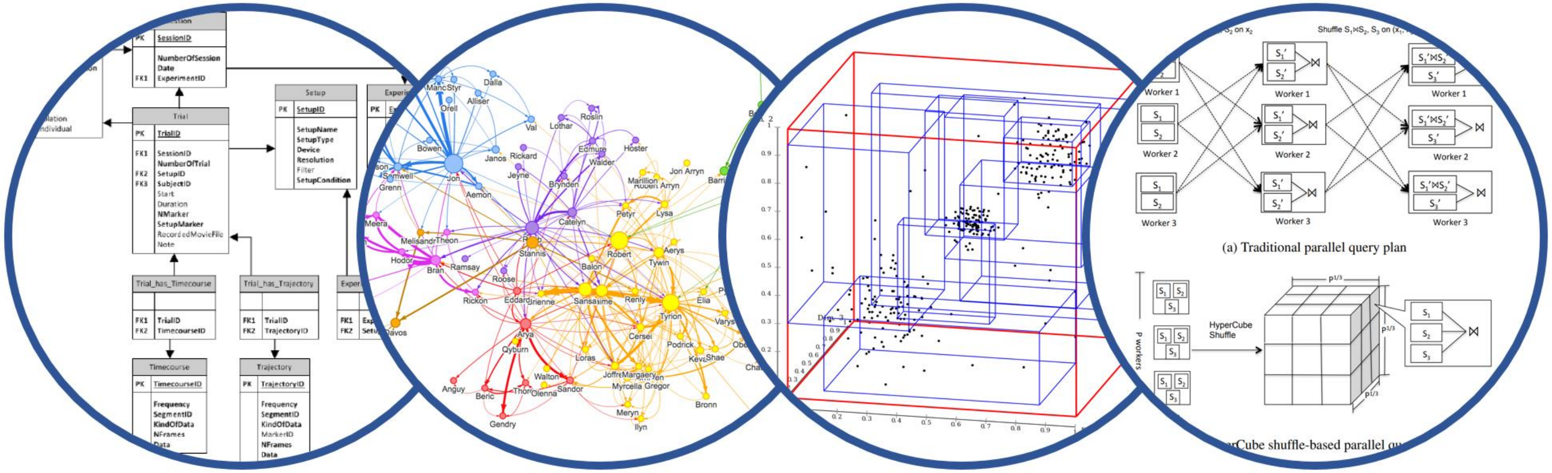


CSE 344: Intro to Data Management

SQL: Review (so Far!)

Paul G. Allen School of Computer Science and Engineering
University of Washington, Seattle

Announcements (1/2)

- HW2 due on Wednesday
- HW3
  - Please accept the invite from Azure: it expires soon!
  - Instructions for HW3 to be posted on Wednesday
  - Sections on Thursday will walk you through the setup

Announcements (2/2)

No in-person lectures Monday&Wednesday next week!

- Lectures will be recorded: canvas→zoom
- Please watch the lectures

Today’s Agenda

- Slow down a bit

- Some simple SQL constructs

- Recap/review SQL learned so far

The WITH Clause

The WITH Clause

- Define temporary tables
- Use them in a query

The WITH Clause

What is the average salary of car drivers?

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

The WITH Clause

What is the average salary of car drivers?

```
SELECT avg(P.Salary)
FROM Payroll P, Regist R
WHERE P.UserID = R.UserID;
```

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

The WITH Clause

What is the average salary of car drivers?

```
SELECT avg(P.Salary)
FROM Payroll P, Regist R
WHERE P.UserID = R.UserID;
```

```
SELECT P.Salary
FROM ...;
```

Name	Salary
Jack	50000
Magda	90000
Magda	90000

Duplicate!

avg(…)
76667

Wrong!

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

The WITH Clause

What is the average salary of car drivers?

```
SELECT avg(DISTINCT P.Salary)
FROM Payroll P, Regist R
WHERE P.UserID = R.UserID;
```

Does DISTINCT fix it?

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

The WITH Clause

What is the average salary of car drivers?

```
SELECT avg(DISTINCT P.Salary)
FROM Payroll P, Regist R
WHERE P.UserID = R.UserID;
```

Does DISTINCT fix it?

```
SELECT DISTINCT P.Salary
FROM ...;
```

Salary
50000
90000

avg(...)
70000

Wrong!

Correct answer: 63333

Payroll

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

Regist

UserID	Car
123	Charger
345	Tesla
567	Civic
567	Pinto

The WITH Clause

What is the average salary of car drivers?

We will solve this query by computing a temporary table using the WITH clause

Payroll

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

Regist

UserID	Car
123	Charger
345	Tesla
567	Civic
567	Pinto

The WITH Clause

What is the average salary of car drivers?

```
WITH Cardrivers AS
  (SELECT DISTINCT P.UserID, P.Salary
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID)
SELECT avg(Salary)
FROM Cardrivers;
```

Payroll

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

Regist

UserID	Car
123	Charger
345	Tesla
567	Civic
567	Pinto

The WITH Clause

What is the average salary of car drivers?

```
WITH Cardrivers AS
   (SELECT DISTINCT P.UserID, P.Salary
    FROM Payroll P, Regist R
    WHERE P.UserId=R.UserID)
SELECT avg(Salary)
FROM Cardrivers;
```

Payroll

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

Regist

UserID	Car
123	Charger
345	Tesla
567	Civic
567	Pinto

The WITH Clause

What is the average salary of car drivers?

```
WITH Cardrivers AS
   (SELECT DISTINCT P.UserID, P.Salary
    FROM Payroll P, Regist R
    WHERE P.UserId=R.UserID)
SELECT avg(Salary)
FROM Cardrivers;
```

Cardrivers

UserID	Salary
123	50000
345	50000
567	90000

Payroll

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

Regist

UserID	Car
123	Charger
345	Tesla
567	Civic
567	Pinto

The WITH Clause

What is the average salary of car drivers?

```
WITH Cardrivers AS
   (SELECT DISTINCT P.UserID, P.Salary
    FROM Payroll P, Regist R
    WHERE P.UserId=R.UserID)
SELECT avg(Salary)
FROM Cardrivers;
```

Cardrivers

UserID	Salary
123	50000
345	50000
567	90000

Payroll

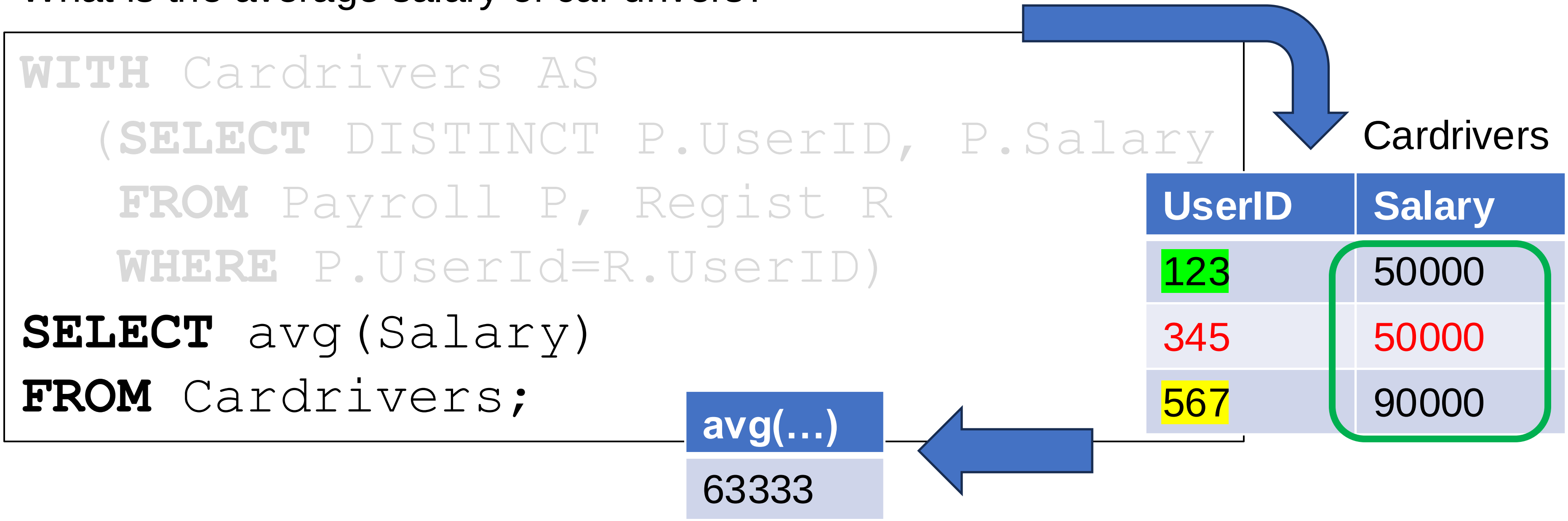
UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	50000
567	Magda	Prof	90000
789	Dan	Prof	100000

Regist

UserID	Car
123	Charger
345	Tesla
567	Civic
567	Pinto

The WITH Clause

What is the average salary of car drivers?



```
WITH Cardrivers AS
   (SELECT DISTINCT P.UserID, P.Salary
    FROM Payroll P, Regist R
    WHERE P.UserId=R.UserID)
SELECT avg(Salary)
FROM Cardrivers;
```

Cardrivers

UserID	Salary
123	50000
345	50000
567	90000

avg(…)
63333

Payroll

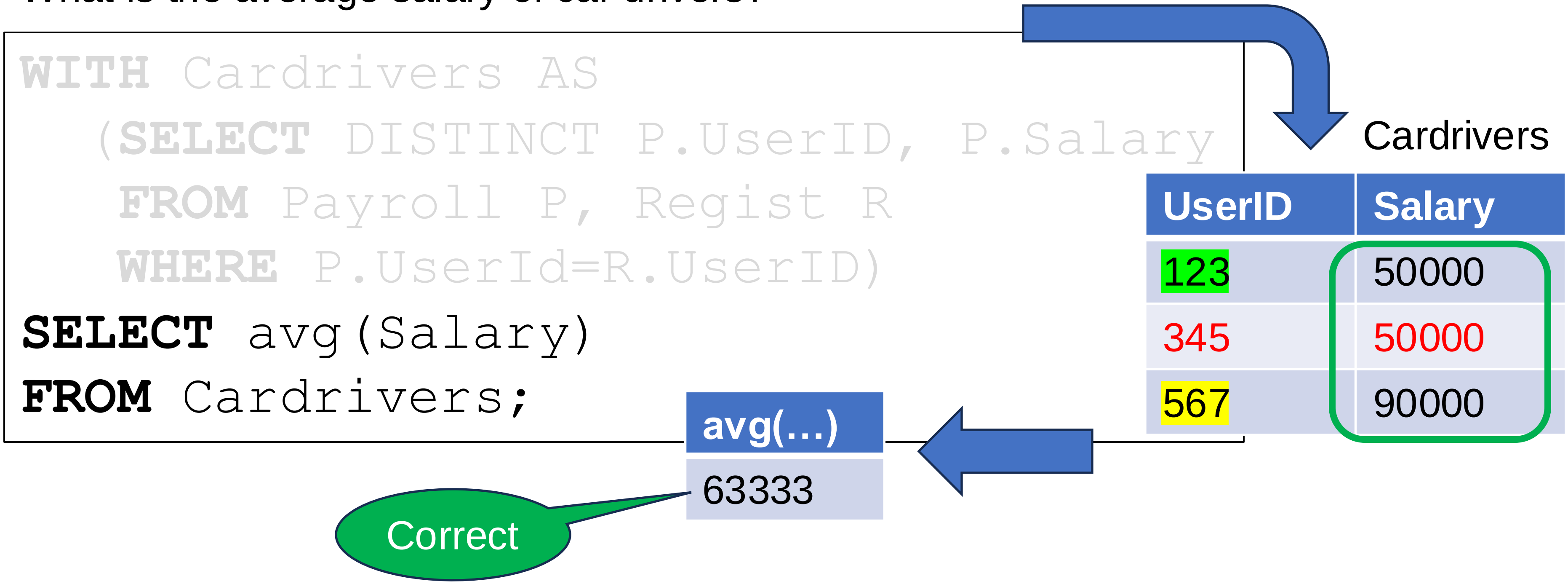
UserID	Name	Job	Salary
123	Jack	TA	50000
345	Allison	TA	50000
567	Magda	Prof	90000
789	Dan	Prof	100000

Regist

UserID	Car
123	Charger
345	Tesla
567	Civic
567	Pinto

The WITH Clause

What is the average salary of car drivers?



```
WITH Cardrivers AS
   (SELECT DISTINCT P.UserID, P.Salary
    FROM Payroll P, Regist R
    WHERE P.UserId=R.UserID)
SELECT avg(Salary)
FROM Cardrivers;
```

Cardrivers

UserID	Salary
123	50000
345	50000
567	90000

avg(…)
63333

Payroll

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

Regist

UserID	Car
123	Charger
345	Tesla
567	Civic
567	Pinto

The WITH Clause

General form:

```
WITH tbl1 as (SELECT ...),
     tbl2 as (SELECT ...),
     . . .
SELECT ...
FROM .....
WHERE ...
...
```

Here we may use tbl1, tbl2, …

Discussion

- A WITH construct is a simple form of a subquery
- We could also write the subquery in the FROM clause, but it is less readable

Next lecture

Views

Views

- A view is a table that is defined using a SQL query
- The table content is computed only when used

- The view becomes part of the persistent database

Views

- A view is a table that is defined using a SQL query
- The table content is computed only when used

Same as WITH

Different from WITH

- The view becomes part of the persistent database

Views

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

Views

Persistent database

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

Views

```
CREATE VIEW CarDrivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

Views

```
CREATE VIEW CarDrivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

Persistent database

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

CarDrivers

```
SELECT ...
FROM ...
```

Views

```
CREATE VIEW CarDrivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

Persistent database

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

CarDrivers

```
SELECT ...
FROM ...
```

Views

```
CREATE VIEW CarDrivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

```
SELECT *
FROM CarDrivers;
```

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

CarDrivers

```
SELECT ...
FROM ...
```

Views

```
CREATE VIEW CarDrivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

```
SELECT *
FROM CarDrivers;
```

UserID	Name	Job	Salary
123	Jack	TA	50000
567	Magda	Prof	90000

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

CarDrivers

```
SELECT ...
FROM ...
```

Views

```
CREATE VIEW CarDrivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

```
SELECT *
FROM CarDrivers;
```

UserID	Name	Job	Salary
123	Jack	TA	50000
567	Magda	Prof	90000

The view is computed at query time, with fresh data. Let's see that…

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

CarDrivers

```
SELECT ...
FROM ...
```

Views

```
CREATE VIEW CarDrivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

```
INSERT INTO Regist
VALUES(345,‘Tesla’);
```

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

CarDrivers

```
SELECT ...
FROM ...
```

Views

```
CREATE VIEW CarDrivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

```
INSERT INTO Regist
VALUES(345,'Tesla');
```

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto
345	Tesla

CarDrivers

```
SELECT ...
FROM ...
```

Views

```
CREATE VIEW CarDrivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

```
SELECT *
FROM CarDrivers;
```

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto
345	Tesla

CarDrivers

```
SELECT ...
FROM ...
```

Views

```
CREATE VIEW CarDrivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

```
SELECT *
FROM CarDrivers;
```

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

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto
345	Tesla

CarDrivers

```
SELECT ...
FROM ...
```

Views

```
CREATE VIEW CarDrivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

```
SELECT *
FROM CarDrivers;
```

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

Uses updated data

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto
345	Tesla

CarDrivers

```
SELECT ...
FROM ...
```

Views

All DBMS

Virtual View: means computed at query time

Materialized View: computed at definition time

Not in Sqlite

What are their pros and cons?

Views

All DBMS

Virtual View: means computed at query time

Materialized View: computed at definition time

Not in Sqlite

Advantage of virtual views:

- Always contains fresh data
- Query-time optimization (in class)

Disadvantages:

- Need to re-compute every time it is queried

Views

All DBMS

Virtual View: means computed at query time

Materialized View: computed at definition time

Not in Sqlite

Advantage of virtual views:

- Always contains fresh data
- Query-time optimization (in class)

Disadvantages:

- Need to re-compute every time it is queried

Advantage of materialize views:

- Computed only once

Disadvantages:

- Need to be updated when the input data is updated
- Incremental View Maintenance (IVM)

Views

All DBMS

Virtual View: means computed at query time

Materialized View: computed at definition time

Not in Sqlite

Advantage of virtual views:

- Always contains fresh data
- Query-time optimization (in class)

Disadvantages:

- Need to re-compute every time it is queried

Advantage of materialize views:

- Computed only once

Disadvantages:

- Need to be updated when the input data is updated
- Incremental View Maintenance (IVM)

We don’t discuss materialized views in this class

Materializing Query Outputs

Materializing Query Outputs

```
CREATE TABLE Drivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

Materializing Query Outputs

```
CREATE TABLE Drivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

Materializing Query Outputs

```
CREATE TABLE Drivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

Only the attributes from Payroll

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

Materializing Query Outputs

```
CREATE TABLE Drivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

Drivers

UserID	…	Salary
123	…	50000
567	…	90000

Materializing Query Outputs

```
CREATE TABLE Drivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

Persistent database

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

Drivers

UserID	…	Salary
123	…	50000
567	…	90000

Materializing Query Outputs

```
CREATE TABLE Drivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

How does this differ from a materialized view?

Persistent database

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

Drivers

UserID	…	Salary
123	…	50000
567	…	90000

Materializing Query Outputs

```
CREATE TABLE Drivers AS
  SELECT DISTINCT P.*
  FROM Payroll P, Regist R
  WHERE P.UserId=R.UserID;
```

How does this differ from a materialized view?

System will not update automatically

Persistent database

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

Drivers

UserID	…	Salary
123	…	50000
567	…	90000

Materializing Query Outputs

```
CREATE TABLE Drivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

```
SELECT *
FROM Drivers;
```

UserID	…	Salary
123	…	50000
567	…	90000

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto

Drivers

UserID	…	Salary
123	…	50000
567	…	90000

Materializing Query Outputs

```
CREATE TABLE Drivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

```
INSERT INTO Regist
VALUES(345,'Tesla');
```

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto
345	Tesla

Drivers

UserID	…	Salary
123	…	50000
567	…	90000

Materializing Query Outputs

```
CREATE TABLE Drivers AS
   SELECT DISTINCT P.*
   FROM Payroll P, Regist R
   WHERE P.UserId=R.UserID;
```

```
SELECT *
FROM Drivers;
```

UserID	…	Salary
123	…	50000
567	…	90000

Uses stale data

Payroll

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

Regist

UserID	Car
123	Charger
567	Civic
567	Pinto
345	Tesla

Drivers

UserID	…	Salary
123	…	50000
567	…	90000

More about GROUP BY

More GROUP BY

- So far, we grouped only by attributes

- We can also group by expressions!

More GROUP BY

Find the total revenue per company and decade

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

More GROUP BY

Find the total revenue per company and decade

We want this:

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

Start	End	Name	Total
1990	1999	Acme	250000
1990	1999	IBM	…
2000	2009	Acme	…
…			
2010	2019	IBM	450000
…			

More GROUP BY

Find the total revenue per company and decade

```
SELECT Year/10*10 AS Start, Year/10*10 + 9 AS End,
       Name, sum(Revenue)
FROM Company
GROUP BY Year/10*10, Year/10*10 + 9, Name;
```

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

Start	End	Name	Total
1990	1999	Acme	250000
1990	1999	IBM	…
2000	2009	Acme	…
…			
2010	2019	IBM	450000
…			

More GROUP BY

Find the total revenue per company and decade

```
SELECT Year/10*10 AS Start, Year/10*10 + 9 AS End,
       Name, sum(Revenue)
FROM Company
GROUP BY Year/10*10, Year/10*10 + 9, Name;
```

Integer division
or use cast(…)

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

Start	End	Name	Total
1990	1999	Acme	250000
1990	1999	IBM	…
2000	2009	Acme	…
…			
2010	2019	IBM	450000
…			

More GROUP BY

Find the total revenue per company and decade

```
SELECT Year/10*10 AS Start, Year/10*10 + 9 AS End,
       Name, sum(Revenue)
FROM Company
GROUP BY Year/10*10, Year/10*10 + 9, Name;
```

Beginning of decade

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

Start	End	Name	Total
1990	1999	Acme	250000
1990	1999	IBM	…
2000	2009	Acme	…
…			
2010	2019	IBM	450000
…			

More GROUP BY

Find the total revenue per company and decade

```
SELECT Year/10*10 AS Start, Year/10*10 + 9 AS End,
       Name, sum(Revenue)
FROM Company
GROUP BY Year/10*10, Year/10*10 + 9, Name;
```

Beginning of decade

End of decade

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

Start	End	Name	Total
1990	1999	Acme	250000
1990	1999	IBM	…
2000	2009	Acme	…
…			
2010	2019	IBM	450000
…			

More GROUP BY

Find the total revenue per company and decade

Needs to occur in GROUP BY

```
SELECT Year/10*10 AS Start, Year/10*10 + 9 AS End,
       Name, sum(Revenue)
FROM Company
GROUP BY Year/10*10, Year/10*10 + 9, Name;
```

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

Start	End	Name	Total
1990	1999	Acme	250000
1990	1999	IBM	…
2000	2009	Acme	…
…			
2010	2019	IBM	450000
…			

More GROUP BY

Find the total revenue per company and decade

```
SELECT Year/10*10 AS Start, Year/10*10 + 9 AS End,
       Name, sum(Revenue)
FROM Company
GROUP BY Start, End, Name;
```

Sqlite allows this. Nice ☺

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

Start	End	Name	Total
1990	1999	Acme	250000
1990	1999	IBM	…
2000	2009	Acme	…
…			
2010	2019	IBM	450000
…			

More GROUP BY

Find the total revenue in a sliding window of 10 years

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

More GROUP BY

Find the total revenue in a sliding window of 10 years

We want this:

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

Start	End	Name	Total
1990	1999		
1991	2000		
1992	2001		
…			
2013	2022		
…			

More GROUP BY

Find the total revenue in a sliding window of 10 years

```
SELECT X.Year, X.Year+9, X.Name, sum(Y.Revenue)
FROM Company X, Company Y
WHERE X.Name = Y.Name
  and X.Year <= Y.Year and Y.Year < X.Year+10
GROUP BY X.Year, X.Year+9, X.Name
ORDER BY X.Year;
```

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

Start	End	Name	Total
1990	1999		
1991	2000		
1992	2001		
…			
2013	2022		
…			

More GROUP BY

Find the total revenue in a sliding window of 10 years

```
SELECT X.Year, X.Year+9, X.Name, sum(Y.Revenue)
FROM Company X, Company Y
WHERE X.Name = Y.Name
  and X.Year <= Y.Year and Y.Year < X.Year+10
GROUP BY X.Year, X.Year+9, X.Name
ORDER BY X.Year;
```

Gaps if year not in database

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

Start	End	Name	Total
1990	1999		
1991	2000		
1992	2001		
…			
2013	2022		
…			

More GROUP BY

Find the total revenue in a sliding window of 10 years

```
SELECT X.Year, X.Year+9, X.Name, sum(Y.Revenue)
FROM Company X, Company Y
WHERE X.Name = Y.Name
  and X.Year <= Y.Year and Y.Year < X.Year+10
GROUP BY X.Year, X.Year+9, X.Name
ORDER BY X.Year;
```

Gaps if year not in database

Postgres: generate_series(1999,2024)

Company

Name	Revenue	Year
Acme	100000	1995
IBM	200000	2012
Apple	300000	2012
IBM	250000	2019
…		

Start	End	Name	Total
1990	1999		
1991	2000		
1992	2001		
…			
2013	2022		
…			

Discussion

- GROUP-BY is versatile and powerful
- Optimizers can often find every efficient plans
- SQL also has “windows function”: very complex
    - Will not discuss in class

The Witness

The Witness

- SQL provides min/max, but not argmin/argmax

- Record that achieves min/max: The Witness

- Several ways to compute it:
  - WITH
  - Self-join and HAVING

The Witnessing Problem

Find the person with highest salary for each job

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

Desired answer:

Job	Name	Salary
TA	Allison	60000
Prof	Dan	100000

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

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

Job	Salary
TA	60000
Prof	100000

Finding max is easy.

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

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

Job	Salary
TA	60000
Prof	100000

Finding max is easy.

Payroll

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

But we want argmax.
How do we find
the witness?

The Witnessing Problem

Find the person with highest salary for each job

Solution 1: Using WITH
Solution 2: Using HAVING

Plan:

1. Compute the max(Salary) for each Job
2. Join back with Payroll on Job
3. Return the users where Salary = max(Salary)

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

```
WITH JobSal AS
  (SELECT Job, max(Salary) AS M
   FROM Payroll
   GROUP BY Job)
SELECT J.Job, P.Name, P.Salary
FROM JobSal J, Payroll P
WHERE J.Job = P.Job
   and J.M = P.Salary;
```

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

```
WITH JobSal AS
  (SELECT Job, max(Salary) AS M
   FROM Payroll
   GROUP BY Job)
SELECT J.Job, P.Name, P.Salary
FROM JobSal J, Payroll P
WHERE J.Job = P.Job
   and J.M = P.Salary;
```

Job	M
TA	60000
Prof	100000

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

```
WITH JobSal AS
  (SELECT Job, max(Salary) AS M
   FROM Payroll
   GROUP BY Job)
SELECT J.Job, P.Name, P.Salary
FROM JobSal J, Payroll P
WHERE J.Job = P.Job
   and J.M = P.Salary;
```

Job	M
TA	60000
Prof	100000

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

```
WITH JobSal AS
  (SELECT Job, max(Salary) AS M
   FROM Payroll
   GROUP BY Job)
SELECT J.Job, P.Name, P.Salary
FROM JobSal J, Payroll P
WHERE J.Job = P.Job
   and J.M = P.Salary;
```

Job	M
TA	60000
Prof	100000

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

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

```
WITH JobSal AS
  (SELECT Job, max(Salary) AS M
   FROM Payroll
   GROUP BY Job)
SELECT J.Job, P.Name, P.Salary
FROM JobSal J, Payroll P
WHERE J.Job = P.Job
   and J.M = P.Salary;
```

Job	M
TA	60000
Prof	100000

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

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

```
WITH JobSal AS
  (SELECT Job, max(Salary) AS M
   FROM Payroll
   GROUP BY Job)
SELECT J.Job, P.Name, P.Salary
FROM JobSal J, Payroll P
WHERE J.Job = P.Job
   and J.M = P.Salary;
```

Payroll

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

Job	M
TA	60000
Prof	100000

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

Job	Name	Salary
TA	Allison	60000
Prof	Dan	100000

The Witnessing Problem

Find the person with highest salary for each job

Solution 1: Using WITH
Solution 2: Using HAVING

Plan:

1. Compute the max(Salary) for each Job
2. Join back with Payroll on Job
3. Return the users where Salary = max(Salary)

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

We first join

Plan:

1. Compute the max(Salary) for each Job
2. Join back with Payroll on Job
3. Return the users where Salary = max(Salary)

Goes in HAVING

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

```
SELECT P1.Job, MAX(P1.Salary)
FROM Payroll AS P1

GROUP BY P1.Job
```

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

```
SELECT P1.Job, MAX(P1.Salary)
FROM Payroll AS P1

GROUP BY P1.Job
```

Similar to JobSal
in our first solution

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

```
SELECT P1.Job, ~~MAX(P1.Salary)~~
FROM Payroll AS P1

GROUP BY P1.Job
```

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

```
SELECT P1.Job
FROM Payroll AS P1

GROUP BY P1.Job
```

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

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

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

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

Similar to joining JobSal with Payroll

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

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

Incorrect!

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

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

Correct; but not done!

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

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

Which P2 should we return for each Job?

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

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

Payroll

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

The Witnessing Problem

Find the person with highest salary for each job

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

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

The Witnessing Problem

Find the person with highest salary for each job

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

Payroll join with Payroll

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

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

The Witnessing Problem

Find the person with highest salary for each job

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

Group by

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

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

The Witnessing Problem

Find the person with highest salary for each job

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

Compute max(P1.Salary)

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

max(salary)=60000

max(salary)=60000

max(salary)=100000

max(salary)=100000

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

The Witnessing Problem

Find the person with highest salary for each job

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

Check HAVING

P1 P2

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

max(salary)=60000

max(salary)=60000

max(salary)=100000

max(salary)=100000

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

The Witnessing Problem

Find the person with highest salary for each job

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

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

max(salary)=60000

max(salary)=60000

max(salary)=100000

max(salary)=100000

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

P1.Job	P2.Name	P2.Salary
TA	Allison	60000
Prof	Dan	100000

The Witnessing Problem

Find the person with highest salary for each job

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

Final output has the witnesses

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

P1.Job	P2.Name	P2.Salary
TA	Allison	60000
Prof	Dan	100000

Summary

- We have covered now almost all SQL that we discuss in class

- Only one subtle topic remains: subqueries!

- Will discuss subqueries on Wednesday and Friday