Remember Database Terminology…

- A database is a set of tables
- The structure of a database is displayed in its schema
- A database instance is the current contents of a database

- Fields (columns) have a type …all items in a column have that type
- A database table stores information about entities

Logical Relationships Among Tables

- A database schema will represent the structure of your database. This includes the distinguishing attribute(s) of every entity the Primary Key
- Keys are also the way to represent relationships between entities
- The diagram (model) that shows how tables are related is known as the Entity Relationship Diagram (ERD)
- Separating information out into separate tables and establishing relationships among those tables allows database designers to avoid redundancy and keep data accurate
- The student information database shown in class last week represented a one-to-one relationship
- There are also one-to-many relationships and many-to-many relationships. You have seen one example in Lab.

Sample Entity Relationship Diagrams

- Relationships between tables will determine the key structures of the tables
How to Structure Tables to Show Relationships

- In a one-to-one relationship, tables are associated together through the primary key in each.

- In a one-to-many relationship, the primary key attribute in the one must be listed as an additional attribute in the many. The tables are associated together through the similar attributes.

- In a many-to-many relationship, a new table is ALWAYS created and the primary key attributes of the original tables are made attributes of the new ‘bridge’ table and are often combined to form the primary key – or a surrogate key is created.

Operations on Databases

Tables are useful, but they become much more powerful when we can manipulate them to create new tables from existing tables. For that, we need special operations.

Tables that Produce Other Tables

- Table operations can involve one or many tables.

- These basic operations are usually used together to create specific ‘views’ of the database.

- Let’s look at the basic operations performed on tables…

Basic Operations On Tables

- An obvious use of tables would be to construct other tables from them.

- The Dean’s list of students shown last week was a table constructed from other tables using several of the following operations: Select, Project, Union, Difference, Product.

  This table does exist by itself. It is a view of certain rows and columns from other tables.
A Sample Schema

Advisor

<table>
<thead>
<tr>
<th>Advisor_ID</th>
<th>Integer</th>
</tr>
</thead>
<tbody>
<tr>
<td>FirstName</td>
<td>String</td>
</tr>
<tr>
<td>LastName</td>
<td>String</td>
</tr>
<tr>
<td>Department</td>
<td>String</td>
</tr>
<tr>
<td>HireDate</td>
<td>Date</td>
</tr>
</tbody>
</table>

PK: Advisor_ID

Table Operation: Select

- Select picks a subset of records (rows) based on some criteria
- Select from <table> On <T/F condition>

We could create a subset from the Advisors table of just those advisors from the CSE Department

Select from Advisors On Department = "CSE"

<table>
<thead>
<tr>
<th>Advisor_ID</th>
<th>FirstName</th>
<th>LastName</th>
<th>Department</th>
<th>HireDate</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>Larry</td>
<td>Snyder</td>
<td>CSE</td>
<td>1982</td>
</tr>
<tr>
<td>2</td>
<td>Alex</td>
<td>Burton</td>
<td>CSE</td>
<td>1996</td>
</tr>
</tbody>
</table>

CSE_Advisors = Select from Advisors On Department = "CSE"

Table Operation: Project

- Project extracts columns from a table
- Project <attribute list> From <table>
- Picking out the essential information from CSE_Advisor

Project FirstName, LastName From CSE_Advisor

NameCSE_Advisor = Project FirstName, LastName From CSE_Advisor

Table Operations: Union and Difference

- Union combines two tables with like attributes
- Example: You want to combine two tables of scheduled games for different teams. One shows home games, the other away games. All the attributes are the same.

- Difference removes a table from a table with like attributes
- Example: You want to remove all games starting after 5PM from your schedule of games
**Table Operation: Product**
- Product multiplies two tables together creating a “super table”
- For each row in the first table, concatenate every row in the second table
  \[
  \langle \text{table} \rangle \times \langle \text{table} \rangle
  \]
- Product can be used to add new fields
- Product’s Column Rule: If TableA has m columns and TableB has n columns, then TableA \_x\_TableB has m+n columns
- Product creates a table of “all pairs”
- Product’s Row Rule: If TableA has m rows and TableB has n rows, then TableA \_x\_TableB has mn rows

**Product: The Rules Always Apply**
- Visualize a Product …
  \[
  \begin{array}{c}
  A \times B
  \end{array}
  \]
- The row and column rules always apply
- Column Rule: If TableA has m columns and TableB has n columns, then TableA \_x\_TableB has m+n columns
- Row Rule: If TableA has m rows and TableB has n rows, then TableA \_x\_TableB has mn rows

**Join – Product With a Match**
- Join is a combination performing a product and a select
- The Join symbol in database research literature is a “bow tie”
- Natural Join… suppose two tables have the same attribute, then use the Product operation to pair all rows of the two tables, but keep only those rows that match on the common attribute and remove duplicates
- Other joins are those done with other relational operators: <, >, <=, etc.

**Constructing the Dean’s View**
- The Dean’s View, which doesn’t literally exist, is created by joining two tables

<table>
<thead>
<tr>
<th>Home_Base</th>
<th>Top_Scholar</th>
</tr>
</thead>
<tbody>
<tr>
<td>Student_ID</td>
<td>Student_ID</td>
</tr>
<tr>
<td>Street_Address</td>
<td>Major</td>
</tr>
<tr>
<td>Nick_Name</td>
<td>GPA</td>
</tr>
<tr>
<td>City</td>
<td></td>
</tr>
<tr>
<td>State</td>
<td>PK: Student_ID</td>
</tr>
<tr>
<td>Country</td>
<td>PostalCode</td>
</tr>
<tr>
<td>PK: Student_ID</td>
<td></td>
</tr>
</tbody>
</table>

- The common field is student ID, so…
- Top_Scholar \_\_\_Home_Base creates the result
Joining Tables

Join is very useful because it allows us to construct more complete database views from small tables.

Joins in tables reveal stored relationships and provide the data users want to see.

Summary Of Table Operations

- The five basic operations on tables are:
  - Select
  - Project
  - Union
  - Difference
  - Product

- Join is a powerful operation created from project/select

- Table operations allow the data to be exhibited to users in whatever form they want.

For Wednesday

- Project 3, Part 2 due if you are re-submitting or submitting for the first time.
- Project 4, Part 1 is also due.
- Friday is Quiz 4 on Chapters 15 and 16 as well as lecture.