Relational Data Processing

Everything Data

CompSci 290.01 Spring 2014

DUKE
COMPUTER SCIENCE

Announcements (Thu. Jan. 23)

- Two UTAs on board!
  - **Dan Deng**
  - **Kathleen Miyoko Oshima**
- **Homework #3** will be posted by tonight
  - Due midnight Monday
- We are working on a cloud-based solution for those with slow VMs—stay tuned

Structure is good

- More structure ➠
  easier, more powerful analysis
- What's your favorite structure?

Allows sorting, filtering, grouping, counting, summing, …

Image credit: http://www.zazzle.co.uk/funny+spreadsheet+tshirts

But remember this exercise?

Find the longest-serving current members of the Congress

- A member may serve multiple terms
- So data has a ***nested*** structure

lore@datacourse: ~

File Edit Tabs Help

```
  name:
    first: Maria
    last: Cantwell
    official_full: Maria Cantwell
  bio:
    birthday: '1958-10-13'
    gender: F
    religion: Roman Catholic
  terms:
  - type: rep
    start: '1993-01-05'
    end: '1994-12-01'
    state: WA
    district: 1
    party: Democrat
  - type: sen
    start: '2001-01-03'
    end: '2006-12-09'
    state: WA
    class: 1
    party: Democrat
    url: http://cantwell.senate.gov
  - type: sen
    start: '2007-01-04'
    end: '2013-01-03'
    state: WA
    class: 1
    party: Democrat
    url: http://cantwell.senate.gov
    address: 311 HART SENATE OFFICE BUILDING WASHINGTON DC 20510
    phone: 202-224-3441
    fax: 202-228-0514
    contact_form: http://www.cantwell.senate.gov/contact/
    office: 311 Hart Senate Office Building
  - type: sen
    start: '2013-01-03'
    end: '2019-01-03'
    state: WA
    party: Democrat
/opt/datacourse/data/govtrack/legislators-current.yaml lines 114-152/37539
```

Spreadsheet chokes…

Show as: **rows** records Show: 5 **10** 25 50 rows Sort « first ‹ previous **1 - 10** next › last »

rson - id	roles	person - link	person - osid	person - name	party
412600	{ "meta": { "limit": 100, "offset": 0, "total_count": 1 }, "objects": [{ "congress_numbers": [113], "current": true, "description": "Representative for Massachusetts's 5th congressional district", "district": 5, "enddate": "2015-01-03", "id": 43246, "leadership_title": null, "party": "Democrat", "person": { "bioguideid": "C001101", "birthday": "1963-07-17", "cspanid": null, "firstname": "Katherine", "gender": "female", "gender_label": "Female", "id": 412600, "lastname": "Clark", "link": "https://www.govtrack.us/congress/members/katherine_clark/412600", "middlename": "M.", "name": "Rep. Katherine Clark [D-MA5]", "namemod": "", "nickname": "", "osid": "N00035278", "pvsid": "35858", "sortname": "Clark, Katherine (Rep.) [D-MA5]", "twitterid": null, "youtubeid": null }, "phone": null, "role_type": "representative", "role_type_label": "Representative", "senator_class": null, "senator_rank": null, "startdate": "2013-12-12", "state": "MA", "title": "Rep.", "title_long": "Representative", "website": "http://katherineclark.house.gov" }] }	https://www.govtrack.us/congress/members/katherine_clark/412600	N00035278	Rep. Katherine Clark [D-MA5]	D
400170	{ "meta": { "limit": 100, "offset": 0, "total_count": 11 }, "objects": [{ edit "congress_numbers": [103], "current": false, "description": "Representative for Florida's 23rd congressional district", "district": 23, "enddate": "1994-12-01", "id": 1377, "leadership_title": null, "party": "Democrat", "person": { "bioguideid": "H000324", "birthday": "1936-09-05", "cspanid": 1858, "firstname": "Alcee", "gender": "male", "gender_label": "Male", "id": 400170, "lastname": "Hastings", "link": "https://www.govtrack.us/congress/members/alcee_hastings/400170", "middlename": "L.", "name": "Rep. Alcee Hastings [D-FL20]", "namemod": "", "nickname": "", "osid": "N00002884", "pvsid": "26798", "sortname": "Hastings, Alcee (Rep.) [D-FL20]", "twitterid": null, "youtubeid": "RepAlceeHastings" }, "phone": null, "role_type": "representative", "role_type_label": "Representative", "senator_class": null, "senator_rank": null, "startdate": "1993-01-05", "state": "FL", "title": "Rep.", "title_long": "Representative", "website": "" }, { "congress_numbers": [104], "current": false, "description": "Representative for Florida's 23rd congressional district", "district": 23, "enddate": "1996-10-04", "id": 1378, "leadership_title": null, "party": "Democrat", "person": { "bioguideid": "H000324", "birthday": "1936-09-05", "cspanid": 1858, "firstname": "Alcee", "gender": "male", "gender_label": "Male", "id": 400170, "lastname": "Hastings", "link": "https://www.govtrack.us/congress/members/alcee_hastings/400170", "middlename": "L.", "name": "Rep. Alcee Hastings [D-FL20]", "namemod": "", "nickname": "", "osid": "N00002884", "pvsid": "26798", "sortname": "Hastings, Alcee (Rep.) [D-FL20]", "twitterid": null, "youtubeid": "RepAlceeHastings" }, "phone": null, "role_type": "representative", "role_type_label": "Representative", "senator_class": null, "senator_rank": null, "startdate": "1995-01-04", "state": "FL", "title": "Rep.", "title_long": "Representative", "website": "" }, { "congress_numbers": [105], "current": false, "description": "Representative for Florida's 23rd congressional district", "district": 23, "enddate": "1998-12-19", "id": 1379, "leadership_title": null, "party": "Democrat", "person": { "bioguideid": "H000324", "birthday": "1936-09-05", "cspanid": 1858, "firstname": "Alcee", "gender": "male", "gender_label": "Male", "id": 400170, "lastname": "Hastings", "link":	https://www.govtrack.us/congress/members/alcee_hastings/400170	N00002884	Rep. Alcee Hastings [D-FL20]	D

… on this and other more complex structures

How do we structure data now?

One table ➛ multiple tables

Image credit: http://www.zazzle.co.uk/funny+spreadsheet+tshirts

Persons & their roles

persons
One row per person

person_roles
One row for each term served by a person

← Birthday

← Gender

Start/end of a term →

Party →

State →

House/senate? →

← Person ID →

← Name

Relational data model

How is data structured/constrained?

- Organize data in tables (AKA *relations*)
  - Each table has a list of (typed) columns
    - Data is stored as rows
    - Each row has a value for every column
- Declare structure + constraints as ***schema***

How is data queried/updated?

- A "declarative" language called ***SQL***
  - Say ***what*** result you want, ***not how*** to compute it

persons schema

String of length 10

Key constraint: no two rows in this table can have the same key value

The database will use the PRIMARY key to to identify rows

```
CREATE TABLE persons (
  id CHAR(10) NOT NULL PRIMARY KEY,
  id_govtrack INTEGER NOT NULL UNIQUE,
  id_lis CHAR(4) UNIQUE,
  first_name VARCHAR(50) NOT NULL,
  middle_name VARCHAR(50),
  last_name VARCHAR(50) NOT NULL,
  birthday DATE,
  gender CHAR(1)
   CHECK (gender IS NULL OR gender IN ('F', 'M'))
);
```

Cannot be NULL, a special value used to indicate missing or inapplicable values

Specifies additional constraints on the column value

person_roles schema

Foreign-key constraint: every person_id value must be some id value found in persons; i.e., no "dangling" references

```
CREATE TABLE person_roles (
  person_id CHAR(10) NOT NULL REFERENCES persons(id),
  type CHAR(3) NOT NULL CHECK (type IN ('rep', 'sen')),
  start_date DATE NOT NULL,
  end_date DATE NOT NULL,
  state CHAR(2) NOT NULL REFERENCES states(id),
  district INTEGER
    CHECK ((type = 'rep' AND district IS NOT NULL) OR
                (type = 'sen' AND district IS NULL)),
  party VARCHAR(20)
);
```

Example data

persons

id	id_govtrack	id_lis	first_name	middle_name	last_name	birthday	gender
A000369	412500		Mark	E.	Amodei	1958-06-12	M
D000622	412533		Tammy		Duckworth	1968-03-12	F
...	...	...	...	...	...	...	...

person_roles

person_id	type	start_date	end_date	state	district	party
A000369	rep	2013-01-03	2015-01-03	NV	2	Republican
A000369	rep	2011-09-13	2013-01-03	NV	2	Republican
D000622	rep	2013-01-03	2015-01-03	IL	8	Democrat
...	...	...	...	...	...	...

For simplicity…

Assume we have this "table":

- Columns in the primary key are underlined

cur_members(<u>id</u>, first_name, last_name,
gender, birthday,
type, party, state)

We will come back to how to create this "table" later

Show me a table… sorted

- List all current members of the Congress

```
SELECT *
FROM cur_members;
```

A shorthand for "all columns"

Marks the end of the query

A table

- *Sorting* options
  - … **ORDER BY** birthday; (Default is ascending order)
  - … ORDER BY birthday DESC;
  - … ORDER BY type, party;

cur_members(id, first_name, last_name, gender, birthday, type, party, state)

Picking columns

- AKA *projection*

A special value that always holds the current date

```
SELECT id, first_name, last_name, state, type,
  (date_part('year', current_date) –
   date_part('year', birthday)) AS age
FROM cur_members
ORDER BY age;
```

Extract the year part of a date

You can compute a new column to output

… and give it a name

cur_members(id, first_name, last_name, gender, birthday, type, party, state)

Picking rows

- AKA *filtering* or *selection*

```
SELECT *
FROM cur_members
WHERE type = 'sen' AND birthday >= '1950-01-01';
```

Comparison, not assignment

Strings are enclosed by single quotes

... AND ...
... OR ...
NOT (...)

cur_members(id, first_name, last_name, gender, birthday, type, party, state)

Grouping and *aggregating* rows

```
SELECT party, COUNT(*)
FROM cur_members
GROUP BY party;
```

Put members of the same party in one group

Count the size of each group

cur_members(id, first_name, last_name, gender, birthday, type, party, state)

More *grouping*/*aggregation*

```
SELECT party, gender, COUNT(*),
   AVG(date_part('year', current_date) –
         date_part('year', birthday))
FROM cur_members
GROUP BY party, gender;
```

Again, one output row per group

Rows now must match on *both* columns to be in the same group

Other aggregation functions include **SUM**, **MAX**, **MIN**

cur_members(id, first_name, last_name, gender, birthday, type, party, state)

Joining tables

- How did we get cur_members(id, first_name, last_name, gender, birthday, type, party, state)?
- Need to "join" tables together

Only in persons

Only in person_roles

persons(id, id_govtrack, id_lis, first_name, last_name, birthday, gender)
person_roles(person_id, type, start_date, end_date, state, district, party)

A SQL Query walks into a bar. In the corner of the bar are two tables. The Query walks up to the tables and asks,

"Mind if I join you?"

Image credit: http://www.swcasinoproductions.com/images/party-rentals/lighted-tables-buffets-furniture/party-rentals-lighted-bar-acrylic.jpg

Join = pairing "related" rows

persons

id	first_name	last_name	...
A000369	Mark	Amodei	...
D000622	Tammy	Duckworth	...
...	...	...	...

person_roles

person_id	...	start_date	end_date	...
A000369	...	2013-01-03	2015-01-03	...
A000369	...	2011-09-13	2013-01-03	...
D000622	...	2013-01-03	2015-01-03	
...	...	...	...	...

"Join condition" is persons.id = person_roles.person_id

Output table:

id	first_name	last_name	...	person_id	...	start_date	end_date	...
A000369	Mark	Amodei	...	A000369	...	2013-01-03	2015-01-03	...
A000369	Mark	Amodei	...	A000369	...	2011-09-13	2013-01-03	...
D000622	Tammy	Duckworth	...	D000622	...	2013-01-03	2015-01-03	
...	...	...	...	...	...	...	...	...

cur_members in SQL

Want to "save" your output for later querying?
Create a *view*—a "virtual" table

```
CREATE VIEW cur_members AS
SELECT p.id, p.first_name, p.last_name, p.gender, p.birthday,
       r.type, r.party, r.state
FROM persons p, person_roles r
WHERE p.id = r.person_id
  AND r.end_date >= '2014-01-01';
```

List tables to be joined

Join condition

Selection condition local to r

Make p an alias for persons—think of it
as a variable iterating through persons rows

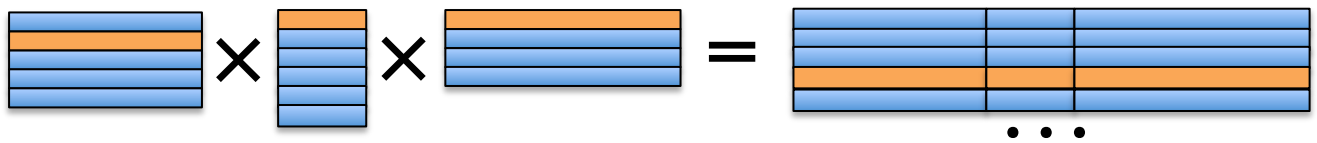
Putting it together

SELECT *columns or expressions*

4. Compute one output row for each "wide row"
(or for each group of them if query has grouping/aggregation)

FROM *tables*

1. Generate all combinations of rows, one from each table; each combination forms a "wide row"

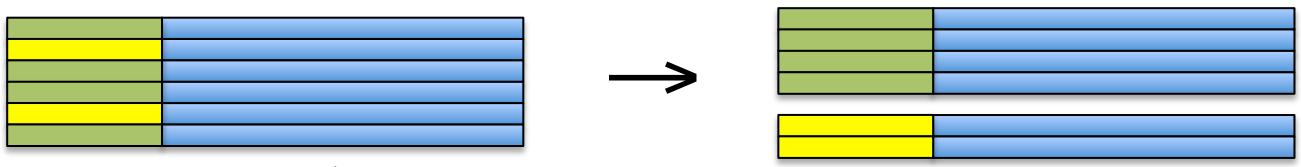


WHERE *conditions*

2. Filter—keep only "wide rows" satisfying *conditions*

GROUP BY *columns*

3. Group—"wide rows" with matching values for *columns* go into the same group



ORDER BY *output columns*;

5. Sort the output rows

Subqueries and **LIMIT**

- Find the ten longest serving members

```
SELECT id, first_name, last_name, birthday,
        (SELECT SUM(end_date - start_date)
         FROM person_roles r
         WHERE r.person_id = p.id) AS duration
FROM persons p
ORDER BY duration DESC LIMIT 10;
```

Pretend that for every p in persons we examine, we run the subquery with p's id value

Just give me the first 10 rows

One more example

Rep. (& Prof.) David Price
(D, NC 4th District)

How does he vote?

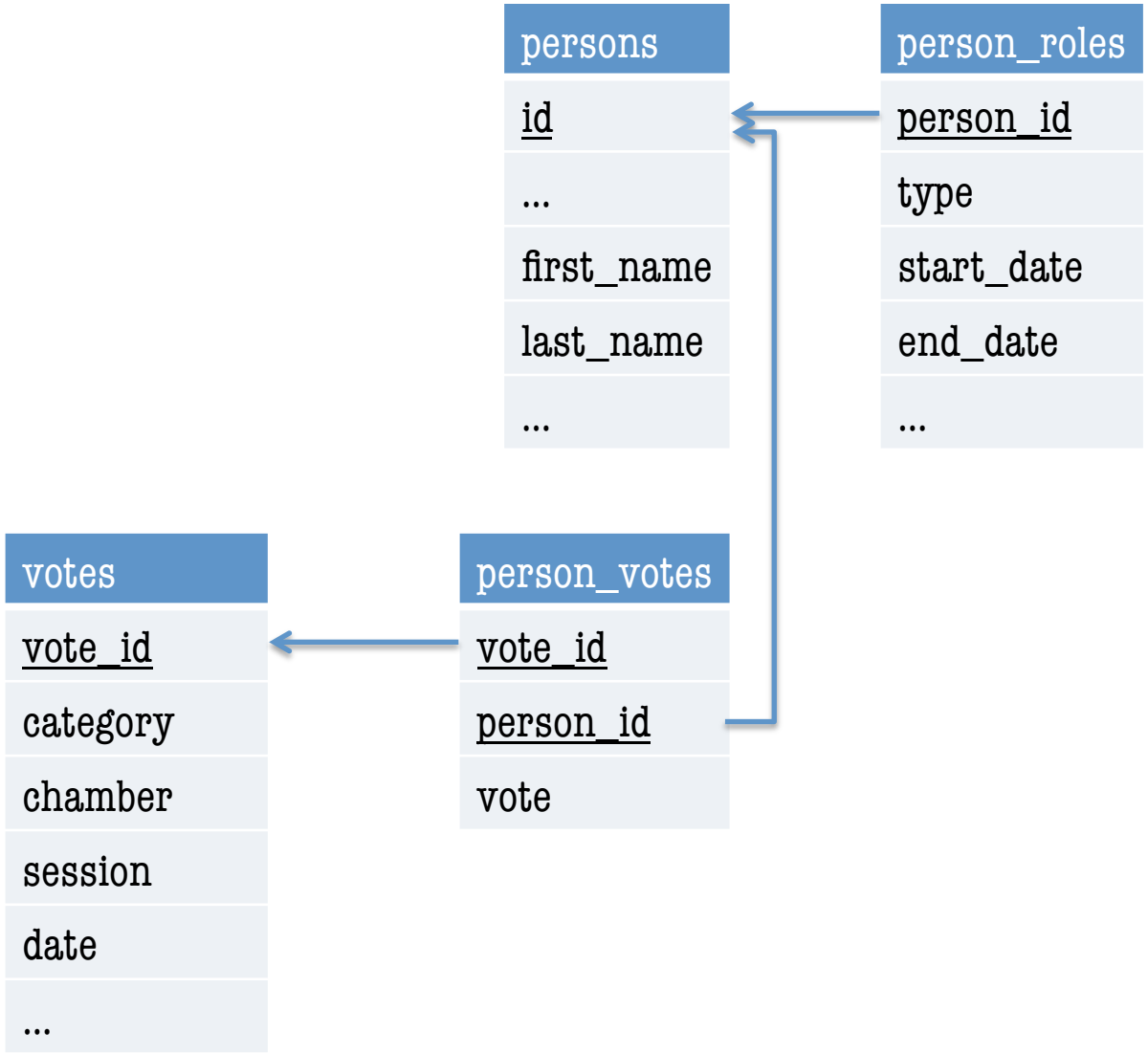
Say, comparing with

- Pelosi (D, CA), minority leader

- Cantor (R, VA), majority leader

Image credits:
http://en.wikipedia.org/wiki/File:David_Price,_official_Congressional_photo_portrait.JPG
http://en.wikipedia.org/wiki/File:Nancy_Pelosi_2013.jpg
http://en.wikipedia.org/wiki/File:Eric_Cantor,_official_113th_Congress_photo_portrait.jpg

Expanded schema



Here we go…

WITH creates a temporary view for the query that follows

```
WITH votes_compare(vote_id, vote1, vote2) AS
(SELECT v1.vote_id, v1.vote, v2.vote
 FROM votes v, persons p1, persons p2, person_votes v1, person_votes v2
 WHERE v.chamber = 'h' AND v.session = 2013
     AND p1.last_name = 'Price' AND p2.last_name = 'Pelosi'
     AND v1.person_id = p1.id AND v2.person_id = p2.id
     AND v1.vote_id = v2.vote_id AND v.id = v1.vote_id)
```

Or try 'Cantor'

```
SELECT COUNT(*) AS agree,
  (SELECT COUNT(*) FROM votes_compare) AS total,
   COUNT(*)*100.00 / (SELECT COUNT(*) FROM votes_compare)
     AS percent
FROM votes_compare
WHERE vote1 = vote2;
```

… and the results are:

vs. Pelosi (D, CA), minority leader

```
 agree | total |        percent
-------+-------+---------------------
   717 |  1282 | 55.9282371294851794
```

What's going on? Isn't Price a Democrat?

vs. Cantor (R, VA), majority leader

```
 agree | total |        percent
-------+-------+---------------------
   768 |  1282 | 59.9063962558502340
```

It's your job to clear Prof. Price's name!

Dear chain-mailer: This is an exercise teaching students how to catch subtle errors in SQL queries—the numbers here are ***WRONG*** and just don't grab them mindlessly