Database Structure

Chapter 16

Structure Of A Database

- We want to arrange the information in a database in a way that users see a relevant-to-their-needs view of the data that they will use continually

Physical vs. Logical Databases

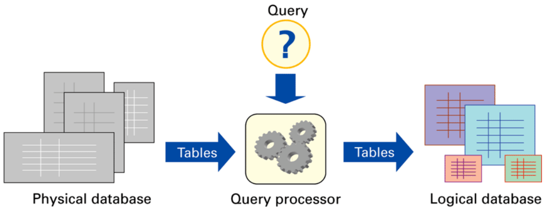


Figure 16.15 Structure of a database system. The physical database is the permanent repository of the data; the logical database, or view of the database, is the form of the database the users see. The transformation is implemented by the query processor, and is based on queries that define the logical database tables from the physical database tables.

Physical vs. Logical Databases

- The point of the two-level system is to separate the management of the data (physical database) from the presentation of the data (logical view of the database)

Physical Databases

- Designed by database administrators
- Fast to access
- No redundancy/duplicate information
  - Multiple data can lead to inconsistent data
- Backup copies in case of accidental data deletion or disk crash

Logical Database

- Creating specialized versions/views of the data for different users' needs
- Creating a new copy from the single data each time

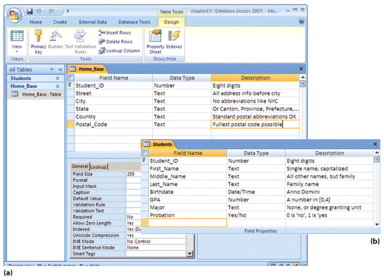
Queries

- A query is a specification using operations that define a table from other tables
- SQL (Structured Query Language)
  - Seen in last lecture
  - Standard database language to write queries

Defining Physical Tables

- Database schemas
  - Metadata specification that describes the database design

Database Schema



Home_Base

Field Name	Data Type	Description
Student_ID	Number	Eight digits
Street	Text	All address info before city
City	Text	No abbreviations like NYC
State	Text	Or Canton, Province, Prefecture,...
Country	Text	Standard postal abbreviations OK
Postal_Code	Text	Fullest postal code possible

Students

Field Name	Data Type	Description
Student_ID	Number	Eight digits
First_Name	Text	Single name, capitalized
Middle_Name	Text	All other names, but family
Last_Name	Text	Family name
Birthdate	Date/Time	Anno Domini
GPA	Number	A number in [0,4]
Major	Text	None, or degree granting unit
Probation	Yes/No	0 is 'no', 1 is 'yes

Field Properties

General | Lookup

Field Size	255
Format	
Input Mask	
Caption	
Default Value	
Validation Rule	
Validation Text	
Required	No
Allow Zero Length	Yes
Indexed	Yes (Du
Unicode Compression	Yes
IME Mode	No Control
IME Sentence Mode	None
Smart Tags	

Figure 16.16 Table declarations from Microsoft Access 2007: (a) `Home_Base` table declaration shown in the design view; and (b) students table declaration. Notice that the key is specified by the tiny key next to `Student_ID` in the first column.

Connecting Database Tables by Relationships

- Different tables can have different security access restrictions based on their data
  - For example, some can access `Home_Base` data without having access to more sensitive data in `Students`
- Separate tables but not independent
  - `Student_ID` connects (establishes a relationship) between the two tables

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 find the address for each student (*Lives_At*) AND can also find the student for each address (*Home_Of*)

Relationships In Practice

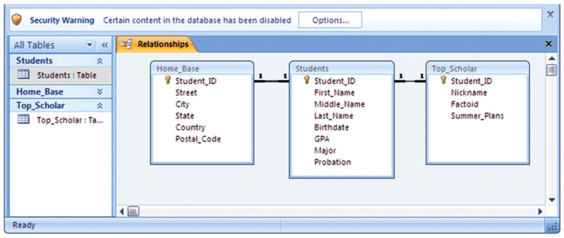
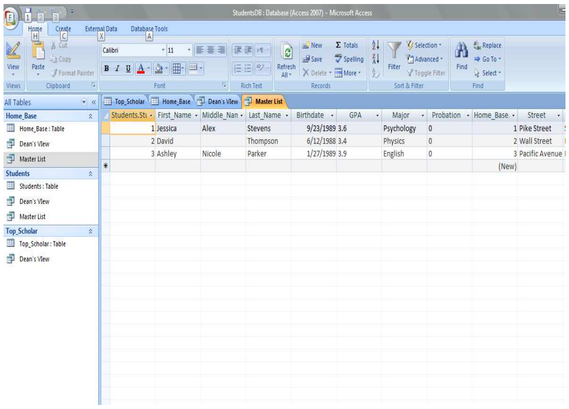


Figure 16.17 The *Relationships* window from the Microsoft Access database system; the 1-to-1 *Lives_At* and *Home_Of* relationships are shown between `Home_Base` and `Students`.

Defining Logical Tables

- Create a Master Table which combines 2 tables.
- Construction Using `Join`
  - Match on the common field of `Student_ID`

```
SELECT *
FROM Students INNER JOIN Home_Base
ON Students.Student_ID = Home_Base.Student_ID
```



Students.Stu	First_Name	Middle_Nan	Last_Name	Birthdate	GPA	Major	Probation	Home_Base.	Street
1	Jessica	Alex	Stevens	9/23/1989	3.6	Psychology	0	1	Pike Street
2	David		Thompson	6/12/1988	3.4	Physics	0	2	Wall Street
3	Ashley	Nicole	Parker	1/27/1989	3.9	English	0	3	Pacific Avenue
								(New)	

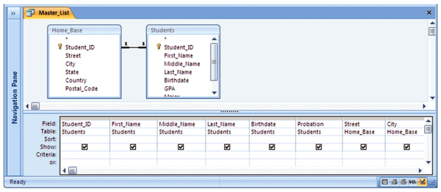
Join Resulting Attributes

```
Student_ID
First_Name
Middle_Name
Last_Name
Birthdate
On_Probation
Street_Address
City
State
Country
Postal_Code
```

Figure 16.18 Attributes of the `Master_List` table. Being created from `Student` and `Home_Base` allows `Master_List` to inherit its data types and key (`Student_ID`) from the component tables.

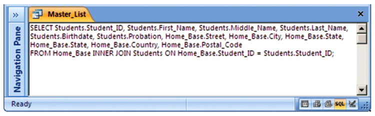
Practical Construction Using QBE

- Query By Example
  - Given a template of a table we fill in what we want in the fields



Field:	Student_ID	First_Name	Middle_Name	Last_Name	Birthdate	Probation	Street	City
Table:	Students	Students	Students	Students	Students	Students	Home_Base	Home_Base
Sort:								
Show:	☑	☑	☑	☑	☑	☑	☑	☑
Criteria:								
or:								

Figure 16.19 The Query By Example definition of the `Master_List` table from MS Access.

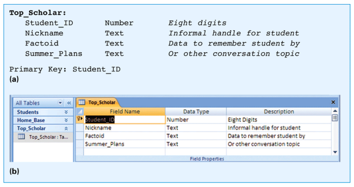


```
SELECT Students.Student_ID, Students.First_Name, Students.Middle_Name, Students.Last_Name,
Students.Birthdate, Students.Probation, Home_Base.Street, Home_Base.City, Home_Base.State,
Home_Base.State, Home_Base.Country, Home_Base.Postal_Code
FROM Home_Base INNER JOIN Students ON Home_Base.Student_ID = Students.Student_ID;
```

Figure 16.20 SQL query created from the Query By Example data in Figure 16.19.

The Dean's View

- Storing the Dean's Data
  - Top_Scholar is information of interest only to the dean



```
Top_Scholar:
   Student_ID      Number      Eight digits
   Nickname        Text        Informal handle for student
   Factoid         Text        Data to remember student by
   Summer_Plans    Text        Or other conversation topic

Primary Key: Student_ID
```

(a)

Field Name	Data Type	Description
Student_ID	Number	Eight Digits
Nickname	Text	Informal handle for student
Factoid	Text	Data to remember student by
Summer_Plans	Text	Or other conversation topic

(b)

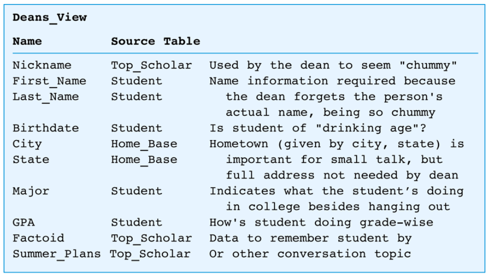
Figure 16.21 The `Top_Scholar` definition: (a) informal form, (b) in MS Access.

Join Three Tables into One

- `Join` using **Top_Scholar**, **Student**, and **Home_Base** tables matching on the `Student_ID` attribute across all three tables

- Trim the Table
    - `Project` – retrieve certain columns

Creating A Dean's View

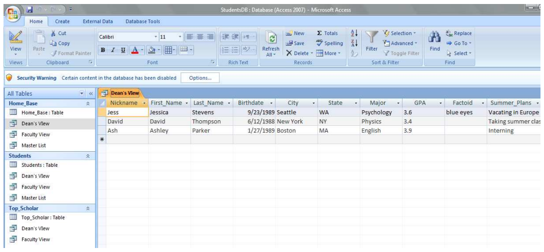


Deans_View

Name	**Source Table**	
Nickname	Top_Scholar	Used by the dean to seem "chummy"
First_Name	Student	Name information required because the dean forgets the person's actual name, being so chummy
Last_Name	Student	
Birthdate	Student	Is student of "drinking age"?
City	Home_Base	Hometown (given by city, state) is important for small talk, but full address not needed by dean
State	Home_Base	
Major	Student	Indicates what the student's doing in college besides hanging out
GPA	Student	How's student doing grade-wise
Factoid	Top_Scholar	Data to remember student by
Summer_Plans	Top_Scholar	Or other conversation topic

Figure 16.22 The Dean's View fields showing their source in physical database tables.

- SELECT Top_Scholar.Nickname, Students.First_Name, Students.Last_Name, Students.Birthdate, Home_Base.City, Home_Base.State, Students.Major, Students.GPA, Top_Scholar.Factoid, Top_Scholar.Summer_Plans
  FROM (Home_Base INNER JOIN Students ON Home_Base.Student_ID=Students.Student_ID) INNER JOIN Top_Scholar ON Students.Student_ID=Top_Scholar.Student_ID;



Home_Base
Home_Base : Table
Dean's View
Faculty View
Master List
Students
Students : Table
Dean's View
Faculty View
Master List
Top_Scholar
Top_Scholar : Table
Dean's View
Faculty View

Dean's View

Nickname	First_Name	Last_Name	Birthdate	City	State	Major	GPA	Factoid	Summer_Plans
Jess	Jessica	Stevens	9/23/1989	Seattle	WA	Psychology	3.6	blue eyes	Vacating in Europe
David	David	Thompson	6/12/1988	New York	NY	Physics	3.4		Taking summer clas
Ash	Ashley	Parker	1/27/1989	Boston	MA	English	3.9		Interning

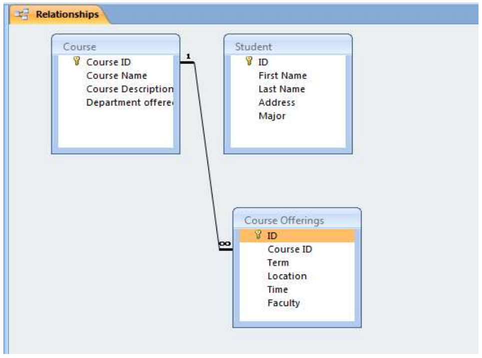
Exercise- Designing a DB

- Create a Database for administrative services in UW to manage courses and student info.

- We need to store
    - Student's Basic information
    - Courses offered (term, faculty, location etc)
    - Basic Information about courses(course no, department, name, etc)

Data to Capture

- Student id
  first_name
  middle_name
  last_name
  gpa
  Course id
  department
  course_name
  course_description
  SLN
  term
  location
  when_meet
  faculty

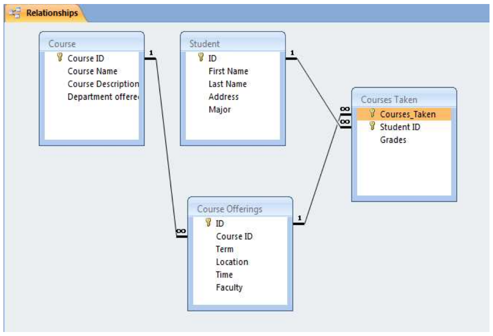


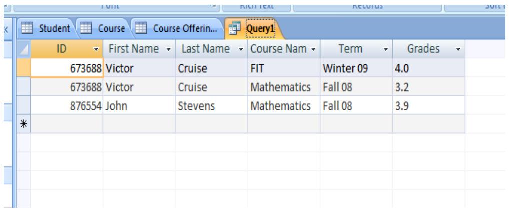
Course
Course ID
Course Name
Course Description
Department offere

Student
ID
First Name
Last Name
Address
Major

Course Offerings
ID
Course ID
Term
Location
Time
Faculty

- To find out courses taken by the Student.
  - Need another Table
    - course_offering_id (matches with CourseOffering.id)
    - student_id (matches with StudentInfo.id)
    - grade_received





ID	First Name	Last Name	Course Nam	Term	Grades
673688	Victor	Cruise	FIT	Winter 09	4.0
673688	Victor	Cruise	Mathematics	Fall 08	3.2
876554	John	Stevens	Mathematics	Fall 08	3.9