

INF1343, Winter 2012

# Data Modeling and Database Design

Yuri Takhteyev

Faculty of Information  
University of Toronto



This presentation is licensed under Creative Commons Attribution License, v. 3.0. To view a copy of this license, visit <http://creativecommons.org/licenses/by/3.0/>. This presentation incorporates images from the Crystal Clear icon collection by Everaldo Coelho, available under LGPL from <http://everaldo.com/crystal/>.

**Week 2**

**One Table SQL**

# Answers to the Quiz

```
select id, name
from spare_parts
where number_in_stock<=0;

select number_in_stock
from spare_parts
where
    name="Type 2 Holoprojector";
```

# Relational Algebra

persona

<b>name</b>	<b>occupation</b>	<b>species</b>
Obi-Wan Kenobi	Jedi Master	Human
Yoda	Jedi Master	NULL
Jabba	crime lord	Hutt
Chewbacca	co-pilot	Wookiee
Luke Skywalker	Jedi Knight	Human
Padmé Amidala	queen	Human

# Projection

persona

<b>name</b>	<b>occupation</b>	<b>species</b>
Obi-Wan Kenobi	Jedi Master	Human
Yoda	Jedi Master	NULL
Jabba	crime lord	Hutt
Chewbacca	co-pilot	Wookiee
Luke Skywalker	Jedi Knight	Human
Padmé Amidala	queen	Human



# Projection

persona

name	species
Obi-Wan Kenobi	Human
Yoda	NULL
Jabba	Hutt
Chewbacca	Wookiee
Luke Skywalker	Human
Padmé Amidala	Human



# Selection

persona

<b>name</b>	<b>occupation</b>	<b>species</b>
Obi-Wan Kenobi	Jedi Master	Human
Yoda	Jedi Master	NULL
Jabba	crime lord	Hutt
Chewbacca	co-pilot	Wookiee
Luke Skywalker	Jedi Knight	Human
Padmé Amidala	queen	Human

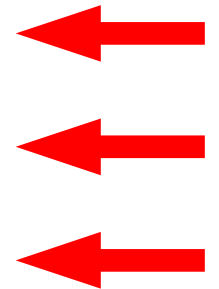


(AKA "restriction".)

# Selection

persona

<b>name</b>	<b>occupation</b>	<b>species</b>
Obi-Wan Kenobi	Jedi Master	Human
Luke Skywalker	Jedi Knight	Human
Padmé Amidala	queen	Human



(AKA "restriction".)



# Columns vs Rows

## **Projection:**

choosing columns (fields)  
by name

## **Selection:**

choosing rows with a condition

# Back to SQL

**use starwars;** **projection**

**select** **name, occupation**

**from** **persona**

**where** **species="Wookiee";**

**selection**

**selection followed by projection**

# Skipping Projection

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

---

```
select *  
from persona  
where species="Human";
```

# Skipping Projection

```
select
```

```
*           to be substituted
```

```
from «table»
```

```
where «condition»;
```

```
the new parts
```

general  
pattern

---

```
select *
```

```
from persona
```

```
where species="Human";
```

specific  
example

# Skipping Selection

```
select
```

```
*
```

```
from «table»;
```

---

```
select *
```

```
from persona;
```

# LIMIT

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

---

```
select name from persona  
limit 5;
```

# Sorting the Results

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

---

```
select name from persona  
order by size;
```

# Descending Sort

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

---

```
select name from persona  
order by size desc;
```



# Putting It Together

Who are the 3 tallest human characters (species="Human")?

---

```
select name, size
from persona
where species="Human"
order by size desc
limit 3;
```

# Putting It Together

Who are the 3 **shortest** human characters?

---

```
select name, size
from persona
where species="Human"
order by size desc
limit 3;
```

# Putting It Together

What is the species of the shortest character?

---

```
select species
from persona
order by size
limit 1;
```

# Putting It Together

What which Jedi Master character comes first alphabetically?

---

```
select name
from persona
where occupation="Jedi Master"
order by name
limit 1;
```

# Moving On

What is the name of the world that Humans are originally from?

---

Hmm. Where is this information?

# Browsing

```
show tables;  
describe «table»;
```

---

```
describe species;
```

# Complex Conditions

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

---

```
use menagerie;  
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»;
```

---

```
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»;
```

---

```
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»)
```

---

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

```
+-----+
```

```
| name |
```

```
+-----+
```

```
| Fang |
```

```
| Bowser |
```

```
+-----+
```

```
2 rows in set (0.00 sec)
```

# Things to Compare

```
«field» = «constant»  
where name="Harald";
```

vs.

```
«field1» = «field2»  
where name=owner;
```

# Combinations

**where** «field1» = «field2»

---

```
select name 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»)

---

```
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»;
```

---

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



# Tests: Inequality

```
where <<value>> != <<value>>;
```

---

```
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»;
```

---

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

```
+-----+
| name   |
+-----+
| Fluffy |
| Buffy  |
| Puffball |
+-----+
```

3 rows in set (0.00 sec)

More: >, <, <=, >=

where `<<value>><<value>>;`

---

```
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»

---

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

```
+-----+
| name   |
+-----+
| Fluffy |
| Fang   |
| Bowser |
+-----+
```

3 rows in set (0.01 sec)



# Combinations

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

# Combinations

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

```
+-----+
```

```
| name |
```

```
+-----+
```

```
| Fang |
```

```
| Bowser |
```

```
+-----+
```

```
2 rows in set (0.00 sec)
```

# Complex Projection

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

# Complex Projection

```
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 6<sup>th</sup> 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;
```

---

```
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 string "NULL"

**death=NULL**

always evaluates to NULL (the condition fails)

**death is NULL**

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

# IS NOT NULL

where `«field» is not null;`

---

```
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

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

```
+-----+
| species |
+-----+
| cat     |
| bird    |
| bird    |
+-----+
```

3 rows in set (0.00 sec)



# Removing Duplicates

```
select distinct «fields» ...
```

---

```
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)
```

# Aggregation

getting back less than 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 ...;
```

---

```
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";
```





# 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**

---

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

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

# Data Modification

## **INSERT**

add a new record

## **UPDATE**

modify existing record

## **DELETE**

remove a record

# INSERT

```
insert into my_pet values  
( "Lilly", "Margie", "pony", "f",  
  "2001-04-19", NULL, 72, 2);
```

here order matters!

# INSERT

```
insert into my_pet values
( "Frank", "Margie", "dog", "m",
  "2011-09-21", NULL, 6, 2),
( "Zé", "Margie", "fish", NULL,
  "2011-12-17", NULL, .1, 2);
```



# INSERT

```
insert into my_pet  
  (name, owner)  
values (  
  "Minnie", "Margie"  
);
```

# INSERT with SELECT

```
insert into my_pet  
select * from pet;
```

# UPDATE

```
update my_pet  
set weight=100  
where name="Buffy";
```

# DELETE

```
delete from my_pet  
where name="Buffy";
```

# DELETE

```
delete from my_pet;
```

# CREATE TABLE

```
use okenobi;  
create table «name» (  
    «field» «type»  
);
```

# Domains / Data Types

011010010111100000111111000001101000110111001010110010010000  
010000011101101010010001100100111100100000010000001011100110  
000111000000001101000111101011100111110001110100110011010101  
00111111011010011101100101100011011110100001111011101111001  
10010000010110011100110001100011111011011000011000110011011  
00111011101110110011101000110001001111100110000110010011011  
00001100100001011001111111011011000010010101110101000110101  
11011010110010010000011000010011100100011011110011011000011  
001000000000000010110110011011111000001011011111000110100100  
11100110010110110010100111111111001010000001111101101000010  
0011011101101000001011110011101100100111111111100001111111  
00001001100111111101100110011010101101111111101111001010101  
10101111111001111101111010101001111111111011100111100010110  
10010001101010011101001011100111011001111010101000110010100  
10111100100010000000001001011001101000110000101111100100010  
00110001000000000000101000011111011011010011010001110011100  
01100110100111100001100000100011111011111011010111100100011  
00001001000001001001001110111111001110001000000000100000010  
00111011101101100010001111010011111011001000010100100010100

# Domains / Data Types

011010010111100000111111000001101000110111001010110010010000  
010000011101101010010001100100111100100000010000001011100110  
000111000000001101000111101011100111110001110100110011010101  
001111110110100111011001011000110111101000001111011101111001  
100100000101100111001100011000111110110110000011000110011011  
001110111011101100111010001100010011111001100000110010011011  
0000110010000010110011111110110110000010010101110101000110101  
11011010110010010000011000010011100100011011110011011000011  
001000000000000001011101100110111110000001011011111000110100100  
111001100101101100101001111111110010100000001111101101000010  
001101110110100000010111100111011001001111111111000001111111  
00001001100111111101100110011010101101111111101111001010101  
10101111111001111101111010101001111111111011100111100010110  
100100011010100111010010111001110110011110101010000110010100  
1011110010001000000000010010110011010001100000101111100100010  
0011000100000000000000101000011111011011010011010001110011100  
0110011010011110000011000000100011111011111011010111100100011  
000010010000001001001001110111111001110001000000000100000010  
0011101110110110000100011110100111110110010000010100100010100



# Domains / Data Types

11011010110010010000011000010011100100011011110011011000011  
0010000000000000010110110011011111000001011011111000110100100

“Harold\n”?

1300457425722198016?

130045742.5722198016?

0.5237304801548 \* 10<sup>-36</sup>?

# String (Text)

CHAR (n)

**VARCHAR (n)**

TEXT

BINARY (n)

VARBINARY (n)

BLOB

ENUM

# Numeric

**INTEGER**

UNSIGNED

SMALLINT

BIGINT

NUMERIC (n, m)

**DECIMAL (n, m)**

**BOOLEAN**

REAL

**DOUBLE PRECISION**

FLOAT (m, d)

# Date and Time

**DATE**

TIME

**DATETIME**

TIMESTAMP

# Extensions

**Spatial:**

POINT

POLYGON

LINESTRING

GEOMETRY

# Choosing a Type

**table student**

name

student\_no

date\_of\_birth

no\_of\_credits

gpa

amount\_owed

# CREATE TABLE

```
use okenobi;  
create table student (  
    name varchar(100)  
);  
describe student;
```

# DROP TABLE

```
drop table student;  
describe student;
```



# CREATE TABLE

```
create table student (  
  name varchar(100) ,  
  student_no integer ,  
  date_of_birth date ,  
  no_of_credits integer ,  
  gpa decimal(3,2) ,  
  amount_owed decimal(10,2)  
);
```

# An Exercise

1. Create a table to store info about films: title, year, country, director, duration.
2. Insert several movies.
3. Find the duration of the most recent movie.