



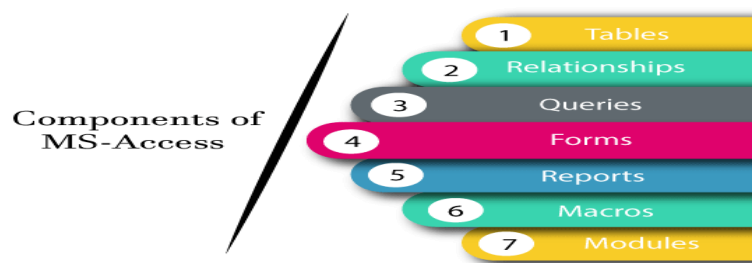
الجمهورية الجزائرية الديمقراطية الشعبية
وزارة التعليم العالي والبحث العلمي
جامعة وهران 2 محمد بن احمد



People's Democratic Republic of Algeria
Ministry of Higher Education and Scientific Research
University of Oran 2 Mohamed Ben Ahmed
Faculty of Economic Sciences, Commercial Sciences and Management Sciences
Department of Economic Sciences.

Course Handout :

Computer Sciences 2



Dr. KHELLAT KIHIL Badra

This course is intended for second year Economics students.

2024-2025

Foreword :

The objective of this course is to learn the basic elements of information systems and to acquire the necessary knowledge of databases management using Access.

At the end of this course students will:

- Learn the basics of Microsoft Access.
- Be able to create and manage databases under Access
- Know how to create forms.
- Know how to develop queries.

This course is intended for second year Economics students.

Table of Contents

- Foreword
- Table of contents

Chapter 1: The evolution of information systems and design methods

- **Part 1:** MERISE Method
- **Part 2:** The Conceptual Data Model (CDM)
- **Part 3:** The Logical Data Model (LDM)

Chapter 2 : Overview of Microsoft Access

1. Introduction
2. Launch Microsoft Access
3. Elements of the Access 2013 Window
4. Manipulating Databases
5. Elements of a Database

Chapter 3 : Database table creation

1. Definition of a Table
2. Data types
3. Different Operations on Tables
4. Data Entry and Editing in Datasheet View
5. Relationships

Chapter 4 : Creating database queries

1. Definition of a Query
2. Creating a Query

Chapter 5: Creating database forms

1. Definition of a Form
2. Methods of Creating a Form
3. Creating a Form Using the Wizard
4. Modifying the Form's Structure
5. Adding a Label
6. Adding a Command Button
7. Subforms

Chapter 6 : Creation of database reports

1. Definition of a report
2. Creating a report
3. Creating a Report Using the Wizard
4. Report View Modes
5. Report Components

Chapter7 : Creating Macros for a Database

1. Definition of a Macro
2. Creating a macro
3. Attaching a Macro to Form Button
4. Running a macro
5. Operations on Macro Actions
6. Controlling Program Flow with If, Else If, and Else Conditions
7. Creating Submacros
8. Sharing a Macro with Others
9. Converting a Macro into VBA Code

Practical work sheets with solutions

- Practical work sheet 1
- Practical work sheet 2
- Practical work sheet 3
- Practical work sheet 4
- Practical work sheet 5
- Practical work sheet 6
- Practical work sheet 7

Bibliography

Part 1: MERISE Method

1. Introduction

The information system (IS) is now a central element in the functioning of an organization. An information system can be defined as a set of resources (personnel, software, processes, data, hardware, computing and telecommunications equipment, etc.) that enables the collection, storage, structuring, modeling, management, manipulation, analysis, transport, exchange, and dissemination of information (texts, images, sounds, video, etc.) within an organization. Among the computing resources are particularly data files, databases, and database management systems (DBMS).

Designing an information system is not straightforward, as it requires considering the entire organization that needs to be set up. The design phase necessitates methods that allow for the establishment of a model on which to base the work. Modeling involves creating a virtual representation of a reality in such a way as to highlight the points of interest.

Therefore, modeling languages or models are necessary. The MERISE method is one of the most widely used in IS design.

2. Presentation of MERISE method

Derived from systemic analysis, the MERISE method (**M**éthode d'**E**tude et de **R**éalisation Informatique pour les **S**ystèmes d'**E**ntreprise) is the result of work conducted by Hubert Tardieu in the 1970s, which was part of an international reflection, particularly around the relational model. It became an operational project in the early 1980s at the request of the Ministry of Industry and has been used in France by major public and private administrations.

The MERISE method is a methodology for designing, developing, and implementing IT projects. It is based on the separation of data and the processes to be performed into several models. This separation ensures the longevity of the model. Indeed, the organization of data does not need to be frequently modified, while processes are more often subject to changes.

3. The Approaches of the MERISE Method

MERISE is characterized by a dual approach: by construction stages and by levels of abstraction.

3.1. Approach by Construction Stages

MERISE proposes a six-step approach as follows:

A. Master Plan

In this stage, a comprehensive study of the information system (IS) will be conducted:

- Development of a strategy regarding the IS
- Definition of the study areas
- Planning for the development of each area
- Evaluation of the human and financial resources needed for each area.

The result of this stage is the master plan.

B. Preliminary Study:

This focuses on a representative subset of the studied area, encompassing the most important activities. It is carried out in three phases:

- **Collection:** This involves redefining the mission and studying the existing situation.
- **Design:** This involves defining organizational and technical solutions.
- **Evaluation:** This consists of determining implementation scenarios for the proposed solutions and quantifying costs, services provided, risks, and stakes.

The result of this stage is the preliminary study document.

C. Detailed Study:

This stage allows for:

- Specifying the detailed organization of the chosen solution.
- Logically defining the data and the IT processes of the solution.
- Defining the interfaces: screens, output reports.
- Constructing the implementation schedule.

The result of this stage is a functional specifications document or user requirements specification.

D. Implementation:

This consists of producing programs according to the specifications from the detailed study, including testing and validation. It is carried out in three phases:

- **Technical study:** This aims to achieve a physical description of the data and processes.
- **Program production:** This involves developing the programs and conducting tests.
- **Commissioning:** Installation of the IT application.

E. Maintenance:

This involves making corrections and adaptations to the software. It addresses the changes that arise after operational launch.

F. Quality Control:

This involves verifying the conformity of the results to the requests.

3.2 Approach by Level of Abstraction

MERISE uses a three-level modeling approach. At each level corresponds a model for the representation of data and a model for the representation of processes, with a formalism of representation associated with each model. The MERISE method distinguishes three levels of abstraction:

A. Conceptual Level:

This is the highest level of abstraction. It includes the most stable elements. It describes the classes of objects and the significant rules based on the objectives set by decision-makers.

At the conceptual level, the questions to be answered are: what? What do we want to do?

With what data?

At this level, two models are developed:

- The conceptual data model (CDM).
- The conceptual process model (CPM).

B. Logical/Organizational Level:

This represents the resources used to support the descriptions from the conceptual level. The logical/organizational level takes into account organizational issues based on the conceptual data models, incorporating organizational constraints.

At this level, two models are elaborated:

- The logical data model (LDM).
- The organizational process model (OPM).

C. Physical/Operational Level:

This provides a physical representation of the data and an operational representation of the processes, considering constraints and technical choices.

At this level, two models are developed:

- The physical data model (PDM).
- The operational process model (OPM).

The abstraction levels with their resulting models are summarized in the following table:

LEVEL	DATA	TREATMENTS
Conceptual <i>Management choice: What?</i>	Conceptual Data Model (CDM)	Conceptual Model of Treatments (CTM)
Organizational <i>Organizational choice: Who? Where? When?</i>	Logical Data Model (LDM)	Organizational Model of Treatments (OMT)
Physical <i>Technical choices: How?</i>	Physical Data Model (PDM)	Physical Model of Treatments (PMT)

4. Advantages and Disadvantages of the MERISE Method

4.1 Strengths:

- The method is based on a systemic approach: it is therefore a global approach.
- The concepts are few and simple.
- It is quite independent of technology.
- It is the most widely used in management fields.

4.2 Weaknesses

- It does not address user interface design.
- It does not allow for rapid validation by users.
- It is very difficult to validate processes against data at both the conceptual and organizational levels.

Part 2: The Conceptual Data Model (CDM)

1. Introduction

The Conceptual Data Model (CDM) is the most well-known and widely used element of the Merise method. It allows for a clear representation of data and the relationships between data in the studied domain.

A CDM has several purposes:

- It serves as a communication tool: communication with the client, within a development team, with users, etc.
- It is a normalization tool: transitioning from implicit and verbal rules to written management rules, integrating parallel systems, etc.

To construct data models, Merise offers tools. Here, we focus on a necessary tool to aid in the construction of the Conceptual Data Model (CDM): the data dictionary. Indeed, before developing the CDM, the data must be collected and assembled into a table called the data dictionary.

2. Data dictionary

The expression of needs is a step that involves defining what is expected from the automated information system. To do this, it is necessary to:

- Inventory the elements needed for the information system
- Define the scope of the system by consulting future users

Note:

When establishing the data dictionary, terms that do not necessarily qualify as data may slip in. These can include data processing or information flows.

Some expressions may be polysemous, covering multiple meanings, while others may be synonymous. These issues can be addressed in the subsequent steps of creating the CDM. In general, the aim is to create a comprehensive data dictionary.

Each term in the data dictionary is called a Property. The next step is to group the properties into entities.

3. The Conceptual Data Model (CDM)

The Conceptual Data Model (CDM) aims to formally describe the data that will be used by the information system. It is a representation of the data that is easily understandable, allowing for the description of the information system using entities and associations. Entities are the concrete or abstract objects of the real world, while associations represent the links between these entities. Entities and associations can be identified by performing a discourse analysis, which involves sentences of the type "subject-verb-complement." The subjects and complements are the entities, and the verb models the association.

The representation of the entity-association model is based on three basic concepts:

- The object or entity;
- The association;
- The attribute.

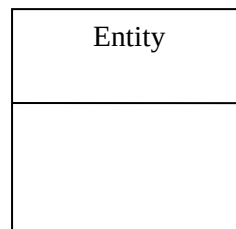
3.1. Entities

An entity is the representation of a tangible or intangible element that plays a role in the system to be described. An entity class is a set composed of entities of the same type, meaning their definitions are the same. An entity is an instantiation of the class. Each entity is made up of properties, which are elementary data that describe it.

Entity classes are represented by a rectangle. This rectangle is divided into two sections:

- The upper section contains the label. This label is usually an abbreviation for the sake of simplifying the writing. It is essential to ensure that each entity class corresponds to one and only one label, and vice versa.
- The lower section contains the list of properties of the entity class.

Graphical representation



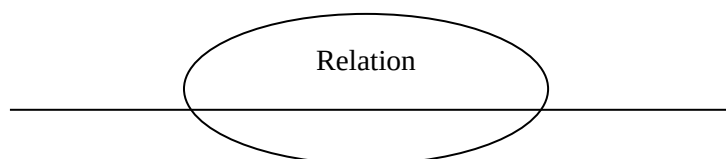
3.2. Relations

A relation (sometimes referred to as an association) represents the semantic links that can exist between multiple entities. A relation class thus contains all relations of the same type. A relation class can connect more than two entity classes. Here are the names of relation classes based on the number of participants:

- A recursive (or reflexive) relation class connects the same entity class.
- A binary relation class connects two entity classes.
- A ternary relation class connects three entity classes.
- An n-ary relation class connects n entity classes.

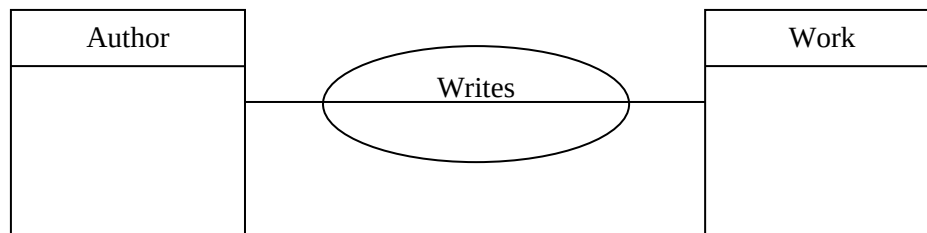
Relation classes are represented by ellipses, with the label describing the type of relation that links the entity classes (usually a verb). An identifier of the form R_i is defined for each relation class to uniquely designate the relation class to which it is associated. Properties can optionally be added to relation classes.

Graphical Representation



Example

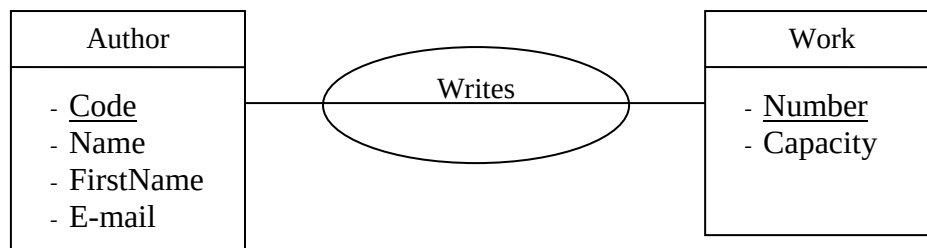
An author writes a work



3.3. Properties or Attributes

Properties or attributes are the characteristics that describe the entities and should be represented as a simple list of words within the corresponding entity framework. The properties of the entity are indicated in the lower rectangle, below the name of the entity.

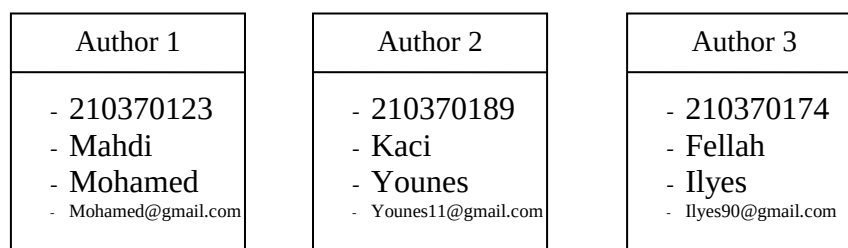
Example



3.4. Occurrence

The occurrence of a property or attribute is one of the values that this property can take.

Example



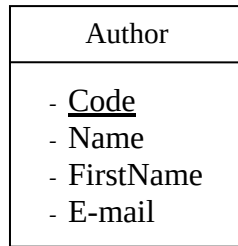
3.5. Identifiers

An identifier is a set of properties (one or more) that uniquely designates a single entity. The original definition is as follows:

An identifier is a specific property of an object such that there do not exist two occurrences of that object for which this property could take the same value.

Examples of identifiers: The social security number for a person, the registration number for a car...

Graphical Representation



3.6. Cardinality

Cardinalities allow us to characterize the link that exists between an entity and the relation to which it is connected. The cardinality of a relation consists of a pair comprising a maximum limit and a minimum limit, which defines the interval within which the cardinality of an entity can take its value:

- The minimum limit (usually 0 or 1) describes the minimum number of times an entity can participate in a relation.
- The maximum limit (usually 1 or n) describes the maximum number of times an entity can participate in a relation.

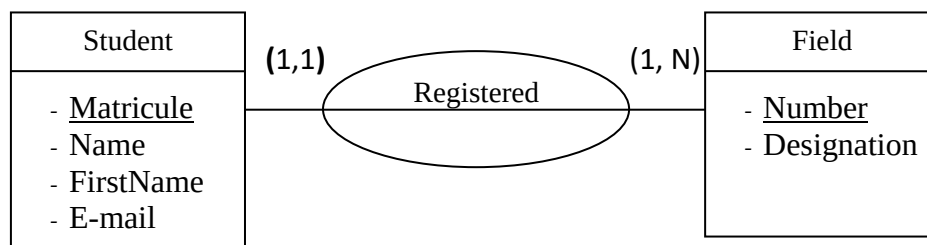
A cardinality of 1.N means that each entity belonging to an entity class participates at least once in the relation.

A cardinality of 0.N means that each entity belonging to an entity class does not necessarily participate in the relation.

There are four types of association:

- (0,1): An entity from A may not be linked, but it may be linked to a single entity from B.
- (1,1): An entity from A is linked to a single entity from B.
- (0, N): An entity from A may not be linked, but it may be linked to multiple entities from B.
- (1, N): An entity from A may be linked to one or more entities from B.

Exemple



4. Normalization of the CDM

We will discuss data models that comply with the 1st, 2nd, or 3rd normal form. In a relational database, a normal form refers to a particular type of relationship between entities. As its name indicates, "normal" describes a standard. The normalization of data models will thus ensure the robustness of the database design to improve modeling (and therefore achieve a better representation) and optimize data storage by avoiding redundancy and issues related to updates or consistency. Normalization applies to all entities and to relations that carry properties.

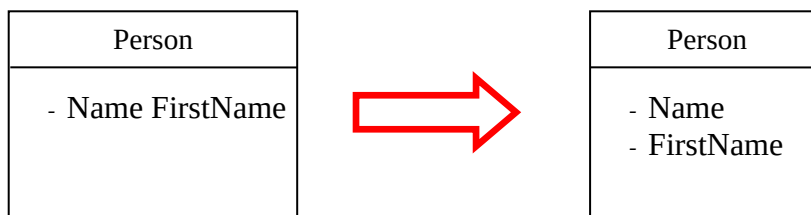
A model is said to be in the first, second, or third normal form. These are the most important normalization rules; other rules may be applied but are not systematic.

4.1. First Normal Form (1NF):

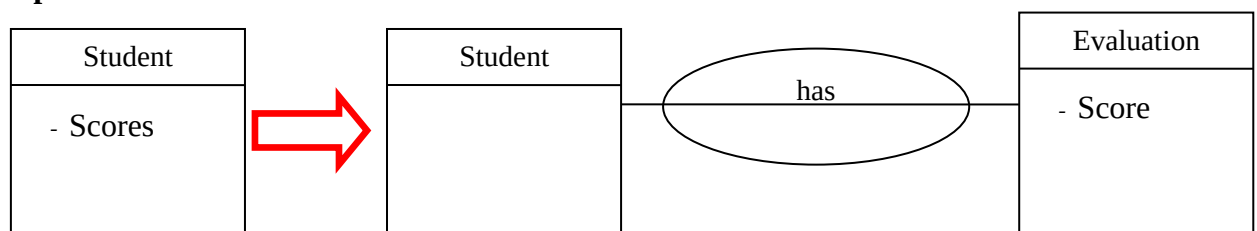
A relation is in the first normal form if all its attributes:

- Contain atomic values.
- Contain non-repetitive values.
- Are constant over time.

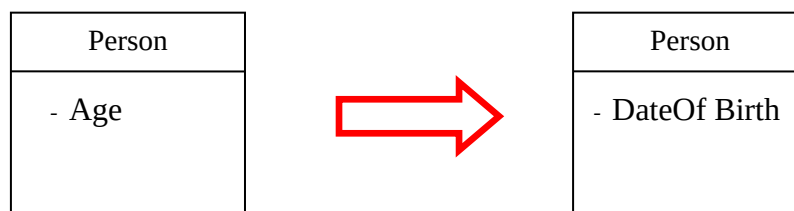
Example 1



Example 2



Example 3



4.2. Second Normal Form (2NF)

A relation is in the second normal form if it complies with the first normal form and if:

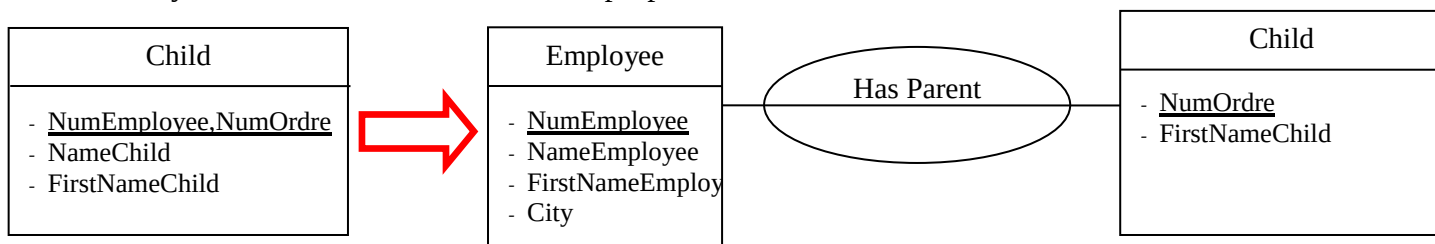
- All non-key attributes depend on the entirety of the primary key, not just a part of it.

Exemple

Consider an information system related to personnel management in a factory, with:

- There are no female employees in the factory.
- A child is associated with exactly one employee.

The entity Child is not in 2NF. Here is the proposed solution:

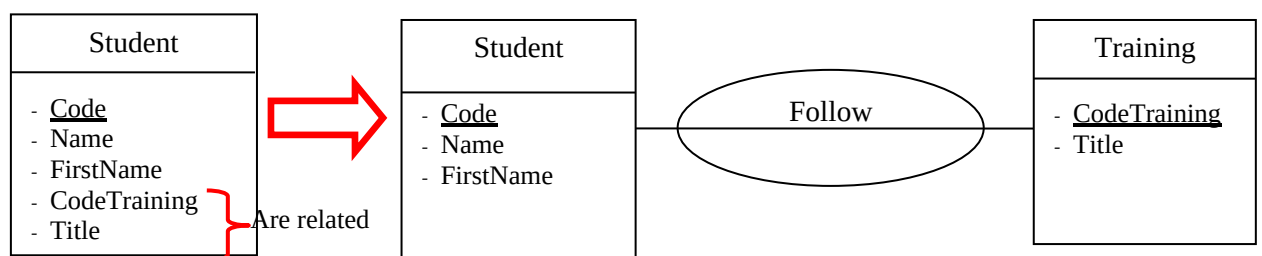


4.3 Third Normal Form (3NF)

A relation is in the third normal form if it complies with the second normal form and if:

- All attributes that do not belong to the key do not depend on a non-key attribute. In other words, the functional dependency is direct.

Exemple



5. Case Study :

The creation of a CDM can be carried out in three steps. We will address these steps through a simple case study. The domain under study is a Purchasing department.

5.1. Constitution of the data dictionary

This step involves referencing all the data in the domain. For our Purchasing department, we can identify the following elements:

- Supplier name
- Order date
- Order number
- Stock quantity of the Product
- Product reference
- Unit price of the Product
- Product description
- Quantity ordered
- Supplier Phone number
- Supplier address

5.2. CDM: Entities

In our case, three entities seem to emerge:

- The Supplier entity
- The Product entity
- The Order entity

Once these entities are defined, we associate them with the properties from the data dictionary.

Supplier	Product	Order
<ul style="list-style-type: none">- Name- FirstName- Phone	<ul style="list-style-type: none">- Reference- Price- Description	<ul style="list-style-type: none">- Number- Date

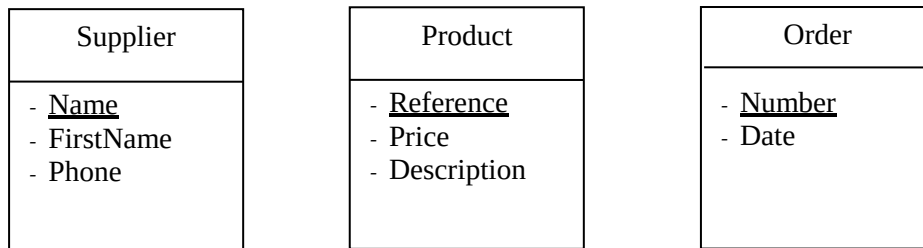
A property cannot belong to multiple entities. Here, for example, the price of an Product is a property of the Product entity. However, since this price is negotiated based on the order, it will be necessary to have two price properties: one property for Product Price in the Product entity and one property for Order Price in the Order entity.

Each time a property is associated with an entity, it is noted in the data dictionary.

- **S** Supplier name
- **O** Order date
- **O** Order number
- Stock quantity of the Product
- **P** Product reference
- **P** Unit price of the Product
- **P**Product description
- Quantity ordered
- **S** Supplier Phone number
- **S** Supplier address

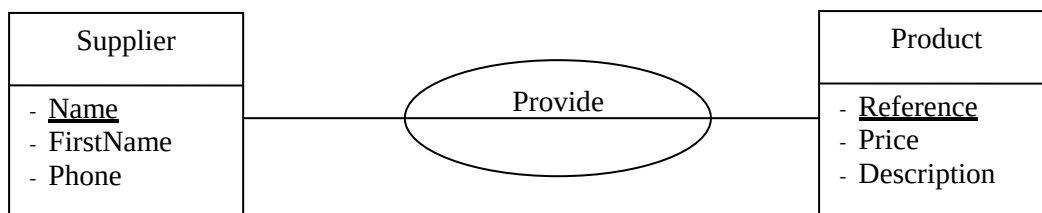
We notice that some elements in the data dictionary do not appear in the entities. This can be due to several reasons: Some elements are not within the studied domain. This is the case for the **stock quantity of the Product**. It is indeed a data point, but it pertains to **inventory management** rather than the **Purchasing department**.

To complete the creation of the entities, each entity must have an identifier. An identifier allows for the discrimination of each occurrence of an entity. Therefore, each identifier will have a unique value. For example, for the Product entity, the identifier is the Product Reference, as two different Products cannot have the same reference.

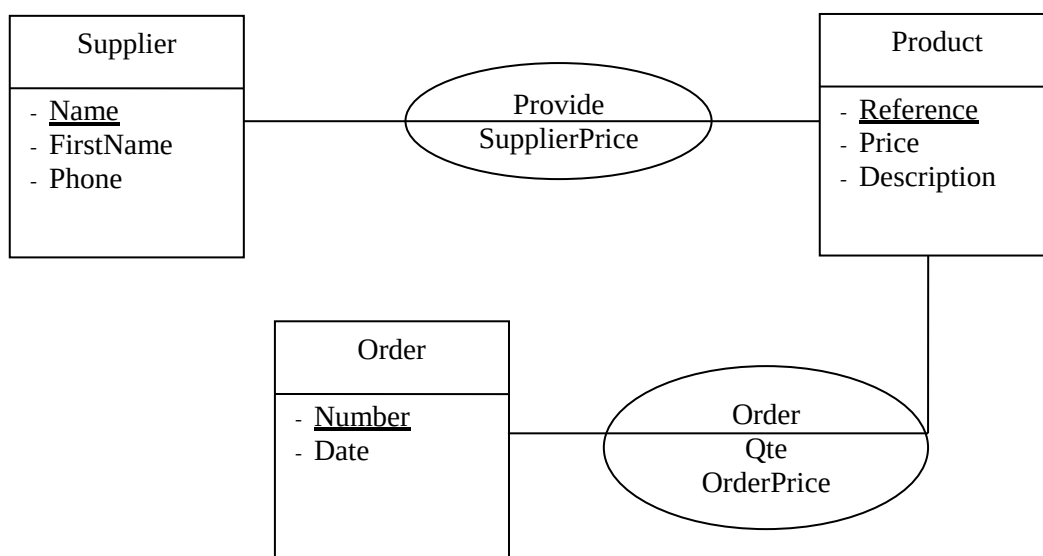


5.3. CDM: Relations

A relation is a link between different entities. These links are established by asking the question: "Which entity interacts with this or these entities?" For example, a supplier provides a Product, and a Product is supplied by a supplier. Indeed, there is interaction between the Supplier and Product entities, which is represented by the action of Providing.



Similarly, an order consists of Products, and the Products belong to an order. The relationship between the Order and Product entities is represented by the action of Ordering.

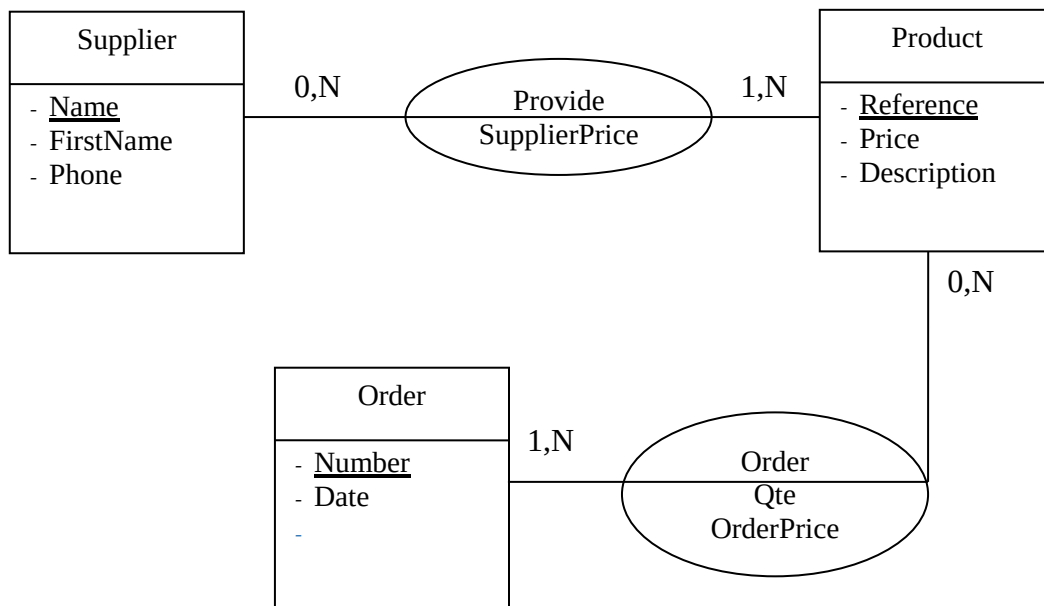


A relation can have properties. This is the case here with the property Quantity, which is a property of the Ordering relation.

Similarly, the ability for relations to have properties will allow us to create new data to clarify the ambiguity of the property Price. Indeed, the property Unit Price is associated with the Product entity, the property Order Price is associated with the Ordering relation, and the property Supplier Price is associated with the Providing relation.

5.4. CDM : Cardinality

Our conceptual data model becomes:



For example, the cardinality (0,n) of the Supplier entity for the Providing relation means that a supplier can provide at least zero Products and at most n Products. Similarly, the cardinality (1,n) of the Product entity for the Providing relation means that an Product is supplied by at least one supplier and at most n suppliers.

Part 3: The Logical Data Model (LDM)

1. Introduction

The conceptual description has allowed for a faithful representation of the realities of the problem to be computerized. However, this representation cannot be directly manipulated or accepted by a computer system, as the MCD (Conceptual Data Model) is a representation of data in a formalism understood by designers, not by machines. Therefore, it is necessary to transition from the conceptual level to a second level that is closer to the capabilities of computer systems. This level is called the logical level.

2. The Logical Data Model

A logical data model is the representation of the data of an information system designed for implementation within a relational database management system (RDBMS). Data is represented in the form of tables and relationships between those tables. The logical model builds on the conceptual model while addressing the constraints of data organization. It allows us to anticipate access times and the necessary storage space.

The MCD (Conceptual Data Model) is the most abstract representation we create of the data structure in an information system. It consists of entities that represent sets of data of the same nature and associations between those entities.

The LDM (Logical Data Model) is a representation that takes into account the technological choices for the future database. It serves as an intermediary step for transitioning from the Entity-Relationship model, which is a semantic model, to a physical representation of the data (database files).

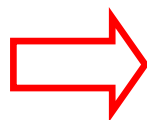
3. The rules for converting from CDM to LDM

3.1.Entities to Tables

Each entity class in the conceptual model becomes a table in the logical model. The identifiers of the entity class are called table keys, while the standard attributes become table attributes, that is, columns.

Exemple:

Student
- <u>StudentID</u>
- Name
- FirstName
- E-mail



Student (StudentID, Name, FirstName, Email)

Student			
StudentID	Name	FirstName	Email
20229513	Bouazza	Mohamed	BouazzaMohamed@gmail.com
20227894	Tazi	Amina	TaziAmina@gmail.com
.....			

3.2.Translation of a relationship class

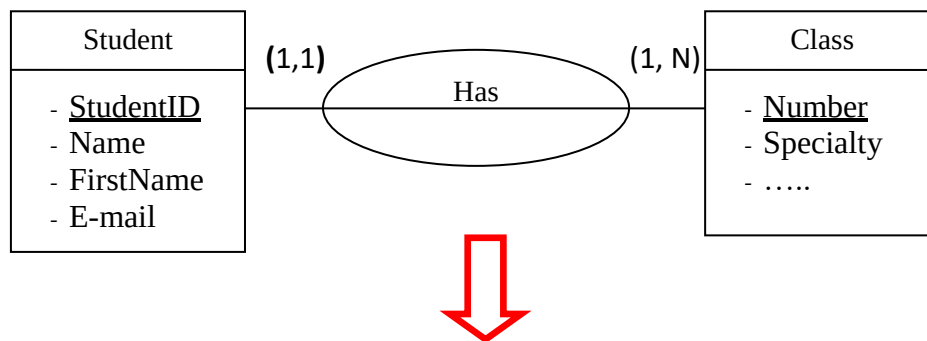
The transition from the conceptual model to the logical model at the level of relationship classes is done according to the cardinalities of the entity classes participating in the relationship:

3.2.1. Binary association (x , 1) – (x , n)

The entity with a cardinality of type 1,1 or 0,1 absorbs the identifier of the stronger entity (0,n or 1,n). This identifier is then called the foreign key. In other words, a foreign key (identifier of the entity with cardinality 0,n) is added to the table derived from the entity whose cardinality is 1,1.

Note: x can take on values of 0 or 1.

Exemple :



Student (StudentID, #Number, Name, FirstName, E-mail)

Class (Number, Specialty)

Student				
StudentID	Number	Name	FirstName	Email
20229513	2	Bouazza	Mohamed	BouazzaMohamed@gmail.com
20227894	1	Tazi	Amina	TaziAmina@gmail.com
.....				

Class	
Number	Specialty
1	EGE
2	EMF
.....	

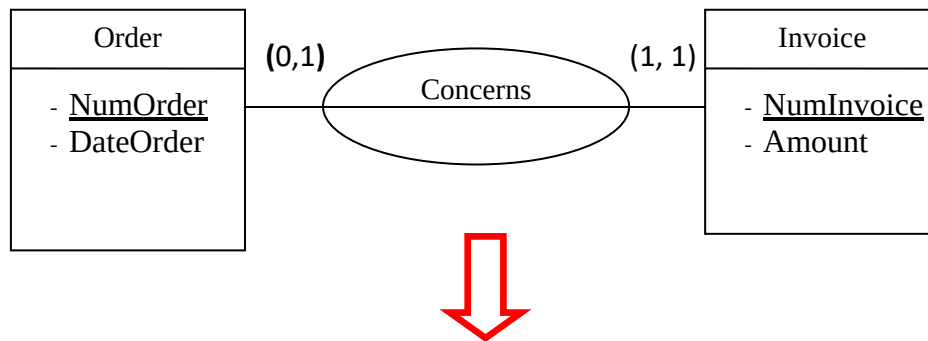
3.2.2. Binary association (x , 1) – (x , 1)

- **Binary association (1 , 1) – (0 , 1)**

The entity with a cardinality of type (1,1) absorbs the identifier of the entity (0,1). This identifier is then called the foreign key.

In other words, a foreign key (identifier of the entity with cardinality (0,1)) is added to the table derived from the entity whose cardinality is (1,1).

Example :



Invoice (NumInvoice, #NumOrder, Amount)

Order (NumOrder, DateOrder)

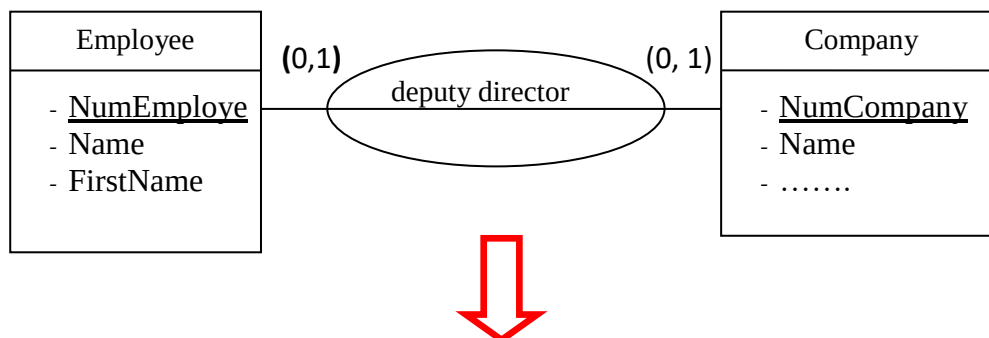
Invoice		
NumInvoice	NumOrder	Amount
001	1	23000.00
002	2	4500.00
.....		

Order	
NumOrder	DateOrder
1	12/11/2022
2	20/11/2022
.....	

- **Binary association (0 , 1) – (0 , 1)**

In this case, we duplicate the key from one table into the other. When the relationship itself has properties, these properties also become attributes of the table in which the foreign key has been added.

Example :



Employee (NumEmployee, #NumCompany, Name, FirstName)

Company (NumCompany, Name,)

OR

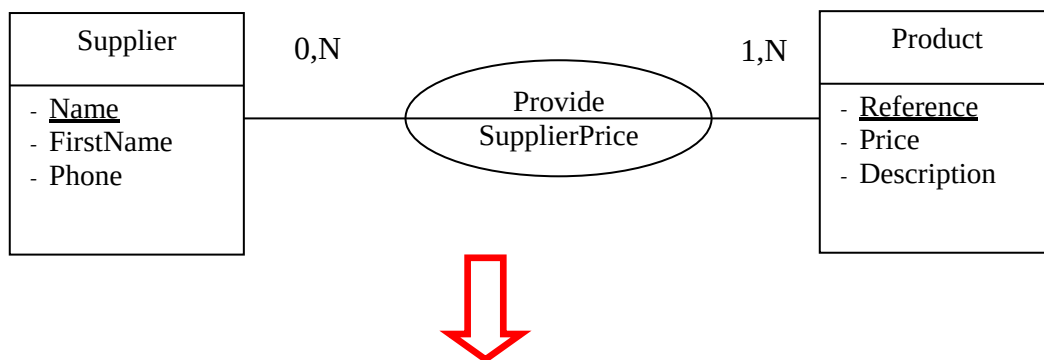
Employee (NumEmployee, Name, FirstName)

Company (NumCompany, # NumEmployee, Name,)

3.2.3. Binary association (x , n) – (x , n)

Here, x can take the value of 0 or 1. In this case, we create an additional table with a primary key that is a composite key made up of the primary keys from the two tables. When the relationship itself has properties, these properties become attributes of the additional table. A property of the relationship that is underlined should be included in the composite primary key of the additional table.

Example :



Supplier (Name, FirstName, Phone)
 Product (Reference, Price, Description)
 Provide (Name ,Reference, SupplierPrice)

Supplier		
Name	FirstName	Phone
Bouazza	Mohamed	041 41 42 43
Tazi	Amina	041 23 25 26
.....		

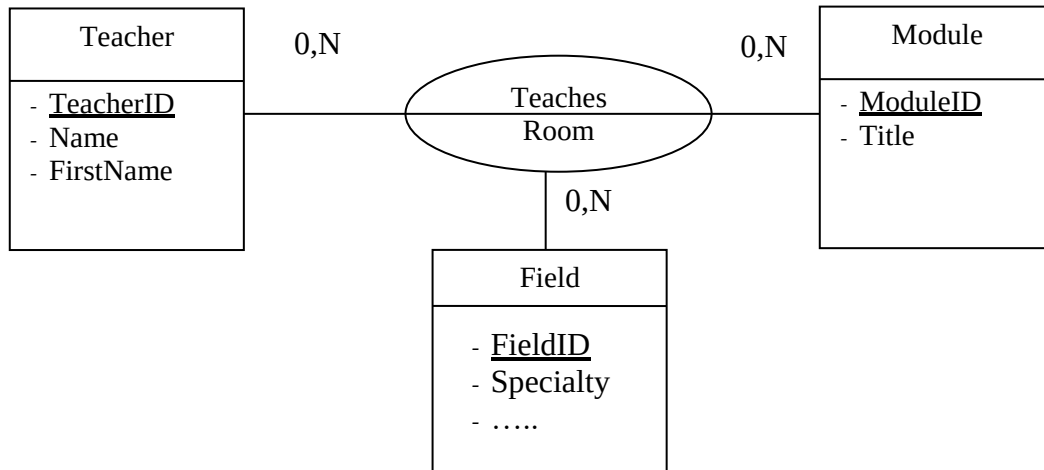
Product		
Reference	Price	Description
1	12000	Ecran
2	6500	Chaise
.....		

Provide		
Name	Reference	SupplierPrice
Bouazza	2	6000
Bouazza	5	8600
.....		

3.2.4. N-ary Association

A supplementary table is created with a primary key composed of the primary keys of all the connected tables. This rule applies independently of the different cardinalities. When the relationship itself contains properties, these become attributes of the supplementary table. A property of the relationship that is underlined must belong to the composite primary key of the supplementary table.

Example :



Teacher (TeacherID, Name, FirstName)

Module (ModuleID, Title)

Field (FieldID, Specialty)

Teache (TeacherID, ModuleID, FieldID, Room)

Teacher		
TeacherID	Name	FirstName
1	Bouazza	Mohamed
2	Tazi	Amina
.....		

Module	
ModuleID	Title
1	Mathématiques
2	Informatique
.....	

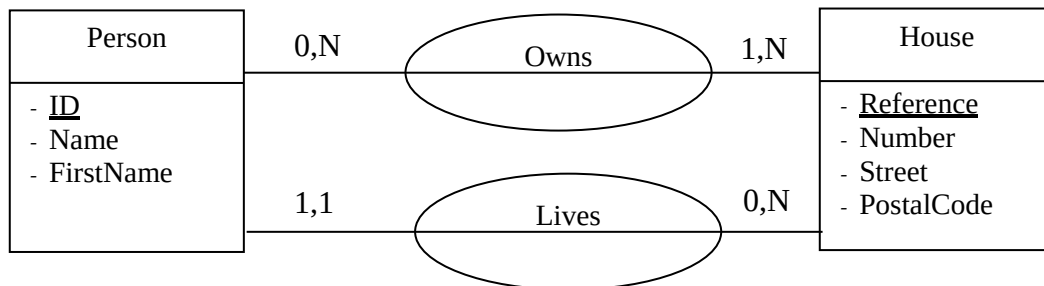
Teache			
TeacherID	ModuleID	FieldID	Room
1	1	2	110
2	1	2	109

Field	
FieldID	Specialty
1	Economie
2	Gestion
.....	

3.2.5. Transformation of Multiple Relationships Between Two Entities

If multiple relationships exist between two entities, they are handled separately. The general rules apply. Each association is treated independently of the others, which may result in the addition of multiple references.

Exemple :



Person (ID, Name, FirstName, #Reference)
House (Reference, Number, Street, PostalCode)
Owns (ID, Reference)

4. Conclusion

According to the concepts presented in this chapter, Merise is an approach to building information systems characterized by a dual approach: by level of abstraction and by construction stage. The Merise modeling process is based on the separation between data modeling and process modeling. Regarding data modeling, Merise uses the conceptual data model, which is based on the fundamental concepts of the Entity-Association model. Subsequently, it transforms the results obtained into a logical data model known as the relational model.

1. Introduction

Access is an RDBMS (Relational Database Management System) that allows for the management of large volumes of data. It enables users to create databases more quickly and manage them more easily. Additionally, Access provides tools that facilitate communication and collaboration.

2. Launch Microsoft Access

There are several ways to launch Microsoft Access:

- Go to the Start Menu → All Programs → Microsoft Office 2013 and click on the Microsoft Access 2013.
- Click on the Microsoft Access shortcut on the desktop if it exists.
- Click on the Microsoft Access icon in the Quick Launch area of the taskbar if it exists.

When you start Access, you will see a window that looks like this:

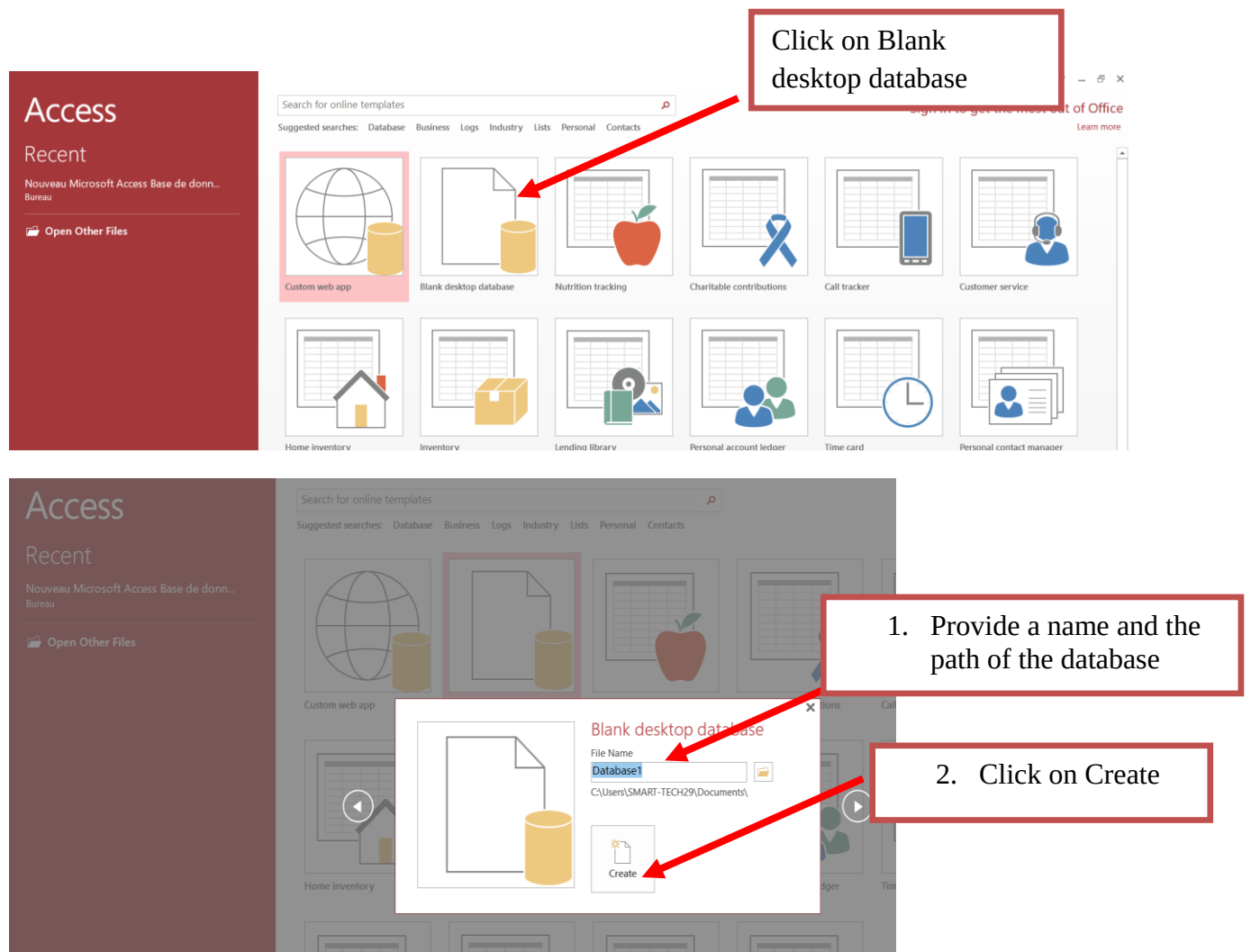


Figure 2.1: Launching and the Interface of Access 2013

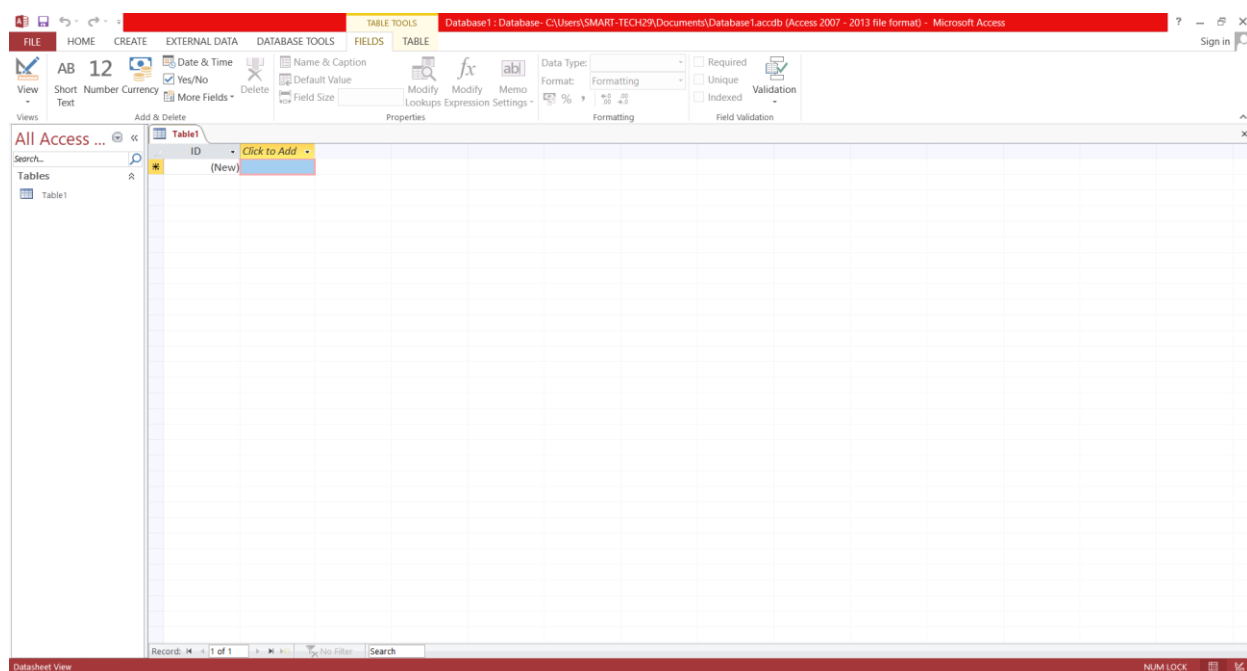


Figure 2.2: The Access 2013 User Interface

3. Elements of the Access 2013 Window

The three main components of the Access 2013 user interface are:

- **The Ribbon:** This is the tabbed toolbar at the top of the program window that contains groups of commands.
- **Backstage View:** This refers to the set of commands available in the File tab of the Ribbon.
- **The Navigation Pane:** This is the pane on the left side of the Access program window that allows you to work with database objects. The Navigation Pane replaced the database window in Access 2007.

3.1. The Ribbon

The Ribbon is the primary element replacing the menus and toolbars from earlier versions prior to Access 2007. It consists mainly of tabs containing groups of buttons. The Ribbon contains main tabs that group commonly used related commands, contextual tabs that appear only when relevant, and the Quick Access Toolbar, a small toolbar that you can customize with your favorite commands.

Under the Ribbon tabs, some buttons provide a gallery of choices, while others execute a command. The Ribbon serves as the main tool replacing menus and toolbars; it provides the primary command interface for Access 2013. One of its main advantages is that it offers a centralized location for tasks or entry points that previously required displaying menus, toolbars, Office panes, and other UI components. This single space thus gives access to all commands. When you open a database, the Ribbon appears at the top of the main Access window, and the commands are displayed under the active command tab.

Chapter 2 : Overview of Microsoft Access

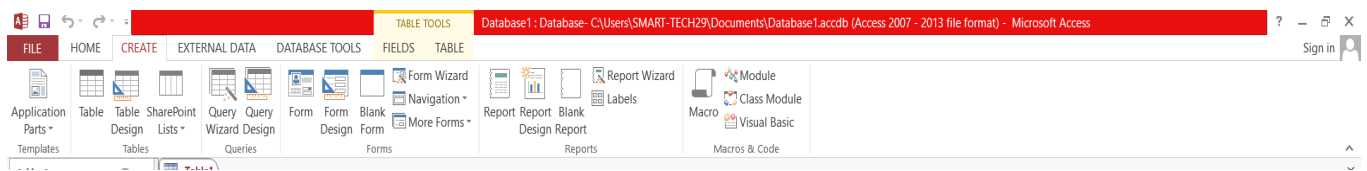


Figure 2.3 : The Ribbon in Access 2013

The Ribbon contains a series of command tabs that hold commands. In Access 2013, the main command tabs are: File, Home, Create, External Data, and Database Tools.

To execute a command in Access, click on the tab containing the desired command. The table below illustrates representative examples of tabs and commands available in each tab. These tabs and commands vary based on your tasks.

- **The Home Tab**

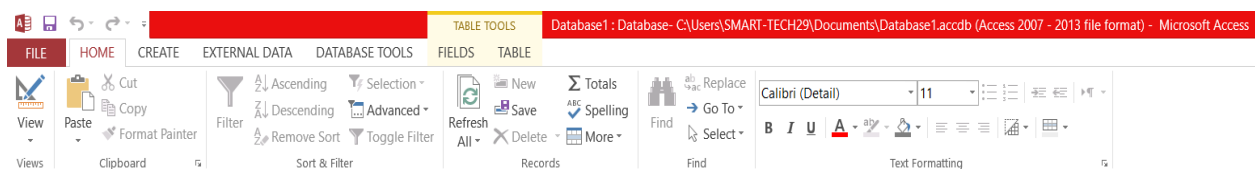


Figure 2.4: The Home Tab

Command Tab	Common Actions
Home	Select a different view.
	Copy and paste data from the clipboard.
	Set the current font characteristics.
	Set the current font alignment.
	Use record operations (Refresh, New, Save, Delete, Totals, Spelling, More).
	Sort and filter records.
	Find records

- **The Create Tab**

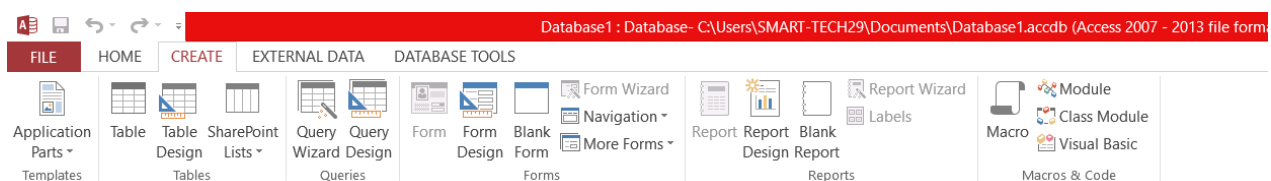


Figure 2.5 : The Create Tab

Create	Create an empty table.
	Create a table using a table template.

Create a list on a SharePoint site and a table in the active database that links to the newly created list.

Create an empty table in Design View.

Create a new form based on the active table or query.

Create a report based on the active table or query.

Create a new query, macro, module, or class module.

• The External Data Tab

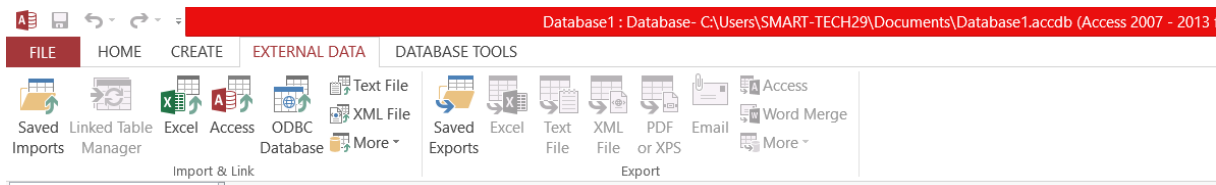


Figure 2.6 : The External Data Tab

External Data

Import or link to external data.

Export data.

Gather and update data via email.

Create saved imports and exports.

Run the Attachment Manager.

• The Database Tools Tab

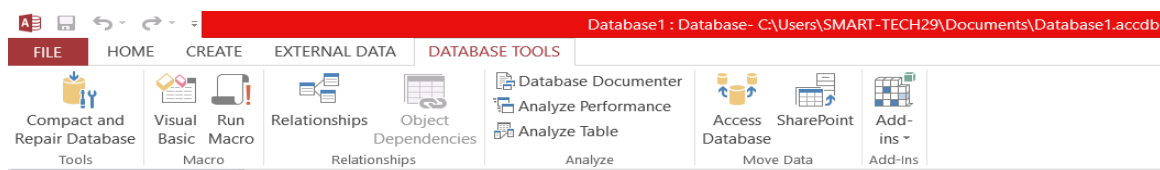


Figure 2.7 : The Database Tools Tab

Database Tools

Compact a database.

Launch Visual Basic Editor or run a macro.

Create and display table relationships.

Show/hide object dependencies.

Run database documentation or analyze performance.

Move some or all sections of a database to a new or existing SharePoint site.

Move data to Microsoft SQL Server or to an Access database (tables only).

Manage Access add-ins.

Create or modify a Visual Basic for Applications (VBA) module.

You may sometimes need to enlarge the workspace area. To do this, you can minimize the Ribbon to keep only the bar with the command tabs visible. To hide the Ribbon, double-click on the active command tab. To show it again, double-click on the active command tab once more.

- Double-click on the active command tab (the active tab is the one that is highlighted).
- Double-click on the active command tab to restore the Ribbon.

3.2. Quick Access Toolbar

The Quick Access Toolbar is a small area next to the Ribbon that allows you to access commands with a single click. The default set of commands includes **Save**, **Undo**, and **Redo**. You can customize the Quick Access Toolbar to include other frequently used commands, change its location, and replace it with the larger toolbar (by default, the small toolbar is displayed). The small toolbar appears next to the command tabs in the Ribbon. If you choose the larger toolbar, it will appear below the Ribbon and extend the full width of the window.

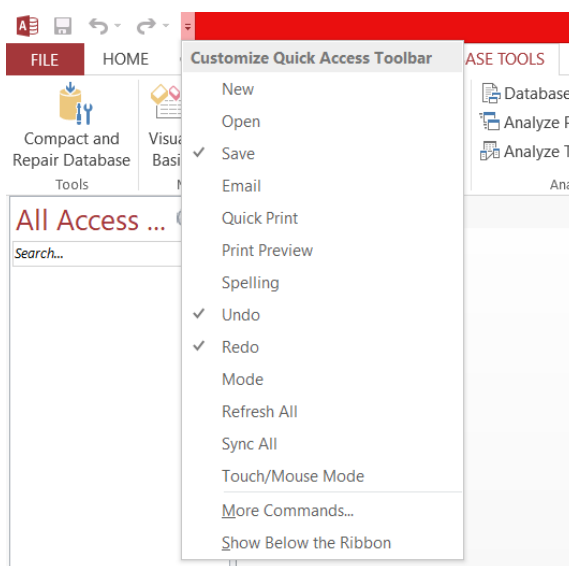


Figure 2.8 : Quick Access Toolbar

3.3. Navigation Pane

When you open a database or create a new one, the names of the objects it contains are displayed in the Navigation Pane. Database objects include tables, forms, reports, pages, macros, and modules. The Navigation Pane replaces the Database Window used in earlier versions of Access. For example, if you want to add a row to a table in Datasheet View, open that table in the Navigation Pane.

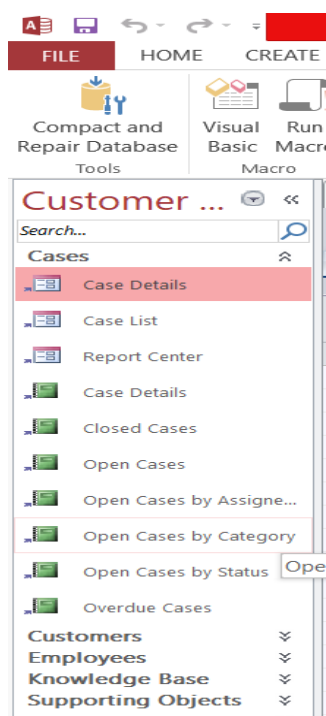


Figure 2.9 : Navigation Pane

3.4. Status Bar

In earlier versions of Access, such as Access 2010, you could display a status bar at the bottom of the window. This standard UI (User Interface) element shows status messages, property indicators, progress indicators, and other information. In Access 2013, the status bar also offers two standard functions found in the status bars of other Office 2013 programs: Datasheet View and Design View.

You can quickly switch from the active window to any of the available views using the controls on the status bar. You can enable or disable the status bar in the **Access Options** dialog.

To show or hide the status bar:

1. Click on the **File** tab, then click on **Options**.
The **Access Options** dialog appears.
2. In the left pane, click on **Current Database**.
3. In the **Application Options** section, check or uncheck the box for **Display Status Bar**. If you uncheck it, the status bar will no longer be visible.
4. Click **OK**.

3.5. Backstage View

The Backstage View occupies the **File** tab on the Ribbon and contains many commands that were found in the **File** menu of earlier versions of Access. It also includes other commands that apply to an entire database file.

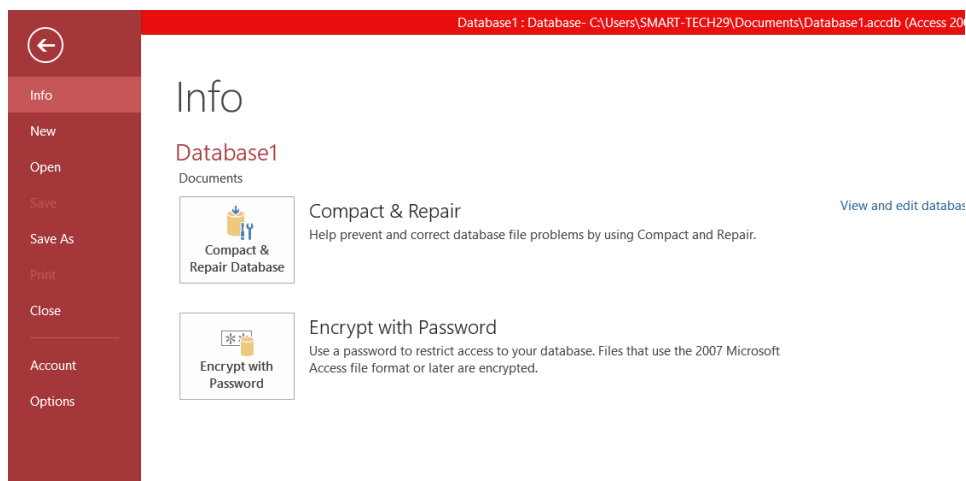


Figure 2.10: Backstage view

In Backstage View, you can create a database, compact and repair databases, and encrypt them with a password.

4. Manipulating Databases

4.1. Open an Existing Database

There are several ways to open an existing database:

- From Windows Explorer, simply navigate to the location of your database and double-click on it.
- Open a database from the File menu:
 - Start Access.
 - Under the File tab, click on Open.
 - In the Open dialog box, locate the database you want to open.
 - Double-click on the database to open it.

4.2. Creating a Database

When you start Access, you will see a window that looks like this:

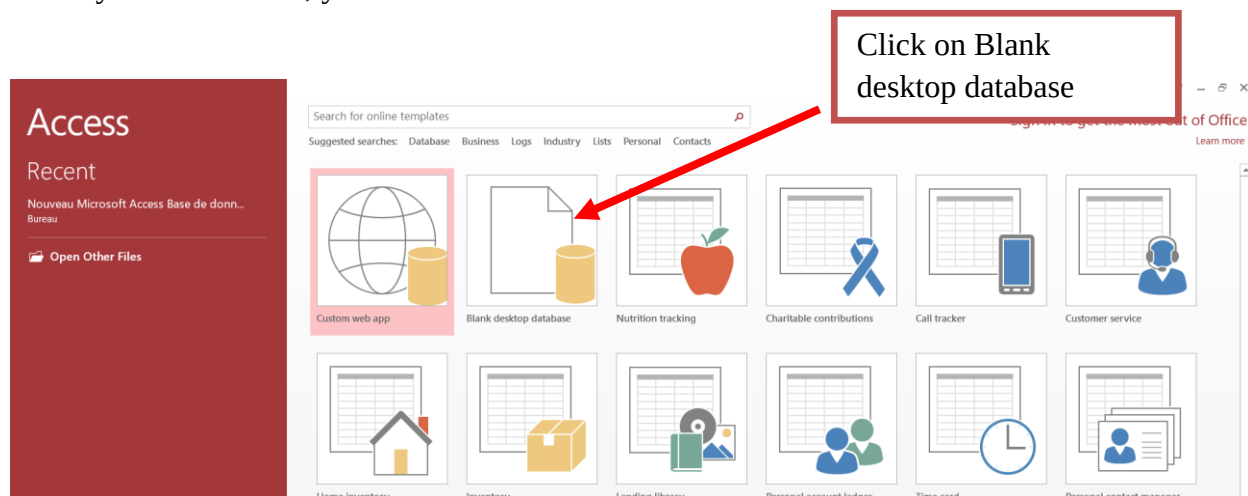


Figure 2.11.a: Steps to Create a Blank Database

Chapter 2 : Overview of Microsoft Access

You will now decide on the name of your database, as well as the location on your computer where you will place it.

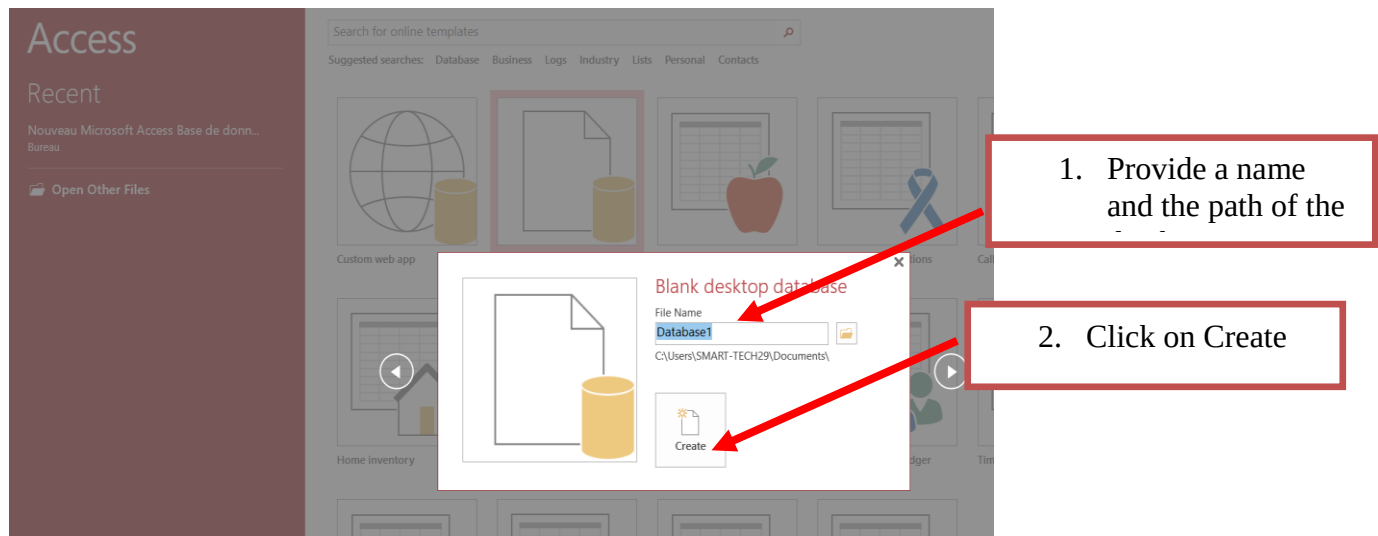


Figure 2.11.b: Steps to Create a Blank Database

Now, here's what your screen should look like:

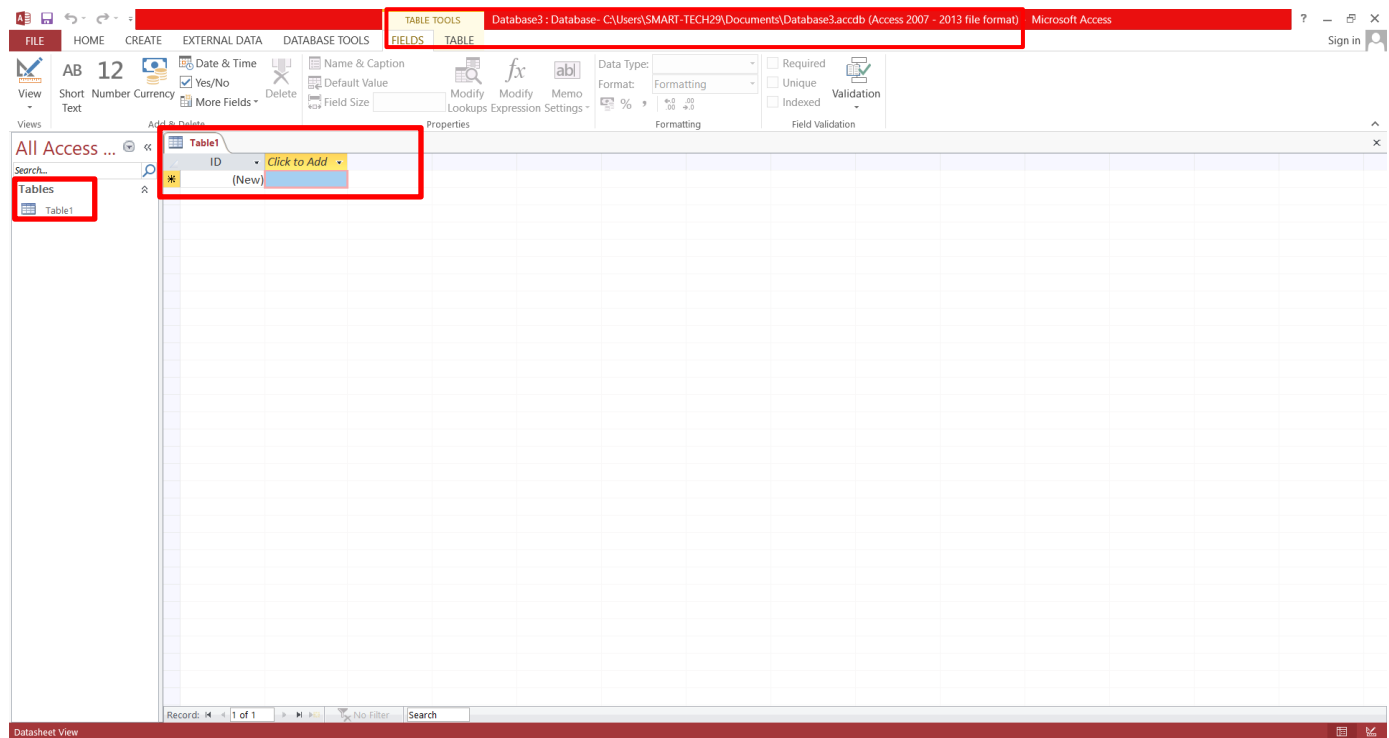


Figure 2.12: Database Creation Interface

4.1. Compact a Database

To reduce the size of the database to its original size, you need to compact it.

Click on Database Tools, then Compact Database:

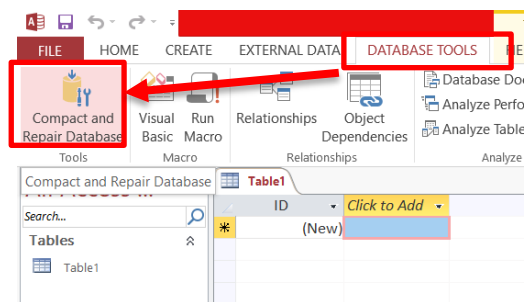


Figure 2.13 : Compact a Database Menu

➤ Automatic Compacting

You can set Access to automatically compact your database every time you close it. Click on **File/Options/Current Database/Compact on Close**.

This option is not checked by default because if you are working with small databases, compacting will have little effect. However, when dealing with large databases, it is beneficial to compact them to reduce their size.

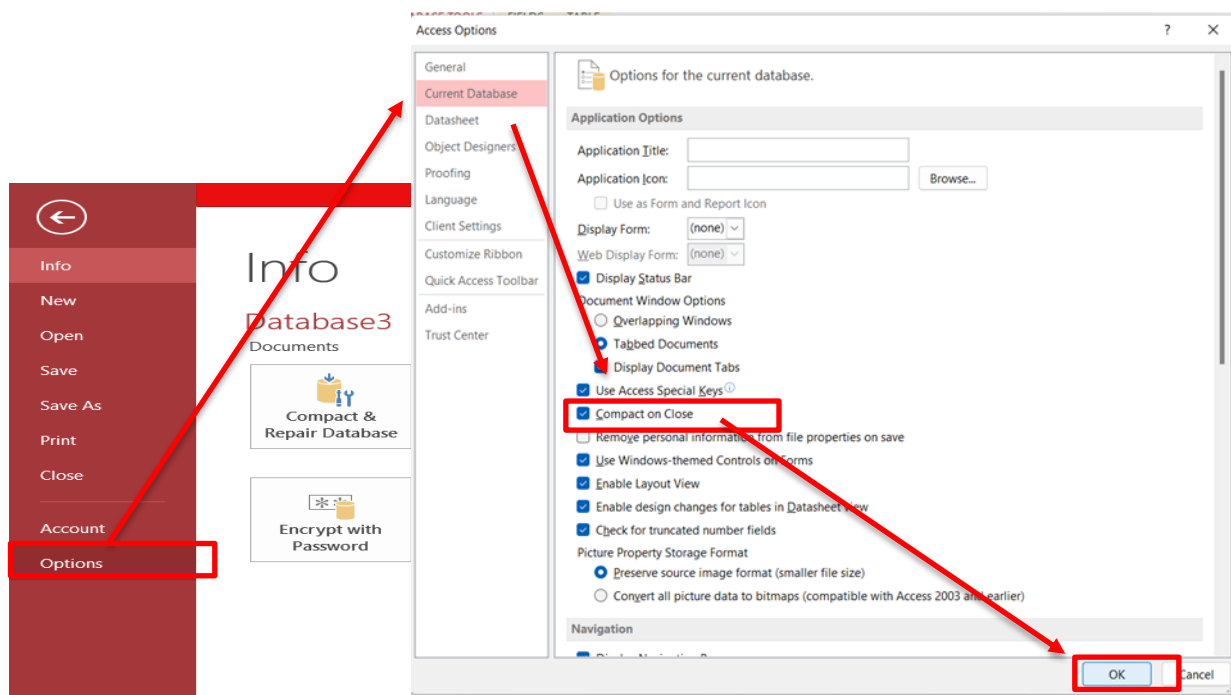


Figure 2.13 : Automatic Compacting Menu

5. Elements of a Database

Microsoft Access offers a set of objects, namely:

- **Tables:** Used to store information. These are files containing a set of information about a common theme or concept.
- **Queries:** Used to filter data based on specific criteria. They also allow actions to be performed on this data (calculations, modifications, deletions, etc.).
- **Forms:** Allow for the entry and modification of information in tables in a simpler manner than in tables (including checkboxes, text boxes, images, etc.).

- **Reports:** Used to print data and allow for presenting the same data file in different ways.
- **Macros:** Automate certain actions by programming command buttons.
- **Modules:** Allow for more advanced programming than macros.
- **SQL Language:** The universal programming language that allows manipulation of objects and data in databases.

5.1. A table

A table is a collection of data related to a specific subject. Tables organize data into columns (FIELDS) and rows (RECORDS). Each field contains a piece of information about a record. Each record contains all the information related to a specific entry.

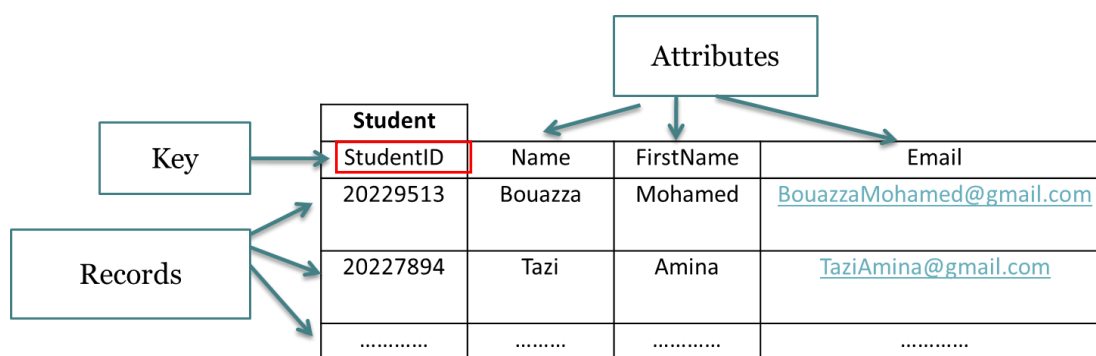


Figure 2.14: A table in Access

5.2. A Query

A query is a question posed to the database. The answer is provided in a dynamic response sheet. You can:

- Display related data from multiple tables.
- Modify values in the query datasheet.
- Update the table.

5.3. A Form

A form is used to display and modify information in the database record by record. The data usually comes from a table or a query. It consists of CONTROLS such as text boxes, checkboxes, and dropdown lists. You can also add graphical elements. To create a form, you can use wizards that will set up the default controls, which you can then customize.

5.4. A Report

Reports are used to display and print information from your database. You can perform groupings at multiple levels and establish calculations. The data in a report typically comes from a table or a query.

5.4. A Macro

A macro is a set of automated actions. These allow you to automate tasks and can be attached to controls, forms, and reports.

5.5. A Module

Modules are units of code written in Access Basic. You can write and use modules to automate and customize the database in a highly tailored way.

1. Definition of a Table

A table is a database object that you use to store data related to a specific subject, such as employees or products. A table consists of records and fields. Each record contains data about an instance of the table's object, such as a specific employee. A record is also referred to as a row or an instance. Each field contains data about an aspect of the table's object, such as first name or email address. A field is also called a column or an attribute.

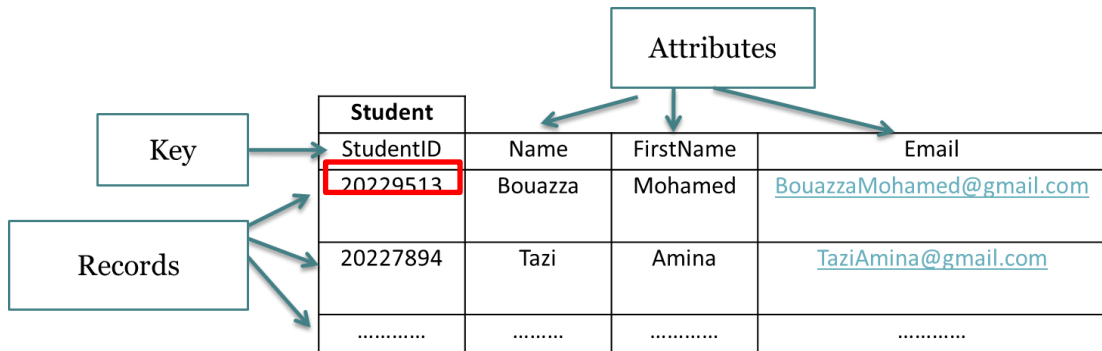


Figure 3.1: A Table in Access

An Access database can contain multiple tables, each storing information about a different subject. Each table can contain several fields of different data types, such as text, numbers, dates, and hyperlinks.

2. Data types

2.1. Different data types

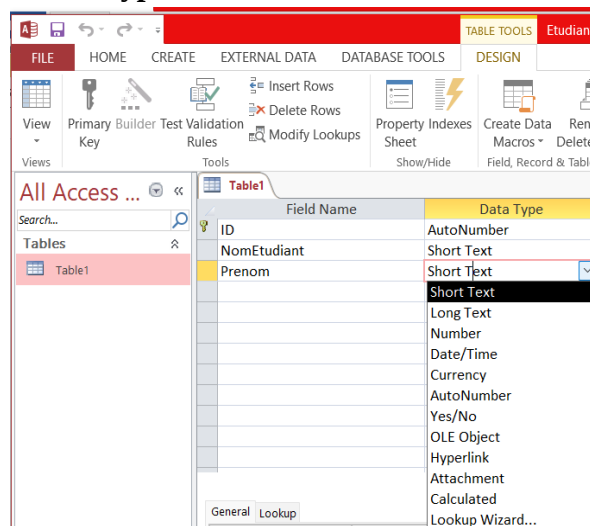


Figure 3.2: Data types in Access

- **Short Text:** Text fields can contain from 0 to 255 characters.
- **Long Text:** Useful for entering comments. The size of the field is not fixed. The maximum is 65,535 characters.

- **Number:** Numeric fields contain numbers, but the exact type of data is determined by the field's *Field Size* property. Additionally, the *Field Size* property offers several internal representations. It is simplest to use Long Integer for integer values and Double Precision for values that may contain decimals.
- **Date/Time:** Date/Time fields can contain date or time data. It is preferable to use a field of this type instead of a text field because Access automatically checks the validity of entries and allows calculations between dates and/or times.
- **Currency:** Currency fields store numeric values related to monetary amounts. The difference from numeric fields is that the number of decimal places is fixed, which can help avoid rounding errors.
- **Auto Number:** AutoNumber fields are really of *numeric* type and have a *Long Integer* size. The AutoNumber option, as the name suggests, automatically assigns a new value to each new record added to the table. The values can be assigned sequentially or randomly. Once assigned, the value cannot be modified. This field type is useful for automatically and uniquely identifying each record in the table and can be used as a *primary key*.
- **Yes/No :** Yes/No fields contain logical (Boolean) values, represented as Yes or No, True or False, -1 (Yes) or 0 (No).
- **OLE Object:** OLE Object fields store data from other Windows applications that support OLE (Object Linking and Embedding). This can include images, sound, clips, spreadsheets, or word processing documents.
- **Hyperlink:** Hyperlink fields contain an address pointing to a document or a web page.

2.2. Field Properties

Below are the characteristics or properties of the text field.

Field size	Determines the number of characters that can be contained in a field. This property is only valid for fields of the Text type. Other field types already have a predetermined size.
Format	Determines how the information will be displayed in the field.
Input mask	Determines how the information will be entered into the field.
Caption	Text that will be displayed in the datasheet, forms, and reports instead of the field name.
Default value	Determines an initial value when adding new records.
Validation rule	Sets boundaries or limits on the type of information that can be entered into a field.
Validation text	Displays this message if the content of the field does not meet the restrictions of the Validation Rule property.
Required	If active, prevents the field from having empty content.
Allow zero lenght	Allows the insertion of spaces (" ") in a field.
Indexed	This property allows the information to be sorted in ascending order when displayed. It is also required when creating relationships with other tables.

These last properties are available for fields of the **Text** type. Here are some additional properties available for other field types.

Decimal places (Number, Currency)	Determines the number of decimal places that will be retained with the number.
New value (Auto Number)	Determines whether the next number will follow the sequence (Increment) or be a random number (Random).

3. Different Operations on Tables

3.1. Building a Table in Design View

A database is composed of tables, queries, forms, and reports. Microsoft recommends that a database should contain at least one table.

In the Create ribbon, click on Table Design:

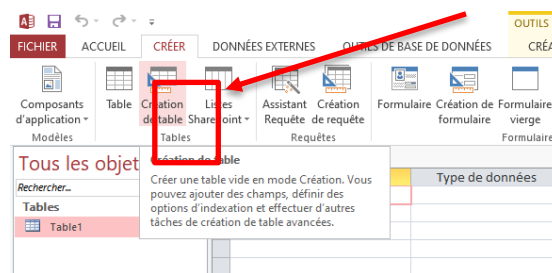


Figure 3.3: Create Tab

You will get this display.

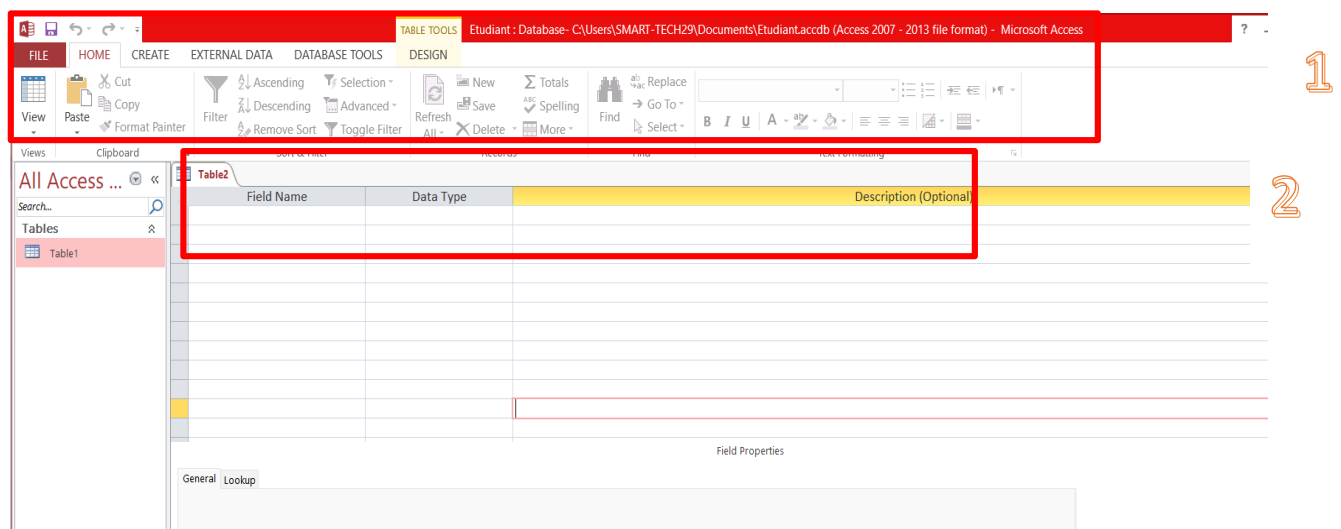


Figure 3.4: Create a table

1: A contextual ribbon has appeared. It appears with the yellow title Table Tools. It contains all the buttons (tools) necessary for managing the table. This ribbon is only useful in this context and disappears when you exit the table.

2: This is the table we just created in Design View.

3.2. Adding a Table

Follow the same procedure to create a new table.

3.3. Saving a Table

Click on the **Save** button and name the new table in the following dialog box:

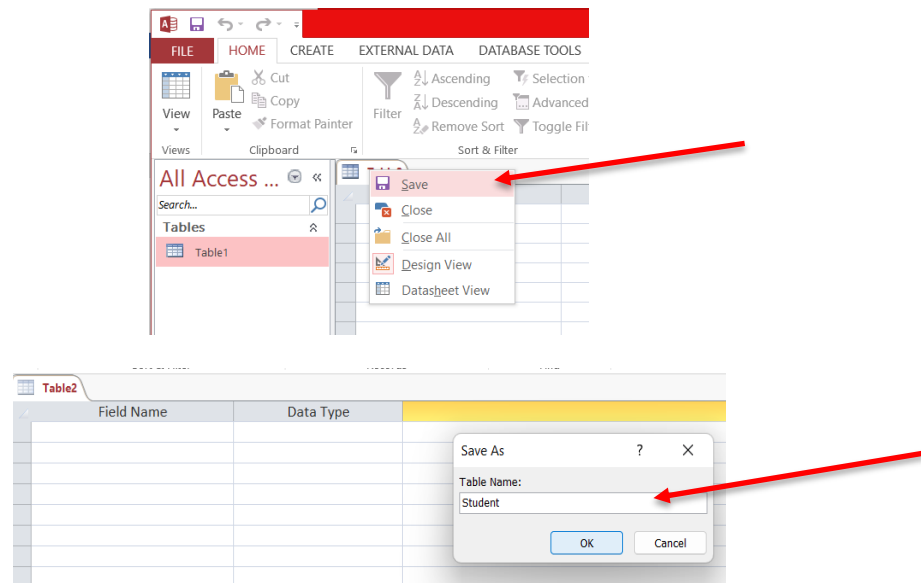


Figure 3.5: Save a table

The name of the new table will appear in the title bar of the Table window and will be added to the list of table objects in the navigation pane.

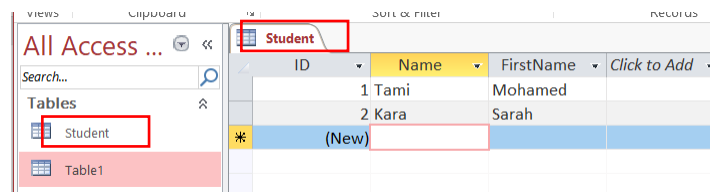


Figure 3.6: Table display after saving

3.4. Closing a Table

To close the table window, right-click on the table tab, then click Close.

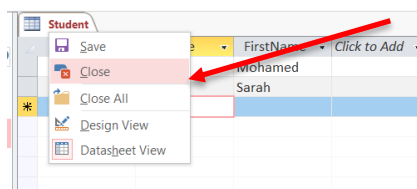


Figure 3.7: Close a table.

3.5. Deleting a Table

From the navigation pane:

- Select the table to delete from the list of objects.
- Click on the Delete button (or right-click on the table and then click Delete).

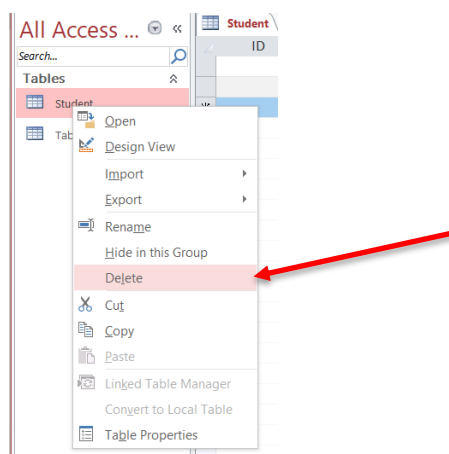


Figure 3.8: Deleting a table.

3.6. Renaming a Table

From the navigation pane:

- Right-click on the table you want to rename in the list of objects.
- Choose Rename and type the new name for the table.

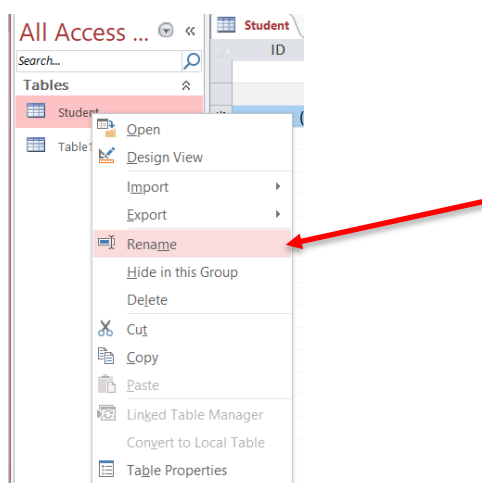


Figure 3.9: Rename a table.

3.7. Defining Fields in a Table

Complete the following window in Design View:

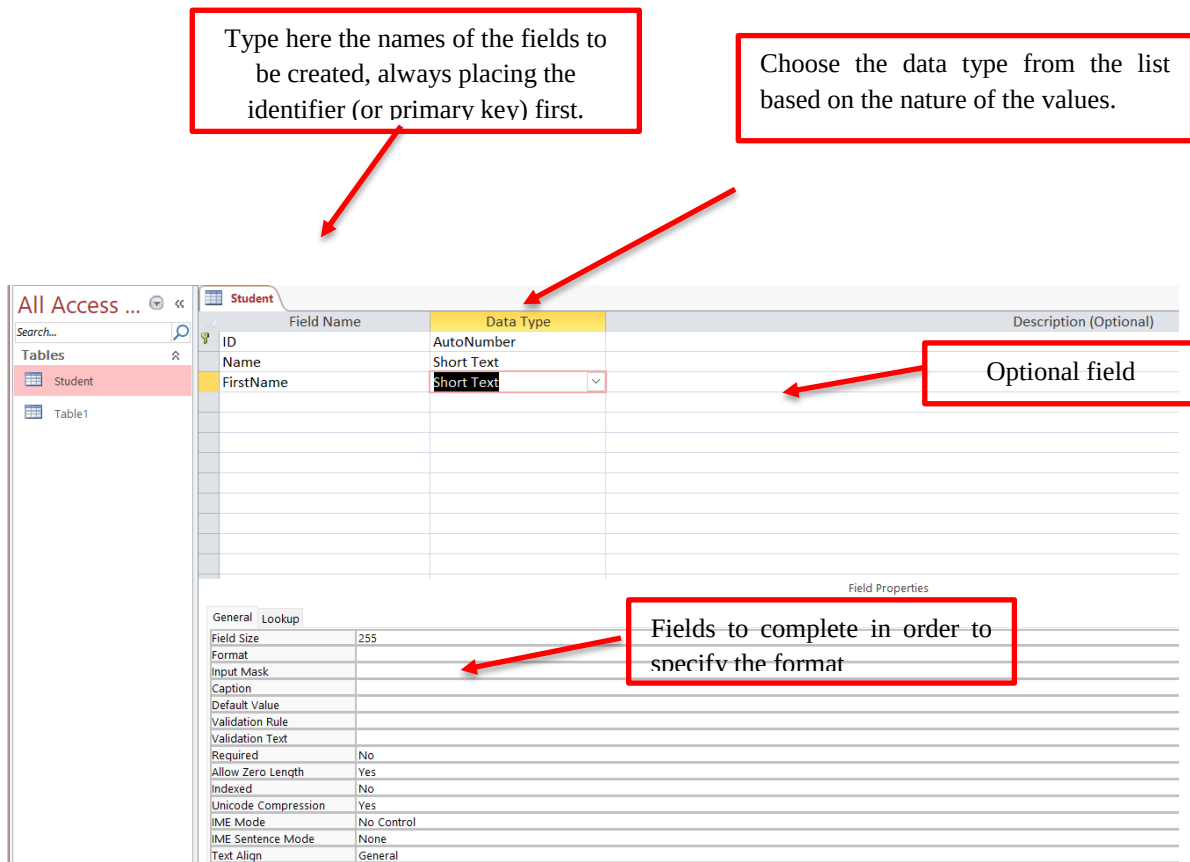


Figure 3.10: Defining Fields in a Table.

3.8. Defining the Primary Key

To define the primary key:

- Click on the field header that you want to set as the primary key.
- Click on the Primary Key button in the toolbar; a symbol (a key) should appear next to the field, indicating it is the primary key.

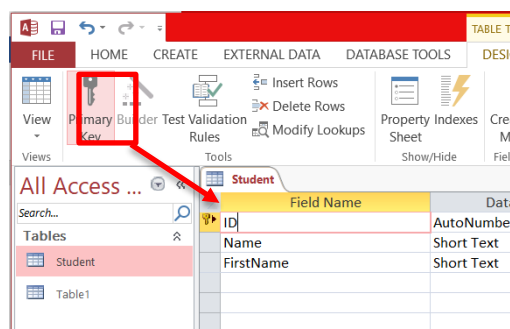


Figure 3.11: Defining the Primary Key.

3.9. Manipulating Fields in a Table

➤ Adding a field

From the Design View window: Add the field to the list, define its format, and then save.

➤ Renaming a field

From the Design View window: Modify the field's name and then save.

➤ Deleting a field

From the Design View window:

- Right-click on the field and delete the row.
- After the field is deleted, save the table.

➤ Moving a field

- Select the field by clicking on its header.
- Drag the field to the desired position using the mouse.

➤ Modifying a field

Make changes in the field properties window, and then save.

4. Data Entry and Editing in Datasheet View

Once the tables and fields are properly defined, you need to enter values into the fields. Each entry is called a **record**.

4.1. Displaying a Datasheet View

From the **Table** tab, click the **View** button, then select **Datasheet View**. Alternatively, click **Datasheet View** in the status bar.

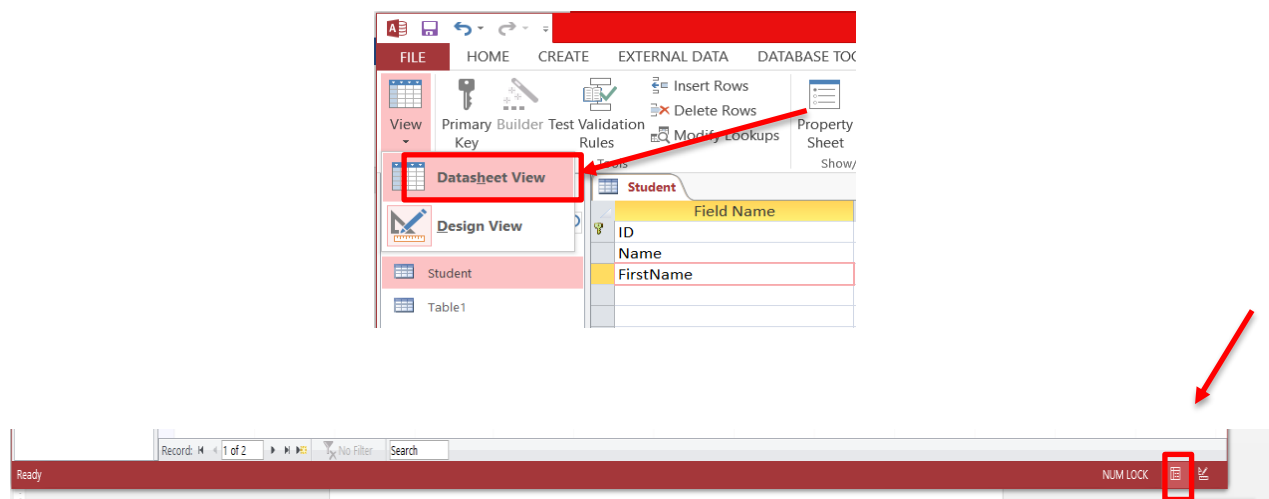


Figure 3.12: Displaying the Datasheet View.

4.2.Entering Field Values in a Datasheet View

Datasheets must be filled in row by row, not by columns.

- Enter each record in the datasheet.
- Once a row is completed, move to the next record.

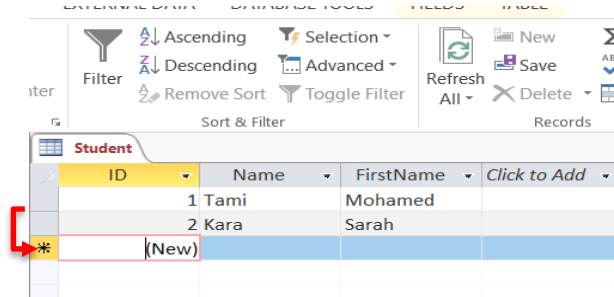


Figure 3.13 : Entering Field Values. .

4.3.Deleting a Record

From the datasheet view:

- Right-click on the record header you want to delete.
- Click on Delete Record.
- Confirm the deletion.

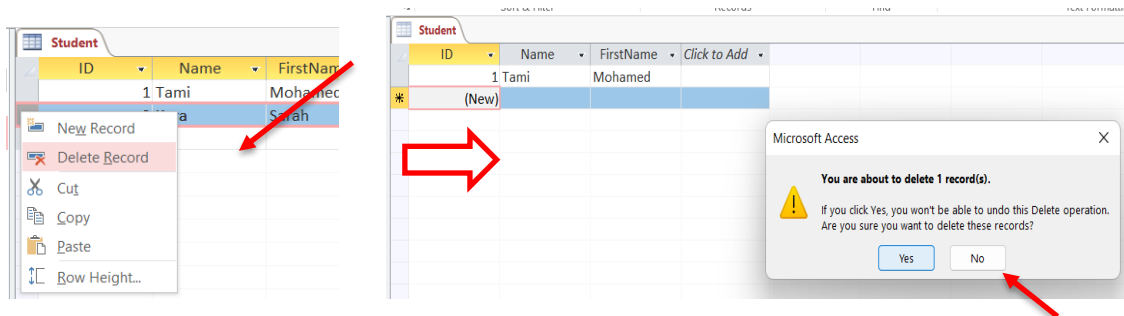


Figure 3.14 : Deleting a Record.

4.4.Modifying a Record

From the datasheet view: Modify the values and then save.

4.5.Printing a Datasheet

To print a datasheet :

- Click on *File / Print / Print Preview* to modify the page layout if needed.
- Click Print to start the printing process

5. Relationships

5.1. Concept of a Relationship

A relationship is a link between two tables, associating data from the first table with data from the second.

The practical aspect of linking tables on common fields is to avoid redundancy. There's no need to repeatedly write the same information. For example, with a relational database like Access, you don't need to write the same client information on every invoice. The strength of linked tables is that they give you access to information from another table that you wouldn't have access to without a relationship between the tables.

For instance, in the **Invoices** and **Customers** tables, the **Invoices** table contains the **Customer Code** field, but no other information about the customer. This field is the key to linking the two tables. It can be linked to the **Customer ID** field in the **Customers** table. Once linked, you can retrieve additional information about the customer, such as their name, postal code, and city.

5.2. Creating a Relationship

- Click on the **Database Tools** tab, then click on **Relationships**.

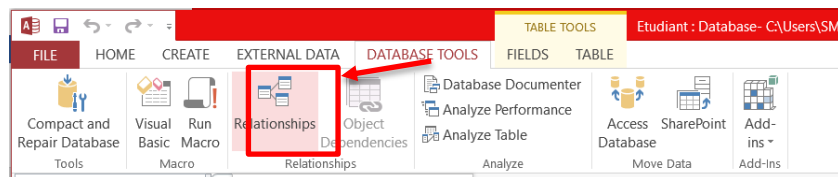


Figure 3.15: The Relationship Tab.

The **Show Table** dialog box displays all the tables in the database.

- Select multiple tables, then click **Add**. Once you have finished adding tables to the **Relationships** tab, click **Close**.

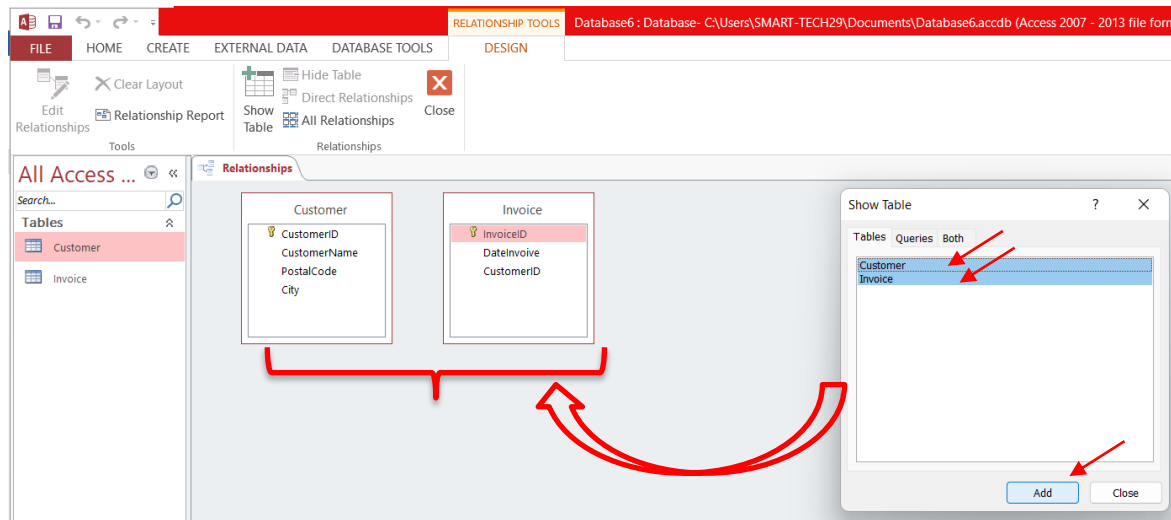


Figure 3.16: The Show Table Dialog Box.

- Drag a field (usually a primary key) from one table to the common field (foreign key) in the other table. To drag multiple fields, hold down **Ctrl**, click on each field, and then drag them. The **Edit Relationships** dialog box will appear.

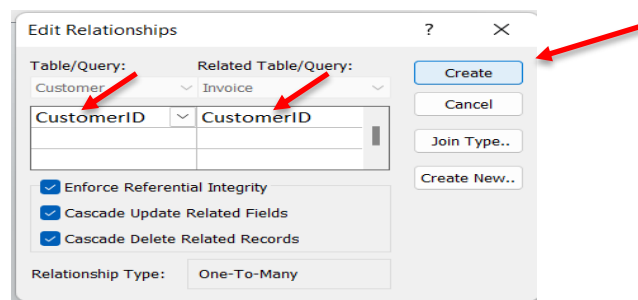


Figure 3.17: The Edit Relationships Dialog Box.

- Verify that the field names displayed are the common fields for the relationship. If any field name is incorrect, click on it and select the correct field from the list.
- Click **Create**.

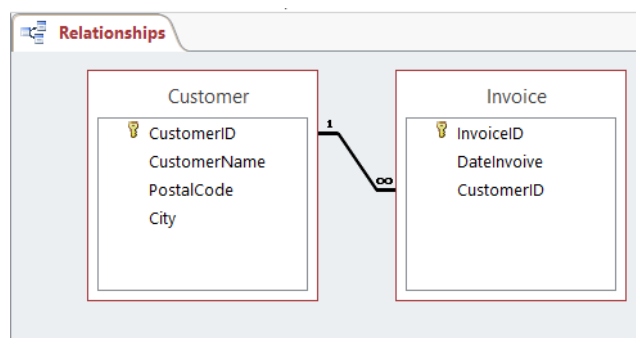


Figure 3.18: Example of a Relationship.

5.3. Viewing a Relationship

Go to the **Relationships** view. Click on **Show All Relationships**.

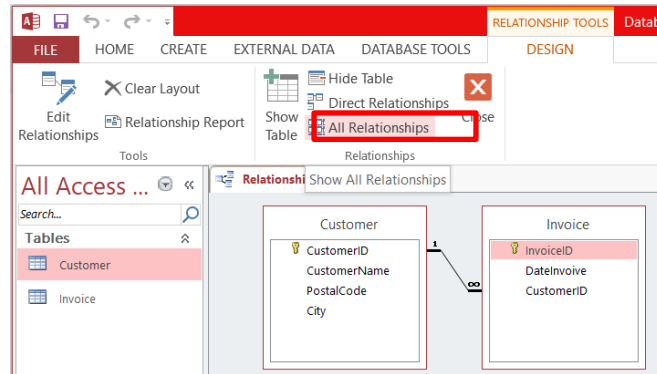


Figure 3.19: Viewing Relationships.

5.4. Removing a Table from the Relationship View

You may see some tables that you don't recognize: these are tables required for the internal functioning of Access, but they don't need to be visible to you. You won't delete them, but simply hide them from the Relationships window. To hide a table, click on each one and press the **Delete** key, or right-click and select **Hide Table**.

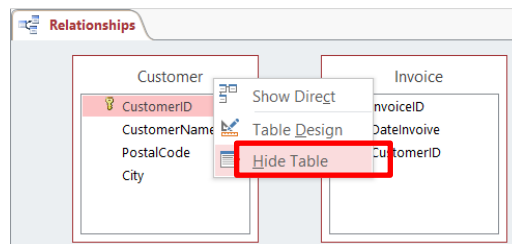


Figure 3.20: Removing a Table from a Relationship.

5.5. Deleting a Relationship

Now, to delete the relationship between **Customer** and **Invoice**:

- Right-click on the line representing the relationship.
- Select **Delete**.

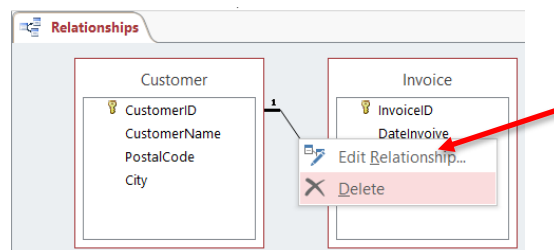


Figure 3.21: Deleting a Relationship.

A confirmation message will appear:

- Confirm that you want to delete the relationship by clicking **Yes**.

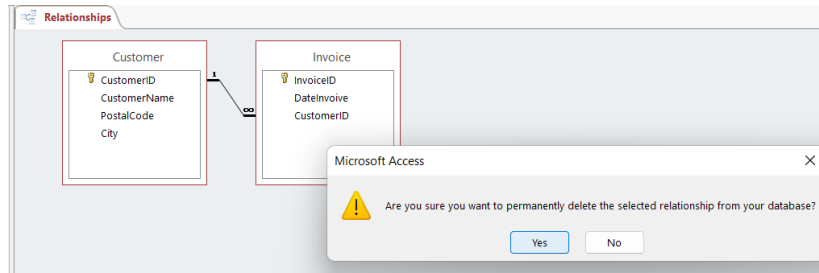


Figure 3.22: Confirmation Message for Deleting a Relationship.

2. Definition of a Query

Queries are used to display only specific data contained in tables based on certain criteria. They can also perform calculations on the data, as well as create data sources for forms, reports, or even other queries (you can create a query based on the result of another query). Additionally, queries are used to modify existing tables or create new ones.

3. Creating a Query

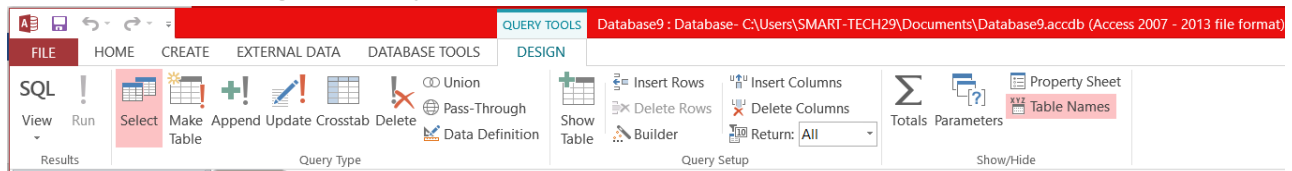


Figure 4.1: Query Tools Tab

a. Creating a Query Using the Query Wizard

- Click on the **Create** tab.
- Press the **Query Wizard** button.

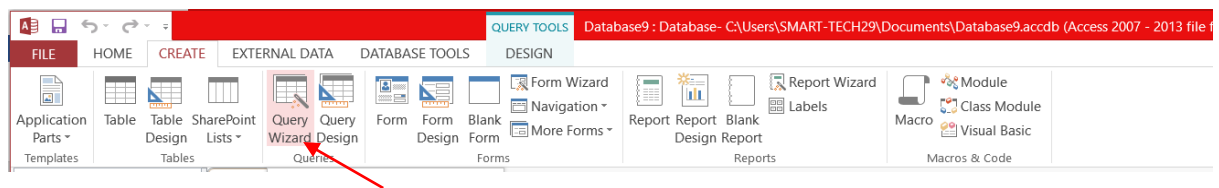


Figure 4.2 : Query Wizard

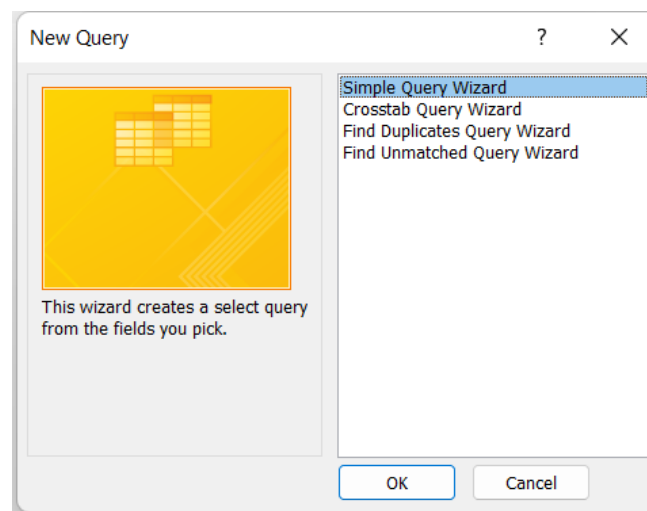


Figure 4.3: New Query Window

The Query Wizard can assist you in creating certain types of queries. However, for most queries, it is preferable to go directly into Design View to create a query that meets your needs. Below is a description of the types of queries that the wizard can help you create:

- **Simple Query Wizard:** This wizard asks you to choose the fields you want to display and then shows the results. There are no selection criteria. It displays the requested fields for all records.
- **Crosstab Query Wizard:** This wizard asks you a few questions to create a crosstab query. You can easily calculate the count, sum, average, and other operations for one field relative to another. For example, you can find out how many employees work in the company by location and job title, or what types of products different customers buy from the company. This can become a powerful analytical tool.
- **Find Duplicates Query Wizard:** This is very useful when you're having trouble creating relationships between tables. One reason Access may refuse to create a relationship is due to illegal duplicates in one of the tables. This query will help you find which records contain duplicate field values, allowing you to correct the issue.
- **Find Unmatched Query Wizard:** This is also useful for table relationships. What is the point of a relationship if there is no corresponding data in the other table? This query helps you find records that have no equivalent or shared value in the related table. This ensures that all records are linked to relevant information in another table.

b. Creating a Query in Design View

The steps to create an Access query are as follows:

1. **Choose the table(s) and query(ies) needed**
2. **Choose the type of query**
3. **Select the fields required**
4. **Determine if the fields need to be sorted**
5. **Hide fields if necessary**
6. **Specify the selection criteria**
7. **Run the query**
8. **Special options:** Grouping, Expression Builder
9. **Link tables**

Example: Using the TP3 database, create a query that displays the first name and last name of all employees whose first name is Mohamed.

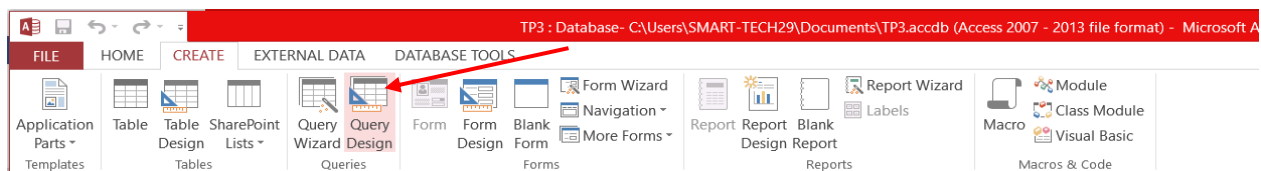


Figure 4.4: Create Tab

Step 1: Choose the Necessary Tables and Queries

You can create a query from a table or even from another query. It is also possible to have a query composed of fields from multiple tables or queries. Access will prompt you with a list of available tables and queries and ask you to select which one(s) you want to add to your new query.

To proceed:

- Select the tables and queries you need, one at a time.
- Press the **Add** button to include each selected table or query.
- Once you have selected the required table(s), query(ies), or a combination of both, press the **Close** button.

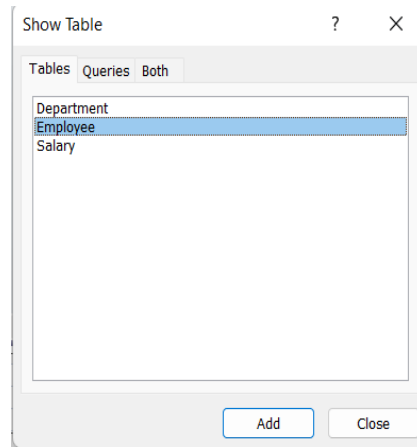


Figure 4.5: Display Table Dialog Box

- Select the *Employee* table.
- Press the Add button.
- Since there are no other tables or queries to choose from, press the Close button.

It is possible to create a query from multiple tables, queries, or a combination of both. If, for any reason, you forgot to include a table or query, you can add it in Design View by clicking the **Show Table** button. Then, repeat the process described above.

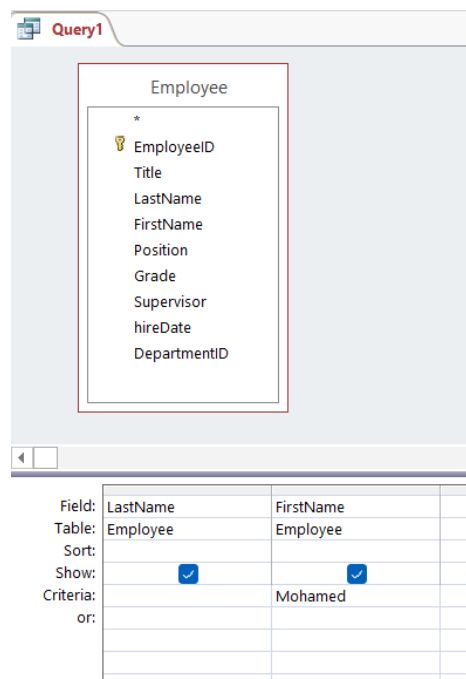


Figure 4.6: Query Design View

You are now in the query creation screen. The upper part of the screen contains the tables and queries you selected earlier. The lower part is the most important, as it is where you define the fields you need and the options you want to apply.

- **The first row** is where you select the fields you need for your query.
- **The second row** shows the name of the table from which each field comes.
- **The third row** is for specifying the order in which the data should be displayed.
- **The fourth row** determines whether the field should be displayed or hidden. In some cases, it may be preferable not to display a field.
- **The following rows** allow you to define the criteria for displaying a record.

Step 2: Choose the Type of Query

Access offers six types of queries, each specialized for a specific type of search or operation.

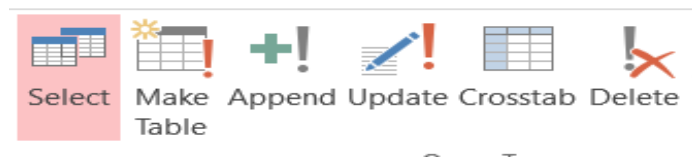


Figure 4.7: Types of Queries

Access provides several types of queries, each designed for specific tasks or operations. The six main types of queries are:

1. **Select Query:** This is the most common type of query. It allows you to retrieve data from one or more tables or queries based on specified criteria. It is used to display records.
2. **Action Queries:** These queries perform operations on data, such as adding, updating, or deleting records. There are four main types of action queries:
 - **Append Query:** Adds new records to an existing table.
 - **Update Query:** Modifies existing records in a table based on specific criteria.
 - **Delete Query:** Deletes records from a table based on certain conditions.
 - **Make Table Query:** Creates a new table from the results of the query.
3. **Crosstab Query:** This type of query summarizes data in a compact, grid-like format, allowing you to perform calculations like sums, averages, or counts across different categories (for example, summarizing sales data by product and region).
4. **Parameter Query:** This type of query prompts the user to enter a value (such as a date or a keyword) at runtime, which is then used as a criterion for filtering the data.
5. **Find Duplicates Query:** This query identifies records with duplicate values in one or more fields, helping you maintain data integrity.
6. **Find Unmatched Query:** This query identifies records in one table that do not have a corresponding record in another related table.

Each query type serves a different purpose, depending on whether you're simply retrieving data, performing calculations, or modifying the structure or content of your database.

Step 3: Choose the Necessary Fields

From the tables and queries you have selected, you need to choose the fields that are required for your new query. This means you don't need to include all fields—only those necessary for your query. Additionally, the asterisk (" * ") at the beginning of each table in a query indicates that you can add all fields from that table to the query, in the same order as they appear in the table. This also applies to other queries that you may add to your query.

There are three ways to choose a field:

- **Select from the list:** Choose the field you need from the available list.
- **Double-click the field:** Double-click the field to add it to the query design grid.
- **Drag and drop the field:** Click and drag the field to the appropriate place in the query design grid.

Step 4: Determine if the Fields Need to Be Sorted

Once you have chosen the necessary fields for your query, you need to decide whether these fields should be displayed in a specific order. Below each field name, there is an option to activate sorting for that field. You can choose not to apply sorting or to sort in ascending or descending order.

To set the sort order:

1. Click in the **Sort** row under the field you want to sort.
2. Click the button with the downward-pointing triangle.
3. From the list, select the sort type you want (Ascending or Descending).

This allows you to organize the results of your query based on specific fields, either in ascending or descending order, depending on your needs.

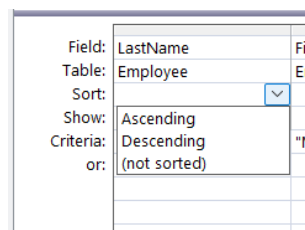


Figure 4.8: Field Sorting Window

Step 5: Hide Fields If Necessary

By default, all the fields you select will be displayed in the query results. However, there are times when you may need a field for searching or filtering purposes but don't want it to appear in the final display. For example, you might want to display the first and last names of people earning more than 50,000 DA, but you don't want to show their income.

To hide a field from the display:

- Uncheck the box under the **Show** row for the field you wish to hide.

- A checkmark in the box indicates that the field will be shown in the query results. By unchecking the box, you ensure that the field is included in the query's search or filtering process but does not appear in the output.

Step 6: Determine the Selection Criteria

Once you have chosen and arranged the necessary fields, sorted them, and decided which ones to display or hide, you need to set the **selection criteria**. If you don't specify any criteria, all records will be displayed in the results.

The criteria are used to filter the records and display only those that meet your specific needs. Access provides several operators that you can use in different situations. Here's a list of common operators and their descriptions:

- **Equals (=):** Filters records that match the specified value exactly.
Example: = "Mohamed" will find records where the field equals "Mohamed".
- **Not Equal to (<>):** Filters records that do not match the specified value.
Example: <> "Mohamed" will find records where the field does not equal "Mohamed".
- **Greater than (>):** Filters records where the value is greater than the specified value.
Example: > 50000 will find records where the value is greater than 50,000.
- **Less than (<):** Filters records where the value is less than the specified value.
Example: < 50000 will find records where the value is less than 50,000.
- **Greater than or equal to (>=):** Filters records where the value is greater than or equal to the specified value.
Example: >= 50000 will find records where the value is 50,000 or more.
- **Less than or equal to (<=):** Filters records where the value is less than or equal to the specified value.
Example: <= 50000 will find records where the value is 50,000 or less.
- **Between ... And:** Filters records that fall within a specified range.
Example: BETWEEN 30000 AND 50000 will find records where the value is between 30,000 and 50,000.
- **Like:** Filters records based on pattern matching using wildcards.
Example: LIKE "M*" will find records where the value starts with "M" (e.g., "Mohamed", "Maya").
- **Is Null:** Filters records where the field has no value (i.e., is blank).
Example: IS NULL will find records where the field has no value.
- **Is Not Null:** Filters records where the field has a value (i.e., is not blank).
Example: IS NOT NULL will find records where the field is not blank.

These operators help you define more specific criteria for your query, ensuring you only retrieve the records that meet your exact needs.

Step 7: Run the Query

To execute the query and view the results:

- **Click the Run button** (the red exclamation mark) on the toolbar to execute the query immediately.

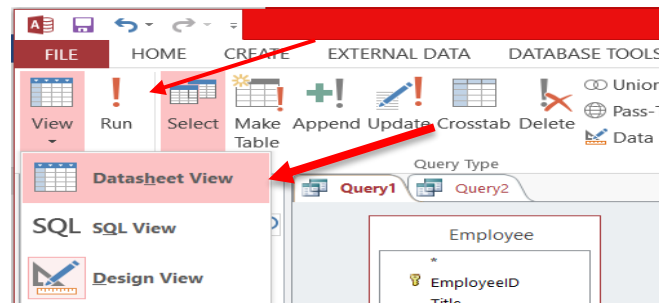


Figure 4.9: Running a Query

The records that meet the specified criteria will be displayed in the results.

To return to **Design View** after viewing the results:

1. Click on the **View** button.
2. Select the **Design View** option from the drop-down menu.

This allows you to make any changes or adjustments to the query if needed, such as modifying criteria or adding/removing fields.

Query1	
LastName	FirstName
AYYADI	mohamed
*	

Figure 4.10: Query Results

Step 8: Choose Specialized Options

In addition to performing searches with criteria or using built-in Access functions, you can also **group fields** to perform operations like summing or averaging numerical fields. Other operations, such as counting the number of records that meet certain criteria, are also possible. To access these operations:

- Click the **Totals** button on the toolbar (it looks like the Greek letter sigma: Σ).

Once the **Totals** option is enabled, you'll see a new row labeled **Total** in the query design grid, where you can select various operations for each field (such as **Sum**, **Average**, **Count**, etc.). This allows you to perform more advanced calculations and aggregations within your query.

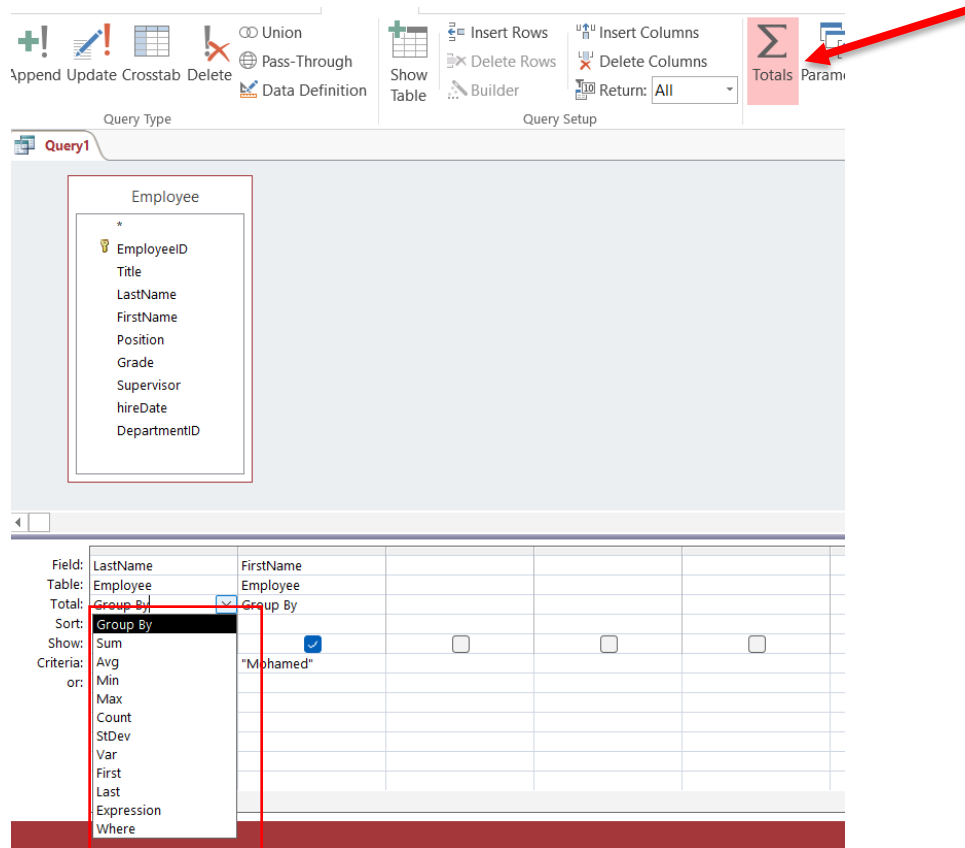


Figure 4.11: Operations in a Query

Here is a list of the operations you can use in a query, along with what each one accomplishes:

1. **Sum**
 - **Purpose:** Adds up the values in a numeric field.
 - **Example:** Sum of all sales amounts.
2. **Average**
 - **Purpose:** Calculates the average value of a numeric field.
 - **Example:** Average salary of employees.
3. **Count**
 - **Purpose:** Counts the number of records or non-null values in a field.
 - **Example:** Count the number of employees in each department.
4. **Min**
 - **Purpose:** Finds the smallest (minimum) value in a numeric field.
 - **Example:** Minimum price of a product.
5. **Max**
 - **Purpose:** Finds the largest (maximum) value in a numeric field.
 - **Example:** Maximum salary in a department.
6. **First**
 - **Purpose:** Returns the first value in a field (based on the sort order).
 - **Example:** First entry in a sorted list of employee names.
7. **Last**
 - **Purpose:** Returns the last value in a field (based on the sort order).

- **Example:** Last entry in a sorted list of employee names.
- 8. **Group By**
 - **Purpose:** Groups records by the values in a field (often used in conjunction with aggregation operations like Sum or Count).
 - **Example:** Group sales data by region or by department.
- 9. **Where**
 - **Purpose:** Filters records based on specific criteria (used in conjunction with other operations).
 - **Example:** Count the number of orders over a certain amount.
- 10. **StdDev** (Standard Deviation)
 - **Purpose:** Measures the variation or dispersion of values in a field.
 - **Example:** Standard deviation of employee salaries to see how varied they are.
- 11. **Var** (Variance)
 - **Purpose:** Calculates the variance of values in a numeric field, showing how spread out the values are.
 - **Example:** Variance in monthly sales figures.

By using these operations, you can perform advanced calculations, summary statistics, and data aggregations directly within your query results.

Step 9: Link Tables

If your query requires fields from multiple tables or queries, you need to **link them** using common fields. It is also possible that you may need additional tables to create "indirect" relationships between two tables.

If a relationship between two tables does not automatically appear, you have two options:

1. **Create a Permanent Relationship**
 - From the **Database Tools** tab on the Ribbon, select the **Relationships** option.
 - This allows you to create a permanent relationship with referential integrity between the tables. Referential integrity ensures that the data in your database remains consistent (e.g., no orphan records).
2. **Manually Create a Link**
 - You can manually create a link between tables within the query design window by dragging the common field from one table to the matching field in another table.
 - This method does not enforce referential integrity and applies only to that specific query. The link will not be permanent, and it will not affect the overall database structure.

By linking the tables properly, you ensure that the data returned in the query is correctly related, which allows you to retrieve comprehensive and meaningful results.-

1. Definition of a Form

Forms are used to present information from different tables in a more organized way. It is also more convenient to enter information into a form than directly into a table. Additionally, it is possible to display only the fields that are necessary on the screen. It is also possible to "lock" a field. This allows the user of the form to view the contents of a field without being able to modify it. As with a query, it is possible to have calculated fields to perform calculations such as subtotals or taxes. The list of Access functions allows you to perform much more than simple calculations.

Generally, a form requires a table or a query that will serve as its "source." A query is more practical when you need a form with fields from multiple tables. It is also more useful if you want to filter the records to display only those that meet your criteria. You can also create a form with no fields. This form could contain buttons that would allow you to open other forms, run queries or macros, or print reports.



Figure 5.1 : Form Tools Tab

2. Methods of Creating a Form

There are several ways to create a form. The simplest method is to click on the "Create" tab and select the "Form" command. This automatically creates a form with all the fields from the selected table (or query). The form can be quickly formatted using the themes that appear in the "AutoFormat" group.

It is also possible to use the form wizard: click on "Create" / "Form Wizard," which leads to a well-presented form in just a few steps.

Finally, you can start with a blank form by clicking on "Create" / "Form Design." The "Controls" group will immediately appear, providing the necessary tools. However, this method is quite slow.

Access offers a large list of predefined forms. Here is a brief description of each.

Instant Form	Automatically generates a form with all the fields from the selected table or query.
Form Design	Starts in design mode with a blank form.
Blank Form	Starts in design mode with a blank form.

Form Wizard	The wizard helps you generate a form that meets your needs by asking a few questions.
Navigation	Allows you to create a form with navigation buttons. You can add other elements afterward.

As its name suggests, the More Forms button offers even more options for presenting data.

Multiple Elements	What was previously called a "Continuous Form." It allows you to view the content of records one after another on the same screen.
Datasheet View	Offers the same layout as a datasheet. However, you can select the fields and choose the display format.
Split Form	Displays the content of a record in the upper part and the complete list of a table or query in the lower part.
Modal Dialog Box	Allows Access to fetch additional information to complete a query.

3. Creating a Form Using the Wizard

The wizard asks you a few questions to determine the fields you need and the type of layout you want to display. It's a quick way to create a form that contains the fields you need in a professional format.

- From the Ribbon, select the **Create** tab.
- Click on the **Form Wizard** button.
- For this example, select the **Employee** table.

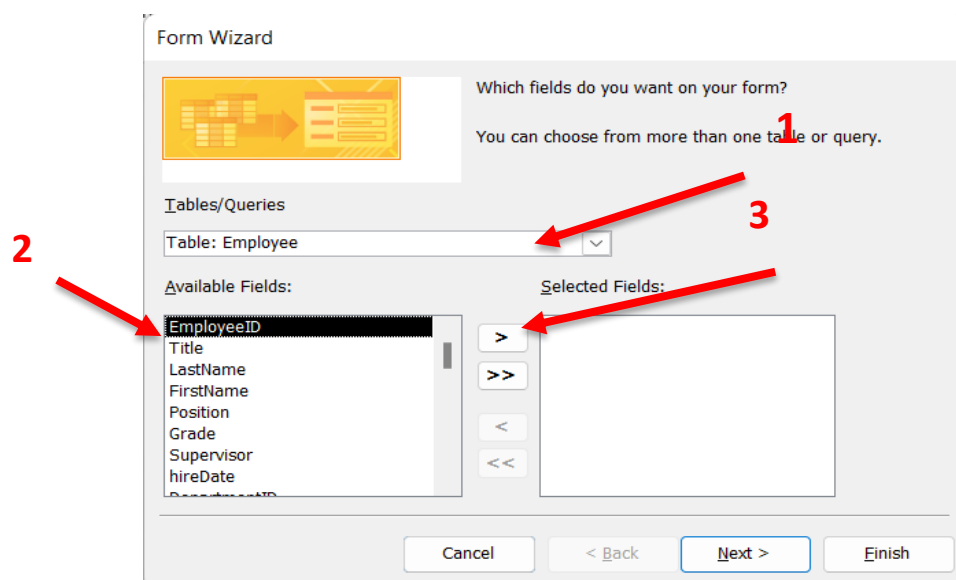


Figure 5.2: Form Wizard (1)

The first step in creating a form is to choose the fields that will be present on the form. To do this, you need to select the source table (or query). Access then displays the available fields. Using the > button, you can transfer available fields to the selected fields list. The selected fields will be displayed on the form. The >> button allows you to transfer all fields at once. You can remove already selected fields using the < and << buttons. Fields that you wish to hide from the user should not be selected.

The wizard allows you to choose fields from multiple tables or queries without having to first create a query with all the desired fields. However, you must ensure that permanent relationships have been established between the tables before selecting fields from different tables or queries. Otherwise, the form will display all possible combinations between the tables, instead of showing related data. You must create a query first if you need to display or access only part of the records, or if there are no relationships between the tables or queries you need.

For our example, we will select all the fields from the **Employee** table and click **Next**. A second dialog box will appear:

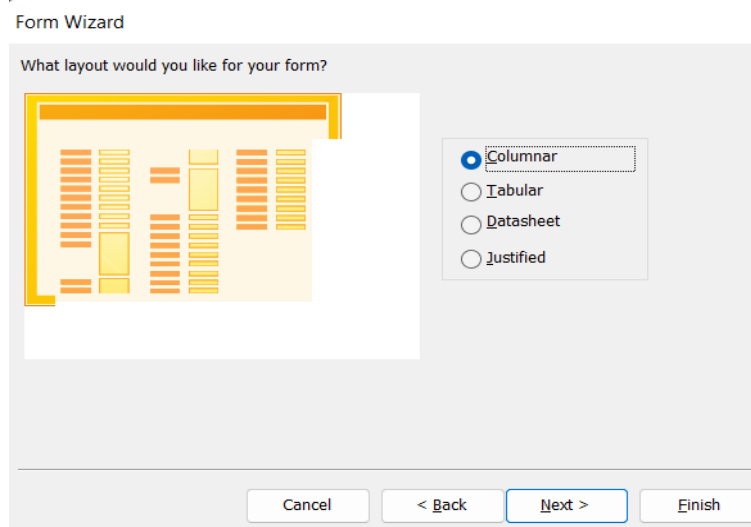
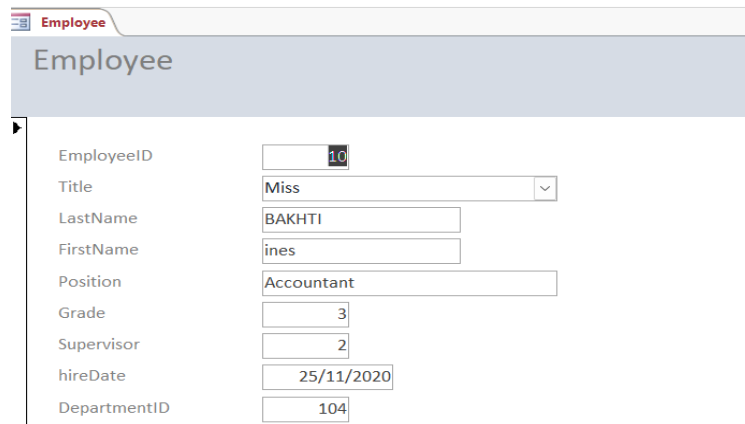


Figure 5.3: Form Wizard (2)

This dialog box asks how the fields will be displayed on the screen. Access even provides a preview of your choice on the left side of the window before you make your final selection. The **Single Column** option will display the fields one below the other. The **Tabular** option will display the fields side by side. Access will resize the fields to ensure they appear next to each other. The **Datasheet** option will display the fields in the same way as when entering data into a table. This is not practical if the user is unfamiliar with all the fields. The **Justified** option will display the fields side by side, but it will not resize the fields. It will display the other fields below. The left side of the window shows an image of the selected layout type.

Here are the four most commonly used layouts:

- **Single Column:**



Employee

EmployeeID: 10

Title: Miss

LastName: BAKHTI

FirstName: ines

Position: Accountant

Grade: 3

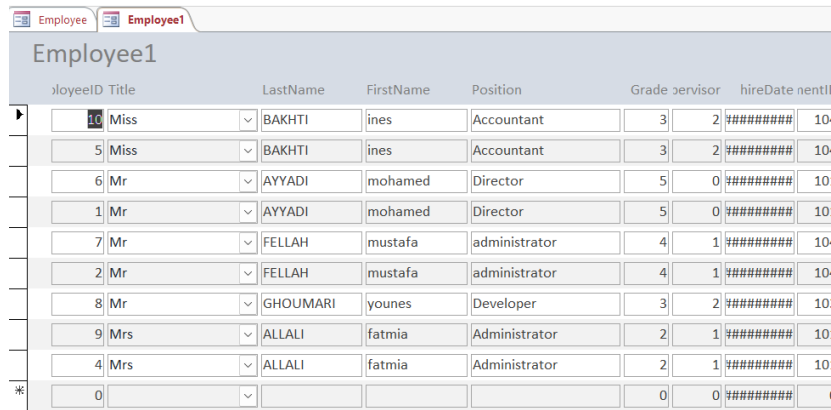
Supervisor: 2

hireDate: 25/11/2020

DepartmentID: 104

Figure 5.4: Single Column Layout of the Form

- **Tabular:**

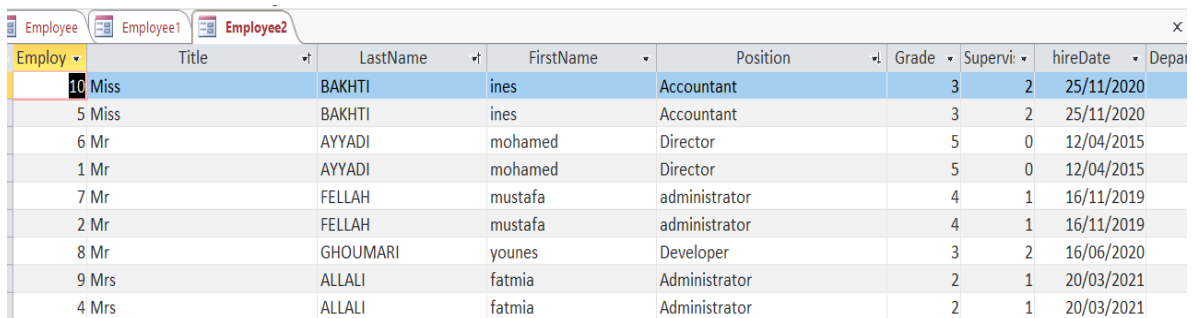


Employee1

EmployeeID	Title	LastName	FirstName	Position	Grade	Supervisor	hireDate	DepartmentID
10	Miss	BAKHTI	ines	Accountant	3	2	25/11/2020	104
5	Miss	BAKHTI	ines	Accountant	3	2	25/11/2020	104
6	Mr	AYYADI	mohamed	Director	5	0	12/04/2015	104
1	Mr	AYYADI	mohamed	Director	5	0	12/04/2015	104
7	Mr	FELLAH	mustafa	administrator	4	1	16/11/2019	104
2	Mr	FELLAH	mustafa	administrator	4	1	16/11/2019	104
8	Mr	GHOUMARI	younes	Developer	3	2	16/06/2020	104
9	Mrs	ALLALI	fatmia	Administrator	2	1	20/03/2021	104
4	Mrs	ALLALI	fatmia	Administrator	2	1	20/03/2021	104
0					0	0		

Figure 5.5: Tabular Layout of the Form

- **Datasheet:**



Employee2

EmployeeID	Title	LastName	FirstName	Position	Grade	Supervisor	hireDate	DepartmentID
10	Miss	BAKHTI	ines	Accountant	3	2	25/11/2020	104
5	Miss	BAKHTI	ines	Accountant	3	2	25/11/2020	104
6	Mr	AYYADI	mohamed	Director	5	0	12/04/2015	104
1	Mr	AYYADI	mohamed	Director	5	0	12/04/2015	104
7	Mr	FELLAH	mustafa	administrator	4	1	16/11/2019	104
2	Mr	FELLAH	mustafa	administrator	4	1	16/11/2019	104
8	Mr	GHOUMARI	younes	Developer	3	2	16/06/2020	104
9	Mrs	ALLALI	fatmia	Administrator	2	1	20/03/2021	104
4	Mrs	ALLALI	fatmia	Administrator	2	1	20/03/2021	104

Figure 5.6: Datasheet Layout of the Form

- **Justified:**

EmployeeID	Title	LastName	FirstName	Position	Grade	Supervisor	hireDate	DepartmentID
10	Miss	BAKHITI	ines	Accountant	3	2	25/11/2020	104

Figure 5.7: Justified Layout of the Form

For our example, select the **Single Column** option.
Click the **Next** button.

Form Wizard

What title do you want for your form?
Employee4

That's all the information the wizard needs to create your form.

Do you want to open the form or modify the form's design?

☒ Open the form to view or enter information.

☐ Modify the form's design.

Cancel < Back Next > Finish

Figure 5.8: Form Wizard - Choosing the Form Title

The final window first asks for the title of the form.

- In addition, Access asks what you want to do next. You can either view the form or modify it immediately.
- Select the **Open the form to view or enter information** option.
- Click the **Finish** button.

Here is a preview of the form once completed.

The screenshot shows the 'Employees List' form in Microsoft Access. At the top, there are several tabs: 'Employee', 'Employee1', 'Employee2', 'Employee3', 'Employee4', and 'Employees List'. The form itself has a title bar 'Employees List'. Below this, there are fields for 'EmployeeID', 'Title', 'LastName', 'FirstName', 'Position', 'Grade', 'Supervisor', 'hireDate', and 'DepartmentID'. Each field has a corresponding input box. The 'EmployeeID' field contains the value '10'. The 'Title' field is a dropdown menu showing 'Miss'. The 'LastName' field contains 'BAKHTI'. The 'FirstName' field contains 'ines'. The 'Position' field contains 'Accountant'. The 'Grade' field contains '3'. The 'Supervisor' field contains '2'. The 'hireDate' field contains '25/11/2020'. The 'DepartmentID' field contains '104'. At the bottom of the form, there is a status bar with the text 'record: 1 of 9'. To the right of this text are several navigation buttons: a left arrow, a right arrow, a double left arrow, a double right arrow, and a search icon. To the right of these buttons is a search box with the text 'Unfiltered' and a 'Search' button. Numbered callouts 1 through 10 point to various parts of the form: 1 points to the 'EmployeeID' field, 2 points to the 'Title' dropdown, 3 points to the 'record: 1 of 9' text, 4 points to the left arrow button, 5 points to the right arrow button, 6 points to the double left arrow button, 7 points to the double right arrow button, 8 points to the search icon, 9 points to the search box, and 10 points to the 'Search' button.

Figure 5.9: Preview of the Employee Form

The Access user will use forms to search for information, modify data, and add records. They will not directly use the tables. Therefore, it is important to know how to use a form. Here are its main components:

1. **Field Name:** Cannot be modified by the user.
2. **Field Value:** This is where the user can view and even modify the information.
3. **Button to go to the first record.**
4. **Button to go to the previous record.**
5. **Current record number.**
6. **Total number of records.**
7. **Button to go to the next record.**
8. **Button to go to the last record.**
9. **Button to create a new record.**
10. **Search box:** Type text here, and each time you press Enter, Access will take you to the next record where a field contains that text. The searched text will also be highlighted.

4. Modifying the Form's Structure

Similar to tables and queries, forms also have several display modes: **Form View**, **Page View**, and **Design View**. To switch between these modes, use the **View** button in the **Home** tab or right-click on the form, as you would with tables and queries. Here is how a form looks in **Design View**:

The screenshot shows a Microsoft Access form in Design View. The form is titled 'Employees List' in the header section. The detail section contains nine fields, each with a text box control: EmployeeID, Title, LastName, FirstName, Position, Grade, Supervisor, hireDate, and DepartmentID. The form is divided into three sections: Form Header, Detail, and Form Footer. The fields are arranged in two columns: the first column contains EmployeeID, Title, LastName, FirstName, Position, Grade, Supervisor, hireDate, and DepartmentID; the second column contains EmployeeID, Title, LastName, FirstName, Position, Grade, Supervisor, hireDate, and DepartmentID. The form is displayed on a grid with a ruler at the top.

Figure 5.10: Form Design View

In **Design View**, several sections of the form can be resized or modified.

5. Adding a Label

A label is used to display text on the form. There are already labels on the form: one in the header section (which contains the title) and nine in the body of the form (to contain the field names). You can add another label, for example, in the footer of the form.

To do this, click on the **Label Control** in the **Form Design Tools / Design** tab. Once the **Label Control** is selected, draw a rectangle in the footer of the form. Then, enter the text "Created by" followed by your name, and adjust the size and style of the text using the buttons in the **Font** group.

Note that for the buttons to be usable, you must click on the border of the label, not while modifying the text it contains. Of course, you can resize and move the label as needed. Here is the result:

Figure 5.11: Adding a Label to a Form

6. Adding a Command Button

You can also add command buttons to the form. Command buttons allow the user to open another form, run queries, etc. Select the **Button Control** from the **Form Design Tools / Design** tab. Then, draw a rectangle in the footer of the form. The **Command Button Wizard** will appear suddenly:

Figure 5.12: Command Button Wizard (1)

The wizard first asks what will happen when the user clicks the button. Several actions are available, and these actions are grouped into categories. For now, let's choose the **Miscellaneous** category and the **Run Query** action. Click **Next**. A second dialog box will appear:

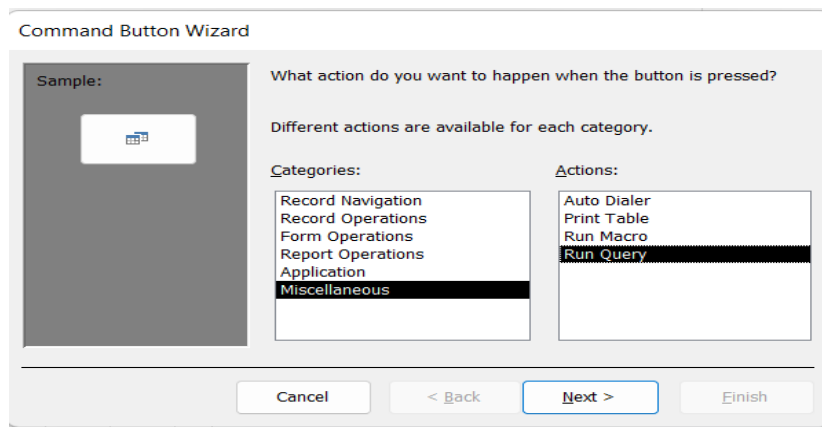


Figure 5.13: Command Button Wizard (2)

The wizard displays a list of existing queries in your database and asks you to select the query that the command button will open. Select a query and click **Next**.

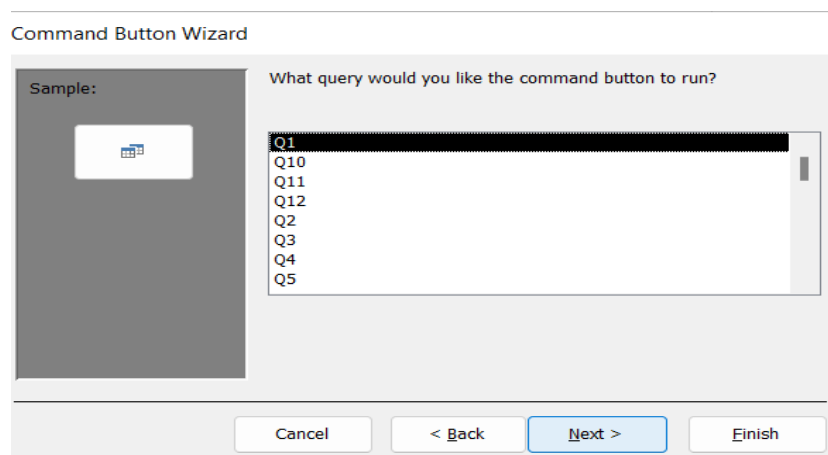


Figure 5.14: Command Button Wizard (3)

Our button should indeed open a query and display the number of employees by position. Click Next.

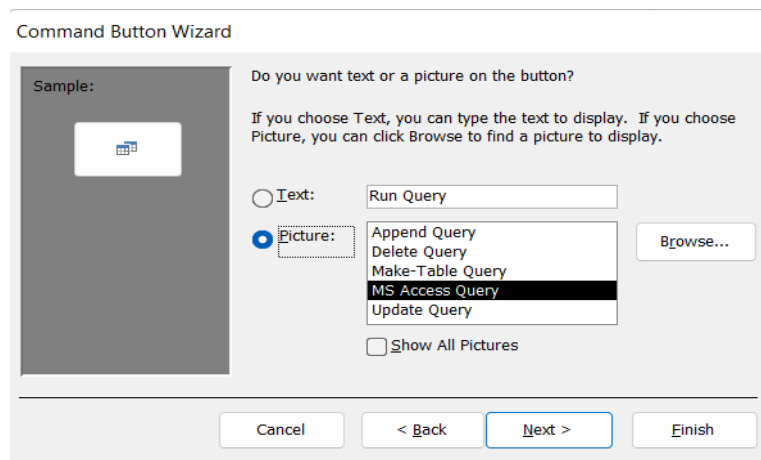


Figure 5.15: Command Button Wizard (4)

This dialog allows us to choose what will be displayed on the button: an image or text. If you choose to display an image, select one of the images provided by Access. If you choose to display text, enter the desired text. It is important that the text is clear and indicates the button's purpose. Once this step is completed, click **Next**. A final dialog will appear:

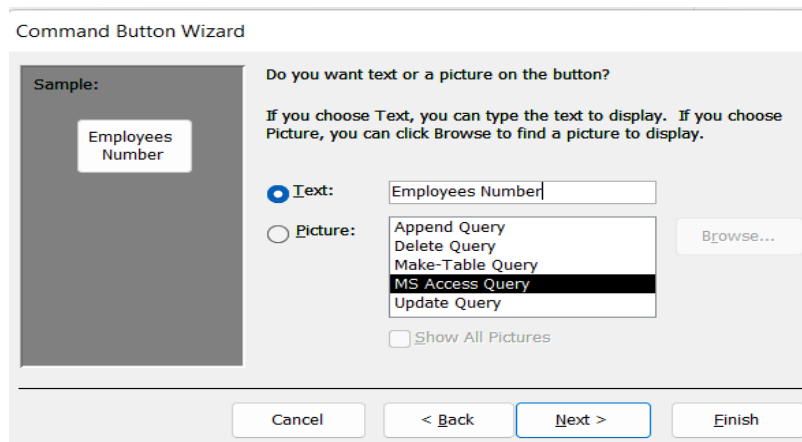


Figure 5.16: Command Button Wizard (5)

The final dialog allows you to modify the name of the button. You must also enter the message that will appear on the button in the caption.

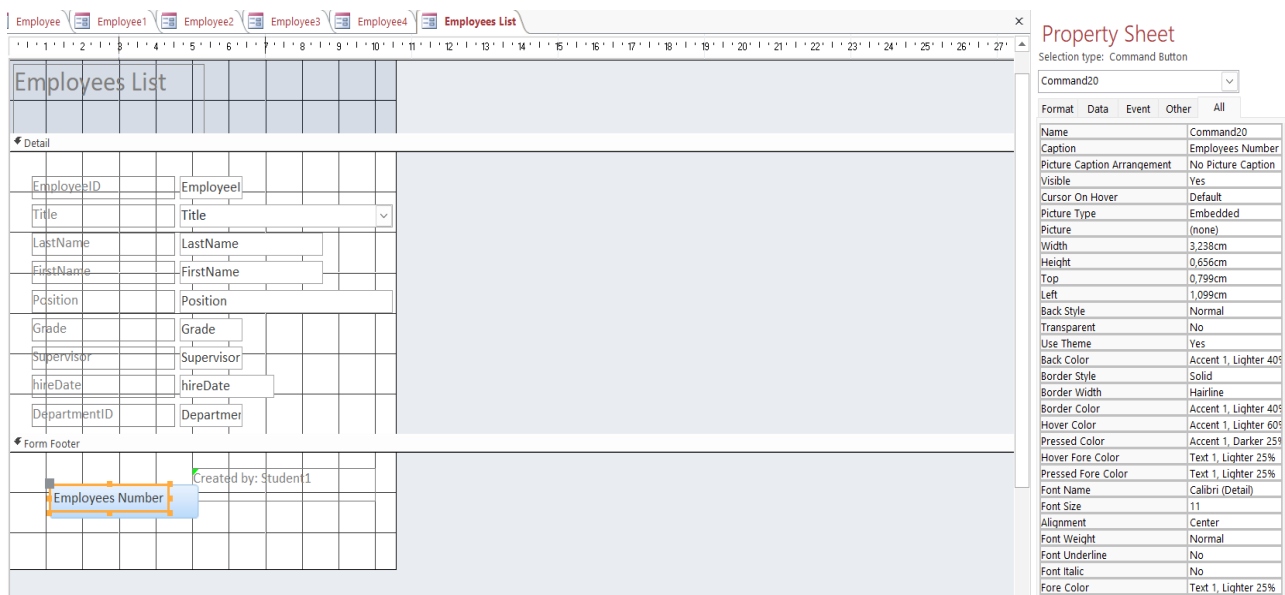


Figure 5.17: Preview of the Button Added to a Form

When using an Access form, you may encounter all kinds of buttons.

7. Subforms

A subform allows you to simultaneously enter information into two tables linked by a one-to-many relationship (up to three). The table on the "one" side of the relationship will be associated with the main form, while the table on the "many" side will be associated with the subform. These two forms will be visible under two different names, so it is advisable to give them the same name with a suffix to differentiate them.

Select the **Subform/Subreport** control from the **Form Design Tools / Design** tab. Then, draw a rectangle in the desired location.

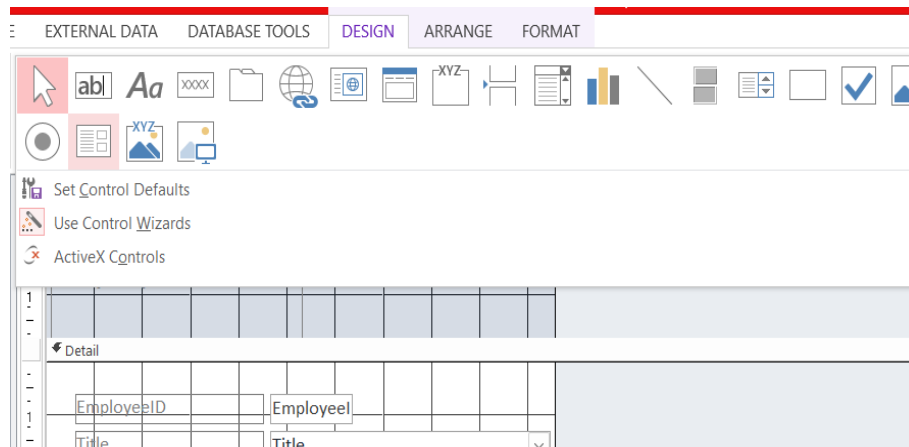


Figure 5.18: Adding a Subform to a Form

The **Subform Wizard** appears:

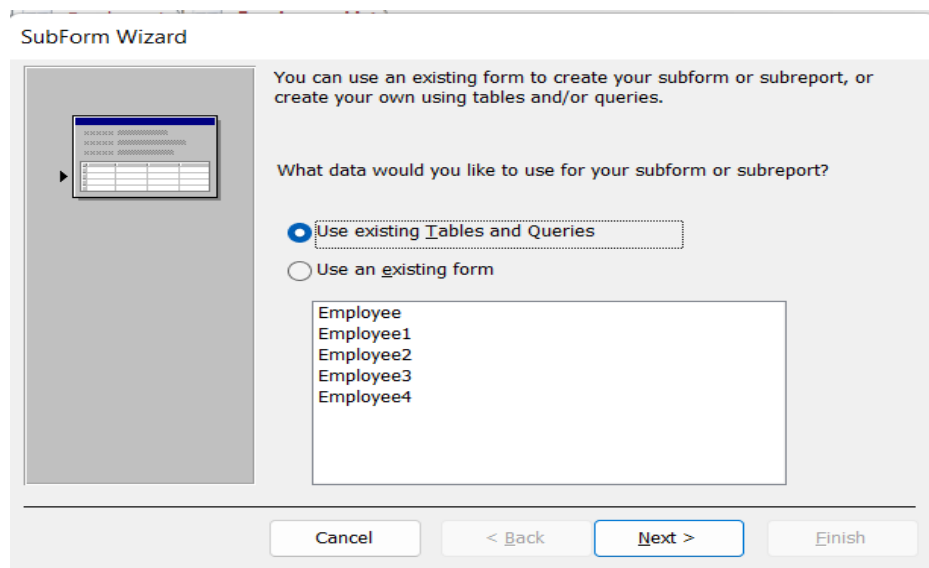


Figure 5.19: Subform Wizard (1)

Continue selecting the fields from the table, then click **Next**.

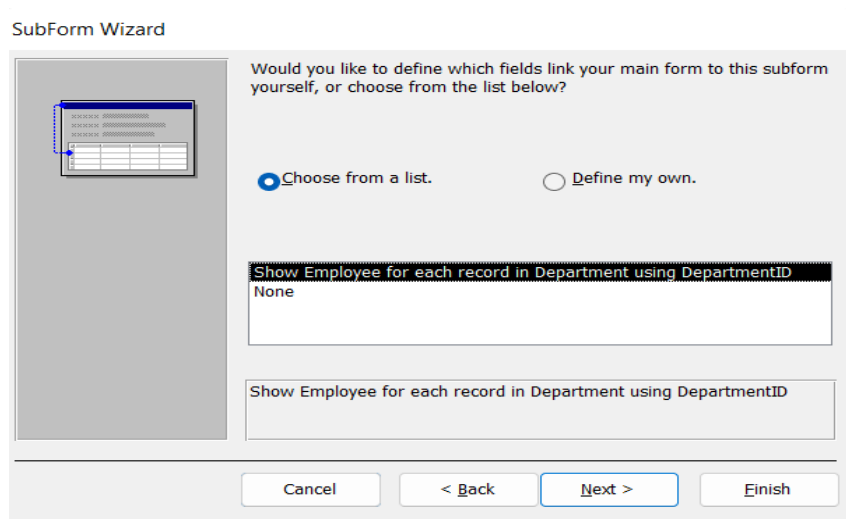


Figure 5.20: Subform Wizard (2)

Click **Next** to confirm and switch to **Form View** to see the result of displaying the list of employees by department:

Department

DepartmentID

DepartmentName

Employee subform1

Employ	Title	LastName	FirstName
6 Mr		AYYADI	mohamed
1 Mr		AYYADI	mohamed
9 Mrs		ALLALI	fatmia
4 Mrs		ALLALI	fatmia

Figure 5.21: Preview of Subform Added to a Form

1. Definition of a report

A report is a representation of the records from a database that is put on paper. It is possible to extract a summary from this data that would otherwise be difficult to obtain. Additionally, it is easier to sort and group information based on relevant fields.

2. Creating a report

Microsoft Access 2013 offers several ways to create a report. Under the "Create" tab, you will find a grouping of options to create a report.

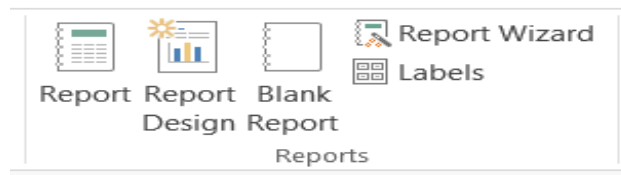


Figure 6.1 : Report Tools Tab

The first option automatically generates a report with all the fields from the table or query you selected previously. The "Create Report" option generates a blank report and places you in design view. You have all the commands at your disposal to create the report of your choice. The "Blank Report" option also generates a blank report but keeps you in the report view mode. The pane with the field list appears on the right side of the screen, and you also have access to all the report creation tools. You can then add fields or make changes to the report and immediately see the result in report preview mode. The Report Wizard helps you create a report by asking you questions before generating the report based on your responses. There is also a wizard to help you create labels that you can use to identify individuals at events or for mail merges.

3. Creating a Report Using the Wizard

The following example demonstrates how to create a report using the wizard, step by step, while also exploring the options at each stage:

- Click the "Create" tab.
- Select the "Report Wizard" option.

The wizard will then prompt you to choose the table or query and the fields you need for your report. Pay attention to the order of the fields you select, as they will appear in this order in your new report. It is also possible to select fields from multiple tables or queries. Simply choose the table or query from the list, then add the selected fields from the available field list to the selected fields list. After selecting the desired fields, click "Next."

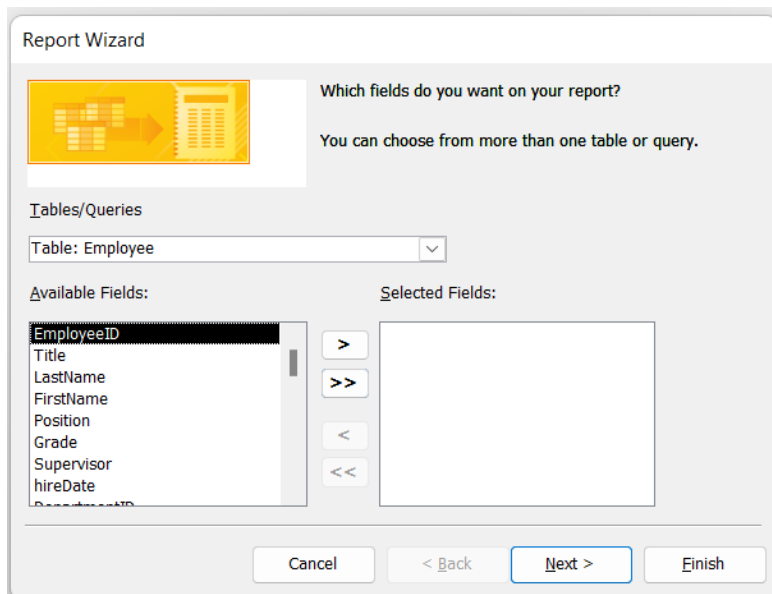


Figure 6.2 : The Report Wizard (1)

a. Grouping Records

The second step of the Wizard allows you to specify a grouping of records based on a field in the report. The grouping suggestion from Access is customizable.

- Select a field from the left list, then click the button showing the ">" symbol to define a grouping on that field. The field will appear in blue on the right side. A grouping can be created on multiple fields, following an order that you can define using the buttons in the "Priority" section.

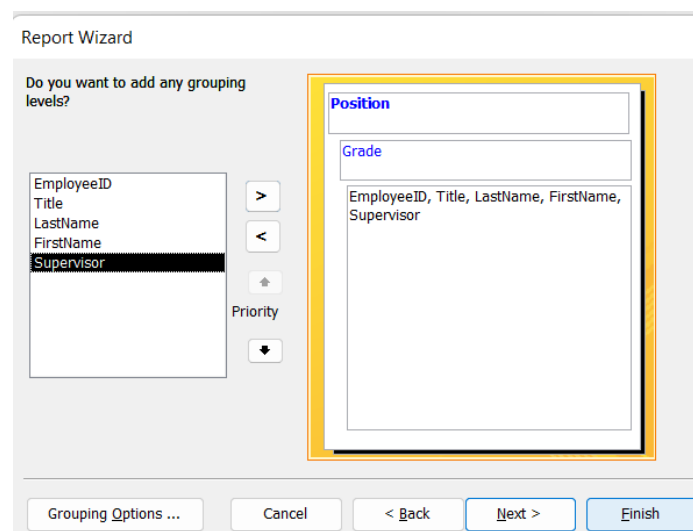


Figure 6.3 : The Report Wizard (2): Defining a Grouping

The wizard asks if you want to group records by a specific field. You can also create "subgroupings," or groupings within groupings. By clicking the "Grouping Options" button, you can apply settings:

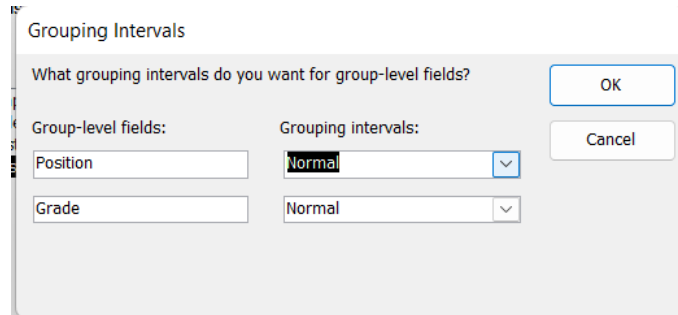


Figure 6.4: The Report Wizard (3): Grouping Intervals

- Leave the option set to "Normal."
- Click the "OK" button.
- Click the "Next" button.

b. Sorting Records

The third step of the Wizard allows you to define the sorting of the report data.

- In the first box, choose the first field to sort by. To switch from ascending to descending order, use the button to the right of the box. This button will toggle between the "Ascending" and "Descending" labels when clicked. The following boxes can be used to define secondary sorting criteria.

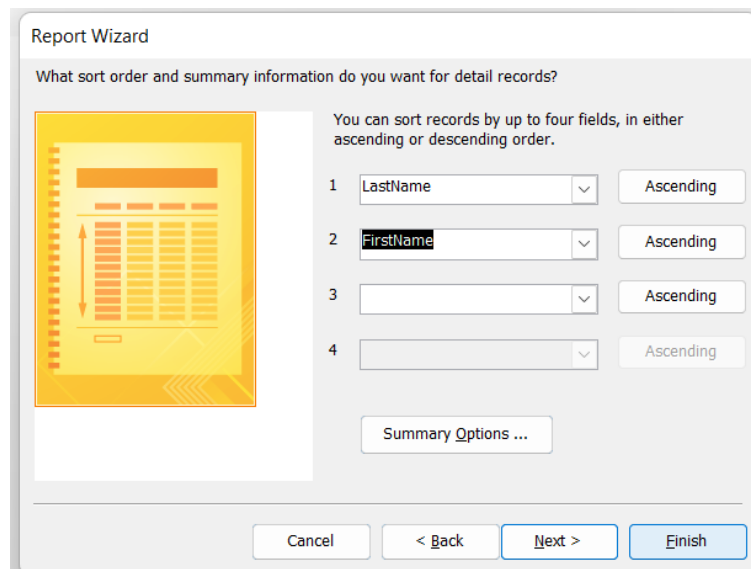


Figure 6.5: The Report Wizard (4): Choosing the Order of Report Values

The wizard asks if you want to sort the records by one or more fields. You can use up to four fields to sort the records. The field in the first box takes priority over the second, and so on. Additionally, you can choose whether to sort the records in ascending or descending order by clicking the button to the right of the field name. Click the "Next" button.

c. Organizing Data in the Report

The fourth step allows you to define the organization of the data in the report.

- Choose one of the layouts offered under the "Layout" section, as well as an orientation under the "Orientation" section.

- Click the "Next" button.

The available layouts may change if a grouping has been set up.

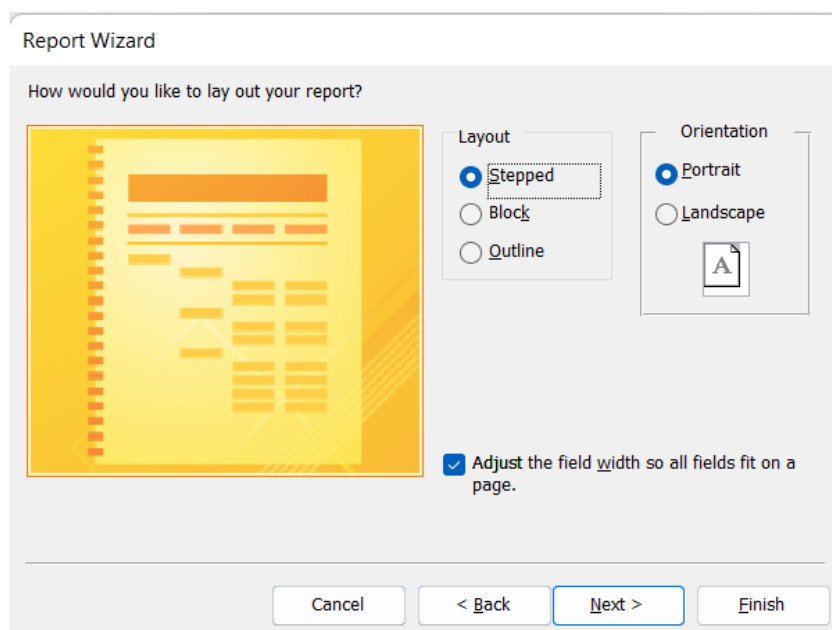


Figure 6.6: The Report Wizard (5): Layouts and Orientations

- Give your report a name.

Report Wizard

What title do you want for your report?

Employee Report

That's all the information the wizard needs to create your report.

Do you want to preview the report or modify the report's design?

☒ Preview the report.

☐ Modify the report's design.

Cancel < Back Next > Finish

Figure 6.7: The Report Wizard (6): Report Title.

Employee Report

Position	Grade	ployeeID	Title	LastName	FirstN
Accountant		3			
		10	Miss	BAKHTI	ines
		5	Miss	BAKHTI	ines
Administrator		2			
		9	Mrs	ALLALI	fatmi
		4	Mrs	ALLALI	fatmi
		4			
		7	Mr	FELLAH	musta
		2	Mr	FELLAH	musta
Developer		3			
		8	Mr	GHOUMARI	youn
Director		5			
		6	Mr	AYYADI	moha
		1	Mr	AYYADI	moha

Figure 6.8: The Report Wizard (7): Viewing the Report in Print Preview Mode

As in Excel, cells with ### mean that the cell is not wide enough to properly display the content. You can change the display options when in design view.

4. Report View Modes

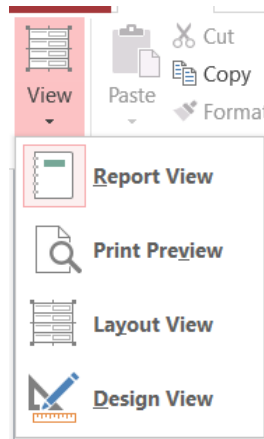


Figure 6.9: Report View Modes.

a. Page View

After creating the report, Page View allows you to set up the report (a practical mode for formatting). In this mode, you can:

- Change the size and color of the text.
- Modify the width of the columns.
- Adjust the height of the rows.
- Modify the field formats.
- Sort the data...

b. Report View

Report View allows you to view the report on the screen, with the option to navigate using the Tab key. This mode is similar to Page View but does not allow any modifications.

You can select data and copy it to the clipboard, then paste it into another application..

c. Print Preview

Print Preview mode allows you to preview the report before printing.

The report is displayed as it will appear when printed. In this mode, you can:

- Modify margins, print orientation (Portrait, Landscape), paper size, the number of columns, etc. ...

d. Design View

Design View allows you to configure each element of the report and provides more precise control over the various report controls.

5. Report Components

A report consists of the following sections:

- **Report Header:** Printed once on the first page of the report.
- **Page Header:** Printed at the top of each page (often contains column titles, e.g., table fields).
- **Group Header:** Printed at the top of each specified group.
- **Detail:** Printed once per record (contains the actual fields).
- **Group Footer:** Printed once at the bottom of each group.
- **Page Footer:** Printed once at the bottom of each page (what will appear at the bottom of each page when printed).
- **Report Footer:** Printed once on the last page of the report (what will appear after the last line on the last page).

1. Definition of a Macro

A macro allows you to execute a series of commands one after another. It automates certain actions by programming command buttons that will be visible in forms or reports.

2. Creating a macro

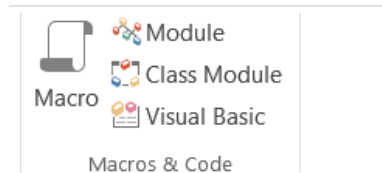


Figure 7.1: Macro Tools Tab

To create a macro:

- Under the **Create** tab, go to the **Macros & Code** group, then click on **Macro**. This procedure creates a standalone macro object that appears under **Macros** in the navigation pane. Standalone macros are useful when you want to reuse the macro in multiple locations within the application. By calling the macro from other macros, you can avoid duplicating the same code in several places. The **Action Catalog** window will appear.

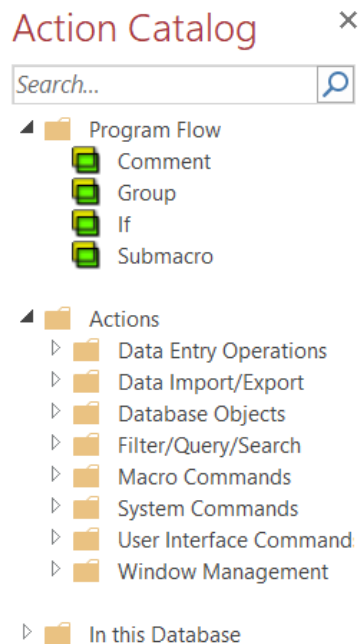


Figure 7.2 : Action Catalog (1)

- Choose the desired action.
- For this example: Continue with the lower section **Add actions to a macro**.

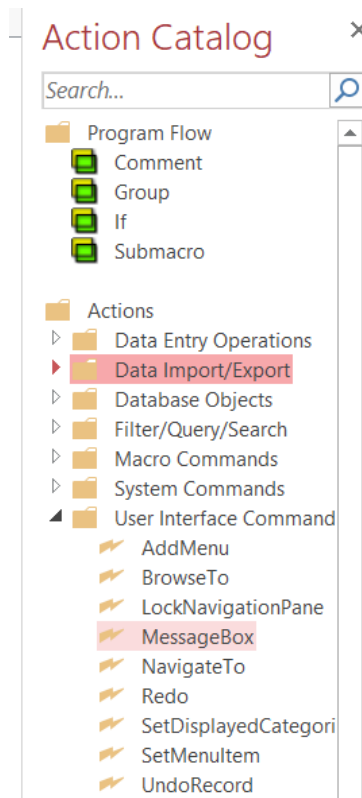


Figure 7.3: Action Catalog (2)

Under the **Actions** column, you will find a list of instructions you can execute. As you can see, Access offers several instructions.

To continue creating the macro, you need to enter the "**Action arguments**" or the characteristics of the action. These arguments change depending on the type of action. For the **MessageBox** action:

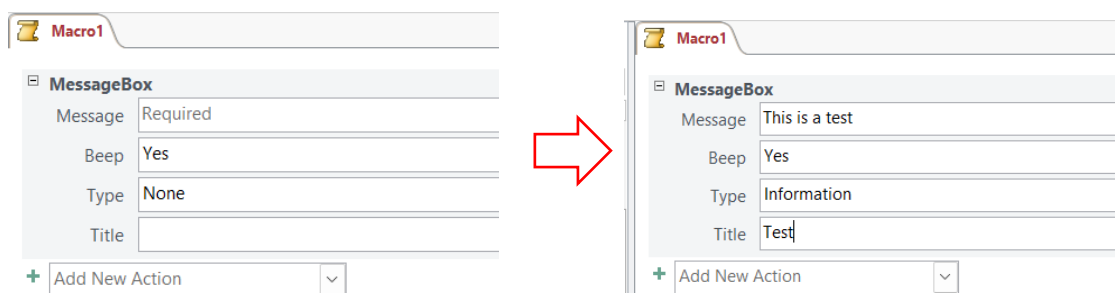


Figure 7.4 : Action Arguments for MessageBox

- Enter the following text in the **Message** field: This is a test.
- Select **Yes** for the **Beep** option.
- For the **Window type**, select **Information**.
- For the **Window title**, write the text **Test**.

If you want to add another action, place the pointer below the last action and select it from the list of actions.

To save the macro:

- In the **Quick Access Toolbar**, click **Save**.
- In the **Save As** dialog box, type a name for the macro (Macro1 in this example), then click **OK**.

Here is the result:

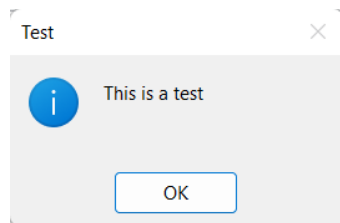


Figure 7.5: The result of Macro1

3. Attaching a Macro to Form Button

The next part involves attaching the newly created macro, Macro1, from the previous section to a form button. There are two ways to attach a macro: attach it to a new button or to an existing button. This section will cover both methods..

a. Attaching to a new Button

- Open an existing form and switch to **Design View**.
- Insert a **Command Button**.

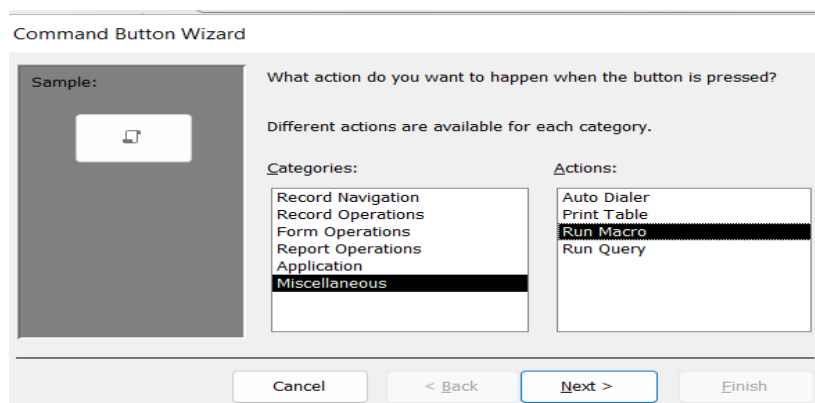


Figure 7.6: Inserting a Command Button

The **Command Button Wizard** offers several options. It includes several button templates to quickly create the most commonly used buttons in different categories.

- From the list of categories, select the **Miscellaneous** category.
- Select the **Run Macro** option from the list of actions. Then click **Next**.
- From the list of available macros, select the **Macro1** macro. Then click **Next**.

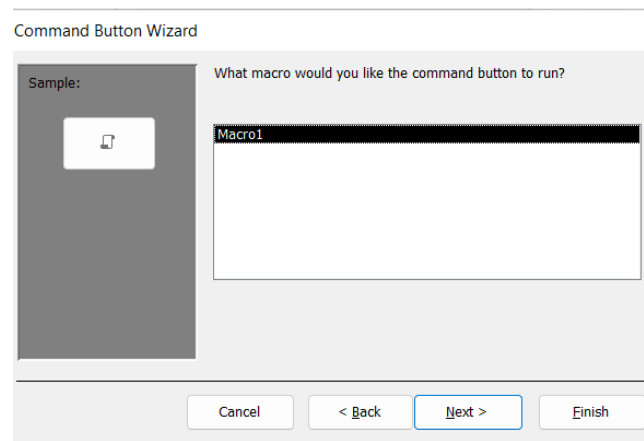


Figure 7.7 : Linking a Macro to a Command Button (1)

Access gives you the option to display either text or an image on the button. For this exercise, the text **Run Macro** will be written on the button.

- Click on the **Text** option.
- Leave the default text in the field to the right of the **Text** option. Then click **Next**.

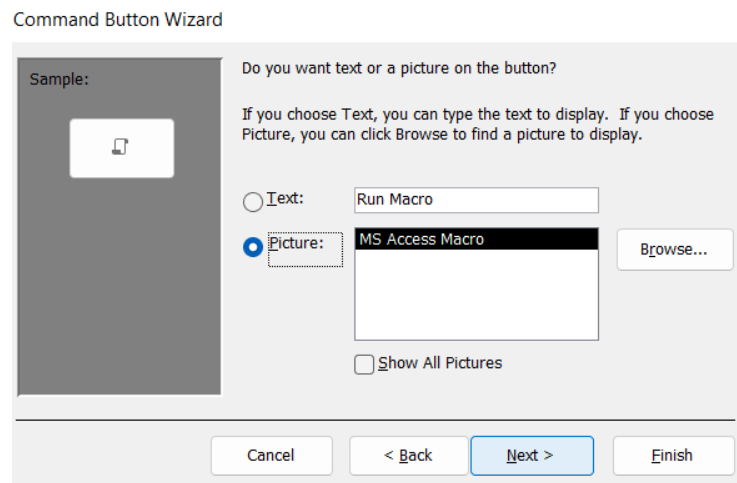


Figure 7.8 : Linking a Macro to a Command Button (2)

b. Attaching to an Existing Button

This part of the lesson shows you how to change the macro assignment to a button and the list of possible events in Access.

- Select **Design View** for the form.
- Hover the pointer over the button to modify.
- Right-click on the button.
- From the list of options, select **Properties**.
- Click on the **Event** tab.

Access will display a list of possible events. You can attach a macro or module to each of these events.

It is possible for Access to execute a macro when you click on a button or field, and execute a different event if you double-click on the same control.

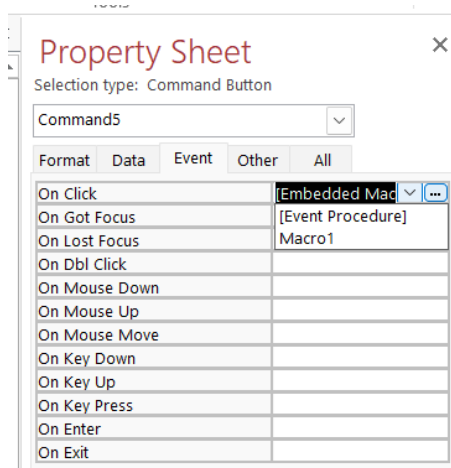


Figure 7.9: Property Sheet/ Event

If the property box contains the words **embedded macro**, it means a macro has already been created for this event. You can modify the macro by following the remaining steps. If the property box contains the words **event procedure**, it means a Visual Basic for Applications (VBA) procedure has already been created for this event. To embed a macro in the event, you must delete the procedure. To do this, you can remove the words **event procedure**, but you must first examine the event procedure to ensure that removing it does not remove any required functionality in the database. In some cases, you can recreate the functionality of the VBA procedure using an embedded macro.

- Click in the box to the right of the **On Click** event.
- Click the button with a downward-pointing triangle at the end of the box.
- From the list of macros and modules, select the **Macro1** macro.

When you click the selected button in the form (in form view), the following window will appear:

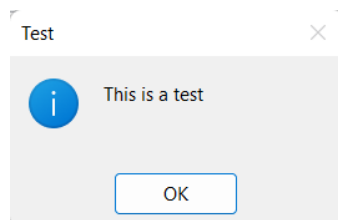


Figure 7.10 : The result of Macro1

4. Running a macro

You can run a macro using one of the following methods:

- Double-click on the macro in the navigation pane.
- Call the macro using the **RunMacro** or **OnError** macro action.
- Enter the macro name in the event property of an object. The macro will run when the event is triggered.

Note: The **AutoExec** macro runs before other macros or VBA code elements, performing an action or a series of actions when a database is first opened.

5. Adding Actions to macro

Actions are the individual commands that make up a macro, and each is named based on its function, such as **FindRecord** or **CloseWindow**.

Step 1: Find a Macro Action

The first step in adding an action is to find it in the **Add New Action** dropdown list or in the **Action Catalog**.

If the Action Catalog is not displayed, go to the **Create** tab, in the **Show/Hide** group, and click on **Action Catalog**.

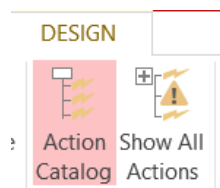


Figure 7.11 : Show/Hide the Action Catalog

To search for an action, use one of the following methods:

- Click the dropdown arrow next to **Add New Action** and scroll down to search for the action. Program flow items appear first, then macro actions are listed alphabetically.
- Search for the action in the **Action Catalog** pane. Actions are grouped by category. Expand each category to display the actions. When you select an action, a brief description of the action appears at the bottom of the Action Catalog.
- Search for the action in the **Action Catalog** pane by typing text in the search box at the top of the pane. As you type, the list of actions is filtered to display all macros containing that text. Access searches through both macro names and their descriptions for the text you enter.

Step 2: Add an Action to a Macro

When you've found the macro action you want, add it to the macro using one of the following methods:

- Select an action from the **Add New Action** list or start typing the action name in the box. Access will add the action at the point where the list **Add New Action** is displayed.
- Drag the action from the **Action Catalog** into the macro pane.
- Double-click on the action in the **Action Catalog**.

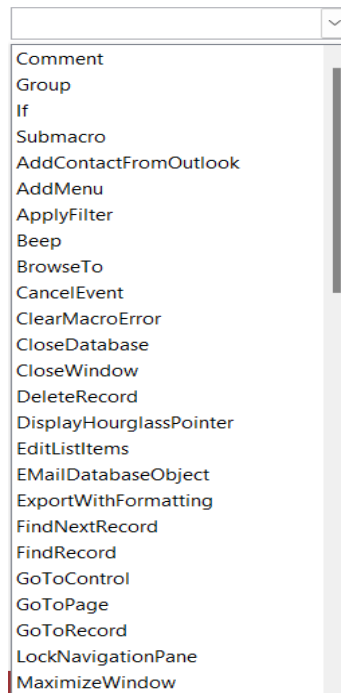


Figure 7.12 : The List of actions

Step 3: Fill in the Arguments

Most macro actions require at least one argument. You can view a description of each argument by selecting the action and hovering the pointer over the arguments. For many arguments, you can select a value from a dropdown list. If the argument requires entering an expression, Access allows you to type the expression and suggests possible values as you type. When you see a value you want to use, add it to your expression by double-clicking on it or using the arrow keys to highlight it, then pressing **TAB** or **Enter**.



Figure 7.13 : Filling in the Arguments of a Macro

6. Operations on Macro Actions

a. Moving an action

Actions are executed in order, from top to bottom of the macro. To move an action up or down in the macro, use one of the following methods:

- Drag the action up or down to the desired position.
- Select the action, then press **CTRL + Up Arrow** or **CTRL + Down Arrow**.
- Select the action, then click the **Up** or **Down** arrow on the right side of the macro pane.

b. Copying and Pasting Macro Actions

If you need to repeat actions that you have already added to a macro, you can copy and paste them in the same way as text paragraphs in a word processor. When you paste actions, they are inserted just below the currently selected action. If a block is selected, the actions are pasted inside the block.

c. Deleting an Action

To delete a macro action: Select the action, then press the **Delete** key. You can also click the **Delete (X)** button on the right side of the macro pane.

d. Grouping Related Actions

You can improve the readability of a macro by grouping actions together and giving the group a meaningful name. For example, you can group the actions that open and filter a form into a group called "**Open and Filter Form**". This makes it easier to see which actions belong together. A group block does not affect how the actions are executed, and the group cannot be called or executed individually. Its primary use is to label a group of actions to help you understand the macro when reading it. Additionally, when editing a large macro, you can collapse each group block into a single line, which reduces the amount of scrolling required. If the actions you want to group are already present in the macro, use the following procedure to add them to a group block:

- Select the actions you want to group.
- Right-click on the selected actions, then click **Create Group Block**.
- In the box at the top of the group block, type a name for the group.

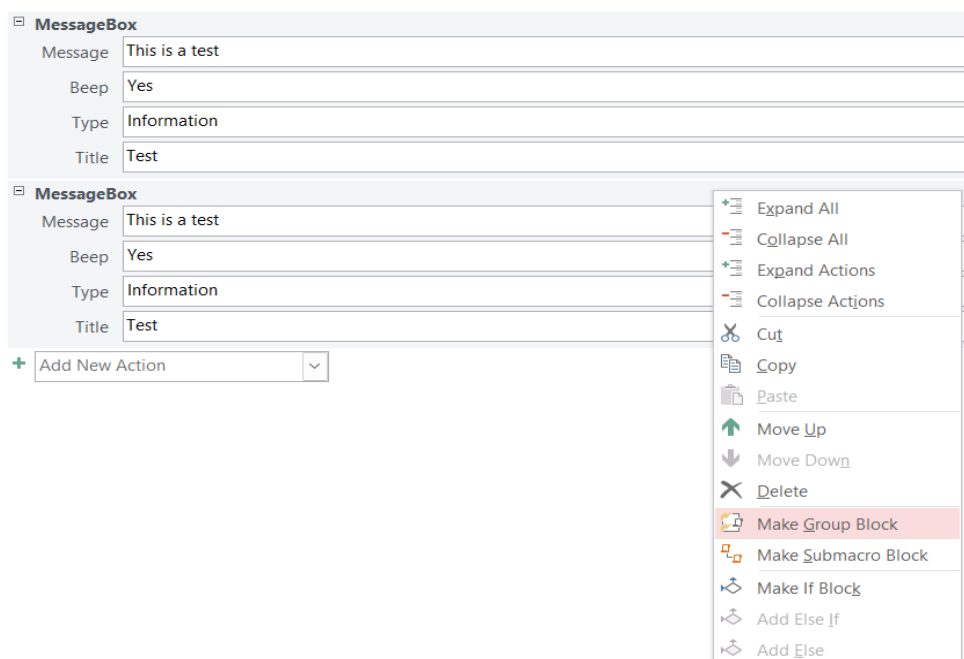


Figure 7.14: Creating a Group Block (1)

If the actions are not already present:

- Drag the **Group Block** from the Action Catalog into the macro pane.
- In the box at the top of the group block, type a name for the group.
- Drag macro actions from the Action Catalog into the group block, or select actions from the **Add New Action** list that appears within the block.

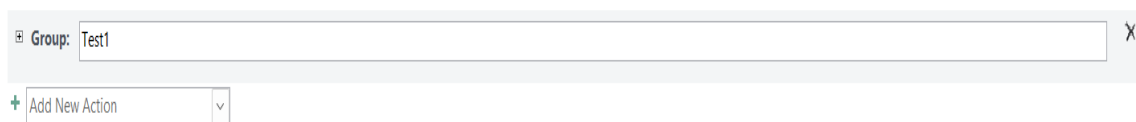


Figure 7.15: Creating a Group Block (2)

e. Expanding and Collapsing Macro Actions or Blocks

When you create a new macro, the macro builder displays macro actions with all the arguments visible. Depending on the size of the macro, you may want to collapse some or all macro actions (and action blocks) while editing the macro. This allows you to get a clearer overview of the structure of your macro. You can expand the necessary actions or blocks to modify them.

A. Expanding or Collapsing a Single Macro Action or Block

Click the **plus (+)** or **minus (-)** sign to the left of the macro or block name. You can also use the **up** and **down** arrow keys to select an action or block, then press the **left** or **right** arrow keys to collapse or expand it.

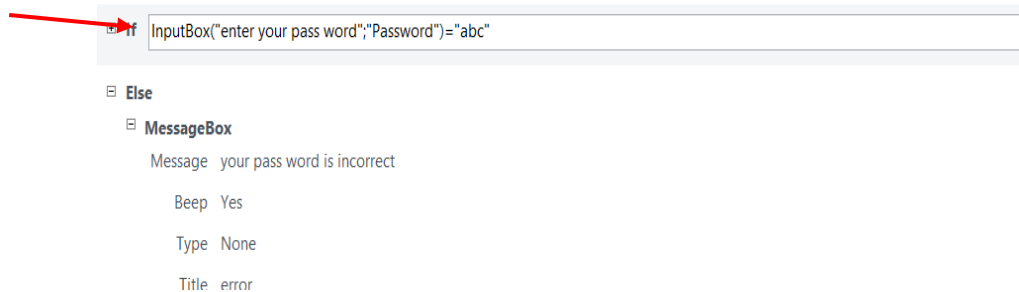


Figure 7.16 : Expanding or Collapsing a Single Macro Action

B. Expanding or Collapsing All Macro Actions (But Not Blocks)

In the **Create** tab, in the **Expand/Collapse** group, click **Expand Actions** or **Collapse Actions**.



Figure 7.17: Expanding or Collapsing All Macro Actions

C. Expanding or Collapsing All Macro Actions and Blocks

In the **Create** tab, in the **Expand/Collapse** group, click **Expand All** or **Collapse All**.



Figure 7.18: Expanding or Collapsing All Macro Actions and Blocks

7. Controlling Program Flow with If, Else If, and Else Conditions

To execute macro actions only when certain conditions are true, use an **If** block. You can extend an **If** block with additional **Else If** and **Else** blocks, just like in other sequential programming languages such as VBA.

- In the area at the top of the **If** block, type an expression that determines when the block will execute. The expression must be Boolean (i.e., a value of yes or no).
- Add actions to the **If** block by selecting them from the **Add New Action** dropdown list that appears in the block, or by dragging them from the **Action Catalog** pane into the **If** block.
- If you add an **Else If** block, type an expression that determines when the block will execute. The expression must be Boolean (i.e., true or false).
- To add actions to the **Else If** or **Else** dropdown list, select them from the **Add New Action** dropdown that appears in the block, or drag them from the **Action Catalog** pane into the block.

The following illustration shows a simple **If** block:

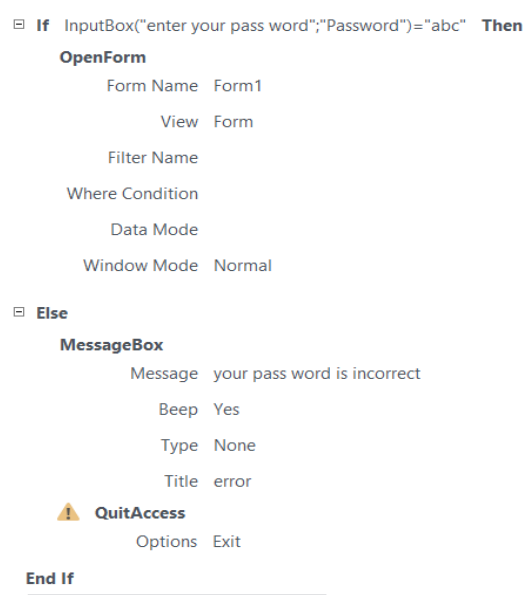


Figure 7.19: Illustration of the If Function

The **If** block is executed if the first name entered in the dialog box is Mohamed. The **Else** block is executed if the first name entered in the dialog box is not Mohamed.

8. Creating Submacros

Each macro can contain several submacros. A submacro is designed to be called by its name from the **RunMacro** or **OnError** macro actions.

To add a submacro block to a macro, do so the same way you would add a macro action, as described in the previous section. Once you've added a submacro block, you can drag macro actions into it or select actions from the **Add New Action** list that appears in the block.

9. Sharing a Macro with Others

When you copy macro actions to the clipboard, they can be pasted in XML (Extensible Markup Language) format into an application that accepts text. This allows you to send a macro to a colleague via email or post it on a discussion forum, blog, or other website. The recipient can then copy the XML code and paste it into their macro builder. The macro will be recreated exactly as you wrote it.

```
<?xml version="1.0" encoding="UTF-16" standalone="no"?>
<UserInterfaceMacros
xmlns="http://schemas.microsoft.com/office/accessservices/2009/11/application"><UserInterfaceMacro
MinimumClientDesignVersion="14.0.0000.0000"><Statements><ConditionalBlock><If><Condition
n>InputBox("Entrer un prénom" condition Si)="Mohamed"</Condition><Statements><Action
Name="MessageBox"><Argument Name="Message">Mohamed existe dans la
table</Argument><Argument Name="Type">Information</Argument><Argument
Name="Title">Condition
vérifiée</Argument></Action></Statements></If></ConditionalBlock></Statements></UserInterfaceMacro></UserInterfaceMacros>
```

Figure 7.20: Example of an Action in XML Format

10. Converting a Macro into VBA Code

Macros provide a subset of commands available in the Visual Basic for Applications (VBA) programming language. If you need more functionality than the provided macros offer, you can easily convert a standalone macro object into VBA code, and then use the extended set of features provided by VBA. However, note that this VBA code will not run in a browser; any VBA code you add to a web database will only execute when the database is opened in Access. You can use Access to automatically convert macros into VBA modules. To convert a macro into VBA code:

- In the navigation pane, right-click on the macro object, then click **Design View**.
- In the **Design** tab, go to the **Tools** group, and click **Convert Macros to Visual Basic**.

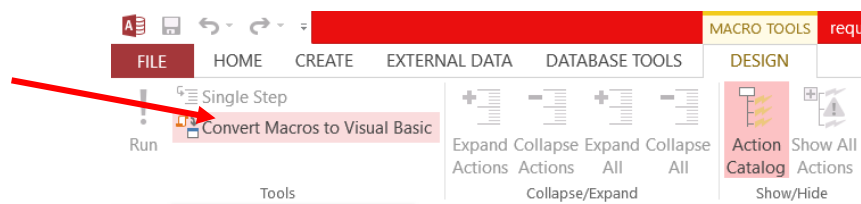


Figure 7.21 : Converting a Macro to Visual Basic

- In the **Convert Macro** dialog box, specify whether you want the error-handling code and comments to be added to the VBA module, then click **Convert**. Access will confirm that the macro has been converted and will open the **Visual Basic Editor**. Double-click the converted macro in the **Project** pane to view and modify the module.

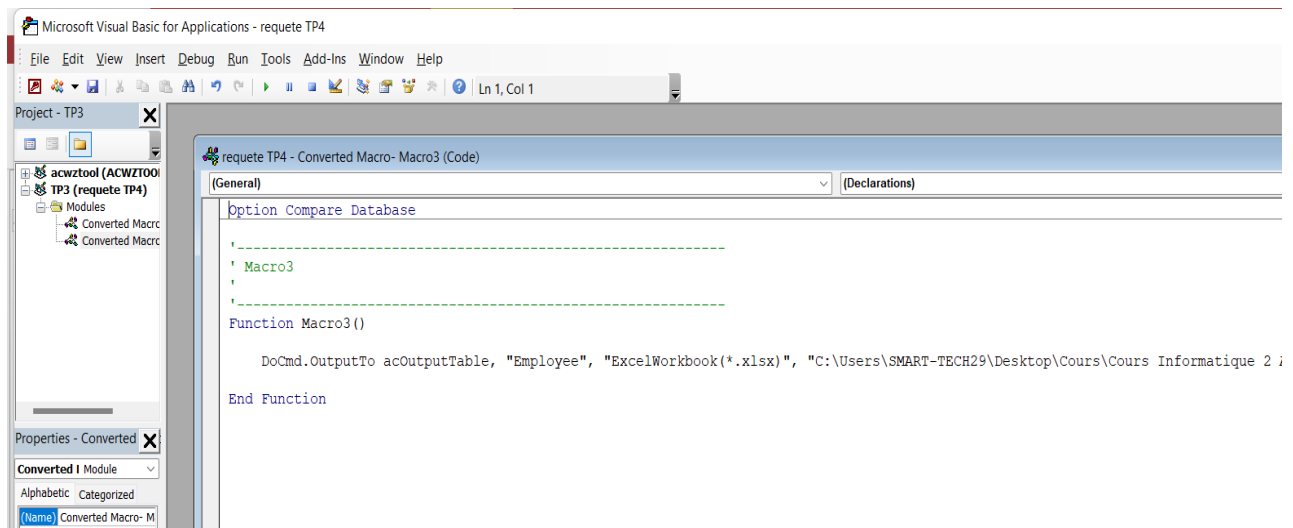


Figure 7.22: The Macro in Visual Basic

a. Converting Macros Attached to a Form or Report

This process converts all macros referenced by (or embedded in) a form or report into VBA code and adds the VBA code to the class module of the form or report. The class module becomes an integral part of the form or report and moves with it when the form or report is moved or copied.

- In the **Navigation Pane**, right-click on the form or report, and then click **Design View**.
- In the **Design** tab, go to the **Tools** group, then click **Convert Form Macros to Visual Basic** or **Convert Report Macros to Visual Basic**.

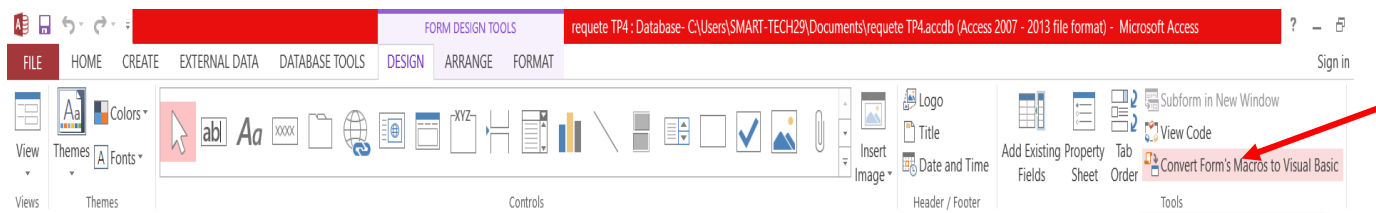


Figure 7.23: Converting Form Macros to Visual Basic

- In the **Convert Form Macros** or **Convert Report Macros** dialog box, specify whether you want Access to add error handling code to the functions it generates. Additionally, if your macros contain comments, indicate whether you want them to be included as comments in the functions. Click **Convert** to continue.

Access converts the macro and opens the **Visual Basic Editor**. To view and edit the VBA code:

- In **Visual Basic Editor**, if the **Project Explorer** pane is not visible, click **Project Explorer** in the **View** menu.
- Expand the tree under the name of the database you are working in.
- Under **Modules**, double-click the **Converted Macro Module - Macro Name**. **Visual Basic Editor** will open the module.

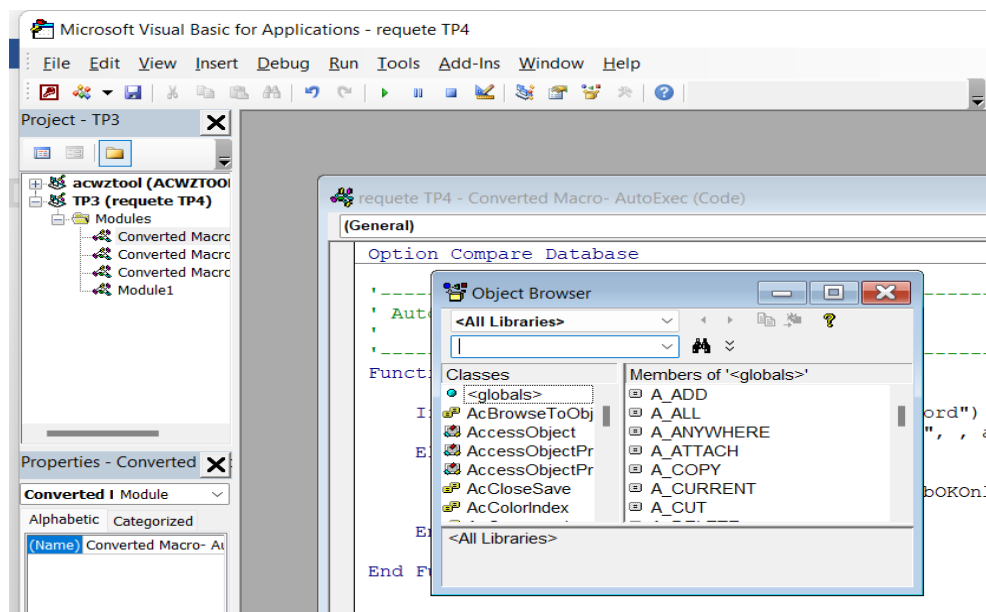


Figure 7.24: Visual Basic Object Explorer

b. Attaching a VBA Function to an Event Property

When you convert a global macro to VBA code, the VBA code is placed in a standard module. Unlike a class module, a standard module is not part of a form or report. You can link the function to an event-type property on a form, report, or control so that the code is executed exactly when and where you want. To do this, you can either copy the VBA code into a class module and link it to an event-type property, or you can make a special call from the event property to the standard module using the following procedure:

1. In **Visual Basic Editor**, note the function name. For example, if you converted a macro named **MyMacro**, the function name will be **MyMacro()**.
2. Close **Visual Basic Editor**.
3. In the **Navigation Pane**, right-click the form or report to which you want to link the function, then click **Design View**.
4. Click the control or section to which you want to attach the function.
5. Show the **Property Sheet**.
6. In the **Event** tab of the property sheet, click the event property box to which you want to associate the function.
7. In the property box, type an equal sign (=) followed by the function name, for example, **=MyMacro()**. Be sure to include the parentheses.
8. Save the form or report by clicking **Save**.
9. In the **Navigation Pane**, double-click the form or report and test it to verify that the code runs as expected.

You now know the basic steps to add VBA code to your database. This course only covers the basics to get you started. There are many reference books and online resources that can help you develop your programming skills.

Practical work sheets

Practical work N 1

Exercise 1

Consider the following invoice:



Camping le source

Liste des Achats

NumCli 156

Nom:

Prénom:

Adresse:

Code postal

Ville

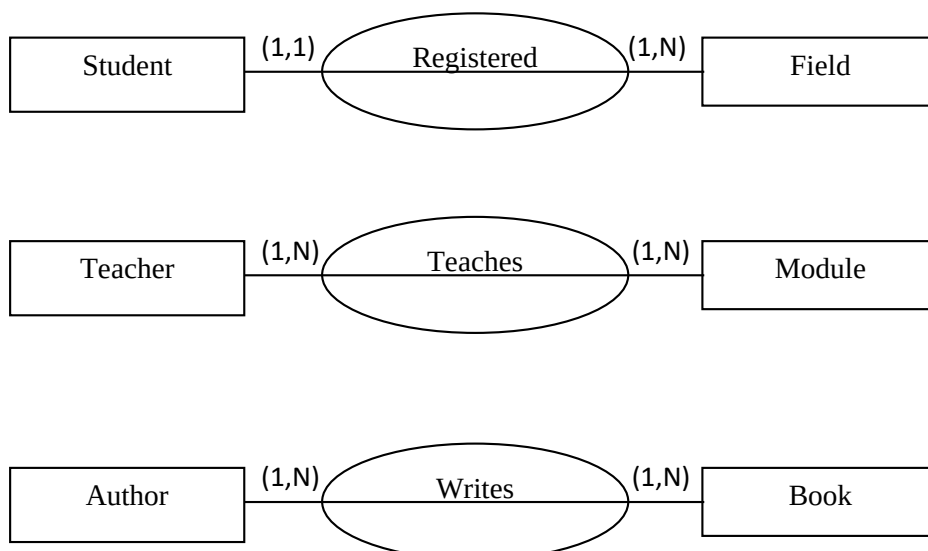
Téléphone:

Date	CodeArticle	Désignation	Qté	Prix	Total
14/07/2012	RP003	Repas "poisson"	4	40	160
15/07/2012	B001	Café	1	10	10
15/07/2012	GL004	Glace "Magnum"	2	15	30
16/07/2012	BG020	Baguette	1	1,2	1,2
				Total dû:	201,2

Task requested: develop the data dictionary.

Exercise 2:

Translate the following Relations:



Exercise 3: Library Management

The computer system of a library records the national number, the last name, the first name, and the address (street, number, postal code, city) of each of its clients. The system records for each available book its unique ISBN number, its title, the names of its authors, and its purchase date. The system also records all client loans. These loans are identified by a unique number, and the loan date and return date are recorded when the book is returned.

- Identify the entities, associations, and attributes of each entity.
- Provide the conceptual data model (CDM).

Exercise 4: Purchasing department

Based on the following collected data, develop the corresponding conceptual data model (CDM). For our purchasing service, we can identify a certain number of elements:

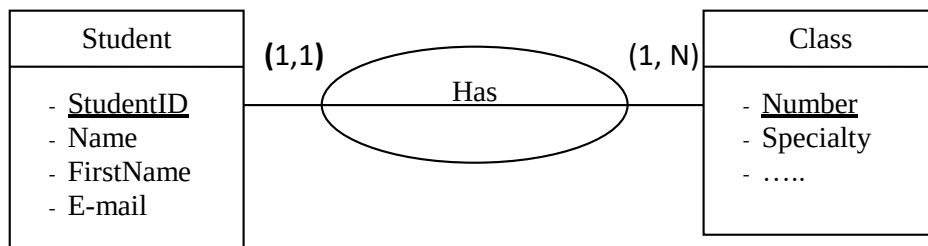
- | | |
|---------------------------------|-----------------------------|
| • Supplier name | • Unit price of the Product |
| • Order date | • Product description |
| • Order number | • Quantity ordered |
| • Stock quantity of the Product | • Supplier Phone number |
| • Product reference | • Supplier address |

Practical work N 2

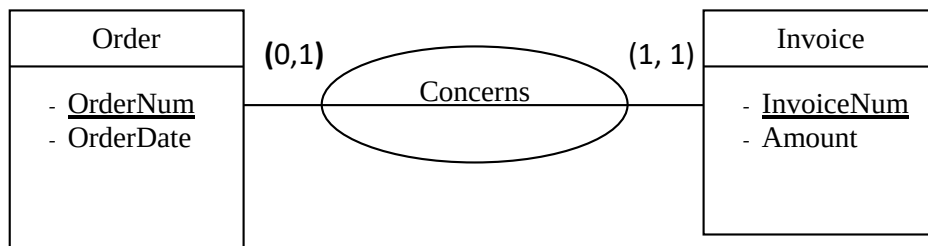
Exercise 1

Transform the following CDMs into LDMs while respecting all the rules for transitioning from CDM to LDM.

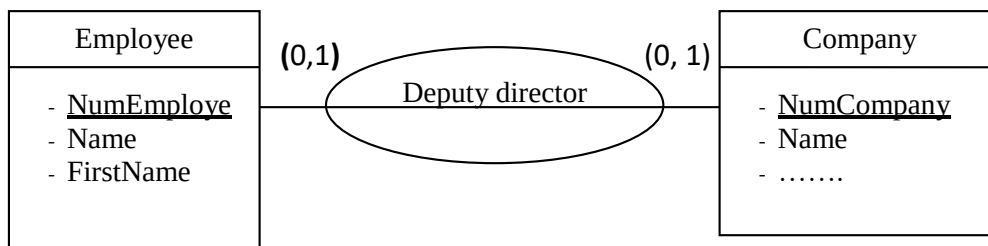
1.



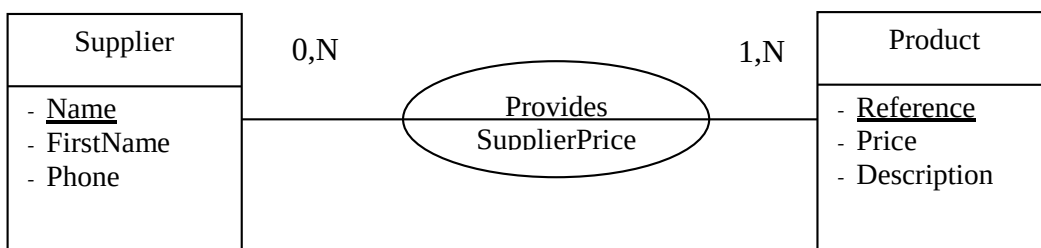
2.



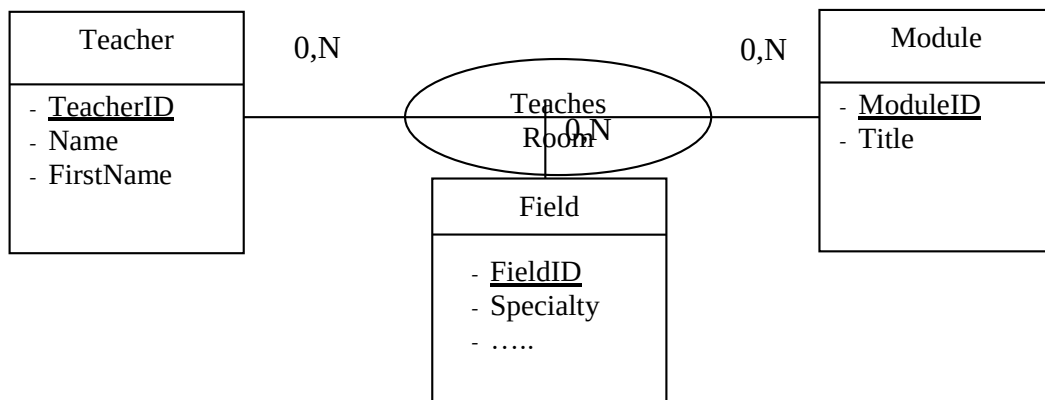
3.



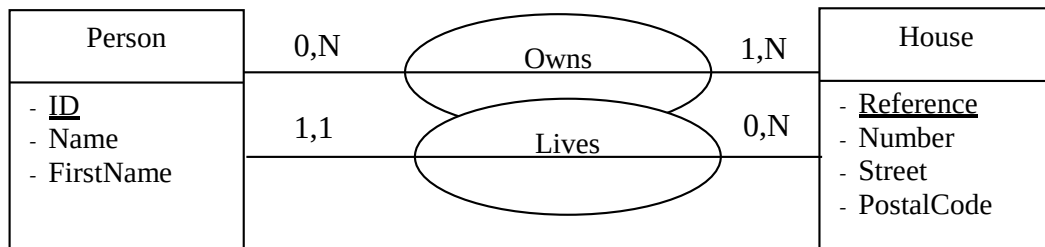
4.



5.

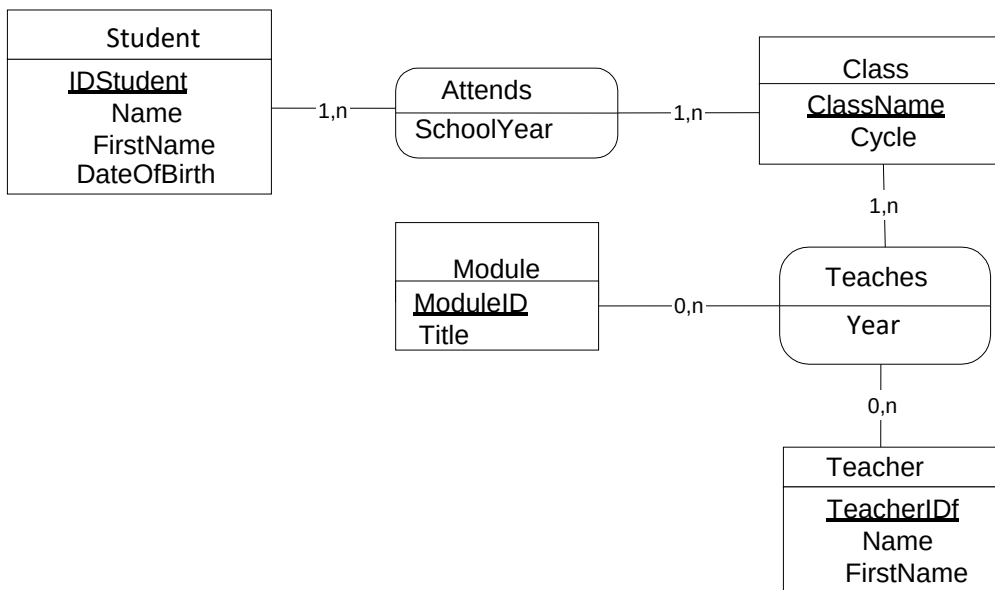


6.



Exercise 2:

Transform the following CDM, which represents 'the management of a school,' into an LDM while adhering to all the rules for transitioning from CDM to LDM.



Exercise 3:

Consider the following relational schema:

Flight (FlightID, #PilotID, #AirplaneID, DepartureTime, ArrivalTime, DepartureCity, ArrivalCity)

Pilot (PilotID, name, City)

Airplane (AirplaneID, Brand, Price)

- Propose the MCD that was the source of this relational schema.

Practical work N 3
Creation of tables in a database.

Part 1:

Create a database, Create tables, Enter data, Create relationships

We want to create a database for managing employees in a company based on the following relational schema:

EMPLOYEE (EmployeeID, LastName, FirstName, Position, Grade, Supervisor, HireDate, DepartmentID)

DEPARTMENT (DepartmentID, DepartmentName)

SALARY (Grade, MinSalary, MaxSalary)

Explanation of attributes:

Attribute	Meaning	Type	Lenght/ Size	Note
<i>EmployeeID</i>	Employee ID	Number	Integer	Primary Key - Employee
<i>LastName</i>	Employee's Last name	Text	20	
<i>FirstName</i>	Employee's First name	Text	20	
<i>Position</i>	Employee's Position	Text	30	
<i>Grade</i>	Grade de l'employé	Number	Integer	Primary Key - Salary
<i>Supervisor</i>	Supérieure hiérarchique	Number	Integer	Can be null
<i>HireDate</i>	Recruitment date	Date jj/mm/aa	Date	
<i>DepartmentID</i>	Department ID	Number	Integer	Primary Key - Department
<i>DepartmentName</i>	Department Name	Text	15	
<i>MinSalary</i>	Minimum salary for a given grade	Currency	Standard	
<i>MaxSalary</i>	Maximum salary for a given grade	Currency	Standard	

Work to be done :

1. Create the **DEPARTMENT**, **SALARY**, and **EMPLOYEE** tables in Design View.
2. Define the primary keys for all three tables.
3. Enter the data below into the following tables:

- **SALARY :**

Grade	MinSalary	MaxSalary
1	28 000,00 €	32 000,00 €
2	32 001,00 €	36 000,00 €
3	36 001,00 €	40 000,00 €
4	40 001,00 €	44 000,00 €
5	44 001,00 €	50 000,00 €

- **DEPARTMENT :**

DepartmentID	DepartmentName
101	Humaine Ressources
102	IT
103	Production
104	Accounting

- **EMPLOYEE :**

EmployeeID	LastName	FirstName	Position	Grade	Supervisor	HireDate	DepartmentID
1	Ayyadi	Mohamed	Director	5		12/04/2015	101
2	Fellah	Mustafa	Administrator	4	1	16/11/2019	104
3	Ghoumari	Younes	Developer	3	2	16/06/2020	103
4	Allal	Fatima	Administrator	2	1	20/03/2021	101
5	Bakhti	Ines	Accountant	3	2	25/11/2020	104

4. By following these steps, you will modify your database according to the given specifications.
 - A. Require the entry of the following fields: EmployeeID, departmentID, Grade
(Go to General Tab / Property: Required / Set to Yes)
 - B. Create an input mask for HireDate in the format dd/mm/yyyy
(Go to General Tab / Property: Input Mask / Enter 00/00/0000) to ensure the date is entered in the DD/MM/YYYY format.
 - C. Force the display of the LastName field in uppercase:
In the **EMPLOYEE** table, select the **LastName** field.
In the **General** tab, set the **Format** property to >, which will automatically convert the text to uppercase.
 - D. Insert a "Title" field between EmployeeID and LastName
In the **EMPLOYEE** table in Design View, right-click on the EmployeeID field and select Insert Field.

Name the new field Title, set its data type to Text and adjust the field size if necessary.

- E. Add a drop-down list for the "Title" field with the following options: Mrs; Miss; Mr

In the **Title** field, go to the **Data Type** column and select **Lookup Wizard**.

Choose **I will type in the values that I want** and enter the values:

Mrs; Miss; Mr.

- F. Set the default value for HireDate to "Now" (current date and time)
In the **EMPLOYEE** table, select the **HireDate** field.

In the **General** tab, set the **Default Value** property to `Now ()`, which will automatically fill in the current date and time when a new record is created.

- G. Validate the MinSalary field to ensure it is greater than 18,000. If not, display an error

In the **SALARY** table, select the **MinSalary** field.

In the **General** tab, under the **Validation Rule** property, set the following: `>=18000` to ensure the salary is above the minimum.

In the **Validation Text** property, set the message to: (The salary is below the minimum).

- H. Set a legend for the SALARY table to display: "Grade", "Minimum Salary", "Maximum Salary"

In the **SALARY** table, go to the **General** tab for each field:

For **Grade**, set the **Caption** property to **Grade**.

For **MinSalary**, set the **Caption** to **Minimum Salary**.

For **MaxSalary**, set the **Caption** to **Maximum Salary**.

- I. Hide the Grades of Employees
(In the **DataSheet View**, right-click on the **Grade** field header and choose **Hide Field** or uncheck it in the table properties).

After completing these steps, your database should be properly set up with all the necessary modifications! Make sure to save your work and test the changes by entering some data to confirm everything works as expected.

5. Establish relationships between the EMPLOYEE, DEPARTMENT, and SALARY tables.

Part 2:

Sorting, Filtering, and Searching.

Use the previously created database, containing the tables: EMPLOYEE, DEPARTMENT, and SALARY.

Tasks to Complete:

1. Sorting

- Display the EMPLOYEE table sorted in descending order by Position.
- Display the EMPLOYEE table sorted in ascending order by title and employee last name.

2. Filtering

- Display the list of employees who work in department 103.

3. Searching

- Search for employees with the last name "ALLAL".
- Replace the last name of the employee "ALLAL" with "ALLALI".

Practical work N 4
With Solution

Using the database "BaseTP3" created in TP3 for managing employees of a company, create the following queries:

Selection Queries :

1. Display the list of employees (first name and last name) sorted in ascending alphabetical order.

Field:	LastName	FirstName			
Table:	Employee	Employee			
Sort:	Ascending				
Show:	☒	☒	☐	☐	☐
Criteria:					
or:					

2. Display the list of Administrator employees (first name and last name).

Field:	LastName	FirstName	Position	
Table:	Employee	Employee	Employee	
Sort:				
Show:	☒	☒	☐	☐
Criteria:			"Administrator"	
or:				

3. Display the list of employees (first name, last name) whose grade is less than 4.

Field:	LastName	FirstName	Grade	
Table:	Employee	Employee	Employee	
Sort:				
Show:	☒	☒	☐	☐
Criteria:			>4	
or:				

4. Display the list of employees (first name, last name) whose last name starts with "A".

Field:	LastName	FirstName		
Table:	Employee	Employee		
Sort:				
Show:	☒	☒	☐	
Criteria:	Like "A*"			
or:				

5. Display the list of employees (first name, last name, department name) with a minimum salary < 38,000 DA.

The screenshot shows the Microsoft Access Query Design View for a query named "Salary". The query is based on the Employee and Department tables. The criteria for the query are: LastName, FirstName, and DepartmentName are all checked in the "Show" row, and MinSalary is set to "< 38000" in the "Criteria" row.

Field:	LastName	FirstName	DepartmentName	MinSalary		
Table:	Employee	Employee	Department	Salary		
Sort:						
Show:	☒	☒	☒	☐	☐	☐
Criteria:				< 38000		
or:						

Parameterized Selection Queries:

6. Display the list of employees (first name, last name, and department) whose position is entered by the user.

The screenshot shows the Microsoft Access Query Design View for a query named "Employee". The query is based on the Employee and Department tables. The criteria for the query are: LastName, FirstName, and DepartmentName are all checked in the "Show" row, and Position is set to "[Enter the position]" in the "Criteria" row.

Field:	LastName	FirstName	DepartmentName	Position
Table:	Employee	Employee	Department	Employee
Sort:				
Show:	☒	☒	☒	☐
Criteria:				[Enter the position]
or:				

7. Display the list of employees (first name, last name) whose salary is lower than the salary entered by the user.

Field:	LastName	FirstName	MinSalary
Table:	Employee	Employee	Salary
Sort:			
Show:	☒	☒	☐
Criteria:			<[Enter the minimum sa
or:			

Calculation Queries:

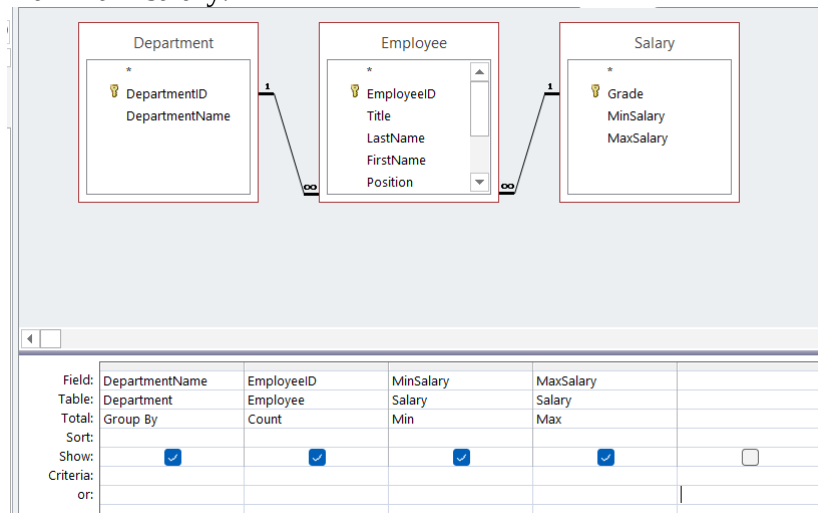
8. Display the number of employees by Position.

Field:	EmployeeID	Position		
Table:	Employee	Employee		
Total:	Count	Group By		
Sort:				
Show:	☒	☒	☐	☐
Criteria:				
or:				

9. Display the number of employees in the "Accounting" department.

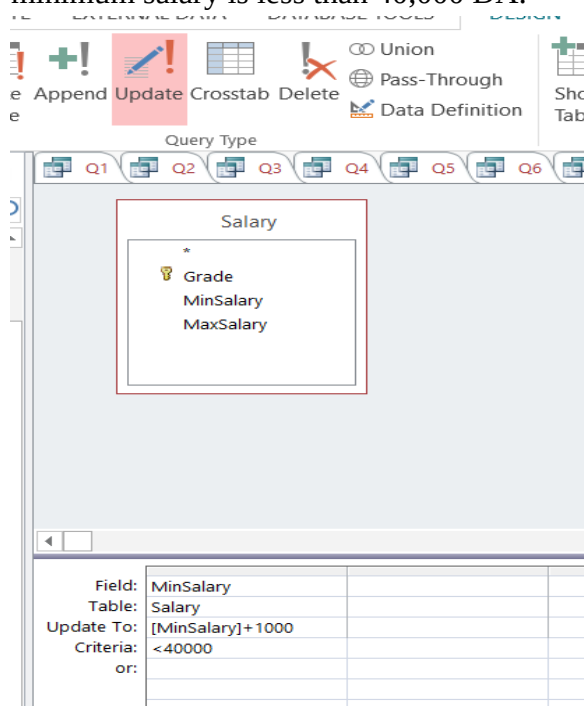
Field:	EmployeeID	DepartmentName
Table:	Employee	Department
Total:	Count	Group By
Sort:		
Show:	☒	☒
Criteria:		"Accounting"
or:		

10. For each department, display the number of employees, the minimum salary, and the maximum salary.



Update Queries :

11. Create a query to increase the minimum salary (+1,000 DA) for employees whose minimum salary is less than 40,000 DA.



Creation Queries :

12. Create a table Employee1 (EmployeeID, LastName, Position, Grade) containing the list of employees whose grade is less than 4.

The screenshot shows the Microsoft Access Query Design view for a query named 'Employee'. The design grid is empty. The table 'Employee' is selected in the 'Show Table' task pane. The table fields are: Title, LastName, FirstName, Position, Grade, and Supervisor. The 'Criteria' row in the design grid is set to '< 4' for the 'Grade' field.

Field:	EmployeeID	LastName	Position	Grade	
Table:	Employee	Employee	Employee	Employee	
Sort:					
Show:	☒	☒	☒	☒	☐
Criteria:				< 4	
or:					

Append Queries :

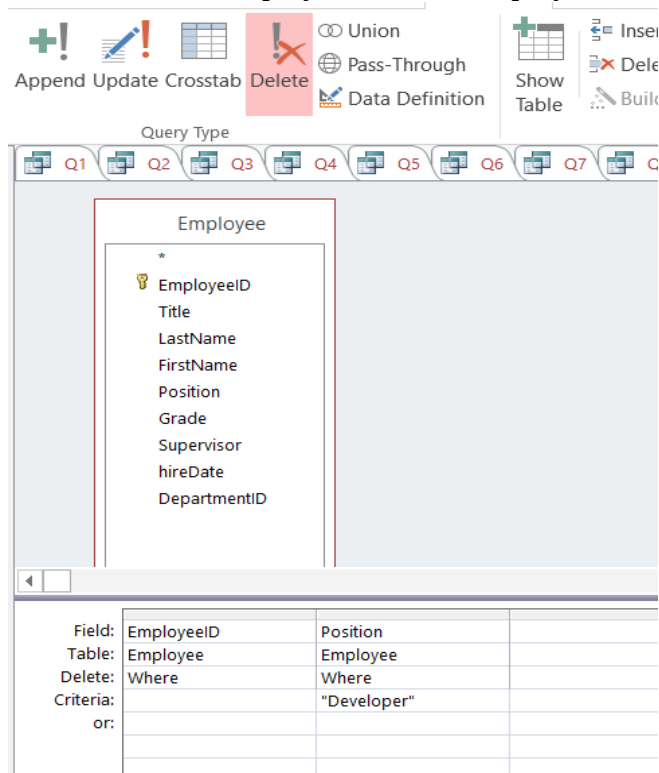
13. Add to the Employee table the list of employees grouped in the NewEmployee table, which has the same schema as the Employee table.

The screenshot shows the Microsoft Access Query Design view for an append query. The design grid is empty. The table 'NewEmployee' is selected in the 'Show Table' task pane. The table fields are: *, EmployeeID, Title, LastName, FirstName, and Position. The 'Append To' row in the design grid is set to 'Employee.*'.

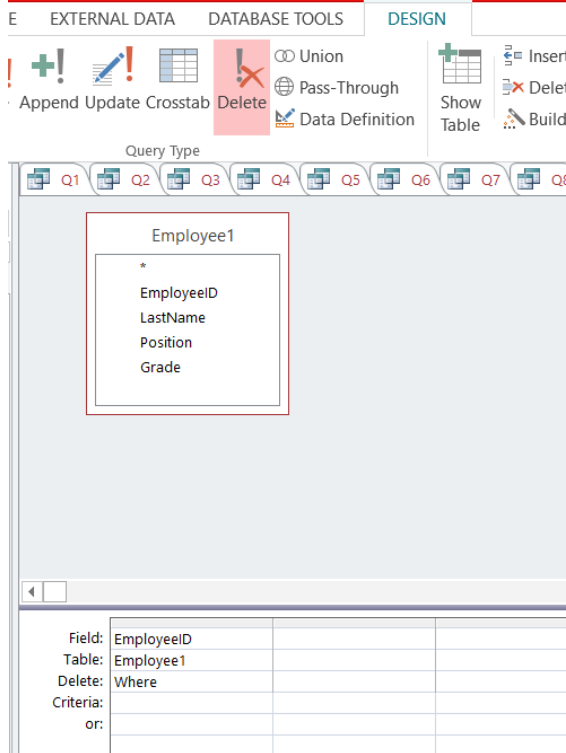
Field:	NewEmployee.*				
Table:	NewEmployee				
Sort:					
Append To:	Employee.*				
Criteria:					
or:					

Delete Queries :

14. Delete from the Employee table the employees with the Position is Designer.



15. Delete all records from the Employee1 table.



Practical work N 5
Creation of Forms in a database.

Using the database "BaseTP3" created during TP3 for employee management in a company, create the following forms:

1. Using the form wizard, create a form for Departments. Then, apply the necessary formatting.
2. Create a form for salaries:
 - With the title: Salary Management Form.
 - Disable the visibility of the selector and navigation buttons.
 - Add control buttons to navigate between records.
 - Save the form.
3. Create a form for employees:
 - With the title: Employee Management Form.
 - Add control buttons for: New, Search, and Delete.
 - Modify the form created to add the company header and the current date.
 - Save the form.
4. Create a form (with a subform) to display the list of employees by department.
5. Create a welcome form that contains navigation buttons to move between the different forms, along with a button to exit.

Practical work N 6
Creation of Reports in a database.

Using the "BaseTP3" database created during TP3 for managing employees in a company, create the following reports:

1. Using the Report Wizard, create a report to display the list of employees grouped by their department and sorted by employee ID. Then, apply the necessary formatting:
 - Add the title: Employee List.
 - In the Report Footer, insert the total number of employees.
 - Make the necessary formatting adjustments.
2. In the main form, add a button to print the list of all employees.
3. Create a report to display information about each selected employee.
4. Create a report to display salaries.
5. Create a report to display departments.
6. Create a report to display the results of query Q2 (created in TP4) which shows the list of Administrator employees (first name and last name).
7. Create a report to display the results of query Q6 (created in TP4) which shows the list of employees (first name, last name, and department) based on the position entered by the user.
8. In the main form, add another button to print this report.

Practical work N 7
Creation of Macros in a database.

Using the "BaseTP3" database created during TP3 for employee management, create the following macros:

1. Create a macro (macro1) that opens the form containing the employee list.
2. Link this macro to a button in the "Department Management" form.
3. Create a new macro (macro2) containing several actions:
 - Open the Salary table in read-only mode
 - Display a message box to close the Salary table.
 - Close the Salary table.
4. Link this macro (macro2) to the same button created in the "Department Management" form under the (On Click) event.
5. Create a macro (macro3) "**Export with formatting**" that exports the Employee table to an Excel file named **EmployeeList.xlsx** in your working folder.
6. Convert this macro (macro3) to VBA code.
7. Create a macro (macro4) that launches the welcome form with a password condition (password = abc). If the password is incorrect, an error message is displayed, and access to the database is denied.
8. Protect the database by using **AutoExec** to automatically run this macro (macro4) when the database starts.

Bibliography

Students may refer to the following references:

- **Flory, A.** (2000). *Modélisation des systèmes d'information: UML, Merise et les autres...* Vuibert.
- **Date, C. J.** (2003). *An Introduction to Database Systems* (8th ed.). Addison-Wesley.
- **Microsoft Corporation.** (2020). *Microsoft Access Help and Training*. Retrieved from: <https://support.microsoft.com/access>
- **Chapman, C. N.** (2004). *Microsoft Access 2003 Bible*. Wiley Publishing.
- **Fouché, A.** (2012). *Systèmes d'information et bases de données relationnelles*. Dunod.