Database Management Systems

Lecture 3: SQL Basics
(Inner) Joins

```
SELECT x1.a1, x2.a2, ... xm.am
FROM R1 as x1, R2 as x2, ... Rm as xm
WHERE Cond
```

for t1 in x1:
  for t2 in x2:
    ...
    for tm in xm:
      if Cond(t1, t2, ...):
        output(t1.a1, t2.a2, ... tm.am)
(Inner) Joins

```plaintext
SELECT x1.a1, x2.a2, ... xm.am
FROM R1 as x1, R2 as x2, ... Rm as xm
WHERE Cond

for t1 in x1:
    for t2 in x2:
        ...
        for tm in xm:
            if Cond(t1, t2, ...):
                output(t1.a1, t2.a2, ... tm.am)
```
(Inner) joins

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

```
SELECT DISTINCT cname
FROM Product, Company
WHERE country = 'USA' AND category = 'gadget' AND manufacturer = cname
```
(Inner) joins

```sql
SELECT DISTINCT cname
FROM Product, Company
WHERE country = 'USA' AND category = 'gadget' AND manufacturer = cname
```
(Inner) joins

```sql
SELECT DISTINCT cname
FROM Product, Company
WHERE country = 'USA' AND category = 'gadget' AND manufacturer = cname
```

**Product**

<table>
<thead>
<tr>
<th>pname</th>
<th>category</th>
<th>manufacturer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
<td>GizmoWorks</td>
</tr>
<tr>
<td>Camera</td>
<td>Photo</td>
<td>Hitachi</td>
</tr>
<tr>
<td>OneClick</td>
<td>Photo</td>
<td>Hitachi</td>
</tr>
</tbody>
</table>

**Company**

<table>
<thead>
<tr>
<th>cname</th>
<th>country</th>
</tr>
</thead>
<tbody>
<tr>
<td>GizmoWorks</td>
<td>USA</td>
</tr>
<tr>
<td>Canon</td>
<td>Japan</td>
</tr>
<tr>
<td>Hitachi</td>
<td>Japan</td>
</tr>
</tbody>
</table>
(Inner) joins

```sql
SELECT DISTINCT cname
FROM Product, Company
WHERE country = 'USA' AND category = 'gadget' AND manufacturer = cname
```
(Inner) joins

```
SELECT DISTINCT cname
FROM Product, Company
WHERE country = 'USA' AND category = 'gadget' AND manufacturer = cname
```
(Inner) joins

```
SELECT DISTINCT cname
FROM Product, Company
WHERE country = 'USA' AND category = 'gadget' AND manufacturer = cname
```
#### (Inner) joins

```
SELECT DISTINCT cname
FROM Product, Company
WHERE country = 'USA' AND category = 'gadget' AND manufacturer = cname
```

<table>
<thead>
<tr>
<th>Product</th>
<th>Company</th>
</tr>
</thead>
<tbody>
<tr>
<td>pname</td>
<td>cname</td>
</tr>
<tr>
<td>category</td>
<td>country</td>
</tr>
<tr>
<td>Gizmo</td>
<td>GizmoWorks</td>
</tr>
<tr>
<td>Camera</td>
<td>Canon</td>
</tr>
<tr>
<td>OneClick</td>
<td>Hitachi</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
</tbody>
</table>
### (Inner) joins

**SELECT** DISTINCT `cname`  
**FROM** `Product`, `Company`  
**WHERE** `country` = 'USA' **AND** `category` = 'gadget' **AND** `manufacturer` = `cname`

**SELECT** DISTINCT `cname`  
**FROM** `Product` JOIN `Company`  
**ON** `country` = 'USA' **AND** `category` = 'gadget' **AND** `manufacturer` = `cname`
### Tuple variables

**SELECT DISTINCT** `cname`

**FROM** Product P1, Product P2, Company

**WHERE** country = 'USA' AND P1.category = 'gadget' AND P2.category = 'photo' AND P1.manufacturer = `cname` AND P2.manufacturer = `cname`;

---

<table>
<thead>
<tr>
<th>P1 (Product)</th>
<th>P2 (Product)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>pname</strong></td>
<td><strong>pname</strong></td>
</tr>
<tr>
<td>Gizmo</td>
<td>Gizmo2</td>
</tr>
<tr>
<td>Gizmo2</td>
<td>Gizmo</td>
</tr>
<tr>
<td>category</td>
<td>category</td>
</tr>
<tr>
<td>gadget</td>
<td>photo</td>
</tr>
<tr>
<td>gadget</td>
<td>Gadget</td>
</tr>
<tr>
<td>manufacturer</td>
<td>manufacturer</td>
</tr>
<tr>
<td>GizmoWorks</td>
<td>GizmoWorks</td>
</tr>
<tr>
<td>GizmoWorks</td>
<td>GizmoWorks</td>
</tr>
</tbody>
</table>

---

<table>
<thead>
<tr>
<th>Company</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>cname</strong></td>
<td><strong>country</strong></td>
</tr>
<tr>
<td>GizmoWorks</td>
<td>USA</td>
</tr>
</tbody>
</table>

---
Tuple variables

```
SELECT DISTINCT cname
FROM Product P1, Product P2, Company
WHERE country = 'USA' AND P1.category = 'gadget' AND P2.category = 'photo' AND P1.manufacturer = P2.manufacturer = cname;
```
Tuple variables

SELECT DISTINCT cname
FROM Product P1, Product P2, Company
WHERE country = 'USA' AND P1.category = 'gadget' AND P2.category = 'photo' AND P1.manufacturer = cname AND P2.manufacturer = cname;

<table>
<thead>
<tr>
<th>P1 (Product)</th>
<th>P2 (Product)</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>pname</strong></td>
<td><strong>pname</strong></td>
</tr>
<tr>
<td><strong>category</strong></td>
<td><strong>category</strong></td>
</tr>
<tr>
<td><strong>manufacturer</strong></td>
<td><strong>manufacturer</strong></td>
</tr>
<tr>
<td>Gizmo</td>
<td>Gizmo2</td>
</tr>
<tr>
<td>gadget</td>
<td>photo</td>
</tr>
<tr>
<td>GizmoWorks</td>
<td>GizmoWorks</td>
</tr>
<tr>
<td>Gizmo2</td>
<td>Gizmo</td>
</tr>
<tr>
<td>photo</td>
<td>Gadget</td>
</tr>
<tr>
<td>GizmoWorks</td>
<td>GizmoWorks</td>
</tr>
</tbody>
</table>

Company

<table>
<thead>
<tr>
<th><strong>cname</strong></th>
<th><strong>country</strong></th>
</tr>
</thead>
<tbody>
<tr>
<td>GizmoWorks</td>
<td>USA</td>
</tr>
</tbody>
</table>
Outer joins

Product(`name`, `category`)  
Purchase(`prodName`, `store`) -- `prodName` is foreign key

An “inner join”:

```
SELECT Product.name, Purchase.store
FROM Product, Purchase
WHERE Product.name = Purchase.prodName
```

Same as:

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

But some Products are not listed! Why?
Outer joins

Product(name, category)
Purchase(prodName, store) -- prodName is foreign key

If we want to include products that never sold, then we need an “outerjoin”:

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

```sql
SELECT Product.name, Purchase.store
FROM Product
LEFT 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>

### Name and Store 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>
Outer Joins

• Left outer join:
  – Include the left tuple even if there’s no match

• Right outer join:
  – Include the right tuple even if there’s no match

• Full outer join:
  – Include both left and right tuples even if there’s no match
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

Null values are not used in aggregates

```sql
insert into Purchase
values(12, 'gadget', NULL, NULL, 'april')
```

Let’s try the following

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

select sum(quantity) from Purchase

select sum(quantity)
from Purchase
where quantity is not null;
```
Counting Duplicates

COUNT applies to duplicates, unless otherwise stated:

```
SELECT Count(product)  
FROM    Purchase  
WHERE   price > 4.99
```

same as Count(*) if no nulls

We probably want:

```
SELECT Count(DISTINCT product)  
FROM    Purchase  
WHERE   price > 4.99
```
More Examples

\[
\begin{align*}
\text{SELECT} & \quad \text{Sum(price} \times \text{quantity}) \\
\text{FROM} & \quad \text{Purchase}
\end{align*}
\]

\[
\begin{align*}
\text{SELECT} & \quad \text{Sum(price} \times \text{quantity}) \\
\text{FROM} & \quad \text{Purchase} \\
\text{WHERE} & \quad \text{product} = \text{‘bagel’}
\end{align*}
\]

What do they mean?
### Simple Aggregations

**Purchase**

<table>
<thead>
<tr>
<th>Product</th>
<th>Price</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bagel</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>Bagel</td>
<td>1.50</td>
<td>20</td>
</tr>
<tr>
<td>Banana</td>
<td>0.5</td>
<td>50</td>
</tr>
<tr>
<td>Banana</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>Banana</td>
<td>4</td>
<td>10</td>
</tr>
</tbody>
</table>

```sql
SELECT Sum(price * quantity) FROM Purchase WHERE product = 'Bagel'
```

90 \(= 60+30\)
### Simple Aggregations

**Purchase**

<table>
<thead>
<tr>
<th>Product</th>
<th>Price</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bagel</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>Bagel</td>
<td>1.50</td>
<td>20</td>
</tr>
<tr>
<td>Banana</td>
<td>0.5</td>
<td>50</td>
</tr>
<tr>
<td>Banana</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>Banana</td>
<td>4</td>
<td>10</td>
</tr>
</tbody>
</table>

```
SELECT Sum(price * quantity) FROM Purchase WHERE product = 'Bagel'
```

90 (= 60 + 30)
Grouping and Aggregation

Purchase(product, price, quantity)

Find total quantities for all sales over $1, by product.

```
SELECT product, Sum(quantity) AS TotalSales
FROM Purchase
WHERE price > 1
GROUP BY product
```
Grouping and Aggregation

1. Compute the FROM and WHERE clauses.

2. Group by the attributes in the GROUPBY

3. Compute the SELECT clause: grouped attributes and aggregates.
1&2. FROM-WHERE-GROUPBY

<table>
<thead>
<tr>
<th>Product</th>
<th>Price</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bagel</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>Bagel</td>
<td>1.50</td>
<td>20</td>
</tr>
<tr>
<td>Banana</td>
<td>0.5</td>
<td>50</td>
</tr>
<tr>
<td>Banana</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>Banana</td>
<td>4</td>
<td>10</td>
</tr>
</tbody>
</table>

WHERE price > 1
3. SELECT

```
SELECT product, SUM(quantity) AS TotalSales
FROM Purchase
WHERE price > 1
GROUP BY product
```

<table>
<thead>
<tr>
<th>Product</th>
<th>Price</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bagel</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>Bagel</td>
<td>1.50</td>
<td>20</td>
</tr>
<tr>
<td>Banana</td>
<td>0.5</td>
<td>50</td>
</tr>
<tr>
<td>Banana</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>Banana</td>
<td>4</td>
<td>10</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Product</th>
<th>TotalSales</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bagel</td>
<td>40</td>
</tr>
<tr>
<td>Banana</td>
<td>20</td>
</tr>
</tbody>
</table>
Other Examples

Compare these two queries:

**Example 1**
```
SELECT product, count(*)
FROM Purchase
GROUP BY product
```

**Example 2**
```
SELECT month, count(*)
FROM Purchase
GROUP BY month
```

**Example 3**
```
SELECT product,
    sum(quantity) AS SumQuantity,
    max(price) AS MaxPrice
FROM Purchase
GROUP BY product
```

What does it mean?
Need to be Careful…

```
SELECT product, max(quantity)
FROM Purchase
GROUP BY product
```

```
SELECT product, quantity
FROM Purchase
GROUP BY product
```

sqlite is WRONG on this query. (why?)

<table>
<thead>
<tr>
<th>Product</th>
<th>Price</th>
<th>Quantity</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bagel</td>
<td>3</td>
<td>20</td>
</tr>
<tr>
<td>Bagel</td>
<td>1.50</td>
<td>20</td>
</tr>
<tr>
<td>Banana</td>
<td>0.5</td>
<td>50</td>
</tr>
<tr>
<td>Banana</td>
<td>2</td>
<td>10</td>
</tr>
<tr>
<td>Banana</td>
<td>4</td>
<td>10</td>
</tr>
</tbody>
</table>

Advanced DBMS (e.g. SQL Server) gives an error