VBA and Databases
(see Chapter 14)

Kipp Martin

February 29, 2012
Lecture Files

Files for this module:

- retailerSQL.m
- retailer.accdb
Outline

Motivation

Modern Database Systems

SQL

Bringing Data Into MATLAB/Excel
Motivation

Motivation: Getting data from external sources!

Data that is needed for a model may reside in text files (done).

Data that is needed for a model may reside in other worksheets and workbooks (done).

Data that is needed for a model may reside in modern relational databases (Access, SQL Server, Oracle, DB2)

Data that is needed for a model may reside in XML files (next lecture).
Characteristics of a database management system (DBMS):

- Data Manipulation Language: get information in and out of the database. SQL is the most important standard.

- System Catalog or Data Dictionary: meta database, e.g. a tables listing all the tables, keys, attributes, indices, etc.

- Utility programs
  1. scheduled backups
  2. user account management

- Report Generation
Modern Database Systems

Characteristics of a database management system (DBMS):

- Data import and export
- Creation of SQL based procedures and interface to programming language
- Locking mechanisms
- Transaction processing – must end up completed (committed) or completely undone
Modern Database Systems

Modern database systems:

- **Popular Commercial Packages:** Oracle, DB2/Informix, SQL Server, Sybase, Ingres

- **Open Source:** MySQL, PostgreSQL, InterBase, Firebird, SAP DB

- **Desktop:** MS Access is the most popular
Key Ideas:

- **Relation or table**: a flat file. Think of this as a matrix with rows and columns. A range in an Excel spreadsheet

- **Record**: a row in the table

- **Field or attribute**: a column in the table

- **Primary Key**: a subset of the fields with the property that for a given setting of the key values there is a unique value for all other fields. For example, Social Security number. Every table must have a primary key.

- **Composite or multi-field key**: is a key with more than one attribute
Modern Database Systems

Key Ideas:

▶ **Universal Relation**: a relation containing all the attributes in the database. This is often called a flat file database. Many databases are simple flat files.

▶ **Relational Database**: A database that allows for multiple tables which can be joined together in order to get information.

▶ **SQL**: Structured Query Language. A way to get information in and out of a database. SQL is to the database world what HTML is to the Web.
The key for this database is `rep_e_mail` and `sku_num` (we have a multi-attribute key).
Modern Database Systems

Problems with a flat file database:

- **Redundancy** – consider the retailer example, we may have hundreds or even thousands of SKUs supplied by the same supplier. Why store the same supplier address thousands of times?

- **Deletion Anomalies** – if we discontinue all of the SKUs of a particular supplier and delete these records, we lose information about the supplier.

- **Insertion Anomalies** – when we insert a new record we may create inconsistencies.

- **Null Values** – How do we treat null values: leave it blank, insert a zero, or a special character?
Big breakthrough was the development of a relational database!

A relational database allows multiple users throughout a corporation to easily and directly access all of the data.

The concept of a relational database is due to Ted Codd, an IBM research fellow, who wrote the theory paper in 1970. This paper was promptly ignored by IBM.

Larry Ellison and his associates (Ed Oates) realized the potential. Ellison was a COBOL programmer who wanted to get rich. He did. Ellison has a U of C connection.
Modern Database Systems

Normalized Relational Database:

<table>
<thead>
<tr>
<th>SUP_ID</th>
<th>REP_E_MAIL</th>
<th>SKU_NUM</th>
<th>PRICE</th>
</tr>
</thead>
<tbody>
<tr>
<td>s45</td>
<td><a href="mailto:tjr@s45.com">tjr@s45.com</a></td>
<td>441</td>
<td>$53.00</td>
</tr>
<tr>
<td>s45</td>
<td><a href="mailto:tjr@s45.com">tjr@s45.com</a></td>
<td>577</td>
<td>$55.00</td>
</tr>
<tr>
<td>s55</td>
<td><a href="mailto:sne@s55.com">sne@s55.com</a></td>
<td>577</td>
<td>$57.00</td>
</tr>
<tr>
<td>s55</td>
<td><a href="mailto:teg@s55.com">teg@s55.com</a></td>
<td>643</td>
<td>$63.00</td>
</tr>
<tr>
<td>s57</td>
<td><a href="mailto:arn@s57.com">arn@s57.com</a></td>
<td>342</td>
<td>$89.00</td>
</tr>
<tr>
<td>s99</td>
<td><a href="mailto:pm@s99.com">pm@s99.com</a></td>
<td>667</td>
<td>$56.00</td>
</tr>
</tbody>
</table>
# Modern Database Systems

## Normalized Relational Database:

### supplier

<table>
<thead>
<tr>
<th>SUP_ID</th>
<th>STREET</th>
<th>CITY</th>
</tr>
</thead>
<tbody>
<tr>
<td>s45</td>
<td>78 State</td>
<td>Plano</td>
</tr>
<tr>
<td>s55</td>
<td>64 Dole</td>
<td>Lajitas</td>
</tr>
<tr>
<td>s57</td>
<td>72 Vine</td>
<td>Uvalde</td>
</tr>
<tr>
<td>s99</td>
<td>15 Lake</td>
<td>Hondo</td>
</tr>
</tbody>
</table>

### sku

<table>
<thead>
<tr>
<th>SKU_NUM</th>
<th>COLOR</th>
</tr>
</thead>
<tbody>
<tr>
<td>342</td>
<td>red</td>
</tr>
<tr>
<td>441</td>
<td>green</td>
</tr>
<tr>
<td>577</td>
<td>blue</td>
</tr>
<tr>
<td>643</td>
<td>green</td>
</tr>
<tr>
<td>667</td>
<td>red</td>
</tr>
</tbody>
</table>
A one-to-many join:
Don Chamberlin and Ray Boyce at IBM developed a prototype of SQL and published their results.

This is the **Gold Standard** for databases. All major commercial databases support SQL.
Refer back to the retailer table:

Example query: Get the e-mail of everyone supplying part number 577

Here is the SQL for the query:

```sql
SELECT retailer.rep_e_mail, retailer.sku_num
FROM retailer
WHERE retailer.sku_num=577
```
The result of the query is

<table>
<thead>
<tr>
<th>rep_e_mail</th>
<th>sku_num</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="mailto:sne@s55.com">sne@s55.com</a></td>
<td>577</td>
</tr>
<tr>
<td><a href="mailto:tjr@s45.com">tjr@s45.com</a></td>
<td>577</td>
</tr>
</tbody>
</table>

A slight modification of the previous query is:

```sql
SELECT retailer.rep_e_mail, retailer.sku_num
FROM retailer
WHERE retailer.sku_num=577
ORDER BY retailer.rep_e_mail DESC
```

with result:

<table>
<thead>
<tr>
<th>rep_e_mail</th>
<th>sku_num</th>
</tr>
</thead>
<tbody>
<tr>
<td><a href="mailto:tjr@s45.com">tjr@s45.com</a></td>
<td>577</td>
</tr>
<tr>
<td><a href="mailto:sne@s55.com">sne@s55.com</a></td>
<td>577</td>
</tr>
</tbody>
</table>
The SQL for the query *Get the city of everyone supplying red parts* is:

```
SELECT retailer.city, retailer.color
FROM retailer
WHERE retailer.color="red"
```

the result of the query is

<table>
<thead>
<tr>
<th>city</th>
<th>color</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ulvade</td>
<td>red</td>
</tr>
<tr>
<td>Honda</td>
<td>red</td>
</tr>
</tbody>
</table>
Here is the SQL for a join – *get the cities that supply sku 577*

```sql
SELECT orders.sku_num, supplier.city
FROM orders INNER JOIN supplier ON
    orders.sup_id = supplier.sup_id
WHERE orders.sku_num=577
```

the result of the query is

<table>
<thead>
<tr>
<th>sku_num</th>
<th>city</th>
</tr>
</thead>
<tbody>
<tr>
<td>577</td>
<td>Plano</td>
</tr>
<tr>
<td>577</td>
<td>Lajitas</td>
</tr>
</tbody>
</table>
Note the key words used in these examples:

- **SELECT**: specifies the fields you want
- **FROM**: specifies the table you want
- **WHERE**: specifies the criterion or criteria you want

A few other key words:

1. **DISTINCT**
2. **GROUP BY**
3. **HAVING**
4. **ORDER BY**
5. **AS**
Some References:

➤ See Section 14.4 of Albright

➤ A nice tutorial
   http://www.w3schools.com/SQL/default.asp

➤ Test some queries in the tutorial
   http://www.w3schools.com/SQL/sql_tryit.asp
Bringing Data Into MATLAB/Excel

Key Idea:

▶ In most modern organizations most of the data resides in databases (Oracle, DB2, SQL Server, etc.).

▶ We want to bring that data into MATLAB/Excel and do something useful with it.

▶ Therefore, MATLAB/Excel must communicate with the database server!

▶ In Excel (with VBA) and MATLAB we are going to send SQL querier to a database server and get the results back.

How is this done in practice?
Bringing Data Into Excel

There are two primary ways for Excel to talk to database servers.

- Use Web Services
- Use Microsoft ADO (ActiveX Data Objects)

We are going to use Web Services – far, far, more generic than ADO! The way we access databases in Excel will be identical to how we do this in MATLAB.

It is best to use open, platform independent, standards.
### In Class Exercise - Practice SQL

**SQL Query:**

```sql
SELECT * FROM records
```

**Table:**

<table>
<thead>
<tr>
<th>id</th>
<th>title</th>
<th>group</th>
<th>year</th>
<th>price</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>In the Still of the Night</td>
<td>The Five Satins</td>
<td>1956</td>
<td>17</td>
</tr>
<tr>
<td>5</td>
<td>The Closer You Are</td>
<td>Earl Lewis and the Channels</td>
<td>1957</td>
<td>17</td>
</tr>
<tr>
<td>3</td>
<td>Angel Baby</td>
<td>Rosie and the Originals</td>
<td>1961</td>
<td>25</td>
</tr>
<tr>
<td>4</td>
<td>One Summer Night</td>
<td>The Danleers</td>
<td>1958</td>
<td>7.5</td>
</tr>
<tr>
<td>2</td>
<td>The Duke of Earl</td>
<td>Gene Candler</td>
<td>1962</td>
<td>5.5</td>
</tr>
<tr>
<td>8</td>
<td>Can I Come Over Tonight</td>
<td>The Velours</td>
<td>1956</td>
<td>9.5</td>
</tr>
<tr>
<td>6</td>
<td>Over the Mountain, Across the Sea</td>
<td>Johnnie and Joe</td>
<td>1957</td>
<td>11</td>
</tr>
<tr>
<td>11</td>
<td>In the Still of the Night</td>
<td>Soul's of Acapella</td>
<td>2008</td>
<td>12</td>
</tr>
<tr>
<td>10</td>
<td>Uncle Sam's Man</td>
<td>The Elgins</td>
<td>1960</td>
<td>13</td>
</tr>
<tr>
<td>9</td>
<td>Angel Maureen</td>
<td>Jimmy and The Crestones</td>
<td>1964</td>
<td>12</td>
</tr>
<tr>
<td>7</td>
<td>Deserie</td>
<td>The Charts</td>
<td>1957</td>
<td>25</td>
</tr>
<tr>
<td>12</td>
<td>Looking for an Echo</td>
<td>Kenny Vance and the Planotones</td>
<td>2006</td>
<td>19</td>
</tr>
<tr>
<td>13</td>
<td>Over the Mountain, Across the Sea</td>
<td>The Orlons</td>
<td>1962</td>
<td>9</td>
</tr>
</tbody>
</table>

13 rows fetched.

Selected schema 'doowop'.
In Class Exercise - Practice SQL

Query 1: Get all records in the database

Query 1: Get all records and sort in decreasing order by price

Query 1: Get all records in the database from 1956

Query 1: Get all records in the database recorded by the Velours or Jimmy and The Crestones

Query 5: Query the doowop database and using a single SQL command return each unique song title and a count of how many times that title appears in the database. Order the result in descending order by count.