Introduction to Data Management

Join and Aggregate Practice

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

July 1, 2019
Announcements

- HW 1 due Wednesday, 11pm
  - Turnin script has a fix
  - Run “git pull upstream master” and then do a git commit

- Friday (7/5) class is cancelled
Another example

Product(pname, price, category, manufacturer)
Company(cname, country)
-- manufacturer is foreign key to Company

Find US companies that manufacture both ‘gadgets’ and ‘photo’ products

SELECT DISTINCT z.cname
FROM Product x, Product y, Company z
WHERE z.country = 'USA'
    AND x.manufacturer = z.cname
    AND y.manufacturer = z.cname
    AND x.category = 'gadget'
    AND y.category = 'photography';

Need to include Product twice!
Self-Joins and Tuple Variables

- Joining Product with Company is insufficient: need to join Product, with Product, and with Company

- When a relation occurs twice in the FROM clause we call it a self-join; in that case we must use tuple variables (aka table aliases) (why?)

Find US companies that manufacture both ‘gadgets’ and ‘photo’ products
SELECT DISTINCT z.cname
FROM Product x, Product y, Company z
WHERE z.country = 'USA'
    AND x.category = 'gadget'
    AND y.category = 'photo'
    AND x.manufacturer = z.cname
    AND y.manufacturer = z.cname;
Self-Joins

```
SELECT DISTINCT z.cname
FROM Product x, Product y, Company z
WHERE z.country = 'USA'
  AND x.category = 'gadget'
  AND y.category = 'photo'
  AND x.manufacturer = z.cname
  AND y.manufacturer = z.cname;
```
SELECT DISTINCT z.cname
FROM Product x, Product y, Company z
WHERE z.country = 'USA'
    AND x.category = 'gadget'
    AND y.category = 'photo'
    AND x.manufacturer = z.cname
    AND y.manufacturer = z.cname;

<table>
<thead>
<tr>
<th>Product</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>y</td>
</tr>
<tr>
<td>pname</td>
<td>category</td>
</tr>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>SingleTouch</td>
<td>photo</td>
</tr>
<tr>
<td>MultiTouch</td>
<td>Photo</td>
</tr>
<tr>
<td>GizmoWorks</td>
<td>USA</td>
</tr>
<tr>
<td>Hitachi</td>
<td>Japan</td>
</tr>
</tbody>
</table>
Self-Joins

```
SELECT DISTINCT z.cname
FROM Product x, Product y, Company z
WHERE z.country = 'USA'
    AND x.category = 'gadget'
    AND y.category = 'photo'
    AND x.manufacturer = z.cname
    AND y.manufacturer = z.cname;
```
### Self-Joins

```sql
SELECT DISTINCT z.cname
FROM Product x, Product y, Company z
WHERE z.country = 'USA'
    AND x.category = 'gadget'
    AND y.category = 'photo'
    AND x.manufacturer = z.cname
    AND y.manufacturer = z.cname;
```

<table>
<thead>
<tr>
<th>Product</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>y</td>
</tr>
<tr>
<td>z</td>
<td></td>
</tr>
<tr>
<td><strong>pname</strong></td>
<td><strong>cname</strong></td>
</tr>
<tr>
<td><strong>category</strong></td>
<td><strong>country</strong></td>
</tr>
<tr>
<td><strong>manufacturer</strong></td>
<td></td>
</tr>
</tbody>
</table>

- **Gizmo** gadget | **GizmoWorks**
- **SingleTouch** photo | **Hitachi**
- **MultiTouch** Photo | **GizmoWorks**

- **GizmoWorks** USA
- **Hitachi** Japan
Self-Joins

```
SELECT DISTINCT z.cname
FROM Product x, Product y, Company z
WHERE z.country = 'USA'
AND x.category = 'gadget'
AND y.category = 'photo'
AND x.manufacturer = z.cname
AND y.manufacturer = z.cname;
```

<table>
<thead>
<tr>
<th>Product</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>x</td>
<td>y</td>
</tr>
<tr>
<td>Gizmo</td>
<td>SingleTouch</td>
</tr>
<tr>
<td>gadget</td>
<td>photo</td>
</tr>
<tr>
<td>GizmoWorks</td>
<td>Hitachi</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>z</td>
</tr>
<tr>
<td>cname</td>
</tr>
<tr>
<td>country</td>
</tr>
<tr>
<td>GizmoWorks</td>
</tr>
<tr>
<td>Hitachi</td>
</tr>
</tbody>
</table>
Self-Joins

```
SELECT DISTINCT z.cname
FROM Product x, Product y, Company z
WHERE z.country = 'USA'
  AND x.category = 'gadget'
  AND y.category = 'photo'
  AND x.manufacturer = z.cname
  AND y.manufacturer = z.cname;
```
Self-Joins

```
SELECT DISTINCT z.cname
FROM Product x, Product y, Company z
WHERE z.country = 'USA'
AND x.category = 'gadget'
AND y.category = 'photo'
AND x.manufacturer = z.cname
AND y.manufacturer = z.cname;
```
Self-Joins

```
SELECT DISTINCT z.cname
FROM Product x, Product y, Company z
WHERE z.country = 'USA'
  AND x.category = 'gadget'
  AND y.category = 'photo'
  AND x.manufacturer = z.cname
  AND y.manufacturer = z.cname;
```
SELECT DISTINCT z.cname
FROM Product x, Product y, Company z
WHERE z.country = 'USA'
AND x.category = 'gadget'
AND y.category = 'photo'
AND x.manufacturer = z.cname
AND y.manufacturer = z.cname;
The join we have just seen is sometimes called an inner join
- Each row in the result **must come from both tables in the join**

Sometimes we want to include rows from only one of the two table: **outer join**
Inner Join

\[
\text{Employee}(id, \text{name}) \\
\text{Sales}(\text{employeeID}, \text{productID})
\]

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joe</td>
</tr>
<tr>
<td>2</td>
<td>Jack</td>
</tr>
<tr>
<td>3</td>
<td>Jill</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>employeeID</th>
<th>productID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>344</td>
</tr>
<tr>
<td>1</td>
<td>355</td>
</tr>
<tr>
<td>2</td>
<td>544</td>
</tr>
</tbody>
</table>

Retrieve employees and their sales
Inner Join

Employee(id, name)
Sales(employeeID, productID)

<table>
<thead>
<tr>
<th>Employee</th>
<th>Sales</th>
</tr>
</thead>
<tbody>
<tr>
<td>id</td>
<td>name</td>
</tr>
<tr>
<td>1</td>
<td>Joe</td>
</tr>
<tr>
<td>2</td>
<td>Jack</td>
</tr>
<tr>
<td>3</td>
<td>Jill</td>
</tr>
</tbody>
</table>

Retrieve employees and their sales

```sql
SELECT *
FROM Employee E, Sales S
WHERE E.id = S.employeeID
```
Inner Join

Employee(id, name)
Sales(employeeID, productID)

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joe</td>
</tr>
<tr>
<td>2</td>
<td>Jack</td>
</tr>
<tr>
<td>3</td>
<td>Jill</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>employeeID</th>
<th>productID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>344</td>
</tr>
<tr>
<td>1</td>
<td>355</td>
</tr>
<tr>
<td>2</td>
<td>544</td>
</tr>
</tbody>
</table>

Retrieve employees and their sales

```
SELECT * 
FROM Employee E, Sales S 
WHERE E.id = S.employeeID
```
**Inner Join**

Employee\((id, \text{name})\)
Sales\((\text{employeeID}, \text{productID})\)

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joe</td>
</tr>
<tr>
<td>2</td>
<td>Jack</td>
</tr>
<tr>
<td>3</td>
<td>Jill</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>employeeID</th>
<th>productID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>344</td>
</tr>
<tr>
<td>1</td>
<td>355</td>
</tr>
<tr>
<td>2</td>
<td>544</td>
</tr>
</tbody>
</table>

**Retrieve employees and their sales**

```
SELECT *
FROM Employee E, Sales S
WHERE E.id = S.employeeID
```

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>employeeID</th>
<th>productID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joe</td>
<td>1</td>
<td>344</td>
</tr>
<tr>
<td>1</td>
<td>Joe</td>
<td>1</td>
<td>355</td>
</tr>
<tr>
<td>2</td>
<td>Jack</td>
<td>2</td>
<td>544</td>
</tr>
</tbody>
</table>
Inner Join

Employee(id, name)
Sales(employeeID, productID)

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>employeeID</th>
<th>productID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joe</td>
<td>1</td>
<td>344</td>
</tr>
<tr>
<td>2</td>
<td>Jack</td>
<td>1</td>
<td>355</td>
</tr>
<tr>
<td>3</td>
<td>Jill</td>
<td>2</td>
<td>544</td>
</tr>
</tbody>
</table>

Retrieve employees and their sales

```
SELECT *
FROM Employee E
INNER JOIN Sales S
ON E.id = S.employeeID
```

Alternative syntax

Jill is missing

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>employeeID</th>
<th>productID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joe</td>
<td>1</td>
<td>344</td>
</tr>
<tr>
<td>1</td>
<td>Joe</td>
<td>1</td>
<td>355</td>
</tr>
<tr>
<td>2</td>
<td>Jack</td>
<td>2</td>
<td>544</td>
</tr>
</tbody>
</table>
### Outer Join

**Employee**

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joe</td>
</tr>
<tr>
<td>2</td>
<td>Jack</td>
</tr>
<tr>
<td>3</td>
<td>Jill</td>
</tr>
</tbody>
</table>

**Sales**

<table>
<thead>
<tr>
<th>employeeID</th>
<th>productID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>344</td>
</tr>
<tr>
<td>1</td>
<td>355</td>
</tr>
<tr>
<td>2</td>
<td>544</td>
</tr>
</tbody>
</table>

**Employee(id, name)**

**Sales(employeeID, productID)**

---

**Retrieve employees and their sales**

```sql
SELECT *
FROM Employee E
LEFT OUTER JOIN Sales S
ON E.id = S.employeeID
```

<table>
<thead>
<tr>
<th>id</th>
<th>name</th>
<th>employeeID</th>
<th>productID</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Joe</td>
<td>1</td>
<td>344</td>
</tr>
<tr>
<td>1</td>
<td>Joe</td>
<td>1</td>
<td>355</td>
</tr>
<tr>
<td>2</td>
<td>Jack</td>
<td>2</td>
<td>544</td>
</tr>
<tr>
<td>3</td>
<td>Jill</td>
<td>NULL</td>
<td>NULL</td>
</tr>
</tbody>
</table>

Jill is present
Product(name, category)
Purchase(prodName, store)

-- prodName is foreign key

```
SELECT Product.name, Purchase.store
FROM Product LEFT OUTER JOIN Purchase ON
Product.name = Purchase.prodName
```
<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

```
SELECT Product.name, Purchase.store
FROM Product JOIN Purchase ON
  Product.name = Purchase.prodName
```
### SQL Query

```
SELECT Product.name, Purchase.store
FROM Product JOIN Purchase ON Product.name = Purchase.prodName
```

### Product Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

### Purchase Table

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>
### SELECT Query

```sql
SELECT Product.name, Purchase.store
FROM Product JOIN Purchase ON Product.name = Purchase.prodName
```

### Product Table
<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

### Purchase Table
<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

### Output Table
<table>
<thead>
<tr>
<th>Name</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
</tbody>
</table>
SELECT Product.name, Purchase.store
FROM Product JOIN Purchase
ON Product.name = Purchase.prodName


```sql
SELECT Product.name, Purchase.store
FROM Product JOIN Purchase ON Product.name = Purchase.prodName
```

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Name</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
</tbody>
</table>
### Product

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

### Purchase

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

### Output

<table>
<thead>
<tr>
<th>Name</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
</tbody>
</table>
```sql
SELECT Product.name, Purchase.store
FROM Product JOIN Purchase
ON Product.name = Purchase.prodName
```

| Product | | |
|---------|----------------|
| Name    | Category       |
| Gizmo   | gadget         |
| Camera  | Photo          |
| OneClick| Photo          |

| Purchase | | |
|----------|----------------|
| ProdName | Store          |
| Gizmo    | Wiz            |
| Camera   | Ritz           |

| Output | | |
|--------|----------------|
| Name   | Store          |
| Gizmo  | Wiz            |
| Camera | Ritz           |
### Product

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

### Purchase

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

### Output

<table>
<thead>
<tr>
<th>Name</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

**SQL Query**: 
```
SELECT Product.name, Purchase.store
FROM Product JOIN Purchase ON Product.name = Purchase.prodName
```
### Product

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

### Purchase

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

### SELECT

```sql
SELECT Product.name, Purchase.store
FROM Product JOIN Purchase ON Product.name = Purchase.prodName
```
### Product

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

### Purchase

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

### Output

<table>
<thead>
<tr>
<th>Name</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

The SQL query is:

```
SELECT Product.name, Purchase.store
FROM Product
LEFT OUTER JOIN Purchase ON Product.name = Purchase.prodName
```
```
SELECT Product.name, Purchase.store
FROM Product LEFT OUTER JOIN Purchase ON
    Product.name = Purchase.prodName
```
SELECT Product.name, Purchase.store
FROM Product FULL OUTER JOIN Purchase ON Product.name = Purchase.prodName

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

Output
### SQL Query

```sql
SELECT Product.name, Purchase.store
FROM Product FULL OUTER JOIN Purchase
ON Product.name = Purchase.prodName
```

### Product Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

### Purchase Table

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

### Output Table

<table>
<thead>
<tr>
<th>Name</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
<tr>
<td>OneClick</td>
<td>NULL</td>
</tr>
</tbody>
</table>
### Product

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

### Purchase

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

### Output

<table>
<thead>
<tr>
<th>Name</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
<tr>
<td>OneClick</td>
<td>NULL</td>
</tr>
</tbody>
</table>
### Product

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

### Purchase

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
<tr>
<td>Phone</td>
<td>Foo</td>
</tr>
</tbody>
</table>

**Output**

```sql
SELECT Product.name, Purchase.store
FROM Product
FULL OUTER JOIN Purchase
ON Product.name = Purchase.prodName
```
### Product

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

### Purchase

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
</tbody>
</table>

### Output

<table>
<thead>
<tr>
<th>Name</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
<tr>
<td>OneClick</td>
<td>NULL</td>
</tr>
</tbody>
</table>

**SELECT** Product.name, Purchase.store  
**FROM** Product **FULL OUTER JOIN** Purchase **ON**  
Product.name = Purchase.prodName
### Product

<table>
<thead>
<tr>
<th>Name</th>
<th>Category</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
</tr>
</tbody>
</table>

### Purchase

<table>
<thead>
<tr>
<th>ProdName</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
<tr>
<td>Phone</td>
<td>Foo</td>
</tr>
</tbody>
</table>

### Output

<table>
<thead>
<tr>
<th>Name</th>
<th>Store</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>Wiz</td>
</tr>
<tr>
<td>Camera</td>
<td>Ritz</td>
</tr>
<tr>
<td>Camera</td>
<td>Wiz</td>
</tr>
<tr>
<td>OneClick</td>
<td>NULL</td>
</tr>
<tr>
<td>NULL</td>
<td>Foo</td>
</tr>
</tbody>
</table>
Outer Joins

- **Left outer join:**
  ```sql
tableA (LEFT/RIGHT/FULL) OUTER JOIN tableB ON p
  ```

- **Right outer join:**
  - Include tuples from `tableB` even if no match

- **Full outer join:**
  - Include tuples from both even if no match

- **In all cases:**
  - Patch tuples without matches using `NULL`
Simple Aggregations

Five basic aggregate operations in SQL

```sql
select count(*) from Purchase
select sum(quantity) from Purchase
select avg(price) from Purchase
select max(quantity) from Purchase
select min(quantity) from Purchase
```

Except count, all aggregations apply to a single attribute
Aggregates and NULL Values

Purchase(id, name, quantity, amt, month)

Null values are not used in aggregates

```sql
insert into Purchase
values(1, 'gadget', NULL, NULL, 'april')
```

Try the following

```sql
select count(*) from Purchase
select count(quantity) from Purchase
select sum(quantity) from Purchase

select count(*)
from Purchase
where quantity is not null;
```
Aggregates and NULL Values

Purchase(id, name, quantity, amt, month)

Null values are not used in aggregates

```
insert into Purchase
values(1, 'gadget', NULL, NULL, 'april')
```

Try the following

```
select count(*) from Purchase     = 1
select count(quantity) from Purchase = 0
select sum(quantity) from Purchase  = NULL

select count(*)
from Purchase
where quantity is not null;
```
COUNT applies to duplicates, unless otherwise stated:

\[
\text{SELECT} \quad \text{count(product)} \\
\text{FROM} \quad \text{Purchase} \\
\text{WHERE} \quad \text{price} > 4.99
\]

same as \text{count(\*)} if no nulls

We probably want:

\[
\text{SELECT} \quad \text{count(DISTINCT product)} \\
\text{FROM} \quad \text{Purchase} \\
\text{WHERE} \quad \text{price} > 4.99
\]
More Examples

```
SELECT  Sum(P.price * P.quantity)
FROM    Purchase as P

SELECT  Sum(P.price * P.quantity)
FROM    Purchase as P
WHERE   P.product = 'bagel'
```

What do they mean?
Aggregation Semantics

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

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UserID</th>
<th>Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Charger</td>
</tr>
<tr>
<td>567</td>
<td>Civic</td>
</tr>
<tr>
<td>567</td>
<td>Pinto</td>
</tr>
</tbody>
</table>
Aggregation Semantics

SELECT AVG(P.Salary)
FROM Payroll AS P, Regist AS R
WHERE P.UserID = R.UserID;

What am I aggregating over in a SELECT-FROM-WHERE query?
Aggregation Semantics

SELECT AVG(P.Salary) 
FROM Payroll AS P, Regist AS R 
WHERE P.UserID = R.UserID;

What am I aggregating over in a SELECT-FROM-WHERE query?

Answer:
The resulting tuples AFTER the join

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UserID</th>
<th>Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Charger</td>
</tr>
<tr>
<td>567</td>
<td>Civic</td>
</tr>
<tr>
<td>567</td>
<td>Pinto</td>
</tr>
</tbody>
</table>
Aggregation Semantics

\[
\text{SELECT } \text{AVG(P.Salary)} \\
\text{FROM Payroll AS P, Regist AS R} \\
\text{WHERE P.UserID = R.UserID;}
\]

<table>
<thead>
<tr>
<th>P.UserID</th>
<th>P.Name</th>
<th>P.Job</th>
<th>P.Salary</th>
<th>R.UserID</th>
<th>R.Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
<td>123</td>
<td>Charger</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
<td>567</td>
<td>Civic</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
<td>567</td>
<td>Pinto</td>
</tr>
</tbody>
</table>

\[P.UserID = R.UserID\]

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UserID</th>
<th>Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Charger</td>
</tr>
<tr>
<td>567</td>
<td>Civic</td>
</tr>
<tr>
<td>567</td>
<td>Pinto</td>
</tr>
</tbody>
</table>
SELECT AVG(P.Salary) 
FROM Payroll AS P, Regist AS R 
WHERE P.UserID = R.UserID;

\[ Y_{AVG(P.Salary)} \]

\[ \bowtie \]

\[ P.UserID = R.UserID \]

<table>
<thead>
<tr>
<th>P.UserID</th>
<th>P.Name</th>
<th>P.Job</th>
<th>P.Salary</th>
<th>R.UserID</th>
<th>R.Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
<td>123</td>
<td>Charger</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
<td>567</td>
<td>Civic</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
<td>567</td>
<td>Pinto</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UserID</th>
<th>Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Charger</td>
</tr>
<tr>
<td>567</td>
<td>Civic</td>
</tr>
<tr>
<td>567</td>
<td>Pinto</td>
</tr>
</tbody>
</table>
### Aggregation Semantics

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

\[ \gamma \text{AVG}(P.Salary) \]

<table>
<thead>
<tr>
<th>P.UserID</th>
<th>P.Name</th>
<th>P.Job</th>
<th>P.Salary</th>
<th>R.UserID</th>
<th>R.Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
<td>123</td>
<td>Charger</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
<td>567</td>
<td>Civic</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
<td>567</td>
<td>Pinto</td>
</tr>
</tbody>
</table>

\[ \Join_{P.UserID=R.UserID} \]

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>UserID</th>
<th>Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Charger</td>
</tr>
<tr>
<td>567</td>
<td>Civic</td>
</tr>
<tr>
<td>567</td>
<td>Pinto</td>
</tr>
</tbody>
</table>
### Aggregation Semantics

**SQL Query:**

```sql
SELECT AVG(P.Salary)
FROM Payroll AS P, Register AS R
WHERE P.UserID = R.UserID;
```

- **Result:**
  - $\overline{76666}$

**Table 1:**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Name</th>
<th>Job</th>
<th>Salary</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Jack</td>
<td>TA</td>
<td>50000</td>
</tr>
<tr>
<td>345</td>
<td>Allison</td>
<td>TA</td>
<td>60000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>567</td>
<td>Magda</td>
<td>Prof</td>
<td>90000</td>
</tr>
<tr>
<td>789</td>
<td>Dan</td>
<td>Prof</td>
<td>100000</td>
</tr>
</tbody>
</table>

**Table 2:**

<table>
<thead>
<tr>
<th>UserID</th>
<th>Car</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>Charger</td>
</tr>
<tr>
<td>345</td>
<td>Civic</td>
</tr>
<tr>
<td>567</td>
<td>Pinto</td>
</tr>
<tr>
<td>567</td>
<td>Pinto</td>
</tr>
</tbody>
</table>

**Note:** $P.UserID = R.UserID$
Aggregation Semantics

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

What am I aggregating over in a SELECT-FROM-WHERE query?

Answer:
The resulting tuples AFTER the join
What am I aggregating over in a SELECT-FROM-WHERE query?

Answer:
The resulting tuples AFTER the join

There is an implicit “order of operations”
Order of operations

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

FROM -> WHERE -> SELECT