What Kind of Relationships?

- EmailAddresses
- PhoneNumbers
- Degrees
- Hobbies
- Favorites
- Pets
- FamilyMembers

Reconstructing Our Data

Part One: Joins

So Far… Atomize

- **Break information up into tiny bits**
- **Break information out into multiple tables**
  - Create relations using PKs and FKs
    - 1:1
    - 1:M
    - M:M
- **Define some constraints**
  - Field types
  - Field sizes
  - Required
  - More to come

Reconstructing Atomized Data

- Before we go too far…
- Assurance that you'll recover
- Then, back to atomizing!

Query the Parent

```
SELECT PersonID, FirstName, LastName
FROM <ParentTable>
WHERE FirstName = 'Bob'
```

- Returns the unique "PersonID" values

Anatomy of a Query

Get these fields:	**SELECT PersonID, FirstName, LastName**	3
From this table:	**FROM Persons**	1
Only these records:	**WHERE FirstName = 'Bob'**	2

Query the Children of One Parent

Using the "PersonID" value to find Bob's pets

```
SELECT PetID, PetName
FROM Pets
WHERE PersonID = <PersonID>
```

1:1 = one child record returned

1:M = potentially many child records returned

Query Parents with Children

```
SELECT Persons.FirstName,
Persons.LastName
FROM Persons

…
```

Query Parents with Children

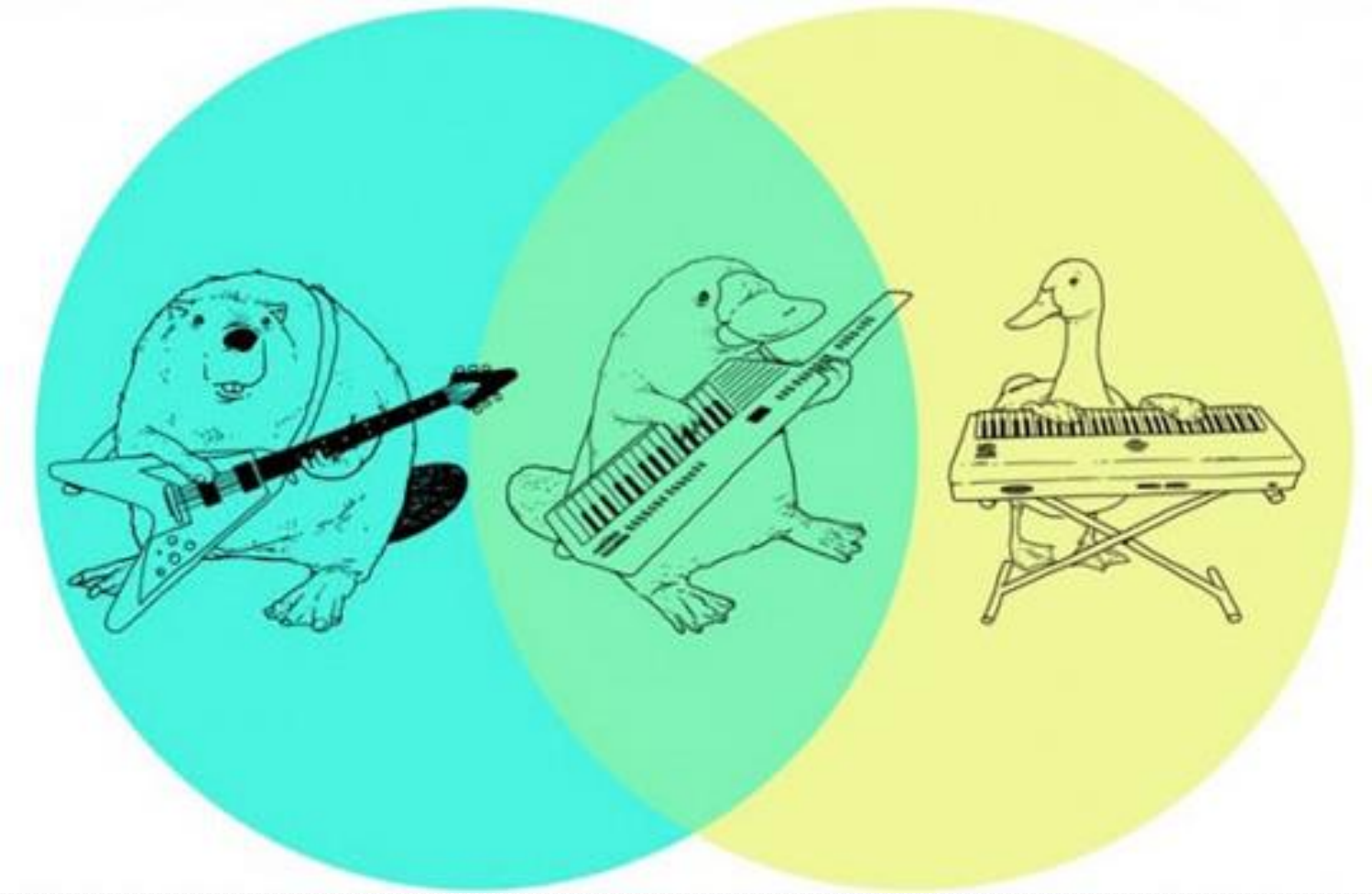
```
SELECT Persons.FirstName,
Persons.LastName, Pets.PetName
FROM Persons
INNER JOIN Pets
ON Persons.PersonID=Pets.PersonID;
```

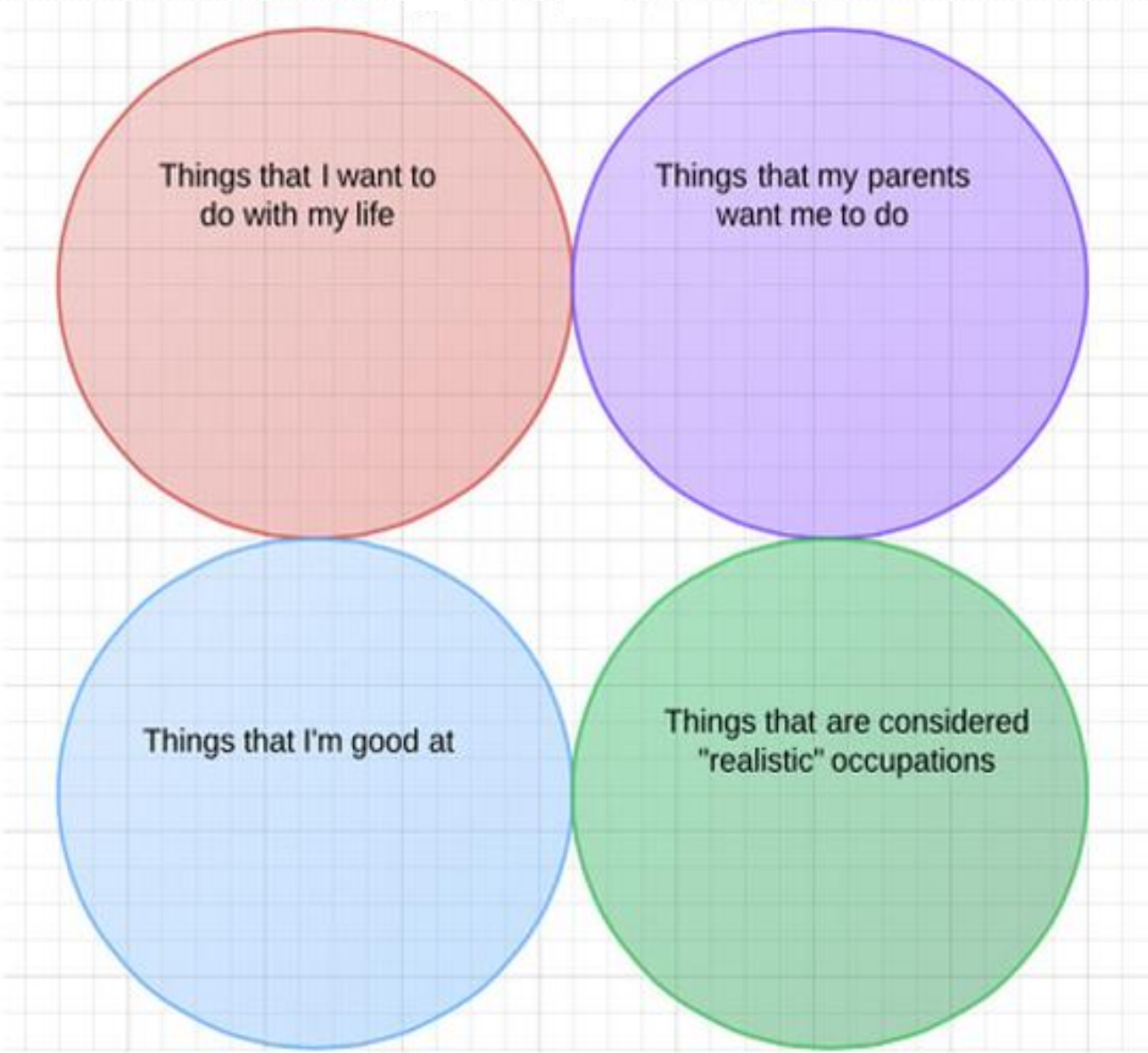
- Must have a match in both tables

JOIN… ON

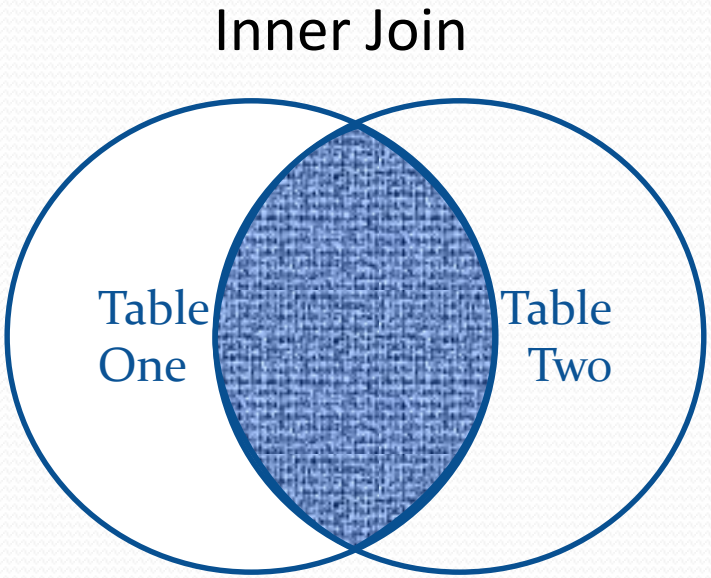
```
SELECT whatever
FROM Table1
INNER JOIN Table2
ON Table1.PK = Table2.FK;
```

The Venn Diagram




Things that I want to do with my life
Things that my parents want me to do
Things that I'm good at
Things that are considered "realistic" occupations

Inner Joins



Selects all records that have matches in both tables.

Query Parents with Children

```
SELECT Persons.FirstName,
Persons.LastName, Pets.PetName
FROM Persons
INNER JOIN Pets
ON Persons.PersonID=Pets.PersonID;
```

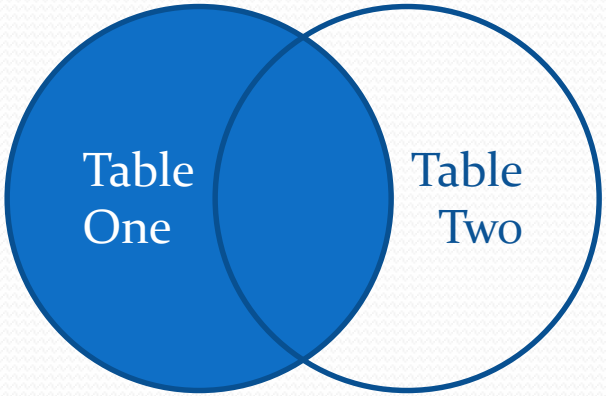
- Must have a match in both tables

Anatomy of a Query Join

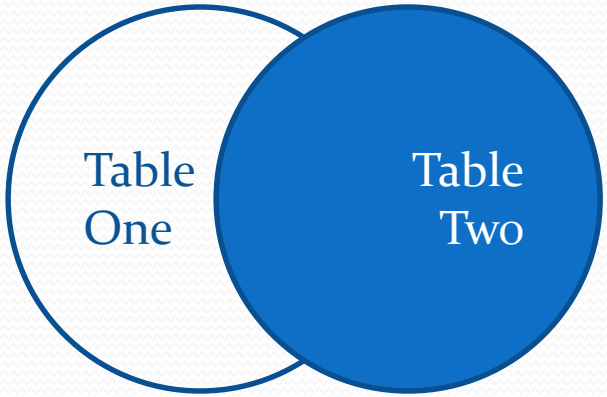
Get these fields:	**SELECT PersonID, FirstName, LastName, PetID, Petname** **SELECT Persons.PersonID, Persons.FirstName, Persons.LastName, Pets.PetID, Pets.Petname**
From this table:	**FROM Persons** **INNER JOIN Pets** **ON Persons.PersonID=Pets.PersonID** *Virtual Table*
Only these records:	**WHERE FirstName = 'Bob'**

Outer Joins

Left Outer Join



Right Outer Join



Selects all records from the indicated table and only the matches from the other.

Query All Parents, Whether or Not They Have Matching Children

```
SELECT Persons.FirstName,
Persons.LastName, Pets.PetName
FROM Persons
LEFT OUTER JOIN Pets
ON Persons.PersonID=Pets.PersonID;
```

- "LEFT JOIN" and "LEFT OUTER JOIN" mean the same thing in MySQL

Query All Children, Whether or Not They Have Matching Parents

```
SELECT Persons.FirstName,
Persons.LastName, Pets.PetName
FROM Persons
RIGHT OUTER JOIN Pets
ON Persons.PersonID=Pets.PersonID;
```

- "RIGHT JOIN" and "RIGHT OUTER JOIN" mean the same thing in MySQL

Concatenation

Drop Kick The Cat

- Gluing the bits back together
- CONCAT

Concatenation

- Gluing the bits back together

```
SELECT CONCAT ('Dearest ', Title,' ', FirstName,
' ', LastName) FullName FROM Persons;
```

Concatenation

- Gluing the bits back together

```
SELECT CONCAT ('Dearest ', Title,' ', FirstName,
' ', LastName) FullName FROM Persons;
```

FullName = Dearest Mr Helmuth Ranklesbone