Announcements

• Should now have seats for all registered

Outline

• Inner joins (6.2, review)
• Outer joins (6.3.8)
• Aggregations (6.4.3 – 6.4.6)

UNIQUE

• PRIMARY KEY adds implicit "NOT NULL" constraint while UNIQUE does not
  – you would have to add this explicitly for UNIQUE:
    CREATE TABLE Company(
      name VARCHAR(20) NOT NULL, ...
      UNIQUE (name));
• You almost always want to do this (in real schemas)
  – SQL Server behaves strangely with NULL & UNIQUE
  – otherwise, think through NULL for every query
  – you can remove the NOT NULL constraint later

(Inner) Joins

SELECT a1, a2, ..., an
FROM R1, R2, ..., Rm
WHERE Cond

for t1 in R1:
  for t2 in R2:
    ...
  for tm in Rm:
    if Cond(t1.a1, t1.a2, ...):
      output(t1.a1, t1.a2, ..., tm.an)

(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
```

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>

Not output because country != 'USA' (also cname != manufacturer)

CSE 414 - Fall 2017

---

(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>

Not output because category != 'gadget' (and ...)

CSE 414 - Fall 2017

---

(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>

Not output because category != 'gadget'

CSE 414 - Fall 2017

---

(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>

Not output because category != 'gadget'

CSE 414 - Fall 2017

---

(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>

Not output because category != 'gadget'

CSE 414 - Fall 2017
(Inner) joins

\[
\text{SELECT DISTINCT \textit{cname}} \\
\text{FROM Product, Company} \\
\text{WHERE country = 'USA' AND category = 'gadget' AND} \\
\text{manufacturer = \textit{cname}}
\]

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>

Not output because category \(\neq\) ‘gadget’

(CSE 414 - Fall 2017)

13

(Inner) joins

\[
\text{SELECT DISTINCT \textit{cname}} \\
\text{FROM Product, Company} \\
\text{WHERE country = 'USA' AND category = 'gadget' AND} \\
\text{manufacturer = \textit{cname}}
\]

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>

Not output because category \(\neq\) ‘gadget’ (with any Company)

(CSE 414 - Fall 2017)

14

(Inner) joins

\[
\text{SELECT DISTINCT \textit{cname}} \\
\text{FROM Product, Company} \\
\text{WHERE country = 'USA' AND category = 'gadget' AND} \\
\text{manufacturer = \textit{cname}}
\]

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>

restrict to category = ‘gadget’

(CSE 414 - Fall 2017)

15

(Inner) joins

\[
\text{SELECT DISTINCT \textit{cname}} \\
\text{FROM Product, Company} \\
\text{WHERE country = 'USA' AND category = 'gadget' AND} \\
\text{manufacturer = \textit{cname}}
\]

Product (where category = 'gadget')

<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>

restrict to country = ‘USA’

(CSE 414 - Fall 2017)

16

(Inner) joins

\[
\text{SELECT DISTINCT \textit{cname}} \\
\text{FROM Product, Company} \\
\text{WHERE country = 'USA' AND category = 'gadget' AND} \\
\text{manufacturer = \textit{cname}}
\]

Product (where category = 'gadget')

<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 (where country = 'USA')

<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>

Now only one combination to consider

(Query optimizers do this too.)

(CSE 414 - Fall 2017)

17

(Inner) joins

\[
\text{SELECT DISTINCT \textit{cname}} \\
\text{FROM Product, Company} \\
\text{WHERE country = 'USA' AND category = 'gadget' AND} \\
\text{manufacturer = \textit{cname}}
\]

Product (where category = 'gadget')

<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>
</tbody>
</table>

Company (where country = 'USA')

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

Alternative syntax:

\[
\text{SELECT DISTINCT \textit{cname}} \\
\text{FROM Product JOIN Company ON} \\
\text{country = 'USA' AND category = 'gadget' AND} \\
\text{manufacturer = \textit{cname}}
\]

Emphasizes that the predicate is part of the join.

(CSE 414 - Fall 2017)

18
Self-Joins and Tuple Variables

- **Ex:** find companies that manufacture both products in the 'gadgets' category and in the 'photo' category
- Just joining `Company` with `Product` is insufficient: need to join `Company` with `Product` with `Product`

```
FROM Company, Product, Product
```
- When a relation occurs twice in the FROM clause we call it a self-join; in that case every column name in `Product` is ambiguous (why?)
  - are you referring to the tuple in the 2nd or 3rd loop?

**Name Conflicts**

- When a name is ambiguous, qualify it:
  ```
  WHERE Company.name = Product.name AND ...
  ```
- For self-join, we need to distinguish tables:
  ```
  FROM Product x, Product y, Company
  ```
- These new names are called "tuple variables"
  - can think of as name for the variable of each loop
  - can also write "Company AS C" etc.
  - can make SQL query shorter: `C.name` vs. `Company.name`

```
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;
```

Not output because `y.category != 'photo'`

restrict to country = 'USA'

we used `cname / pname` to avoid this problem
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>pname</th>
<th>category</th>
<th>manufacturer</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gizmo</td>
<td>gadget</td>
<td>GizmoWorks</td>
</tr>
<tr>
<td>SingleTouch</td>
<td>photo</td>
<td>Hitachi</td>
</tr>
<tr>
<td>MultiTouch</td>
<td>photo</td>
<td>GizmoWorks</td>
</tr>
</tbody>
</table>

Not output because y.manufacturer != z.cname

Outer joins

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

Or equivalently:

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

But some Products may not be not listed. Why?

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

If we want to include products that never sold, then we need an "outer join":

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

Outer joins
### 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

  (Also something called a UNION JOIN, though it’s rarely used.)

  (Actually, all of these are used much more rarely than inner joins.)
Outer Joins Example

See lec04-sql-outer-joins.sql...

Aggregation in SQL

```sql
>sqlite3 lecture04
sqlite> create table Purchase(
    pid int primary key,
    product text,
    price float,
    quantity int,
    month varchar(15));
sqlite> -- download data.txt
sqlite> .import lec04-data.txt Purchase
```

Other DBMSs have other ways of importing data

Comment about SQLite

• One cannot load NULL values such that they are actually loaded as null values

• So we need to use two steps:
  – Load null values using some type of special value
  – Update the special values to actual null values

```sql
update Purchase
set price = null
where price = '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 value

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
```

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
-- NULL is counted in count(*)
select count(quantity) from Purchase
-- NULL is ignored in count(quantity)
select sum(quantity)
from Purchase
where quantity is not null;
-- "is not null" is redundant
```
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

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

What do they mean?

More Examples

```
SELECT Sum(price * quantity) / count(*) FROM Purchase WHERE product = 'bagel'
```

Lesson: disallow NULLS unless you need to handle them