Aggregation

Announcements

Grouping Rows

```
Rows in a table can be grouped, and aggregation is performed on each group
```

```
SELECT [columns] FROM [table] GROUP BY [expression] HAVING [expression];
```

One output row for each unique value of **expression**

Only keep groups for which **expression** is true

```
[expression] AS [name], [expression] AS [name], ...
```

```
SELECT category, COUNT(*) AS total
FROM principals GROUP BY category;
```

2 rows in the output: actor director

category	total
actor	2
director	1

principals

tconst	ordering	nconst	category	character
tt0012349	2	nm0701012	actor	The Woman
tt0012349	13	nm0000122	director	\N
tt0017136	1	nm0375609	actor	Maria

(Demo)

Grouping Rows

Rows in a table can be grouped, and aggregation is performed on each group

```
SELECT [columns] FROM [table] GROUP BY [expression] HAVING [expression];
```

One output row for each unique value of **expression**

Only keep groups for which **expression** is true

```
[expression] AS [name], [expression] AS [name], ...
```

```
SELECT category, COUNT(*) AS total
FROM principals GROUP BY category;
```

category	total
actor	2
director	1

An aggregate function in the [columns] clause computes a value from a group of rows (or all rows, if there are no groups):

- **COUNT**(*): number of rows in a group
- **MAX**([expression]): largest value of [expression] for any row in a group (also **MIN**, **SUM**, & **AVG**)

(Demo)

Writing Select Statements

```
Describe the output table:

1) Determine which existing rows are needed to express the result (FROM & WHERE)

2) Form groups and determine which groups should appear as output rows (GROUP BY & HAVING)

3) Format the output rows (SELECT)
```

```
SELECT: Values each output row contains (and column labels)
   FROM: Source of input rows
   WHERE: Which input rows
   GROUP BY: Form output rows
   HAVING: Which output rows
```

Grouping Rows

```
SELECT [columns] FROM [table] GROUP BY [expression] HAVING [expression];
```

One output row for each unique value of **expression**

Only keep groups for which **expression** is true

- COUNT(*): number of rows in a group
- MAX([expression]): largest value of [expression] for any row in a group (also MIN, SUM, & AVG)

principals

tconst	ordering	nconst	category	character
tt0012349	2	nm0701012	actor	The Woman
tt0012349	13	nm0000122	director	\N
tt0017136	1	nm0375609	actor	Maria

Select the nconst and the total number of characters for each actor who had more than 15 characters played

```
SELECT nconst, COUNT(*)          SELECT: Values each output row contains (and column labels)
    FROM principals              FROM: Source of input rows
    WHERE category="actor"       WHERE: Which input rows
    GROUP BY nconst              GROUP BY: Form output rows
    HAVING COUNT(*) > 15;        HAVING: Which output rows
```

pollev.com/cs61a

Grouping Rows: Remakes

```
SELECT [columns] FROM [table] GROUP BY [expression] HAVING [expression];
```

- **COUNT**(*): number of rows in a group
- **MAX**([expression]): largest value of [expression] for any row in a group (also **MIN**, **SUM**, & **AVG**)

titles

tconst	title	year	runtime	genres
tt8404614	The Two Popes	2019	125	Biography,Drama
tt0012349	The Kid	1921	68	Comedy,Drama,Family

Create a table of remakes that have the same title

```
title                            first  second
-------------------------------  -----  ------
How to Train Your Dragon         2010   2025
The Girl with the Dragon Tattoo  2009   2011
```

```
SELECT title, MIN(year) AS old, MAX(year) AS new
    FROM titles
    GROUP BY title
    HAVING COUNT(*) > 1;
```

pollev.com/cs61a

SELECT: Values each output row contains (and column labels)

FROM: Source of input rows

GROUP BY: Form output rows

HAVING: Which output rows

Ratings for Each Actor

titles

tconst	title	year	runtime	genres
tt8404614	The Two Popes	2019	125	Biography,Drama

ratings

tconst	averageRating	numVotes
tt8613070	8.0	123438

principals

tconst	ordering	nconst	category	character
tt0012349	2	nm0701012	actor	The Woman

names

nconst	name	birth	death	profession	knownforTitles
nm0000002	Lauren Bacall	1924	2014	actress,miscellaneous,soundtrack	tt0037382,tt0075213,tt0038355,tt0117057

```
Select each actor, rating pair:

SELECT names.name, ratings.averageRating
FROM ratings JOIN names JOIN principals
ON ratings.tconst=principals.tconst AND names.nconst=principals.nconst
GROUP BY names.nconst
ORDER BY averageRating DESC LIMIT 10;
```

Ratings for Each Actor

titles

tconst	title	year	runtime	genres
tt8404614	The Two Popes	2019	125	Biography,Drama

ratings

tconst	averageRating	numVotes
tt8613070	8.0	123438

principals

tconst	ordering	nconst	category	character
tt0012349	2	nm0701012	actor	The Woman

Why store the data like this?
pollev.com/cs61a

names

nconst	name	birth	death	profession	knownforTitles
nm0000002	Lauren Bacall	1924	2014	actress,miscellaneous,soundtrack	tt0037382,tt0075213,tt0038355,tt0117057

~~Select each actor, rating pair:~~ Select each actor and their average rating:

```
SELECT names.name, SUM(ratings.averageRating * ratings.numVotes) / SUM(ratings.numVotes)
FROM ratings JOIN names JOIN principals
ON ratings.tconst=principals.tconst AND names.nconst=principals.nconst
GROUP BY names.nconst
ORDER BY averageRating DESC LIMIT 10;
```

What should we change?
pollev.com/cs61a

Group By Practice

Spring 2023 CS 61A Final Question 7

The finals table has columns hall (strings) and course (strings), and has rows for each lecture hall in which a course is holding its final exam.

The sizes table has columns room (strings) and seats (numbers), and has one row per unique room on campus containing the number of seats in that room. All lecture halls are rooms.

Create a table with two columns, course (string) and seats (number), and with one row containing the **name of the course** and the **total number of seats in final rooms** for that course. Only include a row **for each course that uses at least two rooms for its final.**

finals:

hall	course
RSF	61A
Wheeler	61A
RSF	61B

sizes:

room	seats
RSF	900
Wheeler	700
310 Soda	40

result:

course	seats
61A	1600

```
SELECT course, SUM(seats) AS seats
  FROM finals, sizes
  WHERE hall=room
  GROUP BY course
  HAVING COUNT(*) > 1 ;
```

pollev.com/cs61a

Recap

```
SELECT [columns] FROM [table] GROUP BY [expression] HAVING [expression];
```

One output row for each unique value of **expression**

Only keep groups for which **expression** is true

```
[expression] AS [name], [expression] AS [name], ...
```

An aggregate function computes a value from a group of rows:

- **COUNT**(*): number of rows in a group
- **MAX**([expression]): largest value of [expression] for any row in a group (also **MIN**, **SUM**, & **AVG**)

principals

tconst	ordering	nconst	category	character
tt0012349	2	nm0701012	actor	The Woman
tt0012349	13	nm0000122	director	\N
tt0017136	1	nm0375609	actor	Maria

count of rows for each category:

```
SELECT category, COUNT(*)
FROM principals GROUP BY category;
```