



# Announcements

- HW 1 due today @11pm
- HW 2 out today, due 1/22 @11pm
- No lecture Monday (holiday)

# Recap: Grouping

## GROUP BY:

Group rows based on matching attribute values

## HAVING:

Eliminate groups based on aggregate information

# Recap: Grouping

```
SELECT Product, SUM(quantity)
FROM Purchases
GROUP BY Product
HAVING SUM(quantity) > 20
```

Product	Price	Quantity	Month
Bagel	3	20	Jan
Bagel	1.50	20	Feb
Banana	0.5	50	Feb
Banana	5	10	March
Apple	4	10	March

# Recap: Grouping

```
SELECT Product, SUM(quantity)
FROM Purchases
GROUP BY Product
HAVING SUM(quantity) > 20
```

Product	Price	Quantity	Month
Bagel	3	20	Jan
Bagel	1.50	20	Feb
Banana	0.5	50	Feb
Banana	5	10	March
Apple	4	10	March



Product	SUM(quantity)
Bagel	40
Banana	60

# Recap: Grouping

```
SELECT Job, MAX(Salary)
FROM Payroll
GROUP BY Job
HAVING MIN(Salary) > 80000
```

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

Job	MAX(Salary)
Prof	100000

# Aggregation RA

How is aggregation processed internally?

```
SELECT Job, MAX(Salary)
FROM Payroll
GROUP BY Job
HAVING MIN(Salary) > 80000
```

# Aggregation RA

How is aggregation processed internally?

```
SELECT Job, MAX(Salary)
FROM Payroll
GROUP BY Job
HAVING MIN(Salary) > 80000
```

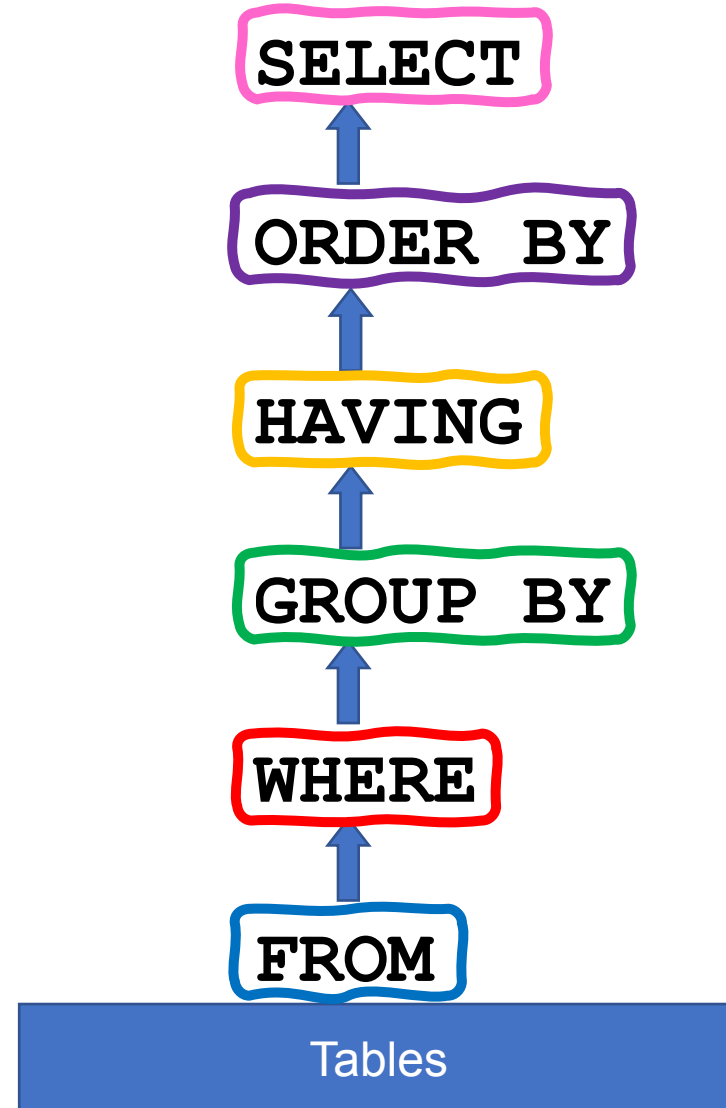
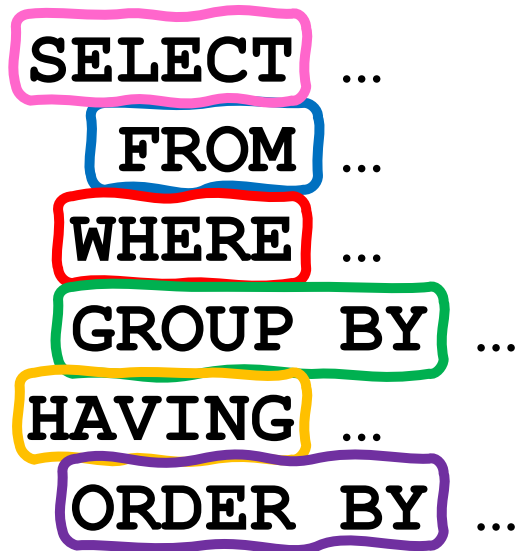
Our first preview of Relational Algebra:

“Having” applies **after** grouping the big intermediate table



# SQL and RA Vocab Summary

## FWGHOS™



# Aggregation RA

```
SELECT Job, MAX(Salary)
FROM Payroll
GROUP BY Job
HAVING MIN(Salary) > 80000
```

UserID	Name	Job	Salary
...	...	...	...

# Aggregation RA

```
SELECT Job, MAX(Salary)
FROM Payroll
GROUP BY Job
HAVING MIN(Salary) > 80000
```

Aggregate  $Job, MAX(P.Salary) \rightarrow maxSal, MIN(P.Salary) \rightarrow minSal$

UserID	Name	Job	Salary
...	...	...	...

# Aggregation RA

```
SELECT Job, MAX(Salary)
  FROM Payroll
 GROUP BY Job
HAVING MIN(Salary) > 80000
```

Job	maxSal	minSal
TA	60000	50000
Prof	100000	90000

Aggregate  $Job, MAX(P.Salary) \rightarrow maxSal, MIN(P.Salary) \rightarrow minSal$

UserID	Name	Job	Salary
...	...	...	...

# Aggregation RA

```
SELECT Job, MAX(Salary)
  FROM Payroll
 GROUP BY Job
HAVING MIN(Salary) > 80000
```

Job	maxSal	minSal
Prof	100000	90000

Having  $minSal > 80000$

Job	maxSal	minSal
TA	60000	50000
Prof	100000	90000

Aggregate  $Job, MAX(P.Salary) \rightarrow maxSal, MIN(P.Salary) \rightarrow minSal$

UserID	Name	Job	Salary
...	...	...	...

# Aggregation RA

```
SELECT Job, MAX(Salary)
  FROM Payroll
 GROUP BY Job
HAVING MIN(Salary) > 80000
```

Select<sub>*Job, maxSal*</sub>

Job	maxSal	minSal
Prof	100000	90000

Having<sub>*minSal > 80000*</sub>

Job	maxSal	minSal
TA	60000	50000
Prof	100000	90000

Aggregate<sub>*Job, MAX(P.Salary) → maxSal, MIN(P.Salary) → minSal*</sub>

UserID	Name	Job	Salary
...	...	...	...

# Aggregation RA

```
SELECT Job, MAX(Salary)
  FROM Payroll
 GROUP BY Job
HAVING MIN(Salary) > 80000
```

Job	maxSal
Prof	100000

Select<sub>Job, maxSal</sub>

Job	maxSal	minSal
Prof	100000	90000

Having<sub>minSal > 80000</sub>

Job	maxSal	minSal
TA	60000	50000
Prof	100000	90000

Aggregate<sub>Job, MAX(P.Salary)→maxSal, MIN(P.Salary)→minSal</sub>

UserID	Name	Job	Salary
...	...	...	...

# Semantics

First evaluate the FROM clause

Next evaluate the WHERE clause

Group the attributes in the GROUPBY

Eliminate groups based on HAVING

Sort the results based on ORDER BY

Last evaluate the SELECT clause

**FWGHOS™**



# General Form of Group By

```
SELECT S
FROM R1, ..., Rn
WHERE C1
GROUP BY a1, ..., ak
HAVING C2
```

S = any attributes  $a_1, \dots, a_k$  and/or any aggregates, but no other attributes

C1 = any condition on the attributes in  $R_1, \dots, R_n$

C2 = any condition on the aggregate expressions and attributes  $a_1, \dots, a_k$

# General Form of Group By

```
SELECT S
FROM R1, ..., Rn
WHERE C1
GROUP BY a1, ..., ak
HAVING C2
```

S = any attributes  $a_1, \dots, a_k$  and/or any aggregates, but no other attributes

C1 = any condition on the attributes in  $R_1, \dots, R_n$

C2 = any condition on the aggregate expressions and attributes  $a_1, \dots, a_k$

# General Form of Group By

To say SELECT  $\alpha_1$

```
SELECT S
FROM R1, ..., Rn
WHERE C1
GROUP BY  $a_1, \dots, a_k$ 
HAVING C2
```

or HAVING  
 $\alpha_1 = \langle \text{something} \rangle$

... \*must\* GROUP BY  $\alpha_1$

S = any attributes  $a_1, \dots, a_k$  and/or any aggregates, but no other attributes

C1 = any condition on the attributes in  $R_1, \dots, R_n$

C2 = any condition on the aggregate expressions and attributes  $a_1, \dots, a_k$

# Outline

- Combining joins and aggregates
- The witnessing problem
  - (also known as 'argmax')

# Aggregates + Joins

Goal: how many cars made before 2017 does each person drive?

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

UserID	Car	Year
123	Charger	2009
567	Civic	2016
567	Pinto	2000
789	Camry	2018

# Aggregates + Joins

Goal: **how many** cars made before **2017** does **each person** drive?

Aggregate - COUNT  
and likely a GROUP

Attributes from two tables  
= JOIN

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

UserID	Car	Year
123	Charger	2009
567	Civic	2016
567	Pinto	2000
789	Camry	2018

# Aggregates + Joins

Goal: how many cars made before 2017 does each person drive?

Step 1: think about the join

```
SELECT ...  
  FROM Payroll p, Regist r  
  WHERE p.UserID = r.UserID
```

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

UserID	Car	Year
123	Charger	2009
567	Civic	2016
567	Pinto	2000
789	Camry	2018

# Aggregates + Joins

Goal: how many cars made before 2017 does each person drive?

Step 1: think about the join

(F WGHOS)

**SELECT** . . .

**FROM** Payroll p, Regist r

**WHERE** p.UserID = r.UserID

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

UserID	Car	Year
123	Charger	2009
567	Civic	2016
567	Pinto	2000
789	Camry	2018



# Aggregates + Joins

Goal: how many cars made before 2017 does each person drive?

Step 1: think about the join

```
SELECT ...  
  FROM Payroll p, Regist r  
  WHERE p.UserID = r.UserID
```

p.UserID	p.Name	p.Job	p.Salary	r.UserId	r.Car	r.Year
123	Jack	TA	50000	123	Charger	2009
567	Magda	Prof	90000	567	Civic	2016
567	Magda	Prof	90000	567	Pinto	2000
789	Dan	Prof	100000	789	Camry	2018

# Aggregates + Joins

Goal: how many cars made before 2017 does each person drive?

Step 1: think about the join...and where

```
SELECT ...  
  FROM Payroll p, Regist r  
  WHERE p.UserID = r.UserID AND  
         r.Year < 2017
```

p.UserID	p.Name	p.Job	p.Salary	r.UserId	r.Car	r.Year
123	Jack	TA	50000	123	Charger	2009
567	Magda	Prof	90000	567	Civic	2016
567	Magda	Prof	90000	567	Pinto	2000

# Aggregates + Joins

Goal: how many cars made before 2017 does each person drive?

Step 1: think about the join...and where

Step 2: do the group-by on the join

```
SELECT p.Name, COUNT (*)
FROM Payroll p, Regist r
WHERE p.UserID = r.UserID AND
        r.Year < 2017

GROUP BY p.Name
```

p.UserID	p.Name	p.Job	p.Salary	r.UserId	r.Car	r.Year
123	Jack	TA	50000	123	Charger	2009
567	Magda	Prof	90000	567	Civic	2016
567	Magda	Prof	90000	567	Pinto	2000

# Aggregates + Joins

Goal: how many cars made before 2017 does each person drive?

Step 1: think about the join...and where

Step 2: do the group-by on the join

```
SELECT p.Name, COUNT(*) AS count
FROM Payroll p, Regist r
WHERE p.UserID = r.UserID AND
        r.Year < 2017
GROUP BY p.Name
```

p.Name	count
Jack	1
Magda	2

# Aggregates + Joins

Goal: how many cars made before 2017 does each person drive?

Step 1: think about the join...and where

Step 2: do the group-by on the join

```
SELECT p.Name, COUNT(*) AS count
FROM Payroll p, Regist r
WHERE p.UserID = r.UserID AND
        r.Year < 2017
GROUP BY p.UserID, p.Name
```

Probably want to group by UserID  
too, in case multiple people have  
the same name

p.Name	count
Jack	1
Magda	2

# Including Empty Groups

Notice that empty groups were not included

- some people didn't have a car
- some people only had a new car

```
SELECT p.Name, COUNT(*) AS count
FROM Payroll p, Regist r
WHERE p.UserID = r.UserID AND
        r.Year < 2017
GROUP BY p.UserID, p.Name
```

p.Name	count
Jack	1
Magda	2

# Including Empty Groups

Notice that empty groups were not included

- some people didn't have a car
- some people only had a

COUNT(\*) will  
never be 0 for  
groups

```
SELECT p.Name, COUNT(*) AS count
FROM Payroll p, Regist r
WHERE p.UserID = r.UserID AND
      r.Year < 2017
GROUP BY p.UserID, p.Name
```

p.Name	count
Jack	1
Magda	2

# Including Empty Groups

How many cars does each person drive?  
(Remove our “older than 2017” constraint.)

Any ideas for which type of join we could use?

```
SELECT p.Name, COUNT(*) AS count
FROM Payroll p, Regist r
WHERE p.UserID = r.UserID
GROUP BY p.UserID, p.Name
```



# Including Empty Groups

```
SELECT p.Name, COUNT(r.UserID) AS count
FROM Payroll p LEFT OUTER JOIN
      Regist r
ON p.UserID = r.UserID
GROUP BY p.UserID, p.Name
```

p.UserID	p.Name	p.Job	p.Salary	r.UserId	r.Car	r.Year
123	Jack	TA	50000	123	Charger	2009
456	Allison	TA	60000	NULL	NULL	NULL
567	Magda	Prof	90000	567	Civic	2016
567	Magda	Prof	90000	567	Pinto	2000
789	Dan	Prof	100000	789	Camry	2018

# Including Empty Groups

```
SELECT p.Name, COUNT(r.UserID) AS count
FROM Payroll p LEFT OUTER JOIN
    Regist r
ON p.UserID = r.UserID
GROUP BY p.UserID, p.Name
```

p.UserID	p.Name	p.Job	p.Salary	r.UserId	r.Car	r.Year
123	Jack	TA	50000	123	Charger	2009
456	Allison	TA	60000	NULL	NULL	NULL
567	Magda	Prof	90000	567	Civic	2016
567	Magda	Prof	90000	567	Pinto	2000
789	Dan	Prof	100000	789	Camry	2018

# Including Empty Groups

```
SELECT p.Name, COUNT(r.UserID) AS count
FROM Payroll p LEFT OUTER JOIN
      Regist r
ON p.UserID = r.UserID
GROUP BY p.UserID, p.Name
```

p.Name	count
Jack	1
Allison	0
Magda	2
Dan	1

# Including Empty Groups

COUNT(attr)  
excludes NULL, so  
can be 0

```
SELECT p.Name, COUNT(r.UserID) AS count
FROM Payroll p LEFT OUTER JOIN
      Regist r
ON p.UserID = r.UserID
GROUP BY p.UserID, p.Name
```

p.Name	count
Jack	1
Allison	0
Magda	2
Dan	1

# Difference between count(attr) and count(\*)

```
SELECT p.Name, COUNT(*) AS count
FROM Payroll p LEFT OUTER JOIN
      Regist r
ON p.UserID = r.UserID
GROUP BY p.UserID, p.Name
```

BE CAREFUL!!!  
COUNT(\*) makes the  
answer wrong

p.Name	count
Jack	1
<b>Allison</b>	<b>1</b>
Magda	2
Dan	1

Return the person with the highest salary for each job type

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

# New Pattern

Return the person with the **highest salary** for each job type



Aggregate  
value

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

# New Pattern

Return the **person** with the **highest salary** for each job type

Single field  
equality with  
aggregate

Aggregate  
value

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000



# The Witnessing Problem

- Also known as argmax/argmin
- Ex: Return the person with the highest salary for each job type

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

**Definition:** In math we call this the argmax problem as we're trying to find the person (argument) that maximizes the salary (function of person)

# The Witnessing Problem

- Also known as argmax/argmin
- Ex: Return the person with the highest salary for each job type

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

```
SELECT Name, MAX(Salary)
FROM Payroll
GROUP BY Job
```

Easy right?

# The Witnessing Problem

- Also known as argmax/argmin
- Ex: Return the person with the highest salary for each job type

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

```
SELECT Name, MAX(Salary)
FROM Payroll
GROUP BY Job
```

Easy right?

**WRONG!**

# The Witnessing Problem

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

Name	MAX(Salary)
???	60000
???	100000

```
SELECT Name, MAX(Salary)
FROM Payroll
GROUP BY Job
```

# The Witnessing Problem

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

Name	MAX(Salary)
???	60000
???	100000

```
SELECT Name, MAX(Salary)
FROM Payroll
GROUP BY Job
```

# The Witnessing Problem

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

“Failed to execute query. Error: Column 'Payroll.name' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause.”

```
SELECT Name, MAX(Salary)
FROM Payroll
GROUP BY Job
```

# The Witnessing Problem

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

“Failed to execute query. Error: Column 'Payroll.name' is invalid in the select list because it is not contained in either an aggregate function or the GROUP BY clause.”

**WARNING: SQLite will allow this, and it shouldn't!!!**

**FROM** Payroll  
**GROUP BY** Job

# The Witnessing Problem

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

SELECT, HAVING, ORDER BY  
Must use aggregate functions  
or attributes in GROUP BY

	MAX(Salary)
	60000
???	100000

```
SELECT Name, MAX(Salary)
FROM Payroll
GROUP BY Job
```



# The Witnessing Problem

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

Return the person with the highest salary for each job type

How do we witness the maxima for a group?

**Discuss!**

Conceptual ideas are great

# The Witnessing Problem

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

Return the person with the highest salary for each job type

Main idea: we need to join the respective maxima to each row

# The Witnessing Problem

UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	60000
567	Magda	Prof	90000
789	Dan	Prof	100000

How do we get the maximum for each job type?

```
SELECT Job, MAX(Salary)
FROM Payroll
GROUP BY Job
```

Job	MAX(Salary)
TA	60000
Prof	100000

# The Witnessing Problem

UserID	Name	Job	Salary	maxima
123	Jack	TA	50000	60000
345	Allison	TA	60000	60000
567	Magda	Prof	90000	100000
789	Dan	Prof	100000	100000

Return the person with the highest salary for each job type

Main idea: we need to join the respective maxima to each row

```
SELECT Job, MAX(Salary)
FROM Payroll
GROUP BY Job
```

Job	maxima
TA	60000
Prof	100000

# The Witnessing Problem

UserID	Name	Job	Salary	maxima
123	Jack	TA	50000	60000
345	Allison	TA	60000	60000
567	Magda	Prof	90000	100000
789	Dan	Prof	100000	100000

Return the person with the highest salary for each job type

Main idea: we need to join the respective maxima to each row

```
SELECT Job, MAX(Salary)
FROM Payroll
GROUP BY Job
```

# The Witnessing Problem

UserID	Name	Job	Salary	maxima
123	Jack	TA	50000	60000
345	Allison	TA	60000	60000
567	Magda	Prof	90000	100000
789	Dan	Prof	100000	100000

Return the person with the highest salary for each job type

```
SELECT P1.Name, MAX(P2.Salary)
FROM Payroll AS P1, Payroll AS P2
WHERE P1.Job = P2.Job
GROUP BY P2.Job, P1.Salary, P1.Name
HAVING P1.Salary = MAX(P2.Salary)
```

# The Witnessing Problem

UserID	Name	Job	Salary	maxima
123	Jack	TA	50000	60000
345	Allison	TA	60000	60000
567	Magda	Prof	90000	100000
789	Dan	Prof	100000	100000

Return the person with the highest salary for each job type

the maxima

```
SELECT P1.Name, MAX(P2.Salary)
FROM Payroll AS P1, Payroll AS P2
WHERE P1.Job = P2.Job
GROUP BY P2.Job, P1.Salary, P1.Name
HAVING P1.Salary = MAX(P2.Salary)
```

# The Witnessing Problem

```
SELECT P1.Name, MAX(P2.Salary)
FROM Payroll AS P1, Payroll AS P2
WHERE P1.Job = P2.Job
GROUP BY P2.Job, P1.Salary, P1.Name
HAVING P1.Salary = MAX(P2.Salary)
```

Join on “original”  
grouping attributes

P1				P2			
UserID	Name	Job	Salary	UserID	Name	Job	Salary
123	Jack	TA	50000	123	Jack	TA	50000
123	Jack	TA	50000	345	Allison	TA	60000
345	Allison	TA	60000	345	Allison	TA	60000
345	Allison	TA	60000	123	Jack	TA	50000
567	Magda	Prof	90000	567	Magda	Prof	90000
567	Magda	Prof	90000	789	Dan	Prof	100000
789	Dan	Prof	100000	789	Dan	Prof	100000
789	Dan	Prof	100000	567	Magda	Prof	90000



# The Witnessing Problem

```
SELECT P1.Name, MAX(P2.Salary)
FROM Payroll AS P1, Payroll AS P2
WHERE P1.Job = P2.Job
GROUP BY P2.Job, P1.Salary, P1.Name
HAVING P1.Salary = MAX(P2.Salary)
```

Group on additional attributes that you are argmax-ing for

P1				P2			
UserID	Name	Job	Salary	UserID	Name	Job	Salary
123	Jack	TA	50000	123	Jack	TA	50000
123	Jack	TA	50000	345	Allison	TA	60000
345	Allison	TA	60000	345	Allison	TA	60000
345	Allison	TA	60000	123	Jack	TA	50000
567	Magda	Prof	90000	567	Magda	Prof	90000
567	Magda	Prof	90000	789	Dan	Prof	100000
789	Dan	Prof	100000	789	Dan	Prof	100000
789	Dan	Prof	100000	567	Magda	Prof	90000

# The Witnessing Problem

```
SELECT P1.Name, MAX(P2.Salary)
  FROM Payroll AS P1, Payroll AS P2
 WHERE P1.Job = P2.Job
GROUP BY P2.Job, P1.Salary, P1.Name
HAVING P1.Salary = MAX(P2.Salary)
```

Group on additional  
attributes that you are  
argmax-ing for

P1				P2			
UserID	Name	Job	Salary	UserID	Name	Job	Salary
123	Jack	TA	50000	123	Jack	TA	50000
123	Jack	TA	50000	345	Allison	TA	60000
345	Allison	TA	60000	345	Allison	TA	60000
345	Allison	TA	60000	123	Jack	TA	50000
567	Magda	Prof	90000	567	Magda	Prof	90000
567	Magda	Prof	90000	789	Dan	Prof	100000
789	Dan	Prof	100000	789	Dan	Prof	100000
789	Dan	Prof	100000	567	Magda	Prof	90000

# The Witnessing Problem

```
SELECT P1.Name, MAX(P2.Salary)
FROM Payroll AS P1, Payroll AS P2
WHERE P1.Job = P2.Job
GROUP BY P2.Job, P1.Salary, P1.Name
HAVING P1.Salary = MAX(P2.Salary)
```

Group on additional attributes that you are argmax-ing for

P1				P2			
UserID	Name	Job	Salary	UserID	Name	Job	Salary
123	Jack	TA	50000	123	Jack	TA	50000
123	Jack	TA	50000	345	Allison	TA	60000
345	Allison	TA	60000	345	Allison	TA	60000
345	Allison	TA	60000	123	Jack	TA	50000
567	Magda	Prof	90000	567	Magda	Prof	90000
567	Magda	Prof	90000	789	Dan	Prof	100000
789	Dan	Prof	100000	789	Dan	Prof	100000
789	Dan	Prof	100000	567	Magda	Prof	90000

# The Witnessing Problem

```
SELECT P1.Name, MAX(P2.Salary)
FROM Payroll AS P1, Payroll AS P2
WHERE P1.Job = P2.Job
GROUP BY P2.Job, P1.Salary, P1.Name
HAVING P1.Salary = MAX(P2.Salary)
```

P1

P2

UserID	Name	Job	Salary	UserID	Name	Job	Salary
123	Jack	TA	50000	123	Jack	TA	50000
123	Jack	TA	50000	345	Allison	TA	60000
345	Allison	TA	60000	345	Allison	TA	60000
345	Allison	TA	60000	123	Jack	TA	50000
567	Magda	Prof	90000	567	Magda	Prof	90000
567	Magda	Prof	90000	789	Dan	Prof	100000
789	Dan	Prof	100000	789	Dan	Prof	100000
789	Dan	Prof	100000	567	Magda	Prof	90000

# The Witnessing Problem

```
SELECT P1.Name, MAX(P2.Salary)
FROM Payroll AS P1, Payroll AS P2
WHERE P1.Job = P2.Job
GROUP BY P2.Job, P1.Salary, P1.Name
HAVING P1.Salary = MAX(P2.Salary)
```

Name	MAX(Salary)
Allison	60000
Dan	100000

# Takeaways

- FWGHOS
- Combining techniques (aggregates and joins) allows you to answer complex questions (e.g. witnessing queries)