CCT1343, Week 2

Single-Table SQL

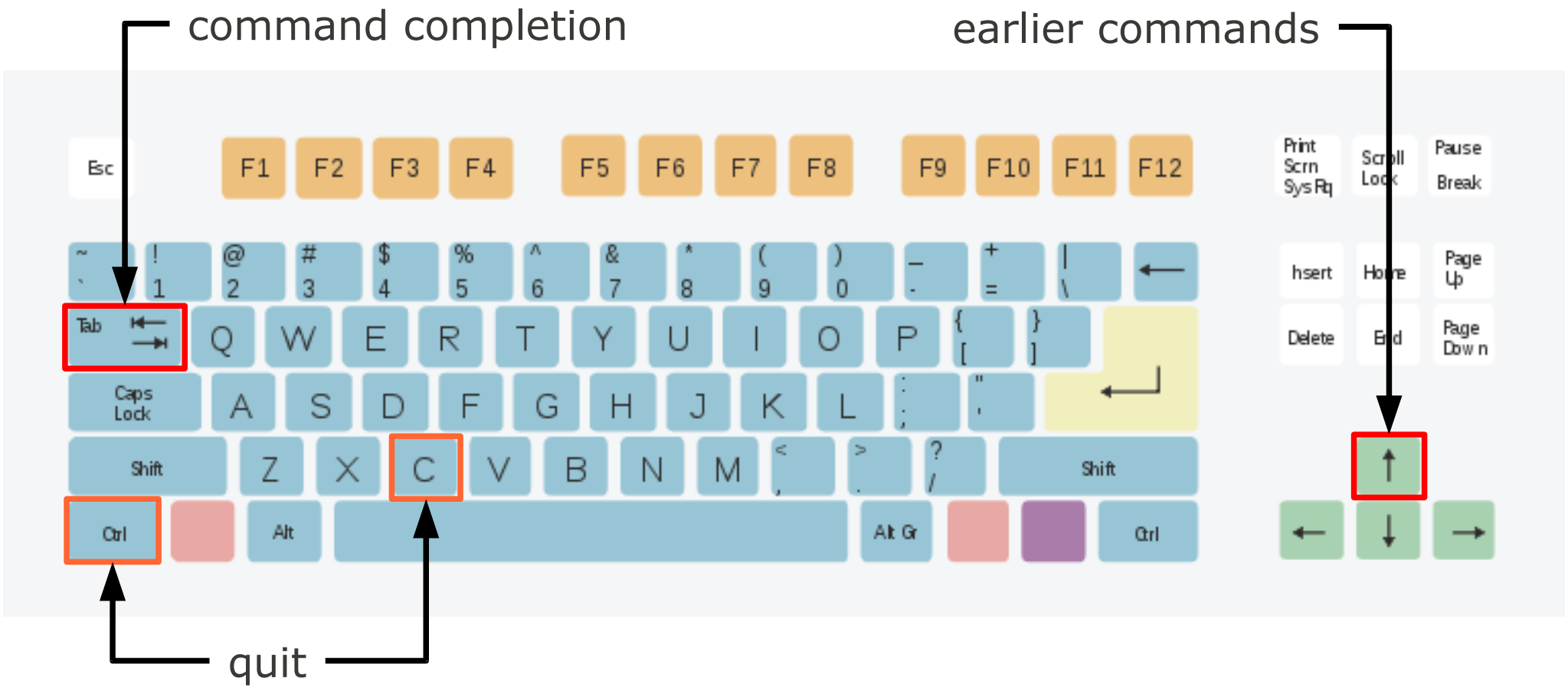
Yuri Takhteyev
University of Toronto
January 10, 2011

Connect to the Server

ssh **kenobio7**@yoda.ischool.utoronto.ca

If you are on Windows and do not have git-bash, install from:
http://code.google.com/p/msysgit/

Important Keys



"Ctrl+C" is usually represented as "^C"

image source:http://upload.wikimedia.org/wikipedia/commons/thumb/3/3a/Qwerty.svg/1000px-Qwerty.svg.png

MySQL Prompt

mysql>

do not confuse with the bash prompt!

Hint: type “exit” or ^C to exit.

What do we enter at the mysql prompt?

Available Databases

```
mysql> show databases;
+--------------------+
| Database           |
+--------------------+
| information_schema |
| child_care         |
| diveshop           |
| kenobio7           |
| menagerie          |
| starwars           |
+--------------------+
6 rows in set (0.00 sec)
```

Selecting a Database

```
mysql> use menagerie;

Database changed
```

Listing Tables

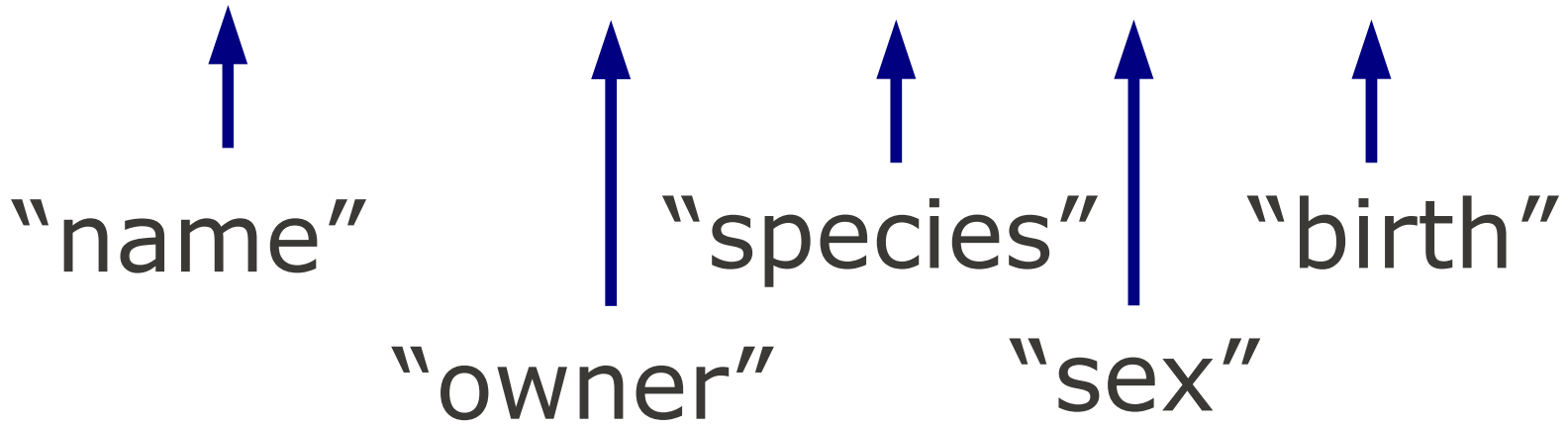
```
mysql> show tables;
+----------------------+
| Tables_in_menagerie |
+----------------------+
| event                |
| owner                |
| pet                  |
| species              |
+----------------------+
4 rows in set (0.00 sec)
```

Describing a Table

```
mysql> describe pet;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| name    | varchar(20) | YES  |     | NULL    |       |
| owner   | varchar(20) | YES  |     | NULL    |       |
| species | varchar(20) | YES  |     | NULL    |       |
| sex     | char(1)     | YES  |     | NULL    |       |
| birth   | date        | YES  |     | NULL    |       |
| death   | date        | YES  |     | NULL    |       |
| weight  | float       | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
7 rows in set (0.00 sec)
```

Relation

(Fluffy, Harold, cat, f, 1993-02-04)
(Buffy, Harold, dog, f, 1989-05-13)
(Chirpy, Gwen, bird, f, 1998-09-11)



A Table

name	owner	species	sex	birth
Fluffy	Harold	cat	f	1993-02-04
Bluffy	Harold	dog	f	1989-05-13
Chirpy	Gwen	bird	f	1998-09-11

Order Doesn't Matter
(in most cases)

birth	owner	name	sex	species
1993-02-04	Harold	Fluffy	f	cat
1989-05-13	Harold	Bluffy	f	dog
1998-09-11	Gwen	Chirpy	f	bird

Projection

name	owner	species	sex	birth
Fluffy	Harold	cat	f	1993-02-04
Bluffy	Harold	dog	f	1989-05-13
Chirpy	Gwen	bird	f	1998-09-11

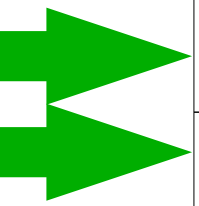
Projection

name	species	sex
Fluffy	cat	f
Bluffy	dog	f
Chirpy	bird	f

$$\pi_{\text{name, species, sex}}(R)$$

Selection

("Restriction" in Harrington)



name	owner	species	sex	birth
Fluffy	Harold	cat	f	1993-02-04
Bluffy	Harold	dog	f	1989-05-13
Chirpy	Gwen	bird	f	1998-09-11

Selection

("Restriction" in Harrington)

name	owner	species	sex	birth
Fluffy	Harold	cat	f	1993-02-04
Bluffy	Harold	dog	f	1989-05-13

$$\sigma_{\text{owner="Harold"}} (R)$$

Columns vs Rows

Projection:

choosing columns (fields)
by name

Selection:

choosing rows with a condition

Basic SELECT

```
   select
3. «list of fields»
1. from «souce table»
2. where «conditions»;
```

selection followed by projection

For Instance:

```
select
name, weight
from pet
where owner="Harold";
```

From table **pet**,
select rows satisfying the condition
owner=”Harold”,
then return fields **name** and **weight**.

Selection in SQL

```
mysql> select name from pet
    -> where owner="Harold";
+--------+--------+
| name   | weight |
+--------+--------+
| Fluffy |    9.5 |
| Buffy  |    2.3 |
+--------+--------+
2 rows in set (0.00 sec)
```

Skipping Projection

```
select
*
from «table»
where «condition»;
```

For instance:

```
select *
from pet
where owner="Harold";
```

```
+--------+--------+---------+------+------------+-------+--------+
| name   | owner  | species | sex  | birth      | death | weight |
+--------+--------+---------+------+------------+-------+--------+
| Fluffy | Harold | cat     | f    | 1993-02-04 | NULL  |    9.5 |
| Buffy  | Harold | dog     | f    | 1989-05-13 | NULL  |    2.3 |
+--------+--------+---------+------+------------+-------+--------+
2 rows in set (0.01 sec)
```

Complex Conditions

```
select «fields» from «table»
where «condition1» and
«condition2»;
```

For instance:

```
select name, weight
from pet
where owner="Harold"
and species="dog";
```

```
+--------+---------+
| name   | weight  |
+--------+---------+
| Buffy  |     2.3 |
+--------+---------+
1 row in set (0.00 sec)
```

Or We Can Use "OR"

```
select «fields» from «table»
where «condition1» or
«condition2»;
```

For instance:

```
select name, owner, species
from pet
where owner="Harold"
or species="dog";
```

```
+--------+--------+---------+
| name   | owner  | species |
+--------+--------+---------+
| Fluffy | Harold | cat     |
| Buffy  | Harold | dog     |
| Fang   | Benny  | dog     |
| Bowser | Diane  | dog     |
+--------+--------+---------+
4 rows in set (0.00 sec)
```

And Why Not "NOT"?

```
select «fields» from «table»
where not «condition»;
```

For instance:

```
select name, owner from pet
where not owner="Harold";
```

```
+----------+-------+
| name     | owner |
+----------+-------+
| Claws    | Gwen  |
| Fang     | Benny |
| Bowser   | Diane |
| Chirpy   | Gwen  |
| Whistler | Gwen  |
| Slim     | Benny |
| Puffball | Diane |
+----------+-------+
7 rows in set (0.00 sec)
```

Combinations

«condition1» and
not («condition2» or
«condition3»)

For instance:

```
select name from pet
where species="dog" and
not (owner="Harold" or
owner="Gwen");
```

```
+---------+
| name    |
+---------+
| Fang    |
| Bowser  |
+---------+
2 rows in set (0.00 sec)
```

Fields and Values

«field» = «value»

«value» = «value»

«field» = «field»

For instance:

```
select name, owner
from pet
where name=owner;
```

```
+--------+--------+
| name   | owner  |
+--------+--------+
| Margie | Margie |
+--------+--------+
1 row in set (0.00 sec)
```

An Odd Case

"Harold"="Harold"

What will we get back?

```
select name, owner
from pet
where "Harold"="Harold";
```

```
+----------+--------+
| name     | owner  |
+----------+--------+
| Fluffy   | Harold |
| Claws    | Gwen   |
| Buffy    | Harold |
| Fang     | Benny  |
| Bowser   | Diane  |
| Chirpy   | Gwen   |
| Whistler | Gwen   |
| Slim     | Benny  |
| Puffball | Diane  |
| Margie   | Margie |
+----------+--------+
10 rows in set (0.00 sec)
```

Another Odd Case

"Harold"="Gwen"

What will we get back?

```
select name, owner
from pet
where "Harold"="Gwen";
```

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

Functions

«value» = «function»(«value»)

For instance:

```
select name, owner
from pet
where upper(name)="buffy";
```

```
+-------+---------+
| name  | owner   |
+-------+---------+
| Buffy | Harold |
+-------+---------+
1 row in set (0.00 sec)
```

Tests: Equality

```
where «value»=«value»;
```

For instance:

```
select name from pet
where owner="Harold";
```

Tests: Inequality

```
where «value»!=«value»;
```

For instance:

```
select name, owner from pet
where owner!="Harold";
```

```
+----------+-------+
| name     | owner |
+----------+-------+
| Claws    | Gwen  |
| Fang     | Benny |
| Bowser   | Diane |
| Chirpy   | Gwen  |
| Whistler | Gwen  |
| Slim     | Benny |
| Puffball | Diane |
+----------+-------+
7 rows in set (0.00 sec)
```

Tests: LIKE

```
where «field» like «pattern»;
```

For instance:

```
select name from pet
where name like "%uff%";
```

```
+-----------+
| name      |
+-----------+
| Fluffy    |
| Buffy     |
| Puffball  |
+-----------+
3 rows in set (0.00 sec)
```

More: >, <, <=, >=

where «value»<«value»;

For instance:

```
select name, birth from pet
where birth<"1980-01-01";
```

```
+--------+------------+
| name   | birth      |
+--------+------------+
| Bowser | 1979-08-31 |
+--------+------------+
1 row in set (0.00 sec)
```

Expressions

«expression» > «expression»

For instance:

```
select name from pet
where weight>birth_weight*50;
```

```
+---------+
| name    |
+---------+
| Fluffy |
| Fang    |
| Bowser |
+---------+
3 rows in set (0.01 sec)
```

Combinations

For instance:

```
select name from pet
where species="dog" and
not (owner="Harold" or
owner="Gwen")
and  weight>birth_weight*50;
```

Combinations

For instance:

```
select name from pet
where
  species="dog"
  and
     not (owner="Harold"
           or owner="Harold")
  and  weight>birth_weight*50;
```

```
+---------+
| name    |
+---------+
| Fang    |
| Bowser  |
+---------+
2 rows in set (0.00 sec)
```

Complex Projection

For instance:

```
select
 upper(name),
 weight/1000
from pet
where species="dog";
```

Complex Projection

For instance:

```
select
 upper(name),
 round(weight/1000,3)
from pet
where species="dog";
```

```
+--------------+------------------------+
| upper(name)  | round(weight/1000,3)   |
+--------------+------------------------+
| BUFFY        |                  0.002 |
| FANG         |                  0.013 |
| BOWSER       |                  0.037 |
+--------------+------------------------+
3 rows in set (0.00 sec)
```

The 6th Column

```
mysql> describe pet;
+---------+-------------+------+-----+---------+-------+
| Field   | Type        | Null | Key | Default | Extra |
+---------+-------------+------+-----+---------+-------+
| name    | varchar(20) | YES  |     | NULL    |       |
| owner   | varchar(20) | YES  |     | NULL    |       |
| species | varchar(20) | YES  |     | NULL    |       |
| sex     | char(1)     | YES  |     | NULL    |       |
| birth   | date        | YES  |     | NULL    |       |
| death   | date        | YES  |     | NULL    |       |
| weight  | float       | YES  |     | NULL    |       |
+---------+-------------+------+-----+---------+-------+
7 rows in set (0.00 sec)
```

What's in it?

```
mysql> select name, death
    -> from pet;
+----------+------------+
| name     | death      |
+----------+------------+
| Fluffy   | NULL       |
| Claws    | NULL       |
| Buffy    | NULL       |
| Fang     | NULL       |
| Bowser   | 1995-07-29 |
| Chirpy   | NULL       |
| Whistler | NULL       |
| Slim     | NULL       |
| Puffball | NULL       |
+----------+------------+
9 rows in set (0.00 sec)
```

NULL = no value provided (unknown, does not apply, etc.)

IS NULL

```
where «field» is null;
```

For instance:

```
select name, death from pet
where death is null;
```

```
+----------+--------+
| name     | death  |
+----------+--------+
| Fluffy   | NULL   |
| Claws    | NULL   |
| Buffy    | NULL   |
| Fang     | NULL   |
| Chirpy   | NULL   |
| Whistler | NULL   |
| Slim     | NULL   |
| Puffball | NULL   |
+----------+--------+
8 rows in set (0.00 sec)
```

“IS NULL” vs “= NULL”

`death="NULL"`

true if the value of death is equal to the four-letter sting “NULL”

`death=NULL`

always evaluates to NULL (the condition fails)

`death is NULL`

evaluates to TRUE is death is null and to FALSE if death is not null

IS NOT NULL

```
where «field» is not null;
```

For instance:

```
select name, death from pet
where death is not null;
```

```
+---------+-------------+
| name    | death       |
+---------+-------------+
| Bowser | 1995-07-29 |
+---------+-------------+
1 row in set (0.00 sec)
```

Null with And and Or

Null and True → Null
Null and False → False
Null or True → True
Null or False → Null

Duplicates

For instance:

```
select species
from pet
where owner="Gwen";
```

```
+----------+
| species  |
+----------+
| cat      |
| bird     |
| bird     |
+----------+
3 rows in set (0.00 sec)
```

Removing Duplicates

```
select distinct «fields» ...
```

For instance:

```
select distinct species
from pet where owner="Gwen";
```

```
+---------+
| species |
+---------+
| cat     |
| bird    |
+---------+
2 rows in set (0.00 sec)
```

vs

```
+---------+
| species |
+---------+
| cat     |
| bird    |
| bird    |
+---------+
3 rows in set (0.00 sec)
```

More SELECT

```
    select
4.  «list of fields»
1.  from «souce table»
2.  where «conditions»
3.  order by «field»
5.  limit «field»;
```

Sorting the Results

```
select ... from ... where ...
order by «expression»;
```

For instance:

```
select name, birth from pet
order by birth;
```

```
+----------+------------+
| name     | birth      |
+----------+------------+
| Bowser   | 1979-08-31 |
| Buffy    | 1989-05-13 |
| Fang     | 1990-08-27 |
| Fluffy   | 1993-02-04 |
| Claws    | 1994-03-17 |
| Slim     | 1996-04-29 |
| Whistler | 1997-12-09 |
| Chirpy   | 1998-09-11 |
| Puffball | 1999-03-30 |
+----------+------------+
9 rows in set (0.01 sec)
```

Descending Order

```
select ... from ... where ...
order by «expression» desc;
```

For instance:

```
select name, birth from pet
order by birth desc;
```

LIMIT

```
select ... from ... limit «N»;
```

For instance:

```
select name, birth from pet
order by birth limit 5;
```

```
+---------+------------+
| name    | birth      |
+---------+------------+
| Bowser | 1979-08-31 |
| Buffy  | 1989-05-13 |
| Fang   | 1990-08-27 |
| Fluffy | 1993-02-04 |
| Claws  | 1994-03-17 |
+---------+------------+
5 rows in set (0.00 sec)
```

Aggregation

getting back less then one row per matched record

A “Normal” Function

```
select upper(name) from pet
where birth<"1994-12-31";
```

```
+--------------+
| upper(name)  |
+--------------+
| FLUFFY       |
| CLAWS        |
| BUFFY        |
| FANG         |
| BOWSER       |
+--------------+
5 rows in set (0.00 sec)
```

Aggregating: COUNT

```
select count(<fields>)
from ...;
```

For instance:

```
select count(name) from pet
where birth<"1994-12-31";
```

```
+--------------+
| count(name)  |
+--------------+
|            5 |
+--------------+
1 row in set (0.00 sec)
```

Counting Distinct

```
select count(distinct owner)
from pet
where birth<"1994-12-31";
```

```
+-------------------------+
| count(distinct owner) |
+-------------------------+
|                       4 |
+-------------------------+
1 row in set (0.00 sec)
```

Summation

```
select sum(weight) from pet
where species="cat";
```

```
+------------------+
| sum(weight)      |
+------------------+
| 13.130000114441 |
+------------------+
1 row in set (0.00 sec)
```

More Aggregation

AVG, MIN, MAX

For instance:

```
select min(weight) from pet;
```

```
+--------------------+
| min(weight)        |
+--------------------+
| 0.032000001519918 |
+--------------------+
1 row in set (0.00 sec)
```

Grouping

```
... group by <fields>;
```

For instance:

```
select species, sum(weight)
from pet group by species;
```

```
+---------+--------------------+
| species | sum(weight)        |
+---------+--------------------+
| bird    | 0.453000005334616 |
| cat     |  13.1300001144409 |
| dog     |  52.3299984931946 |
| hamster | 0.127000004053116 |
| snake   | 0.209999993443489 |
+---------+--------------------+
5 rows in set (0.02 sec)
```

HAVING

```
select species, sum(weight)
from pet group by species
having sum(weight)<1;
```

Similar to “where” conditions, but uses aggregate functions (or the grouping fields).

```
+---------+--------------------+
| species | sum(weight)        |
+---------+--------------------+
| bird    | 0.453000005334616 |
| hamster | 0.127000004053116 |
| snake   | 0.209999993443489 |
+---------+--------------------+
3 rows in set (0.00 sec)
```

HAVING

```
group by species...
...having sum(weight)>10      +
...having avg(weight)>10      +
...having weight<1            -
...having species="dog"       +
...having name="Buffy"        -
...having count(name)>1       +
...having count(species)>1    +
```

HAVING

```
select species, sum(weight)
from pet group by species
having sum(weight)<1;
```

Similar to “where” conditions, but uses aggregate functions (or the grouping fields).

The Order

```
   select
6. «list of fields»
1. from «souce table»
2. where «conditions»
3. group by «field»
4. having «conditions»
5. order by «field»
7. limit «field»;
```

The Order

```
   select
6. weight
1. from pet
2. where weight>1
3. group by species
4. having count(name)>1
5. order by sum(weight)
7. limit 1;
```

```
+-------------+
| birth       |
+-------------+
| 1989-05-13 |
+-------------+
1 row in set (0.00 sec)
```

Questions?

Data Modification

INSERT

add a new record

UPDATE

modify existing record

UPDATE

```
update pet
set weigh=100
where name="Buffy";
```

INSERT

```
insert into pet
values ("Margie", "Margie",
"pony", "f", "2001-04-19",
NULL, 72);
```

here order matters!

Assignment 1

1. Will be posted later today
2. To be done individually*
3. Provide explanations for queries
4. Build queries in steps
5. Plan for trial and error