Lab 6 - SQL

Big Data Spring 2017

March 6, 2017

Review: Basic SQL Queries

- A basic SQL query has the form

```
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 correlation name)
- qualification: comparisons using defined operators (e.g., >, <, =), which can be combined using AND, OR, and NOT
- DISTINCT: an optional keyword indicating that answer should not contain duplicates

MySQL

- You should have already installed MySQL before coming to class. (Instructions for both Windows and OS X posted on NYU Classes).

Start MySQL Server

- Start MySQL server and login as root

- Type the following command:

```
show databases;
```

- If there is no database named ‘test’, type:

```
create database test;
```

- Now type:

```
use test;
show tables;
```

Our Example Today

- 3 tables: sailors, boats, reserves
- Step 1: Create the tables and populate them.
    - We have given you a script to do this,

      `sailors-mysql.sql`

    - Type

```
source path_to_file/sailors-mysql.sql
```

where `path_to_file` is the path to where you've saved this file on your laptop

Boats, Sailors, Reservations

- This script created three tables: boats, sailors, and reserves. You can see this is you type

```
show tables;
```

- You can see the complete tables if you use the command, e.g.,

```
SELECT * FROM boats;
```

```
create table sailors(
        sid int PRIMARY KEY,
        sname varchar(30),
        rating int,
        age int
);

create table reserves(
        sid int,
        bid int,
        day date,
        PRIMARY KEY (sid, bid, day)
);

create table boats(
        bid int PRIMARY KEY,
        bname char(20),
        color char(10)
);
```

Table 1: boats

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

Table 2: reserves

sid	bid	day
22	101	10-OCT-98
22	102	10-OCT-98
22	103	10-AUG-98
22	104	10-JUL-98
31	102	10-NOV-98
64	101	05-SEP-98
64	102	08-SEP-98
74	103	08-SEP-98
31	103	06-NOV-98
31	104	12-NOV-98

Table 3: sailors

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

Practice SQL Queries

- We have given you a long list of SQL queries to write using this data

- We will work through a few select queries together

Practice Writing SQL Queries

- Find the names and ages of all sailors

Practice Writing SQL Queries

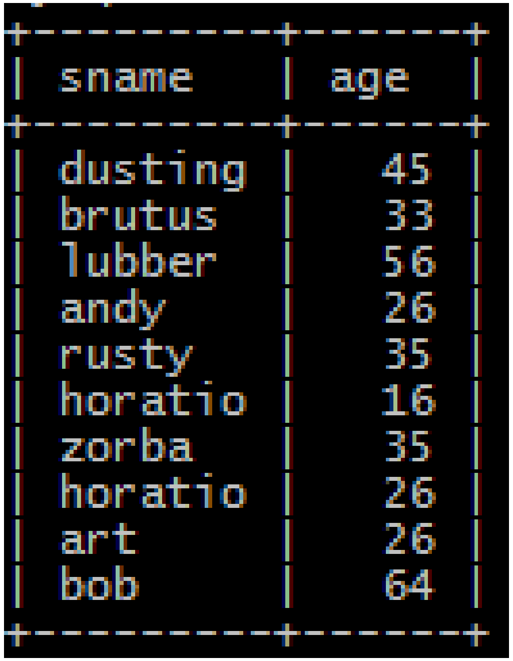
- Find the names and ages of all sailors

```
SELECT sname, age FROM sailors;
```

Practice Writing SQL Queries

- Find the names and ages of all sailors

```
SELECT sname, age FROM sailors;
```



```
+----------+-------+
| sname    | age   |
+----------+-------+
| dusting  |    45 |
| brutus   |    33 |
| lubber   |    56 |
| andy     |    26 |
| rusty    |    35 |
| horatio  |    16 |
| zorba    |    35 |
| horatio  |    26 |
| art      |    26 |
| bob      |    64 |
+----------+-------+
```

Practice Writing SQL Queries

- Find the names and ages of all sailors

```
SELECT sname, age FROM sailors;
```

```
SELECT S.sname, S.age FROM sailors S;
```

Correlation name. Not always necessary, but good practice to use this.

Practice Writing SQL Queries

- Find all sailors with a rating above 7.

Practice Writing SQL Queries

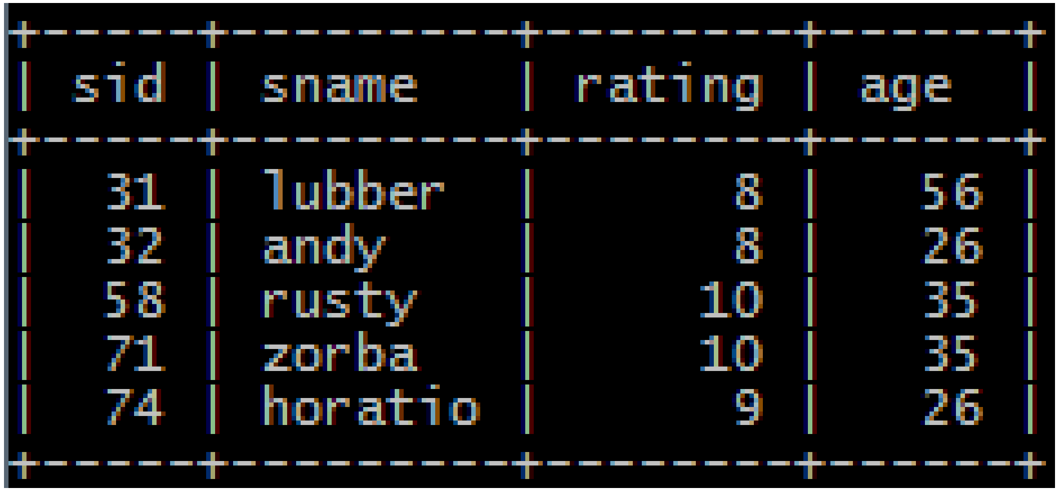
- Find all sailors with a rating above 7.

```
SELECT *
FROM sailors
WHERE rating > 7;
```

Practice Writing SQL Queries

- Find all sailors with a rating above 7.

```
SELECT *
FROM sailors
WHERE rating > 7;
```



```
+-----+---------+--------+-------+
| sid | sname   | rating | age   |
+-----+---------+--------+-------+
|  31 | lubber  |      8 |    56 |
|  32 | andy    |      8 |    26 |
|  58 | rusty   |     10 |    35 |
|  71 | zorba   |     10 |    35 |
|  74 | horatio |      9 |    26 |
+-----+---------+--------+-------+
```

Practice Writing SQL Queries

- Find the names of sailors who have reserved boat number 103

Practice Writing SQL Queries

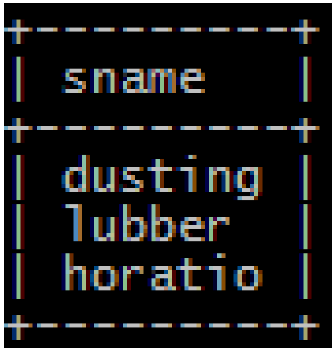
- Find the names of sailors who have reserved boat number 103

```
SELECT sname
FROM sailors S, reserves R
WHERE S.sid = R.sid AND bid = 103;
```

Practice Writing SQL Queries

- Find the names of sailors who have reserved boat number 103

```
SELECT sname
FROM sailors S, reserves R
WHERE S.sid = R.sid AND bid = 103;
```



```
+---------+
| sname   |
+---------+
| dusting |
| lubber  |
| horatio |
+---------+
```

Practice Writing SQL Queries

- Find the names of sailors who have reserved boat number 103

```
SELECT sname
FROM sailors S, reserves R
WHERE S.sid = R.sid AND bid = 103;
```

Using a nested query:

```
SELECT sname
FROM sailors S
WHERE S.sid in (SELECT R.sid
                FROM reserves R
                WHERE R.bid = 103);
```

Practice Writing SQL Queries

- Find the sids of sailors who have reserved a red boat.

Practice Writing SQL Queries

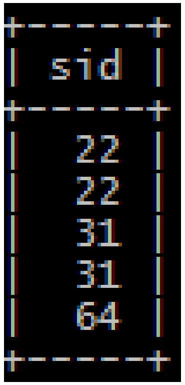
- Find the sids of sailors who have reserved a red boat.

```
SELECT sid
FROM  reserves R, boats B
WHERE R.bid = B.bid AND color = 'red';
```

Practice Writing SQL Queries

- Find the sids of sailors who have reserved a red boat.

```
SELECT sid
FROM  reserves R, boats B
WHERE R.bid = B.bid AND color = 'red';
```

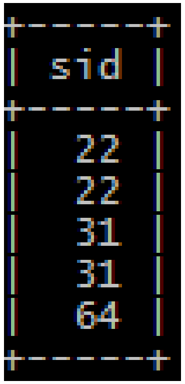


```
+------+
| sid  |
+------+
|   22 |
|   22 |
|   31 |
|   31 |
|   64 |
+------+
```

Practice Writing SQL Queries

- Find the sids of sailors who have reserved a red boat.

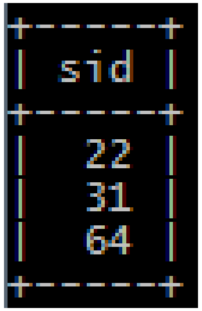
```
SELECT sid
FROM  reserves R, boats B
WHERE R.bid = B.bid AND color = 'red';
```



```
+------+
| sid  |
+------+
|   22 |
|   22 |
|   31 |
|   31 |
|   64 |
+------+
```

This contains duplicates. To remove duplicates, use DISTINCT keyword:

```
SELECT DISTINCT sid
FROM  reserves R, boats B
WHERE R.bid = B.bid AND color = 'red';
```



```
+------+
| sid  |
+------+
|   22 |
|   31 |
|   64 |
+------+
```

Practice Writing SQL Queries

- Find the names of sailors who have reserved both a red and a green boat.

Practice Writing SQL Queries

- Find the names of sailors who have reserved both a red and a green boat.

Here is one incorrect query:

```
SELECT sname
FROM sailors S, reserves R, boats B
WHERE S.sid = R.sid AND R.bid = B.bid AND (color =
'red' AND color = 'green');
```

What happens?

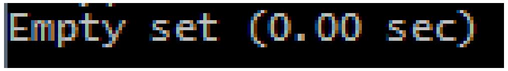
Practice Writing SQL Queries

- Find the names of sailors who have reserved both a red and a green boat.

Here is one incorrect query:

```
SELECT sname
FROM sailors S, reserves R, boats B
WHERE S.sid = R.sid AND R.bid = B.bid AND (color =
'red' AND color = 'green');
```

What happens?



```
Empty set (0.00 sec)
```

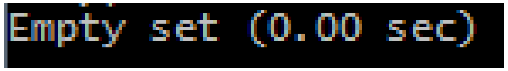
Practice Writing SQL Queries

- Find the names of sailors who have reserved both a red and a green boat.

Here is one incorrect query:

```
SELECT sname
FROM sailors S, reserves R, boats B
WHERE S.sid = R.sid AND R.bid = B.bid AND (color =
'red' AND color = 'green');
```

What happens?



```
Empty set (0.00 sec)
```

Practice Writing SQL Queries

- Find the names of sailors who have reserved both a red and a green boat.

Another mistake: a sailor named Horatio has reserved a red boat, and a *different* sailor named Horatio has reserved a green boat – make sure to write your query such that Horatio is not returned as a sailor that has reserved both a red and green boat!

Practice Writing SQL Queries

- Find the names of sailors who have reserved both a red and a green boat.

Another mistake: a sailor named Horatio has reserved a red boat, and a *different* sailor named Horatio has reserved a green boat – make sure to write your query such that Horatio is not returned as a sailor that has reserved both a red and green boat!

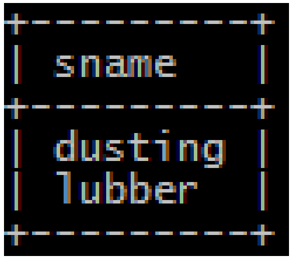
→ Need to use SIDs rather than name since these are the primary key in the sailors table (i.e., they are unique)

Practice Writing SQL Queries

- Find the names of sailors who have reserved both a red and a green boat.

Here is one example of a correct query:

```
SELECT DISTINCT S.sname
FROM sailors S, boats B, reserves R
WHERE S.sid = R.sid AND R.bid = B.bid AND
B.color='red' AND S.sid IN
      (SELECT S2.sid
      FROM sailors S2, boats B2, reserves R2
      WHERE S2.sid=R2.sid AND R2.bid=B2.bid AND
      B2.color='green');
```



```
+-----------+
| sname     |
+-----------+
| dusting   |
| lubber    |
+-----------+
```

Practice Writing SQL Queries

- Find the names of sailors who have not reserved boat number 103.

Practice Writing SQL Queries

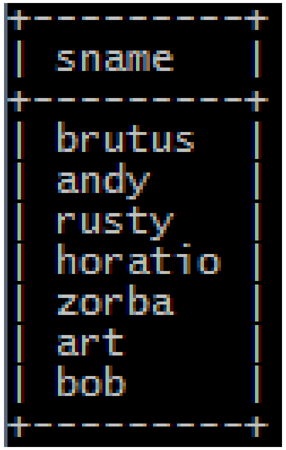
- Find the names of sailors who have not reserved boat number 103.

```
SELECT sname
FROM sailors S
WHERE S.sid NOT IN
      (SELECT sid
       FROM Reserves
      WHERE bid = 103);
```

Practice Writing SQL Queries

- Find the names of sailors who have not reserved boat number 103.

```
SELECT sname
FROM sailors S
WHERE S.sid NOT IN
      (SELECT sid
       FROM Reserves
      WHERE bid = 103);
```



```
+-----------+
| sname     |
+-----------+
| brutus    |
| andy      |
| rusty     |
| horatio   |
| zorba     |
| art       |
| bob       |
+-----------+
```

Practice Writing SQL Queries

- Find the names of sailors whose rating is better than some sailor called Horatio.

Practice Writing SQL Queries

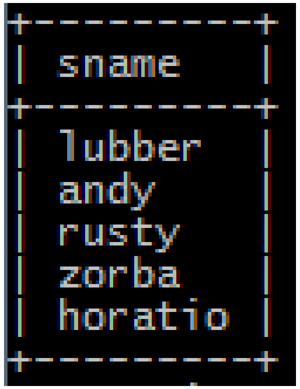
- Find the names of sailors whose rating is better than some sailor called Horatio.

```
SELECT sname
FROM sailors S1
WHERE S1.rating > ANY (SELECT rating
                       FROM sailors S2
                       WHERE sname='horatio');
```

Practice Writing SQL Queries

- Find the names of sailors whose rating is better than some sailor called Horatio.

```
SELECT sname
FROM sailors S1
WHERE S1.rating > ANY (SELECT rating
                       FROM sailors S2
                       WHERE sname='horatio');
```



```
+-----------+
| sname     |
+-----------+
| lubber    |
| andy      |
| rusty     |
| zorba     |
| horatio   |
+-----------+
```

Practice Writing SQL Queries

- Find the average age of sailors with a rating of 10.

Practice Writing SQL Queries

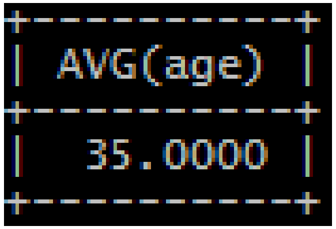
- Find the average age of sailors with a rating of 10.

```
SELECT AVG(age)
FROM Sailors
WHERE rating = 10;
```

Practice Writing SQL Queries

- Find the average age of sailors with a rating of 10.

```
SELECT AVG(age)
FROM Sailors
WHERE rating = 10;
```



```
+----------+
| AVG(age) |
+----------+
|  35.0000 |
+----------+
```

Practice Writing SQL Queries

- Find the name and age of the oldest sailor.

Practice Writing SQL Queries

- Find the name and age of the oldest sailor.

What happens if we write the query:

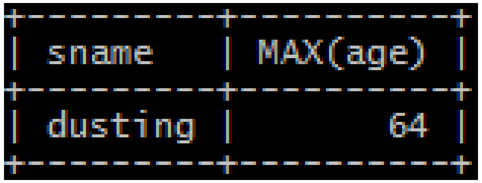
```
SELECT sname,MAX(age)
FROM sailors;
```

Practice Writing SQL Queries

- Find the name and age of the oldest sailor.

What happens if we write the query:

```
SELECT sname,MAX(age)
FROM sailors;
```



```
+----------+-----------+
| sname    | MAX(age)  |
+----------+-----------+
| dusting  |        64 |
+----------+-----------+
```

Table 3: sailors

sid	sname	rating	age
22	dusting	7	45

Practice Writing SQL Queries

- Find the name and age of the oldest sailor.

What happens if we write the query:

```
SELECT sname,MAX(age)
FROM sailors;
```

The answer is incorrect! sname is neither in an aggregate nor in GROUP BY

Practice Writing SQL Queries

- Find the name and age of the oldest sailor.

A correct query:

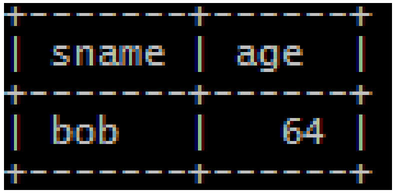
```
SELECT S.sname, S.age
FROM sailors S
WHERE S.age = (SELECT MAX(S2.age) FROM sailors S2);
```

Practice Writing SQL Queries

- Find the name and age of the oldest sailor.

A correct query:

```
SELECT S.sname, S.age
FROM sailors S
WHERE S.age = (SELECT MAX(S2.age) FROM sailors S2);
```



```
+--------+-------+
| sname  | age   |
+--------+-------+
| bob    |    64 |
+--------+-------+
```

Practice Writing SQL Queries

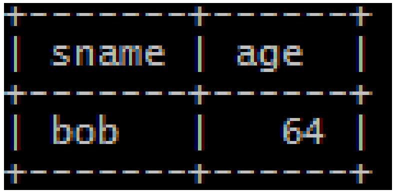
- Find the name and age of the oldest sailor.

A correct query:

```
SELECT S.sname, S.age
FROM sailors S
WHERE S.age = (SELECT MAX(S2.age) FROM sailors S2);
```

Does this match the age that was input by the script?

Can you see why?



```
+--------+-------+
| sname  | age   |
+--------+-------+
| bob    |    64 |
+--------+-------+
```

Practice Writing SQL Queries

- Find the name and age of the oldest sailor.

Table 3: sailors

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

```
create table sailors(
        sid int PRIMARY KEY,
        sname varchar(30),
        rating int,
        age int
);
```

Practice Writing SQL Queries

- Find the name and age of the oldest sailor.

Table 3: sailors

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

```
create table sailors(
        sid int PRIMARY KEY,
        sname varchar(30),
        rating int,
        age int
);
```

Deliverable

(due Wednesday, March 8, 2017, 6pm):

Write SQL queries for the following:

1. Find the names of sailors who do not have any boat reservations
2. Find the sids of all sailors who have reserved a red boat but not a green boat.
3. Find the names of sailors whose rating is better than all sailors called Horatio.

- **You only have to submit the queries you wrote, not the output tables.**
- **Can submit via text box or upload text file.**
- **You are encouraged to work with partners or in small groups**