



# Thinking of Databases

*Databases are organized on two levels: 'physical' is how the data is stored, 'logical' is how it's viewed*



## Big Picture

A database is made of ...

- \* **Physical database** -- tables actually stored on the hard disk
- \* **Logical database** -- created on-the-fly virtual tables specified by ...
- \* **Queries** -- (programs written in SQL that) define how to make a logical table from physical tables
- \* **GUIs** -- the interface for users to DBs



# Avoiding Redundancy

Redundancy is bad because it can lead to inconsistent data ... very bad!

- Keep only one copy of any data  
... does that make it right???
- Rather than repeating data, reference it in the places where it is needed
  - Keep data in its own table
  - Save its key wherever it is needed

When users want the data, get it using its key!



# Physical Database

Physical databases store data in the “best” way -- no redundancy, ...

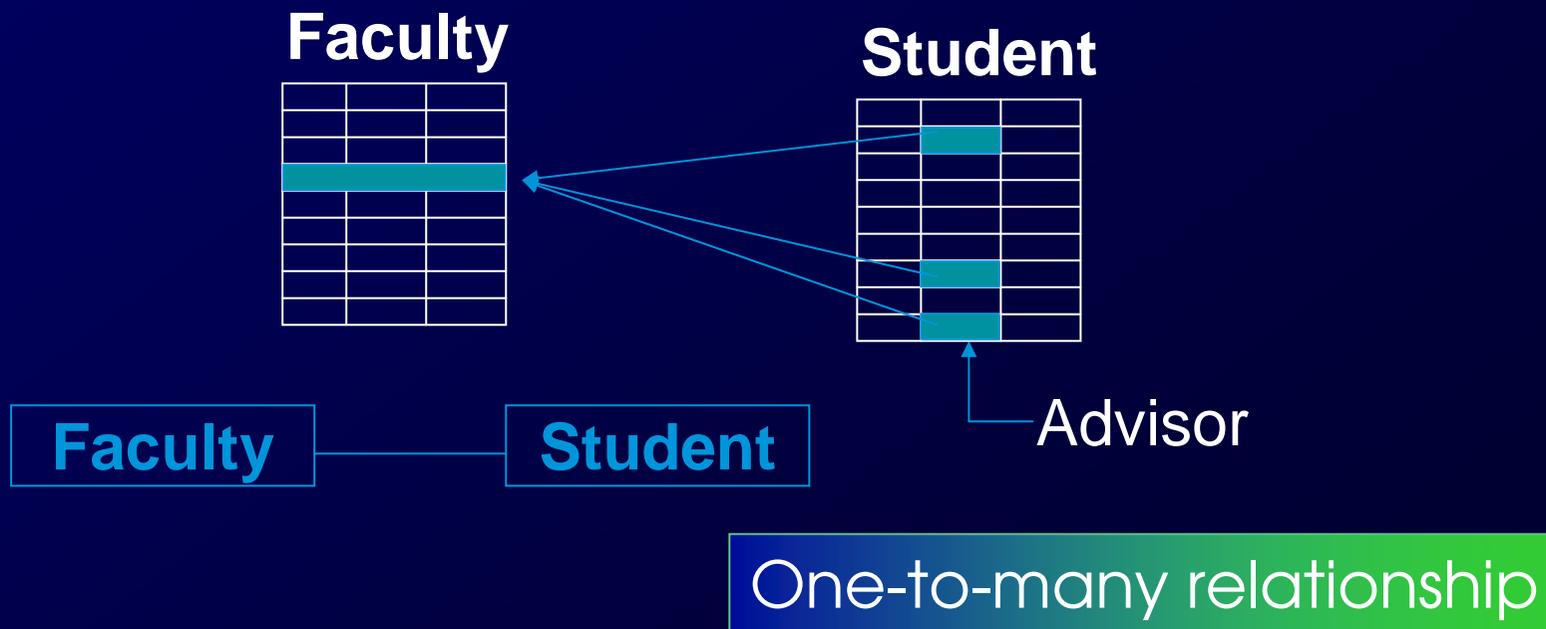
- Expect many tables of “simple” entities
- “Physical” means that the data is actually stored on the disk -- contrast with logical DBs that are “virtual tables”
- Physical databases are designed “for the computer” not for the user
- The “physical schema” gives table definitions and the relationships



# Relationships

The table data entries are not just text & numbers, but they have meaning

- Relationships spell out that meaning





# Kinds of Relationships

## One-to-One



## One-to-Many



## Many-to-Many



Name relationships by their meaning



# Logical Databases

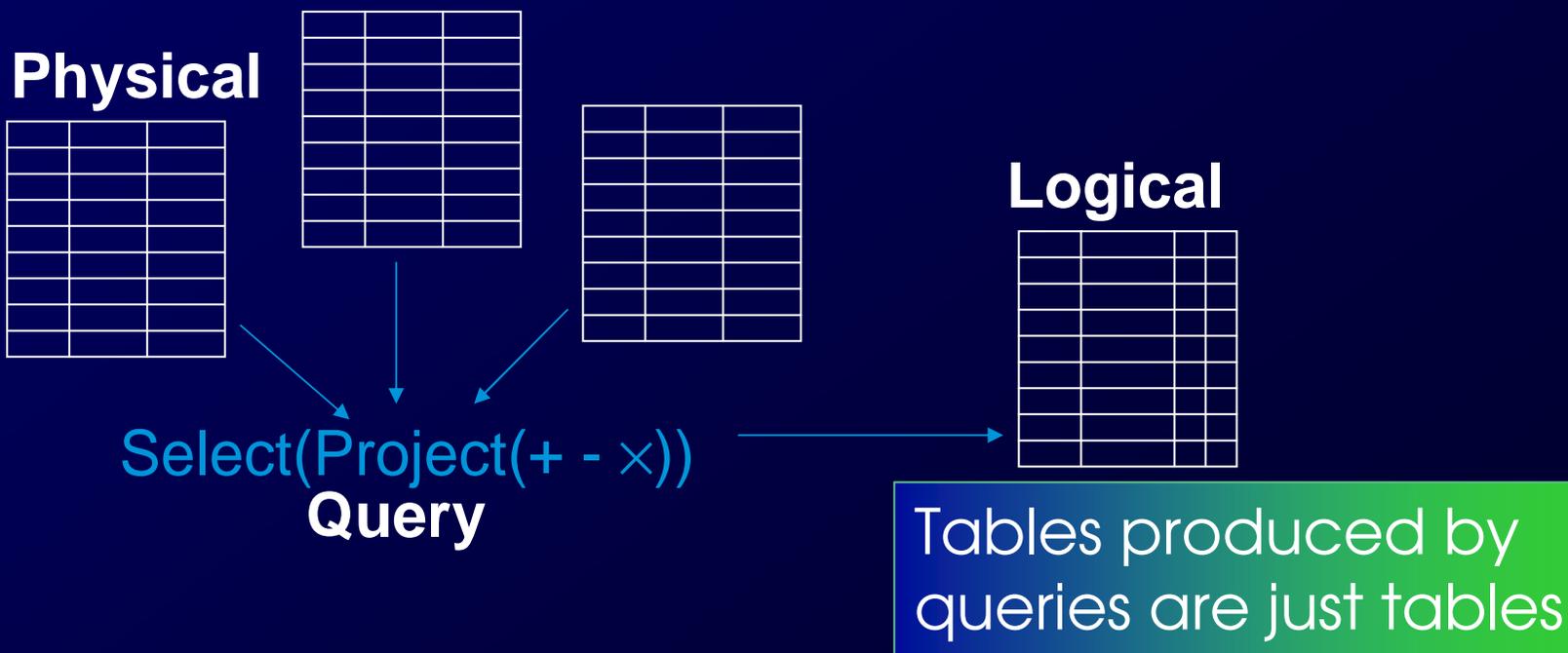
## Users want & need different information

- Different tasks require different information
  - Different authority levels, e.g. need to know
  - Customizing to users means everyone sees exactly what they need to see
- \* A *view* is a user's customized database
  - \* *Views* are virtual, built on-the-fly from the physical database and not kept
    - Data is always current
    - Custom structure can be very adaptable



# Queries

Queries are commands (using the 5 table operations) that create logical database (views) from physical





# SQL

The structured query language is the industry standard query language

“Structured” means the queries have a standard form

Common clauses --

SELECT <fields desired>

FROM <list of tables>

INNER JOIN <table> ON <conditions>

WHERE <criterion>

← Like Project!

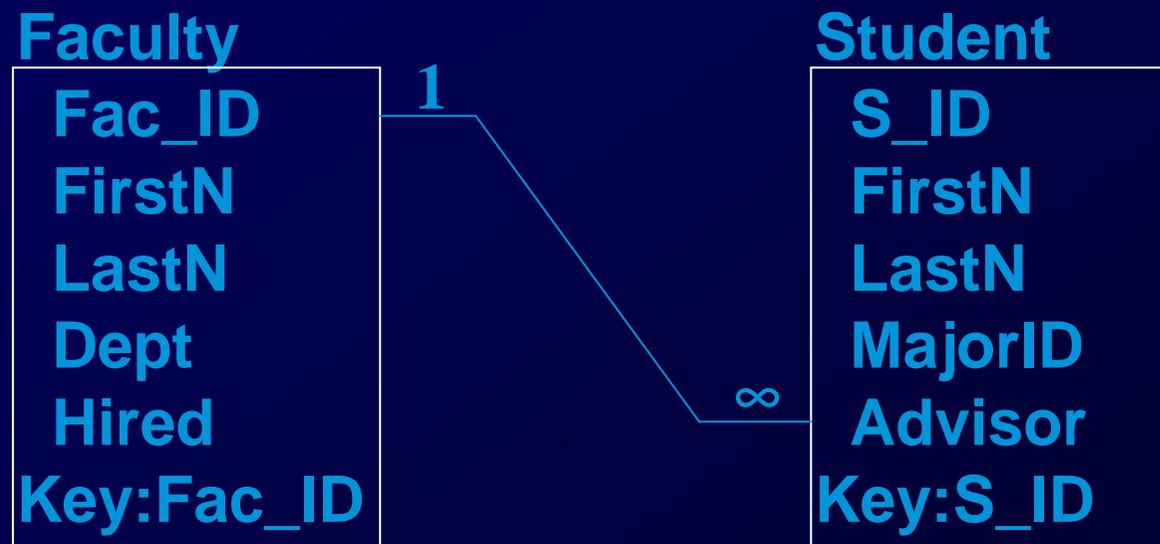
SQL is not case sensitive



# Sample Database

Define a university DB schema

- ER Diagram **Faculty** — **Student**
- Specifying a 1-to-many relationship





# Sample SQL Queries

Typical: `SELECT<attribs>FROM<tables>`

```
SELECT Student.FirstN, Student.LastN,  
       Student.MajorID  
FROM Student  
WHERE Student.S_ID= 0206125;
```

```
SELECT Student.FirstN, Student.LastN  
FROM Student  
WHERE MajorID=14;
```



## Join Example

Find the students of a given professor

```
SELECT Student.FirstN, Student.LastN,  
       Faculty.LastN  
FROM (Student INNER JOIN Faculty  
      ON Student.Advisor = Faculty.Fac_ID)
```

- Notice that selection comes from the combined (by Inner Join) table



# DB Design Paradigm

## Guidelines for good databases:

- Build physical DB to avoid redundancy, etc
- Each physical table represents 1 entity
- Expect that no physical table gives any user their exact view
- To build view, build a query that ...
  - Joins tables together into a 'super' table
  - Trims out only the items the user wants

These guidelines are not an algorithm,  
but they usually produce good results



Check out Project 3



## The Summary

A database is made of ...

- \* **Physical database** -- tables actually stored on the hard disk
- \* **Logical database** -- created on-the-fly virtual tables specified by ...
- \* **Queries** -- (programs written in SQL that) define how to make a logical table from physical tables
- \* **GUIs** -- the interface for users to DBs