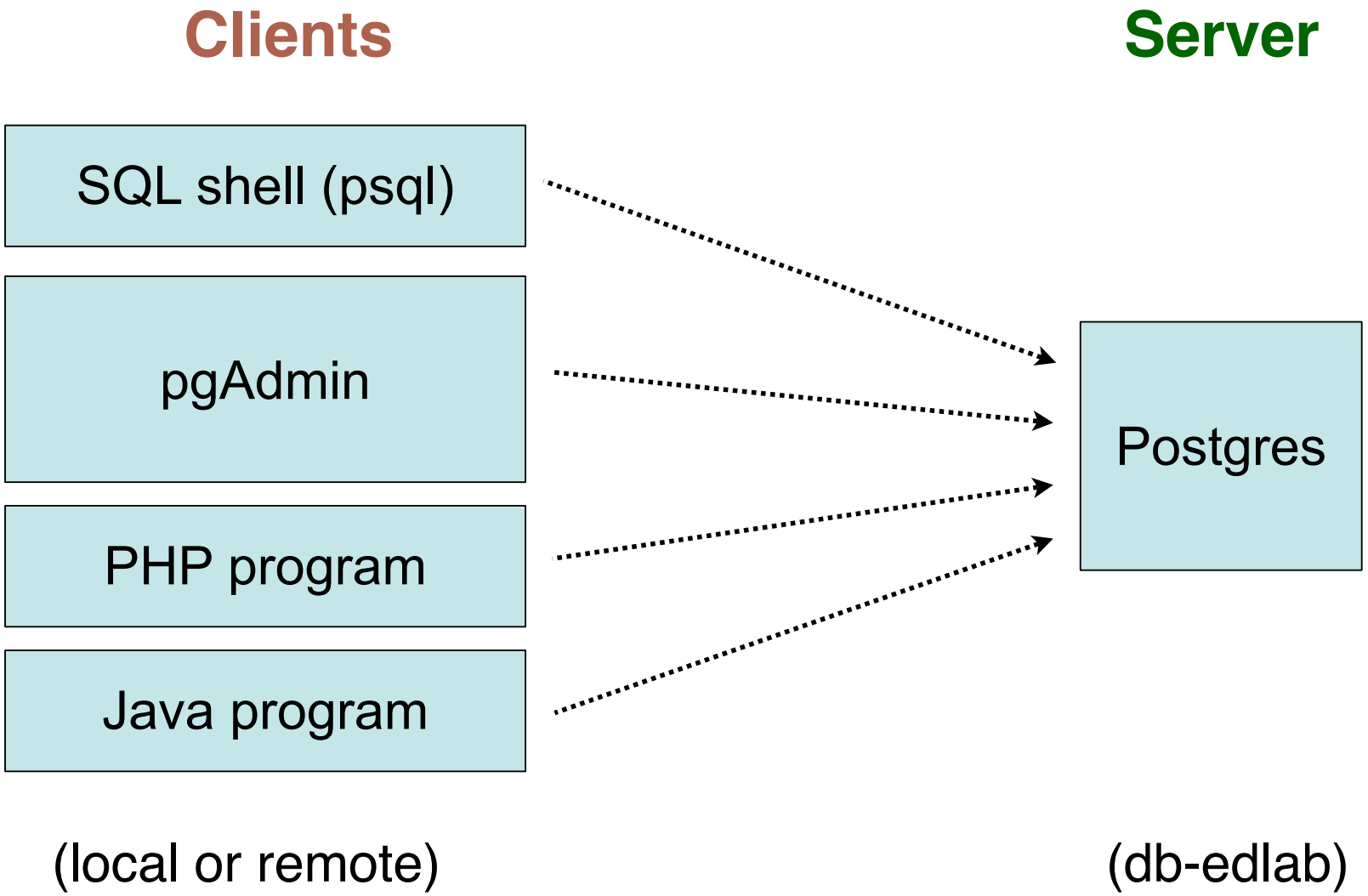
Connecting to Postgres



Getting started with Postgres

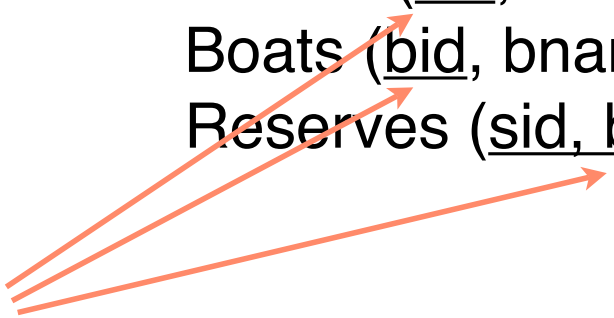
- Resources available
  - –course website, from “Assignments” page
  - –Postgres Reference Manual
    - http://www.postgresql.org/docs/8.1/static/index.html
    - especially, Part I: Tutorial

SQL Overview

- Query capabilities
  - SELECT-FROM-WHERE blocks,
  - Basic features, ordering, duplicates
  - Set operations (union, intersect, except)
  - Aggregation & Grouping
  - Nested queries (correlation)
  - Null values

Example database

Sailors (sid, sname, rating, age)
Boats (bid, bname, color)
Reserves (sid, bid, day)



Key for each table indicated by underlined attributes.

Sailors

sid	sname	rating	age
29	brutus	1	33
85	art	3	25.5
95	bob	3	63.5
96	frodo	3	25.5
22	dustin	7	45
64	horatio	7	35
31	lubber	8	55.5
32	andy	8	25.5
74	horatio	9	35
58	rusty	10	35
71	zorba	10	16

Reserves

sid	bid	day
22	101	10/10
22	102	10/10
22	103	10/8
22	104	10/7
31	102	11/10
31	103	11/6
31	104	11/12
64	101	9/5
64	102	9/8
74	103	9/8

Boats

bid	bname	color
101	Interlake	blue
102	Interlake	red
103	Clipper	green
104	Marine	red

SQL Query

Basic form: (plus many many extensions)

```
SELECT    [DISTINCT]  target-list
FROM      relation-list
WHERE     qualification conditions
```

For example:

```
SELECT    sid, sname, rating, age
FROM      Sailors
WHERE     age > 21
```

Basic SQL Query

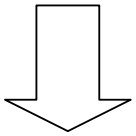
```
SELECT    [DISTINCT] target-list
FROM      relation-list
WHERE     qualification
```

- ***target-list*** A list of attributes of relations in *relation-list*
- ***relation-list*** A list of relation names (possibly with a *range-variable* after each name).
- ***qualification*** Comparisons (Attr *op* const or Attr1 *op* Attr2, where *op* is one of $<, >, =, \leq, \geq, \neq$) combined using AND, OR and NOT.
- DISTINCT is an optional keyword indicating that the answer should not contain duplicates. Default is that duplicates are *not* eliminated!

Simple SQL Query

Sailors

sid	sname	rating	age
22	dustin	7	45
31	lubber	8	55.5
58	rusty	10	35
71	zorba	10	16



```
SELECT   *
FROM     Sailors
WHERE    age > 21
```

sid	sname	rating	age
22	dustin	7	45
31	lubber	8	55.5
58	rusty	10	35

Conditions in the WHERE clause are like selection: $\sigma_{age>21}$

Selection conditions

What goes in the WHERE clause:

- x = y, x < y, x <= y, x != y, etc
  - For number, they have the usual meanings
  - For CHAR and VARCHAR: lexicographic ordering
  - For dates and times, what you expect...
- Also, pattern matching on strings: s **LIKE** p

The **LIKE** operator

- s **LIKE** p: pattern matching on strings
- p may contain two special symbols:
  - % = any sequence of characters
  - _ = any single character

Find all students whose name begins and ends with 'b':

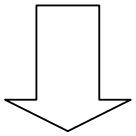
```
SELECT  *
FROM    Sailors
WHERE   sname LIKE 'b%b'
```

sid	sname	rating	age
29	brutus	1	33
85	art	3	25.5
95	bob	3	63.5
96	frodo	3	25.5
22	dustin	7	45
64	horatio	7	35

Simple SQL Query

Sailors

sid	sname	rating	age
22	dustin	7	45
31	lubber	8	55.5
58	rusty	10	35
71	zorba	10	16



```
SELECT   sname, age
FROM     Sailors
WHERE    age > 21
```

sname	age
dustin	45
lubber	55.5
rusty	35

Conditions in the SELECT clause are like projection: $\Pi_{sname,age}$

Note confusing terminology ☹

```
SELECT    sname, age
FROM      Sailors
WHERE     age > 21
```

Conditions in the WHERE clause are like selection: $\sigma_{age<21}$

Conditions in the SELECT clause are like projection: $\Pi_{sname,age}$

Eliminating Duplicates

```
SELECT  DISTINCT sname
FROM    Sailors
```

Compare to:

```
SELECT  sname
FROM    Sailors
```

sname
brutus
art
bob
frodo
dustin
horatio
lubber
andy

sname
brutus
art
bob
frodo
dustin
horatio
lubber
andy
horatio

Default behavior does not eliminate duplicates.

Ordering the Results

```
SELECT   sname, rating, age
FROM     Sailors
WHERE    age > 18
ORDER BY  rating, sname
```

Ordering is ascending, unless you specify the DESC keyword.

Ties are broken by the second attribute on the ORDER BY list, etc.

Conceptual Evaluation Strategy

```
SELECT      [DISTINCT]  target-list
FROM        relation-list
WHERE       qualifications
```

- Semantics of an SQL query defined in terms of a conceptual evaluation strategy:
  - Compute the cross-product of ***relation-list***.
  - Discard resulting tuples if they fail ***qualifications***.
  - Delete attributes that are not in ***target-list***.
  - If **DISTINCT** is specified, eliminate duplicate rows.
- Probably the least efficient way to compute a query -- optimizer will find more efficient plan.

RA equiv:

$\times$

σ

Π

Example of Conceptual Evaluation

sid	sname	rating	age
22	dustin	7	45.0
31	lubber	8	55.5
58	rusty	10	35.0

sid	bid	day
22	101	10/10/96
58	103	11/12/96

```
SELECT  S.sname
FROM    Sailors S, Reserves R
WHERE   S.sid=R.sid AND R.bid=103
```

(sid)	sname	rating	age	(sid)	bid	day
22	dustin	7	45.0	22	101	10/10/96
22	dustin	7	45.0	58	103	11/12/96
31	lubber	8	55.5	22	101	10/10/96
31	lubber	8	55.5	58	103	11/12/96
58	rusty	10	35.0	22	101	10/10/96
58	rusty	10	35.0	58	103	11/12/96

Example

```
SELECT sname
FROM Sailors S, Reserves R, Boats B
WHERE S.sid = R.sid AND R.bid = B.bid
      AND B.color = 'red'
```

What does this query compute?

Find the names of sailors who have reserved a red boat

Please write in SQL

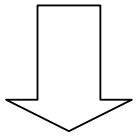
Find the colors of boats reserved by 'Lubber'

```
SELECT B.color
FROM Sailors S, Reserves R, Boats B
WHERE S.sid = R.sid AND R.bid = B.bid
        AND S.sname = 'Lubber'
```

Renaming Columns

Sailors

sid	sname	rating	age
22	dustin	7	45
31	lubber	8	55.5
58	rusty	10	35
71	zorba	10	16



```
SELECT   sname AS name, age AS x
FROM     Sailors
WHERE    age > 21
```

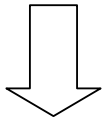
name	x
dustin	45
lubber	55.5
rusty	35

Disambiguating Attributes

- Sometimes two relations have the same attr:
  Person(pname, address, worksfor)
  Company(cname, address)

```
SELECT   DISTINCT pname, address
FROM     Person, Company
WHERE    worksfor = cname
```

Which address ?



```
SELECT   DISTINCT Person.pname, Company.address
FROM     Person, Company
WHERE    Person.worksfor = Company.cname
```

Range Variables in SQL

Purchase (buyer, seller, store, product)

Find all stores that sold at least one product that was sold at 'BestBuy':

```
SELECT DISTINCT  x.store
FROM     Purchase AS x, Purchase AS y
WHERE   x.product = y.product AND y.store = 'BestBuy'
```

Please write in SQL

Self-join on Flights:
The departure and arrival cities of trips consisting of two direct flights.

```
SELECT   F1.depart, F2.arrive
FROM     Flights as F1, Flights as F2
WHERE    F1.arrive = F2.depart
```

FLIGHTS

depart	arrive
NYC	Reno
NYC	Oakland
Boston	Tampa
Oakland	Boston
Tampa	NYC