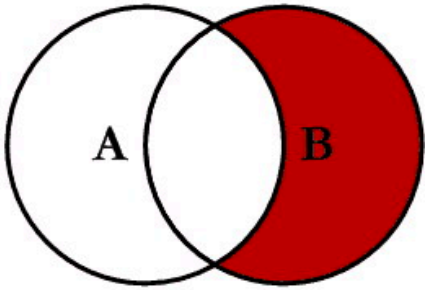

```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
```



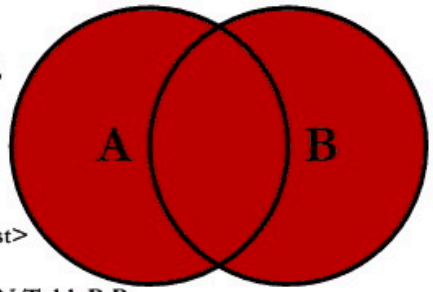

```
SELECT <select_list>
FROM TableA A
INNER JOIN TableB B
ON A.Key = B.Key
```



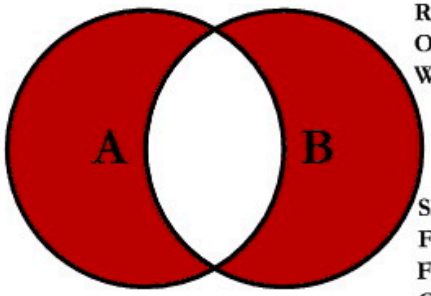

```
SELECT <select_list>
FROM TableA A
LEFT JOIN TableB B
ON A.Key = B.Key
WHERE B.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
RIGHT JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
```




```
SELECT <select_list>
FROM TableA A
FULL OUTER JOIN TableB B
ON A.Key = B.Key
WHERE A.Key IS NULL
OR B.Key IS NULL
```


http://www.codeproject.com/KB/database/Visual_SQL_Joins/Visual_SQL_JOINS_orig.jpg

More information

- http://en.wikipedia.org/wiki/Join_%28SQL%29
- http://blog.codinghorror.com/a-visual-explanation-of-sql-joins/

Doing all of this from Python

- Python can connect to a variety of databases
- SQLite module comes built into Python (sqlite3)

- We will also use a library called pandas ("Python Data Analysis Library")
  - http://pandas.pydata.org
  - Pandas provides high-performance data structures for manipulating and analyzing tabular data

Demo (IPython notebook)

More about SQL & Databases

- Interactive SQL tutorial
  - http://sqlzoo.net/wiki/Main_Page
- Introduction to SQL (Stanford)
  - https://class.stanford.edu/courses/DB/SQL/SelfPaced/courseware/ch-sql/seq-vid-introduction_to_sql/
- Introduction to SQL (Phil Spector, Berkeley)
  - https://www.stat.berkeley.edu/~spector/sql.pdf
- Databases in depth: CSE444
  - http://courses.cs.washington.edu/courses/cse444/