CHAPTER 5: MANAGE DATABASE SYSTEMS (MS. Access)

Unit of learning code: ICT/CU/IT/CR/4/6

Related Unit of Competency in Occupational Standard: Manage Database Systems (MS Access)

5.1 introduction to the unit of learning

This unit specifies competencies required to manage database system. Which involve identifying database concepts, designing a basic database, creation and manipulation of database objects, database testing and printing of database objects. This provides the learner with the competencies in achieving the above listed items.

5.2 Summary of Learning Outcomes

1. Identify database concepts.
2. Design basic database
3. Create and manipulate database objects.
4. Perform database testing Print database objects.

5.2.1 Learning Outcome 1: Identify database concepts.

5.2.1.1 Introduction to the learning outcome

This section gives an overview of the various database principles and the basic components that make up a simple database environment. By the end of it, the learner should be able to competently identify the various components of a database, merits and demerits of a database and the various database models.

5.2.1.2 Performance Standard

5.2.1.2.1 Database concepts are defined.

5.2.1.2.2 Database models are identified.

5.2.1.2.3 Identification of merits and demerits of database is done.

5.2.1.3 Information Sheet

Data - Meaningful facts, text, graphics, images, sound, video segments

Database - Shared collection of **logically** related data (and a **description** of this data), designed to meet the information needs of an organization.

Information - Data processed to be useful in decision making

Metadata - Data that describes data

Fields - The place where data is placed within the database is called a field. One field holds one piece of data. If you are storing student details, possible fields would include Name, Admission number, Age

Records - All the fields for one student constitute a record, Records ensure which name relates to which admission number, age

Table - A collection of records that describe similar data is called a table. A database for student details could have tables for admission information, fees, and exams.

Manual filing Systems - These are filing systems that had the following properties with regards to records management.

i. Records were kept in manual workbooks.
ii. Records were Updated manually.
iii. There was Wear and tear of records.
iv. It was prone to accidental loss of data.
v. It was Very difficult to retrieve data.

File based systems - A file-based system is a collection of application programs that perform services for the end users. Each program within a file-based system defines and manages its own data. Because of this, there are limitations as to how that data can be used or transported. There is data dependency.

(a) Advantages of FBS

i. File based systems were developed as better alternatives to paper-based filing systems.
ii. By having files stored on computers, the data could be accessed more efficiently.
iii. It was common practice for larger companies to have each of its departments looking after its own data.

(b) Disadvantages of file based systems

i. Data separation and isolation

Each program maintains its own set of data.

Users of one program may be unaware of potentially useful data held by other programs.

ii. Duplication of data

Same data is held by different programs.

Wasted space and potentially different values and/or different formats for the same item.

iii. Data dependence

File structure is defined in the program code.

If the file structure is changed then the program code will also change.

iv. Incompatible data (different file formats)

Programs are written in different languages, and so cannot easily access each other's files.

v. Lack of flexibility in organizing and querying the data

Users cannot directly query data held in different incompatible program files

vi. Increased number of different application programs

Programs are written to satisfy particular functions.

Any new requirement needs a new program.

Database approach

The database approach involves storing the data using computers in a central repository through the use of a Data Base Management System.

Motivation:

– Definition of data was embedded in application programs, rather than being stored separately and independently.
– No control over access and manipulation of data beyond that imposed by application programs.

Description of database approach

— There is central repository of shared data.
— Data storage and retrieval is done by DBMS.

— Data is stored in standardized and convenient form.

A database is typically made up of the following basic components:

i. Hardware-These are the tangible components that make up the database such as physical storage devices.
ii. Software-These are the software programes that manage the database such as the Database Management System, Antivirus etc.
iii. Data-These are the actual records which are stored inside the database.
iv. Users-These are those people who interact with the database in one way or another such as data entry clerks, database administrators, programmers etc.
v. Rules and procedures-these are the various guidelines and procedures that define how data can be accessed.

(a) Advantages of database approach

i. Data independence

The data is held in such a way that changes to the structure of the database (schema) does not affect any programs used to access the data.

ii. Data consistency

Each item of data (record) is held only once and therefore there is no danger of a record being updated in one system and not the other.

iii. Control's data redundancy

In file-based systems the same information may be held in several files. This wastes space and makes update of records more time consuming. Database approach minimizes data redundancy.

iv. Data integrity

The DBMS provides users with the ability to specify constraints on data such as making a field entry essential or using validation rules.

v. Data security

The DBMS can ensure that only authorized users can access and modify the database.

vi. Centralized control of data

The database administrator will control who has access to what and will structure the database to match the needs of the organization.

vii. More information available to users

Users have access to wider range of data that was previously held in separated departments and sometimes on incompatible formats.

viii. Increased productivity:

The DBMS provides an easy-to-use query language that allows users to get immediate response from their queries rather than having to use programmers who will write queries.

(b) Disadvantages of database approach

i. *Larger size* - More disk space is required and probably a larger more powerful computer.
ii. *Greater complexity of the design* - For optimum use the database must be very carefully designed.
iii. *Greater impact of system failure* - in case the system fails, the operations of all departments will be affected.
iv. *Complex recovery procedures* -The process of recovering data from a central database is complex where backups are not available. It may require reconstruction of disks.

Database Models

These are a variety of ways to organize data within a database. A database model specifies the rules according to which data are structured and the associated operations that are permitted. It may also be a technique for the formal description of data structure, usage constraints and operations. The facilities available vary from one database to another. Each DBMS maintains a particular database model. A Database model is a combination of at least three components:

i. A collection of data structure types.
ii. A collection of operators or rules of inference, which can be applied to any valid instance of the data types listed in (i).
iii. A collection of general integrity rules, which implicitly or explicitly define the set of consistent database states or change of state or both.

a) Flat file database models

A flat file database model consists of one or more readable files normally stored in a text format. Information in these files is stored in fields. Every flat-file database is different because companies store different data and companies have different needs. After a flat- file has been created and data has been stored in these files, a method must be devised in order to retrieve the data, create new records, update records or delete records.

Table 33: Sample flat file system

Empno	FirstName	MidName	Surname	Gender	Age	Country
E0345	Timothy	James	Robert	Male	35	USA
E0767	Fred	Mathew	Wekesa	Male	34	USA
E0783	Odhiambo	Oluoch	Otieno	Male	31	Kenya
E0299	Denis	Musau	Mutie	Male	32	Kenya
E0852	Janeffer	Njeri	Kamau	Female	32	Kenya
E0920	Kellen	Ruth	Wambugu	Female	33	Kenya
E0218	Kelly	Johnson	Samson	Male	34	England
E0830	Henry	Geoffrey	Hesky	Male	35	England
E0666	Erastus	James	Jacob	Male	34	England
E0909	Daniel	Peterson	Job	Male	33	USA
E0606	Peter	William	Matekwei	Male	32	Kenya
E0981	James	Griffith	Ronald	Male	33	USA

Flat-file databases are created and used to migrate data from one database implementation to another particularly in relational databases.

Advantages of flat-file model

i. Easy to come up with

Disadvantages of flat-file model

ii. Do not promote a structure in which data can easily be related.

iii. It is difficult to control data duplication.

iv. The physical location of the data field within the file must be known.

v. A program must be developed to manage the data.

Examples of applications you can use to develop flat-file database word processors, spreadsheets.

b) Hierarchical database model.

A hierarchical database consists of a collection of records that are connected to each other through links (pointers). Each record is a collection of fields (attributes), each of which contains only one data value.

A link is an association between precisely two records. Consider a database that represents a customer-account relationship in a banking system. There are two record types: customer and account. The customer record type can be defined using three fields: customer name, customer-street, and customer-city. Similarly, the account record consists of two fields: account-number and balance. Now consider a sample database that shows that customer Hayes has account A-102, customer Johnson has accounts A-101 and A-201, and customer Turner has account A-305. Note that the set of all customer and account records is organized in the form of a rooted tree, where the root of the tree is a dummy node. A hierarchical database is a collection of such rooted trees, and hence forms a forest. Each such rooted tree shall be referred as a database tree. The content of a particular record may have to be replicated in several different locations. For example, in this customer-account banking system, an account may belong to several customers. The information pertaining to that account, or the information pertaining to the various customers to which that account may belong, will have to be replicated. This replication may occur either in the same database tree or in several different trees. Record replication has two major drawbacks:

- Data inconsistency may result when updating takes place.
- Waste of space.

A tree-structure diagram is the schema for a hierarchical database. Such a diagram consists of two basic components:

- Boxes, which correspond to record types
- Lines, which correspond to links

A tree-structure diagram serves the same purpose as an entity–relationship (E-R) diagram; namely, it specifies the overall logical structure of the database.

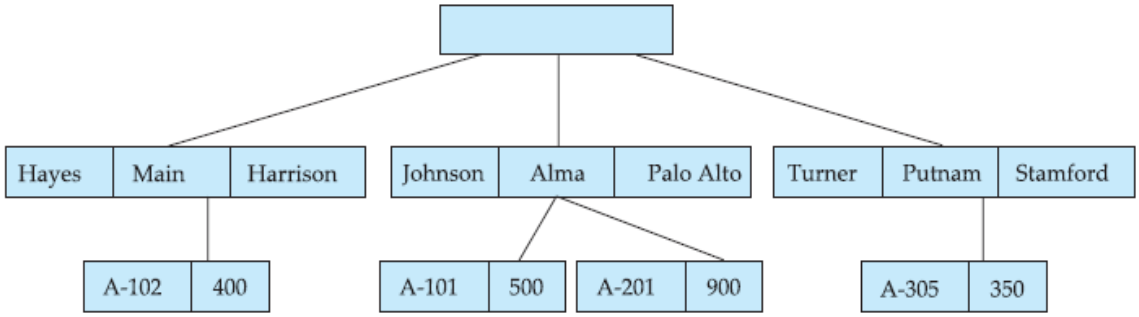


Figure 121: Hierarchical database model

To access the data, query by path navigation is used. Examples of databases designed based on this model are:

- File system
- IMS
- Windows Registry and Active Directory

Benefits of the hierarchical model over the flat-file model

- Data can be retrieved easily.
- Data integrity is easier to manage.

Drawbacks of the hierarchical model

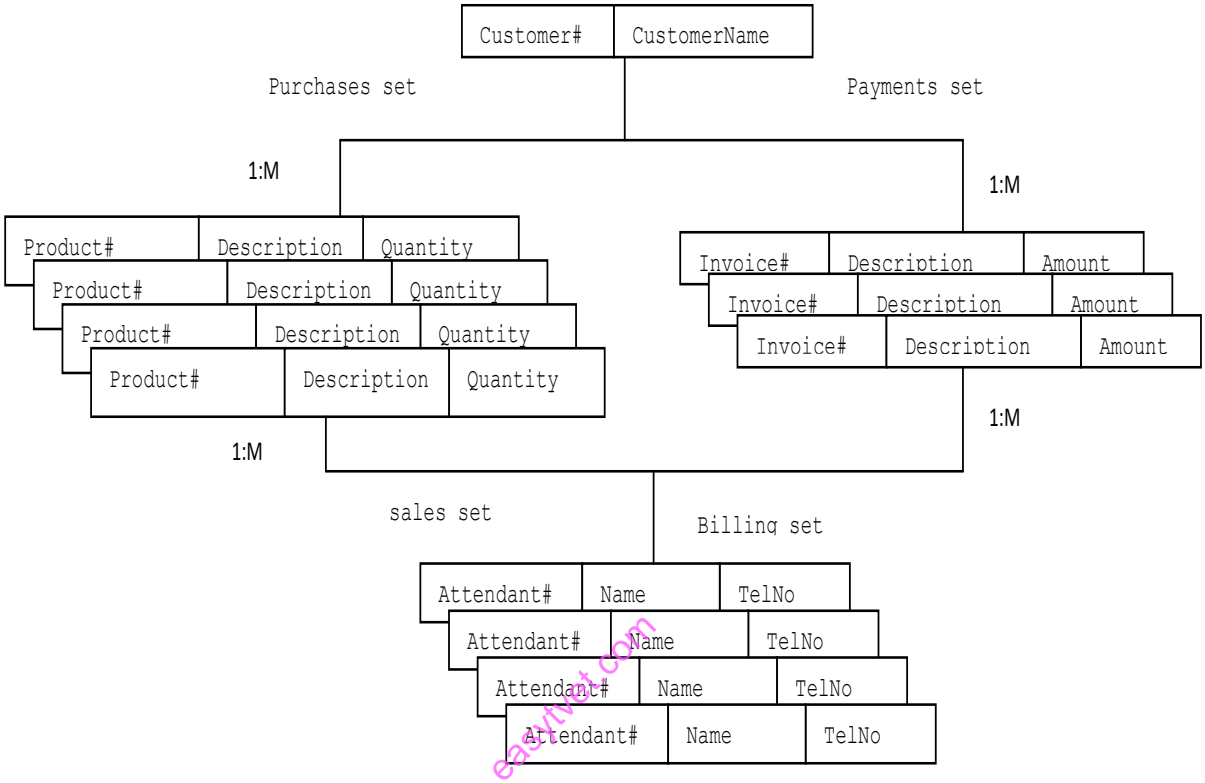
- Users must be familiar with the database structure.
- There is data redundancy.

c) Network Database Model

A network database model is a database model that allows multiple records to be linked to the same owner record type (parent record). The multiple linkages which this information allows the network database model to be very flexible. In addition, the relationship that the information has in the network database model is defined as many-to-many relationship because one owner record type can be linked to many member records and vice versa. This database model was created for three main reasons.

1. To represent complex data relationships more effectively.
2. To improve database performance.
3. Imposing a database standard

Figure 122:Network database model



Advantages of a Network Database Model

- Because it has the many-many relationship, network database model can easily be accessed in any table record in the database.
- For more complex data, it is easier to use because of the multiple relationship founded among its data.
- Easier to navigate and search for information because of its flexibility.

Disadvantage of a Network Database Model

- It may be Difficult to implement for first time users.
- Difficulties with alterations of the database because when information entered can alter the entire database.
- Hardware and software overheads

Applications:

- Integrated Database management system (IDMS)
- Integrated data store (IDS)

d) Object oriented database model

In this model, both data and their relationships are contained in a single structure called object. An object includes information about relationship between the facts within the object as well as information about its relationship with other objects. Objects with similar characteristics will form a class. In this model the database is modelled in form of objects and the relationships.

Class: An entity that has a well-defined role in the application domain, as well as state, behaviour, and identity

- Tangible: person, place or thing
- Concept or Event: department, performance, marriage, registration
- Artifact of the Design Process: user interface, controller, scheduler

Object: a particular instance of a class

State: attribute types (properties) and values

Behaviour: how an object acts and reacts (methods). Behaviour is expressed through operations that can be performed on it.

Example: Consider a scenario player described by player No, Player Name and Age plays for one and only one club. A club is described by club No, Club Name. A club plays several fixtures in a season. A fixture is described by fixture No, Date, venue, and time. Each fixture involves two clubs only. Draw an object database model for this scenario. Operations to be performed on this database include add, search, update and delete of records.

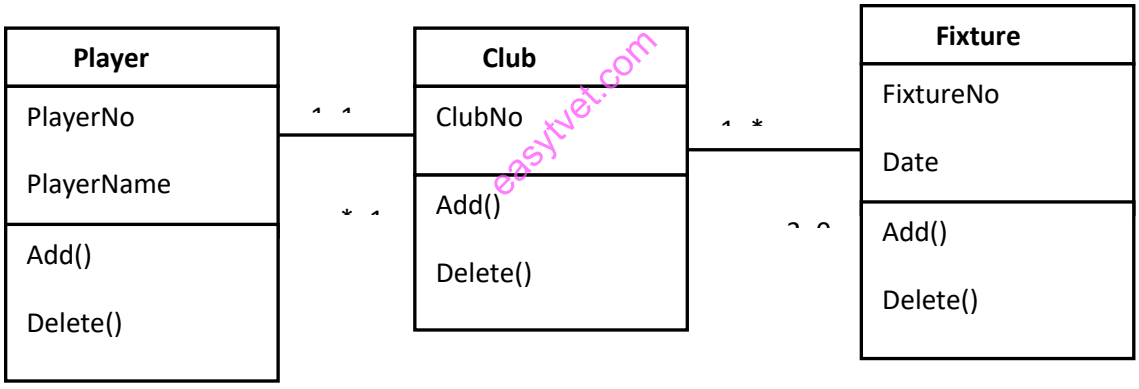


Figure 123: object-oriented database model

Advantages of object-oriented model

- High performance on certain tasks.
- Less programming effort because of inheritance, re-use and extensibility of code.
- Data and methods are bound together.
- Improved communication between users, analysts, designer, and programmers
- Increased consistency in analysis and design

- Explicit representation of commonality among system components
- Reusability of analysis, design, and programming results

Disadvantages of object-oriented model

- Schema change (creating, updating.) can be complex since it involves a system wide recompile.
- Lack of agreed upon standard.
- Lack of universal query language
- Language dependence: tied to a specific language.
- Lacks wide acceptance by industry.

e) Multimedia database

A Multimedia database (MMDB) is a collection of related multimedia data. The multimedia data include one or more primary media data types such as text, images, graphic objects (including drawings, sketches and illustrations) animation sequences, audio and video

f) Hypermedia database

Hypermedia, an extension of the term hypertext, is a nonlinear medium of information which includes graphics, audio, video, plain text and hyperlinks. This contrasts with the broader term multimedia, which may include non-interactive linear presentations as well as hypermedia. The World Wide Web is a classic example of hypermedia, whereas a non-interactive cinema presentation is an example of standard multimedia due to the absence of hyperlinks.

g) Centralized database

A centralized database (sometimes abbreviated CDB) is a database that is located, stored, and maintained in a single location. This location is most often a central computer or database system, for example a desktop or server CPU, or a mainframe computer.

h) Distributed database

A distributed database is a database in which storage devices are not all attached to a common processing unit such as the CPU, controlled by a distributed database management system (together sometimes called a distributed database system). It may be stored in multiple computers, located in the same physical location; or may be dispersed over a network of interconnected computers. Unlike parallel systems, in which the processors are tightly coupled and constitute a single database system, a distributed database system consists of loosely coupled sites that share no physical components.

i) Relational database model

This model is based on mathematical concepts of sets and mappings.

i. Collections of items of the same type
ii. No order
iii. No duplicates

Mappings used in relational model: The mappings can be many-to-many, one-to-one and one-to-many.

In Relation database models a relation is modelled as a table with rows and columns.

Properties of a relation include:

- Values are atomic.
- Each row is unique.
- Column values are of the same kind.
- The sequence of columns is insignificant.
- The sequence of rows is insignificant.
- Each column has a unique name.

Certain fields may be designated as keys, which mean that searches for specific values of that field will use indexing to speed them up.

EmpNo	EmpName	Age
E0023	Dennis	23
E0034	Jacob	23
E0056	Janet	22

DeptNo	EmpNo	DeptName
D011	E0034	Human
D012	E0056	Marketing
D012	E0023	Marketing

Figure 124: Relational database model

In the above model there are two relations Employee relation and department relation. The attribute EmpNo is used to link the two relations.

Advantages of relational database model

- Structural independence is promoted by use of independent tables. Any changes in table structure do not affect data access or application programs.
- Tabular view improves conceptual simplicity, thereby promoting easier database design, implementation, management, and use.
- Ad hoc query capability is based on SQL.
- Widely accepted in many enterprises.

Disadvantages of relational database model

- Requires skills to conceptualize and design.
- Database integrity must be ensured during the design and implementation.
- Takes time and effort to design.
- The RDBMS require hardware and software.

5.2.1.4 Learning Activities

Pick an organization of your choice within the area you reside in. The management of that organization have asked you to provide advice on which database organization best suits the organization's need.

i Advice the organization on the merits of having a database

ii Outline the components that will be required to set up the Database

iii Recommend a database model that fits the organization of choice and give justifications for the chosen model

5.2.1.5 Self-Assessment

i. Discuss the meaning of each of the following terms:

(a) data
(b) database
(c) database management system
(d) application program
(e) data independence
(f) views.

ii. What are the main characteristics of the database approach.

iii. Which are the five components of the DBMS environment and discuss how they relate to each other.

iv. What are the advantages and disadvantages of DBMSs.

5.2.1.6 Tools, Equipment, Supplies and Materials

- Computer
- Database software (Ms Access)
- Printer
- stationery

5.2.1.7 References

Silberschatz A. and et al (2001), Database System Concepts, McGraw-Hill.

Garcia-Molina H. and et al (2000), Database System: The complete Book, Prentice Hall.

Elmasri R. and Navathe S(2000), Fundamentals of Database System, Addison Wesley.

Conolly& Begg(2010), Database System: a practical approach to design, implementation and management, Prentice Hall.

Ullman, and Jennifer Widom (2009). *Database Systems: The Complete Book (2nd edition)* by SBN 978-0131873254, Pearson Prentice Hall,

Jeffrey A. Hoffer, Mary B. Prescott, and Fred R. McFadden(2007). Modern Database Management (8th Ed.). Prentice-Hall.

Roger Jennings(2004) Special Edition Using Microsoft Office Access 2003, Que.

Alan Simpson, Margaret Levine Young, and Alison Barrows: (2003)Access 2003 for DUMMIES, Wiley Publishing.

5.2.1.8 model answers to self-assessment

i. Discuss the meaning of each of the following terms:

(a) data

For end users, this constitutes all the different values connected with the various objects/entities that are of concern to them.

(b) database

A shared collection of logically related data (and a description of this data) designed to meet the information needs of an organization.

(c) database management system

A software system that: enables users to define, create, and maintain the database and provides controlled access to this database.

(d) application program

A computer program that interacts with the database by issuing an appropriate request (typically an SQL statement) to the DBMS.

(e) data independence

This is essentially the separation of underlying file structures from the programs that operate on them, also called program-data independence.

(f) views.

A *virtual table* that does not necessarily exist in the database but is generated by the DBMS from the underlying base tables whenever it's accessed. These present only a subset of the database that is of particular interest to a user. Views can be customized, for example, field names may change, and they also provide a level of security preventing users from seeing certain data.

i. Describe the main characteristics of the database approach.

Focus is now on the data first, and then the applications. The structure of the data is now kept separate from the programs that operate on the data. This is held in the system catalog or data dictionary. Programs can now share data, which is no longer fragmented. There is also a reduction in redundancy, and achievement of program-data independence.

ii. Describe the five components of the DBMS environment and discuss how they relate to each other.

(1) *Hardware*: The computer system(s) that the DBMS and the application programs run on. This can range from a single PC, to a single mainframe, to a network of computers.

(2) *Software*: The DBMS software and the application programs, together with the operating system, including network software if the DBMS is being used over a network.

(3) Data: The data acts as a bridge between the hardware and software components and the human components. As we've already said, the database contains both the operational data and the meta-data (the 'data about data').

(4) Procedures: The instructions and rules that govern the design and use of the database. This may include instructions on how to log on to the DBMS, make backup copies of the database, and how to handle hardware or software failures.

(5) People: This includes the database designers, database administrators (DBAs), application programmers, and the end-users.

iv. Discuss the advantages and disadvantages of DBMSs.

Some advantages of the database approach include control of data redundancy, data consistency, sharing of data, and improved security and integrity. Some disadvantages include complexity, cost, reduced performance, and higher impact of a failure.

5.2.2 Learning Outcome 2: Design basic database

5.2.2.1 Introduction to the learning outcome

This section gives an overview of the various database design concepts focusing on data relationships and database integrity constraints. By the end of it, the learner should be able to competently identify and come up with data relationships, the various database integrity constraints and design a basic Entity-Relationship diagram in database design.

5.2.2.2 Performance Standard

5.2.2.2.1 Database design concepts are identified.

5.2.2.2.2 Appropriate database structures are determined.

5.2.2.2.3 Dsign is implemented.

5.2.2.2.4 Database operations are performed.

5.2.2.3 Information sheet

Database Design Concepts

This is the process of creating a design for a database that will support the enterprise's operations and objectives. Database design is performed in three phases (stages) namely conceptual design, logical design and physical design.

Phase 1: Conceptual design

This is the process of constructing an abstract model of data to be included in the database. This model should be independent of any database management system (DBMS) and other physical considerations.

Steps followed:

a) Identification of entities
b) Identification of relationships
c) Construction of conceptual data model.

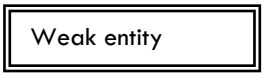
a) Identify entities

- It is a thing or object of importance about which data must be captured.
- To identify entities in the requirements, map nouns to entities
- Database entities appear in a data model as a rectangle with a title.

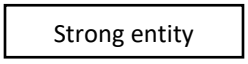
- The title (in upper case) is the name of the entity.

Types of entities

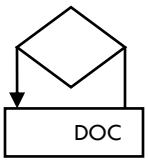
i. **Weak entity:** This is an entity that does not have sufficient attributes to form a primary key (unique identifier). The existence of a weak entity is indicated by a double rectangle in the ER diagram.



i. **Strong entity:** This is an entity that has a primary key (unique identifier). A weak entity is existence dependent.



ii. Recursive Entity: This is an entity in which a relationship can exist between occurrences of the same entity. This occurs in a unary relationship. For instance, a **doctor** can be a **patient.** They are represented as follows:



iii. **Composite Entities:** An entity created to resolve Many-to-Many relationship to one-to-many. It is also referred to as a **bridge entity**. Bridge entity is composed of the primary keys of each of the entities to be connected. A composite entity is represented by a diamond shape within a rectangle in an ER Diagram.

Generalization: The process of defining a more **general entity type** from a set of more specialized entity types. BOTTOM-UP

Specialization: The process of defining one or more subtypes of the supertype and forming supertype/subtype relationships. TOP-DOWN

b) Identification of relationships

A relationship is a meaningful association between entity types.

Classifying relationships:

i. Degree

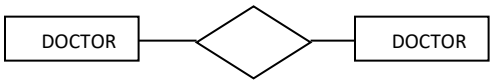
ii. Cardinality

iii. Optionality

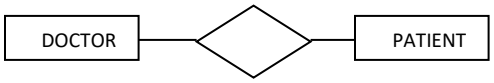
1. Classification according to degree

Degree: The number of **entity types** participating in a relationship.

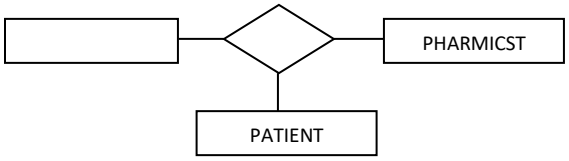
i. Unary (recursive) relationship- One entity type participates in a relationship.



ii. Binary relationship- Two entity types participate in a relationship.



iii. Ternary relationship: Three entity types participate in a relationship.



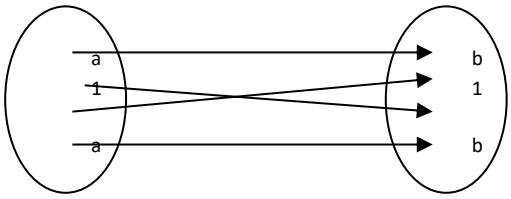
iv. n-array relationship- n entity types participate in a relationship.

2. Classification according to cardinality(connectivity)

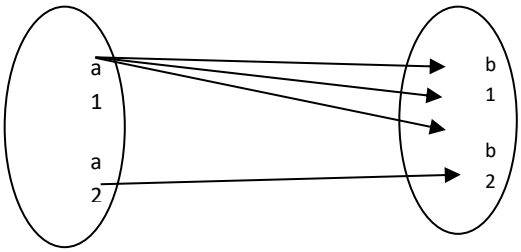
In a relationship, cardinality refers to the number of **entity occurrences** involved in the relationship. Also defined as minimum and maximum number of entity occurrences that can participate in a relationship.

Examples

i). One-to-one relationship (1:1): Each entity occurrence in entity type A can only have one matching entity occurrence in entity type B and vice versa.

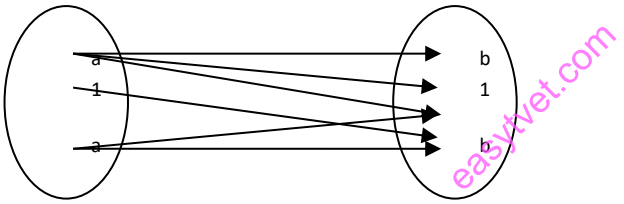


ii). One-to-many relationship(1:m): An entity occurrence in entity type A can have several matching entity occurrences in entity type B, but each entity occurrence in entity type B can only have a maximum of one entity occurrences in entity type A.



iii). Many-to-many relationship(m:n)

An entity occurrence in entity type A can have several matching entity occurrences in entity type B and vice versa.



3. Classification according to optionality (participation)

Optionality refers to the minimum number of entities in a relationship. Can be **mandatory** or **optional.**

— Examples

- Mandatory: There **must be** an entity occurrence participating in the relationship
- Optionality: There **may be** an entity occurrence participating in the relationship. To show optionality, put a circle or `0' at the `optional end' of the relationship.

c) Building conceptual data model

The data model that results after conceptual design is called conceptual data model.

Choose a notation and design the data model. Examples of notations are Chen's notation, Rein85 notation, Crow's foot notation and IDEF1X notations. Most of the differences concern how

relationships are specified and how attributes are shown. In almost all variations, entities are depicted as rectangles with either pointed or rounded corners. The entity name appears inside. Relationships can be displayed as diamonds or can be simply line segments between two entities. For Relationships, there is need to convey relationship name, degree, cardinality and optionality where necessary. Any notation selected must be used consistently.

Table 34: Comparison of ERD notations

Chen's notation	**Crow's foot**	**Rein85**	**IDEF1X**
Relationship Name: Displayed just inside the relationship diamond. **Cardinality:** The cardinality is indicated by placing the appropriate numbers beside the entities using the format (x,y), which represent the minimum and maximum value, respectively. **Optionality:** Mandatory participation indicated by one/ two line(s) linking the entity to the diamond next to mandatory end. Optional participation indicated by a circle near optional end.	**Relationship name:** Relationships name indicated next to entity above or below the link. **Cardinality:** Crow's feet are used to show a many side of a relationship. A single line indicates a one side of the relationship. **Optionality:** Optional participation is shown with an open circle next to optional end. Mandatory participation is shown with two vertical lines.	**Relationship name:** Relationships name indicated next to relationship diamond. **Cardinality:** Not shown except the connectivity. **Optionality:** Optional participation is shown with an open circle next to optional end. Mandatory participation is shown with continuous link line next to mandatory end.	**Relationship name:** Relationships name indicated next to relationship diamond. **Cardinality:** Not shown except the connectivity. **Optionality:** Optional participation is shown with an open circle next to optional end. Mandatory participation is shown with continuous link line next to mandatory end.

Comparison of ERD notations.

a) **Chen's notation**

Chen model moved conceptual modelling into the practical database design arena by establishing basic building blocks: entities and relationships. Dominant player in the CASE tool market during the 1980s and early 1990s.

b) **Crow's Foot notation**

This model combines connectivity and cardinality information in a single symbol set. Popularized by the Knowledge ware modeling tool. Cardinality is limited to 0,1 or N.

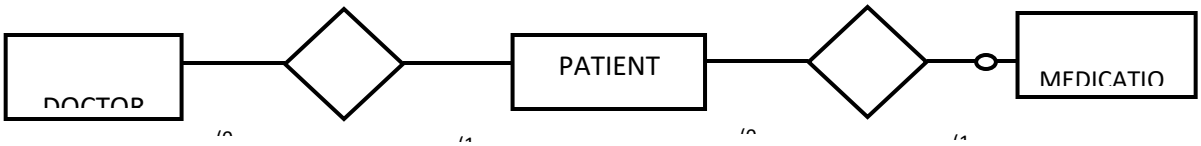
c) **Rein85 notation**

This model is based on the same modelling conventions as the Crow's Foot model, its symbols are different. It does not recognize cardinalities explicitly, relying on connectivity to lead to logical cardinality conclusions.

d) **IDEF1X notation**

This is a derivative of the integrated computer-aided manufacturing (ICAM) studies of the late 1970s. Became the source of graphical methods for defining the functions, data structures and dynamics of manufacturing businesses. The integration of these methods became known as IDEF (ICAM Definition). Hughes Aircraft developed the original version named IDEF1. Later, the extended version was named IDEF1X.

Example 1: Chen's notation



The above model will be read as follows:

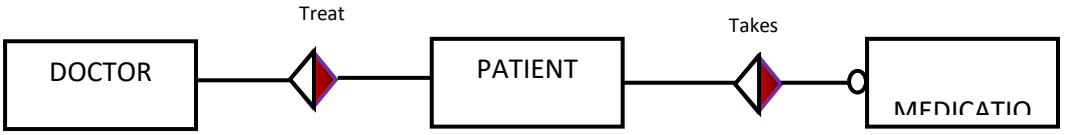
Each DOCTOR must treat one or more PATIENTS. Each patient must be treated by a doctor. Each PATIENT may take one or more MEDICATIONS. Each MEDICATION is given to one PATIENT.

Example1: Crow's foot notation.

The above model will be read as follows:

Each DOCTOR must treat one or more PATIENTS. Each patient must be treated by a doctor. Each PATIENT may take one or more MEDICATIONS. Each MEDICATION is given to one PATIENT.

Example1: Rein85



The above model will be read as follows:

Each DOCTOR must treat one or more PATIENTS. Each patient must be treated by a doctor. Each PATIENT may take one or more MEDICATIONS. Each MEDICATION is given to one PATIENT.

Example 1: IDEF1X

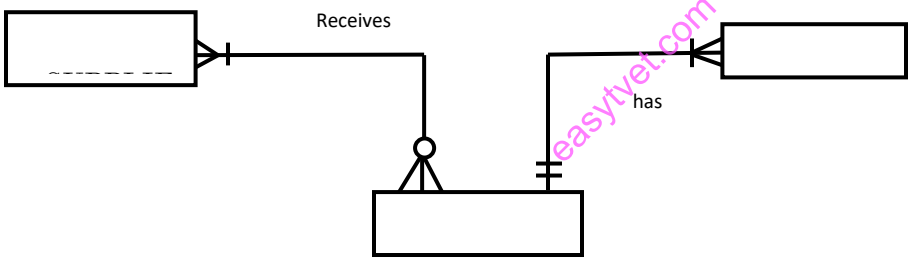
The above model will be read as follows:

Each DOCTOR must treat one or more PATIENTS. Each patient must be treated by a doctor. Each PATIENT may take one or more MEDICATIONS. Each MEDICATION is given to one PATIENT.

The features of the conceptual data model include:

i. Important entities and the relationships among them.
ii. No attribute is specified.
iii. No primary key is specified.

Example: A supplier receives orders for products. A SUPPLIER may receive one or more ORDERS. Each one or more ORDERS must be sent to the SUPPLIER. Each order consists of one or several products. But each PRODUCT is belonging to one ORDER.



— From the figure above, the conceptual data model is the entities that describe the data and the relationships between those entities.

Phase 2. Logical database design

The process of constructing a database based on specific database management system (DBMS) and database model. For example, a relational Database model. It considers how the data are represented using the structures offered by the DBMS:

Steps followed:

- Select database model.
- Map Entity-Relationship Diagrams (from conceptual data model into logical model)
- Identify the attributes.
- Normalize database.
- Design the ERD (logical data model) and validate with the users.
- Create a data dictionary

i). Selecting database model

A database model is a collection of **concepts** and **rules** for the description of the structure of the database. Some of the models one can choose are flat file database, hierarchical, network, relational and object oriented.

ii). Map Entity-Relationship Diagrams

-Identify all the entities and relationships as captured in conceptual data model.

iii). Identify attributes.

— **Attributes are explicit values such as a** quality **or** characteristic **associated with someone or something** (entity).

— **They represent** characteristics **or** values **about a database** relation **(an entity)**

Types of attributes

i. **Key attributes:** Attributes which uniquely identify an instance of an entity (identifier). These attributes are distinct for each individual entity. For example, a student Number or Car registration number etc.

ii. **Non key attributes:** These are attributes which describe a non-unique characteristic of an entity instance (descriptors). Examples year, colour, model.

iii. **Required attributes**: These are attributes which must have a value, or a value must be known for each entity occurrence. Example **employee number.**

iv. **Optional attributes**: These are attributes which could have a value, or a value may be known for each entity occurrence. Example an employee; **employees' spouse** is optional since not every employee will have a spouse.

v. **Composite attribute**: This is an attribute which can be divided into **smaller subparts**. These subparts represent basic attributes with **independent meanings** of their own. For example, take Name attributes. We can divide it into sub-parts like First name, Middle name, and Last name.

vi. **Simple attribute**: This is an attribute that cannot be divided into subparts. It is therefore said to be **simple** or **atomic attribute**. For example, **Employee Number** is a simple attribute. **Age** of a person is a simple attribute.

vii. **Single-valued attribute**: It is an attribute that can have single value at a particular instance of time. For instance, a person cannot have more than one age value.

viii. **Multi-valued attribute:** It is an attribute that can have **more than one value** at one time. For instance, a person can have more than one degree at one time, therefore attribute degree is said to be a multi-valued attribute.

ix. **Stored attribute**: An attribute that supplies a value to the related attribute.

x. **Derived attribute**: An attribute whose value is derived from a stored attribute. These attributes are usually created by a formula or certain operation on other attributes. Example: **Age= [Current-year]- [year_of_birth]**

xi. **Candidate Key**: An attribute or set of attributes that uniquely identifies individual occurrences of an entity type.

xii. **Primary Key:** A unique attribute that is used to identify an entity type e.g., entity type student can be identified by student number.

xiii. **Composite Key:** A candidate key that consists of two or more attributes. e.g., VoterID+NationalID+PassportID

iv). Normalization of the database

Normalization or database normalization is a process used to organize the data into tabular format (database tables). A good database design includes the normalization. Without normalization a database system may slow, inefficient and might not produce the expected result. Normalization reduces the data redundancy and inconsistent data dependency. We organize the data into database tables by using normal forms rules or conditions.

Normalization the process of removing redundant data from the tables to improve storage efficiency, data integrity and scalability. Normalization works through series of steps called normal forms.

Types of data dependencies

Functional dependency

Implies that if A and B are attributes (columns)of a table, B is fully functionally dependent on An if value of A determines the value of B. For example, voter ID, voter name.

Partial Functional Dependency

Implies that if A and B are attributes of a table, B is partially dependent on A if there is some attribute that can be removed from A and yet the dependency still holds. Consider a case of voter ID, voter national ID, voter name, constituency all being attributes in a table. The voter ID value can only determine the value of constituency (Partial dependency).

Transitive Functional Dependency

A condition where if A, B and C are attributes of a table and further that A is the only key attribute then if value of A determines value of B and value of B determines value of C, then we say C is transitively dependent on A via B. This means a non-key attribute(C) is depending on another non key attribute(B). For example, If Vote ID, Constituency, Poll station are all attributes of a table voter. Then Poll station depends on constituency which in turn depends on Vote ID. Poll station is therefore functionally dependent on Vote ID via constituency. This is a transitive dependency.

Reasons for normalization

- To make it feasible to represent any relation in the database.
- To increase speed and flexibility of queries (easier to search).
- To free relations from anomalies.

- To reduce the need of restructuring the relations whenever new data types are introduced(scalability)
- To avoid repetitive entries (data redundancy).

Data anomalies

Relations that have redundant data may have problems called anomalies, which are classified as.

i. Insertion anomalies
ii. Deletion anomalies
iii. Update anomalies

a. Update anomaly

The same information appears on multiple rows; therefore, updates to the relation may result in logical inconsistencies. If the update is not carried through successfully to all the rows, then the relation is left in an inconsistent state. For example if sender No T3 was actually supposed to be ”Timothy Makua” then the user must update all the records where T3 appears so that the table is consistent.

Example

Table 35: Table with anomalies

	Haraka Courier services			
SenderNo	**Sender**	**Description**	**Date**	**Recipient**
T1	Munyalo Mutua	Magazine	11/9/2012	Kasyoka Munyalo
T2	Ezekiel Oduor	Magazine	11/9/2012	Gerald Oduor
T3	Timothy Wafula	Magazine	12/9/2012	Janet Wekesa
T3	Timothy Wafula	Laptop	14/9/2012	Ahmed Amin

T1	Munyalo Mutua	Magazine	11/10/2012	Kasyoka Munyalo
T4	Fred Kariuki	Letter	13/11/2012	Tom Maina
T3	Timothy Wafula	Laptop	14/11/2012	Ahmed Amin
T1	Munyalo Mutua	Magazine	11/12/2012	Kasyoka Munyalo
T3	Timothy Wafula	Laptop	14/12/2012	Ahmed Amin
T1	Munyalo Mutua	Magazine	17/12/2012	Kasyoka Munyalo

b. Insertion anomaly

This anomaly causes an inability to add information to the database due to a lack of some related value. For example, to insert details of a new sender say ”Richard Muriuki” the user must enter sender No value else the system will not allow those details into the system.

c. Delete anomaly.

This may occur when the deletion of data representing certain facts results into deletion of data representing completely different data. For example if the user deletes third row containing sender No, then details containing sender, description, date and recipient in that row will be lost as well.

Normalization steps

Data is organized into database tables by using normal forms rules or conditions. These rules help to make a good database design. Generally, we organize the data up to third normal form. We rarely use the fourth and fifth normal forms.

- Begin with a list of all the fields that must appear in the database. Think of this as one big table.
- Do not include computed fields.
- You can get this information from a printed document used by the system e.g., reports.
- Additional attributes besides those for the entities described on the document can be added to the database.

a. First Normal (FNF):

It is a relation in which the intersection of each row and column contains one and only one value (atomic values). No repeating groups.

Steps

- Remove repeating groups.
- Create separate tables for each group of related data and identify each row with a primary key.

b. Second Normal Form (SNF)

It is a relation that is in first normal form and where every non-primary-key attribute is fully functionally dependent on the primary key. No partial dependencies.

Steps:

- Ensure that the table is in the first normal form.
- Remove partial dependencies and put the partially dependent attributes in a separate table.

c. Third Normal Form (TNF)

A relation that is in first and second normal form, and in which no non-primary-key attribute is transitively dependent on the primary key.

Steps:

- Ensure that the table is in the second normal form.
- Remove transitive dependencies and create relationship between newly created table with their predecessors using foreign keys.

Example:

Consider the report below about customer rental information. Normalize this report to third normal form.

Report

A report

Page 1 | **DreamHome Customer Rental Details** | Date: 19-02-2013

Customer Number: CU034 Customer Name John Mwendwa

Property Number	Property Address	Rent start	Rent finish	Rent	Owner Number	Owner Name
PG4	Box 123 NRB	1-02-2010	1-02-2012	120000	CO123	Mary mutia
PG16	BOX 12 THIKA	2-02-2009	2-02-2013	200000	C096	Mariana Ouma

Un-normalized table

Figure 125: Sample report to be normalized.

Solution:

Begin by listing all the fields that must appear in the database. This will result into a big table. Notice the report title, page number and its date are not necessary since they do not necessary reside in the database. This produces an unnormalized form.

Customer-No	Customer-Name	Property-No	Property-address	Rent-start	Rent-Finish	Rent	Owner-No	Owner-Name
Cu034	John Mwendwa	PG4	BOX 123 NRB	1-02-2010	1-02-2012	120000	C0123	Mary Mutia
Cu034	John Mwendwa	PG16	Box 12 Thika	2-02-2009	2-02-2013	200000	C096	Mariana Ouma

Table 36: un normalized table

Un-normalized form

- Separate repeating groups into new Tables.
- Repeating groups are fields that may be repeated several times for one document/entity.
- Create a new table containing the repeating data.
- The primary key of the new table (repeating group) is always a composite key; Usually document number and a field uniquely describing the repeating line, like an item number.

  In this case Customer Name is repeated. But in this case, there is no document number e.g., receipt number.

First normal form

Customer-No	Property-No	Property-address	Rent-start	Rent-Finish	Rent	Owner-No	Owner-Name
Cu034	PG4	BOX 123 NRB	1-02-2010	1-02-2012	120000	C0123	Mary Mutia
Cu034	PG16	Box 12 Thika	2-02-2009	2-02-2013	200000	C096	Mariana Ouma

PROPERTY

Property-No	Customer-No	Property-address	Rent-start	Rent-Finish	Rent	Owner-
PG4	Cu034	BOX 123 NRB	1-02-2010	1-02-2012	120000	C0123
PG16	Cu034	Box 12 Thika	2-02-2009	2-02-2013	200000	C096

CUSTOMER

Customer-No	Customer-Name
Cu034	John Mwendwa

OWNER

Owner-No	Owner-Name
C0123	Mary Mutia
C096	Mariana Ouma

Second normal form

It's carried out where the table has composite key. All partially dependent attributes are removed and placed in a new table. A copy of the primary key is used to identify the attributes in this new table. In this case there are three key attributes in the original table(Property-No,Customer-No,Owner-No). OwnerName only depends on OwnerNo(Partial dependency). This attribute is removed together with a copy of the key Owner-No.

PROPERTY						
Customer-No	PropertyNo	OwnerNo	Rent No	Property-address	Rent-start	Rent-Fi
Cu034	PG4	C0123	P1	BOX 123 NRB	1-02-2010	1-02-201
Cu034	PG16	C096	P2	Box 12 Thika	2-02-2009	2-02-201

OWNER	
Owner-No	Owner-Name
C0123	Mary Mutia
C096	Mariana Ouma

CUSTOMER	
Customer-No	PropertyName
Cu034	John Mwendwa
Cu034	John Mwendwa

RENT	
Rent-No	Rent
P1	120000
P2	200000

Third normal form

In this step, remove transitive dependencies. Transitive dependency is a type of functional dependency where an attribute is functionally dependent on an attribute other than the primary key. Thus, its value is only indirectly determined by the primary key.

- Create a separate table containing the attribute and the fields that are functionally dependent on it. Keep a copy of the key attribute in the original table.
- In this case rent (Amount) is transitively dependent on the rent start and rent finish.
- Remove this attribute create a primary to identify its value e.g., payment No. The copy of this key must remain in the original table.

Database Integrity.

The main types of integrity constraints that exist for a database include the following:

i. **Domain integrity**: This is restricting what users can enter the field. It is the validity of entries for a given column. It can be enforced by restricting the type (through data types), check constraints and rules can be used to validate the data that users try to enter against a list of acceptable data and defaults can be used to enter data for the users if they forget.

ii. **Entity integrity**: This is the process of making sure that each record in a table is unique in some way. Primary keys are used to enforce entity integrity (Gunderloy & L.Jorden, 2001).

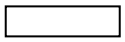
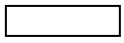
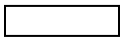
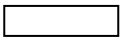
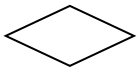
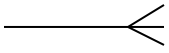
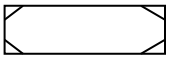
iii. **Referential integrity**: This is the process of protecting related data that is stored in some separate tables.

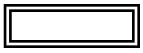
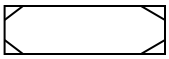
iv. **User defined Integrity:** User-defined integrity allows users to define specific business rules. Example Age>=18.

v). Design the ERD (logical data model) and validate with the users.

In this step we use the normalized forms and standard notations to draw the ERD. Such notations include Rein95, IDEFl1X, Chen, Crow's foot among others.

Table 37: ERD elements notations

	Chen	**Crow's Foot**	**Rein85**	**IDEFI1X**
Entity				
Relationship line				
Relationship				
Optional symbol				
One(1) symbol	1			
Many(M) symbol	M			
Composite entity				

Weak entity				

A logical data model describes the data in as much detail as possible, without regard to how they will be physically implemented in the database. A logical data model is used to explore the domain concepts, and their relationships, of the problem. This could be done for the scope of a single project or for the entire enterprise. The logical data model depicts the logical entity types, typically referred to simply as entity types, the data attributes describing those entities, and the relationships between the entities.

Features of a logical data model include:

- Includes all entities and relationships among them.
- All attributes for each entity are specified.
- The primary key for each entity is specified.
- Foreign keys (keys identifying the relationship between different entities) are specified.
- Normalization occurs at this level

Logical data model versus Conceptual data model

In a logical data model, primary keys are present, whereas in a conceptual data model, no primary key is present.

In a logical data model, all attributes are specified within an entity. No attributes are specified in a conceptual data model.

Relationships between entities are specified using primary keys and foreign keys in a logical data model. In a conceptual data model, the relationships are simply stated, not specified, so we simply know that two entities are related, but we do not specify what attributes are used for this relationship.

Phase 3: Physical database design

This is a process of constructing a model of information used in an organization based on physical considerations like software and hardware. Physical design is concerned on how the

database is organized as files and what kind of structures to use for efficiency of database processing.

Steps:

- Select DBMS.
- Select storage devices.
- Determine access methods.
- Design files and indexes
- Determine database distribution.
- Produce the physical model and validate with the users

i). Select DBMS.

Factors considered when selecting the DBMS.

- Costs
- Features and Tools
- Underlying model
- Portability
- DBMS hardware requirements
- Security
- Usability

Main steps to selecting a DBMS:

- Define terms of reference of study.
- Shortlist two or three products
- Evaluate products.
- Recommend selection and produce report.

Examples of DBMS

- Microsoft access
- Microsoft SQL server
- Oracle
- Informix
- MySQL

Functions of DMBS

- Database creation (storing data in a defined database) using data definition language.
- Retrieval (query and reporting) using data manipulation language.
- Update (Changing the contents of the database) using data manipulation language.
- Programming user facilities for system development.
- Database revision and restructuring using data manipulation language.
- Database integrity control using data definition language.
- Performance Monitoring
- Concurrency control; controlling simultaneous access to the database.

ii) Select storage devices

When selecting storage devices consider:

- Speed with which data can be accessed.
- Cost per unit of data

Reliability: loss on power failure or system crash and physical failure of the storage device

Media

- **Cache**: Fastest and most costly form of storage; volatile; managed by the computer system hardware.
- **Main memory**: Fast access but generally too small (or too expensive) to store the entire database.
- **Flash memory:** Data survives power failure. Data can be written at a location only once, but location can be erased and written to again. Reads are roughly as fast as main memory. But writes are slow (few microseconds), erase is slower.
- **Magnetic disk:** Data is stored on spinning disk and read/written magnetically. It is a primary medium for the long-term storage of data; typically stores entire database. Data must be moved from disk to main memory for access and written back for storage. It is possible to read data on disk in any order, unlike magnetic tape. Survives power failures and system crashes. However, disk failure can destroy data.

- **Optical storage:** This is a non-volatile, data is read optically from a spinning disk using a laser. Reads and writes are slower than with magnetic disk.
- **Tape storage**: Is a non-volatile storage, used primarily for backup (to recover from disk failure), and for archival data. Allows sequential-access and thus much slower than disk. Very high capacity.

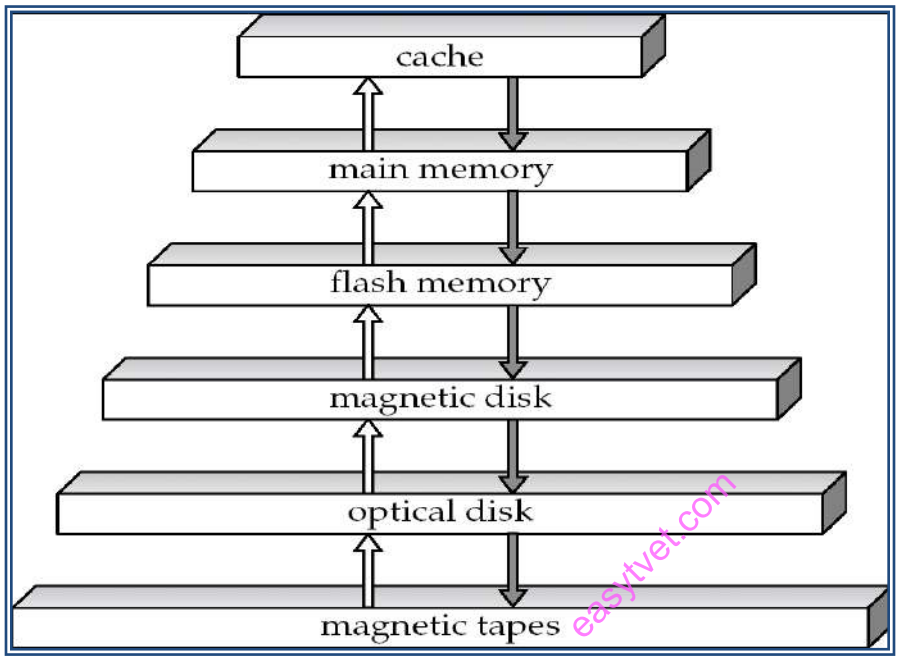


Figure 126: Memory hierarchy

iii). Determine access methods.

In this step identify specific structures and access methods for the data to achieve optimum performance for the database system. The access method depends on storage structure. Some of the access methods are:

- Sequential
- Indexed sequential.
- Hashing

iv) Design files and indexes

Creating set of tables and constraints on these tables from the information given in logical data model. The tables are designed, and their field properties specified.

FieldName	Data type	Length
SupplierID	Number	8
SupplierName	Text	50
Address	Text	50
Street	Text	50

Table 38 :Supplier table

FieldName	Data type	Length
OrderID	Text	50
OrderName	Text	50
Dateprepared	Date/Time	
SupplierID	Number	8

Table 39: Orders table

FieldName	Data type	Length
ProductID	Text	30
ProductName	Text	60

Quantity	Number	8
OrderID	Text	50

Table 40: Products table

v). Determine database distribution.

There is need to consider location and number of users when deciding the database distribution. For instance, the database may single user database in which case the database can be stored in a local computer, multiuser database may store in a dedicated server where users can access simultaneously. If the users are not within the same location and the organization has several branches in different locations, then a distributed database may be used.

Physical data model

Physical data model represents how the model will be built in the database.

A physical database model shows all table structures, including column name, column data type, column constraints, primary key, foreign key, and relationships between tables.

Features of a physical data model include:

i. Specification of all tables and columns.
ii. Foreign keys are used to identify relationships between tables.
iii. De-normalization may occur based on user requirements.
iv. Physical considerations may cause the physical data model to be quite different from the logical data model.
v. Physical data model will be different for different RDBMS. For example, data type for a column may be different between MySQL and SQL Server.

The steps for physical data model design are as follows:

i. Convert entities into tables.
ii. Convert relationships into foreign keys.
iii. Convert attributes into columns.
iv. Modify the physical data model based on physical constraints / requirements.

Logical data model versus Physical data model

Comparing the logical data model shown above with the logical data model diagram, we see the main differences between the two:

i. Entity names are now table names.
ii. Attributes are now column names.
iii. Data type for each column is specified. Data types can be different depending on the actual database being used.

Database Implementation

This is the physical realisation of the database and application designs. The detailed model is converted to the appropriate implementation model, **the data dictionary is built**, the database is populated, application programs are developed, and users are trained.

Database implementation activities

i. Hardware/Software Acquisition if needed.
ii. Coding
iii. Testing (program, subsystem, system tests)
iv. Training (lead users, train the trainer)
v. Conversion and loading (from old to new systems)

i. Hardware/Software Acquisition

It is the selection, purchase and, if applicable, implementation of technology-related products and services. Consultation with the ICT department and purchasing early in the process will help avoid delays in the database development. The product chosen must meet user and technical requirements. The process may involve the following:

- Implementation of a new or upgraded multi-user system.
- Solutions requiring an interface to an existing system, such as Banner
- Contracting with a third-party service for software
- Computer accessories, peripherals, and supplies
- DVDs, CDs and videotapes
- Printers and toner cartridges
- Backup tapes

ii. Coding

This is the process of translating application design (algorithms) into executable statements. The tools for designing a program are top-down charts, flowcharts, decision tables and Pseudo-code. The commonly used tools however are flow charts and pseudo codes. The flowchart provides a pictorial representation of the application design while the pseudo code is a non-formal language that provides a way to create a logical structure, describing the actions, which will be executed by the application. Coding is done using a suitable high level programming language and following the application design. When coding, follow the following best practices.

- Know what the code block must perform.
- Indicate a brief description of what a variable is for (reference to commenting)
- Correct errors as they occur.
- Keep your code simple.
- Maintain naming conventions which are uniform throughout.

iii. Testing

It is the process of evaluating a system or its component(s) with the intent to find that whether it satisfies the specified requirements or not. Testing results in the actual expected and difference between their results. It is used to identify any gaps, errors, or missing requirements in contrary to the actual desire or requirements.

iv. Training of users

Training involves creating training documentation and end-user training. Determine how end users will be instructed on the new product? This can in classrooms, on the job. It can be done by individuals or done through computer e.g., use of videos. The documentation can be distributed to the end user describing how to use the product (i.e.: help files, training manual, user manual). Most importantly the focus should be to train people about what they need to know. It may therefore differ for each job.

v. Conversion and loading (from old to new system)

The process of transferring any existing data into new database and converting any existing applications to run on the new system. This is only required if a new database will be replacing an existing database. Modern database management systems (DBMS) have utilities for loading existing files into new database. Such utilities require the specification of the source file and the target database and then automatically converts the data to the required format of the new database file

Database Operations

a) SELECT operation

The SELECT statement is probably the most used SQL command. The SELECT statement is used for retrieving rows from the database and enables the selection of one or many rows or columns from one or many tables in the database.
In the simplest form we can use the SELECT statement as follows:

select <column names> from <table names>

If we want all columns, we use the symbol "*"
Note! SQL is not case sensitive. SELECT is the same as select.
The full syntax of the SELECT statement is complex, but the main clauses can be summarized.
as:
SELECT [ALL | DISTINCT]
[TOP (expression) [PERCENT] [WITH TIES]] select list [INTO new table] [FROM table source] [WHERE search condition] [GROUP BY _expression] HAVING search condition] ORDER BY order expression [ASC | DESC]]

example:

select * from CUSTOMER.

This simple example gets all the data in the table CUSTOMER. The symbol "*" is used when you want to get all the columns in the table.

If you only want a few columns, you may specify the names of the columns you want to retrieve,

example:

select CustomerId, Last Name, FirstName from CUSTOMER

b) INSERT operation

The INSERT INTO statement is used to insert a new row in a table. It is possible to write the INSERT INTO statement in two forms.
The first form does not specify the column names where the data will be inserted, only their values:

INSERT INTO table name.
VALUES (value1, value2, value3,)
Example:
INSERT INTO CUSTOMER VALUES ('1000', 'Kaparo', 'John', 12,
'Nairobi', '11111111')

The second form specifies both the column names and the values to be inserted:

INSERT INTO table name (column1, column2, column3...)
VALUES (value1, value2, value3...)

Note: This form is recommended!

Example:

INSERT INTO CUSTOMER (Customer Number, Last Name, FirstName, Area Code, Address, Phone)
VALUES ('1000', 'Kaparo', 'John', 12, 'Nairobi', '11111111')

Insert Data Only in Specified Columns:

It is also possible to only add data in specific columns.

Example:

INSERT INTO CUSTOMER (Customer Number, Last Name, FirstName)
VALUES ('1000', 'kaparo', 'John')

Note! You need at least to include all columns that cannot be NULL.

c) UPDATE operation

The UPDATE statement is used to update existing records in a table. The syntax is as follows:

UPDATE table name
SET column1=value, column2=value2, ...
WHERE some column=some value

Note! Notice the WHERE clause in the UPDATE syntax. The WHERE clause specifies which record or records that should be updated. If you omit the WHERE clause, all records will be updated!

Example:

update CUSTOMER set Area Code=46 where CustomerId=2

d) DELETE operation.

The DELETE statement is used to delete rows in a table.

The Syntax is shown below:

DELETE FROM table name.
WHERE some column=some value

Note! Notice the WHERE clause in the DELETE syntax. The WHERE clause specifies which record or records that should be deleted. If you omit the WHERE clause, all records will be deleted!

Example:

delete from CUSTOMER where CustomerId=2

Delete All Rows:

It is possible to delete all rows in a table without deleting the table. This means that the table structure, attributes, and indexes will be intact:

DELETE FROM table name.

Note! Make sure to do this only when you really mean it! You cannot UNDO this statement!

3.2.4.3 Learning Activities

The report below is how an inexperienced database developer might create a table. Required: -

i. Normalize it upto 3rd Normal Form.

ii. Prepare an ERD to show the relationships

Figure 127: Sample report to be normalized

Maina Music stores

Email: mainsh@main.co.ke

Movie issuance report

Customer ID	Last Name	Movie ID	Movie Title	Vendor	Type	Check Out Date	Return Date
1001	Barns	101	Movie 1	ACM	ACT	1/1/2002	1/2/2002
1001	Barns	102	Movie 2	ACM	COM	1/1/2002	1/2/2002
1001	Barns	103	Movie 3	ACM	DRA	1/1/2002	1/5/2002
1001	Barns	104	Movie 4	ACM	DRA	1/1/2002	1/6/2002
1001	Barns	105	Movie 5	ACM	DRA	1/1/2002	
1001	Barns	106	Movie6	BB	DRA	1/1/2002	
1001	Barns	107	Movie7	BB	COM	1/1/2002	
1001	Barns	108	Movie8	ACM	COM	1/1/2002	
1001	Barns	109	Movie9	ACM	COM		

5.2.3.4 Self-Assessment

i. Define the two principal integrity rules for the relational model. Discuss why it is desirable to enforce these rules.

ii. Briefly describe the stages of the database system development lifecycle.

iii. Describe what relationships represent in an ER model and provide examples of unary, binary, and ternary relationships.

iv. Describe what attributes represent in an ER model and provide examples of simple, composite, single-value, multi-value, and derived attributes.

5.2.2.5 Tools, Equipment, Supplies and Materials

- Computer
- Database software (Ms Access)
- Printer
- stationery

5.2.2.6 References

Silberschatz A. and et al (2001), Database System Concepts, McGraw-Hill.

Garcia-Molina H. and et al (2000), Database System: The complete Book, Prentice Hall.

Elmasri R. and Navathe S(2000), Fundamentals of Database System, Addison Wesley.

Conolly& Begg(2010), Database System: a practical approach to design, implementation and management, Prentice Hall.

Ullman, and Jennifer Widom (2009). *Database Systems: The Complete Book (2nd edition)* by SBN 978-0131873254, Pearson Prentice Hall,

Jeffrey A. Hoffer, Mary B. Prescott, and Fred R. McFadden(2007). Modern Database Management (8th Ed.). Prentice-Hall.

Roger Jennings(2004) Special Edition Using Microsoft Office Access 2003, Que.

Alan Simpson, Margaret Levine Young, and Alison Barrows(2003) Access 2003 for DUMMIES, Wiley Publishing.

5.2.2.7 Model answers to self-assessment

i. Define the two principal integrity rules for the relational model. Discuss why it is desirable to enforce these rules.

Entity integrity In a base table, no column of a primary key can be null.

Referential integrity If a foreign key exists in a table, either the foreign key value must match a candidate key value of some record in its home table or the foreign key value must be wholly null.

ii. Briefly describe the stages of the database system development lifecycle.

Stages of the database system development lifecycle.

Database planning is the management activities that allow the stages of the database system development lifecycle to be realized as efficiently and effectively as possible.

System definition involves identifying the scope and boundaries of the database system including its major user views. A user view can represent a job role or business application area.

Requirements collection and analysis is the process of collecting and analysing information about the company that is to be supported by the database system, and using this information to identify the requirements for the new system.

There are three approaches to dealing with multiple user views, namely the centralized approach, the view integration approach, and a combination of both. The **centralized approach** involves collating the users' requirements for different user views into a single list of requirements. A data model representing all the user views is created during the database design stage. The **view integration approach** involves leaving the users' requirements for each user view as separate lists of requirements. Data models representing each user view are created and then merged at a later stage of database design.

Database design is the process of creating a design that will support the company's mission statement and mission objectives for the required database. This stage includes the logical and physical design of the database.

The aim of **DBMS selection** is to select a system that meets the current and future requirements of the company, balanced against costs that include the purchase of the DBMS product and any additional software/hardware, and the costs associated with changeover and training.

Application design involves designing the user interface and the application programs that use and process the database. This stage involves two main activities: transaction design and user interface design.

Prototyping involves building a working model of the database system, which allows the designers or users to visualize and evaluate the system.

Implementation is the physical realization of the database and application designs.

Data conversion and loading involves transferring any existing data into the new database and converting any existing applications to run on the new database.

Testing is the process of running the database system with the intent of finding errors.

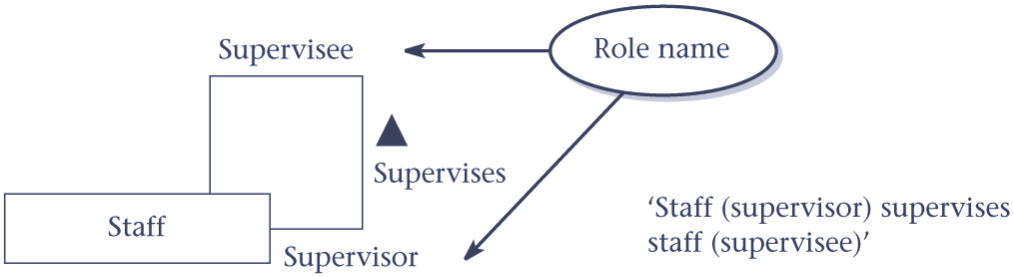
Operational maintenance is the process of monitoring and maintaining the system following installation.

2. Describe what relationships represent in an ER model and provide examples of unary, binary, and ternary relationships.

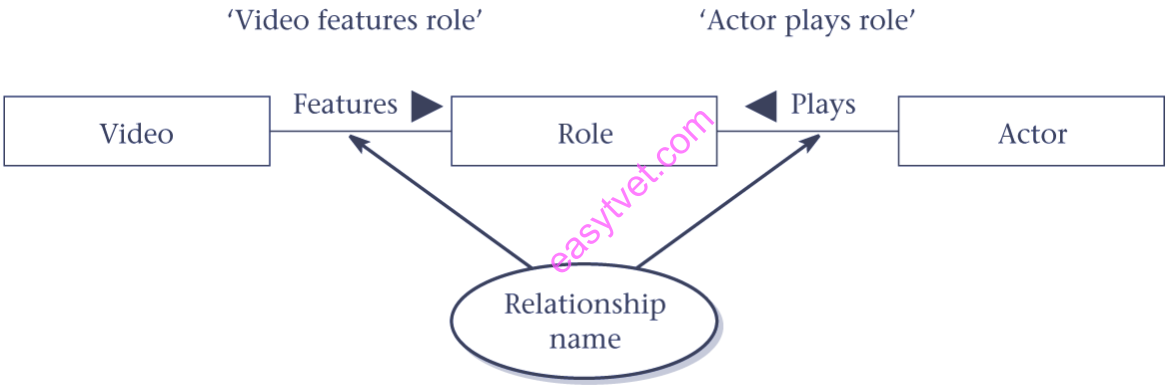
Relationship is a set of meaningful associations among entities. As with entities, each association should be uniquely identifiable within the set. A uniquely identifiable association is called a relationship occurrence. Each relationship is given a name that describes its function. For example, the Actor entity is associated with the Role entity through a relationship called *Plays*, and the Role entity is associated with the Video entity through a relationship called *Features*.

The entities involved in a particular relationship are referred to as participants. The number of participants in a relationship is called the degree and indicates the number of entities involved in a relationship. A relationship of degree one is called **unary**, which is commonly referred to as a *recursive* relationship. A unary relationship describes a relationship where the *same* entity participates more than once in *different* roles. An example of a unary relationship is *Supervises*,

which represents an association of staff with a supervisor where the supervisor is also a member of staff. In other words, the Staff entity participates twice in the *Supervises* relationship; the first participation as a supervisor, and the second participation as a member of staff who is supervised (supervisee). See Figure 7.5 for a diagrammatic representation of the *Supervises* relationship.



A relationship of degree two is called **binary**.



A relationship of a degree higher than binary is called a complex relationship. A relationship of degree three is called **ternary**. An example of a ternary relationship is *Registers* with three participating entities, namely Branch, Staff, and Member. The purpose of this relationship is to represent the situation where a member of staff registers a member at a particular branch, allowing for members to register at more than one branch, and members of staff to move between branches.

3. Describe what attributes represent in an ER model and provide examples of simple, composite, single-value, multi-value, and derived attributes.

An **attribute** is a property of an entity or a relationship.

Attributes represent what we want to know about entities. For example, a Video entity may be described by the catalog no title, category, daily Rental, and price attributes. These attributes hold values that describe each video occurrence and represent the main source of data stored in the database.

Simple attribute is an attribute composed of a single component. Simple attributes cannot be further subdivided. Examples of simple attributes include the category and price attributes for a video.

Composite attribute is an attribute composed of multiple components. Composite attributes can be further divided to yield smaller components with an independent existence. For example, the name attribute of the Member entity with the value 'Don Nelson' can be subdivided into Name ('Don') and Name ('Nelson').

Single-valued attribute is an attribute that holds a single value for an entity occurrence. Most attributes are single valued for a particular entity. For example, each occurrence of the Video entity has a single-value for the catalog no attribute (for example, 207132), and therefore the catalogNo attribute is referred to as being single-valued.

Multi-valued attribute is an attribute that holds multiple values for an entity occurrence. Some attributes have multiple values for a particular entity. For example, each occurrence of the Video entity may have multiple values for the category attribute (for example, 'Children' and 'Comedy'), and therefore the category attribute in this case would be multi-valued. A multi-valued attribute may have a set of values with specified lower and upper limits. For example, the category attribute may have between one and three values.

5.2.3 Learning Outcome 3: Create and manipulate database objects

5.2.3.1 Introduction to the learning outcome

This section gives an overview of the various database objects including tables, queries, reports, and forms. It also focuses on creation of tables, linking these tables and extracting data from these tables using query objects. By the end of it, the learner should be able to competently identify and come up with the above-mentioned objects.

5.2.3.2 Performance Standard

5.2.3.2.1 Database objects are identified.

5.2.3.2.2 Appropriate ***data Attributes*** are applied.

5.2.3.2.3 Data relationships are established as per the tables created.

5.2.3.2.4 Data is extracted from database using Access.

5.2.3.3 Information Sheet

Creation and Manipulation of Database Objects

A database is a collection of information that is related. MS Access allows you to manage your information in one database file. Within Access there are four major objects: **Tables, Queries, Forms** and **Reports**

- **Tables** store your data in your database
- **Queries** ask questions about information stored in your tables.
- **Forms** allow you to input data into your tables
- **Reports** allow you to print data based on queries/tables that you have created.

The Navigation Panel in MS Access

The **Navigation Pane** is a list containing every object in your database. For easier viewing, the objects are organized into groups by type. You can **open**, **rename**, and **delete** objects using the Navigation Pane.

To Minimize and Maximize the Navigation Pane:

Click the **double arrow** « in the upper-right corner of the Navigation Pane to minimize and maximize.

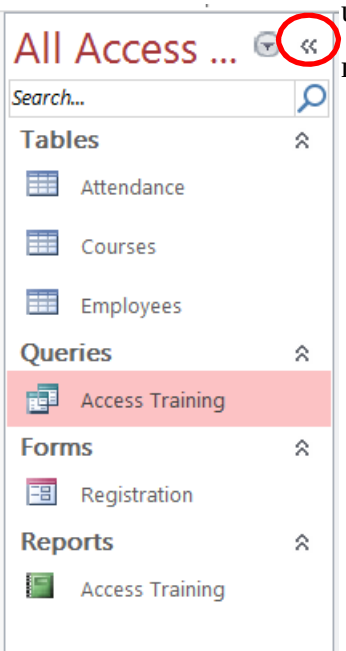


Figure 128: Navigation pane in Ms Access

Sorting the Objects in the Navigation Pane:

By default, objects are sorted by type, with the tables in one group, the forms in another, etc. However, you can change how the objects are sorted.

Click the **drop-down arrow** to the right of the **All Access Objects** and click on a sort option from the list.

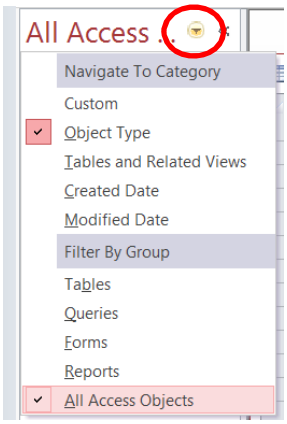


Figure 129: Choosing Access options

Creating a Database

1. Start **Access**
2. Click on **Blank desktop database.**
3. Under **File Name** type a name for the database
4. To change the location of where to store the database, click the folder icon and select a location.

Click **Create**

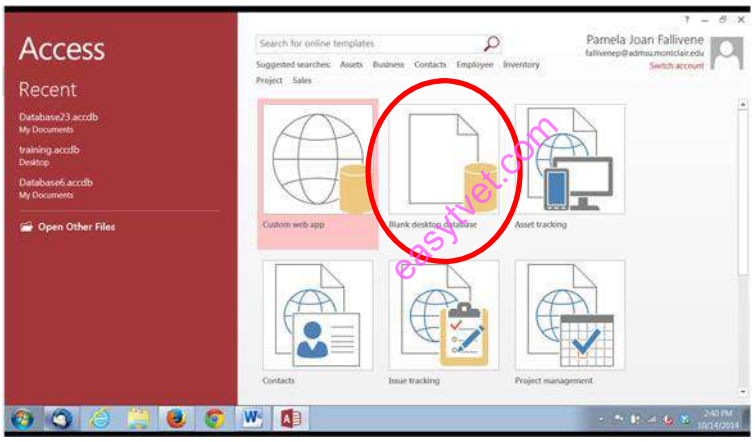


Figure 130:Creating a new database icon.

Access opens in a new table in **Datasheet View**.

Understanding Views

There are multiple ways to view a database object. The two views for tables are **Design View** and **Datasheet View.**

- **Design View** is used to set the data types, insert, or delete fields, and set the Primary Key.

- **Datasheet View** is used to enter and view the data for the records.

Switching Between Design View and Datasheet View:

Click the **View** arrow on the **Home** tab and click on either **Datasheet View** or **Design View**

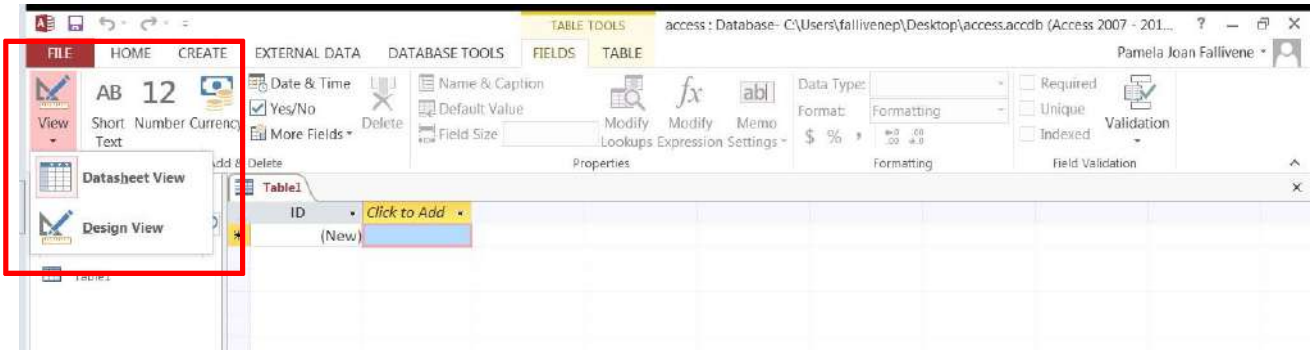


Figure 131: Switching between views

Creating a Table

A table is a collection of data about a specific topic, such as employee information, products, or customers. The first step in creating a table is entering the fields and data types. This can be done in either Datasheet View or Design View, but it is recommended to set up the table in **Design View.**

Understanding Fields and Their Data Types

Field - an element of a table that contains a specific item of information, such as a last name.

Field's Data Type - determines what kind of data the field can store.

Format	**Use to display**
Short Text	Alphanumeric data (names, titles, etc.) - up to 255 characters
Long Text	Large amounts of alphanumeric data: sentences and paragraphs – 64,000 characters

Number	Numeric data
Date/Time	Dates and times
Currency	Monetary values
AutoNumber	Unique value generated by Access for each new record
Yes/No	Yes and No values and fields that contain only one of two values
OLE Object	Pictures, graphs, or other ActiveX objects from another Windows-based application
Hyperlink	A link address to a document or file on the Internet
Attachment	You can attach files such as pictures, documents, spreadsheets, or charts; each Attachment field can contain an unlimited number of attachments per record, up to the storage limit of the size of a database file.
Calculated	You can create an expression that uses data from one or more fields. You can designate different result data types from the expression.
Lookup Wizard	Displays either a list of values that is retrieved from a table or query, or a set of values that you specified when you created the field. The Lookup Wizard starts and you can create a Lookup field. The data type of a Lookup field is either text or number, depending on the choices that you make in the wizard.

Table 41: Data types description summary

To Create a Table in Design View:

1. Click on the **Create tab**
2. Click on **Table**
3. Switch over to **Design View** on the Home tab
4. If prompted to save the table, enter a name and click on **OK**

5. Type the field names and select the appropriate data type for each field
6. Continue until all fields are added

Note: The order that you enter the field names is the order the fields will appear in the table

Setting a Primary Key

The **Primary Key** is the unique identifier for each record in a table. Access will not allow duplicate entries in a primary key field. When creating a new table, Access automatically creates a field "ID" with the auto number data type and assigns this as the Primary Key.

To Set a Primary Key:

1. In **Design View**, position your cursor in the field you wish to set as the Primary Key.
2. Click the **Primary Key** button on the toolbar.
3. **Save** the table.

Note: To turn off the Primary Key simply click on the Primary Key button again.

Input Masks

An **input mask** is used to pre-format a field to "look/act" a certain way when a user inputs data.

Examples: Social Security Number input mask automatically inserts the dashes; phone numbers automatically insert the parentheses and dashes.

The input mask data can either be stored in the table or simply displayed and not stored.

To Create an Input Mask for a Field:

1. In **Design View**, click in a field for which you would like to apply an input mask.
2. In the **Field Properties** section at the bottom of the screen, click in the Input Mask line and notice the **Build** button that appears at the right end of the line (see below):
3. Click the **Build** button to start the Input Mask Wizard (shown below).

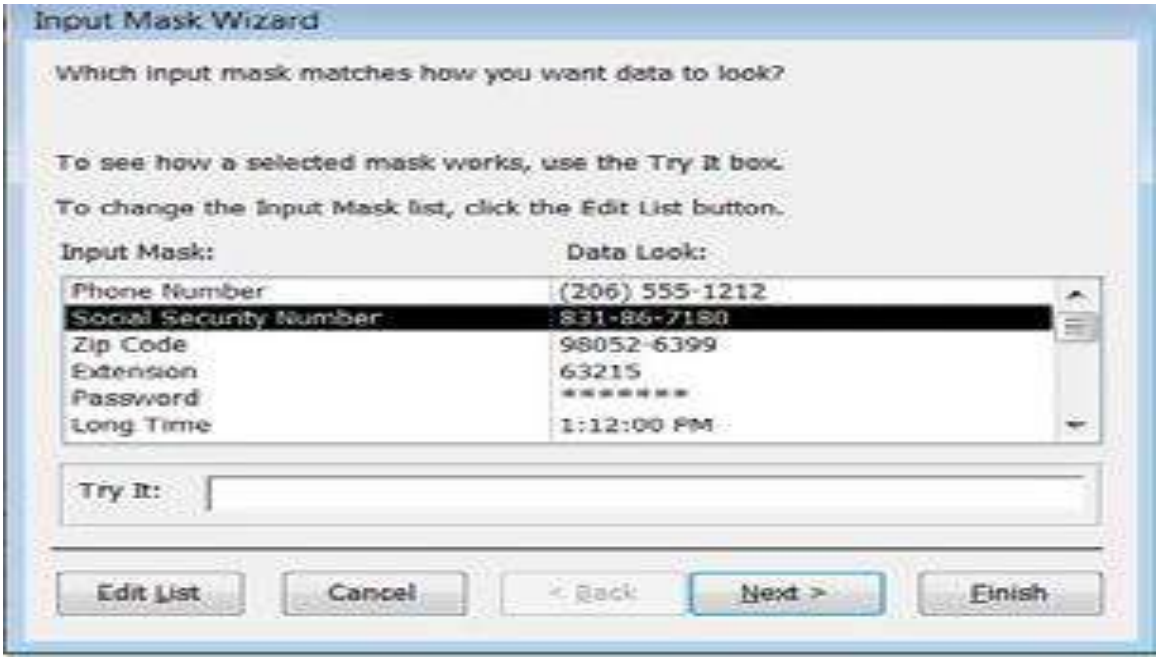


Figure 132 : Options when setting input mask.

4. Select the appropriate input mask.
5. Click **Next**
6. Click **Next** for additional screens on which you can set options for the input mask.
7. Click **Finish** on the last screen of the input mask wizard.

To Save the Table:

1. Click the **Save** icon on the toolbar
2. Enter a name for the table if you haven't done so already
3. Click **OK**

Entering Data in a Table

1. In **Datasheet View**, start typing the data into the table by pressing the tab key to move to the next cell
2. When you have completed the record (row), press **Enter**
3. You can also click on the **New record** icon on the **Home** tab in the **Records** group or at the bottom of the table

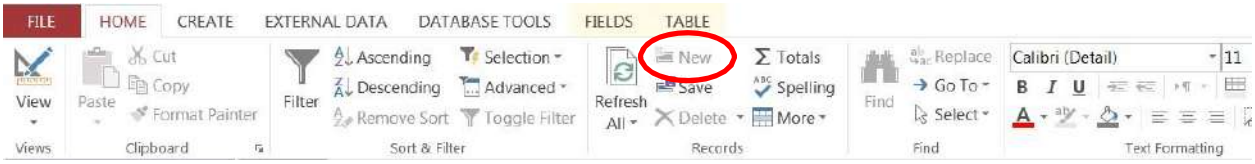


Figure 133: entering new data into a table

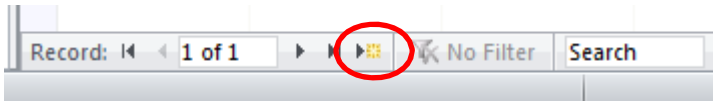


Figure 134: navigation tab

Navigating in a Table

Use the arrows at the bottom of the table to navigate among records.

Sorting Records in a Table

1. Position your cursor in the field that you wish to sort by clicking on any record in the table
2. Click either the **Sort Ascending** or **Sort Descending** icon on the **Home** tab in the **Sort & Filter** group

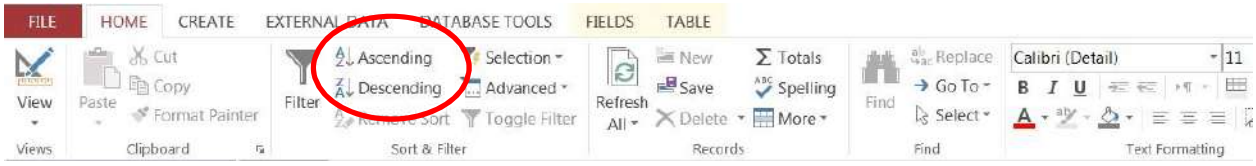


Figure 135: Sorting data options

LINKING TABLES /CREATING RELATIONSHIPS

A primary feature of relational databases is that they contain multiple tables, each of which can have a relationship with any of the other tables.

Assume we have three tables. One table will hold *customer* information, the other will hold *product* information, and the last will hold *order* information.

Figure 136: Customer table fields

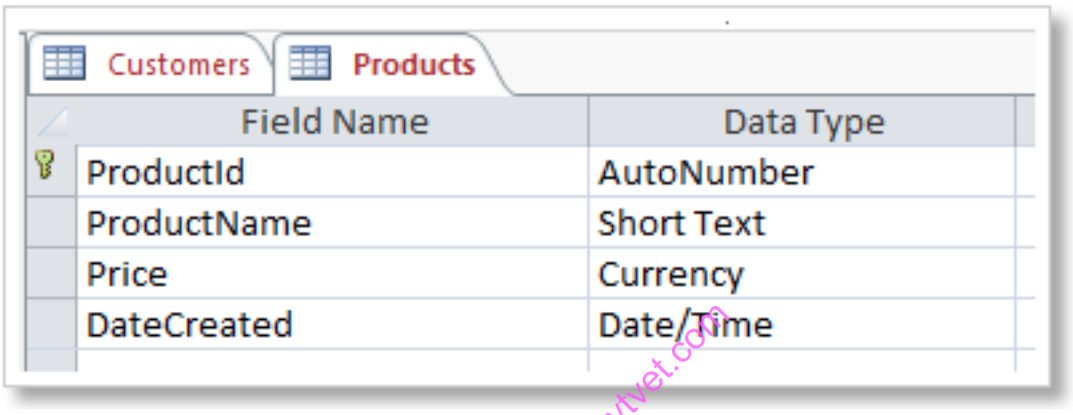


Figure 137: products table fields

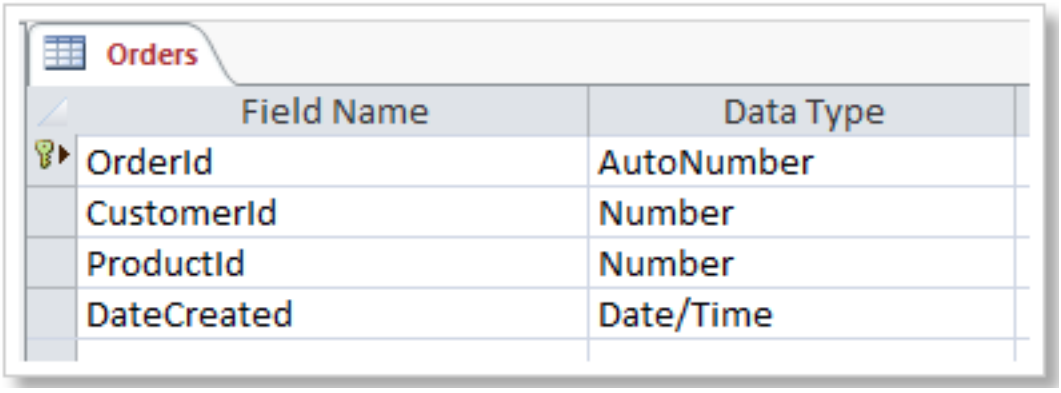


Figure 138: orders table fields

Be sure to set a format and default value for the DateCreated fields on both tables (the user shouldn't need to enter this field - the system can do that).

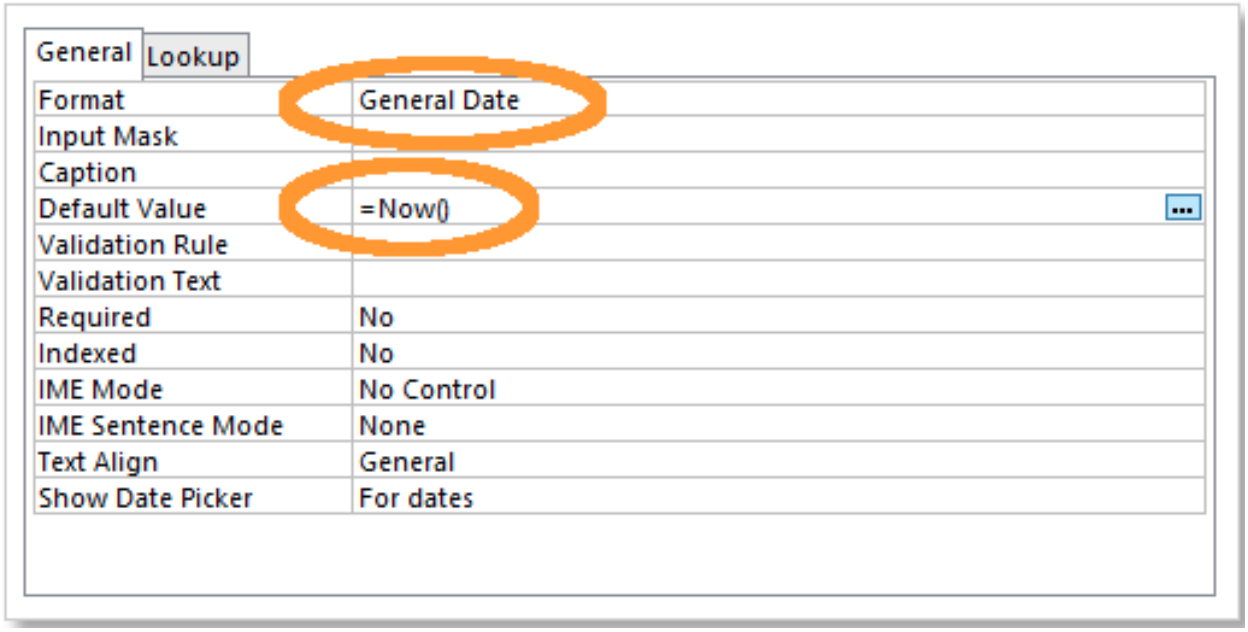


Figure 139: field properties settings

Remember that the key icon indicates that the field is a primary key. A primary key ensures that the data in this field is unique - no two values can be the same. A table must have a primary key before a relationship can be established with another table. Access automatically sets the ID field as a primary key, so as long as you simply rename ID to ProductId (or OrderId as the case may be) it will already be a primary key. If for some reason your table doesn't have a primary key set, or if you want to change the primary key field, you can simply right click on the field header (in Design view), and select Primary Key from the contextual menu.

Now that you've created two extra tables, our database has three tables. Now it's time to create the relationship between all three tables.

To create the relationship between all three tables.

1. While viewing a table in Design view, and ensuring that the DESIGN tab is selected, click Relationships from the Ribbon:

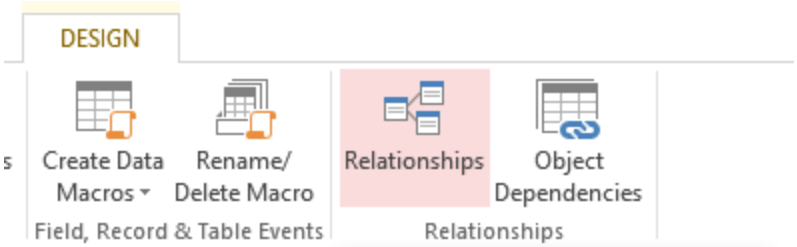


Figure 140: relationships option in Ms Access

2. A Show Table dialog box will pop up, displaying all three tables. Select all of them and click Add:

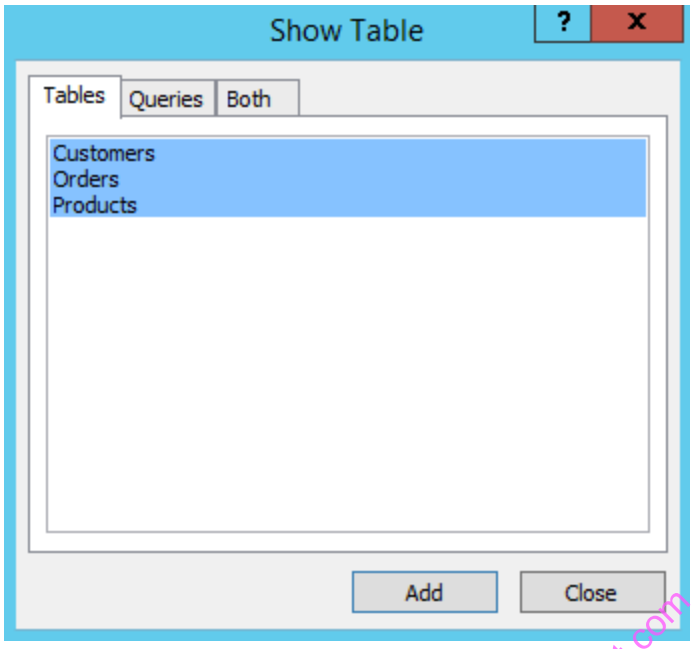


Figure 141: Interface for choosing tables in a relationship

3. Click Close to close the dialog box
4. You will now see three boxes which represent your three tables. Click and drag the CustomerId from the Customers table across to the corresponding CustomerId field on the Orders table:

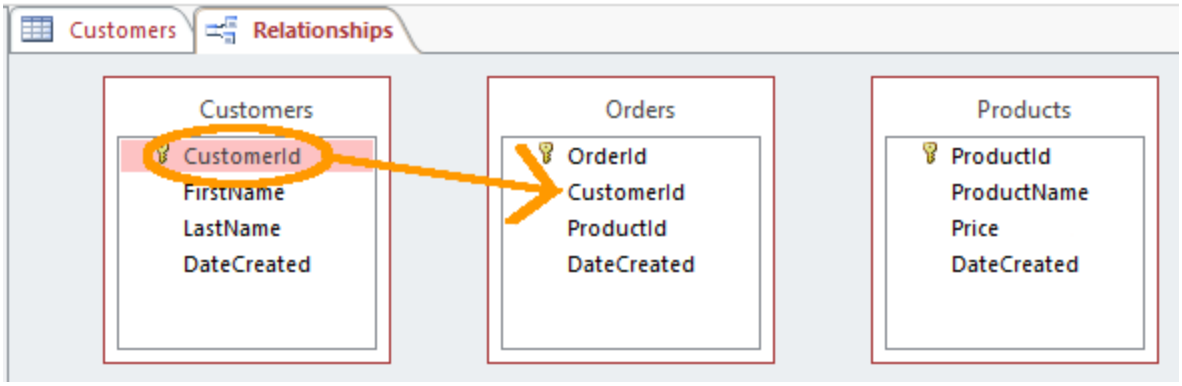


Figure 142: linking tables interface.

5. The Edit Relationships dialog will pop up. Click Enforce Referential Integrity so that it is checked. Check that the values are the same as the following screenshot and click OK:

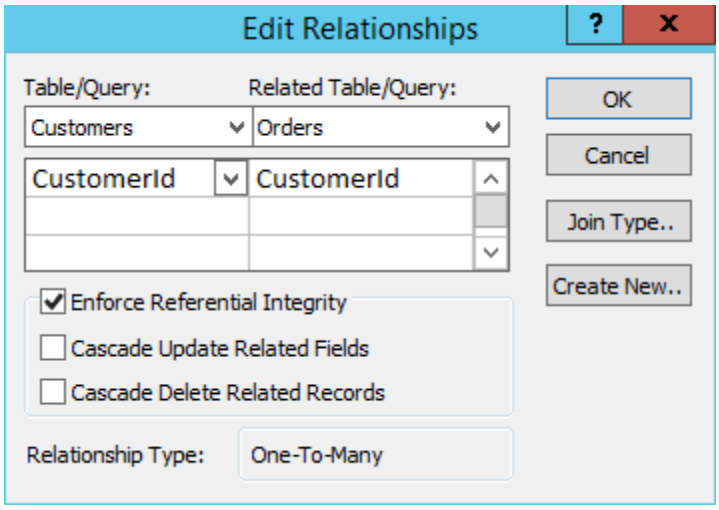


Figure 143: field matching window

You will now see a line established between the CustomerId field on the Customers table and the CustomerId on the Orders table.

6. Now do the same for the Products table. That is, click and drag the ProductId from the Products table across to the corresponding ProductId field on the Orders table. Your table relationships should look like the ones in the following screenshot:

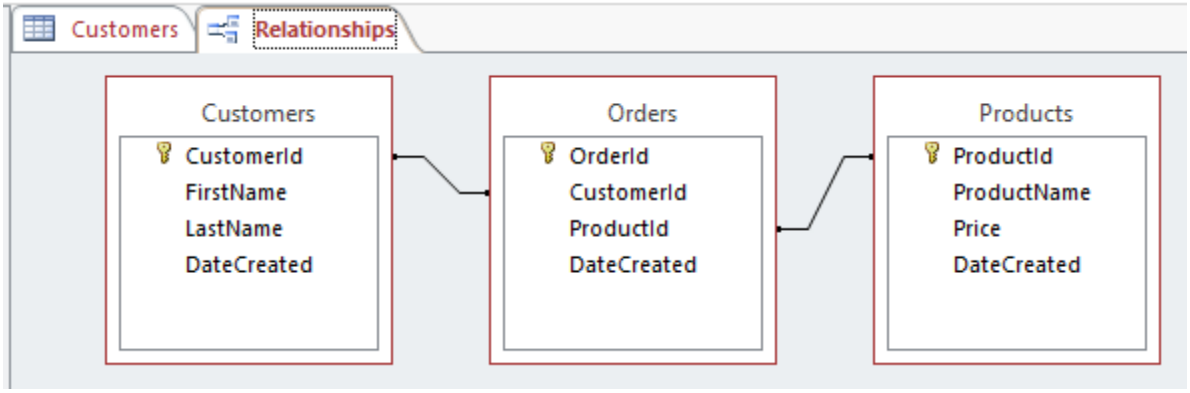


Figure 144: Sample relationship window

Relationship Types

We just established a many-to-many relationship. There are three types of relationships that you can establish between tables. These are as follows:

Many-To-Many Relationship

This is what our example above uses. A row in table A can have many matching rows in table B, and vice versa. In our case, a single customer can order many products, and a single product could have many customers. You create a many-to-many relationship by using a third table, called a junction table (more on that below).

One-To-Many Relationship

This is the most common relationship type. You don't need a third (junction) table for this type of relationship. In this type of relationship, a row in table A can have many matching rows in table B, but a row in table B can have only one matching row in table A. For example, a row in a Gender table (which contains the records Male and Female) can have many matching rows in a Customers table, but a row in the Customers table can only have one matching row in the Gender table. That is unless there was a business rule that allowed customers to be male *and* female at the same time. In this case, a many-to-many relationship would need to be established.

One-To-One Relationship

A row in table A can have only one matching row in table B, and vice versa. This is not a common relationship type, as the data in table B could just have easily been in table A. This relationship type is generally only used for security purposes, or to divide a large table, and perhaps a few other reasons.

The type of relationship that you use depends on the table structure and how the fields are defined.

Junction Table

In the above relationship, the Orders table is known as a *junction table*. A junction table is one that contains common fields from two or more other tables within the same database. It is used as a reference table in a many-to-many relationship (such as we are doing in our example).

Junction tables are known under many different names. Here are some: *cross-reference table*, *bridge table*, *join table*, *map table*, *intersection table*, *linking table*, *many-to-many resolver*, *link table*, *pairing table*, *pivot table*, *transition table*, or *association table*.

So, if you ever hear someone mention one of those, you should have some idea what they are referring to.

Primary Key vs Foreign Key

We already know that a *primary key* ensures that the data in the field is unique. This is important because our Orders table needs a unique value in order to reference any record from the other tables. For example, it can't use the FirstName field because there could be more than one person with a given first name. And it can't use the DateCreated value because it's possible that two records could be created at exactly the same time (eg, if two operators are entering data or if many records are imported from an external source). This is why we need at least one field that we *know* will only ever contain a unique value.

A *foreign key* is simply the primary key's corresponding field in the related table. So, in our example, the CustomerId field in the Orders table is a foreign key, while the CustomerId field in the Customers table is a primary key. Likewise, the Productid field in the Orders table is a foreign key, while the ProductId field in the Products table is a primary key. The fields do not necessarily need to have the same names but it's a good practice to get used to. It will make the database structure easier to understand - especially when you start adding more tables that also need to have relationships established.

Queries

A *query* refers to the action of instructing the database to return some (or all) of the data in your database. In other words, you are "querying" the database for some data that matches a given criteria. The queries are run against one or more tables to return only the data that you're interested in.

For example, you might like to see a list of all individuals whose last name is "Griffin". Or you might like to see a list of all users who have registered with your database over a given time period. You might also want to see which customers have ordered a particular product. To do all this, you need to perform a query.

You use queries to view, change, and analyse data in different ways. You can also use them as a source of records for forms and reports. Types of Query can be SELECT, INSERT, UPDATE, DELETE.

How do they help us?

- In choosing fields
- Choosing records that meet certain criteria
- In sorting records
- Performing calculations - can create calculated fields and add data to them

To Create a Query:

1. Click on the Create tab
2. Click on Query Design in the Queries group
3. Select the table that you would like to base your query on
4. Click Add
5. Repeat steps 3 and 4 until all tables are added
6. Close the Show Table window

The table(s) will now be displayed in the upper part of the Query Design Screen by boxes containing the tables' fields.

7. Double-click on the field names in the field list window which you would like to include in the query

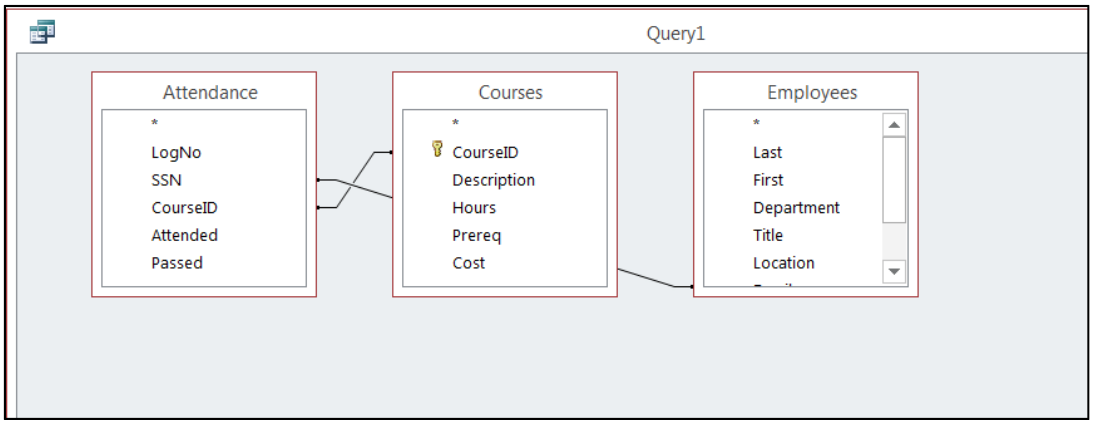


Figure 145: Sample relationship window

NOTE: Before we start querying the database, we should probably make sure we've got enough data in there first. Otherwise, it doesn't matter how good our queries are - they will all return zero results if there's zero data

Practical example:

Feel free to make up your own data. Here's mine:

Customers

CustomerId	FirstName	LastName	DateCreated	Cli
1	Homer	Simpson	13/06/2014 3:33:37 PM	
2	Peter	Griffin	13/06/2014 9:09:56 PM	
3	Stewie	Griffin	13/06/2014 9:16:07 PM	
4	Brian	Griffin	13/06/2014 9:16:36 PM	
5	Cosmo	Kramer	13/06/2014 9:16:41 PM	
6	Philip	Fry	13/06/2014 9:17:02 PM	
7	Amy	Wong	13/06/2014 9:22:05 PM	
8	Hubert J.	Farnsworth	13/06/2014 9:22:19 PM	
9	Marge	Simpson	13/06/2014 9:22:37 PM	
10	Bender	Rodríguez	13/06/2014 9:22:52 PM	
11	Turanga	Leela	13/06/2014 9:23:37 PM	
(New)			15/06/2014 9:00:01 PM	

Figure 146: customers table

Feel free to make up your own data. Here's mine:

Products | Ord...

ProductId	ProductName	Price	DateCreated
1	Venus Carrera ET	$190,000.99	15/06/2014 3:21:23 PM
2	Mars Dreamliner 787	$82,000.00	15/06/2014 3:21:53 PM
3	Mercury Riser 2020	$55,000.00	15/06/2014 3:22:20 PM
4	Pluto Mini Racer	$25,000.00	15/06/2014 3:22:23 PM
5	Mars Daytripper	$35,000.00	15/06/2014 3:22:28 PM
6	Saturn SUV	$65,750.00	15/06/2014 3:22:53 PM
7	Neptune 9000 Turbo	$88,990.00	15/06/2014 3:23:17 PM
(New)		$0.00	15/06/2014 4:08:53 PM

Figure 147: products table

Orders Table

Again, feel free to make up your own **but** you will need to make sure that the value in the CustomerId and ProductId fields match an actual ID in the respective tables. For example, if your Orders table contains a CustomerId of 5, you will need to make sure there's an actual customer in the Customer table with a CustomerId of 5. Here's some sample data for the Orders table:

Orders

OrderId	CustomerId	ProductId	DateCreated
1	1	1	15/06/2014 4:08:55 PM
2	2	2	15/06/2014 4:20:22 PM
3	10	3	15/06/2014 4:20:26 PM
4	10	4	15/06/2014 4:20:46 PM
5	3	5	15/06/2014 4:20:59 PM
6	4	1	15/06/2014 4:21:06 PM
8	5	3	15/06/2014 8:14:44 PM
9	6	6	15/06/2014 8:15:02 PM
10	7	4	15/06/2014 8:19:03 PM
11	8	2	15/06/2014 8:19:06 PM
12	1	6	15/06/2014 8:19:14 PM
13	6	1	15/06/2014 8:19:27 PM
(New)	0	0	15/06/2014 9:08:57 PM

Figure 148: Orders table

Create the Query

Now let's create a query that returns the names of all customers who have ordered a product.

1. Ensuring you have the CREATE tab open on the Ribbon, click Query Design

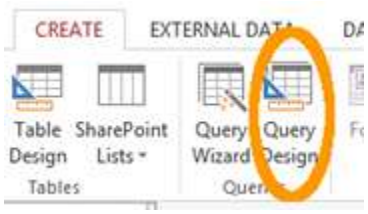


Figure 149: query design option

2. The Show Table dialog box will appear with all of our tables listed. Select all three tables and click Add, then click Close:

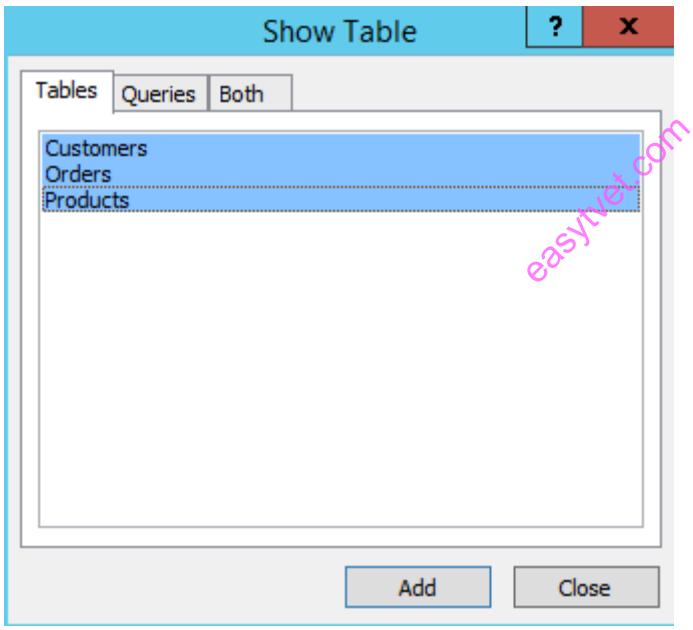


Figure 150: table selection window

3. The three tables are now represented in the top pane (beneath the Ribbon). Choose the fields you'd like to be presented in the results of your query. You can either double click on the field name or click and drag it down to a column in the bottom pane. Select the fields as

follows:

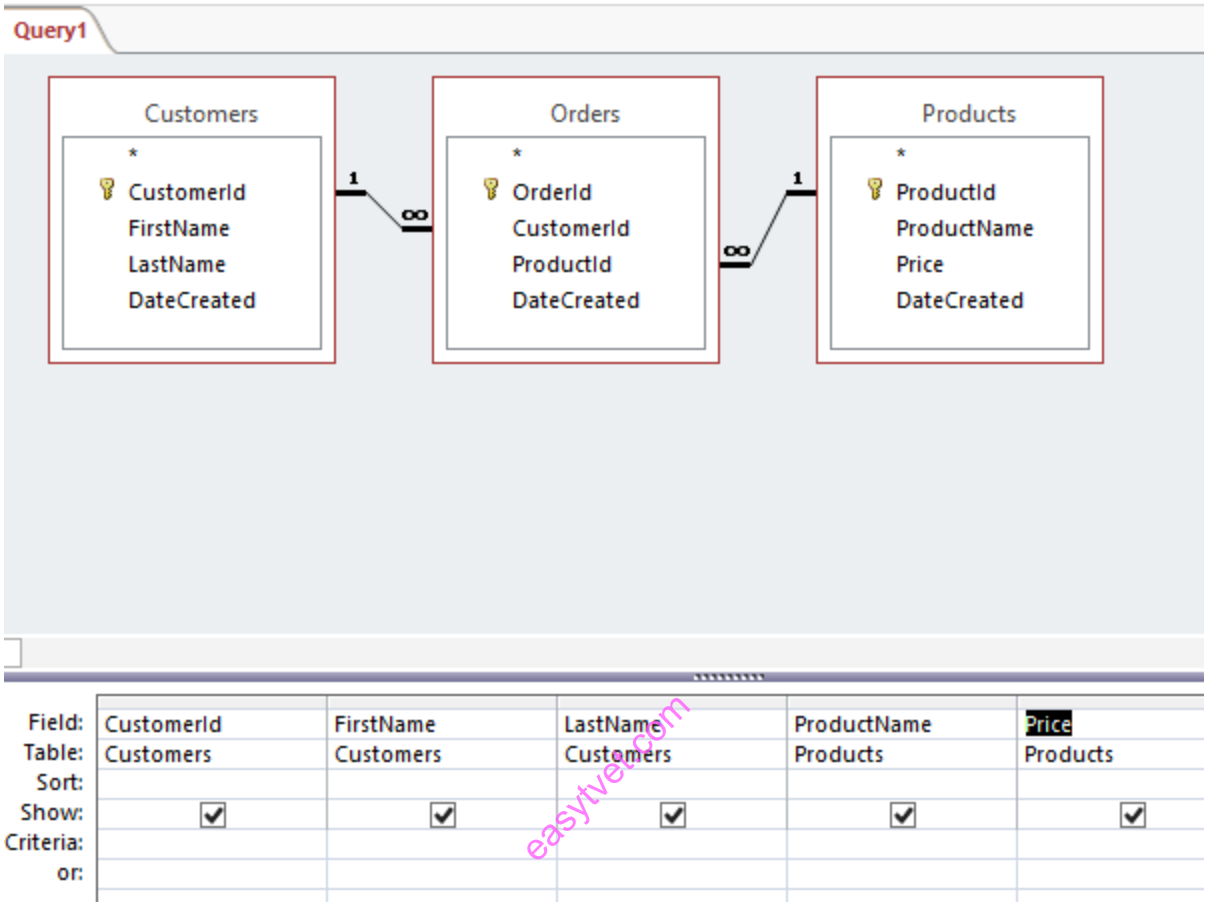


Figure 151: Query properties window

4. Click the Run button at the top-left part of the Ribbon:

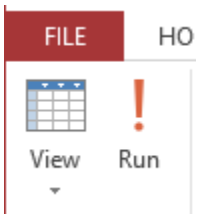


Figure 152: Query run option

5. You should now see the result of the query. Here's mine:

Query1

CustomerId	FirstName	LastName	ProductName	Price
1	Homer	Simpson	Venus Carrera ET	$190,000.99
2	Peter	Griffin	Mars Dreamliner 787	$82,000.00
10	Bender	Rodríguez	Mercury Riser 2020	$55,000.00
10	Bender	Rodríguez	Pluto Mini Racer	$25,000.00
3	Stewie	Griffin	Mars Daytripper	$35,000.00
4	Brian	Griffin	Venus Carrera ET	$190,000.99
5	Cosmo	Kramer	Mercury Riser 2020	$55,000.00
6	Philip	Fry	Saturn SUV	$65,750.00
7	Amy	Wong	Pluto Mini Racer	$25,000.00
8	Hubert J.	Farnsworth	Mars Dreamliner 787	$82,000.00
1	Homer	Simpson	Saturn SUV	$65,750.00
6	Philip	Fry	Venus Carrera ET	$190,000.99
*	(New)			

Figure 153: Sample query output

6. Save the query by right-clicking on the Query1 tab and giving it a name. Call it Customer Orders:

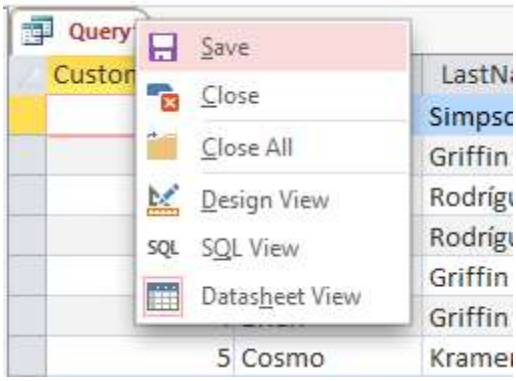


Figure 154: Save query option

What Does Our Query Actually Do?

Our query successfully returns a all customers who ordered at least one product and it includes the product and its price next to their name. You can also see that some customers have ordered more than one product. The query has automatically joined the three tables using the relationship that we created and returned the matching data.

This query will only return those customers who have ordered at least one product. If a customer is in the Customers table but does not order a product, that customer's record will not be displayed in the results of this query. If you look back at the Customers table (near the top of this page), you'll see that Marge Simpson is a customer but she hasn't ordered any products yet (i.e., she doesn't have a corresponding record in the Orders table). Well actually, Marge is only a *potential* customer. She simply expressed interest in buying a space ship.

Defining Criteria in the Query/Modifying a Query

In order to control which records are displayed, you must define criteria in a query. The most common type of query is the **Select Records** query which will be discussed below.

To Define Criteria for Your Query:

1. Position your cursor in the criteria row in the field for which you wish to define the criteria for
2. **Type** the criteria

Example: To find all Excel courses:

3. Position your cursor in the criteria row of the **Course ID** field
4. **Type** Excel (Access adds the quote marks to the criteria automatically when you tab to the next column)

Field:	First	First	CourseID	Description	Attended
Table:	Employees	Employees	Courses	Courses	Attendance
Sort:					
Show:	☑	☑	☑	☑	☑
Criteria:			Excel		

Figure 155: CourseID selection option

5. Click the **Run Query** button

To Save the Query:

1. Click the **Save** icon
2. Enter a name for the query
3. Click **OK**

Practical example case

we can tweak a query a little bit here and there to see what other results we can return.

Product Name

The marketing department wants to see who, if anyone, ordered their most expensive space ship. So, let's return a list of all customers who purchased the most expensive space ship - the "Venus Carrera ET".

1. From Design view, in the bottom pane, enter "Venus Carrera ET" (double quotes included) into the Criteria field of the ProductName field:

Field:	CustomerId	FirstName	LastName	ProductName	Price
Table:	Customers	Customers	Customers	Products	Products
Sort:					
Show:	☑	☑	☑	☑	☑
Criteria:				"Venus Carrera ET"	
or:					

Figure 156: Criteria setting option

2. Click the Run button in the Ribbon to view the results of the query. Alternatively, you can just click the Datasheet View icon. And here's the result:

Customer Orders

CustomerId	FirstName	LastName	ProductName	Price
1	Homer	Simpson	Venus Carrera ET	$190,000.99
4	Brian	Griffin	Venus Carrera ET	$190,000.99
6	Philip	Fry	Venus Carrera ET	$190,000.99

Figure 157: Sample query output

3. Once you're satisfied with the results, return to Design view and remove your modification (i.e. remove "Venus Carrera ET" from the Criteria field).

Price

You could also modify your query so that it returns customers who purchased a product worth over a certain price.

So let's construct a query that returns all customers who purchased any product with a price over 80,000.

1. From Design view, in the bottom pane, add (>80000) into the Criteria field of the Price field:

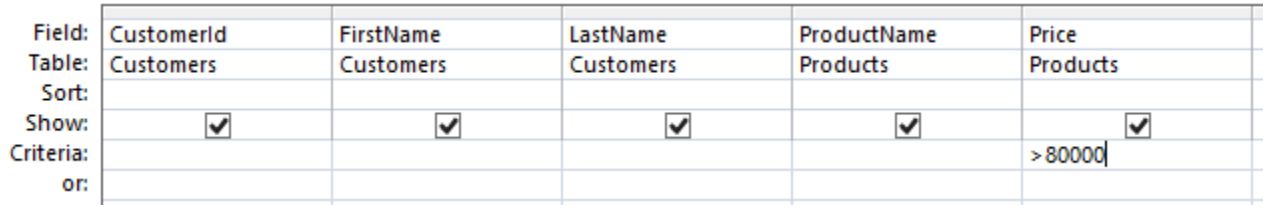


Figure 158: Query criteria setting window

2. Click the Run button in the Ribbon to view the results of the query. Alternatively, you can just click the Datasheet View icon. And here are the results:

Customer Orders

CustomerId	FirstName	LastName	ProductName	Price
1	Homer	Simpson	Venus Carrera ET	$190,000.99
2	Peter	Griffin	Mars Dreamliner 787	$82,000.00
4	Brian	Griffin	Venus Carrera ET	$190,000.99
8	Hubert J.	Farnsworth	Mars Dreamliner 787	$82,000.00
6	Philip	Fry	Venus Carrera ET	$190,000.99

Figure 159: sample query output

Price Using Totals

The previous query is all good and well, but there's one (potential) problem with it. While it certainly allows us to see all customers purchased a product over $80,000, it doesn't really paint the full picture. It doesn't display those customers who may have purchased more than one product, each of which is *less* than $80,000, but the total of which is *more* than $80,000.

Here's how we can capture that one.

1. From Design view, click the Totals button in the Ribbon:

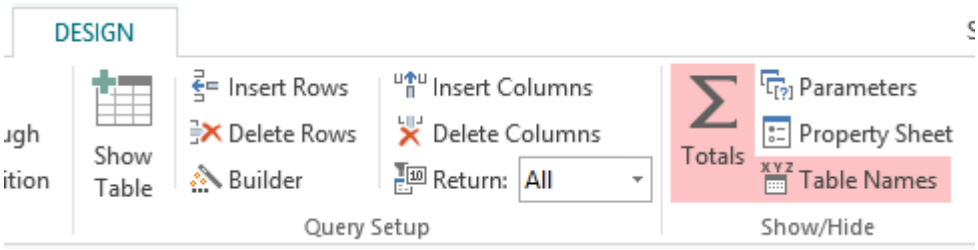


Figure 160: Query computations option

2. You will now see a Total row in the bottom pane. Each field will have Group By. In the Price field, click Group By and select Sum.
3. Click in the next row down (i.e. the Sort field) and select Descending. Keep the >80000 criteria in the Price field.
4. You will also need to delete the ProductName column by selecting it and clicking Delete Columns (next to the Totals button on the Ribbon). It should now look like this:

Field:	CustomerId	FirstName	LastName	Price
Table:	Customers	Customers	Customers	Products
Total:	Group By	Group By	Group By	Sum
Sort:				Descending
Show:	☑	☑	☑	☑
Criteria:				>80000
or:				

Figure 161: Setting query criteria window

5. Run the query. And here are our big spenders:

Customer Orders

CustomerId	FirstName	LastName	SumOfPrice
6	Philip	Fry	$255,750.99
1	Homer	Simpson	$255,750.99
4	Brian	Griffin	$190,000.99
8	Hubert J.	Farnsworth	$82,000.00
2	Peter	Griffin	$82,000.00

Figure 162: sample query output

You could spend a lot of time tweaking this one simple query. For example, here's some ideas:

- You could replace >80000 with >=80000
- You could remove the >80000 altogether

- You could sort by CustomerId instead
- You could use >50000 And <80000 (or any other price range)

QUERY SQL VIEW

The queries that we created and modified were all done in Design view. This is a user-friendly interface that allows you to create queries easily by visualizing the tables used in the query and having options made available to you.

But whenever you use Design view to construct a query, behind the scenes, MS Access is constructing an SQL statement in order to make the query work. For every query you create, there's an SQL statement hiding in the background.

Well, it's not quite "hiding"... after all, you can view it any time you wish. To do this, you need to go into SQL View.

Switching to SQL view

To switch to SQL view, you simply click the SQL icon at the bottom-right corner of Access:

SQL Statements

Once in SQL view, you will see a SQL statement. This SQL statement represents the query that you constructed. Here's the SQL view of the last query that we performed:

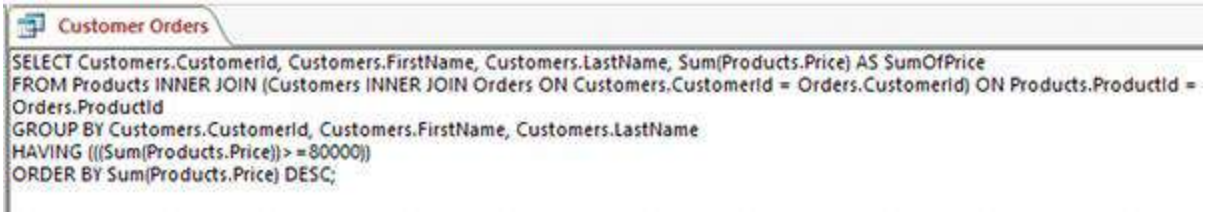


Figure 163: sample SQL query window

If you're not familiar with SQL, this might look a little scary. But it needn't be that way. That SQL statement is simply extracting records from the database using our precise criteria.

A less complex SQL statement:

SELECT customers name, first name, customers last name

FROM Customers

The above SQL statement returns the FirstName and Last Name fields from the Customers table. It returns *all* records because we didn't specify otherwise.

If we wanted to specify only *some* records, we would need to add some criteria to the statement. Here's a modified statement:

SELECT Customers.FirstName, Customers.LastName

FROM Customers

WHERE Customers.LastName = 'Griffin'

Like the first example, the above SQL statement returns the FirstName and Last Name fields from the Customers table. But in this example, we are only returning records where the customer's last name is "Griffin".

Creating a Form

A form is a database object that is used to enter or display data in a database.

To Create a Form:

1. Open the table or query on which you are basing the form
2. Click on the **Create** tab
3. Click on **Form** in the **Forms** group

A form is created and opens in Layout View.

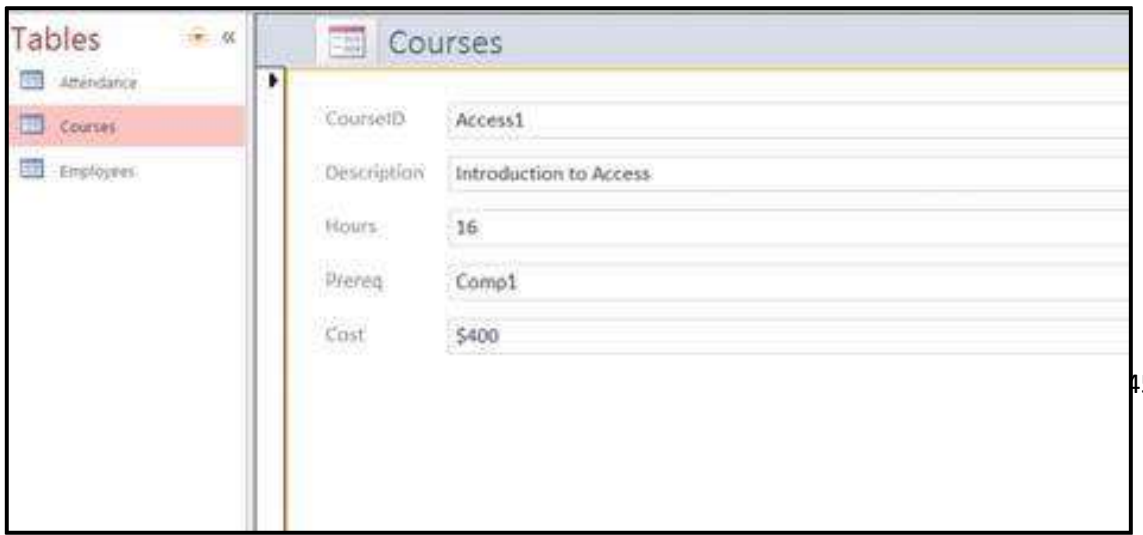


Figure 164: form in a layout view

Different Views:

Form View – this view allows you to view, create and edit records.

Layout View - this view is like Design View but is more visually oriented in that each control displays real data. As a result, this is a very useful view for setting the size of controls or performing many other tasks that affect the visual appearance and usability of the form.

Design View - this view gives you a more detailed view of the structure of the form. You can see the header, detail, and footer sections for the form. You cannot see the underlying data while you are making design changes.

Reports

Reports can be based on tables or queries.

To Create a Report:

1. Open the table or query on which you are basing the report
2. Click on the **Create** tab
3. Click on **Report** in the **Reports** group

A report is created in Layout View.

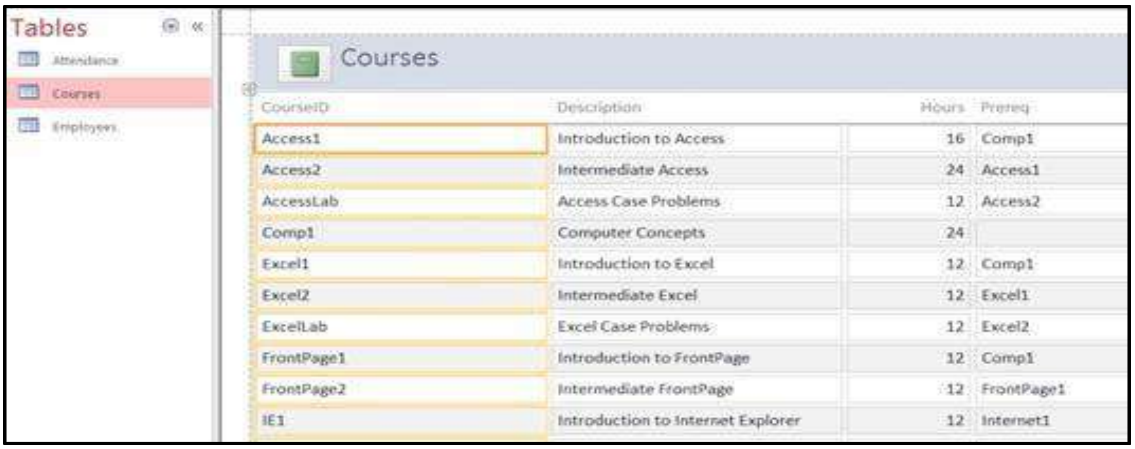


Figure 165: report in Layout view

Different Views:

Print Preview – allows you see what the report would look like on a printed piece of paper.

Report View – allows you to see the data without having to display it in Print Preview.

Layout View – allows you make design changes while browsing your data.

Design View - gives you a more detailed view of the structure of your report

Database Integration

The process of combining information from diverse sources, including databases, cloud, data warehouse, virtual databases, files, and more, to distribute a clean and consolidated version enterprise-wide is known as database integration. The main benefit of database integration system is that it makes data accessible to multiple stakeholders and client applications without the need to duplicate or move data.

One very common data management problem is integrating and synthesizing data from multiple data sources. How many different sources of data do you have to work with? And how are all these data sources related?

Most workplaces use multiple software solutions. The different software platforms may share related data, but it's almost impossible to pull the various data sets together for analysis. And then, beyond the software solutions, there are all those spreadsheets with related data.

At some point it almost becomes overwhelming. You have all this data, out there but no way to really bring it all together.

One of the best products, on the market, for data integration is Microsoft Access. Access has a lot of advantages over other integration software products. Firstly, MS Access is part of the Microsoft Office suite

Specific to integration, Access can link up to, and share information with:

1. Microsoft Excel
2. Microsoft Outlook
3. Microsoft SQL Server
4. Microsoft SharePoint Lists

In addition, MS Access can also linkup to data from the following data sources:

1. Other ODBC Databases, such as MySQL, etc…
2. Text Files
3. XML Files
4. dBase Files

At first the idea of linking data from multiple data sources may be overwhelming. But it really isn't all that hard, once you figure out how to do it. Following are some basic guidelines when using MS Access to integrate various data sources.

Primarily – with MS Access – there are a couple ways to work with external data. The first is simply importing the external data. For instance, you can open up one of your proprietary software applications and run an export file. Then you can import that file into MS Access. Since Access can read and import common text or Excel files, importing data into Access from other software packages is very doable. But there is another option as well.

With Microsoft Access, instead of importing data that has been exported from another software; you can link the other data source directly to Access. This allows you to read "live" production data.

If you open Microsoft Access and click on the **External Data** ribbon, you will see a group of ribbon commands labelled, "**Import & Link"**. These are the commands you want to play with,

to fully grasp all the possibilities for integrating your various data sources. Each ribbon command tool comes with wizards, to walk you through the various steps.

To begin with you may simply want to play with exporting &/or linking data from Excel. Access and Excel work quite well together, as they are both part of the Microsoft Office Suite. The most important thing to remember, before you start working to import/link a spreadsheet, is to prep the spreadsheet. You will want to make sure there are column headings, and that the spreadsheet is straight data (without sectioned data, etc…)

But, after you have prepped your spreadsheet, simply:

1. Open an Access database.
2. Click on the **External Data** ribbon
3. Click the **Excel** command tool on the **Import & Link** group
4. A wizard will pop up.

- You will be able to use a navigator and select the spreadsheet you prepped.
- You can choose to import data, append data to an existing table, or link data. --- Stay away from appending data to an existing table, until you've played with the other options. Once you have played with importing and linking – you'll feel more comfortable selecting the append option. (Do keep in mind, if you append to an existing table, you'll want your spreadsheet data to be formatted so that it is compatible with the table you append to.)

  Once you play with the differences between importing and linking data (with Excel) then you're in a better position to start working with some of the more sophisticated data sources. To import/link data from a SQL database, you'll need to learn how to work with ODBC data sources. ODBC connectors simply define connection strings to the database you want to read.

  After setting up an ODBC data source for the SQL database you want to connect to, then the wizards in Access can walk you through the various steps. Just make sure to save your ODBC data source in a network folder that you can find, when going through the import/linking process.

5.2.3.4 Learning Activities

Elimu secondary school intends to start maintaining student information using a MS Access. Required:

i. Create four tables namely **Students** (To capture students bio data), **Finance** (to capture students payment details), **Books issued** (to maintain a record of books borrowed by the students), Books **inventory** (to hold details of all the books available in the library)

ii. Populate these tables with some dummy student data

iii. Create relationships between these tables

iv. Create a query to show student details and the books their individual fee balance detail

v. Create a form to populate the students Table

vi. Create a report of all the books available in the library.

5.2.3.5 Self-Assessment

i. Differentiate between the functioning of Tables, queries, forms and reports

ii. Describe the following field datatypes

1) Currency
2) Autonumber
3) OLE object
4) Attachment

iii. What are the two major components of SQL and what function do they serve?

iv. Explain the function of each of the clauses in the SELECT statement. What restrictions are imposed on these clauses?

1) FROM
2) WHERE
3) GROUP BY
4) HAVING
5) ORDER BY

v. Describe the various types of relationships that may exist within entities in a database.

5.2.3.6 Tools, Equipment, Supplies and Materials

- Computer
- Database software (Ms Access)

- Printer
- stationery

5.2.3.7 References

Silberschatz A. and et al (2001), Database System Concepts, McGraw-Hill.

Garcia-Molina H. and et al (2000), Database System: The complete Book, Prentice Hall.

Elmasri R. and Navathe S(2000), Fundamentals of Database System, Addison Wesley.

Conolly& Begg (2010), Database System: a practical approach to design, implementation and management, Prentice Hall.

Ullman, and Jennifer Widom (2009). *Database Systems: The Complete Book (2nd edition)* by SBN 978-0131873254, Pearson Prentice Hall,

Jeffrey A. Hoffer, Mary B. Prescott, and Fred R. McFadden(2007). Modern Database Management (8th Ed.). Prentice-Hall.

Roger Jennings(2004) Special Edition Using Microsoft Office Access 2003, Que.

Alan Simpson, Margaret Levine Young, and Alison Barrows (2003) Access 2003 for DUMMIES, Wiley Publishing.

5.2.3.8 Model answers to self-assessment

i. Differentiate between the functioning of queries, forms and reports

- Tables store your data in your database
- Queries ask questions about information stored in your tables
- Forms allow you to view data stored in your tables
- Reports allow you to print data based on queries/tables that you have created

ii. Describe the following field datatypes

Currency- Monetary values

Autonumber- Unique value generated by Access for each new record

OLE object- Pictures, graphs, or other ActiveX objects from another Windows-based application

Attachment- You can attach files such as pictures, documents, spreadsheets, or charts; each Attachment field can contain an unlimited number of attachments per record, up to the storage limit of the size of a database file.

iii. What are the two major components of SQL and what function do they serve?

A data definition language (DDL) for defining the database structure.

A data manipulation language (DML) for retrieving and updating data.

iv. **Explain the function of each of the clauses in the SELECT statement. What restrictions are imposed on these clauses?**

FROM specifies the table or tables to be used;

WHERE filters the rows subject to some condition;

GROUP BY forms groups of rows with the same column value;

HAVING filters the groups subject to some condition;

SELECT specifies which columns are to appear in the output;

ORDER BY specifies the order of the output.

v. cribe the various types of relationships that may exist within entities in a database

Relationship Types

We just established a many-to-many relationship. There are three types of relationships that you can establish between tables. These are as follows:

Many-To-Many Relationship

This is what our example above uses. A row in table A can have many matching rows in table B, and vice versa. In our case, a single customer can order many products, and a single product could have many customers. You create a many-to-many relationship by using a third table, called a junction table (more on that below).

One-To-Many Relationship

This is the most common relationship type. You don't need a third (junction) table for this type of relationship. In this type of relationship, a row in table A can have many matching rows in table B, but a row in table B can have only one matching row in table A. For example, a row in a Gender table (which contains the records Male and Female) can have many matching rows in a

Customers table, but a row in the Customers table can only have one matching row in the Gender table. That is unless there was a business rule that allowed customers to be male *and* female at the same time. In this case, a many-to-many relationship would need to be established.

One-To-One Relationship

A row in table A can have only one matching row in table B, and vice versa. This is not a common relationship type, as the data in table B could just have easily been in table A. This relationship type is generally only used for security purposes, or to divide a large table, and perhaps a few other reasons.

The type of relationship that you use depends on the table structure and how the fields are defined.

5.2.4 Learning Outcome 4: Perform database testing.

5.2.4.1 Introduction to the learning outcome

This section gives an overview of the various database database testing strategies that can be carried out on a Database. By the end of it, the learner should be able to competently be able to perform database tasks as follows prepare test data, run the test data, check the test results, validate the results and report the findings.

5.2.4.2 Performance Standard

5.2.4.2.1 Test data is prepared.

5.2.4.2.2 Run the test data.

5.2.4.2.3 Check the test results.

5.2.4.2.4 Validate the results.

5.2.4.2.5 Report the findings.

5.2.4.3 Information Sheet

Database testing

Database Testing refers to the process of validating the data that is being stored in a database by verifying the objects controlling the data and various functionalities surrounding it.

A database is a systematic collection of data that provides data storage and helps in data manipulation. Data management becomes very easy using these databases as databases use objects for managing the data such as tables for storing data, view for data representations, functions, and triggers for data manipulation. Generally, the activities like checking data validity, testing data integrity, performance check relate, testing various procedures, triggers and functions in the database are covered during the database testing.

But to perform database testing, having sound knowledge of SQL is very important.

Importance of Database Testing

A database is a dump of data where the data is collected in an enormous amount and stored in a structured format. Although DBMS (Database Management System) provides an organized

way of managing, retrieving and storing this data, there are cases where data might get redundant, duplicated, etc. In such cases database testing comes into the picture which helps us in validating the data. Below are various aspects based on which a database needs to be validated:

1. **Data Mapping**

   Data mapping is an integral aspect of database testing which focuses on validating the data which traverses back and forth between the application and the backend database.

2. **ACID Properties Validation**

   ACID stands for **A**tomicity, **C**onsistency, **I**solation, and **D**urability. This is another important aspect that needs to be confirmed against each database transaction.

   - *Atomicity*: This means that all Database Transactions are atomic i.e., the transactions can result in either, Success or Failure. Also known as *All-or-Nothing*.
   - *Consistency*: This means that the database state will stay valid after the transaction is completed.
   - *Isolation*: This means that multiple transactions can be executed all at once without impacting one another and altering the database state.
   - *Durability*: This means that once a transaction is committed, it will preserve the changes without any fail irrespective of the effect of external factors.

3. **Data Integrity**

   Testing the data integrity of a database refers to the process of evaluating all kinds of processes, operations and methods that are used for accessing, managing and updating the database also known as the *CRUD* operations. This solely focuses on testing the accuracy and consistency of the data stored in the database so that we get the expected or desired results.

4. **Business Rule Conformity**

   With the increase in the complexity of the databases various components like relational constraints, triggers, stored procedures, etc also begin to complicate. In order to avoid this, the testers provide some SQL queries which are appropriate enough to validate the complex objects.

Types of Database Testing

There are 3 types of Database Testing as listed below:

1. Structural Testing
2. Functional Testing
3. Non-functional Testing

1. Structural Testing

The structural database testing is the process of validating all the elements that are present inside the data repository and are primarily used for data storage. These elements cannot be manipulated directly by the end-users. Validating database servers is one of the most important considerations and the testers who manage to complete this phase successfully acquire mastery in SQL queries.

Various Types of Structural testing are:

1) **Schema Testing**

This type of testing is also known as mapping testing and is performed to ensure that the schema mapping of the front end and the back end are in sync. Some of the important checkpoints of this testing are:

- Validates various types of schema formats that are associated with the databases.
- Verification is required for unmapped tables/views/columns.
- Verification is also required to ensure the consistency of the heterogeneous databases in an environment with the overall application mapping.
- Provides various tools for database schema validation.

2) **Database Table and Column Testing**

Some of the important checkpoints of this testing are:

- The compatibility of database fields and columns mapping at the back end and the front end.
- Validating the length and naming convention of the database fields and columns as per requirements.
- Detecting and validating any unused/unmapped database tables/columns.

- Validating the compatibility of the data type and field lengths at the backend database columns with the front end of the application.
- Validates that the users are able to provide desired inputs using the database fields which are specified in the business requirement specification documents.

3) **Keys and Indexes Testing**

Some of the important checkpoints of this testing are:

- Ensure that the required Primary Key and the Foreign Key constraints are already there on the required tables.
- Validate the references of the foreign keys.
- Ensure that, in two tables the data type of the primary key and the corresponding foreign keys are the same.
- Validate the names of all the keys and indexes based on the naming conventions.
- Check the required fields and indexes size and length.
- Ensure the creation of the Clustered indexes and Non-Clustered indexes in the required tables as per the business requirements.

4) **Stored Procedures Testing**

Some of the important checkpoints of this testing are:

- Validate the adoption of the required coding standard conventions, exception and error handling for all the stored procedures by the development team in all the modules of the application under testing.
- Ensure that the development team has covered all the conditions/loops by applying the required input data to the application under testing.
- Check if the development team has properly applied the TRIM operations or not each time the data was fetched from the specified database tables.
- Ensure that the required outputs are generated by manually executing the Stored Procedures.
- Ensure that the table fields are updated as specified by the application under testing by manually executing the Stored Procedures.

- Ensure that the required triggers are implicitly invoked by executing the Stored Procedures.
- Detect and validate any unused stored procedures.
- Validating the Null condition at the database level.
- Ensure that all the Stored Procedures and Functions have been executed and tested on the blank database that is under test.
- Validate the overall integration of the stored procedure modules as specified in the requirements of the application under testing.

5) **Trigger Testing**

Some of the important checkpoints of this testing are:

- Validating that the required coding conventions are followed in the coding phase of the Triggers.
- Ensure that the executed triggers are fulfilling the required conditions for the respective DML transactions.
- Check whether the data is updated correctly once the triggers have been executed.
- Validate the functionalities such as Update, Insert, Delete triggers functionality of the application under test.

6) **Database Server Validations**

Some of the important checkpoints of this testing are:

- Validate the database server configurations as specified in the business requirements.
- Ensure that the required user performs only those levels of actions that are required by the application under test.
- Ensure that the database server is capable of catering to the needs of the maximum number of user transactions that are allowed as in the business requirement specifications.

2. Functional Testing

Functional database testing is the process that ensures that the transactions and operations that are performed by the end-users are consistent with the meet the business specifications.

Various Types of Functional Testing are:

- **Black Box Testing**

Black Box Testing refers to the process that checks various functionalities by verifying the integration of the database. In this, the test cases are usually simple and are used to verify the incoming and outgoing data from the function. Various techniques like cause-effect graphing technique, boundary-value analysis, and equivalence partitioning are used to test the database functionality. It is generally performed at the early development stages and costs less when compared to other functional testings. But it comes with some drawbacks like some errors cannot be detected by it and there is no specification on how much of the program should be tested.

- **White Box Testing**

White Box Testing is concerned with the internal structure of the database and the users are unaware of the specification details. This testing requires database triggers and logical views testing which supports the database refactoring. Moreover, database functions, triggers, views, SQL queries, etc., are also tested in this. White box testing is used to validate the database tables, data models, database schema, etc. It adheres to the rules of Referential Integrity and selects the default table values to verify the database consistency. Techniques like condition coverage, decision coverage, statement coverage, etc. are often used to perform White Box testing. Unlike Black box testing coding errors can be easily detected to eliminate the internal bugs present in the database. The only drawback of this type of testing is it doesn't cover the SQL statements.

3. Non-Functional Testing

Nonfunctional testing is the process of performing load testing, stress testing, checking minimum system requirements which are required to meet the business specification along with detecting risks and optimizing the performance of the database.

Major types of Non-Functional Testing are:

- **Load Testing**

The primary function of performing load testing is to validate the performance impact of most of the running transactions in the database. In this testing, a tester is required to check the following conditions.

- What is the response time required for executing transactions for multiple users located remotely?
- What is the time taken by the database for fetching the specific records?

- **Stress Testing**

Stress testing is a testing process that is performed to identify the breakpoint of the system. Thus, in this testing, an application is loaded until the point the system fails. This point is known as a **breakpoint** of the database system. The commonly used Stress Testing Tools are **LoadRunner** and **WinRunner**.

Database Testing Stages

DB testing is not a tedious process and includes various stages in the database testing lifecycle in accordance with the test processes.

The key stages in database testing are:

1. Set Up Testing Pre-Requisites
2. Execute the Tests.
3. Verify Test Status
4. Validate Results
5. Consolidate and Publish Report

Now that you are aware of what is database testing and how to perform it, let me now throw some light on various tools that are majorly used for database testing.

Database Testing Tools

There are numerous tools in the market are used to generate the Test Data, manage it and finally perform database testing like Load Testing and Regression Testing, etc. Below I have listed down a few of the most preferred tools:

Category	Tools
Data Security Tools	• IBM Optim Data Privacy
Load Testing Tools	• Web Performance • Rad View • Mercury
Test Data Generator Tools	• Data Factory • DTM Data Generator • Turbo Data
Test Data Management Tool	• IBM Optim Test Data Management
Unit Testing Tools	• SQLUnit • TSQLUnit • DBFit • DBUnit

Figure 166: summary of database testing tools

Database Security Testing

Database security testing is done to find the loopholes in security mechanisms and also about finding the vulnerabilities or weaknesses of database system.

The main target of database security testing is to find out vulnerabilities in a system and to determine whether its data and resources are protected from potential intruders. Security testing defines a way to identify potential vulnerabilities effectively, when performed regularly.

Given below are the primary objectives of performing database security testing −

i. Authentication
ii. Authorization
iii. Confidentiality
iv. Availability
v. Integrity
vi. Resilience

DB Query Testing

Various SQL frameworks let you define layers of indirection between your SQL and its input(s); i.e., you declare and label the input datasets upon which a query depends. Unit testing frameworks can use this indirection to replace real input data with faked versions.
We can then run the code under test, using some faked input, and compare the output result rows against a set of expected outputs. If the actual output of the code under test matches the expected output, the test passes; if not, it fails.
This technique is simple and gives you real power to verify that a SQL script does what you think it does. You can pass faked inputs to your SQL that your real data may not currently contain, giving you confidence that it can robustly handle a wide range of data.

Guidelines for testing database queries:

1. Use an isolated database for unit testing (e.g., No other test runs or activity)
2. Always insert all the test data you intend to query within the same test.
3. Write the tests to randomly create different volumes of data e.g., random number of inserts say between 1 and 10 rows
4. Randomize the data e.g., for a boolean field random insert and true or false.
5. Keep a count in the test of the variables (e.g. number of rows, number of trues)
6. For the Asserts execute query and compare against local test variables
7. Use Enterprises Services transactions to rollback database to previous state

There are a few tools out there to help you with this. DbUnit is one of them and I also believe Microsoft had a tool Visual Studio for Database Professionals that contained some support for DB testing.

SQL Queries most used in DB Testing

Querying the recent data

Select top 1 * from table name order by date/time desc

Ex: Select top 1 * from employee order by joining date desc

Checking for the duplicate records

Select * from table name group by key column having count (*) > 1

Ex: Select * from employee group by empid having count (*) > 1

Find out the existence of objects.

Select * from sysobjects where type = "<type>" (SQL Server)

where <type> can be

U ------- User table

V ------- View

P ------- Stored procedure

TR ------ Trigger

Select * from User Objects where Object name like 'xxxxxx' (Oracle)

Types of Threats on a Database System

1. *SQL Injection*

This is most common type of attack in a database system where malicious SQL statements are inserted in the database system and are executed to get critical information from the database system. This attack takes advantage of loopholes in implementation of user applications. To prevent this, user inputs fields should be carefully handled.

2. *Privilege Elevation in Database*

In this attack, a user already has some access in the database system and he/she only tries to elevate this access higher level so that he/she can perform some unauthorized activities in database system.

3. *Denial of Service*

In this type of attack, an attacker makes a database system or application resource unavailable to its legitimate users. Applications can also be attacked in ways that render the application, and sometimes the entire machine, unusable.

4. Unauthorized Access to data

Another type of attack is gaining unauthorized access to data within an application or database system. Unauthorized access includes −

- Unauthorized access to data via user-based applications
- Unauthorized access to by monitoring the access of others.
- Unauthorized access to reusable client authentication information

5. Identity Spoofing

In Identity Spoofing, a hacker uses the credentials of a user or device to launch attacks against network hosts, steal data or bypass access controls to database system. Preventing this attack requires IT-infrastructure and network-level mitigations.

6. Data Manipulation

In a data manipulation attack, a hacker changes data to gain some advantage or to damage the image of database owners.

Database Security Testing Techniques

7. Penetration Testing

A penetration test is an attack on a computer system with the intention of finding security loopholes, potentially gaining access to it, its functionality and data.

Risk Finding

Risk Finding is a process of assessing and deciding on the risk involved with the type of loss and the possibility of vulnerability occurrence. This is determined within the organization by various interviews, discussions and analysis.

1. SQL Injection Test

It involves checking the user inputs in application fields. For example, entering a special character like ‘,’ or ‘;’ in any text box in a user application should not be allowed. When a database error occurs, it means that the user input is inserted in some query, which is then executed by the application. In such a case, the application is vulnerable to SQL injection.

These attacks are a big threat to data as the attackers can get access to important information from the server database. To check SQL injection entry points into your web application, find

out code from your code base where direct MySQL queries are executed on the database by accepting some user inputs.

SQL Injection Testing can be performed for Brackets, Commas, and Quotation marks.

2. Password Cracking

This is the most important check while performing database system testing. To access critical information, hackers can use a password-cracking tool or can guess a common username/password. These common passwords are easily available on internet and also password cracking tools exist freely.

Therefore, it is necessary to check at the time of testing if the password policy is maintained in the system. In case of any banking and finance applications, there is a need to set a strict password policy on all the critical information database systems.

Security Audit of Database System

A security audit is a process of evaluating company's security policies at a regular time interval to determine whether necessary standards are followed or not. Various security standards can be followed as per business requirement to define the security policy and then assessment of set policies against those standards can be done.

Example of most common security standards are ISO 27001, BS15999, etc

Database Security Testing Tools

There are various system testing tools available in market, which can be used to test OS and application check. Some of the most common tools are discussed below.

a. Zed Attack Proxy

It is a penetration-testing tool for finding vulnerabilities in web applications. It is designed to be used by people with a wide range of security experience and as such is ideal for developers and functional testers who are new to penetration testing. It is commonly used for Windows, Linux, Mac OS.

b. Paros

All HTTP and HTTPS data between server and client, including cookies and form fields, can be intercepted and modified using these scanners. It is used for Cross-platform, Java JRE/JDK 1.4.2 or above.

c. Social Engineer Toolkit

It is an open-source tool and human elements are attacked rather than the system element. It enables you to send emails, java applets etc. containing the attack code. It is preferred for Linux, Apple Mac OS X and Microsoft Windows.

d. Skip fish

This tool is used to scan their sites for vulnerabilities. Reports generated by the tool are meant to serve as a foundation for professional web application security assessments. It is preferred for Linux, FreeBSD, MacOS X, and Windows.

e. Vega

It is an open source, multiplatform web security tool that is used to find instances of SQL injection, cross-site scripting (XSS), and other vulnerabilities in web applications. It is preferred for Java, Linux, and Windows.

f. Wapiti

Wapiti is an open source and web-based tool that scans the web pages of the web application and check for scripts and forms where it can inject data. It is built with Python and can detect File handling errors, Database, XSS, LDAP and CRLF injections, Command execution detection.

g. Web Scarab

It is written in Java and is used for analysing the applications that communicate through HTTP/HTTPS protocols. This tool is primarily designed for developers who can write code themselves. This tool is not OS dependent.

Database Recovery Testing

Database recovery testing is used to ensure that the database is recovered. Recovery testing allows you to find out whether the application is running properly and to check retrieving invaluable data that would have been lost if your recovery method is not properly setup.

You also check if several critical processes are running smooth to ensure that the data recovery will pass smoothly through the testing phase.

You can perform the following checks for database recovery −

- Any errors or mistakes in the backup software and you need to resolve these issues at an earlier stage.
- You need to conduct the recovery testing so that you will know what to do in case of an emergency.
- You need to check recovery testing needs so that you can plan for an effective recovery strategy.
- You should also know how you can recover the documents.

You need to run the recovery tests in early phase of the project. This allows you to remove and throw away every type of errors from the system. Here is a list of some of important points, which should be considered at the time of testing −

- Time span when changes or modifications occurs in database system.
- The period by which you want your recovery plan conducted.
- The sensitivity of data in database system. More critical the data is, the more regularly you will need to test the software.

Common Steps in Database Backup and Recovery Testing

In database recovery testing, you need to run the test in the actual environment to check if the system or the data can be recovered in case of any disasters and any other unforeseen events in the business environment.

Given below are the common actions performed in Database Recovery Testing −

- Testing of database system
- Testing of the SQL files
- Testing of partial files
- Testing of data backup
- Testing of Backup tool
- Testing log backups

5.2.4.4 Learning Activities

a) Database testing involves among other items, checking on the security of a database. Describe any four internal features provided by a database for ensuring security.

b) Given the primary objectives of performing database security testing below, describe how measures can be used to enforce each of them.

i. Authentication
ii. Authorization
iii. Confidentiality
iv. Availability
v. Integrity

5.2.4.5 Self-Assessment

i. What is the purpose of testing the database system?

ii. Explain the following in terms of providing security for a database.
   a. authorization.
   b. views.
   c. backup and recovery.
   d. integrity.
   e. RAID.
   f. Authorization
   g. Views
   h. Backup and recovery
   i. Integrity constraints
   j. Encryption
   k. Redundant Array of Independent Disks (RAID)

iii. Describe the security features of Microsoft Access.

5.2.4.6 Tools, Equipment, Supplies and Materials

- Computer
- Database software (Ms Access)
- Printer
- stationery
- internet connectivity (to access open-source testing tools)

5.2.4.7 References

Silberschatz A. and et al (2001), Database System Concepts, McGraw-Hill.

Garcia-Molina H. and et al (2000), Database System: The complete Book, Prentice Hall.

Elmasri R. and Navathe S (2000), Fundamentals of Database System, Addison Wesley.

Conolly& Begg (2010), Database System: a practical approach to design, implementation and management, Prentice Hall.

Ullman, and Jennifer Widom (2009). *Database Systems: The Complete Book (2nd edition)* by SBN 978-0131873254, Pearson Prentice Hall,

Jeffrey A. Hoffer, Mary B. Prescott, and Fred R. McFadden (2007). Modern Database Management (8th Ed.). Prentice-Hall.

Roger Jennings (2004) Special Edition Using Microsoft Office Access 2003, Que.

Alan Simpson, Margaret Levine Young, and Alison Barrows (2003) Access 2003 for DUMMIES, Wiley Publishing.

5.2.4.8 Model answers to self-assessment

I. Explain the purpose of testing the database system.

Before going live, the newly developed database system should be thoroughly tested. This is achieved using carefully planned test strategies and realistic data so that the entire testing process is methodically and rigorously carried out. Note that in our definition of testing we have not used the commonly held view that testing is the process of demonstrating that faults are not present. In fact, testing cannot show the absence of faults; it can show only that software faults are present. If testing is conducted successfully, it will uncover errors in the application programs and possibly the database structure. As a secondary benefit, testing demonstrates that the database and the application programs *appear* to be working according to their specification and that performance requirements *appear* to be satisfied. In addition, metrics collected from the testing stage provides a measure of software reliability and software quality.

As with database design, the users of the new system should be involved in the testing process. The ideal situation for system testing is to have a test database on a separate hardware system, but often this is not available. If real data is to be used, it is essential to have backups taken in case of error.

Testing should also cover usability of the database system. Ideally, an evaluation should be conducted against a usability specification. Examples of criteria that can be used to conduct the evaluation include (Sommerville, 2000):

- Learnability - How long does it take a new user to become productive with the system?
- Performance - How well does the system response match the user's work practice?
- Robustness - How tolerant is the system of user error?
- Recoverability - How good is the system at recovering from user errors?
- Adaptability - How closely is the system tied to a single model of work?

Some of these criteria may be evaluated in other stages of the lifecycle. After testing is complete, the database system is ready to be 'signed off' and handed over to the users.

II. Explain the following in terms of providing security for a database:

1. Authorization.
2. Views.

3. Backup and Recovery.
4. Integrity.
5. Encryption.
6. Raid.

Authorization

Authorization is the granting of a right or privilege that enables a subject to have legitimate access to a system or a system's object. Authorization controls can be built into the software and govern not only what database system or object a specified user can access, but also what the user may do with it. The process of authorization involves authentication of a subject requesting access to an object, where 'subject' represents a user or program and 'object' represents a database table, view, procedure, trigger, or any other object that can be created within the database system.

Views

A view is a *virtual table* that does not necessarily exist in the database but can be produced upon request by a particular user, at the time of request. The view mechanism provides a powerful and flexible security mechanism by hiding parts of the database from certain users. The user is not aware of the existence of any columns or rows that are missing from the view. A view can be defined over several tables with a user being granted the appropriate privilege to use it, but not to use the base tables. In this way, using a view is more restrictive than simply having certain privileges granted to a user on the base table(s).

Backup and recovery

Backup is the process of periodically taking a copy of the database and log file (and possibly programs) onto offline storage media. A DBMS should provide backup facilities to assist with the recovery of a database following failure. To keep track of database transactions, the DBMS maintains a special file called a log file (or journal) that contains information about all updates to the database. It is always advisable to make backup copies of the database and log file at regular intervals and to ensure that the copies are in a secure location. In the event of a failure that renders the database unusable, the backup copy and the details captured in the log file are used to

restore the database to the latest possible consistent state. Journaling is the process of keeping and maintaining a log file (or journal) of all changes made to the database to enable recovery to be undertaken effectively in the event of a failure.

Integrity constraints

Contribute to maintaining a secure database system by preventing data from becoming invalid, and hence giving misleading or incorrect results.

Encryption

Is the encoding of the data by a special algorithm that renders the data unreadable by any program without the decryption key. If a database system holds particularly sensitive data, it may be deemed necessary to encode it as a precaution against possible external threats or attempts to access it. Some DBMSs provide an encryption facility for this purpose. The DBMS can access the data (after decoding it), although there is degradation in performance because of the time taken to decode it. Encryption also protects data transmitted over communication lines. There are a number of techniques for encoding data to conceal the information; some are termed irreversible and others reversible. *Irreversible techniques*, as the name implies, do not permit the original data to be known. However, the data can be used to obtain valid statistical information. *Reversible techniques* are more commonly used. To transmit data securely over insecure networks requires the use of a cryptosystem, which includes:

- an encryption key to encrypt the data (plaintext);
- an encryption algorithm that, with the encryption key, transforms the plain text into ciphertext;
- a decryption key to decrypt the ciphertext;
- a decryption algorithm that, with the decryption key, transforms the ciphertext back into plain text.

Redundant Array of Independent Disks (RAID)

RAID works by having a large disk array comprising an arrangement of several independent disks that are organized to improve reliability and at the same time increase performance. The hardware that the DBMS is running on must be *fault-tolerant*, meaning that the DBMS should continue to operate even if one of the hardware components fails. This suggests having

redundant components that can be seamlessly integrated into the working system whenever there is one or more component failures. The main hardware components that should be fault-tolerant include disk drives, disk controllers, CPU, power supplies, and cooling fans. Disk drives are the most vulnerable components with the shortest times between failures of any of the hardware components.

One solution is the use of Redundant Array of Independent Disks (RAID) technology. RAID works by having a large disk array comprising an arrangement of several independent disks that are organized to improve reliability and at the same time increase performance.

i. **Describe the security features of Microsoft Access.**

Access provides a number of security features including the following two methods:

(a) setting a password for opening a database (system security);

(b) user-level security, which can be used to limit the parts of the database that a user can read or update (data security).

In addition to the above two methods of securing a Microsoft Access database, other security features include:

- *Encryption/decryption:* encrypting a database compacts a database file and makes it indecipherable by a utility program or word processor. This is useful if you wish to transmit a database electronically or when you store it on a floppy disk or compact disc. Decrypting a database reverses the encryption.
- *Preventing users from replicating a database, setting passwords, or setting startup options*;
- *Securing VBA code:* this can be achieved by setting a password that you enter once per session or by saving the database as an MDE file, which compiles the VBA source code before removing it from the database. Saving the database as an MDE file also prevents users from modifying forms and reports without requiring them to specify a log on password or without you having to set up user-level security

5.2.5. Learning Outcome 5: Print Database objects

5.2.5.1 Introduction to the learning outcome

This section gives an overview of the procedure of printing the various database objects including tables, queries, reports, and forms. By the end of it, the learner should be able to competently print database objects.

5.2.5.2 Performance Standard

5.2.5.2.1 Database tables are printed.

5.2.5.2.2 Database queries are printed.

5.2.5.2.3 Database forms and reports are printed.

5.2.5.3 Information Sheet

Most database objects including tables, queries, forms and reports, and the information they contain can be printed. Sometimes it is a good idea to preview a database object onscreen to see if something needs to be changed before sending it to the printer. You can preview a database object by clicking the Print Preview button on the toolbar.

While you can print data from a table, query, form or report in your Microsoft Access database, you can set print options for reports that will produce an eye-catching and more readable printed copy of your data.

Printing data directly from tables or queries works when you want to print just a simple list. Forms are better suited for viewing or printing data for a single record or a related set of records, and not for printing larger sets of data or data summaries. By printing data from a report, you can compare, summarize, and total large sets of data, and then present that information in an attractive layout.

Select page setup options.

After you create your report you can set options for the print layout in the **Page Setup** dialog box. These options help you fine tune the format with options for margin settings, use of gridlines and columns, or print only the data from the report without the formatting. To set the print page options, complete the following steps:

1. In the Navigation Pane, right-click the report and then click **Print Preview**.

2. On the **Print Preview** tab, in the **Page Layout** group, click **Page Setup** and set the margins, orientation and column settings that you want.

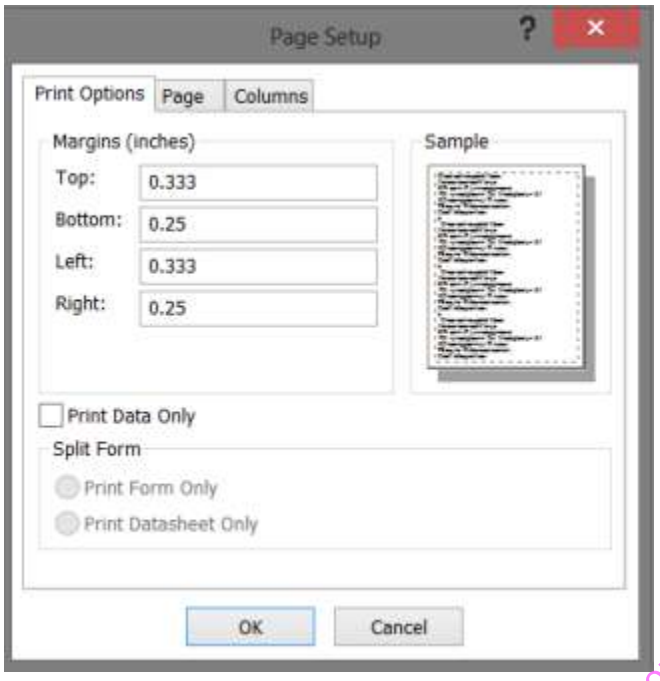
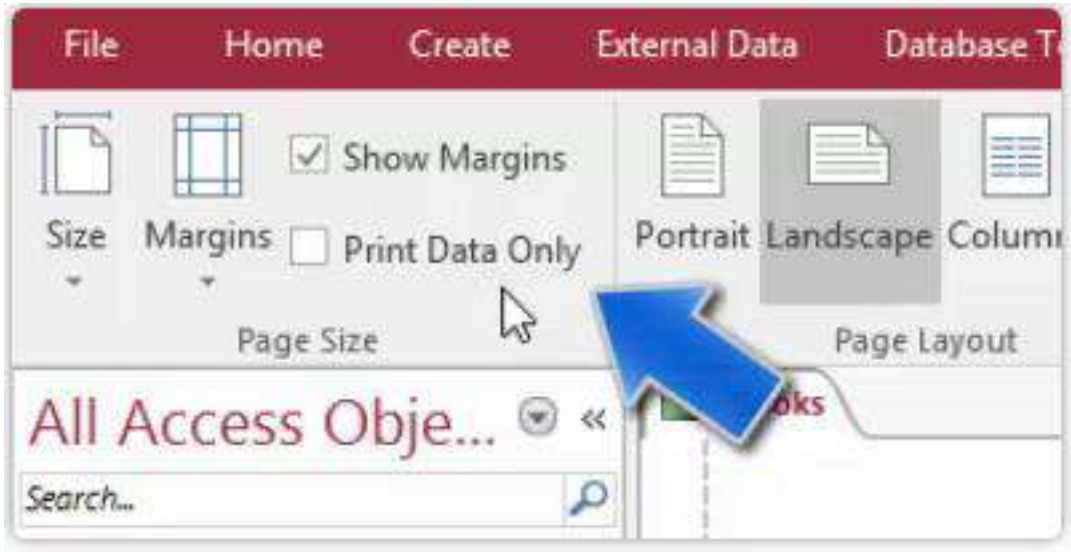


Figure 167: print option tab settings

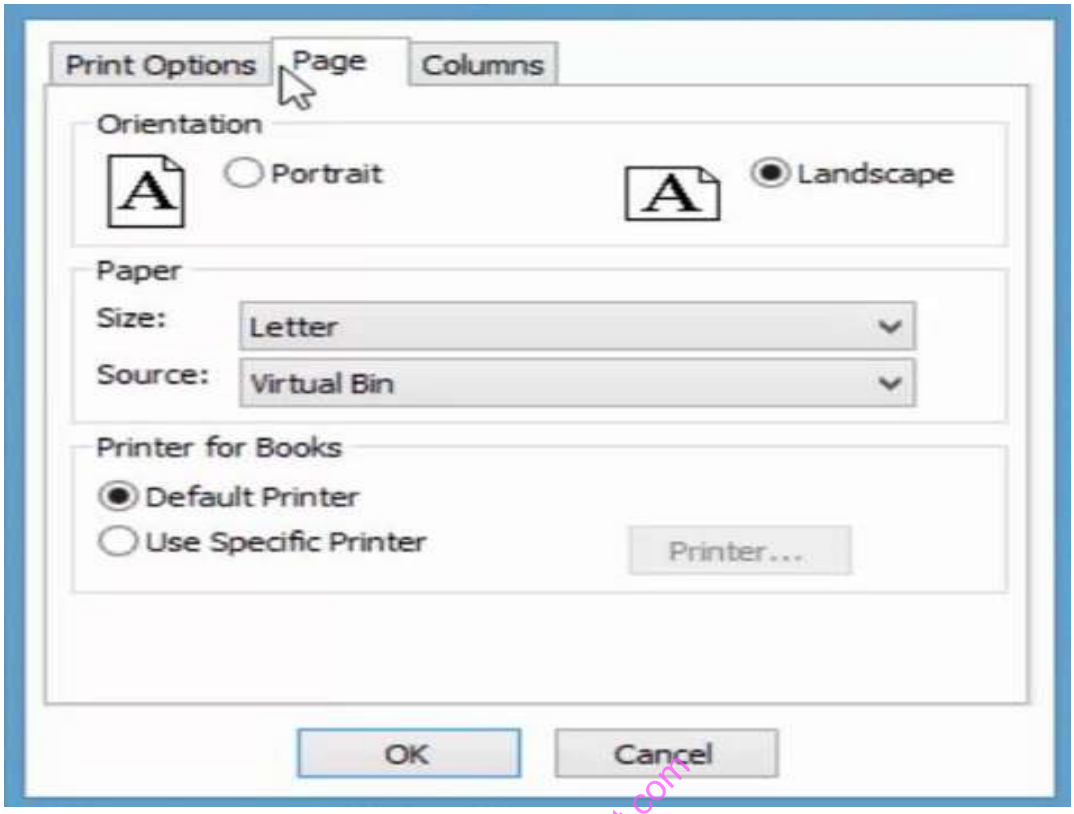


Setting	Description
Margins (inches)	Increase or decrease the default margins as required, and see the changes displayed in the **Sample** page.
Print Data Only	Select this option when you do not want to print the layout features in the report such as grids.
Split Form	This option is not available for a report.

Figure 168:Print data settings option

To customize the margins on your report, on the **Print Preview** tab, in the **Page Size** group, click **Margins** and select a suitable option and a preview of the report with the changed margin settings is displayed.

Page tab settings

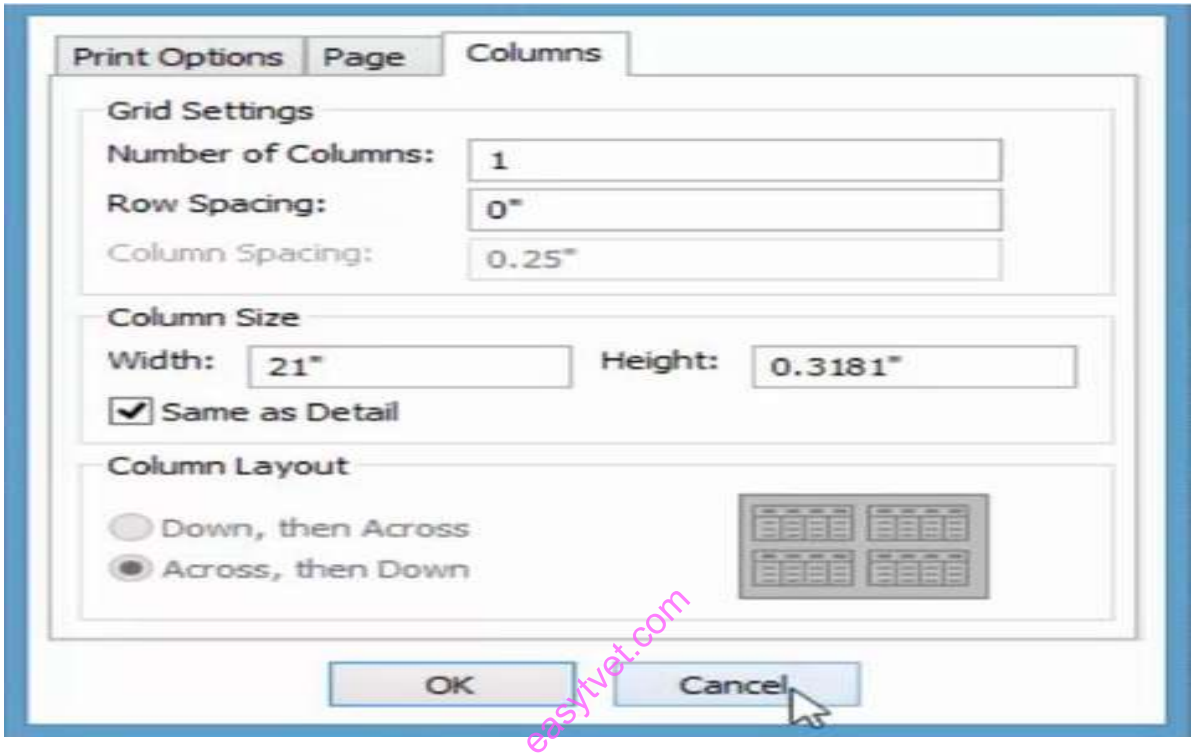


Setting	Description
Orientation	Select either the Portrait or Landscape orientation. If you want to see how the report appears, instead of setting the orientation here set it from the Page Preview tab. On the **Print Preview** tab, in the **Page Layout** group, click **Portrait** or **Landscape**.
Paper	Select the paper size and source.
Printer for Customer Addr...	If you select **Use Specific Printer** you can set several other printer and document format options.

Figure 169:Page data settings option

Columns tab settings

These options are also available on the Print Preview tab, in the Page Layout group, under Columns.



Setting	Description
Grid Settings	Select the number of columns and row spacing. If the setting is for a single column, the **Column Spacing** and **Column Layout** options won't be available.
Same as Detail	Select to print the column in the same size as you set in the report. To adjust the column width and height, clear the selection and type new values.
Column Size	Select the width and height of the columns.
Column Layout	Select the direction in which you want the columns printed. This option is available only when you have more than one column.

Figure 170: column data settings options

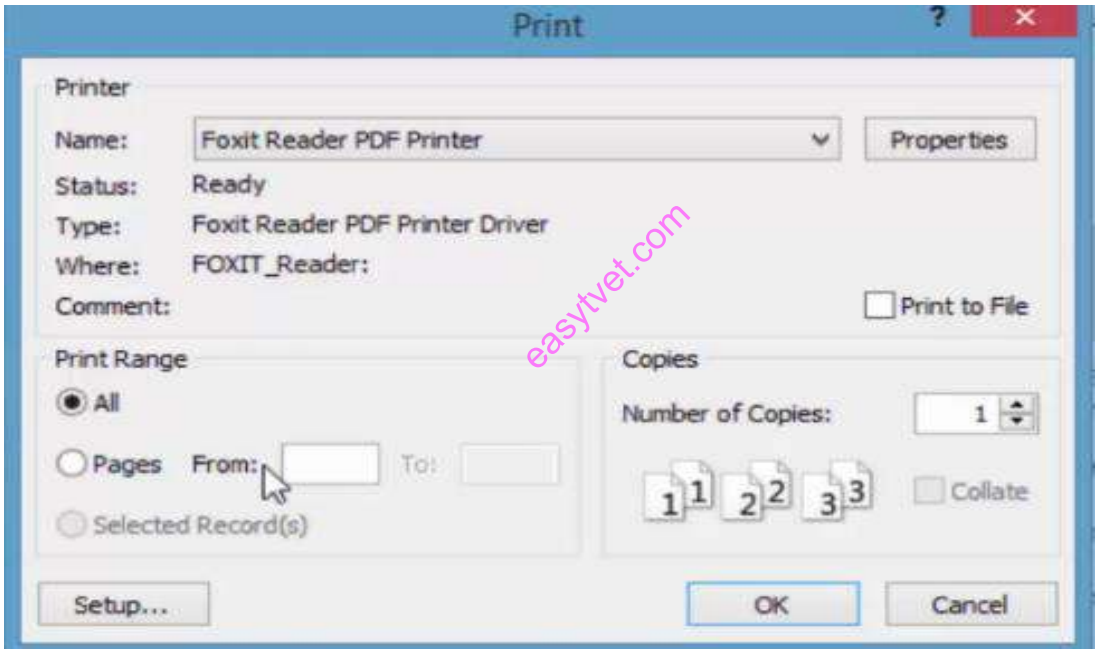
Select printer settings

After you complete the page setup settings for your report, you can select specific printer options to control the print layout or the pages you want printed and the number of copies.

1. In the Navigation Pane, right-click the report that you want to print, and click **Print Preview**.
2. On the **Print Preview** tab, in the **Print** group, click **Print**.

(Keyboard shortcut, Press **Ctrl+P**).

3. Select your print options, and then click **OK** to print your report.



The following table describes the results you can expect from specific printer settings:

Setting	Description
Printer	Select the printer where you want to have the report printed. If you haven't set the orientation and paper size, click the **Properties** button.
Print Range	▪ To print only selected pages of the report, select **Pages** and set the range. For example, to print just page 5 of a report, type **5** in both the **From** and the **To** boxes. ▪ To print just one page of your report, set both the **From** and **To** fields to the same page number.
Copies	Select the number of copies and select **Collate** if you want the copies printed in sets for a multi-page report.

Figure 171 : print data settings options

To save paper, or to send a print job to a commercial printer, or send the report via e-mail, you can create a .pdf or xps format of your report: On the **Print Preview** tab, in the **Data** group, click **PDF or XPS**.

Exporting to PDF is like printing your document, except that instead of a paper copy, you're getting a digital copy. XPS is a format similar to PDF, but created by Microsoft.
Select a report, then click the PDF or XPS button to get started. You will be asked to select a file location for the PDF.

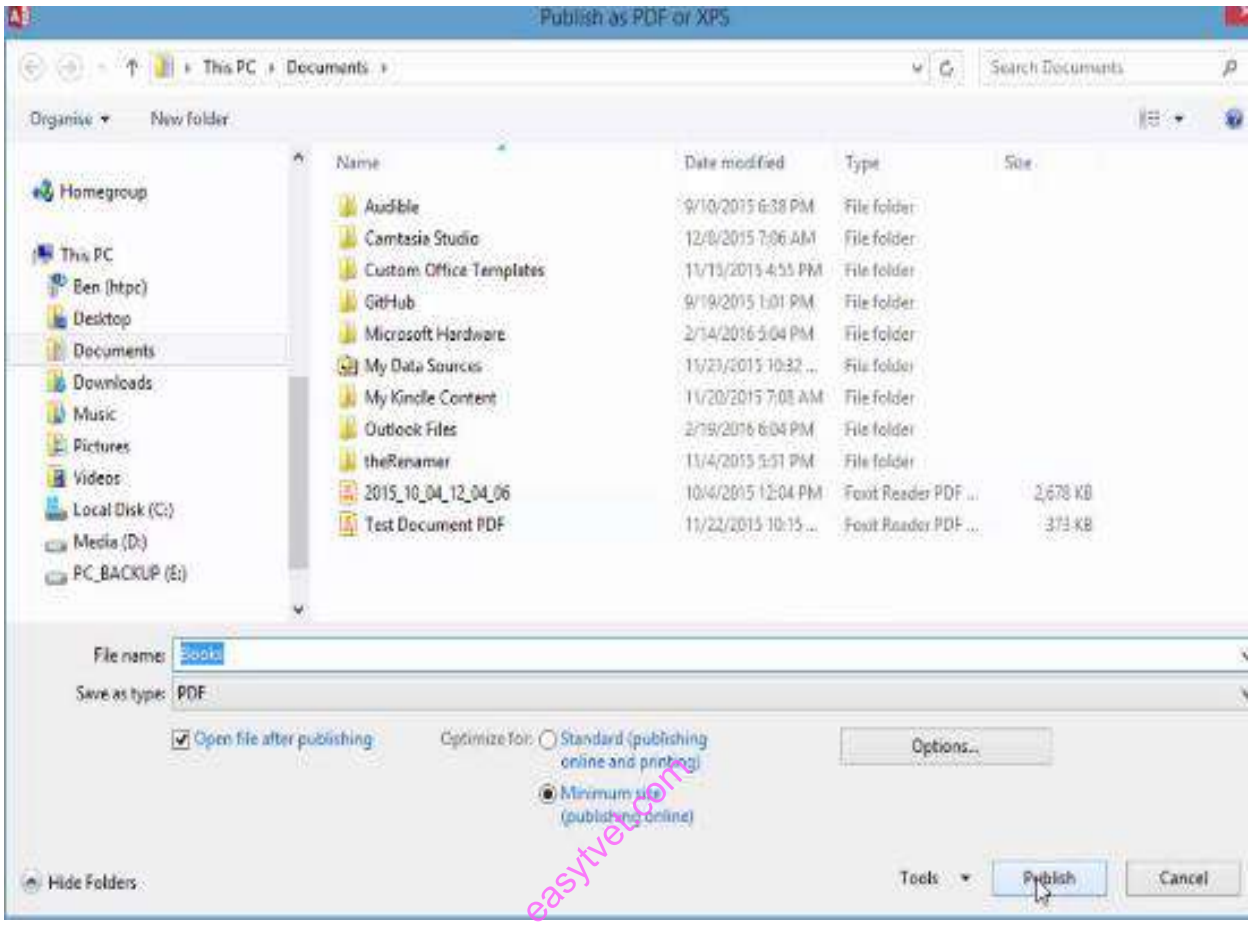


Figure 172: save settings window.

When you click OK, Access asks you if you'd like to save the export to the Saved Export Manager.

More Options

By clicking the More button in the Export group on the ribbon, you access options to save the report as a Microsoft Word file, an HTML file, and others. Since the process to export to each of these files is nearly identical, we won't repeat it here.

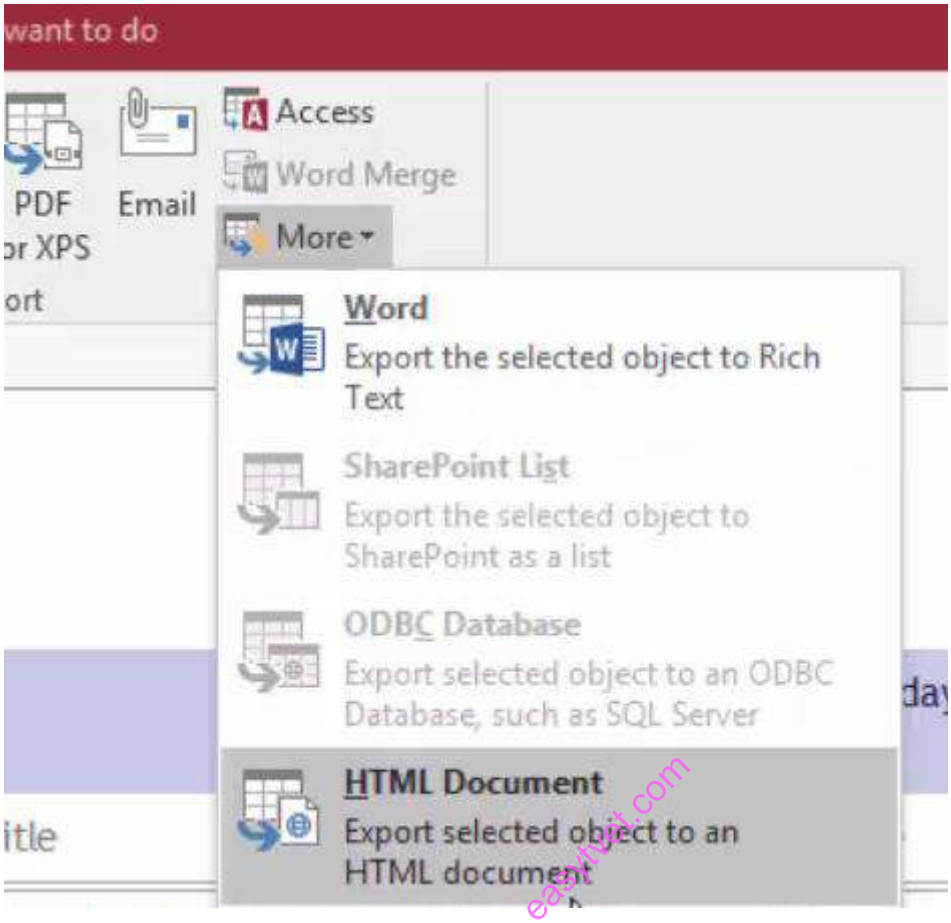


Figure 173 : more save settings options

Saved Exports

If you use the same export settings frequently, you may want to save those settings. The Saved Exports button is basically the saved export manager. As we discussed the export options, we saved some of the export settings. Right now, we're going to examine the Saved Export window.

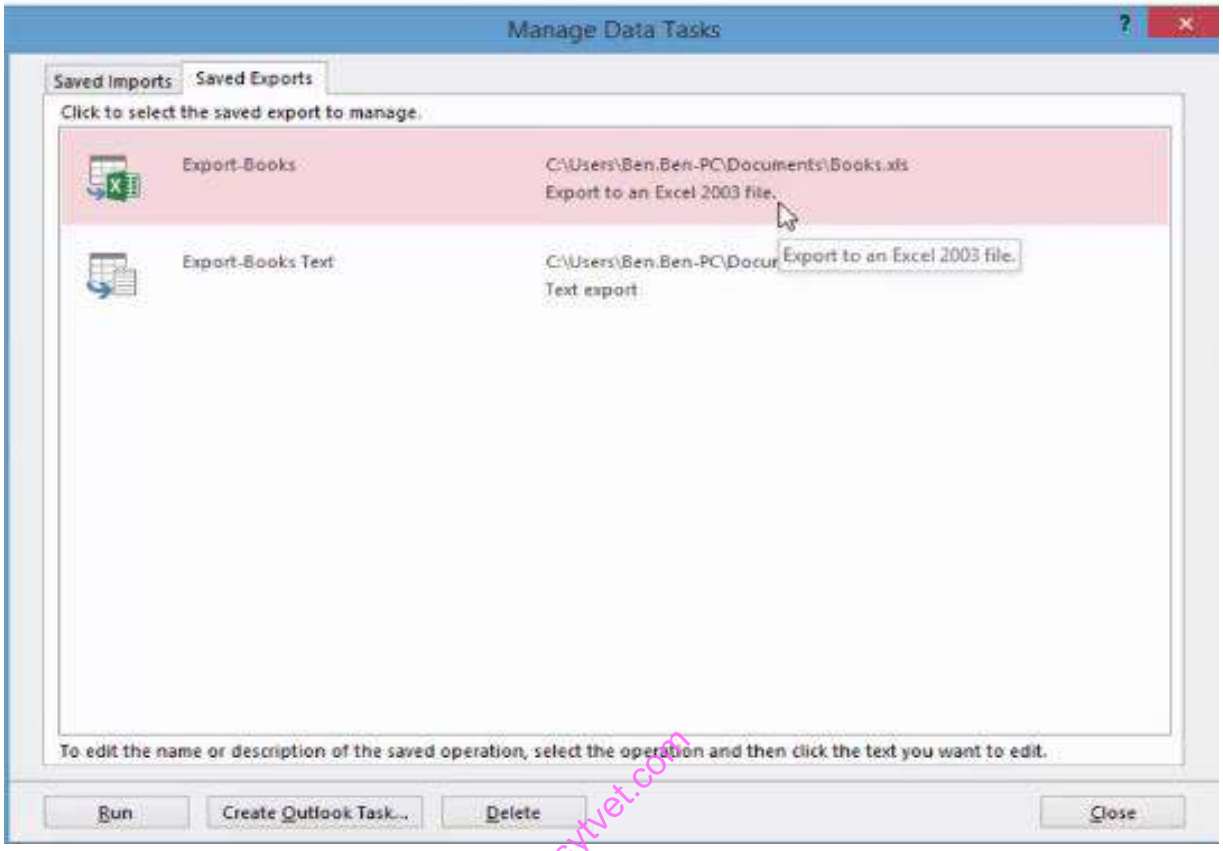


Figure 174 :export data settings options

Here we have the saved settings. One export is called "Export-Books". It has a short description--"Export data to Excel file" - and a location. We can edit any of this information by clicking on it and typing in something new.

To run a saved report, select it from the list and click Run.

To delete a report, select it and click Delete.

Preview before printing

Use the Print Preview option to make check how the printed version of your report will appear.

You can open a report in Print Preview by using one of the following methods:

- To preview a report that is not already open, in the Navigation Pane, right-click the report that you want to preview, and then click **Print Preview**.

or

- Click **File** > **Print**, and then click **Print Preview**.

or

- If the report is open, right-click the document tab for the report, and then click **Print Preview**.

You can make further formatting changes from Print Preview by using the options in the **Preview Print** tab.

Figure 175 : Print preview settings options

If you want to see several pages of the report, move to other pages, or view multiple pages at the same time when previewing you report, try any the following options:

- Click the **Shutter Bar Open/Close** button (<<) in the upper-right corner of the Navigation Pane, and then expand the report view area.
- Position the report by using the horizontal or vertical scroll bars.
- To magnify an area on the report, on the **Print Preview** tab, in the **Zoom** group, click the arrow below **Zoom** and select a magnification percentage. You can also use the Zoom control in the lower right-hand corner of your window.
- To preview multiple pages at a time, in the **Zoom** group, click **Two Pages** or click **More Pages** and then select an option.

Select your print options: In the **Print** group, click **Print** and then select your print options.

Close Print Preview: In the **Close Preview** group, click **Close Print Preview**.

Microsoft Access can also help you prepare printed documentation of your database's design, which you can use for offline reference and planning. Specifically, Access can help you print table relationships as depicted in the Relationship window, and can help you print the design characteristics of database objects, such as object properties.

Prepare the Relationships window for printing

The Relationship window is a window in which you view, create, and edit relationships between tables and queries. Depending on how you configure it, the Relationships window does not always show all the relationships in your database. You should decide which relationships you want to document — and prepare the Relationships window accordingly — before you print table relationships. You can show or hide tables in the Relationships window, and you can rearrange their layout.

Show or hide a table in the Relationships window

1. On the **Database Tools** tab click **Relationships**.
2. Do one of the following:
    - To show a table, on the **Design** tab, in the **Relationships** group, click **Show Table**, and then in the **Show Table** dialog box, add the table that you want.
    - To hide a table, in the Relationship window, click the table that you want to hide, and then press DELETE.
3. Press CTRL+S to save your changes.

Print the Table Relationships window

You can create a report that displays the relationships in your database as they appear in the Relationships window. This report has the default name **Relationships for database name**. By default, the report's header contains the name and the date that you created the report.

1. Open the Access database that has relationships you want to print.
2. On the **Database Tools** tab click **Relationships**.
3. On the **Design** tab, in the **Tools** group, click **Relationship Report**.

(A report that depicts the Relationships window opens in Print Preview mode.)

4. To make any adjustments to the way your report will print, use commands on the **Print Preview** tab.

5. On the **Print Preview** tab, in the **Print** group, click **Print**.

Print the design characteristics of database objects

You can use the Database Documenter to print the design characteristics of database objects.

1. Open the database that you want to document.
2. On the **Database Tools** tab, in the **Analyze** group, click **Database Documenter**.
3. In the **Documenter** dialog box, click the tab that represents the type of database object that you want to document. To create a report on all of the objects in a database, click the **All Object Types** tab.
4. Select one or more of the objects listed on the tab. To select all of the objects on a tab, click **Select All**.
5. Click **OK**.

(The Database Documenter creates a report that contains detailed data for each selected object, and then opens the report in Print Preview. For example, if you run the Database Documenter against a data entry form, the report created by the Documenter lists the properties for the form as a whole, the properties for each of the sections in the form, and the properties for any buttons, labels, text boxes, and other controls on the form, plus any code modules and user permissions that are associated with the form.)

6. To print the report, on the **Print Preview** tab, in the **Print** group, click **Print**.

Preview a report.

When you want to see how your report will appear when printed, use Print Preview. If you preview a report that was created in Access, the record navigation buttons appear in the default location, at the bottom of the Access window.

Start Print Preview

1. If the report is not open, in the Navigation Pane, right-click the report that you want to preview, and then click **Print Preview** on the shortcut menu.

-or-

If the report is open, right-click the document tab for the report and click **Print Preview**.

2. Click the record navigation buttons to move among the pages in the report.

Preview multiple pages.

If the report is open, right-click the document tab for the report and click **Print Preview**.

1. On the **Print Preview** tab, in the **Zoom** group, click **Two Pages** to preview your report two pages at a time.

-or-

In the **Zoom** group, click **More Pages**, and then select an option.

2. Click the record navigation buttons to move among the pages in the report.
3. To revert to previewing a single page, in the **Zoom** group, click **One Page**.

Close Print Preview

You can either print the report after previewing it or close the preview if you want to make changes to the report before printing it.

- On the **Print Preview** tab, in the **Close Preview** group, click **Close Preview**.

Print a report quickly.

If you want to print all the pages in the report without making any changes to the format of the report such as margins, headers, footers, and orientation (portrait or landscape) you can use the Quick Print option.

1. If the report is not open, in the Navigation Pane, right-click the report and click **Print**.

-or-

Make sure that the **Quick Print** button is available in the Quick Access Toolbar. To add the button to the toolbar, click the down-arrow at the end of the toolbar and click **Quick Print**.
On the **Quick Access Toolbar**, click **Quick Print**.

Print selects pages from a report

In a report, you can limit what you print only by selecting a range of pages.

1. Click **File** and then click **Print**.
2. On the **Print** area of the File options, click the **Print** option to open the **Print** dialog box.
3. To print only some of the data in the report, under **Print Range**, click **Pages**, and then enter a value in the **From** and the **To** boxes.

For example, to print just page 5 of a report, type **5** in both the **From** and the **To** boxes.

If you own the necessary permissions to change the design of a report, you can make additional changes to the way that your report prints by changing the print layout in the **Page Setup** dialog box (click **Setup** to display the **Page Setup** dialog box).

Change the page setup.

The **Page Setup** dialog box can be used to change a variety of print options, including the top and bottom margins and whether Access prints just the data in your report. Access provides two ways to display the **Page Setup** dialog box. You can start it from within the **Print** dialog box, or you can start it from the **Print Preview** tab, which appears when you start Print Preview.

The steps in this section explain how to display the **Page Setup** dialog box.

Start the Page Setup dialog box from the Print dialog box

1. In the Navigation Pane, select the report that you want to change. You do not need to open the report, although you can if you need to ensure that the data you are about to print is current.
2. Click **File** and then click **Print**.
3. On the **Print** area of the File options, click the **Print** option to open the **Print** dialog box.
4. Click **Setup** to display the **Page Setup** dialog box.
5. Set or change the options in the dialog box, as needed.

Start the Page Setup dialog box from the Print Preview tab

2. In the Navigation Pane, right-click the report that you want to change and click **Print Preview.**

Access opens the report in Print Preview.

3. On the **Print Preview** tab, in the **Page Layout** group, click the **Page Setup** button.
4. Set or change the options in the dialog box, as needed.
5. Select print options

Follow the steps in this section when you need to print some of the pages in a report or when you need to select a printer or change print settings, such as margins or columns.

4. In the Navigation Pane, select the report that you want to print.
5. Click **File** and then click **Print**.
6. On the **Print** area of the File options, click the **Print** option to open the **Print** dialog box.
7. Change the page format setup or other print options or printer selection as needed, and then click **OK** to print your report.

5.2.5.4 Learning Activities

From a sample database of your choice, print out the following elements of the database and save them as PDF.

a. Table relationship
b. Form
c. Reports
d. Queries

5.2.5.5 Self-Assessment

ii. Describe the procedure of printing a report in Ms Access.
iii. Describe the procedure of Printing Table Relationships window

5.2.5.6 Tools, Equipment, Supplies and Materials

- Computer
- Database software (Ms Access)
- Printer
- stationery

5.2.5.7 References

Silberschatz A. and et al (2001), Database System Concepts, McGraw-Hill.

Garcia-Molina H. and et al (2000), Database System: The complete Book, Prentice Hall.

Elmasri R. and Navathe S(2000), Fundamentals of Database System, Addison Wesley.

Conolly& Begg (2010), Database System: a practical approach to design, implementation and management, Prentice Hall.

Ullman, and Jennifer Widom (2009). *Database Systems: The Complete Book (2nd edition)* by SBN 978-0131873254, Pearson Prentice Hall,

Jeffrey A. Hoffer, Mary B. Prescott, and Fred R. McFadden(2007). Modern Database Management (8th Ed.). Prentice-Hall.

Roger Jennings (2004) Special Edition Using Microsoft Office Access 2003, Que.

Alan Simpson, Margaret Levine Young, and Alison Barrows (2003) Access 2003 for DUMMIES, Wiley Publishing.

5.2.5.8 Model answers to self-assessment

i. Describe the procedure of printing a report in Ms Access

In a report, you can limit what you print only by selecting a range of pages.

1. Click **File** and then click **Print**.
2. On the **Print** area of the File options, click the **Print** option to open the **Print** dialog box.
3. To print only some of the data in the report, under **Print Range**, click **Pages**, and then enter a value in the **From** and the **To** boxes.

For example, to print just page 5 of a report, type **5** in both the **From** and the **To** boxes.

If you own the necessary permissions to change the design of a report, you can make additional changes to the way that your report prints by changing the print layout in the **Page Setup** dialog box (click **Setup** to display the **Page Setup** dialog box).

ii. Describe the procedure of Printing a Table Relationships window

You can create a report that displays the relationships in your database as they appear in the Relationships window. This report has the default name **Relationships for database name**. By default, the report's header contains the name and the date that you created the report.

1. Open the Access database that has relationships you want to print.

2. On the **Database Tools** tab click **Relationships**.
3. On the **Design** tab, in the **Tools** group, click **Relationship Report**.

(A report that depicts the Relationships window opens in Print Preview mode.)

4. To make any adjustments to the way your report will print, use commands on the **Print Preview** tab.
5. On the **Print Preview** tab, in the **Print** group, click **Print**.

Print the design characteristics of database objects