King Fahd University of Petroleum and Minerals

MIS-105--- Introduction to Computer Applications

Lab#10: Introduction to Microsoft Access

By: Syed Arshad Raza

A Relational Database

A database may contain only a single table. The real power of Access, however, is derived from multiple tables and the relationships among those tables. This type of database is known as a ***relational database***. The example of such a database is **Figure 1.8 on Microsoft Access Chapter 1 page#30.** The structure of the database with an additional table (Offices) is shown below (figure1).

EmployeeID	LastName	FirstName	LocationID	TitleID	Salary	Gender	Performance	OfficeNum

(a) The Employees table

LocationID	Location	Address	State	ZipCode	OfficePhone

(b) The Locations table

TitleID	Title	Description	EducationRequired	MinimunSalary	MaximumSalary

(c) The Titles table

OfficeNum	Building	Floor

(d) The Offices Table

Figure1

In such a database each table has a ***primary key (the underlined field)***, which is a field (or combination of fields) that uniquely identifies each record. The tables and their respective primary keys are listed below (see **figure2**).

Table Name	Primary Key
Employees	EmployeeID
Locations	LocationID
Titles	TitleID
Offices	OfficeNum

Figure2

One field, the **LocationID**, appears in both Employees and the Locations tables and links the two tables to one another. Similarly the **TitleID** appears in both the Employees and Titles tables to link those tables to one another.

Note: **In Employees table LocationID, TitleID and OfficeNum are the *foreign keys*.** The name of the ***primary key*** and the ***foreign key*** could be different in the two tables, but the **data types field size etc.** must be the same.

Source (Parent) and Destination (Child) Tables

While entering a record for a new employee, you have to look for the **LocationID** and **TitleID** from the Locations and Titles tables respectively. It means that you can not add a **LocationID** or a **TiltleID** until it already exists in either of the two tables. Similarly you cannot assign an office to an employee if it not in the **Offices Table**.
It means that **Locations, Titles** and **Offices** are the ***source tables*** for the Employees table, which is the ***destination table.***

Types of Relationships

One-to-many: The ***primary key*** in the ***source table*** can have ***zero, one*** or ***several occurences/instances*** in the ***destination table*** as a ***foreign key.*** This can be rephrased as

"A record in the source table (one table) can have zero, one or several related records in the destination (many) table."

e.g. There may be many Employees in ***Atlanta.***

Note: In **Figure1.8 (a)**, there are ***four*** occurrences of ***L01,*** LocationID for Atlanta**,** in the ***Employees Table.***

Such a relationship is shown as **1** on the ***source table*** side and a symbol ¥ **at** ***destination table***, as shown in **figure3**.

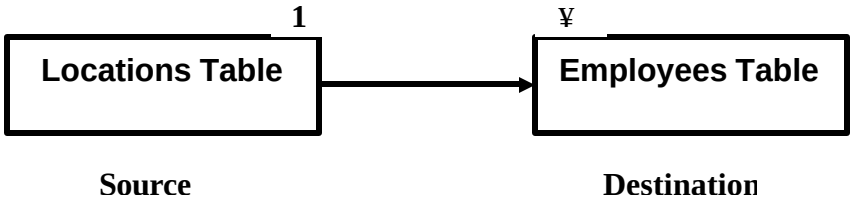

Figure3

One-to-one: The primary key in the ***source table*** can have ***zero*** *or* ***only one occurence/instance*** in the ***destination table*** as a ***foreign key.*** This can be rephrased as

"A record in the source table can have zero or one related records in the destination table."

e.g. there is another table in the same database called **Offices.** Suppose it is the company policy that an office can be assigned to **ONE** employee only. It means that for an office record in the **Offices table**, we can either find ***none*** or ***one*** record **at maximum**.

Such a relationship is shown as **1** on the source table side and a **1**, as shown in **figure4**.

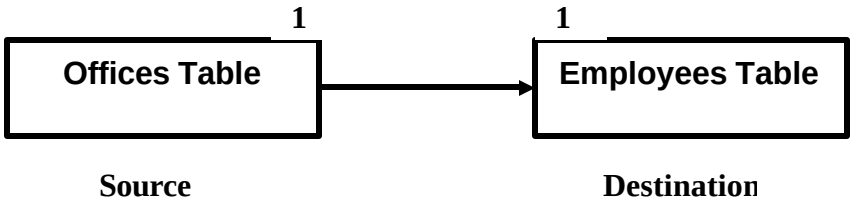

Figure4

Many-to-Many:

In this kind of relationship there could be ***zero or many related records*** in the other tables in both directions. For example consider a **University database**, having **STUDENTS** and **COURSES** tables. Considering the fact that **a student can take many courses and a course can be taken by many students**, it's a ***many to many relationship*** in both directions, as shown in **figure5.**

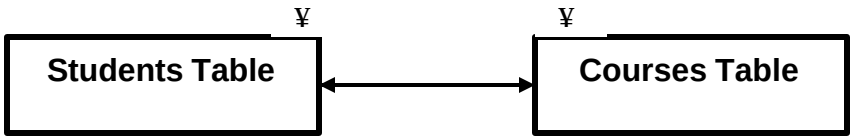

Figure5

Consider the following situation:

SudentID	Name	Major	CourseNo
980123	Ahmed	ICS	ICS101
980123	Ahmed	ICS	MIS105
980123	Ahmed	ICS	ECON101
980345	Ali	MIS	MIS105
980345	Ali	MIS	ICS101

(a) Students Table

CourseCode	CourseName	StudentNo
ICS101	Intro to Computers	980123
ICS101	Intro to Computers	980345
MIS105	Intro to Computer Appl	980123
MIS105	Intro to Computer Appl	980345
ECON101	Intro to Economics	980123

(b) Courses Table

Figure6

Note: **It is clear that Many to many relationship results in *data redundancy* which may also lead to *data inconsistency*.**

Solution: ***One many to many relationship*** **can be broken into *two one-to-many relationships* by introducing an intermediate table, as shown in figure7.**

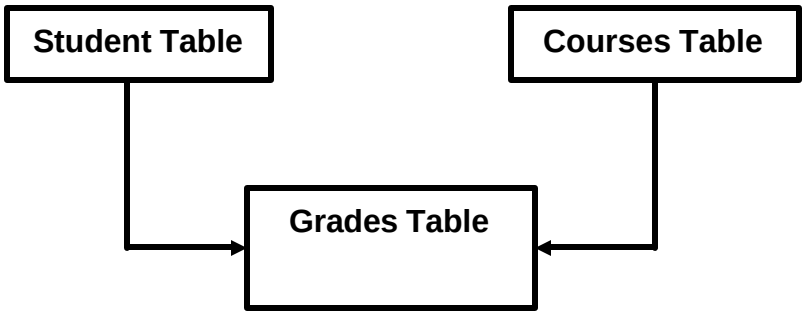

Figure7

The structure of the THREE tables now becomes:

StudentID	Name	Major

CourseCode	CourseName

StudentID	CourseCode	Grade

Figure8

The ***primary key*** of the intermediate table is the combination of the two fields **StudentID** and **CourseCode**. Such a primary key is known as ***composite primary key***.

How to make a composite Key in MS Access
Hold the ctrl key and select the fields.
Make a right click and choose the primary key option.

This intermediate table in addition to solving the redundancy problem can also be used to save some **additional information** such as a **student's grade** in a course, as shown in **figure8.**

Creating Relationships in MS Access

In MS Access the relationships are created in the Relationship window. Create the above relationships among the **FOUR** given tables as instructed by your instructor.

Multiple Table Queries

A query involving more than one tables for selection/updation/deletion is called a Multiple Table query.

1. List **Employee ID, Last Name, Address** and **Office Phone** of those employees who have not been assigned any office yet.

Hint: Use **null** in the criteria for the ***Office Number field***.

2. Select **EmployeeID, FirstName, Title, Salary, Address and Office Number** for the employees having their **office in Building 24.**

Summary Query

3. Design a query to **calculate the average salary of employees.**

Delete Query

4. Design a delete query which will ask you to enter an **Employee ID** in an interactive fashion and deletes his record from the **employees table**.

Hint: Write **=[Enter Employee ID]** in the criteria row.

Please go through the other types of queries like **Make-Table Query, Update Query, Append Query** in your lab book from **page 139 to 144** and try to apply them to this database.