

Introduction to Data Management

Joining Tables

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

Announcements

- HW1 due tonight!
- HW2 will be posted tonight, due next Friday

Recap: Keys and Foreign Keys

Foreign Key

A **Key** is one or more attributes that **uniquely** identify a row.

A **Foreign Key** is one or more attrs that uniquely identify a row in *another table*.

Recap: Keys and Foreign Keys

Foreign Key

A **Key** is one or more attributes that **uniquely** identify a row.

A **Foreign Key** is one or more attrs that uniquely identify a row in *another table*.

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

Recap: Keys and Foreign Keys

Foreign Key

A **Key** is one or more attributes that **uniquely** identify a row.

A **Foreign Key** is one or more attrs that uniquely identify a row in *another table*.

References



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

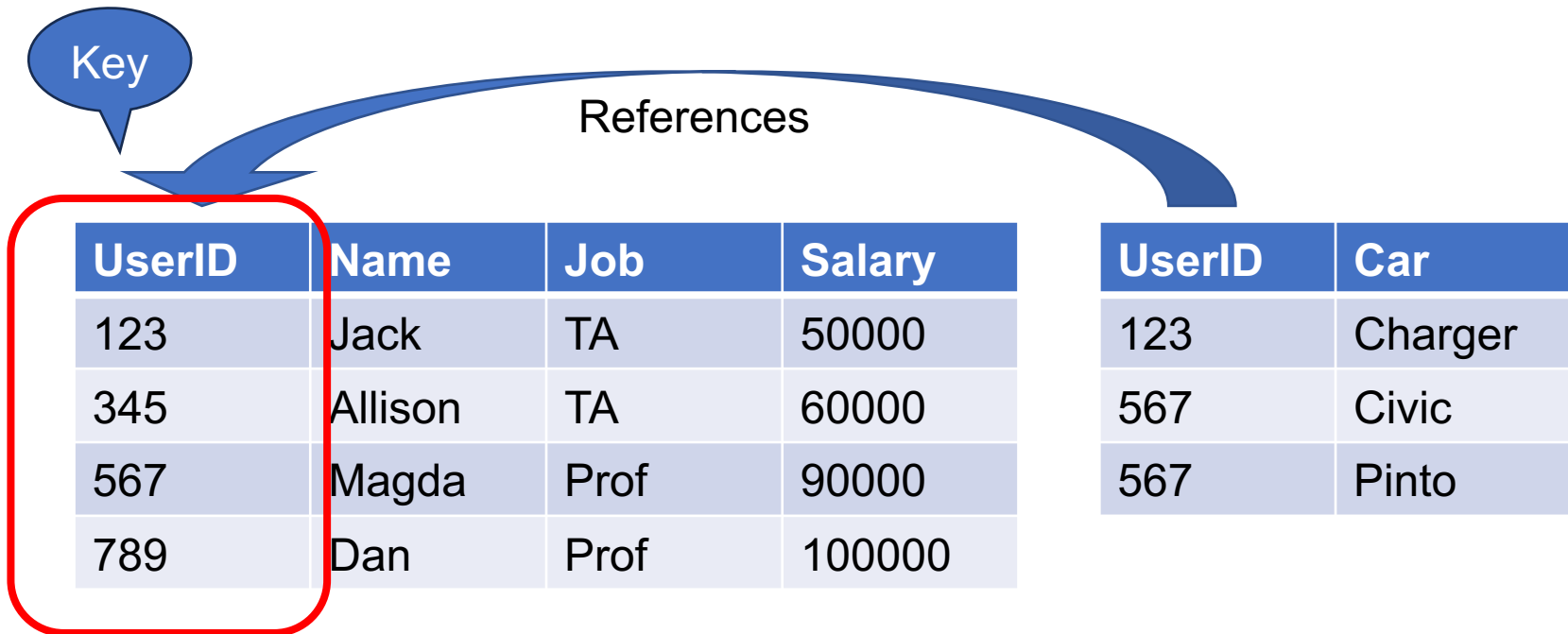
UserID	Car
123	Charger
567	Civic
567	Pinto

Recap: Keys and Foreign Keys

Foreign Key

A **Key** is one or more attributes that **uniquely** identify a row.

A **Foreign Key** is one or more attrs that uniquely identify a row in *another table*.

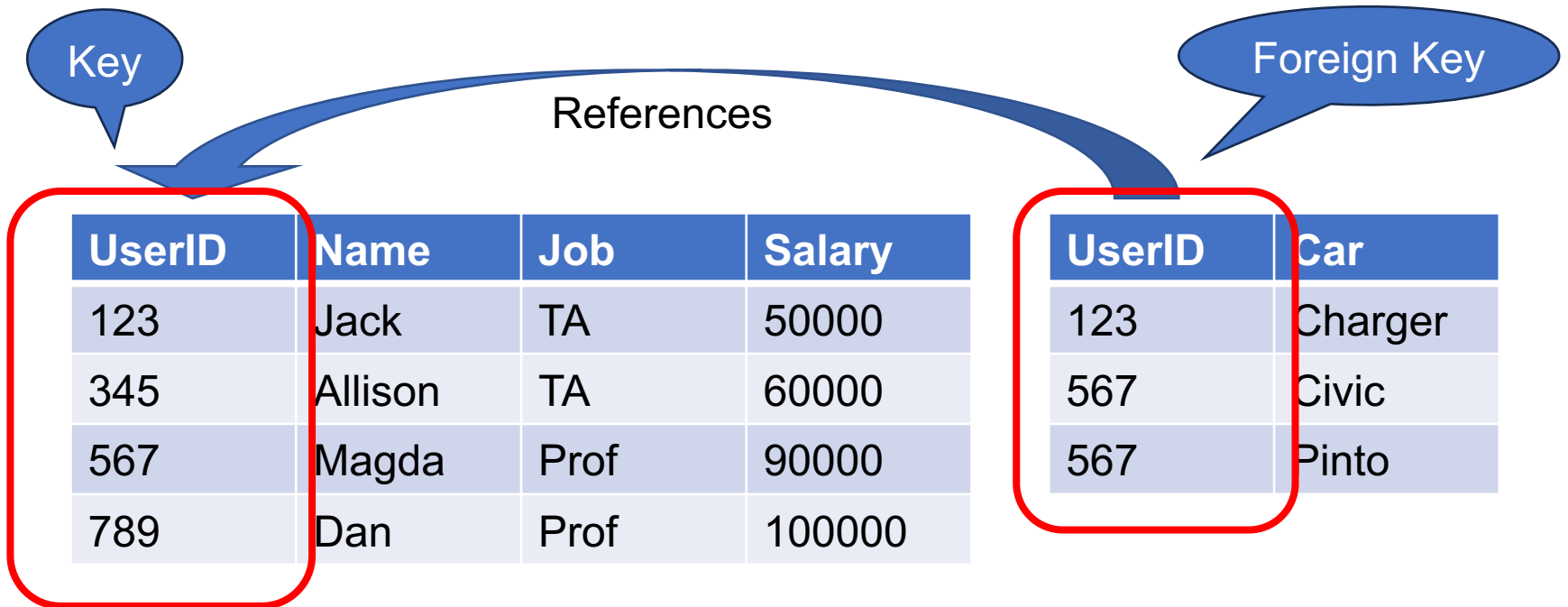


Recap: Keys and Foreign Keys

Foreign Key

A **Key** is one or more attributes that **uniquely** identify a row.

A **Foreign Key** is one or more attrs that uniquely identify a row in *another table*.



Joins

- Joins link records from different tables.
- May use the key / foreign-key relationship, but may also use any other relationships

For each employee, find the cars that they drive

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

Join

For each employee, find the cars that they drive

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS 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

Join

For each employee, find the cars that they drive

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS 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

Join

For each employee, find the cars that they drive

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID;
```



Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

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

Join

For each TA, find the cars that they drive

```
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID
    and P.Job = 'TA';
```

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

Join

For each TA, find the cars that they drive

```
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID
      and P.Job = 'TA';
```



Name	Car
Jack	Charger

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

Join

For each TA, find the cars that they drive

```
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID
    and P.Job = 'TA';
```



Name	Car
Jack	Charger

and is a Boolean expression; let's review.

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

Boolean Expression

- AND, OR, NOT
- Example: are these true or false?

$(5 < 7) \text{ or } (7 < 5)$

$(5 < 7) \text{ or } (5 < 8)$

$(5 < 7) \text{ and } (7 < 5)$

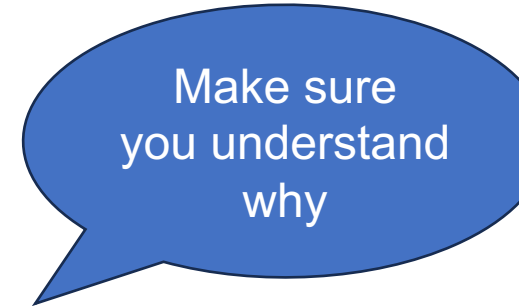
$(5 < 7) \text{ and not}(7 < 5)$

$(1 < 2) \text{ and } ((2 \neq 3) \text{ or } (4 < 3))$

Boolean Expression

- AND, OR, NOT

- Example: are these true or false?



$(5 < 7) \text{ or } (7 < 5)$

TRUE

$(5 < 7) \text{ or } (5 < 8)$

TRUE

$(5 < 7) \text{ and } (7 < 5)$

FALSE

$(5 < 7) \text{ and not}(7 < 5)$

TRUE

$(1 < 2) \text{ and } ((2 \neq 3) \text{ or } (4 < 3))$

TRUE

Boolean Expression

In the WHERE clause: may use AND, OR, NOT

```
SELECT Name  
FROM Payroll  
WHERE Job = 'TA' or (Salary > 55000 and Salary < 95000)
```

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

Boolean Expression

In the WHERE clause: may use AND, OR, NOT

```
SELECT Name
FROM Payroll
WHERE Job = 'TA' or (Salary > 55000 and Salary < 95000)
```

Name

Jack

Allison

Magda

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

Boolean Expression

In the WHERE clause: may use AND, OR, NOT

```
SELECT Name
FROM Payroll
WHERE Job = 'TA' or (Salary > 55000 and Salary < 95000)
```

```
SELECT Name
FROM Payroll
WHERE Job = 'TA' and (Salary > 55000 and Salary < 95000)
```

Name

Jack

Allison

Magda

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

Boolean Expression

In the WHERE clause: may use AND, OR, NOT

```
SELECT Name
FROM Payroll
WHERE Job = 'TA' or (Salary > 55000 and Salary < 95000)
```

```
SELECT Name
FROM Payroll
WHERE Job = 'TA' and (Salary > 55000 and Salary < 95000)
```

Name

Jack

Allison

Magda

Name

Allison

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

- When we use joins we often have multiple conditions in the *WHERE* clause: and/or/not

- Next: two ways to write the join

Join: Two Ways to Write a Join

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS 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

Join: Two Ways to Write a Join

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID;
```

```
SELECT P.Name, R.Car  
FROM Payroll AS P JOIN Regist AS R  
ON P.UserID = R.UserID;
```

Means the same thing

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

Join: Two Ways to Write a Join

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID  
       and P.Job = 'TA';
```

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

Join: Two Ways to Write a Join

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID  
       and P.Job = 'TA';
```

```
SELECT P.Name, R.Car  
FROM Payroll AS P JOIN Regist AS R  
       ON P.UserID = R.UserID  
WHERE P.Job = 'TA';
```

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

Join: Two Ways to Write a Join

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID  
      and P.Job = 'TA';
```

```
SELECT P.Name, R.Car  
FROM Payroll AS P JOIN Regist AS R  
      ON P.UserID = R.UserID  
WHERE P.Job = 'TA';
```

```
SELECT P.Name, R.Car  
FROM Payroll AS P JOIN Regist AS R  
      ON P.UserID = R.UserID  
      and P.Job = 'TA';
```

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

Join: Two Ways to Write a Join

ON same as WHERE
for now; but wait for it...

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID  
      and P.Job = 'TA';
```

```
SELECT P.Name, R.Car  
FROM Payroll AS P JOIN Regist AS R  
      ON P.UserID = R.UserID  
WHERE P.Job = 'TA';
```

```
SELECT P.Name, R.Car  
FROM Payroll AS P JOIN Regist AS R  
      ON P.UserID = R.UserID  
      and P.Job = 'TA';
```

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

- A join is often between a key and a foreign key

- But not always! Let's see some examples

Join

```
-- find the cars they are driving
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID;
```



Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

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

Join

```
-- find the cars they are driving
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID;
```

```
-- find the cars they are not driving
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID != R.UserID;
```



Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

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

Join

```
-- find the cars they are driving
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID;
```

```
-- find the cars they are not driving
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID != R.UserID;
```



Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

Name	Car
Jack	Civic
Jack	Pinto
Allison	Charger
Allison	...
...	

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Join

```
-- find the cars they are driving
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID;
```

```
-- find the cars they are not driving
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID != R.UserID;
```

```
-- find WHAT??
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
```



Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

Name	Car
Jack	Civic
Jack	Pinto
Allison	Charger
Allison	...
...	

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

Join

```
-- find the cars they are driving
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID;
```

```
-- find the cars they are not driving
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID != R.UserID;
```

```
-- find WHAT??
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
```



.....

Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

Name	Car
Jack	Civic
Jack	Pinto
Allison	Charger
Allison	...
...	

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

Discussion

- FROM clause: several table names
- WHERE clause: some condition on these tables
- **Q:** What does it mean?
- **A:** For-Each semantics (Nested Loop Semantics)!

Nested Loop Semantics (again!)

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID;
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID;
```

Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID;
```

How do we
algorithmically
get our results?

Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID;
```

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
------	-----

```
for each row1 in Payroll:
  for each row2 in Regist:
    if (row1.UserID = row2.UserID):
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger

```
for each row1 in Payroll:
  for each row2 in Regist:
    if (row1.UserID = row2.UserID):
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```


Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger

```
for each row1 in Payroll:
  for each row2 in Regist:
    if (row1.UserID = row2.UserID):
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger
Magda	Civic

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger
Magda	Civic

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

```
for each row1 in Payroll:
  for each row2 in Regist:
    if (row1.UserID = row2.UserID):
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

```
for each row1 in Payroll:
  for each row2 in Regist:
    if (row1.UserID = row2.UserID):
      output (row1.Name, row2.Car)
```


Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

```
for each row1 in Payroll:
  for each row2 in Regist:
    if (row1.UserID = row2.UserID):
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID;
```



Key / Foreign-key join

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

Nested-Loop Semantics

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

UserID	Car
123	Charger
567	Civic
567	Pinto

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R  
WHERE P.UserID = R.UserID;
```



Key / Foreign-key join

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    if (row1.UserID = row2.UserID):  
      output (row1.Name, row2.Car)
```

```
SELECT P.Name, R.Car  
FROM Payroll AS P, Regist AS R;
```



Cross product

```
for each row1 in Payroll:  
  for each row2 in Regist:  
    output (row1.Name, row2.Car)
```

Summary: Nested-Loop Semantics

- FROM clause has tables T1, T2, T3, ...
- WHERE clause has condition

```
for each r1 in T1:  
  for each t2 in T2:  
    for each t3 in T3:  
      ...  
      if (condition):  
        output (r1,r2,...)
```

Self-Joins

Self Joins

Find all people who drive a Civic and Pinto

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

UserID	Car
123	Charger
123	Pinto
123	Tesla
567	Civic
567	Pinto

```
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID AND
        R.Car = 'Civic';
```

Self Joins

Find all people who drive a Civic and Pinto

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

UserID	Car
123	Charger
123	Pinto
123	Tesla
567	Civic
567	Pinto

```
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID AND
      R.Car = 'Civic' AND
      R.Car = 'Pinto';
```

Will this work?

Self Joins

Find all people who drive a Civic and Pinto

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

UserID	Car
123	Charger
123	Pinto
123	Tesla
567	Civic
567	Pinto

```
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID AND
      R.Car = 'Civic' AND
      R.Car = 'Pinto';
```

Will this work?
Nope, returns
the empty set.

Self Joins

Find all people who drive a Civic and Pinto

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

UserID	Car
123	Charger
123	Pinto
123	Tesla
567	Civic
567	Pinto

```
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID AND
      (R.Car = 'Civic' OR
       R.Car = 'Pinto');
```

Is this better?

Self Joins

Find all people who drive a Civic and Pinto

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

UserID	Car
123	Charger
123	Pinto
123	Tesla
567	Civic
567	Pinto

```
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID AND
      (R.Car = 'Civic' OR
       R.Car = 'Pinto');
```

Is this better?
Nope, it returns
Jack, Magda.

Self Joins

Find all people who drive a Civic and Pinto

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

UserID	Car
123	Charger
123	Pinto
123	Tesla
567	Civic
567	Pinto

```
SELECT P.Name, R.Car
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID AND
      (R.Car = 'Civic' OR
       R.Car = 'Pinto');
```

Discuss with the people around you how you would solve this.

Self Joins

Find all people who drive a Civic and Pinto

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

UserID	Car
123	Charger
123	Pinto
123	Tesla
567	Civic
567	Pinto

```
SELECT P.Name, R1.Car
FROM Payroll AS P, Regist AS R1, Regist AS R2
WHERE P.UserID = R1.UserID AND
      P.UserID = R2.UserID AND
      R1.Car = 'Civic' AND
      R2.Car = 'Pinto';
```

Self Joins

Find all people who drive a Civic and Pinto

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

UserID	Car
123	Charger
123	Pinto
123	Tesla
567	Civic
567	Pinto

Name	Job
Magda	Civic
Magda	Pinto

```
SELECT P.Name, R1.Car
FROM Payroll AS P, Regist AS R1, Regist AS R2
WHERE P.UserID = R1.UserID AND
        P.UserID = R2.UserID AND
        R1.Car = 'Civic' AND
        R2.Car = 'Pinto';
```



Self Joins

Find all people who drive a Civic and Pinto

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

UserID	Car
123	Charger
123	Pinto
123	Tesla
567	Civic
567	Pinto

Name	Job
Magda	Civic
Magda	Pinto

```
SELECT P.Name, R1.Car
FROM Payroll AS P, Regist AS R1, Regist AS R2
WHERE P.UserID = R1.UserID AND
P.UserID = R2.UserID AND
R1.Car = 'Civic' AND
R2.Car = 'Pinto';
```

The person we look for must drive TWO cars

Self Joins

- When a relation occurs twice in the FROM clause we call it a “self-join”

- If we have a self-join, we must use table aliases; Otherwise, the attribute names are ambiguous

Outer Joins

Join

For each employee, find the cars that they drive

```
SELECT P.Name, R.Car
FROM Payroll AS P
      JOIN Regist AS R
      ON P.UserID = R.UserID;
```



Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

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

Join

For each employee, find the cars that they drive

```
SELECT P.Name, R.Car
FROM Payroll AS P
      JOIN Regist AS R
      ON P.UserID = R.UserID;
```



Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto

Allison, Dan
are missing

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

Join

For each employee, find the cars that they drive

```
SELECT P.Name, R.Car
FROM Payroll AS P
LEFT OUTER JOIN Regist AS R
ON 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

Join

For each employee, find the cars that they drive

```
SELECT P.Name, R.Car
FROM Payroll AS P
LEFT OUTER JOIN Regist AS R
ON P.UserID = R.UserID;
```



Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto
Allison	NULL
Dan	NULL

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

Join

For each employee, find the cars that they drive

```
SELECT P.Name, R.Car
FROM Payroll AS P
LEFT OUTER JOIN Regist AS R
ON P.UserID = R.UserID;
```



Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto
Allison	NULL
Dan	NULL

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

NULL means
"unknown" or
"missing"

Join

For each employee, find the cars that they drive

```
SELECT P.Name, R.Car
FROM Payroll AS P
LEFT OUTER JOIN Regist AS R
ON P.UserID = R.UserID;
```



Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto
Allison	NULL
Dan	NULL

Left outer join:

1. Perform the join with the ON clause
2. Add all missing tuples from LEFT
3. Check the WHERE clause (if present)

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

Join

For each employee, find the cars that they drive

```
SELECT P.Name, R.Car
FROM Payroll AS P
LEFT OUTER JOIN Regist AS R
ON P.UserID = R.UserID;
```



Name	Car
Jack	Charger
Magda	Civic
Magda	Pinto
Allison	NULL
Dan	NULL

Left outer join:

1. Perform the join with the ON clause
2. Add all missing tuples from LEFT
3. Check the WHERE clause (if present)

ON, WHERE differ

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

Outer Joins


- **LEFT OUTER JOIN**
 - Add missing tuples from the **LEFT**

- **RIGHT OUTER JOIN**
 - Add missing tuples from the **RIGHT**

- **FULL OUTER JOIN**
 - Add missing tuples from both

Outer Joins

- **LEFT OUTER JOIN**
 - Add missing tuples from the **LEFT**
- **RIGHT OUTER JOIN**
 - Add missing tuples from the **RIGHT**
- **FULL OUTER JOIN**
 - Add missing tuples from both



Useful, especially
for aggregates
(next week)



Rarely
used