MULTI-TABLE QUERIES

Reading: Chapter 4 (pages 96-105)

Objectives

- Write queries for a database with a one-to-many relationship

Multi-table queries are similar to your single table queries – you just need to first join data from multiple tables to form a single table.
Queries in a DB with Multiple Tables

• If related data exists in multiple tables
  1. Join the tables to form a new table
  2. Execute a SQL query on the joined table

Join

• Create a new table from two existing tables by matching on a common column

```
SELECT * FROM Shares, Nations
WHERE Shares.nationID = Nations.ID;
```
Relationships Summary

- 1:m relationship is provided with a primary key/foreign key pair
- Each foreign key must have a matching primary key
- Every primary key must be non-null
- Key pair must have the same type
- When modeling (ERD), we omit the foreign key attribute since it can be inferred from the relationship arrow

A common, but hard-to-find error when you are first starting

Join Example

SELECT nationName, shareName, price, quantity, exchangeRate, price*quantity*exchangeRate AS stockValue
FROM Shares, Nations
WHERE Shares.nationID = Nations.ID
ORDER BY nationName, shareName;

To avoid name confusion, it is better to qualify the attribute name

You reference does not need to use the table name if the attribute name is unique

Query is performed on the joined table
Data Repetition Resulting From JOIN

- JOIN operation often results in data repetition
- GROUP BY clause is often used to summarize data

GROUP BY Example

- Report by nation the total value of stock holdings

```sql
SELECT Nations.nationName, SUM(price*quantity*exchangeRate) AS shareValue
FROM Shares, Nations
WHERE Shares.nationID=Nations.ID
GROUP BY Nations.nationName;
```
JOIN Access Design View

• Number of stocks and their total value by nation

Are We on Track?

• Report the total dividend payment by nation
• Report in British pounds
• Remember total dividends is shares multiplied by dividend per share
Were We on Track?

• Access SQL view

```
SELECT Nations.nationName, 
SUM(dividend*quantity*exchangeRate) 
AS totalDividends 
FROM Nations INNER JOIN Shares 
ON Nations.ID = Shares.nationID 
GROUP BY Nations.nationName;
```

JOIN Notation

• SQL provides two syntactic approaches to specifying a join
  • Explicit join
  • Implicit join

FROM Nations INNER JOIN Shares 
ON Nations.ID = Shares.nationID

FROM Shares, Nations 
WHERE Shares.nationID=Nations.ID
HAVING Example

- Report the total value of stocks (in pounds) for nations with two or more listed stocks

```
SELECT Nations.nationName, 
SUM(price*quantity*exchangeRate) 
AS marketCapitalization 
FROM Shares, Nations 
WHERE Shares.nationID=Nations.ID 
GROUP BY Nations.nationName 
HAVING COUNT(*)>=2;
```

Subquery Example

- Report the names of all Australian stocks
- Note that the country name is in the Nations table and the stock name is in the Shares table

```
SELECT shareName 
FROM Shares 
WHERE Shares.nationID IN 
(SELECT ID FROM Nations 
WHERE nationName = 'Australia') ;
```

Subquery returns a list (of size 1) of nation IDs
Alternate to Subquery

- The following SQL will obtain the same result, but the subquery approach might be more efficient

```
SELECT Shares.shareName
FROM Nations INNER JOIN Shares ON Nations.ID = Shares.nationID
WHERE Nations.nationName = 'Australia';
```

This approach might be more intuitive

Are We on Track?

- Report the average yield by nation (yield is defined as dividend divided by price, and is reported as a percentage)

<table>
<thead>
<tr>
<th>nationName</th>
<th>Yield</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australia</td>
<td>6.22%</td>
</tr>
<tr>
<td>India</td>
<td>3.91%</td>
</tr>
<tr>
<td>United Kingdom</td>
<td>7.53%</td>
</tr>
<tr>
<td>United States</td>
<td>5.18%</td>
</tr>
</tbody>
</table>
Were We on Track?

- SQL view

```
SELECT Nations.nationName, AVG(dividend/price) AS Yield
FROM Shares, Nations
WHERE Shares.nationID=Nations.ID
GROUP BY Nations.nationName;
```

Specify column formatting in MS Access Design View

Views - Virtual Tables

- An imaginary table constructed by the DBMS when required
- Only the definition of the view is stored, not the result

Not readily available in Access, so we will defer most of this until we get to MySQL
HW Assignment

1. For the previous Track, compute the weighted average (instead of the average) average yield by nation (yield is defined as dividend divided by price, and is reported as a percentage). The weighted average includes the quantity of shares as a weight
2. Report the minimum and maximum yield for each nation
3. Report the nations where the average yield of the shares exceeds the average yield of all shares

Did You Meet the Objectives?

- Write queries for a database with a one-to-many relationship