Query Results

- Query results are either a table or a value
  - E.g. `select * from products` or `select count(*) from products`
- Query results can be used in places where a table/value can be used
  - A value can also be considered as a table with only one row and one column

Subquery Example 1

- Find the most expensive products
  
  ```
  select * from products where price =
  ( select max(price) from products );
  ```

Subquery Example 2

- List the ID’s of the products sold on 2007/6/1
  
  ```
  select d.product_id from order_details d,
  (select * from orders
  where date_ordered = '2007-06-01') as o
  where d.order_id = o.id;
  ```

More Subquery Examples

- List the ID’s of the products sold on 2007/6/1 (Using IN)
- List the descriptions of the products sold on 2007/6/1
- Find the CPU products that are cheaper than Intel Pentium D

Set Operations

- **Union**
  - \{1,2,3\} \cup \{4,5,6\} = \{1,2,3,4,5,6\}
- **Intersect**
  - \{1,2,3\} \cap \{2,3,4\} = \{2,3\}
- **Difference**
  - \{1,2,3\} - \{2,3,4\} = \{1\}
Set Operations in Database - UNION

List all the zip codes from both vendors and customers table.

<table>
<thead>
<tr>
<th>vendor</th>
<th>zip</th>
</tr>
</thead>
<tbody>
<tr>
<td>Intel</td>
<td>91111</td>
</tr>
<tr>
<td>AMD</td>
<td>92222</td>
</tr>
<tr>
<td>Seagate</td>
<td>74444</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>customer</th>
<th>zip</th>
</tr>
</thead>
<tbody>
<tr>
<td>John</td>
<td>91111</td>
</tr>
<tr>
<td>Jane</td>
<td>91111</td>
</tr>
<tr>
<td>Tom</td>
<td>92222</td>
</tr>
</tbody>
</table>

About UNION

- Combine result tables of SELECT statements
- The result tables must have the same number of columns
- The corresponding columns must have the same (or at least "compatible") type
- Duplicates in union results
  - UNION - automatically remove duplicates
  - UNION ALL - keep duplicates

INTERSECT and DIFFERENCE

- Same syntax as UNION
- MySQL 5.0 does not support INTERSECT and DIFFERENCE

Correlated Subquery

- The inner query uses column(s) from the outer query
  - E.g. find the products that are cheaper than the average price of their category

```sql
SELECT * FROM products p WHERE p.price < (SELECT avg(price) FROM products p2 WHERE p2.category = p.category);
```

How Correlated Subqueries Work

Using ANY, ALL and EXISTS

- Find the CPU products that are more expensive than all HD products
- Find the HD products that are more expensive than at least one CPU product
- Find the customers who live with another customer
Summary

- Syntax
  - Subquery, set operations
  - ANY, ALL, EXISTS
- A different way of thinking (vs. Joins)