Basic Definitions

An *information system* is software that helps the user organize and analyze data. Information systems:

- electronic spreadsheets
- database management systems

A *spreadsheet* is a software application that allows the user to organize and analyze data using a grid of labeled *cells*. A cell can contain data or a formula that is used to calculate a value.

A *database* is a collection of records stored in a computer in a systematic (structured) way, so that a computer program can consult it to answer *queries*.

The computer program used to manage and query a database is known as a *database management system (DBMS)*.
Spreadsheet Software (Microsoft Excel): Reminder

*Formulas* begin with =. They may contain values, references to cells, symbols of *arithmetic operations* (+, -, *, /), and calls of *spreadsheet functions*.

A *spreadsheet function* is a computation provided by the spreadsheet software that can be incorporated into formulas.

=AVVERAGE(A1:B1,C1:D1,A2:B2,C2:D2)

A *range* is a rectangular block of cells specified by two endpoints (references to corner cells). Example: A1:D2

=AVVERAGE(A1:D2)

Properties of relative and absolute references for *copy-paste* and *fill* (right and down): F3, $F3, F$3, $F$3

Other features: Fill series; dynamic recalculation; circular reference detection; formatting; search; data analysis; charts
Database management concepts

• Database Management Systems (DBMS)
  • An example of a database (relational)
  • Database schema (e.g. relational)
  • Data independence
  • Architecture of a DBMS
• Types of DBMS
• Basic DBMS types
• Retrieving and manipulating data: query processing
  • Database views
• Data integrity
• Client-Server architectures
• Knowledge Bases and KBS (and area of AI)
• DBMS tasks:
  • Managing large quantity of structured data
  • Efficient retrieval and modification: query processing and optimization
  • Sharing data: multiple users use and manipulate data
  • Controlling the access to data: maintaining the data integrity

• An example of a database (relational):
  • Relations (tables)
  • Attributes (columns)
  • Tuples (rows)
  • Example query: Salesperson='Mary' AND Price>100.
More Basic Definitions

A query is a request for information submitted to a database.

The database schema provides the logical structure of the data in the database, independent of how it is physically stored.

The relational model is a database model in which the data items and the relationships among them are organized into tables.

A table is a collection of database records. A record (a.k.a. database object, entity) is a collection of related fields. Each field (a.k.a. attribute) contains a single data value. The key field(s) uniquely identify a record in the table.
### Relation ITEM

<table>
<thead>
<tr>
<th>Item#</th>
<th>ItemName</th>
<th>Quantity</th>
<th>Location</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>pump</td>
<td>25</td>
<td>A23.2</td>
</tr>
<tr>
<td>235</td>
<td>saw</td>
<td>42</td>
<td>B3.9</td>
</tr>
<tr>
<td>589</td>
<td>hose</td>
<td>110</td>
<td>A23.5</td>
</tr>
<tr>
<td>601</td>
<td>ladder</td>
<td>12</td>
<td>B14.6</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

### Relation SUPPLIES

<table>
<thead>
<tr>
<th>Item#</th>
<th>Supplier#</th>
</tr>
</thead>
<tbody>
<tr>
<td>123</td>
<td>23</td>
</tr>
<tr>
<td>235</td>
<td>23</td>
</tr>
<tr>
<td>589</td>
<td>99</td>
</tr>
<tr>
<td>601</td>
<td>6</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

### Relation SALES

<table>
<thead>
<tr>
<th>Item#</th>
<th>Salesperson</th>
<th>Price</th>
</tr>
</thead>
<tbody>
<tr>
<td>601</td>
<td>Sam</td>
<td>169.95</td>
</tr>
<tr>
<td>123</td>
<td>Sam</td>
<td>99.95</td>
</tr>
<tr>
<td>589</td>
<td>Mary</td>
<td>24.98</td>
</tr>
<tr>
<td>601</td>
<td>John</td>
<td>169.95</td>
</tr>
<tr>
<td>123</td>
<td>Mary</td>
<td>99.95</td>
</tr>
<tr>
<td>601</td>
<td>Mary</td>
<td>169.95</td>
</tr>
<tr>
<td>235</td>
<td>John</td>
<td>25.49</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

### Relation SUPPLIER

<table>
<thead>
<tr>
<th>Supplier#</th>
<th>City</th>
<th>Phone</th>
</tr>
</thead>
<tbody>
<tr>
<td>6</td>
<td>Albany</td>
<td>518-555-1234</td>
</tr>
<tr>
<td>23</td>
<td>Troy</td>
<td>518-555-4321</td>
</tr>
<tr>
<td>48</td>
<td>Schenectady</td>
<td>518-555-6789</td>
</tr>
<tr>
<td>99</td>
<td>New York</td>
<td>201-555-9876</td>
</tr>
<tr>
<td>...</td>
<td>...</td>
<td>...</td>
</tr>
</tbody>
</table>

**Figure 9.1** The inventory relational database
• Database schema (e.g. relational):
  • Names and types of attributes
  • Addresses
  • Indexing
  • Statistics
  • Authorization rules to access data etc.
• Data independence: separation of the physical and logical data
  • Particularly important for distributed systems
  • The mapping between them is provided by the schema
• Architecture of a DBMS - three levels: external, conceptual and internal schema
• Types of DBMS
  • The data structures supported: tables (relational), trees, networks, objects
  • Type of service provided: high level query language, programming primitives
Basic DBMS types

• Linear files
  • Sequence of records with a fixed format usually stored on a single file
  • Limitation: single file
  • Example query: Salesperson='Mary' AND Price>100

• Hierarchical structure
  • Trees of records: one-to-many relationships
  • Limitations:
    • Requires duplicating records (e.g. many-to-many relationship)
    • Problems when updated
    • Retrieval requires knowing the structure (limited data independence): traversing the tree from top to bottom using a procedural language

• Network structure: similar to the hierarchical database with the implementation of many-to-many relationships

• Relational structure

• Object-Oriented structure
  • Objects (collection of data items and procedures) and interactions between them.
  • Is this really a new paradigm, or a special case of network structure?
  • Separate implementation vs. implementation on top of a RDBMS
Structured Query Language (SQL)

The *Structured Query Language (SQL)* is a comprehensive relational database language for data management and queries. SQL is not case sensitive. Spaces are used as separators in a statement.

The basic *select* statement format:

```
select attribute-list from table-list where condition
```

Simple sample query:
```
select Title from Movie where Rating = 'R' order by ProductionCost
```

For this to work, we need…
Database Design

Entity-relationship (ER) modeling is a popular technique for designing relational databases. An ER diagram captures record types, attributes, and relationships in a graphical form.

• Types of records (classes for the database objects) are shown in rectangles
• Fields (attributes) are shown in ovals
• Relationships are shown in diamonds

Cardinality relationships:
• one-to-one
• one-to-many
• many-to-many
Relational structure

- Relations, attributes, tuples
- Primary key (unique combination of attributes for each tuple)
- Foreign keys: relationships between tuples (many-to-many).
  
  Example: SUPPLIES defines relations between ITEM and SUPPLIER tuples.

- Advantages: many-to-many relationships, high level declarative query language (e.g. SQL)

Another SQL example (retrieve all items supplied by a supplier located in Troy):

```sql
SELECT ItemName
FROM ITEM, SUPPLIES, SUPPLIER
WHERE SUPPLIER.City = "Troy" AND
  SUPPLIER.Supplier# = SUPPLIES.Supplier# AND
  SUPPLIES.Item# = ITEM.Item#
```

- Programming language interfaces: including SQL queries in the code
Retrieving and manipulating data: query processing

- Parsing and validating a query: data dictionary - a relation listing all relations and relations listing the attributes
- Plans for computing the query: list of possible way to execute the query, estimated cost for each. Example:
  
  ```
  SELECT ItemNames, Price
  FROM ITEM, SALES
  WHERE SALES.Item# = ITEM.Item# AND Salesperson="Mary"
  ```
- Index: B-tree index, drawbacks - additional space, updating; indexing not all relations (e.g. the keys only)
- Estimating the cost for computing a query: size of the relation, existence/size of the indices. Example: estimating Attribute=value with a given number of tuples and the size of the index.
- Query optimization: finding the best plan (minimizing the computational cost and the size of the intermediate results), subsets of tuples, projection and join.
- Static and dynamic optimization
Database views

- Creating user defined subsets of the database
- Improving the user interface
- Example:

  ```sql
  CREATE VIEW MarySales(ItemName, Price)
  AS SELECT ItemName, Price
  FROM ITEM, SALES
  WHERE ITEM.Item# = SALES.Item# AND Salesperson = "Mary"
  ```

  Then the query:

  ```sql
  SELECT ItemName
  FROM MarySales
  WHERE Price > 100
  ```

  translates to:

  ```sql
  SELECT ItemName
  FROM ITEM, SALES
  WHERE ITEM.Item# = SALES.Item# AND Salesperson = "Mary" AND Price > 100
  ```
Modifying Database Content

The *insert* statement adds a new record to a table.

The *update* statement changes the values in one or more records of a table.

The *delete* statement removes all records from a table matching the specified condition.

delete from Movie where Title like ‘Naked Gun%’