Announcements

- **Grace Hopper Week ~ Dec. 6 – 12**
  Whereas Grace Murray Hopper, one of the first females in the field of computer science, engineered new programming languages (COBOL) and pioneered standards for computer systems which laid the foundation for many advancements in computer science; and Whereas the week of December 7, in honor of Grace Hopper’s birthday, is designated as ‘National Computer Science Education Week’ It’s followed by some appropriate ‘be it resolved’ points about encouraging education and opportunities for females and underrepresented minorities in the field.

Rear Admiral Grace Hopper

- **Computing pioneer (1906-1992)**
- Invented the compiler and COBOL

Announcements

- **This week**
  - Lab 11 and Project 3A due Wednesday at 10pm
  - Best GoPosters survey due Friday at 10pm
- **Finals week**
  - Lab 12 and Project 3B due Wednesday, Dec. 16 at 10pm

Announcements

- **No final for this class**

Announcements

- **Read ch 17 for today**
- **Read the second half of ch 13 for Lab 12**

Announcements

- **Quiz this week in lab**
  - Wednesday and Thursday
  - Topics
    - Spreadsheets
    - XML
    - Privacy
    - Cookies
    - Identity Theft
Announcements

- This week's GoPost discussion is posted

Just a thought…

- Dijkstra: Whether a computer can think is about as interesting as whether a submarine can swim.

Project 3

- Part A
  - Build database tables
  - Brief sample data
  - Answer queries
    - Get information out of the database

- Part B
  - Imagine a year has gone by. . .
    - The database has grown
    - Download the much-larger database
    - Boat Club members had some accidents

Project 3

- Project 3B
  - You're called in to extend the database to make sure sailors are qualified to run the boats they rent
  - Build the queries
  - Answer the questions
    - Save database and queries, and upload to Collect It
    - Submit WebQ

Announcements
Important!

- querySailorAge
  - current age: DateDiff("yyyy", [tableSailors][birthdate], Now())

FIT 100—Fluency with Information Technology

A Table with a View

Primary keys, normalization, and SQL

D.A. Clements

Video

- Relational databases and tables

Video

- Primary Keys (5 min.)

Primary/Foreign Key

- Controlled redundancy:
  - Stores relationship between tables
  - Database tables share common attributes only to enable the tables to be linked
  - True redundancy exists only when there is unnecessary duplication of attribute values

Problem Fields (Don’ts)

- Calculated field—can be computed by mathematical calculation or text concatenation
  - Waste of storage space (redundant)
  - No assurance the calculated value is updated when the user changes the input fields
- Multipart field—contains that should be two or more fields
  - Extra work when you want to analyze your data
- Multivalue field—multiple correct entries for the field
  - Create a separate subset table with each value in its own record
- Derived field—contents of one or more fields absolutely predicts the contents of another
  - Should be dropped from the table
**Entities**

- Entity
  - Anything that can be identified by a fixed number of its characteristics (*attributes*)
  - Attributes have
    - Names—field name, attribute, or column name
    - Values—the data stored in the table

- Attributes have
  - Names—field name, attribute, or column name
  - Values—the data stored in the table

**Island table**

<table>
<thead>
<tr>
<th>Island</th>
<th>Name</th>
<th>Area</th>
<th>Elevation</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Isabel</td>
<td>4568</td>
<td>1707</td>
</tr>
<tr>
<td></td>
<td>Fernanda</td>
<td>642</td>
<td>1494</td>
</tr>
<tr>
<td></td>
<td>Tower</td>
<td>14</td>
<td>76</td>
</tr>
<tr>
<td></td>
<td>Santa Cruz</td>
<td>986</td>
<td>846</td>
</tr>
</tbody>
</table>

Figure 16.4 A table instance for the island entity.

**Properties of Entities**

- A relational database table can be empty
- Instances Are Unordered
  - Order of the rows and columns does not matter in databases
  - Freedom to move the data is limited to exchanging entire rows or exchanging entire columns

**Properties of Entities (cont'd)**

- Uniqueness
  - No two rows can be the same
  - Two rows can have the same value for some attributes, just not all attributes
Properties Of Entities (cont'd)

- Atomic Data
  - Not decomposable into any smaller parts
  - Separate fields for street, city, state, postal code
  - "Only atomic data" rule relaxed for certain types of data
    - Dates, times, currency

Database schemas

- Database schema – way to define a table
  - Collection of table definitions that gives the name of the table, lists the attributes and their data types, and identifies the primary key

<table>
<thead>
<tr>
<th>Island</th>
<th>Island Name</th>
<th>Area</th>
<th>Elevation</th>
<th>Highest point on the island</th>
</tr>
</thead>
<tbody>
<tr>
<td>iName</td>
<td>Text</td>
<td>area</td>
<td>elevation</td>
<td>Number</td>
</tr>
</tbody>
</table>

Database Tables Summary

- Tables in databases have a structure that is specified by metadata
- The structure is separate from its content
- A table structures a set of entities
- Things that we can tell apart by their attributes
- The entities of the table are represented as rows
- Rows and columns are unordered
- Tables and fields should have names that describe their contents
- Fields must be atomic (indivisible)
- One of more attributes define the primary key

Operations on Tables

- A database is a collection of tables
- Main use of database is to look up information
  - Users specify what they want to know and the database software finds it
- We can perform operations on tables to produce new tables
- The questions we ask of a database are answered with a whole new table, or view
**SELECT Operation**

- Takes rows from one table to create a new table
- Specify the table from which rows are to be taken, and the test for selection

**Syntax:** 
```
SELECT Test FROM Table
```

- Test is applied to each row of the table to determine if it should be included in result table
- Test uses attribute names, constants, and relational operators
- If the test is true for a given row, the row is included in the result table; otherwise it is ignored

```sql
SELECT Interest='Beach' FROM Nations
```

**Animation**

- A natural join

**Physical and Logical Database**

**TABLES AND VIEWS**

- Physical database and logical database
  - Physical database is the files, records in any order, no logical organization other than tables
  - Logical database is a view of database that shows only the rows and fields needed by the users
    - Solves Information Overload:
      - Users see only what they need
      - Users see only what they have permission to see

**Physical vs. Logical**

- Figure 16.15: Structure of a database system. The physical database is the permanent repository of the data; the logical database, or view of the database, is the form of the database the users see. The transformation is implemented by the query processor, and is based on queries that define the logical database tables from the physical database tables.
Physical Database
- Designed by database administrators
- Fast to access
- No redundancy/duplicating information
- Multiple data can lead to inconsistent data
- Backup copies in case of accidental data deletion or disk crash

Logical Database
- Creating specialized views of the data for different users’ needs
- Creating a new “result set” from the current data each time
  - Fresh
  - Accurate

Defining Physical Tables
- Database schemes (schema)
  - Metadata specification that describes the database design

The Idea of Relationship
- A relationship is a correspondence between rows of one table and the rows of another table
  - Because the key Student_ID is used in each table, can not only find the address for each student (Lives_At), but can also find the student for each address (Home_Of)
- Relationship examples

Relationships in Practice
Defining Logical Tables

- Constructing a View Using Join

- Match on the common field of Student_ID

Master_List = Student JOIN Home_Base

On Student.Student_ID = Home_Base.StUDENT_ID

<table>
<thead>
<tr>
<th>STUDENT_ID</th>
</tr>
</thead>
<tbody>
<tr>
<td>First_Name</td>
</tr>
<tr>
<td>Middle_Name</td>
</tr>
<tr>
<td>Last_Name</td>
</tr>
<tr>
<td>Address</td>
</tr>
<tr>
<td>Student_Address</td>
</tr>
<tr>
<td>City</td>
</tr>
<tr>
<td>State</td>
</tr>
<tr>
<td>County</td>
</tr>
<tr>
<td>ZipCode</td>
</tr>
</tbody>
</table>

Figure 16.9: Illustration of the Master_List table, being created from Student
D. & Home_Base tables using a Join to insert the data types and key