

University of Baghdad
Collage of Ibn-Al-Haitham for
Pure Science
Department of Computer Science



جامعة بغداد

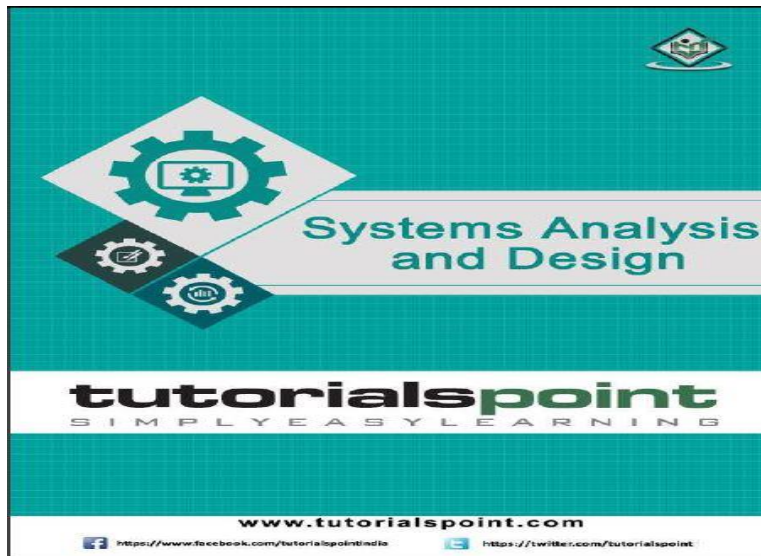
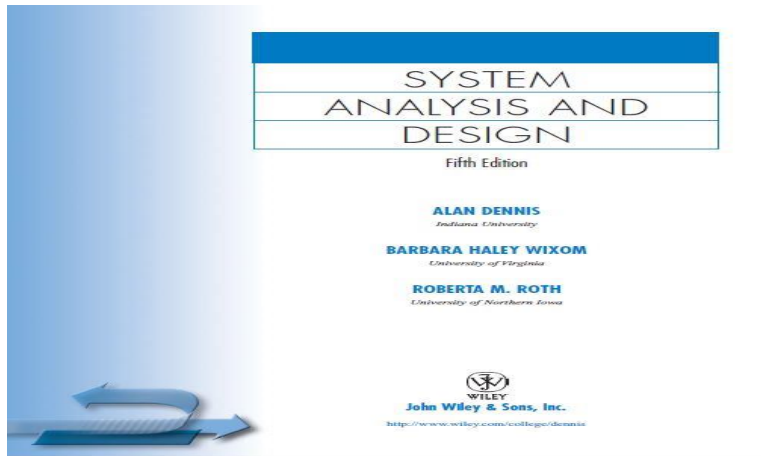
كلية التربية للعلوم الصرفة / ابن الهيثم

قسم علوم الحاسبات

System Analysis & Database

Dr Ahmed Aljuboori

References



DATABASE SYSTEMS The Complete Book Second Edition

Hector Garcia-Molina
Jeffrey D. Ullman
Jennifer Widom
*Department of Computer Science
Stanford University*

PEARSON
Prentice
Hall
Upper Saddle River, New Jersey 07458

التسجيل على حساب في المنصة التعليمية Edmodo

يجب على كافة الطلبة الالتزام بانشاء حساب على المنصة التعليمية ادمودو من اجل ادامة العملية التعليمية مع استاذ المادة من خلال مثلا استلام الملازم المقررة للعام الدارسي، ارسال الواجبات او التقارير المطلوبة ضمن المادة الدراسية، استلام نتائج الامتحانات اليومية، ارسال اي استفسارات تتعلق بالمادة الدراسية، هذا بالإضافة الى الكثير من المهام والتبليغات الممكن انجازها عبر المنصة التعليمية.

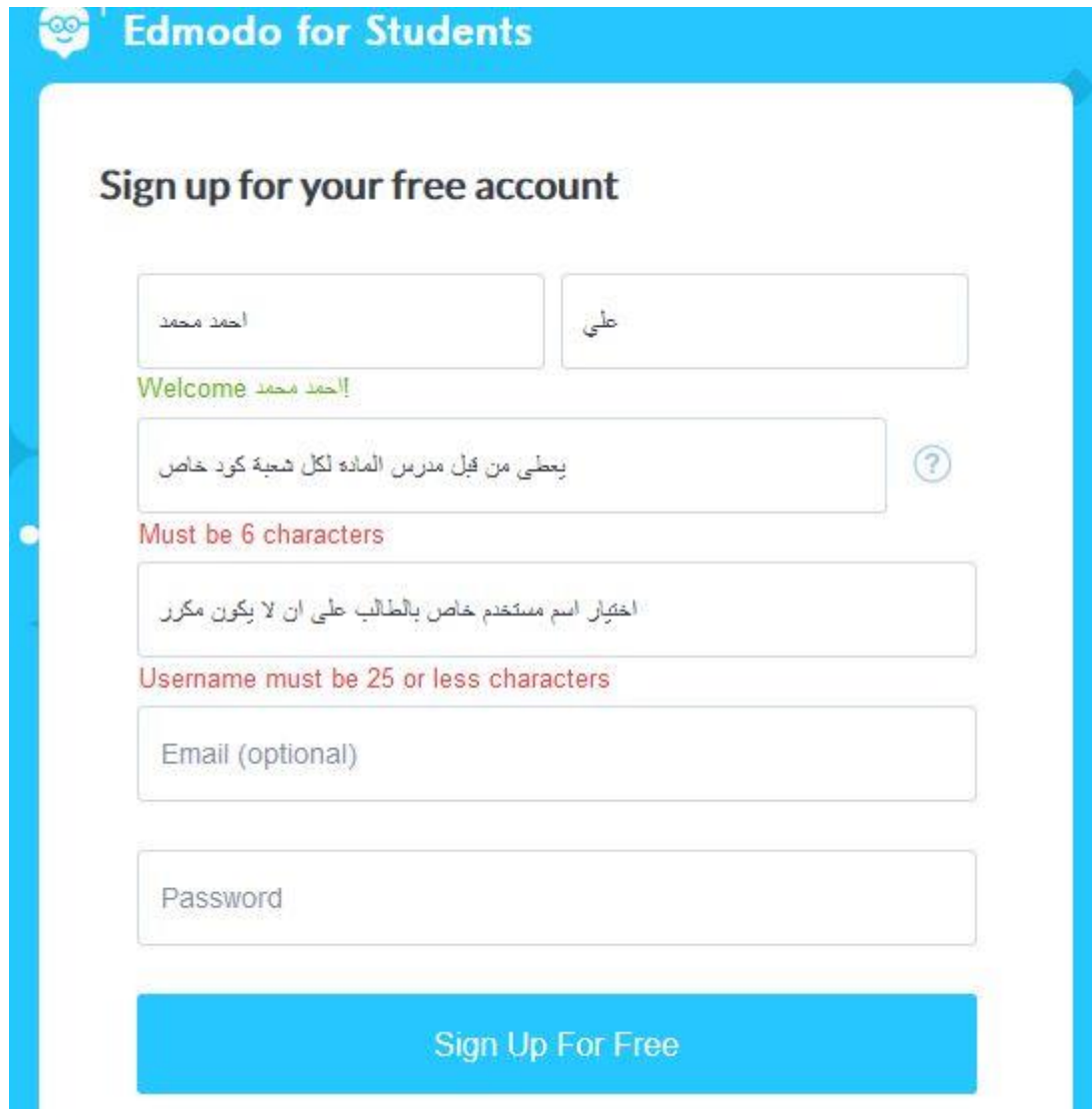
تعتبر المنصة التعليمية وسيلة اتصال رسمية بين الطلبة واستاذ مادة تحليل الأنظمة وتصميم قواعد البيانات لذلك يتوجب على جميع الطلبة انشاء حساب حسب التوجيهات التالية:

- انشاء حساب على موقع Edmodo من خلال <https://www.edmodo.com/>
- التسجيل كطالب عن طريق الضغط على I'm a student عن طريق الرابط https://www.edmodo.com/student-signup?source=landing_page&channel=landing_page
- وضع اسم الطالب واسم الاب في خانة الاسم الاول مثلا كتابة احمد محمد
- وضع اسم الجد في خانة الاسم الاخير مثلا كتابة الاسم علي
- في مكان كتابة الكود : سيتم تزويد كل شعبة بكود خاص للالتحاق بالفصل الدراسي بمعنى ان الكود الخاص للصباحي ا يختلف عن ب ويختلف عن المسائي (ارجو الانتباه).
- اختيار كلمة مرور مناسبة مع الحرص على عدم نسيانها، هذا بالإضافة الى الاستمرار بملئ باقي الحقول المطلوبة لاتمام انشاء البريد الالكتروني.

الرابط ادناه فيديو لتوضيح عملية التسجيل

<https://www.youtube.com/watch?v=5fWuZQVT3BA>

في حال التسجيل في حساب ادمودو خلال الاسبوع الاول من الدوام فسوف يحرم الطالب من الامتحانات المصاحبة لهذه الخدمة وسيكون مسؤولاً عن ذلك .



Edmodo for Students

Sign up for your free account

Welcome احمد محمد!

Must be 6 characters

Username must be 25 or less characters

Sign Up For Free

مثال حول كيفية ملئ البيانات الخاصة للتسجيل في المنصة التعليمية ادمودو للتواصل مع مدرس مادة تحليل الانظمة و قواعد البيانات.

Database System

Introduction

Database and database technology have major impact on the growing use of computer. It is fair to say that database play a critical role in almost all areas where computer are used, including business, electronic commerce, engineering, law, education, and library science. The word *database* is so commonly used that we must begin by defining what a database is.

تكنولوجيا قاعدة البيانات وقاعدة البيانات لها تأثير كبير على الاستخدام المتزايد للكمبيوتر. من الجدير بالقول ان قاعدة البيانات تلعب دوراً حاسماً في جميع المجالات تقريباً التي يتم فيها استخدام الكمبيوتر ، بما في ذلك الأعمال التجارية ، والتجارة الإلكترونية ، والهندسة ، والقانون ، والتعليم ، وعلوم المكتبات. كلمة قاعدة البيانات شائعة الاستخدام لدرجة أننا يجب أن نبدأ بتحديد قاعدة البيانات.

➤ **Database (DB):** Is a collection of interrelated data stored together without harmful or unnecessary redundancy to serve multiple applications, the data stored so that they are independent of the applications which use it.

هي مجموعة من البيانات المترابطة المخزنة معا دون التكرار الضار أو غير الضروري لخدمة تطبيقات متعددة، والبيانات المخزنة بحيث تكون مستقلة عن التطبيقات التي تستخدمها.

قاعدة البيانات العلائقية هي مجموعة من الجداول الموصوفة رسمياً والتي يمكن من خلالها الوصول إلى البيانات أو إعادة تجميعها بالعديد من الطرق المختلفة دون الحاجة إلى إعادة تنظيم جداول قاعدة البيانات.

يتم استخدام Access او عبارات SQL للاستعلامات التفاعلية للحصول على معلومات من قاعدة بيانات علائقية وجمع البيانات للتقارير.

Database Design and Data Redundancy

Every organization needs to manage data about entities. An **entity** is a person, place, object, event or idea about which an organization stores data. Colleges maintain records regarding students, courses, faculty, equipment, and grades.

Businesses need to keep track of customers, sales, employees, inventory and payments. A database management program is used to store, maintain and present data in an efficient and structured fashion. The elements of this structure are illustrated in **Figure 1** and are described in the bulleted paragraphs below.

- A **field** is an **attribute** of an entity. You may think of a field as a category of data. There are 8 fields in the Customers table in Figure 1, including First, Last and Birth Date.
 - A **record** is a collection of related fields, each pertaining to the same entity. The bottom row in the Customers table is the collection of fields for customer Scott Owen. There are 17 records in the Customers table.
 - A **table** is a collection of related records, each having the same field structure. As you see in Figure 1, a table is a two-dimensional structure where the columns represent fields and the rows represent records.
- Tables are used to store data.
- A **database** is a collection of related tables and other objects.

CustID	First	Last	Street	City	State	Zip	Birth Date
1	Ginny	Braithwaite	3 Which Way	Salem	MA	01970	1/10/60
10	Virginia	Rodarmor	123 Main Street	Andover	MA	01810	1/6/70
11	Kristen	Reis	4848 Ashley	Fontanelle	IA	50810	3/18/68
12	Tom	Reis	4848 Ashley	Fontanelle	IA	50810	7/3/65
13	Mark	Eagan	987 Lincoln	Schaumburg	IL	44433	1/29/60
14	Peg	Fox	125 Maple	Des Moines	IA	50625	4/10/59
15	Ron	Fox	125 Maple	Des Moines	IA	50625	8/28/87
16	Amanda	Fox	125 Maple	Des Moines	IA	50625	1/30/88
17	Rebecca	Gross	123 Oak	Bridgewater	KS	50837	9/20/62
2	Robin	Spencer	293 Serenity Dr.	Concord	MA	01742	1/30/52
3	Camilla	Dobbins	486 Intel Circuit	Rio Rancho	NM	87124	3/15/65
4	Pip	Khalsa	1100 Vista Road	Santa Fe	NM	87505	4/16/69
5	Kendra	Majors	530 Spring Street	Lenox	MA	02140	5/4/70
6	Tasha	Williams	530 Spring Street	Lenox	MA	02140	5/8/71
7	Fred	Gonzales	Purgatory Ski Area	Durango	CO	81301	6/10/60
8	John	Black	11 River Road	Brookfield	CT	06830	7/15/65
9	Scott	Owen	72 Yankee Way	Brookfield	CT	06830	3/15/65

CustID	InvoiceNo	InvoiceDate	TourID
1	1	5/1/96	Road0696
2	2	8/1/96	Mtbi0996
3	3	8/2/96	Bung0996
4	4	8/2/96	Mtbi0996
5	5	8/2/96	Bung0996
6	6	8/2/96	Bung0996
7	7	8/2/96	Mtbi0996
8	8	8/5/96	Mtbi0996
9	9	8/5/96	Bung0996
10	10	8/5/96	Mtbi0996
11	11	12/2/96	Big0197
12	12	12/2/96	Big0197
13	13	12/3/96	Big0197
14	14	12/3/96	Big0197
15	15	12/4/96	Big0197
16	16	12/4/96	Big0197
11	17	2/3/97	Bung0997
13	18	2/3/97	Bung0997
14	19	2/4/97	Mtbi0997
15	20	2/4/97	Mtbi0997

Figure 1

Figure 1 Customer & Sales Tables

The preceding discussion focused on a properly designed database, where each entity is placed in a separate table and where a common field is available to join related records between the tables. We now explore the perils of an improper database design.

Take a few moments to inspect **Figure 3**. Notice that all of the fields now appear in a single table. Although this may seem simpler, it turns out to be a terrible idea.

Look at the **third** and **fourth** rows in **Figure 3**. These two rows reflect the sales made to **Kristin Reis**. But look more closely and you'll notice that Kristin's information (**First, Last, Street, City, State, Zip and Birth Date**) is stored **twice** because she has been the customer in two different sales. If she had been the customer in **50 sales**, this design would require that we store her name, address and birth date **50 different** times! Storing the same field values more than once (unnecessarily) are referred to as data redundancy.

Three problems are caused by data redundancy. The **first** is that **storing values multiple times wastes space**. Under a proper design (Figure 1), Kristin's information is stored only once, in her record in the Customers table.

The second problem is that when a field **value changes**, multiple occurrences need to be **updated**. For example, if Kristin moves, we'll need to change the values for her Street, City, State and Zip in multiple records. **The third** problem occurs if we forget to change the values in any of the records. The database would then have **inconsistent** data.

Redundant : Table										
InvoiceNo	InvoiceDate	TourID	CustID	First	Last	Street	City	State	Zip	Birth Date
1	5/1/96	Road0696	1	Ginny	Braithwaite	3 Which Way	Salem	MA	01970	1/10/60
10	8/5/96	Mtbi0996	10	Virginia	Rodarmor	123 Main Street	Andover	MA	01810	1/6/70
11	12/2/96	Bigs0197	11	Kristen	Reis	4848 Ashley	Fontanelle	IA	50810	3/18/68
17	2/3/97	Bung0997	11	Kristen	Reis	4848 Ashley	Fontanelle	IA	50810	3/18/68
12	12/2/96	Bigs0197	12	Tom	Reis	4848 Ashley	Fontanelle	IA	50810	7/3/65
18	2/3/97	Bung0997	13	Mark	Eagan	987 Lincoln	Schaumber	IL	44433	1/29/60
13	12/3/96	Bigs0197	13	Mark	Eagan	987 Lincoln	Schaumber	IL	44433	1/29/60
19	2/4/97	Mtbi0997	14	Peg	Fox	125 Maple	Des Moines	IA	50625	4/10/59
14	12/3/96	Bigs0197	14	Peg	Fox	125 Maple	Des Moines	IA	50625	4/10/59
20	2/4/97	Mtbi0997	15	Ron	Fox	125 Maple	Des Moines	IA	50625	8/28/87
15	12/4/96	Bigs0197	15	Ron	Fox	125 Maple	Des Moines	IA	50625	8/28/87
16	12/4/96	Bigs0197	16	Amanda	Fox	125 Maple	Des Moines	IA	50625	1/30/88
2	8/1/96	Mtbi0996	2	Robin	Spencer	293 Serenity Dr.	Concord	MA	01742	1/30/52
3	8/2/96	Bung0996	3	Camilla	Dobbins	486 Intel Circuit	Rio Ranch	NM	87124	3/15/65
4	8/2/96	Mtbi0996	4	Pip	Khalsa	1100 Vista Roac	Santa Fe	NM	87505	4/16/69
5	8/2/96	Bung0996	5	Kendra	Majors	530 Spring Stret	Lenox	MA	02140	5/4/70
6	8/2/96	Bung0996	6	Tasha	Williams	530 Spring Stret	Lenox	MA	02140	5/8/71
7	8/2/96	Mtbi0996	7	Fred	Gonzales	Purgatory Ski Ar	Durango	CO	81301	6/10/60
8	8/5/96	Mtbi0996	8	John	Black	11 River Road	Brookfield	CT	06830	7/15/65
9	8/5/96	Bung0996	9	Scott	Owen	72 Yankee Way	Brookfield	CT	06830	3/15/65

Figure 3

Figure 2 Redundant Table

Purpose of Database Systems: (Why DB?)

The goal of a database system is to simplify and facilitate access to data from the centralized control there are some advantages such as:

1- Redundancy can be reduced:

This data repetition may occur either if a field is repeated in two or more tables or if the field is repeated within the table. This can mean two different fields within a single database. Data redundancy is the repetition of data.

قد يحدث تكرار البيانات هذا إما إذا تم تكرار حقل في جدولين أو أكثر أو إذا تم تكرار الحقل داخل الجدول. هذا يمكن أن يعني حقلين مختلفين في قاعدة بيانات واحدة. فائض البيانات هو تكرار البيانات.

2- Inconsistency can be avoided:

If we have redundant data in one table, at such time the DB is said to be inconsistent.

The problem occurs if we forget to change the values in any of the records. The database would then have inconsistent data.

إذا كان لدينا بيانات مكررة في جدول واحد ، في وقت ما يقال أن DB هي غير متناسقة . تحدث المشكلة إذا نسينا تغيير القيم في أي من السجلات . عندئذ تكون قاعدة البيانات تحتوي على بيانات غير متسقة.

3- The data can be shared:

We mean that individual pieces of data in the DB may be shared among several different users and applications, in that each of those users may have access to the same piece of data.

4- Standards can be enforced:

Standards are common practices that ensure the consistency and effectiveness of the database environment, such as database naming conventions. Procedures are defined, step-by-step instructions that direct the processes required for handling specific events, such as a disaster recovery plan. Failure to implement database standards and procedures

will result in a database environment that is confusing and difficult to manage. For example (communication, administration,)’ standards.

المعايير يمكن ان تفرض المعايير هي ممارسات شائعة تضمن تناسق وفعالية بيئة قاعدة البيانات ، مثل اصطلاحات قواعد البيانات .يتم تعريف الإجراءات والتعليمات خطوة بخطوة التي توجه العمليات المطلوبة لمعالجة أحداث معينة ، مثل خطة استرداد الحالات المستعصية .سيؤدي عدم تنفيذ معايير وإجراءات قاعدة البيانات إلى بيئة قاعدة بيانات مربكة ويصعب إدارتها.

5- Security restriction can be applied:

- The **DBA (Database Admin)** can ensure that the only means of access to the **DB** is through the proper channels.
- The DBA can define security checks to be carried out whenever access is attempted to sensitive data.

تطبيق قيود الأمان يمكن أن يضمن ال **DBA** أن الوسيلة الوحيدة للوصول إلى **DB** هي من خلال القنوات المناسبة .

يمكن لل **DBA** تحديد عمليات الفحص الأمنية التي يتم تنفيذها عند محاولة الوصول إلى بيانات حساسة.

6- Data Integrity

The following categories of data integrity exist with each RDBMS –

- **Entity Integrity** – There are no duplicate rows in a table.
- **Domain Integrity** – Enforces valid entries for a given column by restricting the type, the format, or the range of values.
- **Referential integrity** – Rows cannot be deleted, which are used by other records.

تكمال الكيان - لا توجد صفوف مكررة في جدول.
تكمال المجال - يكون بفرض إدخالات صالحة لعمود معين عن طريق تقييد النوع أو التنسيق أو نطاق القيم.
التكامل المرجعي - لا يمكن حذف الصفوف ، والتي يتم استخدامها بواسطة السجلات الأخرى.

7- Data Indpendence:

It refers to the immunity of user applications to changes made in the definition and organization of data. Physical data independence deals with hiding the details of the storage structure from user applications.

استقلالية البيانات ويشير إلى حصانة تطبيقات المستخدمين للتغيرات في تعريف وتنظيم البيانات. يتعامل استقلال البيانات المادية مع إخفاء تفاصيل بنية التخزين من تطبيقات المستخدم.

Database Administrator DBA

DBA is a person or persons responsible for the *structure* or *schema* of the DB also responsible for *authorizing software access* and *hardware resource* as needed.

هو شخص أو مجموعة أشخاص مسؤولون عن بنية أو مخطط DB ويكون أيضا مسؤول عن تفويض الوصول إلى البرامج وموارد الأجهزة حسب الحاجة.

Another definition of DBA

DBA: is a person responsible for the performance, integrity and security of a database. He will also be involved in the planning and developing of the DB, as well as troubleshooting any issues on behalf of the users.

Database schema (Data Abstraction)

Database Schema

A database schema is the skeleton structure that represents the logical view of the entire database. It defines how the data is organized and how the relations among them are associated. It formulates all the constraints that are to be applied on the data.

A database schema defines its entities and the relationship among them. It contains a descriptive detail of the database, which can be depicted by means of schema diagrams. It's the database designers who design the schema to help programmers understand the database and make it useful.

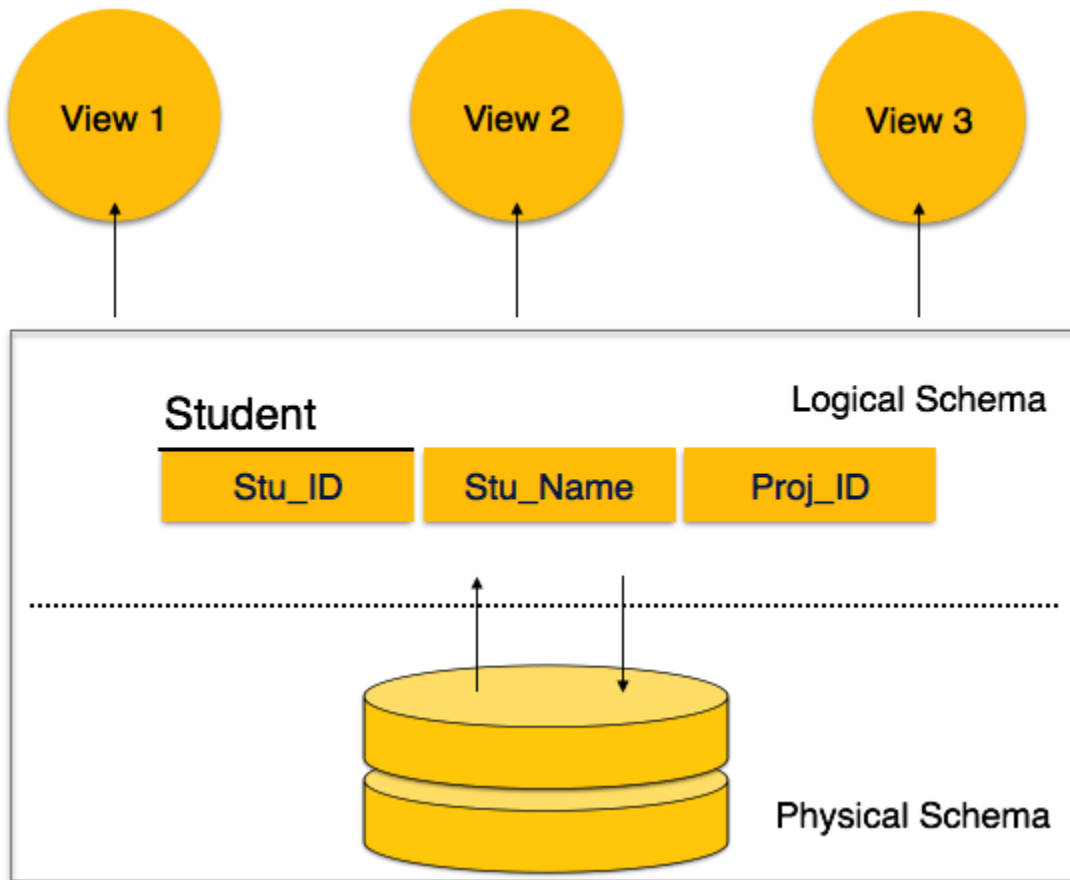


Figure 3 Database schema Division

A database schema can be divided broadly into two categories –

- **Physical Database Schema** – This schema relates to the actual storage of data and its form of storage like files, indices, etc. It defines how the data will be stored in a secondary storage.
- **Logical Database Schema** – This schema defines all the logical constraints that need to be applied on the data stored. It defines tables, views, and integrity constraints.

Database Instance

It is important that we distinguish these two terms individually. Database schema is the skeleton of database. It is designed when the database doesn't exist at all. Once the database is operational, it is very difficult to make any changes to it. A database schema does not contain any data or information.

A database instance is a state of operational database with data at any given time. It contains a snapshot of the database. Database instances tend to change with time. A DBMS ensures that its every instance (state) is in a valid state, by diligently following all the validations, constraints, and conditions that the database designers have imposed.

Instance: The data stored in database at a particular moment of time is called instance of database. Database schema defines the variable declarations in tables that belong to a particular database; the value of these variables at a moment of time is called the instance of that database.

تسمى البيانات المخزنة في قاعدة البيانات في لحظة معينة من الوقت باسم قاعدة البيانات. يعرف مخطط قاعدة البيانات تعريفات المتغيرات في الجداول التي تنتمي إلى قاعدة بيانات معينة؛ تسمى قيمة هذه المتغيرات في لحظة من الزمن باسم قاعدة البيانات تلك.

Database Management System (DBMS Overview)

Database is a collection of related data and data is a collection of facts and figures that can be processed to produce information.

Mostly data represents recordable facts. Data aids in producing information, which is based on facts. For example, if we have data about marks obtained by all students, we can then conclude about toppers and average marks.

Database Management System (DBMS) is system software for creating and managing databases. The DBMS provides users and programmers with a systematic way to create, retrieve, update and manage data.

Characteristics

Traditionally, data was organized in file formats. **DBMS** was a new concept then, and all the research was done to make it overcome the deficiencies in traditional style of data management. A modern DBMS has the following characteristics –

- **Real-world entity** – A modern DBMS is more realistic and uses real-world entities to design its architecture. It uses the behaviour and attributes too. For example, a school database may use students as an entity and their age as an attribute.

- **Relation-based tables** – DBMS allows entities and relations among them to form tables. A user can understand the architecture of a database just by looking at the table names.
- **Isolation of data and application** – A database system is entirely different than its data. A database is an active entity, whereas data is said to be passive, on which the database works and organizes. DBMS also stores metadata, which is data about data, to ease its own process.
- **Less redundancy** – DBMS follows the rules of normalization, which splits a relation when any of its attributes is having redundancy in values. Normalization is a mathematically rich and scientific process that reduces data redundancy.
- **Consistency** – Consistency is a state where every relation in a database remains consistent. There exist methods and techniques, which can detect attempt of leaving database in inconsistent state. A DBMS can provide greater consistency as compared to earlier forms of data storing applications like file-processing systems.
- **Query Language** – DBMS is equipped with query language, which makes it more efficient to retrieve and manipulate data. A user can apply as many and as different filtering options as required to retrieve a set of data. Traditionally it was not possible where file-processing system was used.
- **ACID Properties** – DBMS follows the concepts of Atomicity, Consistency, Isolation, and Durability (normally shortened as ACID). These concepts are applied on transactions, which manipulate data in a database. ACID properties help the database stay healthy in multi-transactional environments and in case of failure.
- **Multiuser and Concurrent Access** – DBMS supports multi-user environment and allows them to access and manipulate data in parallel. Though there are restrictions on transactions when users attempt to handle the same data item, but users are always unaware of them.
- **Multiple views** – DBMS offers multiple views for different users. A user who is in the Sales department will have a different view of database than a person working in the Production department. This feature enables the users to have a concentrate view of the database according to their requirements.
- **Security** – Features like multiple views offer security to some extent where users are unable to access data of other users and departments. DBMS offers methods to impose constraints while entering data into the database and retrieving the same at a later stage. DBMS offers many different levels of security features, which enables multiple users to have different views with

different features. For example, a user in the Sales department cannot see the data that belongs to the Purchase department. Additionally, it can also be managed how much data of the Sales department should be displayed to the user. Since a DBMS is not saved on the disk as traditional file systems, it is very hard for miscreants to break the code.

Users

A typical DBMS has users with different rights and permissions who use it for different purposes. Some users retrieve data and some back it up. The users of a DBMS can be broadly categorized as follows –

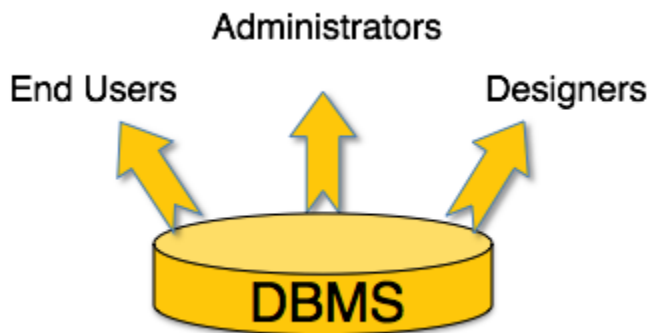


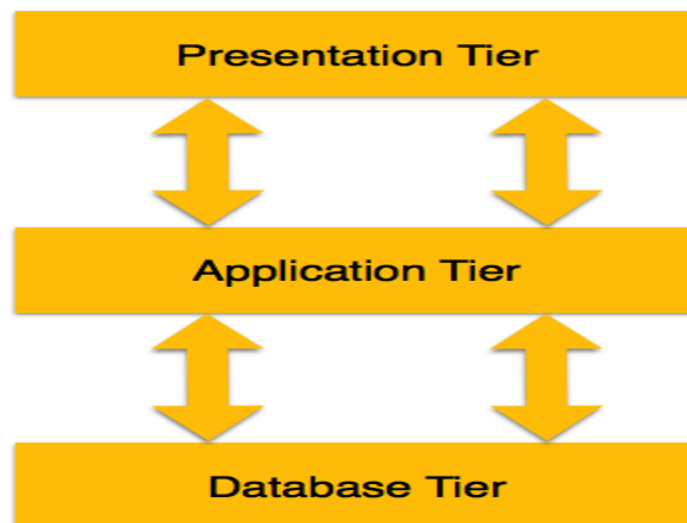
Figure 4 Users

- **Administrators** – Administrators maintain the DBMS and are responsible for administrating the database. They are responsible to look after its usage and by whom it should be used. They create access profiles for users and apply limitations to maintain isolation and force security. Administrators also look after DBMS resources like system license, required tools, and other software and hardware related maintenance.
- **Designers** – Designers are the group of people who actually work on the designing part of the database. They keep a close watch on what data should be kept and in what format. They identify and design the whole set of entities, relations, constraints, and views.
- **End Users** – End users are those who actually reap the benefits of having a DBMS. End users can range from simple viewers who pay attention to the logs or market rates to sophisticated users such as business analysts.

DBMS - Architecture

The design of a DBMS depends on its architecture. It can be centralized or decentralized or hierarchical. The architecture of a DBMS can be seen as either single tier or multi-tier. The tiers are classified as follows :

- **1-tier architecture** involves putting all of the required components for a software application or technology on a single server. It does not provide handy tools for end-users. Database **designers** and **programmers** normally prefer to use single-tier architecture.
- **2-tier**, is like client server application. The direct communication takes place between client and server. There is no intermediate between client and server. Programmers use 2-tier architecture where they access the DBMS by means of an application. Here the application tier is entirely independent of the database in terms of operation, design, and programming.
- **3-tier Architecture** This architecture has different usages with different applications. It can be used in web applications and distributed applications. The strength in particular is when using this architecture over distributed systems.



1. **Database (Data) Tier** – At this tier, the database resides along with its query processing languages. We also have the relations that define the data and their constraints at this level.

2. **Application (Middle) Tier** – At this tier reside the application server and the programs that access the database. For a user, this application tier presents an abstracted view of the database. End-users are unaware of any existence of the database beyond the application. At the other end, the database tier is not aware of any other user beyond the application tier. Hence, the application layer sits in the middle and acts as a mediator between the end-user and the database.
3. **User (Presentation) Tier** – End-users operate on this tier and they know nothing about any existence of the database beyond this layer. At this layer, multiple views of the database can be provided by the application. All views are generated by applications that reside in the application tier.

Multiple-tier database architecture is highly modifiable, as almost all its components are independent and can be changed independently.

DBMS - Data Models

Data models define how the logical structure of a database is modelled. Data Models are fundamental entities to introduce abstraction in a DBMS. “**This process of hiding irrelevant details from user is called data abstraction**”. Data models define how data is connected to each other and how they are processed and stored inside the system.

The very first data model could be flat data-models, where all the data used are to be kept in the same plane. Earlier data models were not so scientific, hence they were prone to introduce lots of duplication and update anomalies.

Entity-Relationship Model

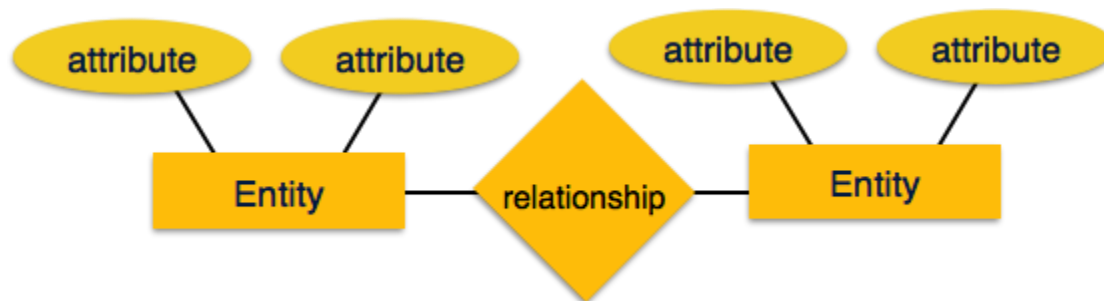
Entity-Relationship (ER) Model is based on the notion of real-world entities and relationships among them. While formulating real-world scenario into the database model, the ER Model creates entity set, relationship set, general attributes and constraints.

ER Model is best used for the conceptual design of a database.

ER Model is based on –

- **Entities** and their *attributes*.
- **Relationships** among entities.

These concepts are explained below.



- **Entity** – An entity in an ER Model is a real-world entity having properties called **attributes**. Every **attribute** is defined by its set of values called **domain**. For example, in a school database, a student is considered as an entity. Student has various attributes like name, age, class, etc.

Let us now learn how the ER Model is represented by means of an ER diagram. Any object, for example, entities, attributes of an entity, relationship sets, and attributes of relationship sets, can be represented with the help of an ER diagram.

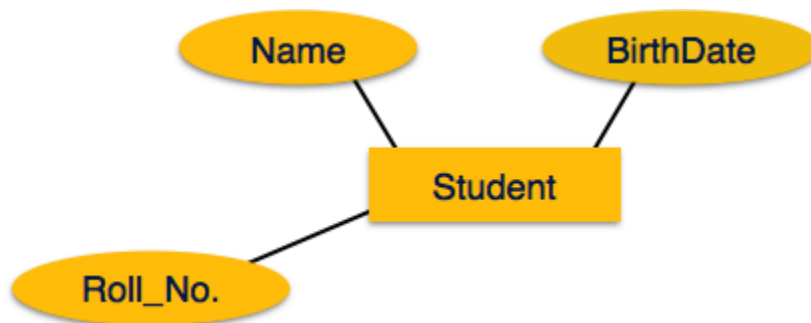
Entity

Entities are represented by means of rectangles. Rectangles are named with the entity set they represent.

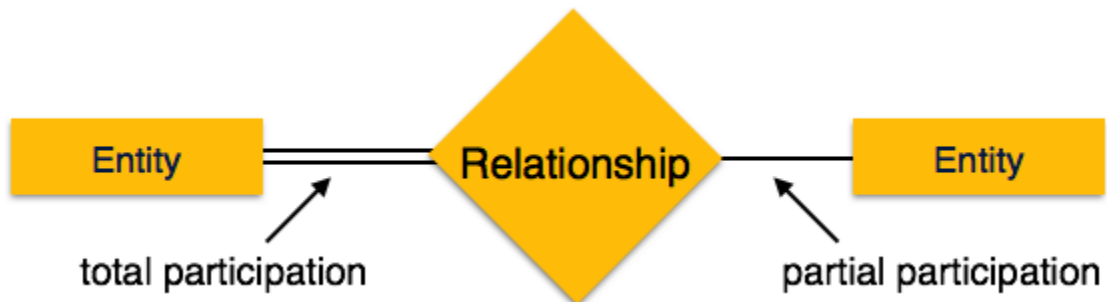


Attributes

Attributes are the properties of entities. Attributes are represented by means of ellipses. Every ellipse represents one attribute and is directly connected to its entity (rectangle).



- Partial participation is represented by single lines.



- **Relationship** – The logical association among entities is called *relationship*. Relationships are mapped with entities in various ways. Mapping cardinalities define the number of association between two entities.

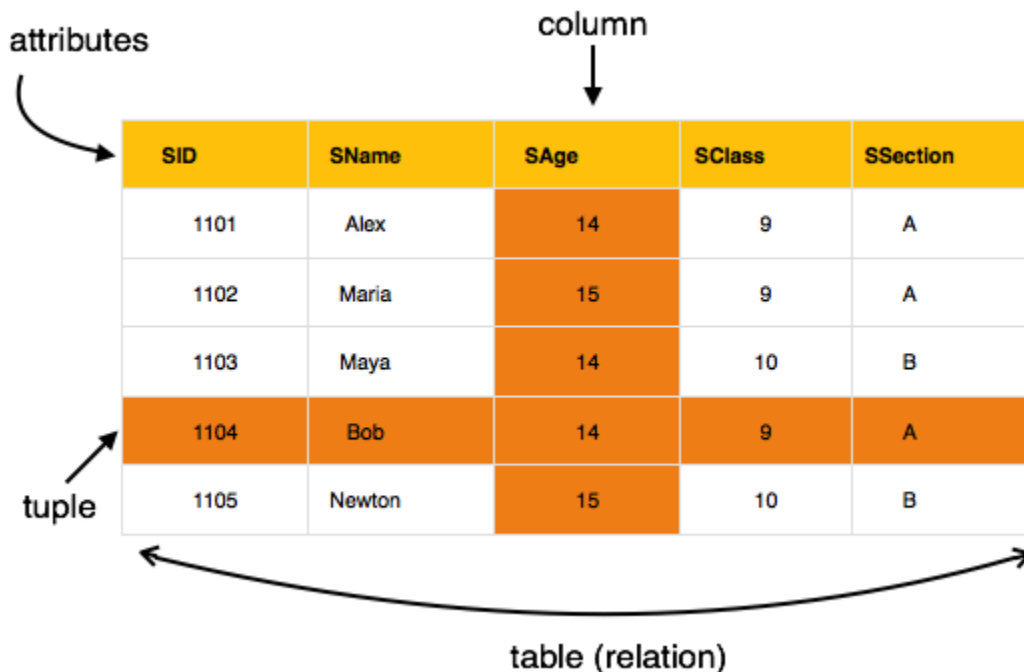
Mapping cardinalities –

- one to one: A single row of table 1 associates with single row of table 2.
- one to many: A single row of table 1 associates with more than one rows of table 2.
- many to one: Many rows of table 1 associate with a single row of table 2.
- many to many: Many rows of table 1 associate with many rows of table 2.

Relationship	Example	left
one-to-one	person \longleftrightarrow id card	1
one-to-one (<i>optional on one side</i>)	person \longleftrightarrow driving license	1
many-to-one	person \longleftrightarrow birth place	1..* or +
many-to-many (<i>optional on both sides</i>)	person \longleftrightarrow book	0..* or *
one-to-many	order \longleftrightarrow line item	1
many-to-one	line item \longleftrightarrow order	1..* or +
many-to-many	course \longleftrightarrow student	1..* or *

Relational Model

The most popular data model in DBMS is the Relational Model. It is more scientific a model than others. This model is based on first-order predicate logic and defines a table as an **n-ary relation**.



The main highlights of this model are –

- Data is stored in tables called **relations**.
- Relations can be normalized.
- In normalized relations, values saved are atomic values.
- Each row in a relation contains a unique value.
- Each column in a relation contains values from a same domain.

Primary key in DBMS

Definition: A **Primary Key (PK)** is a column or set of columns in a table that uniquely identifies tuples (rows) in that table.

Example:

Student Table

Table 1 Student Table

<u>Stu_Id</u>	Stu_Name	Stu_Age
101	Steve	23
102	John	24
103	Robert	28
104	Carl	22

In the above Student table, the Stu_Id column uniquely identifies each row of the table.

Note:

- We denote the primary key by underlining the column name.
- The value of primary key should be unique for each row of the table.
Primary key column cannot contain duplicate values.
- Primary key column should not contain nulls.
- Primary keys are not necessarily to be a single column; more than one column can also be a primary key for a table. For e.g. {Stu_Id, Stu_Name} collectively can play a role of primary key in the above table, but that does not make sense because Stu_Id alone is enough to uniquely identifies rows in a table then why to make things complex. Having that said, we should choose more than one columns as primary key only when there is no single column that can play the role of primary key.

How to choose a primary key?

There are two ways: **Either** to create a column and let database automatically have numbers in increasing order for each row **or** choose a column yourself making sure

that it does not contain duplicates and nulls. For **e.g.** in the above Student table, The Stu_Name column cannot be a primary key as more than one people can have same name, similarly the Stu_Age column cannot play a primary key role as more than one persons can have same age.

Foreign key in DBMS

Definition: Foreign keys are the columns of a table that points to the primary key of another table. They act as a cross-reference between tables.

For example:

In the below example the Stu_Id column in Course_enrollment table is a foreign key as it points to the primary key of the Student table.

Course_enrollment table:

Course_Id	Stu_Id
C01	101
C02	102
C03	101
C05	102
C06	103
C07	102

Student table:

Stu_Id	Stu_Name	Stu_Age
101	Chaitanya	22
102	Arya	26
103	Bran	25
104	Jon	21

Note: Practically, the foreign key has nothing to do with the primary key tag of another table, if it points to a unique column (not necessarily a primary key) of another table then too, it would be a foreign key. So, a correct definition of foreign key would be: Foreign keys are the columns of a table that points to the candidate key of another table.

Composite key in DBMS

A key that consists of more than one attribute to uniquely identify rows (also known as records & tuples) in a table is called composite key. It is also known as compound key.

Example: Table – Sales

cust_Id	order_Id	product_code	product_count
C01	O001	P007	23
C02	O123	P007	19
C02	O123	P230	82
C01	O001	P890	42

Key in above table: {cust_id, order_id}

This is a composite key as it consists of more than one attribute.

Candidate Key in DBMS

Candidate keys are selected from the set of super keys, the only thing we take care while selecting candidate key is: It should not have any redundant attributes. That's the reason they are also termed as minimal super key.

For example:

<u>Emp_Id</u>	Emp_Number	Emp_Name
E01	2264	Steve
E22	2278	Ajeet
E23	2288	Chaitanya
E45	2290	Robert

There are two candidate keys in above table:

{Emp_Id}

{Emp_Number}

Note: A primary key is being selected from the group of candidate keys. That means we can either have Emp_Id or Emp_Number as primary key.

ER diagram consists of:

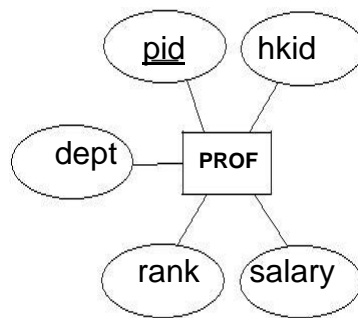
- Entity sets.
- Relationship sets.

We will convert each entity/relationship set to a table, which involves deciding the tables.

- Attributes and candidate key.

Given an entity set, create a table with the same attributes and candidate key.

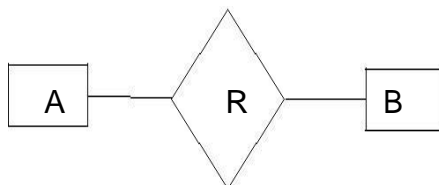
Example:



PROF (pid, hkid, dept, rank, salary)

where pid is set as a candidate key.

Let us rst consider binary relationship sets. To convert R as shown below



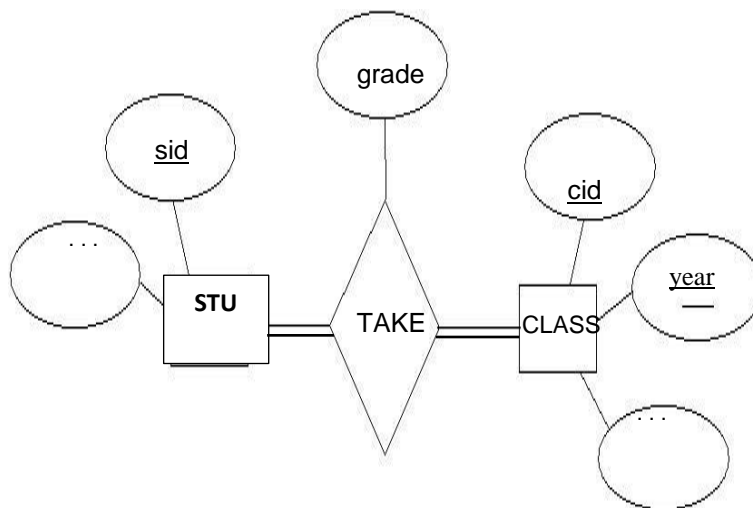
We create a table whose attributes include:

- The candidate keys of both A and B.
- The attributes of R (if any).

The candidate key of the table, however, depends on the cardinality constraint of R. See the next few slides.

Many-to-many: The candidate key of R includes all the attributes in the candidate keys of A and B.

Example:

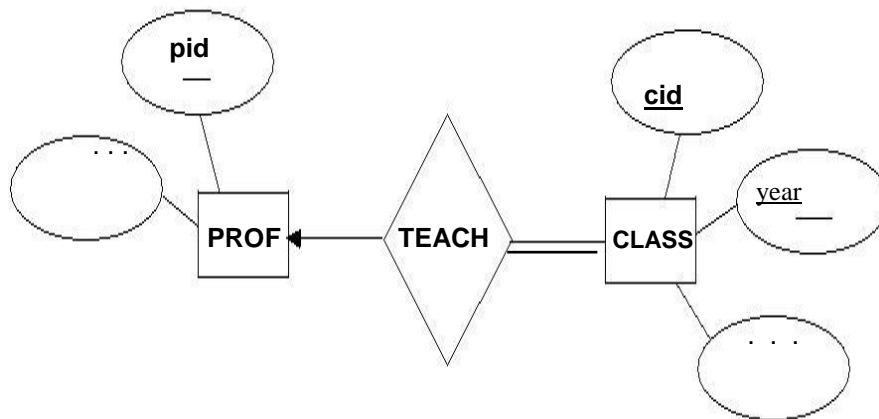


TAKE (sid, cid, year, grade)

where the candidate key is (sid, cid, year).

One-to-many: The candidate key of R is the same as B (i.e., the entity set on the \many" side).

Example:

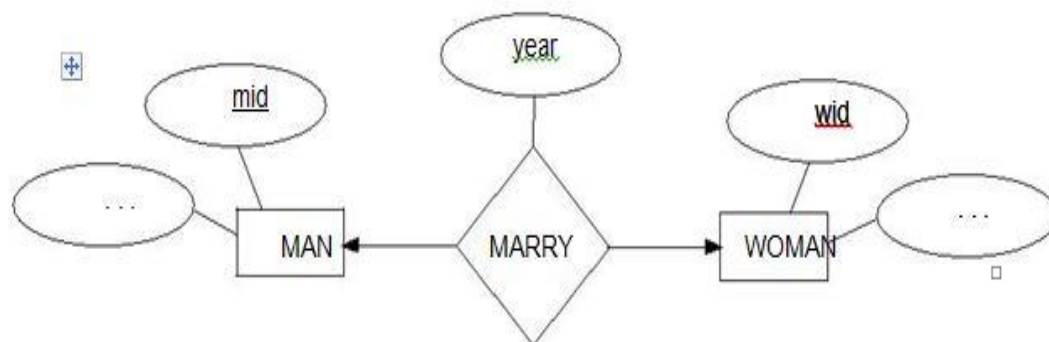


TEACH (pid, cid, year)

where the candidate key is (cid, year).

One-to-one: R has two candidate keys. The rst (second) one is the same as A (B).

Example:



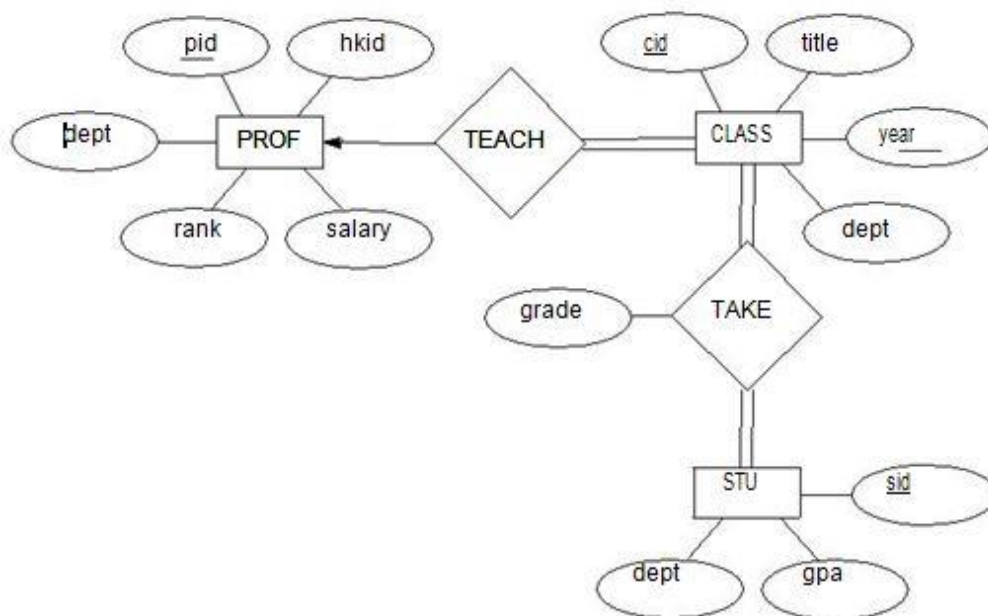
MARRY (mid, wid, year)

where a candidate key is mid, and another is wid.

Multi-way relationship set R: Create a table that includes

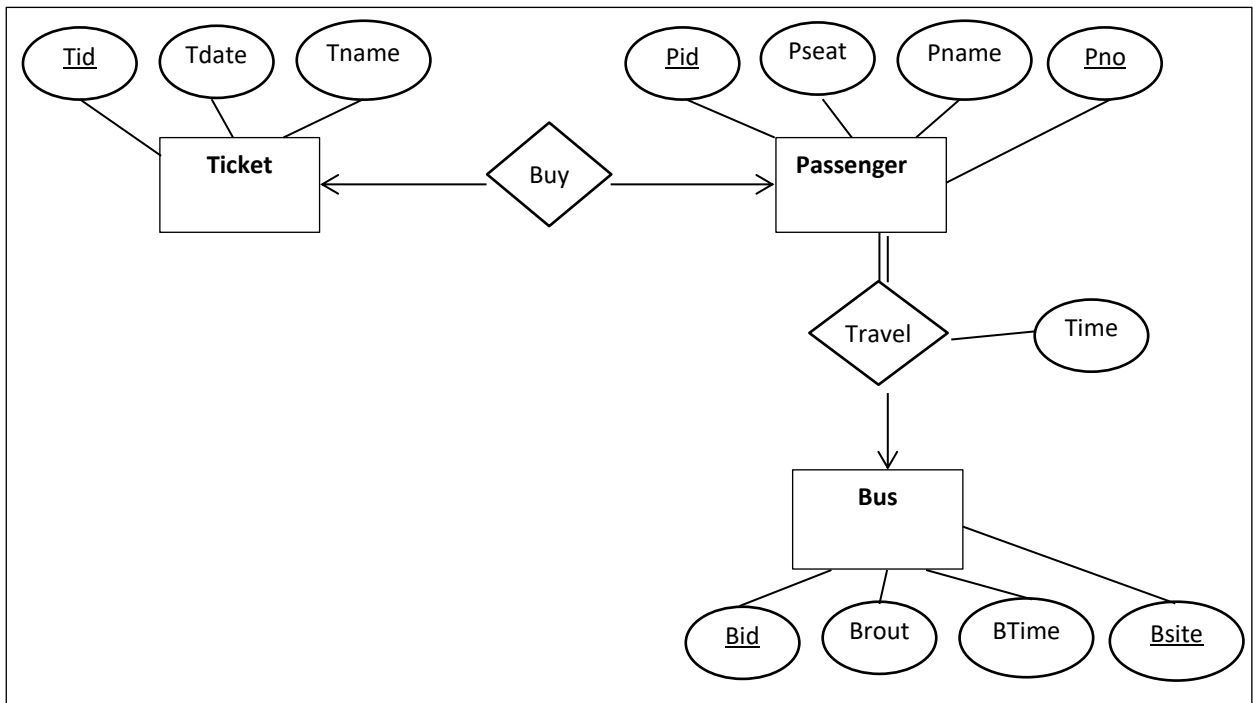
- The candidate keys of the participating entity sets.
- The attributes of R (if any).

The candidate key of the table includes all the attributes of the candidate keys of the participating entity sets.



- PROF (pid, hkid, dept, rank, salary), candidate key pid
- CLASS (cid, title, year, dept), candidate key (cid, year)
- STU (sid, dept, gpa), candidate key sid
- TEACH (pid, cid, year), candidate key (cid, year)
- TAKE (cid, year, sid, grade), candidate key (cid, year, sid)

Q1 Convert the ER diagram into relational tables

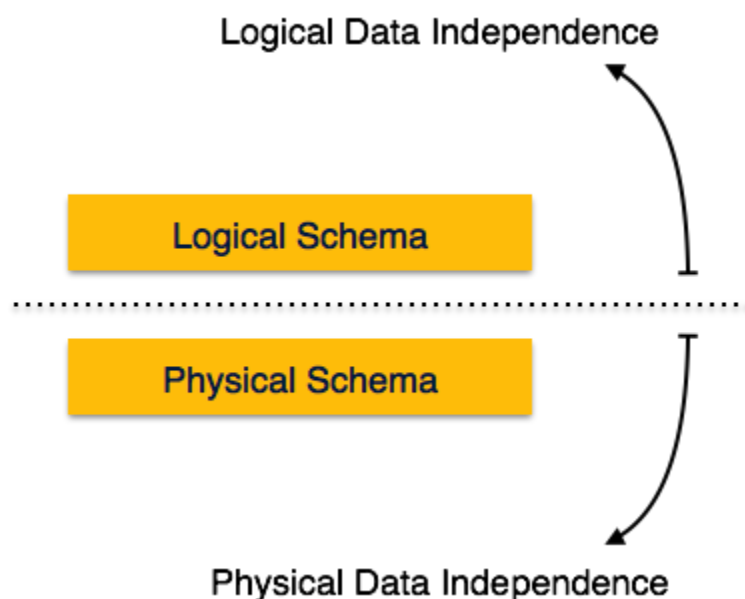


Q2 Convert the resulted tables in Q1 into ERD.

Data Independence

The ability to modify a scheme definition in one level without affecting a scheme definition in the next higher level is called *data independence*. There are two levels of data independence:

ان القدرة على تعديل تعريف المخطط في مستوى واحد دون التأثير على تعريف المخطط في المستوى الأعلى التالي تسمى استقلالية البيانات.



➤ Logical Data Independence

Logical data is data about database, that is, it stores information about how data is managed inside.

Logical data independence is a kind of mechanism, which liberalizes itself from actual data stored on the disk. If we do some changes on table format, it should not change the data residing on the disk.

البيانات المنطقية هي بيانات عن قاعدة البيانات ، بمعنى أنها تخزن معلومات حول كيفية إدارة البيانات بالداخل . على سبيل المثال .

استقلالية البيانات المنطقية هي نوع من الآلية ، التي تحرر نفسها من البيانات الفعلية المخزنة على القرص. إذا قمنا ببعض التغييرات على تنسيق الجدول ، فلا ينبغي أن يغير البيانات الموجودة على القرص.

➤ Physical Data Independence

Is the power to change the physical data without impacting the schema or logical data.

For example, in case we want to change or upgrade the storage system itself – suppose we want to replace hard-disks with SSD – it should not have any impact on the logical data or schemas.

استقلالية البيانات المادية هي القدرة على تغيير البيانات الفعلية دون التأثير على المخطط أو البيانات المنطقية. على سبيل المثال ، في حالة ما إذا كنا نريد تغيير أو ترقية نظام التخزين نفسه - لنفترض أننا نريد استبدال الأقراص الصلبة بقرص - SSD فلا ينبغي أن يكون لها أي تأثير على البيانات أو المخططات المنطقية.

2- Normalization

Database Normalization is a technique of organizing the data in the database. Normalization is a systematic approach of decomposing tables to eliminate data redundancy (repetition) and undesirable characteristics like **Insertion**, **Update** and **Deletion** anomalies.

It is a multi-step process that puts data into tabular form, removing duplicated data from the relation tables.

Normalization usually involves dividing a [database](#) into two or more [tables](#) and defining [relationships](#) between the tables. The objective is to isolate data so that additions, deletions, and modifications of an attribute can be made in just one table and then propagated through the rest of the database via the defined relationships.

There are three main normal forms, each with increasing levels of normalization:

1. **First Normal Form (1NF).**
2. **Second Normal Form (2NF).**
3. **Third Normal Form (3NF).**

And other types *Boyce Codd Normal Form (BCNF)*, *fourth normal form (4NF)* and *fifth normal form (5NF)*.



Normalization is used for mainly two purposes,

- **Eliminating redundant (useless) data.**
- **Ensuring data dependencies make sense i.e data is logically stored.**
- **Simplifying the process of insertion, deletion and updating DB.**

Problems without Normalization

If a table is not properly normalized and have data redundancy then it will not only eat up extra memory space but will also make it difficult to handle and update the database, without facing data loss. Insertion, Updation and Deletion Anamolies are very frequent if database is not normalized. To understand these anomalies let us take an example of a **Student** table.

Table 2 جدول الطلبة مع القسم بصيغة UNF

rollno	name	branch	hod	office_tel
401	Akon	CSE	Mr. X	53337
402	Bkon	CSE	Mr. X	53337
403	Ckon	CSE	Mr. X	53337
404	Dkon	CSE	Mr. X	53337

من الملاحظ في هذا المثال ان البيانات غير مرتبة بشكل صحيح بصيغة (UNF) مما يؤدي الى ليس فقط ضياع في الذاكرة وانما الى مشكلة في عمليات الإضافة والتحديث والحذف. لتكرار البيانات في ثلاث حقول المضللة في الشكل اعلاه

In the table above, we have data of 4 Computer Sci. students. As we can see, data for the fields **branch**, **hod(Head of Department)** and **office_tel** is repeated for the students who are in the same branch in the college, this is **Data Redundancy**.

Insertion Anomaly

Suppose for a new admission, until and unless a student opts for a branch, data of the student cannot be inserted, or else we will have to set the branch information as NULL.

Also, if we have to insert data of 100 students of same branch, then the branch information will be repeated for all those 100 students.

في حال تطلب إضافة 100 طالب سيتطلب إعادة كتابة ثلاثة حقول **hod, office_tel, branch** في كل مرة يضاف فيها الطالب مما يؤدي الى ضياع في الخزن

Updation Anomaly

What if Mr. X leaves the college? or is no longer the HOD of computer science department? In that case all the student records will have to be updated, and if by mistake we miss any record, it will lead to data inconsistency.

في حال مغادرة احد الطلبة للكلية او تعيين رئيس قسم جديد سيتطلب ذلك التعديل على اربع حقول وفي جميع الجداول التي تحتوي على **hod**

Deletion Anomaly

In our **Student** table, two different information are kept together, Student information and Branch information. Hence, at the end of the academic year, if student records are deleted, we will also lose the branch information. This is Deletion anomaly.

في حال تم حذف الطلبة الأربعة في الجدول لمغادرتهم او لسبب اخر سوف يتم حذف السجل كاملا مما يؤدي الى حذف القسم ومعلوماته كاملا المتمثلة بـ **hod, office_tel, branch**

السبب الرئيسي في مشاكل الإضافة والتحديث والحذف هي ان الجدول أعلاه يحتوي على معلومات للطالب والقسم في نفس الجدول لذلك تطلب فصل الجدول الى جدولين او أكثر حسب الحالة مما يسهل عملية الإضافة والتحديث والحذف وهو أساس عمل الـ **Normalization**.

الحل

1- تقسيم الجدول Table 2 الى جدولين جدول الطلبة وجدول القسم لغرض التحديث

STUDENTS TABLE			BRANCH TABLE		
rollno	name	branch	branch	hod	office_tel
1	Akon	CSE	CSE	Mr. Y	53337
2	Bkon	CSE		Mr. Z	53338
3	Ckon	CSE			

في حال التحديث على ال hod او office_tel سيكون التحديث في جدول القسم في مكان واحد ولا داعي للتحديث في جدول الطلبة بدلا من ثلاث أماكن .. افترض ان الجدول يحتوي على 100 طالب ؟

2- لغرض الإضافة ستكون العملية اسهل كما موضح في جدول الطلبة ادناه (الطالب رقم 4) ستكون الاضافة في مكان واحد ... دون الحاجة الى إعادة كتابة معلومات الطالب في ثلاث حقول كما هو سابقا في Table 2 أي حقول office_tel , branck , hod في جدول ال branch مخزنة في جدول مستقل لتسهيل عملية الاضافة

STUDENTS TABLE			BRANCH TABLE		
rollno	name	branch	branch	hod	office_tel
1	Akon	CSE	CSE	Mr. Y	53337
2	Bkon	CSE			
3	Ckon	CSE			
4	Dkon	CSE			

3- تكون عملية الحذف ادق في حال مسح جميع الطلبة من Table 2 دون مسح معلومات جدول ال **branch** كما موضح في الشكل التالي .. حيث تبقى معلومات ال **CSE** في الجدول المذكور حتى وان تم حذف جميع الطلبة من جدول الطلاب

STUDENTS TABLE			BRANCH TABLE		
rollno	name	branch	branch	hod	office_tel
			CSE	Mr. Y	53337

وهنا تكون عملية ال **normalization** مهمة جدا في عملية التصميم لأغراض الإضافة والحذف والتعديل وهي على ثلاثة أنواع **1nf, 2nf, 3nf**

1- First Normal Form (1NF)

For a table to be in the First Normal Form, it should follow the following 4 rules:

1. It should only have single value (atomic) valued attributes/columns.
2. Values stored in a column should be of the same domain
3. All the columns in a table should have unique names.
4. And the order in which data is stored, does not matter.

In the next tutorial, we will discuss about the [First Normal Form](#) in details.

Time for an Example

Although all the rules are self-explanatory still let's take an example where we will create a table to store student data which will have student's roll no., their name and the name of subjects they have opted for.

من الملاحظ ان Table 3 يحتوي في حقل الاسم على قيمة مزدوجة تحتاج الى معالجة وكذلك في حقل الموضوع أيضا ووجود تنوع في حقل الأقطار والعواصم يحتاج الى توحيد ليصبح الجدول بصيغة ال 1NF

Table 3 UNF form – Student Table

roll_no	name	subject	country
101	Akon Charley	OS, CN	Baghdad
103	Ckon Daily	Java	Iraq
102	Bkon John	C, C++	Manchester

How to solve this Problem?

It's very simple, because all we have to do is break the values into atomic values. Here is our updated table and it now satisfies the First Normal Form.

Table 4 1NF normalisation

roll_no	fname	Lname	subject	country
101	Akon	Charley	OS	Iraq
101	Akon	Charley	CN	Iraq
103	Ckon	Daily	Java	Iraq
102	Bkon	John	C	UK
102	Bkon	John	C++	UK

By doing so, although a few values are getting repeated but values for the subject column are now atomic for each record/row.

Using the First Normal Form, data redundancy increases, as there will be many columns with same data in multiple rows but each row as a whole will be unique.

2- Second Normal Form (2NF)

For a table to be in the Second Normal Form,

1. It should be in the First Normal form.
2. And, it should not have Partial Dependency.

We should understand the definition of **dependency**.

Functional dependency

Is a relationship that exists when one attribute uniquely determines another attribute. If R is a relation with attributes X and Y, a functional dependency between the attributes is represented as $X \rightarrow Y$, which specifies Y is functionally dependent on X.

$X \rightarrow Y$ أي ان X يحدد Y او ممكن القول ان Y يعتمد على X

In this table 5, **student_id** is the primary key and will be unique for every row; hence we can use **student_id** to fetch any row of data from this table

Even for a case, where student names are same, if we know the student_id we can easily fetch the correct record.

Table 5 student

<u>student_id</u>	name	reg_no	branch	address
10	Akon	07-WY	CSE	Kerala
11	Akon	08-WY	IT	Gujarat

أي ان جميع الحقول في Table 5 تعتمد على student_id لانه المفتاح الرئيسي PK لأننا عن طريقه يمكن الوصول الى سجل داخل الجدول

Hence we can say a **Primary Key** for a table is the column or a group of columns (composite key) which can uniquely identify each record in the table.

I can ask from branch name of student with **student_id 10**, and I can get it. Similarly, if I ask for name of student with **student_id 10** or **11**, I will get it. So all I need is **student_id** and every other column **depends** on it, or can be fetched using it.

This is **Dependency** and we also call it **Functional Dependency**.

What is Partial Dependency (PD)?

Partial Dependency means that a nonprime attribute is functionally dependent on *part* of a candidate key. (A nonprime attribute is an attribute that's not part of *any* candidate key.)

For example, let's start with $R\{ABCD\}$, and the functional dependencies $AB \rightarrow CD$ and $A \rightarrow C$.

The only candidate key for R is \underline{AB} .

C and D are a nonprime attributes. C is functionally dependent on A .

A is *part* of a candidate key. *That's* a partial dependency

During the exam we need a new score table



SCORE TABLE				
score_id	student_id	subject_id	marks	teacher
1	10	1	82	Mr. J
2	10	2	77	Mr. C++
3	11	1	85	Mr. J
4	11	2	82	Mr. C++
5	11	4	95	Mr. P

student_id + subject_id

can uniquely identify any row of data in SCORE table

score_id	student_id	subject_id	marks	teacher
1	10 ?	1	82	Mr. J
2	10	2	77	Mr. C++
3	11 ?	1	85	Mr. J
4	11	2	82	Mr. C++
5	11	4	95	Mr. P

- السبب في اختيار student_id و subject_id سوياً كـ pk لانه في حال البحث عن **subject_id=1** سوف ترجع نتيجة الاستعلام اما **student_id=10** او **student_id=11** كما موضح أعلاه في الجدول.

- وفي حال طلب student_id=10 سيكون الـ **subject_id=1** او **subject_id=2** لذلك تم اختيارهما معاً

in Score table
Primary key is a composition of two columns

student_id + subject_id

من الملاحظ ان فقط حقل teacher يعتمد على subject_id

score_id	student_id	subject_id	marks	teacher
1	10	1	82	Mr. J
2	10	2	77	Mr. C++
3	11	1	85	Mr. J
4	11	2	82	Mr. C++
5	11	4	95	Mr. P

أي non-prime attribute يعتمد على prime attribute وهذا مايسمى partial dependency الذي يجب ازالته بنقل ال teacher الى جدول ال subject كالتالي :

score_id	student_id	subject_id	marks	teacher
1	10	1	82	Mr. J
2	10	2	77	Mr. C++
3	11	1	85	Mr. J
4	11	2	82	Mr. C++
5	11	4	95	Mr. P

subject_id	subject_name
1	Java
2	C++
3	C#
4	Php

Move teacher column to Subject Table

SUBJECT TABLE		
subject_id	subject_name	teacher
1	Java	Mr. J
2	C++	Mr. C++
3	C#	Mr. C#
4	Php	Mr. P

Makes more sense here...

How to remove Partial Dependency (PD)?

There can be many different solutions for this, but our objective is to remove teacher's name from Score table.

The simplest solution is to remove columns teacher from Score table and add it to the Subject table. Hence, the Subject table will become:

نفس الجدول أعلاه ممكن كتابته بطريقتين **Subject Table**

subject_id	subject_name	teacher
1	Java	Java Teacher
2	C++	C++ Teacher
3	Php	Php Teacher

And our Score table is now in the second normal form, with no partial dependency.

بعد فصل ونقل الجدول سيكون شكل جدول النتائج كالتالي **Score Table**

score_id	student_id	subject_id	marks
1	10	1	70
2	10	2	75
3	11	1	80

Quick Recap

1. For a table to be in the Second Normal form, it should be in the First Normal form and it should not have Partial Dependency.
2. Partial Dependency exists, when we have a composite primary key, any attribute in the table depends only on a part of the primary key and not on the complete primary key.
3. To remove Partial dependency, we can divide the table, remove the attribute which is causing partial dependency, and move it to some other table where it fits in well.

3- Third Normal Form (3NF)

A table is said to be in the Third Normal Form when,

1. It is in the Second Normal form.
2. And, it doesn't have Transitive Dependency.

Transitive Dependency (TD):

Let A, B, and C designate three distinct attributes (or distinct collections of attributes) in the relation. Suppose all three of the following conditions hold:

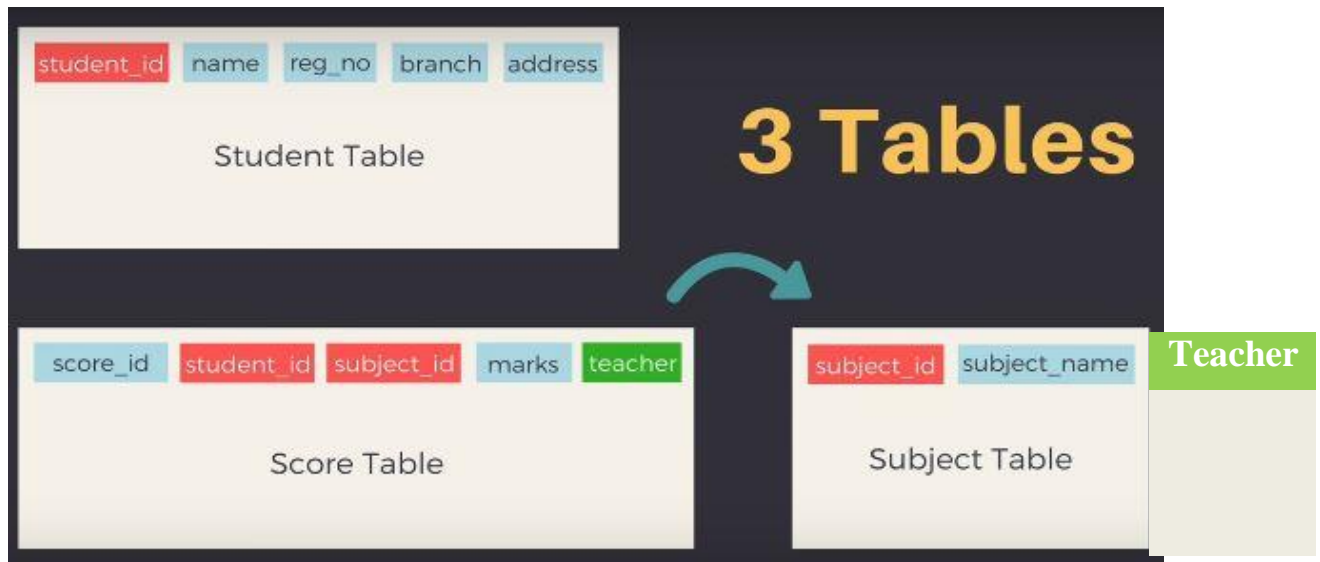
1. $A \rightarrow B$
2. It is not the case that $B \rightarrow A$
3. $B \rightarrow C$

Then the functional dependency $A \rightarrow C$ (which follows from 1 and 3 by the [axiom of transitivity](#)) is a transitive dependency.

It occurs when a non-prime attribute depends on other non-prime attributes rather than depending upon the prime attributes or primary key.

لانتقال الى 3NF يجب ان تكون الجداول بصيغة ال 2NF ويجب إزالة الاعتمادية الانتقالية Transitive dependency (TD)

قد تم تقسيم الجداول الى ثلاثة جداول في ال 2NF في المحاضرة السابقة وكما موضح ان ال PK يظهر باللون الأحمر في الجدول ادناه وقد تم نقل ال حقل teacher الذي يعتمد على subject_id في جدول ال score الى جدول subject لإزالة الاعتمادية الجزئية



For a table to be in 3rd Normal Form:

- It should be in 2nd Normal Form
- And it should not have Transitive Dependency.

لو اردنا إضافة معلومات إضافية لجدول الدرجات وهما حقلين الاول اسم الامتحان والثاني اجمالي الدرجات

We want to save more information in our Score table.

Exam Name & Total Marks

سيصبح الجدول كما هو ادناه

SCORE TABLE					
score_id	student_id	subject_id	marks	exam_name	total_marks

وكما موضح ان student_id و subject_id هما المفاتيح الأساسية

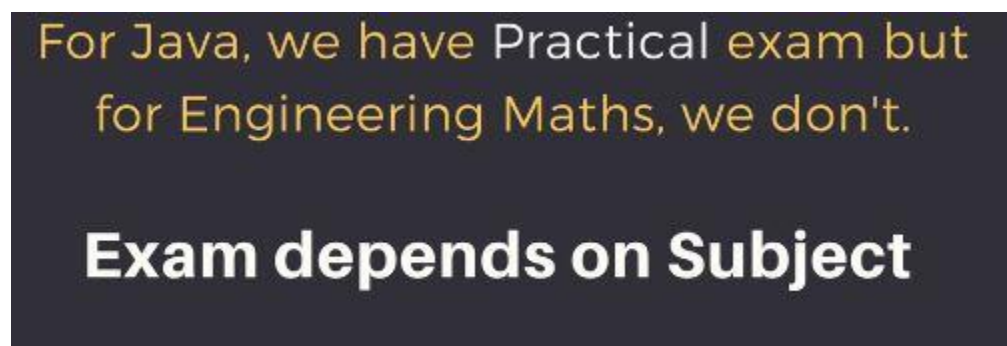
SCORE TABLE					
score_id	student_id	subject_id	marks	exam_name	total_marks

Primary Key for Score table is
a Composite Key

(Student_id, subject_id) as a PK → exam_name



لكن لبعض الحالات سوف يعتمد الامتحان على الموضوع مثلاً



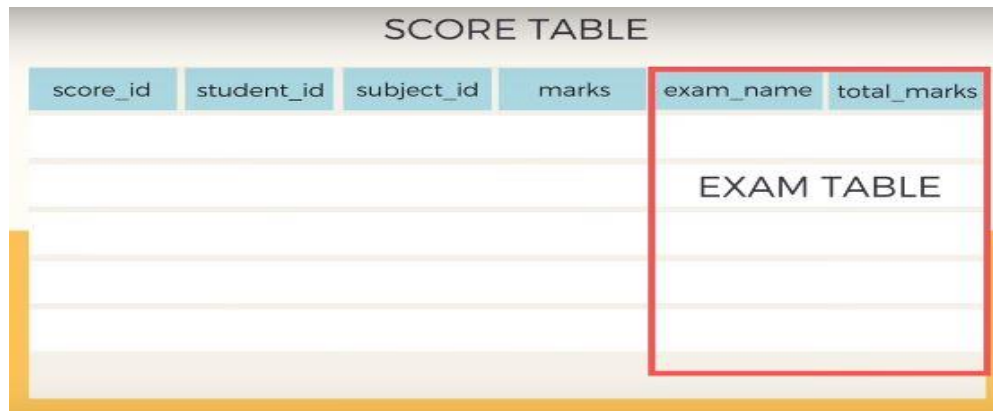
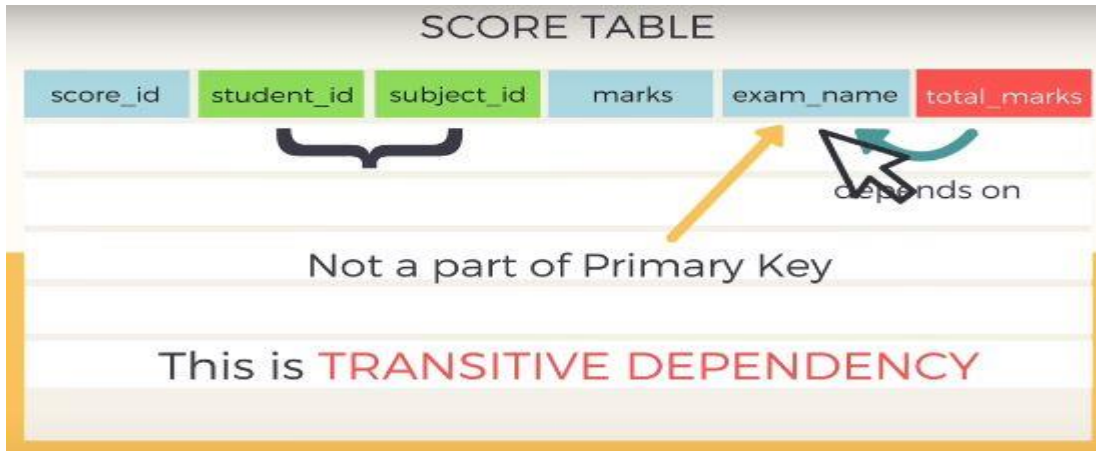
أي ممكن القول ان



ولكن total_marks يعتمد على حقل exam_name وهي ماتسمى الاعتمادية الانتقالية فيتوجب اخراج هذه الحقول بجدول جديد واعطائه اسم جديد.

exam_name → total_marks

What about the second new column **total_marks**?



يتم اخراج الحقليين أعلاه في جدول جديد يسمى exam table



Q1- Based on the functional dependencies, create a database whose tables are at least in 2NF and 3NF normalization showing the dependency diagram for each table.

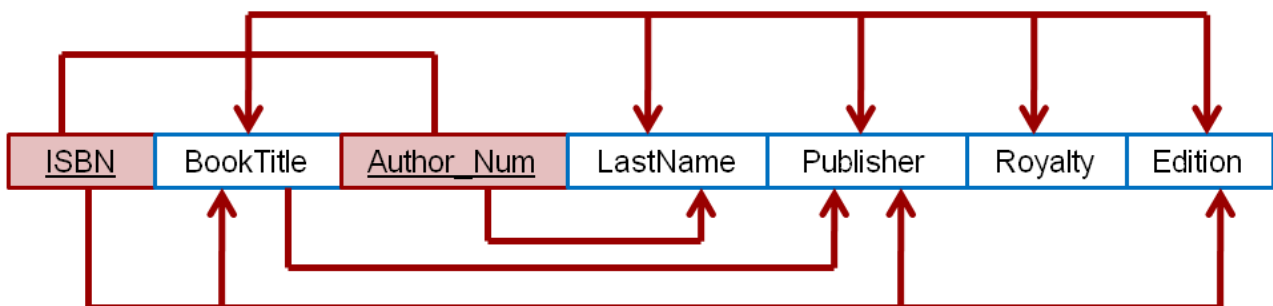
The authors are paid royalties for each book that they write for a publisher. The amount of the royalty can vary by author, by book, and by edition of the book.

(ISBN, BookTitle, Author_Num, LastName, Publisher, Royalty, Edition)

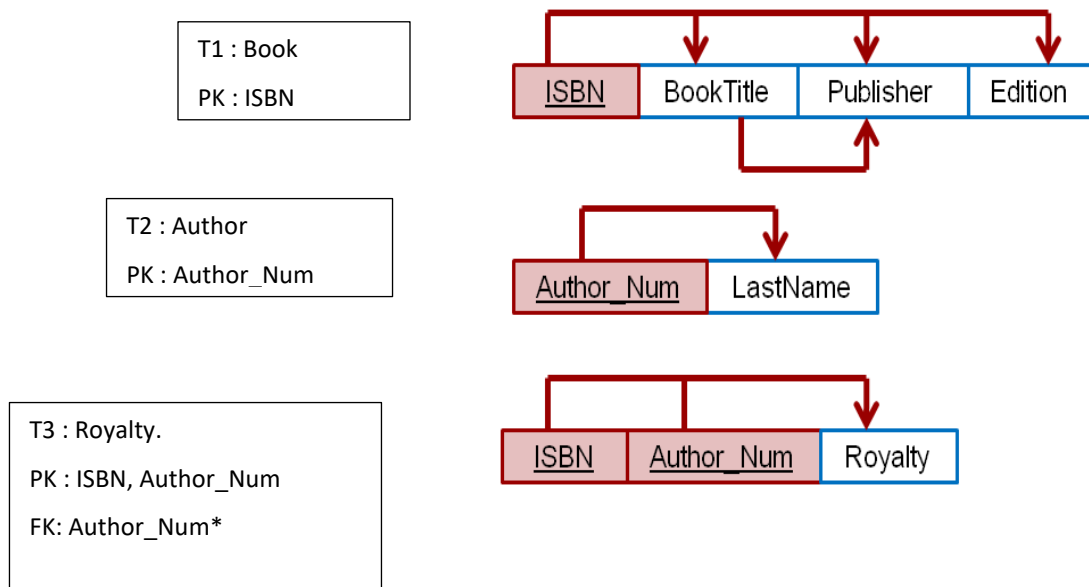
- a- BookTitle, Publisher, Edition are dependents on ISBN.
- b- LastName is a dependent on Author_Num.
- c- Publisher is dependent on BookTitle.
- d- Book Title, LastName, Publisher, Royalty, Edition are dependandants on ISBN, Author_Num .

The answer:

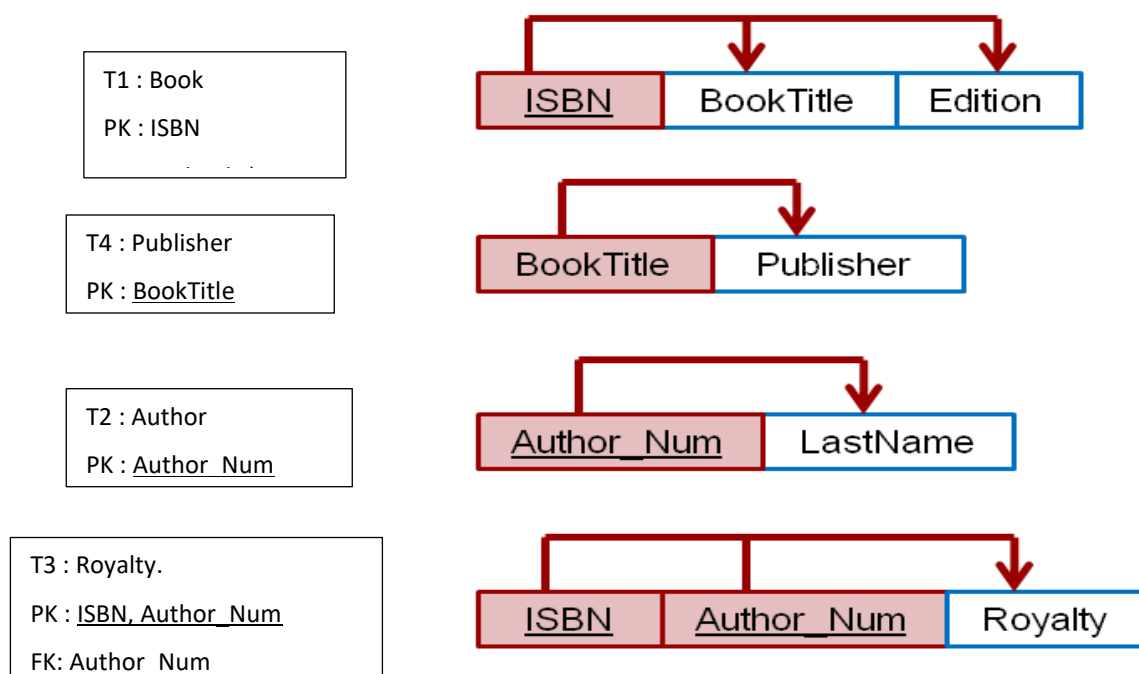
a- 1NF Dependency Diagram.



b- 2NF Removing partial dependency



c- 3NF Removing Transitive dependency



Structure Query Language (SQL)

What is SQL? Stands for "Structured Query Language," It is a query language used for accessing and modifying information in a database.

SQL Types: five main categories of SQL each with its own commands as shown in the table (1) below

Categories	commands	Command descriptions
Data Retrieval	SELECT	Retrieve information from table
Data Manipulation Language (DML)	INSERT	Insert information to table or object
	UPDATE	Edit information on table or object
	DELETE	Delete information from table or object
Data Definition Language (DDL)	CREATE	Create table or object
	Alter	Edit table or object
	DROP	Delete table or object
	RENAME	Edit table or object
	TRUNCATE	Delete a part of table or object
Transaction Control	COMMIT	Commit the inf. On table
	ROLLBACK	Rollback on the new info.
	SAVEPOINT	Go back to the point where the system Is consistence
Data control Language (DCL)	GRANT	Give users access to the info.
	REVOKE	Blocking the user's access of info.

Table (1) SQL categories

Data Retrieval (SELECT):

The basic form of the SELECT statement is formed of the three clauses SELECT, FROM, and WHERE and has the following form

SELECT <attribute list>

FROM <table list>

[WHERE <condition>]

[ORDER BY <attribute list>]

<attribute list> is a list of attribute names whose values are to be retrieved by the query.

<table list> is a list of the relation names required to process the query.

<condition> is a conditional (Boolean) expression that identifies the tuples to be retrieved by the query.

In SQL, the basic logical comparison operators for comparing attribute values with one another and with literal constants are =, <=, >, >=, and

< >. These correspond to the relational algebra operators =, <=, >, ≥, and ≠, respectively.

Employee Table/(emp)	Department Table/(dept)
Empno: employee number (primary key)	Deptno: department number
empname: employee name	Deptname: department name
Sal: month salary	
Job: employee job title	
Hiredate: hire date	
Mgr: the manger name	
Deptno: department number	
Commotion: commotion amount	

Table (2) the employee and department tables

Q1: Show all the information from the employee table.

```
SELECT * FROM emp ;
```

Q2: Show the employees names and numbers from employee table.

```
SELECT empname, Empno FROM emp ;
```

Q3: Show the employees names, salaries and their annual salaries from the employee table, name the last field by annual salary.

```
SELECT empname, sal , sal*12 FROM emp ;
```

Or we can change the attribute name using as

```
SELECT empname, sal , sal*12 as "Annual Salary" FROM emp ;
```

Q4: Show the employees names, salaries, salaries plus commotion for each employees from employee table.

```
SELECT empname, sal, sal+commotion FROM emp;
```

Q5: Show the employees names, salaries, annual salaries plus 100\$ for each employees from employee table.

```
SELECT empname, sal , (sal*12)+100 FROM emp;
```

Q6: Retrieve the hire date and department number of the employee(s) whose name is “John”

```
SELECT hiredate, deptno FROM emp WHERE empname = 'John';
```

Q7. Retrieve the name and job title of all employees who work for the department number 10.

```
SELECT empname, job FROM emp WHERE deptno =10 ;
```

Q8. Show the employee name and the employee job title by the same field give the field new name.

```
SELECT empname, job , empname || job as 'emploJob' FROM
emp ;
```

The concatenation tools use to companies two attribute in one; not in the table just in the result of retrieve of the SQL statement

Q9. Show the name, salary, commotion For the employees whose their salary equal or less than their commotion.

```
SELECT empname, sal , commotion
FROM emp WHERE sal <=commotion;
```

Some operator used with WHERE condition

BETWEEN AND	NOT (BETWEENAN)
IN()	NOT IN()
LIKE	NOT LIKE()
IS NULL	IS NOT NULL

Q10. Retrieve the names of employees who their salary is more than 100 and less than 850.

```
SELECT empname FROM emp
WHERE sal >100 AND sal < 850 ;
```

Or we can use the BETWEEN AND operator

```
SELECT empname FROM emp
WHERE sal BETWEEN 100 AND 850 ;
```



جامعة بغداد
كلية التربية للعلوم الصرفة / ابن الهيثم
قسم علوم الحاسبات
المرحلة الثانية

صباحي + مسائي

تحليل أنظمة و قواعد بيانات

الفصل 1

د.احمد صبيح توفيق

System Analysis & Database

Chapter 1: System Development Life Cycle



System Analysis and Design

System: organized set of related components established to accomplish certain task.

There are two types of systems:

- *Closed System:* Is one that has no interaction with the environment so it completely self-contained.
- *Opened System:* Is one that has an environment and continuously interacts with it, Such as (computer, human etc.).

Computer system: A system that has computers as one of its components.

Information Technology (IT): refers to the combination of hardware, software, and services that people use to manage, communicate, and share information. IT is driving a new digital economy, where advances in hardware, software, and connectivity can provide enormous benefits to businesses and individuals.

Information System: An information system combines information technology, people, and data to support business requirements.

Systems Analysis and Design: Step-by-step process for developing high-quality *Information Systems*.

The IT department team includes systems analysts

Systems Analyst: Professional computer employee who plans, develops, and maintains information systems, to build a successful Information System; analysts must understand business processes and document them carefully. Some of Systems Analyst Personal Qualities are Analytical mind, Good communication skills, Self-discipline etc...

Systems Development Methods

- ❖ The most popular alternatives are structured analysis, which is a traditional method that still is widely used, and object-oriented analysis (O-O), which is a newer approach that many analysts prefer.
- ❖ The important thing is for a systems analyst to understand the various methods and the strengths and weaknesses of each approach.

Structured Analysis: Is a traditional Systems development technique that is time-tested and easy to understand. Structured analysis uses a series of phases, called Systems development life cycle (SDLC), to plan, analyse, design, implement, and support an information system.

Understand The Business:

IT professionals must understand a company's operations to design successful systems. Each business situation is different. For example, a retail store, a medical practice, and a hotel chain all have unique information systems requirements. Systems analysts use a process called business process modelling to represent company operations and information needs. Business process modeling requires a business profile and a series of models that document business processes.

Business Profile A business profile is an overview of a company's mission, functions, organization, products, services, customers, suppliers, competitors, constraints, and future direction. Although much of this information is readily available, a systems analyst usually needs to do additional research and fact-finding. A business profile is the starting point for the modelling process.

Business Process A business process is a specific set of transactions, events, and results that can be described and documented.

Systems Development Life Cycle (SDLC) Waterfall Model

- Usually includes five PHASES:
 - Planning phase.
 - Analysis phase.
 - Design phase.
 - Implementation phase.
 - Maintenance, operation, support, and security phase.

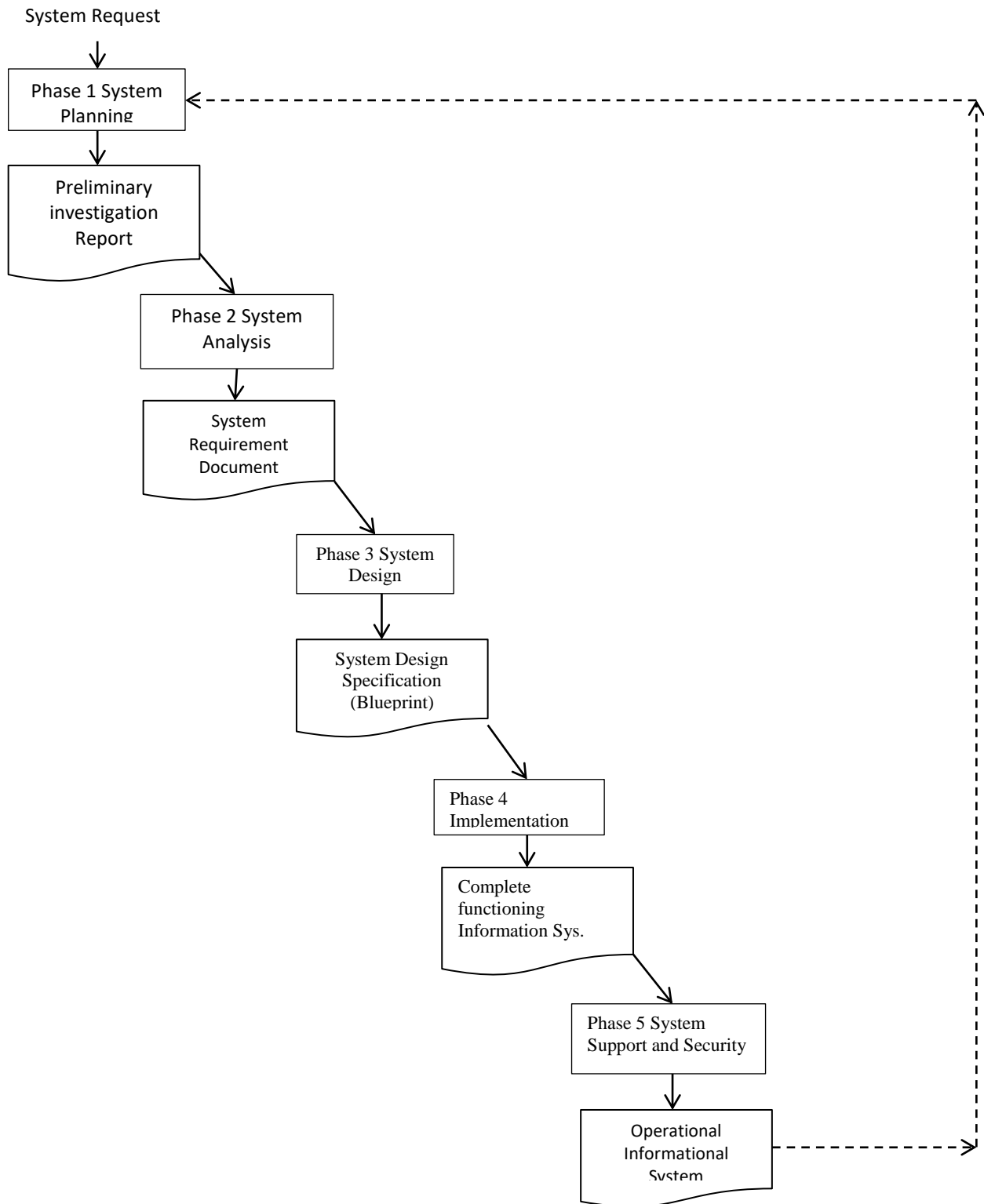


Figure 5 Systems Development Life Cycle (SDLC) in the waterfall model

1- PLANNING Phase:

The systems planning phase usually begins with a formal request to the IT department, called a systems request, that describes problems or desired changes in an information system or a business process. In many companies, IT systems planning is an integral part of overall business planning. When managers and users develop their business plans, they usually include IT requirements that generate systems requests.

A systems request can come from a top manager, a planning team, a department head, or the IT department itself. The request can be very significant or relatively minor. A major request might involve a new information system or the replacement of an existing system that cannot handle current requirements. In contrast, a minor request might ask for a new feature or a change to the user interface.

The purpose of this phase is to perform a preliminary investigation to identify the nature and scope of the business opportunity or problem. The preliminary investigation is a critical step because the outcome will affect the entire development process. A *key part* of the preliminary investigation is a feasibility study that reviews anticipated costs and benefits and recommends a course of action based on operational, technical, economic, and time factors.

- ❖ In this phase, you will learn how IT projects get started and how a systems analyst evaluates a *proposed project* and determines its feasibility.
- ❖ If the development process continues, the next step is the systems analysis phase.

Evaluation of Systems Requests

- ❖ Systems Review Committees (Steering committee) A group of key managers and users work on evaluate systems projects or requests.
- ❖ Most large companies use a systems review committee to evaluate systems requests.
- ❖ Many smaller companies rely on one person to evaluate system requests instead of a committee.
- ❖ The goal is to evaluate the requests and set priorities.

Planning the Preliminary Investigation

– During a preliminary investigation, a systems analyst typically follows a series of steps:

1. Understand the Problem or Opportunity.
2. Define the Project Scope and Constraints.
3. Perform Fact-Finding (information gathering).
4. Evaluate Feasibility.
5. Estimate Project Development Time and Cost.
6. Present Results and Recommendations to Management.

Overview of Feasibility

❖ A systems request must pass several tests, called a feasibility study, to see whether it is worthwhile to proceed further.

❖ Types of Feasibility:

- Operational Feasibility means that a proposed system will be used effectively after it has been developed.
- Technical Feasibility refers to technical resources needed to develop, purchase, install, or operate the system.
- Economic Feasibility means that the projected benefits of the proposed system outweigh the estimated costs usually considered the Total cost of ownership (TCO), which includes ongoing support and maintenance costs, as well as acquisition costs.
- Schedule Feasibility means that the project can be implemented in an acceptable time frame.

2- ANALYSIS Phase:

The purpose of the systems analysis phase is to build a logical model of the new system. The first step is requirements modelling (which involves fact-finding to describe the current system and identification of the requirements for the new system), where you investigate business processes and document what the new system must do. Requirements

modelling continue the investigation that began during the systems planning phase. To understand the system, you perform fact-finding using techniques such as interviews, Questionnaires and surveys, document review, observation, sampling, and research. You use the fact-finding results to build business models, data and process models, and object models.

The end product for the systems analysis phase is the system requirements document. The system requirements document describes management and user requirements, costs and benefits, and outlines alternative development strategies.

3- DESIGN Phase:





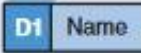
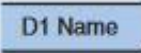


The purpose of the systems design phase is to create a blueprint that will satisfy all documented requirements for the system. At this stage, you design the user interface and identify all necessary outputs, inputs, and processes. In addition, you design internal and external controls, including computer-based and manual features to guarantee that the system will be reliable, accurate, maintainable, and secure. During the systems design phase, you also determine the application architecture, which shows programmers how to transform the logical design into program modules and code.

The result of this phase is documented in the system design specification and presented to management and users for review and approval. Management and user involvement is critical to avoid any misunderstanding about what the new system will do, how it will do it, and what it will cost. All the hardware and software will implement in this phase. The system analyst chooses one of the suggested projects the document it by many tools such as: Data Dictionary, Structure English, and Data flow diagram (DFD).

Data Flow Diagram DFD

The dataflow diagram is also known as a **Process Model**, DFD is a graphical representation of the "flow" of data through an information system.

There are only four types of symbols for DFD – **process**, **dataflow**, **external entity**, **data store**.

Data Flow Diagram Element	Typical Computer-Aided Software Engineering Fields	Gane and Sarson Symbol	DeMarco and Yourdon Symbol
Every <i>process</i> has a number a name (verb phase) a description at least one output data flow at least one input data flow	Label (name) Type (process) Description (what is it) Process number Process description (structured English) Notes		
Every <i>data flow</i> has a name (a noun) a description one or more connections to a process	Label (name) Type (flow) Description Alias (another name) Composition (description of data elements) Notes		
Every <i>data store</i> has a number a name (a noun) a description one or more input data flows one or more output data flows	Label (name) Type (store) Description Alias (another name) Composition (description of data elements) Notes		
Every <i>external entity</i> has a name (a noun) a description	Label (name) Type (entity) Description Alias (another name) Entity description Notes		

1. **Process** (action with number of process).
2. **Data Flow** (→).
3. **Data Store** (to store data, contains (D1, D2, ...))
4. **Entity** – Represent people, student or organizations outside of the system.

Data flow diagram level:

- Data flow diagrams are built in layers.
 - 1- The top level is the **Context level**.
 - 2- The **zero** diagram.
 - 3- The **childe** diagram.
- Processes that do not create a child diagram are called primitive.

The context Diagram

- It contains only one process, representing the entire system.
- The process is given the number zero.
- All external entities are shown on the context diagram as well as major data flow to and from them.
- The diagram does not contain any data stores.
- Each process may be expanded to a lower level.
- The lower level diagram number is the same as the parent process number.

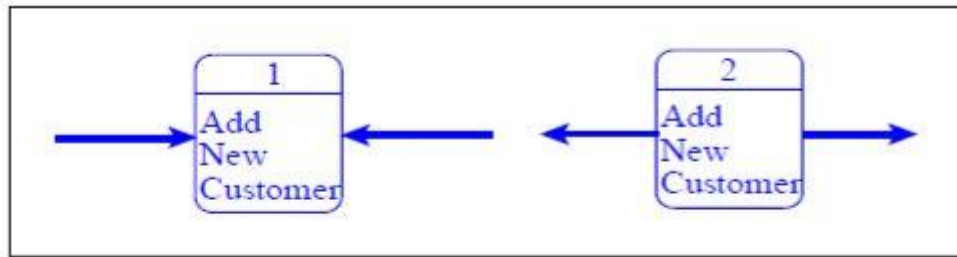
The zero Diagram:

- Diagram 0 is the explosion of the context level diagram.
- It should include maximum 7 or 9 processes.
- Processes are numbered with an integer.
- The major data stores and all external entities are included on Diagram 0.

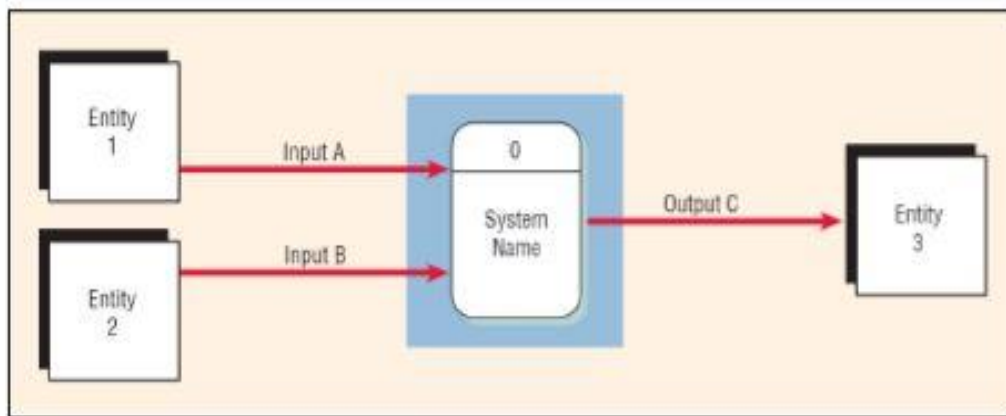
Childe Diagram:

The following conditions are errors that occur when drawing a DFD:

- A process with only input data flow or only output data flow from it.



- Data stores or external entities are connected directly to each other, in any combination.
- Incorrectly labelling data flow or objects
 - Examples :
 - Data flow labelled with a verb.
 - Processes labelled with a noun.
- Too many processes on a data flow diagram.
 - Nine is the suggested maximum.
- Omitting data flow from the diagram
- Unbalanced decomposition between a parent process and a child diagram
 - The data flow in and out of a parent process must be present on the child diagram.



Context Level diagram

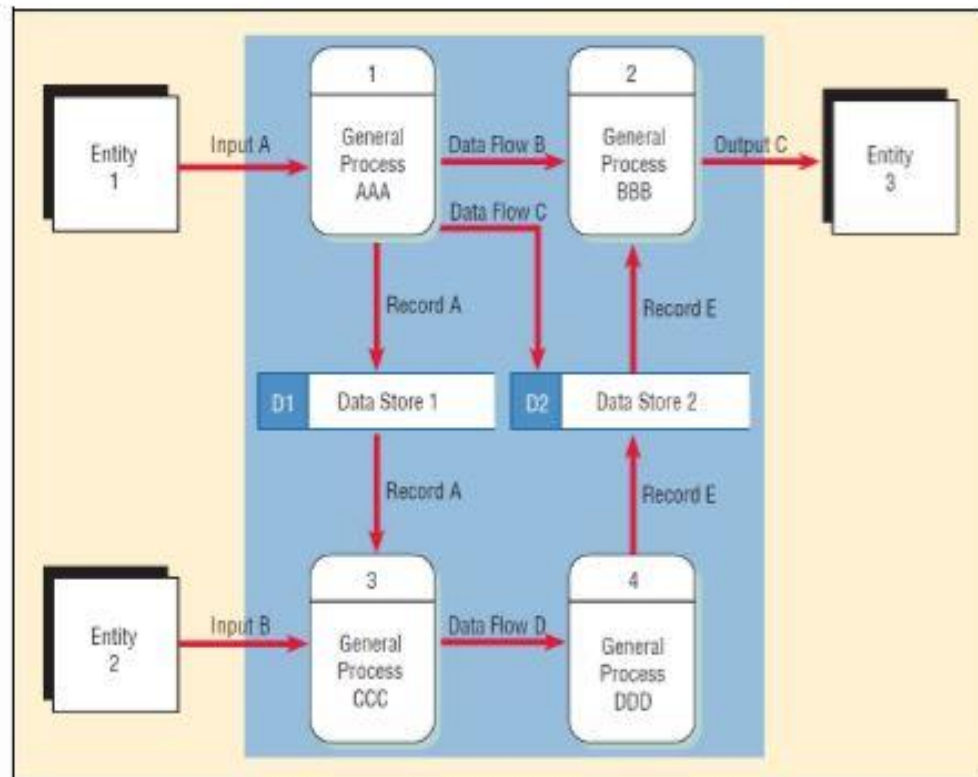
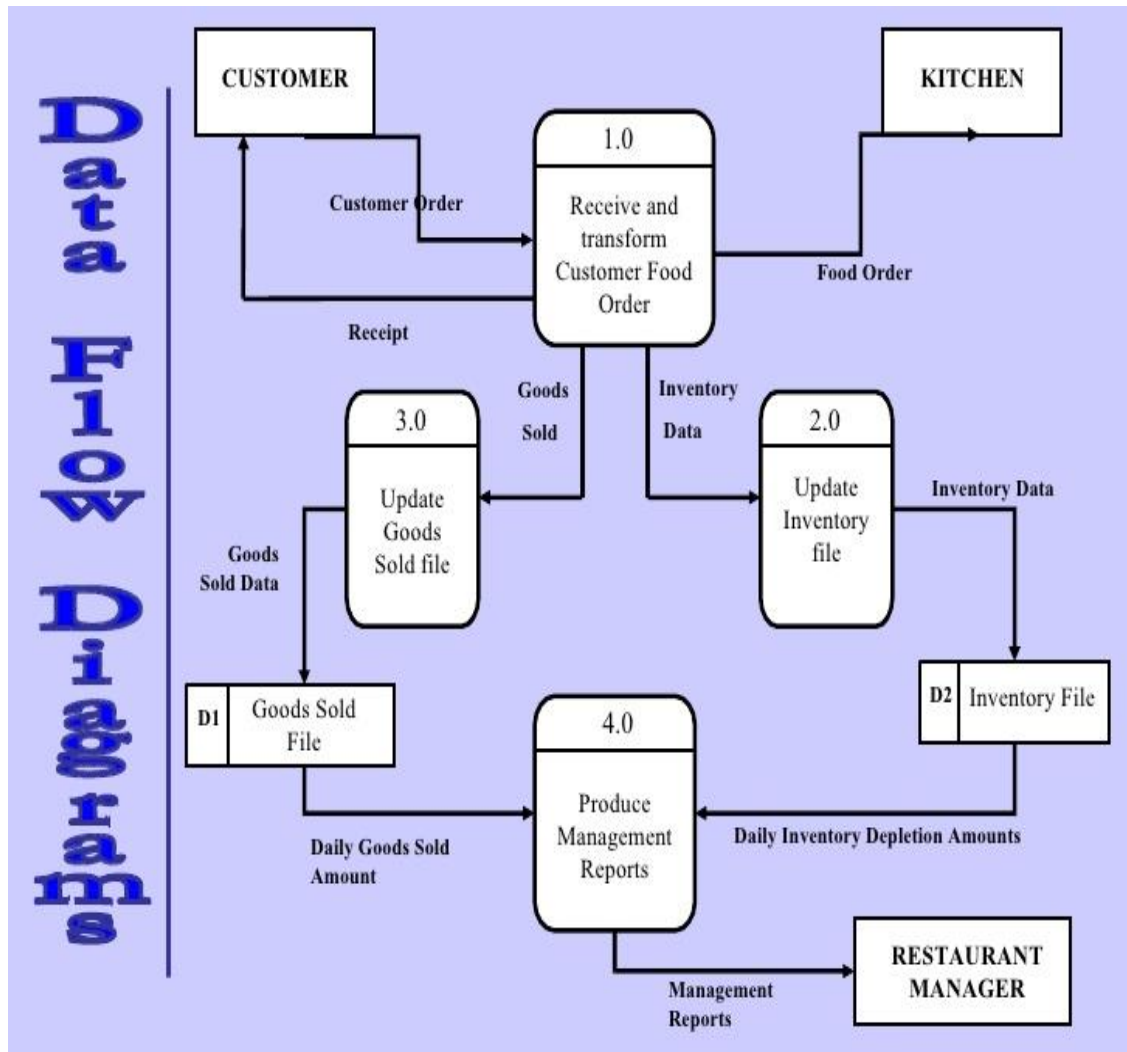


Diagram zero

نظام طلب الطعام بشكل الكتروني (Food Ordering System)

الوصف الاجرائي (Procedural Description)

- 1- يقوم (الزبون) بتقديم طلب الطعام (Customer Order) من خلال نظام (FOS) ثم يحول طلب الطعام (Food Order) الى قسم المطبخ (Kitchen) حيث يكون الطلب جاهزا للتوصيل ثم ارجاع وصل استلام (Receipt) الى الزبون).
- 2- تحديث البيانات (Update) في ملف المخزن (Inventory file) حتى تكون بيانات النفاذ جاهزة الى مدير المطعم او الإدارة (Restaurant Manager).
- 3- تحديث البيانات (Update) في ملف المبيعات (Goods Sold file) حتى تكون بيانات المبيعات جاهزة الى مدير المطعم بشكل يومي.
- 4- انشاء تقرير (Produce Management Reports) حول عملية طلب الطعام التي تمت.



Prototyping

- ❖ Prototyping produces an early, rapidly constructed working version of the proposed information system, called a prototype.
- ❖ Prototyping allows users to examine a model that accurately represents system outputs, inputs, interfaces, and processes

Documentation type

➤ Program Documentation

- Systems analyst usually verifies that program documentation is complete and accurate.

➤ System Documentation

- Includes data dictionary entries, data flow diagrams, object models, screen layouts, source documents, and the systems request that initiated the project.

➤ Operations Documentation

- Such as:
 - Scheduling information for printed output, such as report run frequency and deadlines.
 - Input files and where they originate; and output files and destinations, E-mail and report distribution lists.

➤ User Documentation

- Systems analysts usually are responsible for preparing documentation to help users learn the system.
- Such as: (System overview, Source document, Security and audit, Input, Output, Processing, Requesting changes and reporting problems, Exceptions and error, Frequently asked questions).

4- IMPLEMENTATION Phase:

During the systems implementation phase, the new system is constructed. Whether the developers used structured analysis or O-O methods, the procedure is the same - programs are written, tested, and documented, and the system is installed. If the system was purchased as a package, systems

analysts configure the software and perform any necessary modifications. The objective of the systems implementation phase is to deliver a completely functioning and documented information system.

At the conclusion of this phase, the system is ready for use. Final preparations include converting data to the new system's files, training users, and performing the actual transition to the new system. The systems implementation phase also includes an assessment, called a *systems evaluation*, to determine whether the system operates properly and if costs and benefits are within expectations.

Training Activities

- ❖ A successful information system requires training for the three main groups for training are users, managers, and IT staff members.
- ❖ The entire systems development effort can depend on whether or not people understand the system and know how to use it effectively.

Type of Training:-

- ❖ Outside Training Resources
 - Many training consultants, institutes, and firms are available that provide either standardized or customized training packages.
- ❖ In-House Training
 - The IT staff and user departments often share responsibility.

When developing a training program, you should keep the following guidelines in mind:

- Train people in groups.
- Select the most effective place to conduct the training.
- Provide for learning by hearing, seeing, and doing.

When Training is complete, many organizations conduct a full-scale test, or simulation.

Data Conversion Strategies (System Changeover):

- ❖ System changeover is the process of putting the new information system online and retiring the old system.
- ❖ The *four* Conversion Strategies (changeover) methods are:

➤ Direct Cutover

- Involves more risk than other changeover methods.
- Companies often choose the direct cutover method for implementing commercial software packages.
- Least expensive method because IT group has to operate and maintain only one system at a time.

➤ Parallel Operation

- Easier to verify that the new system is working properly under parallel operation than under direct cutover.
- Running both systems might place a burden on the operating environment and cause processing delay.
- Is not practical if the old and new systems are incompatible technically or the two systems perform different functions.

➤ Pilot Operation

- Involves implementing the complete new system at a selected location of the company. A new sales reporting system, for instance, might be implemented in only one branch office.
- The group that uses the new system first is called (pilot site).
- The old system continues to operate for the entire organization.
- After the system proves successful at the pilot site, it is implemented in the rest of the organization, usually using the direct cutover method. So it is a combination of parallel operation and direct cutover methods.

➤ Phased Operation

- Allows you to implement new system in stages, or modules.
- You give a part of the system to all users
- The risk of errors or failures is limited to the implemented module only.
- Is less expensive than full parallel operation because you have to work with only one part at a time.
- Is not possible, however, if the system cannot be separated easily into logical modules or segments. In addition if the system involves a large number of separate phases, it can cost more than a pilot approach.

5- OPERATION, SUPPORT, SECURITY, and MAINTENANCE Phase

During the systems operation, support, security, and maintenance phase, the IT staff maintains, enhances, and protects the system. Maintenance changes correct errors and adapt to changes in the environment, such as new tax rates. Enhancements provide new features and benefits. The objective during this phase is to maximize return on the IT investment. Security controls safeguard the system from both external and internal threats.

A well-designed system must be secure, reliable, maintainable, and scalable. A scalable design can expand to meet new business requirements and volumes. Information systems development is always a work in progress. Business processes change rapidly and most information systems need to be updated significantly or replaced after several years of operation.