المعهد التقني / موصل

قسم أنظمة الحاسبات

قواعد البيانات

باستخدام

Visual FoxPro

إعداد

ماهر طلال الأسعدي

ماجستير هندسة برمجيات

مدرس مساعد

قسم أنظمة الحاسوب – المعهد التقني / موصل

2015 - 2012

- **قواعد البيانات:** تعرّف قواعد البيانات بأنها حزمة منظمة من البيانات المترابطة منطقياً والتي تتعلق بنشاط معين، كما عرفت بأنها عبارة عن مجموعة كبيرة من البيانات ذات العلاقة نظمت ورتبت في عدد من ملفات البيانات المترابطة بعضها البعض لتشكل بذلك مستودعاً الكترونياً للبيانات. ويتم إدارة هذا المستودع من خلال برمجيات متخصصة تسمى بنظام إدارة قواعد البيانات Data Base Management System(DBMS).

- **Database Management System (DBMS)** :Is a set of programs that manage the process of storing and retrieving data**,** As well as providing users access to the database, They serve as liaison between database users, it receives the requests from users, and then transfer them to the database, and execute the programs to do these requirements, and then provide the user with the desired results. i.e. VFP, Oracle, SQL-SERVER, Access.

- **Data:** It's a facts or raw material, and it can be a text, numbers or images which are stored and processed by computer.

- **البيانات:**_البيان هو حقيقة أولية أو المادة الخام، ويكون عبارة عن نصوص أو أرقام أو صور والتي يتم تخزينها ومعالجتها من قبل الحاسب.

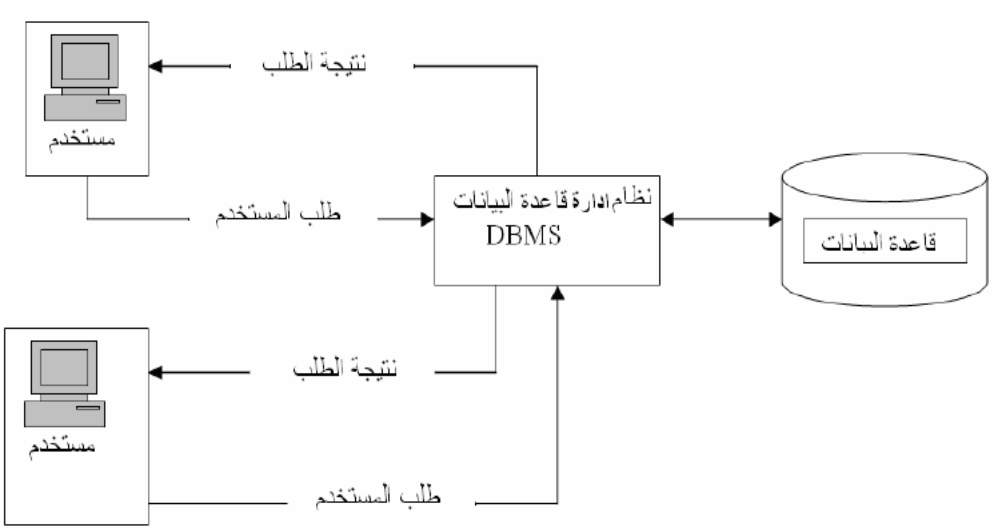
- **Information** :To understand these data, they need to translate or interpret to become information. information is the meaning that is given to the data through interpreted appropriately.

- **المعلومات:** لفهم هذه البيانات، فهي تحتاج إلى ترجمة أو تفسير أو معالجة لتصبح معلومة، فالمعلومات هي المعنى الذي يعطى للبيانات عن طريق تفسيرها بالشكل المناسب.

- **Database:** is an organized collection of data. A large collection of relevant data, organized and arranged in a number of interrelated data files to form an electronic repository of data. This repository is managed through specialized software called Data Base Management System (DBMS).

نظام إدارة قواعد البيانات :

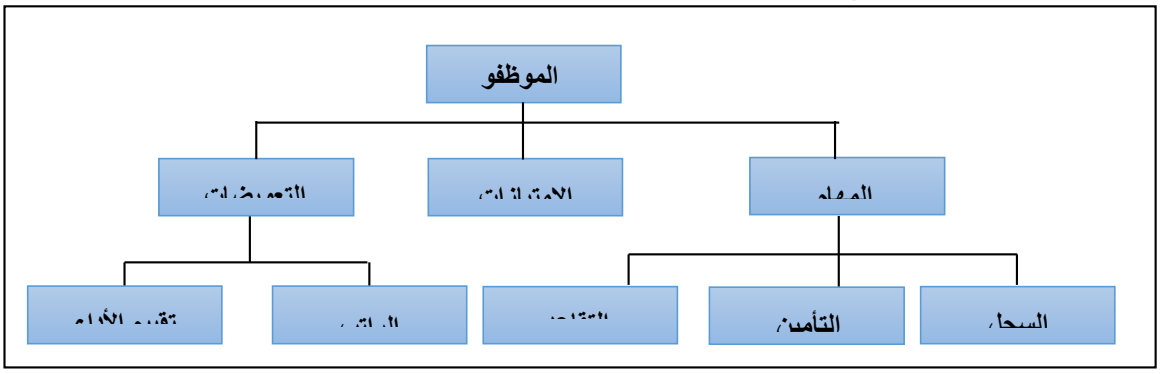
هي مجموعة من البرامج التي تدير عملية تخزين واسترجاع البيانات، وكذلك توفير إمكانية وصول المستخدمين إلى قاعدة البيانات والتعامل معها، وهي تكون حلقة الوصل بين المستخدمين وقاعدة البيانات، إذ تقوم باستقبال طلبات المستخدمين ومن ثم نقلها إلى قاعدة البيانات وتنفيذ البرامج اللازمة لتنفيذ هذه المتطلبات ومن ثم تزويد المستخدم بالنتائج المطلوبة. مثل: VFP, Oracle , SQL-SERVER, Access.



نماذج قواعد البيانات Database Models

1- النموذج الهرمي : Hierarchical Database Management Systems

ظهر هذا النموذج مع نظم الحاسوب الكبيرة، وهو أقدم نموذج لقواعد البيانات المنطقية، وقد صمم هيكله من علاقات بين السجلات التي تشكل هيكل شجري ومستويات هرمية، ولهذا تعبر هذه التركيبة عن نمط العلاقات واحد – إلى كثير (One-to-Many)، وهي تستطيع أن تخزن عدداً كبيراً من الأجزاء، وأن تعالج المعلومات بشكل كبير.



عيوب النموذج الهرمي:

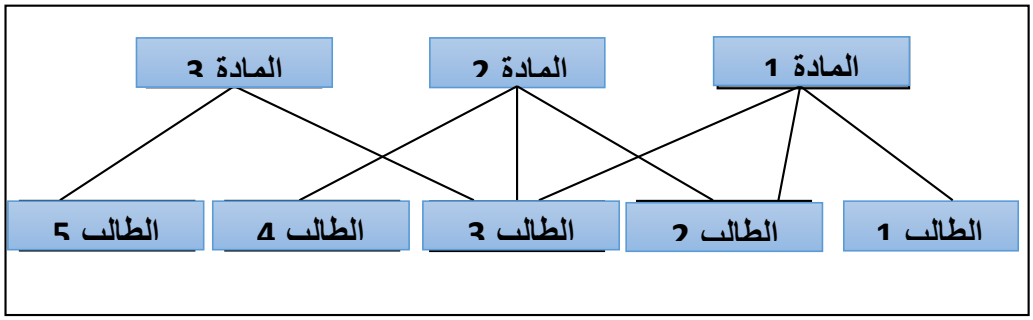
1- يفتقد للمرونة والتجاوب الجيد مع المستخدم.

2- التعقيد في البرمجة.

3- تُخزن البيانات في تركيب هرمي؛ وبالتالي يكون من الصعب إجراء تغيير أو تعديل على هذا التركيب.

2- النموذج الشبكي Network Database Management System

يتم تخزين البيانات في الهيكل الشبكي بصورة سلاسل مترابطة من البيانات؛ وبالتالي يمثل هذا الهيكل علاقات منطقية أكثر تعقيداً. ولا يزال يستخدم مع نظم إدارة قواعد البيانات لنظم الحاسوب الكبيرة، وتمثل هذه التركيبة نمط علاقات الكثير إلى – كثير (Many-to-Many) بين السجلات.

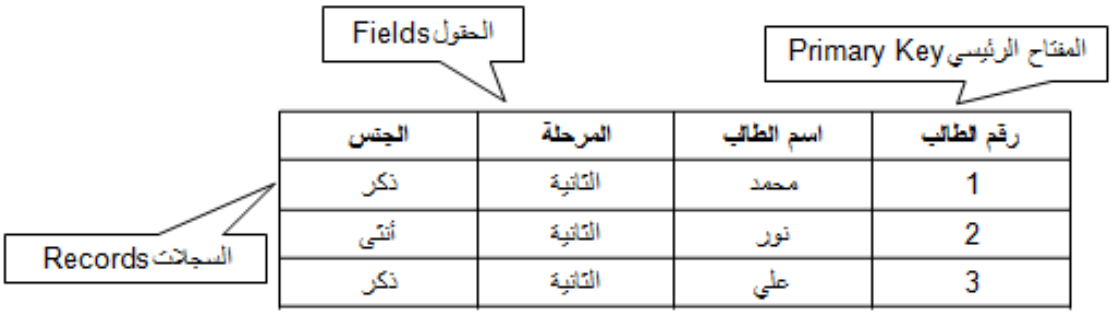


عيوب النموذج الشبكي:

غير مرن ومعقد من الناحية البرمجية والصيانة. إلا انه يعالج المعلومات بشكل كفوء.

3- النموذج العلائقي Relational Database Management System

وهو من أكثر نماذج قواعد قواعد البيانات استخداماً وانتشاراً، ويتكون هيكل قاعدة البيانات في هذا النموذج من جداول Tables تُربط بعلاقات Relations، ويتكون كل جدول من أعمدة Columns تمثل الحقول Fields وصفوف Rows تمثل السجلات Records، ويتم ربط الجدول من خلال الحقول المفتاحية Key Fields (حقل المفتاح الرئيسي Primary Key وحقل المفتاح الثانوي Secondary Key).



رقم الطالب	اسم الطالب	المرحلة	الجنس
1	محمد	الثانية	ذكر
2	نور	الثانية	أنثى
3	علي	الثانية	ذكر

مكونات قواعد البيانات العلائقية Relational Database Components

- **Tables**: is a set of rows which represent records, and a set of columns that represent fields, each row in the table has the same number of fields, but differ in the value, and all fields in the table share the same type and size in a column.

- **الجداول:** هي عبارة عن مجموعة من الصفوف والتي تمثل القيود، ومجموعة من الأعمدة التي تمثل الحقول، وكل صف في الجدول له نفس العدد من الحقول ولكن تختلف في القيمة البيانية، وجميع الحقول في الجدول تشترك بنفس النوع والحجم في العمود الواحد.

- **Field**: represents a set of data elements, and represents distinctive property describing the components of the data. The field may be the name of a student, dept., stage, all these elements are fields in a table of students.

- **الحقل:** يمثل مجموعة من عناصر البيانات، ويمثل خاصية تصف المكونات المميزة للبيانات. والحقل قد يكون اسم طالب، القسم، المرحلة، إذ تعتبر جميع هذه العناصر حقول في جدول الطلبة.

- **Record**: is a group of fields in the table, although a group of fields student's name, dept., and stage represents a one record to a student in the student table.

- **السجل:** هو مجموعة من الحقول في الجدول، وإن مجموعة حقول اسم الطالب، القسم ، والمرحلة تمثل سجلاً واحد لطالب في جدول الطلبة.

- **Primary Key:** Each record is marked by a field called primary key, and this record cannot be repeated and unique to each student, and may not be an empty. And the access to the student's record can be done through this key field.

- **المفتاح الرئيسي .P.K:** يتم تميز كل سجل من خلال حقل مفتاحي يسمى المفتاح الرئيسي، وهذا السجل لا يمكن أن يتكرر وينفرد به كل طالب، كما لا يجوز أن يكون حقلاً فارغاً. ويتم الوصول إلى سجل الطالب من خلال هذا الحقل المفتاحي .

- **Relations**: it's the linking between tables together by a common factor among these tables.

- **العلاقات:** هي التي تربط الجداول مع بعضها عن طريق عامل مشترك بين هذه الجداول.

مستخدمو قواعد البيانات Database Users

- **مدير قواعد البيانات DB Administrator**

Who manages databases, control the permissions, monitor the system and improve the performance of databases.

وهو الذي يقوم بإدارة قواعد البيانات والتحكم في صلاحيات العمل ومراقبة النظام وتحسين أداء قواعد البيانات.

- **مصمم قواعد البيانات DB Designer**

He is designing databases to be created and built with highly efficient manner, according to user requirements.

وهو الذي يقوم بتصميم قواعد البيانات ليتم إنشائها وبنائها بطريقة ذات كفاءة عالية طبقاً لمتطلبات المستخدم.

- **مستخدم قواعد البيانات DB End User**

Some users have a sufficient experience to the preparation of the inquiries by query language, and others have no experience therefore a special programs are created for them.

بعض المستخدمين يكون لديهم الخبرة الكافية لإعداد الاستفسارات المطلوبة بلغة الاستفسارات، وبعض المستخدمين ليس لديهم الخبرة فيتم إنشاء برامج خاصة لهم يقومون بتشغيلها للحصول على المطلوب.

مقارنة قواعد البيانات مع أنظمة الملفات التقليدية Database vs. Files

- نظراً لقصور الأنظمة اليدوية في الوصول إلى متطلبات المؤسسات بجميع أنواعها، فقد تم استخدام أنظمة الملفات التقليدية، وهو نظام محوسب يعتمد على تخزين البيانات في ملفات مستقلة بحيث تكون البيانات معزولة عن بعضها البعض.
- أن أنظمة الملفات التقليدية تشكل تطوراً أفضل مقارنة مع الأنظمة اليدوية، إلا أنها تعاني من بعض القصور، ومن أهم جوانب هذه القصور ما يلي :
  - التكرار
  - عدم توافقية البيانات
  - زيادة زمن بناء الأنظمة
  - الحاجة المستمرة لإعادة هيكلة البرامج والملفات

مميزات استخدام قواعد البيانات Database Features

1. التقليل من تكرار البيانات Reduction of Data Redundancy: يقصد بتكرار البيانات تخزين البيانات نفسها لأكثر من مرة، وقد أدى استخدام قواعد البيانات إلى الحد من هذه المشكلة.
2. تجنب التناقض في البيانات Avoidance of Data Inconsistency: عند حفظ البيانات في أكثر من ملف ووزعت في أكثر من موقع واحد، فان إجراء تعديل على بيانات في احد هذه المواقع قد يودي إلى بقاء نفس البيانات على حالها في المواقع الأخرى مما يتسبب بعدم تجانس البيانات التي تخص حقيقة معينة.
3. المشاركة في البيانات Data Sharing: وتعني السماح لأكثر من مستخدم بالوصول إلى البيانات الموجودة في القاعدة بنفس الوقت.
4. تطبيق الأمنية والسرية للبيانات Data Privacy and Security: يقصد بأمن البيانات هو حمايتها من الدخول غير المشروع، أو العبث بها وضياعها، ويعد أمن البيانات خاصية مهمة لنظم إدارة قواعد البيانات.
5. تكامل البيانات Data Integrity: ويقصد بها وضع نقاط تحقق وتدقيق لتجنب الإدخال أو التحديث غير الصحيح، إضافة إلى ضمان عدم حدوث تناقض في البيانات المخزونة.
6. إمكانية تطبيق مبدأ الاستقلالية Data Independence: وهو يعني تنظيم البيانات على وسائل الخزن، وتحديد أسلوب الوصول للبيانات بمعزل عن متطلبات التطبيق. إذ أن نظم إدارة قواعد البيانات فصلت قواعد البيانات عن البرامج التي تستخدمها، أي أن البرامج لن تتأثر بإعادة تنظيم البيانات.

مفهوم الحقول المفتاحية Introduction to Key Fields

- Key field is a data element or field that is used to identify the records, retrieval and indexing data stored. It may be a simple key, which consists of a one field (column), or it may be the composite key consisting of several fields (columns).

- الحقل المفتاحي هو عنصر من عناصر البيانات أو حقل يستخدم لتعريف السجلات، واسترجاع وفهرسة المعلومات المخزنة. وقد يكون المفتاح بسيطاً، أي يتكون من حقل أو عمود واحد فقط، أو قد يكون المفتاح مركباً، أي مكون من عدة حقول أو أعمدة.
- **ومن أنواع المفاتيح هي:**

1- المفتاح الرئيسي Primary Key

2- المفتاح الأجنبي Foreign Key

3- المفتاح المرشح Candidate Key

4- المفتاح المركب Composite Key

5- المفتاح الأعظم Super Key

6- المفتاح الطبيعي Natural Key

7- المفتاح البديل Surrogate Key

1- المفتاح الرئيسي Primary Key

- The primary key is the field that identify each table in the database, and identifies each record in a table uniquely.

- يعد المفتاح الرئيسي حقل هوية الجدول في قاعدة بيانات، ويقوم المفتاح الرئيسي بتعريف كل سجل في الجدول بشكل فريد.
- ومن مميزات الحقل الرئيسي هي:

1. إنشاء فهرس للمفتاح الرئيسي تلقائياً؛ مما يؤدي إلى تسريع عمليتي استرجاع وفرز البيانات.
2. بناء العلاقات بين الجداول.
3. يتم عرض السجلات مرتبة حسب المفتاح الرئيسي للجدول.
4. عدم السماح بتكرار السجلات.
5. عدم السماح بترك القيمة فارغة للسجل Null.

ومن المسائل التي يجب التقيد بها عن اختيار حقل المفتاح الرئيسي هي:

1. **اختيار حقل لا تتكرر البيانات بداخله (Unique).**

فعلى سبيل المثال، من الخطأ استخدام حقل اسم الزبون من جدول الزبائن ليصبح المفتاح الرئيسي؛ وذلك لأن الاسم قد يتكرر عند استخدام قاعدة بيانات كبيرة، لذلك فإننا سنستخدم رقم الهوية كمفتاح رئيسي لعدم تكرار هذا الحقل في كل الجدول.

2. **اختيار حقل لا يمكن أن يحتوي على قيمة فارغة (Not Null).**

فعلى سبيل المثال، من الخطأ استخدام حقل رقم هاتف الزبون من جدول الزبائن ليصبح المفتاح الرئيسي؛ وذلك لأن ذلك الحقل قد يكون فارغاً وليس إجبارياً.

3. **اختيار حقل يحتوي على عدد قليل من البيانات (Smallest Key).**

فعلى سبيل المثال، قد تستخدم القيمة الموجودة في حقل المفتاح الرئيسي للبحث عن السجلات؛ لذلك يراعى ألا يحتوي على عدد كبير من الأرقام أو الأحرف. وأن حجم المفتاح يؤثر على سرعة العمليات في قاعدة البيانات.

customer_id	customer_name	address	phone
01	Mohammed	Mosul	09567
02	Ali	Baghdad	07654
03	Saad	Baghdad	08654
04	Ali	Basrah	

customer_id	customer_name	address	phone
01	Mohammed	Mosul	09567
02	Ali	Baghdad	07654
03	Saad	Baghdad	08654
04	Ali	Basrah	

2- المفتاح الثانوي (الأجنبي) Foreign Key

It is a field that represents the primary key in another table, it called a foreign keys because it is not from an existing fields in the table, but its added to the table to link with another table.

وهو عبارة عن حقل يمثل المفتاح الرئيسي في جدول آخر، وسُمي المفتاح الأجنبي بهذا الاسم لأنه ليس من الحقول الموجودة أصلاً في الجدول، ولكن هو حقل مضاف إلى الجدول لربطه مع جدول آخر.

ومن مميزات الحقل الثانوي هي:

1. لا يشترط أن يكون المفتاح الثانوي بنفس اسم المفتاح الرئيسي، ولكن يشترط أن يكون مطابقاً للمفتاح الرئيسي في الجدول الأول من حيث النوع، والحجم، وعدد الحقول.
2. لا يشترط عدم تكرار قيمته في الجدول الثاني.

customer_id	customer_name	address	phone
01	Mohammed	Mosul	09567
02	Ali	Baghdad	07654
03	Saad	Baghdad	08654
04	Ali	Basrah	

order_id	cust_id	order_name	order_date
001	01	a	2011-01-05
002	02	b	2010-07-10
003	02	c	2009-05-05
004	04	d	2011-11-17
005	01	e	2006-12-07

نلاحظ من المثال السابق أن رمز الزبون في ملف الزبائن عبارة عن مفتاح رئيسي، أما رمز الزبون في ملف الطلبيات فهو مفتاح أجنبي، وتمت إضافته للربط بين الجدولين، ولتحديد الطلبية التي تتبع له.

3- المفتاح المرشح Candidate Key

- It holds the same properties of primary key, but it is not a primary key for the table, therefore its candidate to be a primary key.
- وهو يحمل خصائص المفتاح الرئيسي ولكنه ليس مفتاح رئيسي للجدول، ولهذا فهو مرشح لكي يكون المفتاح الرئيسي للجدول.
- عند البدء بتصميم الجدول يتم ترشيح عدد من الحقول كي تصبح مفاتيح رئيسية، وعند إدخال البيانات، قد يتبين أن هذه المفاتيح يمكن أن تأخذ قيمة Null، فالمفتاح الذي يأخذ قيمة Null يُستثنى، والمفاتيح التي لا تأخذ قيمة Null ولا تتكرر تبقى وتصبح مفاتيح رئيسية.
- فإن المفتاح المرشح هو الصفة أو مجموعة الصفات التي يتم اختيارها وفحصها حتى يتقرر فيما بعد أنها ستبقى مفاتيح مرشحة أو يتم اعتمادها كمفتاح رئيسي .

customer_id	customer_name	address	phone
01	Mohammed	Mosul	09567
02	Ali	Baghdad	07654
03	Saad	Baghdad	08654
04	Ali	Basrah	

نلاحظ من المثال السابق أن رمز الزبون ورقم الهاتف مفاتيح مرشحة لتكون رئيسية، ولكن تبين فيما بعد أن رقم الهاتف ممكن أن يكون فارغاً ولهذا سوف يستبعد من كونه رئيسياً.

4- المفتاح المركب Composite Key

It is the key field that is used to identify a record uniquely, but differs from the primary key that includes more than one attribute.

وهو المفتاح الذي يستخدم لتعريف السجل بشكل وحيد ومتفرد ولكنه يختلف عن المفتاح الرئيسي بأنه يشمل أكثر من صفة.

stud_name	project	mark
Mohammed	DB	75
Ali	VB	80
Saad	DB	69
Ali	OS	45

نلاحظ في المثال السابق، أنه لا يمكن اعتبار اسم الطالب واسم المشروع أو الدرجة كمفتاح رئيسي يحدد السجل بشكل وحيد ومتفرد، فيتم اللجوء في هذه الحالة إلى اعتبار اسم الطالب مع اسم المشروع مفتاح مركب، على اعتبار أن اسم الطالب قد يتكرر واسم المشروع قد يتكرر، ولكن اسم الطالب مع اسم المشروع كمفتاح مركب لن يتكرر.

5- المفتاح الأعظم Super Key

It's a less number of attributes that can be distinguished a record in the table from the rest of the other records.

هو اقل عدد من الصفات التي يمكن أن تميز السجل في الجدول عن بقية السجلات الأخرى، فمثلاً هذه الصفات يمكن أن تكون المفتاح الأعظم.

Student File

Su per key

1.st.no

2.st.name

3.dep

St.no	St.name	dep	birth
0001	ali	math	1980
0002	ahmed	eng	1990
0003	jasem	arabic	1991

6- المفتاح الطبيعي Natural Key

- A field that consists of the attributes that represent some of the features in the real world. For example, employee ID number, and credit card number, are numbers those exist in the real world. If the key has these qualities, it is considered as a natural key.

- وهو الحقل الذي يتكون من الصفات التي تمثل بعض الخصائص الموجودة في العالم الحقيقي. فعلى سبيل المثال، رقم هوية الموظف، ورقم البطاقة الائتمانية، هي أرقام موجودة في العالم الحقيقي. وإذا كان المفتاح الرئيسي يحتوي على هذه الصفات، فهو يعتبر مفتاحاً طبيعياً.

customer_name	address	phone	credit_card_no
Mohammed	Mosul	09567	009-7665-998
Ali	Baghdad	07654	005-8465-932
Saad	Baghdad	08654	003-1265-548
Ali	Basrah		007-1541-638

7- المفتاح البديل Surrogate Key

- In some situations it may be the natural key is unavailable in the table, or it may be too long, these problems are not being it to be a primary key, in such cases we can use the alternative key or artificial key to be the primary key, it can be added manually or automatically by the computer. For example, you can configure AutoNumber to the customer ID field.

- في بعض الأحيان قد يكون المفتاح الطبيعي غير متوافراً في الجدول، أو قد يكون طويلاً، وهذه المشاكل لا تمكنه من أن يكون مفتاحاً رئيسياً، ففي هذه الحالات يمكن استخدام المفتاح البديل أو المفتاح المصطنع (Artificial Key) ليكون المفتاح الرئيسي، وهو عبارة عن حقل جديد يتم إضافة بياناته يدوياً أو ألياً عن طريق الحاسبة. على سبيل المثال، يمكن تكوين ترقيم تلقائي لحقل رمز الزبون.

customer_id	customer_name	address	phone	credit_card_no
01	Mohammed	Mosul	09567	009-7665-998-6655
02	Ali	Baghdad	07654	005-8465-932-6443
03	Saad	Baghdad	08654	003-1265-548-6744
04	Ali	Basrah		007-1541-638-4533

أمثلة لنظم قواعد البيانات Database Systems Examples

- **نظام المكتبة Library System**

User File

user_id	user_name	user_address	phone

Book File

ISBN	book_title	author	price

Borrowing File

bor_id	user_id	ISBN	date

- **نظام تجاري Commerce System**

Customer File

customer_id	customer_name	customer_addr	phone

Order File

order_id	order_date	order_name	customer_id

Product File

prod_id	prod_name	price	order_id

- **نظام البنك Bank System**

User File

user_id	user_name	user_address	phone

Account File

acc_no	user_id	dept_name	ammout

check File

check_no	user_id	acc_no	ammout

- **نظام العيادات Clinic System**

Patient File

pat_id	pat_name	state	address	doc_id

Doctor File

doc_id	pat_id	doc_name	spec

Report File

rep_id	pat_id	doc_id	state

- **نظام التدريس Teaching System**

Student File

stud_id	stud_name	stage	address

Lecture File

lec_id	lec_name	spec	cert

Subject File

sub_id	sub_name	lec_id

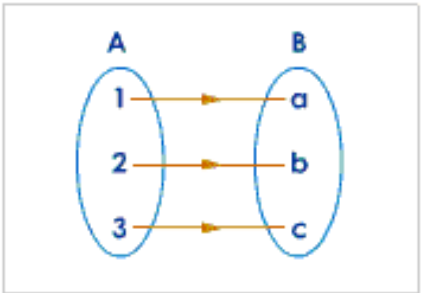
مفهوم العلاقات Introduction to Relations

- The relation is the sharing of a field between the two tables, therefore every value have two records, the first record in the first table and the second record in the other table.

- يقصد بالعلاقة اشتراك حقل بين جدولين، بحيث تملك كل قيمة في هذا الحقل سجلين، السجل الأول في الجدول الأول والسجل الثاني في الجدول الآخر.
- ويعد ربط الجداول أمر ضروري لأن التصميم الجيد لقاعدة البيانات يتطلب منك أن تنشئ جداول صغيرة يشتمل كلا منها على بيانات ذات طبيعة واحدة.
- وربط الجداول يعني إنشاء علاقة ارتباط دائمة بين جدولين أو أكثر، ويكون من نتيجتها استخراج بيانات من كلا الجدولين وإظهارها في نماذج أو تقارير أو استعلامات.
- ويمكن ربط جدولين إذا كان كليهما يشتمل على حقل أو أكثر بهما نفس البيانات، وعادة تسمى الحقول في كلا الجدولين بنفس الاسم. مثل رمز الزبون في جدول بيانات الزبائن ورمز الزبون في جدول الطلبيات.
- ولإيجاد أو إنشاء علاقة، سوف نستخدم مفتاحين هما: المفتاح الرئيسي (Primary Key) والمفتاح الأجنبي (Foreign Key).
- **وأنواع علاقات الارتباط هي:**
- علاقة واحد إلى واحد One-to-One (1−1)
- علاقة واحد إلى متعدد One-to-Many (1−N) or (N-1)
- علاقة متعدد إلى متعدد Many-to-Many (N-N)

1- علاقة واحد إلى واحد One-to-One

- في هذه العلاقة، كل سجل في الجدول الرئيسي Primary Table يقابله سجل واحد في الجدول المرتبط به Related Table.
- ولا يعد هذا النوع من العلاقة شائعاً، لأن معظم المعلومات المرتبطة بهذه الطريقة تكون في جدول واحد.
- وقد تستخدم لتقسيم جدول يحتوي على عدة حقول، أو لعزل جزء من جدول لأسباب أمنية.
- وتنشأ علاقة One-to-One إذا كان كلا الحقلين المرتبطين مفاتيح رئيسية.



- ومن الأمثلة التي تستخدم فيها علاقة One-to-One، إذا كان السجل 01 يمثل المسافر Mohammed فإن السجل 96 يمثل مقعده في الطائرة. وهكذا فأن لكل مسافر مقعده الخاص والوحيد في الطائرة .

pass_id	pass_name	address
01	Mohammed	Mosul
02	Ali	Baghdad
03	Saad	Baghdad
04	Ali	Basrah

seat_no	seat_type
96	A
32	A
74	B
25	C

- ومثال آخر، إذا كان السجل 01 يمثل الشخص Mohammed فإن السجل A9656 يمثل جواز سفره. وهكذا فأن لكل شخص جواز سفره الخاص والوحيد.

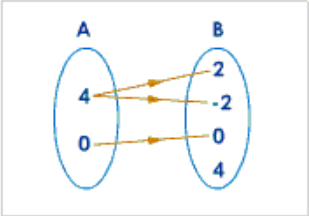
person_id	person_name	address	phone
01	Mohammed	Mosul	09567
02	Ali	Baghdad	07654
03	Saad	Baghdad	08654
04	Ali	Basrah	

passport_no	passport_type	exp_date
A9656	A	2015-01-05
G5432	G	2013-07-10
G7643	G	2014-05-05
S7425	S	2016-11-17

2- علاقة واحد إلى متعدد One-to-Many

- وهي العلاقة الأكثر استخداماً بين الجداول، وتعني أن السجل الواحد في الجدول الرئيسي يقابله أكثر من سجل في الجدول المرتبط.
- يتم إنشاء علاقة One-to-Many إذا كان أحد الحقول المرتبطة مفتاح رئيسي والآخر مفتاح أجنبي.

- فمثلا معلومات الموظف وأولاده هي واحد لمتعدد.



- وكمثال على استخدام علاقة One-to-Many، هي علاقة الطلاب بمشاريع التخرج، أي لدى كل مجموعة من الطلاب مشروع تخرج واحد، ولا يجوز للطالب عمل أكثر من مشروع.

stud_id	stud_name	project_id
01	Mohammed	003
02	Ali	003
03	Saad	001
04	Noor	001
05	Ali	002
06	Reem	002

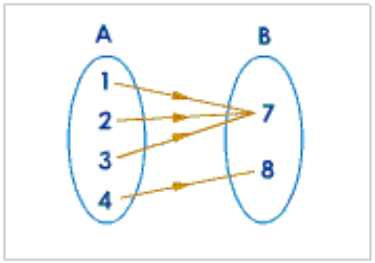
project_id	project_name
001	Resources management
002	Library System
003	Web Application
004	Pharmacy System

- ومثال آخر هي علاقة الزبون بالطلبية، فكل زبون يمكن أن يطلب أكثر من طلبية. ولا يمكن للطلبية من أن تعود لأكثر من زبون.

customer_id	customer_name	address	phone
01	Mohammed	Mosul	09567
02	Ali	Baghdad	07654
03	Saad	Baghdad	08654
04	Ali	Basrah	

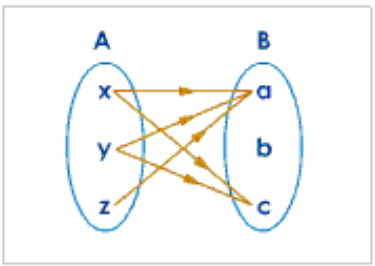
order_id	cust_id	order_name	order_date
001	01	a	2011-01-05
002	02	b	2010-07-10
003	02	c	2009-05-05
004	04	d	2011-11-17
005	01	e	2006-12-07

- أما إذا كانت عدة سجلات مرتبطة بسجل واحد فتسمى العلاقة علاقة متعدد إلى واحد Many-to-One كما في الشكل. وهي عكس العلاقة السابقة، فعلاقة الأولاد بالأب هي متعدد إلى واحد.

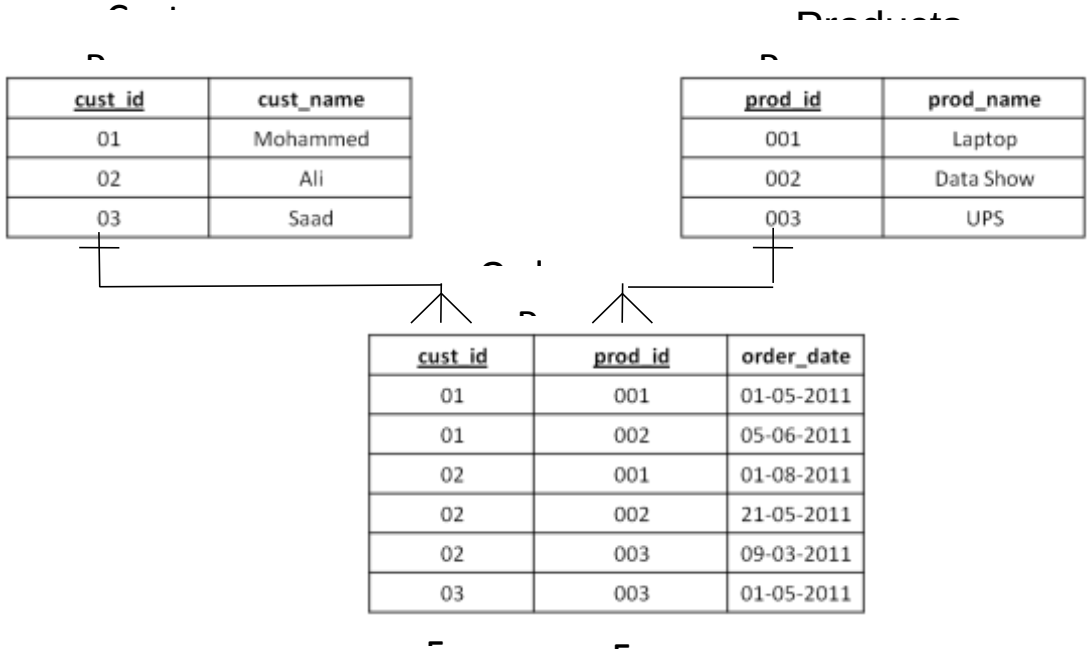


3- علاقة متعدد إلى متعدد Many-to-Many

- في هذه العلاقة، يقابل سجل من الجدول الرئيسي عدة سجلات في الجدول المرتبط، ويقابل سجل في الجدول المرتبط عدة سجلات في الجدول الرئيسي.
- هذا النوع من العلاقات معقد، لذا يجب ربط الجدولين بأسلوب غير مباشر يتلخص في إنشاء جدول ثالث يعمل على تجزئة علاقة Many-to-Many إلى علاقتين من نوع One-to-Many، وفي هذه الحالة تضع المفتاحين الرئيسيين لكلا الجدولين في الجدول الثالث. ويكون المفتاح الرئيسي للجدول الجديد مكون من المفتاحين الرئيسيين للجدولين الآخرين.



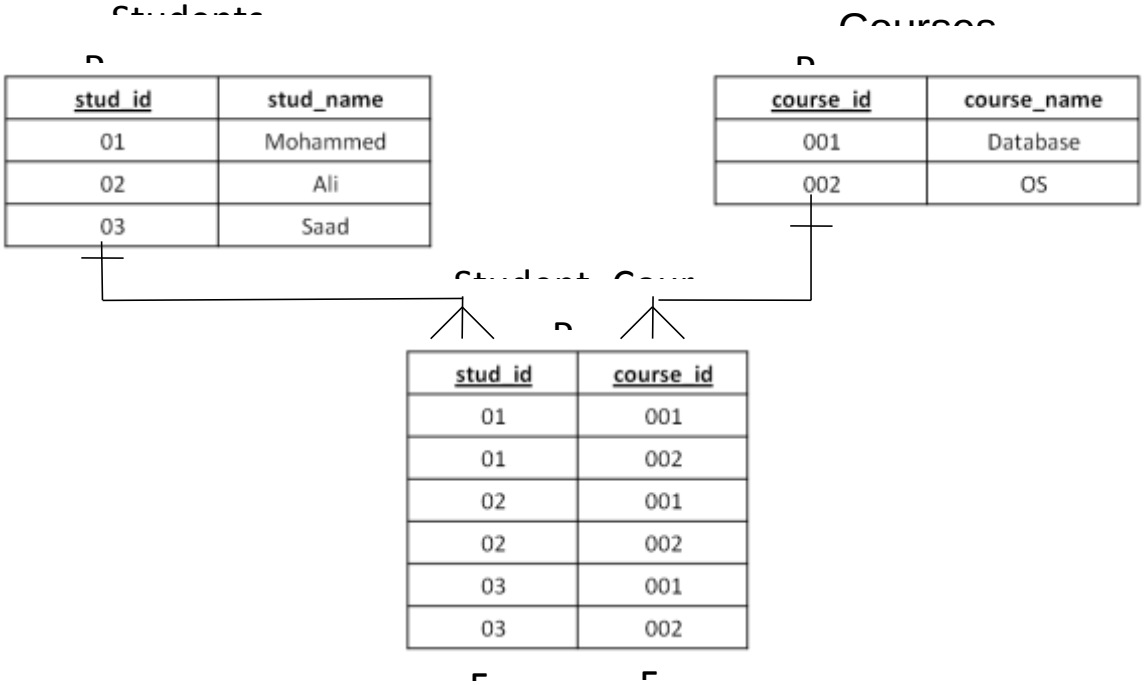
- وكمثال على هذه العلاقة، علاقة الارتباط بين جدول "الزبائن Customers"، وجدول "المنتجات Products"، فأي زبون يمكن أن يشتري أي منتج والعكس صحيح، أي منتج يمكن أن يُباع لأي زبون.
- يتم تجزئة علاقة Many-to-Many إلى علاقتين من نوع One-to-Many، وذلك بإضافة جدول ثالث هو جدول "الطلبيات Orders" وبذلك تصبح العلاقة بين كل جدول والجدول الذي يتعامل معه هي علاقة One-to-Many.



cust_id	cust_name
01	Mohammed
02	Ali
03	Saad

prod_id	prod_name
001	Laptop
002	Data Show
003	UPS

cust_id	prod_id	order_date
01	001	01-05-2011
01	002	05-06-2011
02	001	01-08-2011
02	002	21-05-2011
02	003	09-03-2011
03	003	01-05-2011



stud_id	stud_name
01	Mohammed
02	Ali
03	Saad

course_id	course_name
001	Database
002	OS

stud_id	course_id
01	001
01	002
02	001
02	002
03	001
03	002

دورة حياة قاعدة البيانات Database Life Cycle (DBLC)

- The developing process of database system undergoing in a series of stages, these stages are called the database life cycle.
- إن عملية تطوير قاعدة البيانات تمر بمجموعة من المراحل، هذه المراحل المتتالية تسمى بدورة حياة قاعدة البيانات.
- تتكون دورة حياة قاعدة البيانات من المراحل التالية:

1. مرحلة تحديد المتطلبات Requirement Gathering Phase:

تبدأ هذه المرحلة بدراسة نظام عمل المؤسسة وبيئتها وتحديد متطلبات النظام من وجهة نظر المستخدم، وهذا يتم من خلال:

1) تحديد البيانات التي ستخزن في ملفات القاعدة وتحديد طبيعتها وماهيتها.
2) وضع معايير لوصف البيانات (شكلها، نوعها، حجمها).
3) تحديد رؤى المستخدمين وحاجتهم من البيانات.
4) تحديد متطلبات بناء وتشغيل النظام من أجهزة وبرمجيات وكوادر متخصصة.
5) يكون ناتج المرحلة عبارة عن وثيقة أو تقرير يحدد متطلبات النظام.

2. مرحلة تحليل ونمذجة البيانات Data Analysis and Modeling Phase:

يتم في هذه المرحلة تكوين تصور منطقي للشكل الذي ستكون عليه البيانات، من خلال القيام بما يلي:-

1) إعداد قاعدة البيانات الأولية Conceptual Database وفيها يتم تصميم نموذج اولي للبيانات بواسطة مخططات الكيانات/العلاقات Entity Relationship Diagram (ERD).
2) تحديد وتعريف العلاقات التي تربط مابين عناصر البيانات.
3) عرض النموذج على المستفيدين من النظام لتقديم مقترحاتهم.
4) ويكون ناتج هذه المرحلة بناء ما يسمى بالنموذج المفاهيمي (المنطقي).

3. مرحلة تصميم قاعدة البيانات Database Design Phase

بعد أن يتم الاتفاق على النموذج المقترح لقاعدة البيانات، يتم في هذه المرحلة المباشرة بالأمور التالية:

1) كتابة الوصف المنطقي وكذلك إعداد البرامج اللازمة لانجاز التصميم.
2) وتصميم قاعدة البيانات المنطقية Logical Database وهي تحويل قاعدة البيانات الأولية إلى مخطط البيانات DB Schema وذلك بإتباع قواعد التحويل Mapping Rules.
3) تحسين قاعدة البيانات المنطقية، وذلك بتطبيق قواعد تطبيع البيانات Normalization التي تهدف إلى تقليل تكرارية البيانات، من أجل رفع كفاءة قاعدة البيانات ما أمكن.
4) وينتج عن هذه المرحلة التوصل إلى الهيكل النهائي لقاعدة البيانات.

4. مرحلة تنفيذ قاعدة البيانات Database Implementation Phase

- وفي هذه المرحلة يتم وضع الهيكل المقترح لقاعدة البيانات موضع التنفيذ بما يؤدي إلى بناء الهيكل الداخلي لقاعدة البيانات ويضمن تحديد استراتيجيات الخزن وطرق الوصل والأساليب التي تتبع في استدعاء سجلات البيانات. إذ يتم في هذه المرحلة كتابة أكواد إنشاء قاعدة البيانات بلغة SQL ويحدد فيها بنية الجداول ونوع بيانات الحقول والمفاتيح الاساسية والاجنبية وباقي شروط

تصميم قاعدة البيانات، ثم تنفيذ ذلك ضمن مدير قاعدة بيانات DBMS مناسب، مثل: (oracle, access, sqlserver, FoxPro…. etc).

5. مرحلة فحص اداء قاعدة البيانات Database Testing Phase

- بعد وضع قاعدة البيانات موضع التنفيذ لابد من إخضاعها للمراقبة والفحص لاكتشاف نقاط الضعف في النموذج المقترح، وإجراء التعديلات اللازمة بما يضمن التوصل إلى نظام متكامل.

أنواع البيانات في قواعد البيانات Database Data Types

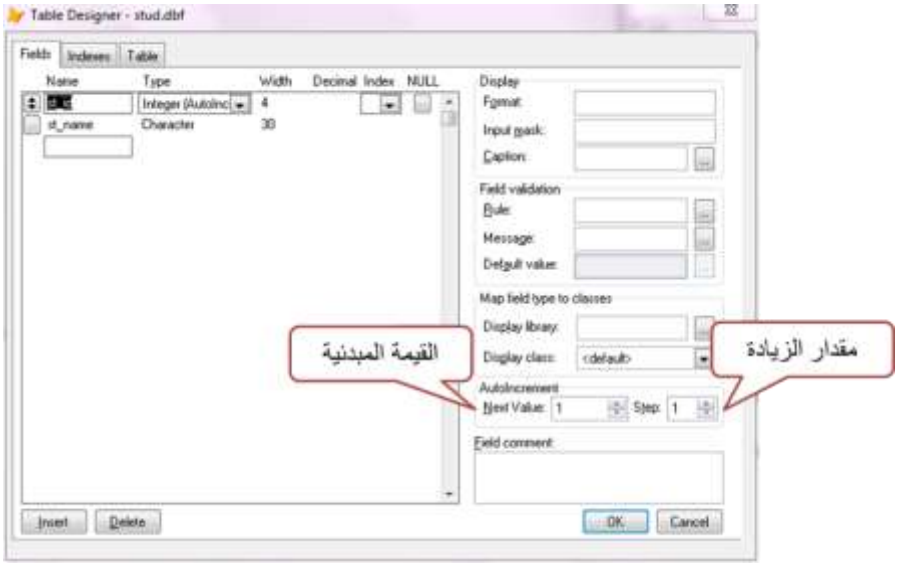
- **Numeric**: يستخدم لحفظ الأرقام ويمكن أن يحوي على الفاصلة العشرية وإشارة السالب.
  - الحجم (8) بايت.
  - أمثلة: (9000) ، (-500) ، (5.05) …

- **Double**: لحفظ الارقام الدقيقة جداً والتي تستخدم الفاصلة العشرية بحجم اكبر.
  - الحجم (8) بايت.
  - أمثلة: (0.000001433)، (-0.006554) …

- **Float**: نفس عمل نوع Numeric، وتغير اسمه ليتلائم مع النظم الاخرى.

- **Integer**: نفس نوع Numeric ولكن بدون فاصلة عشرية، أي عدد صحيح.
  - الحجم (4) بايت.

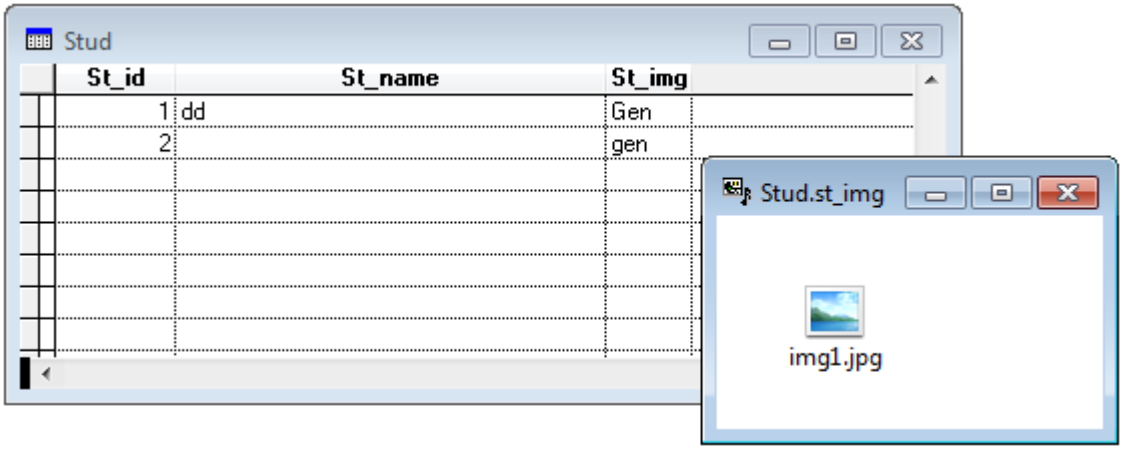
Integer (Autoinc): نفس نوع Integer ولكن القيمة التي بداخله سوف تزداد تلقائياً عند إضافة سجل جديد إلى الجدول، والقيمة المبدئية والزيادة يمكن أن يحددها المستخدم.



- **Character**: يستخدم لحفظ النصوص والأرقام والرموز ويتعامل معها على شكل نص، وهي التي لا تتعامل مع الحسابات الرياضية، مثلاً يمكن حفظ رقم الهاتف على شكل نص.
  - الحجم (1) بايت لكل رمز وصولاً إلى (254) كحد أقصى.
  - أمثلة: "Ahmed"، "40GB"، "50%" ...
- **Memo**: يستخدم لإدخال النصوص الكبيرة، وسوف تحفظ البيانات في ملف مستقل عن الجدول، ويمكن استخدام Memo(Binary) لحفظ البيانات بصيغة الماكنة لإمكانية ترجمتها بترميزات أخرى.
- **Blob**: يستخدم لحفظ بيانات من اي نوع، مثلاً الوثائق أو الملفات التنفيذية (.exe) أو اي Stream من البايتات التي تكوّن ملفات اخرى.
- **Varbinary**: نفس خصائص Blob ولكن بحجم اكبر يمكن تحديده.
- **Date**: يستخدم لحفظ التاريخ فقط بدون وقت، والذي يكون على شكل mm/dd/yy.
  - الحجم (8) بايت

- يمكن تغيير الشكل باستخدام الأمر SET DATE DMY
- التعامل معه برمجياً يكون بالشكل {^9999-12-31}

- **DateTime**: يستخدم لحفظ التاريخ مع الوقت، والذي يكون على شكل:
  mm/dd/yy hh:mm:ss AM|PM.
  - علماً أن AM يقصد بها جميع الاوقات قبل الظهر.
  - وأن PM يقصد بها جميع الاوقات بعد الظهر.
  - التعامل معه برمجياً يكون بالشكل {^9999-12-31, hh : mm : ss a|p}
- **Logical**: يستخدم مع البيانات المنطقية التي تتمثل ب True أو False.
  - الحجم (1) بايت
  - يتم التعامل مع برمجياً بالشكل .T. و .F.
- **General**: يمكن حفظ كائن من نوع OLE بداخله، على سبيل المثال ملف Excel، صورة...



- **Currency**: يستخدم للعملات، مثلاً للأسعار نستخدم نوع العملة بدلاً من النوع الرقمي.
  - الحجم (8) بايت

ملاحظات:

- كل نوع تسبقه كلمة Var فهذا يعني أن حجم الحقل يكون بحجم البيانات المخزنة.
  - مثلاً: Varint،Varchar ...
- كل نوع يسبقه حرف n فهذا يعني أن بيانات الحقل تكون موحدة الترميز Unicode (أي يمكن الكتابة بجميع اللغات).
  - مثلاً: nVarchar،nchar ...

- كل نوع ترفق معه كلمة Binary فهذا يعني أن بيانات الحقل تكون بصيغة Binary.
  - مثلاً: char (binary)، memo (binary) …

نموذج الكيانات والعلاقات Entity Relationship (ER) Diagram

- ER is a model which will display data in a high-level manner, this diagram is usually used in data analysis and modeling phase. The data building is represented by using an easy graphical form.
- هو نموذج يقوم بعرض البيانات بشكل عالي المستوى، ويتم استخدام هذا النموذج عادةً في مرحلة تحليل ونمذجة البيانات. ويتم تمثيل بناء البيانات والقيود المطلوبة عليها باستخدام اشكال رسومية سهلة ومحددة.

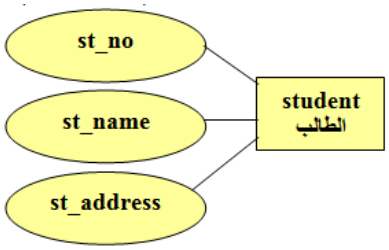
الكيان Entity

- Entity is an object or thing have attention in the system, and we have to collect and record data for this entity. And can look to the entity as a class of data.
- الكيان هو عبارة عن كائن أو شيء محط الاهتمام في النظام، وعلينا أن نقوم بجمع وتسجيل البيانات عن هذا الكيان. ويمكن أن ننظر إلى الكيان على أنه فئة من البيانات
- مثلاً الطالب، المادة، المدرس، الشعبة، تعتبر كيانات مهمة في نظام قاعدة البيانات لجامعة. وكذلك الطبيب، المريض، وصفة العلاج، كيانات مهمة في قاعدة بيانات لمستشفى.
- ويرمز لمجموعة الكيانات بمستطيل يحتوي على اسم الكيان.

- ومجموعة الكيانات تمثل المجموعة التي تنتمي إليها مجموعة الكائنات المتشابهة وتُمثَّل بجدول في قاعدة البيانات العلائقية.

الخصائص أو الصفات Attributes

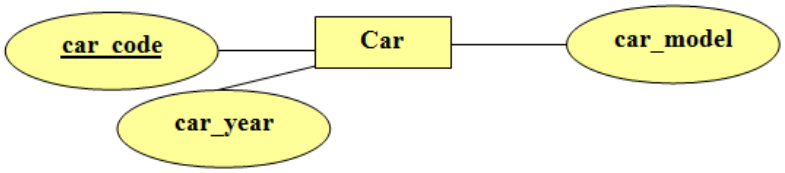
- Attributes is a characteristics of the entity, in other words is the information to be stored for a particular entity, and it represents table columns in a relational database.
- هي عبارة عن الصفات المميزة للكيان، وبعبارة أخرى هي المعلومات الواجب تخزينها عن كائن معين، وتُمثَّل بأعمدة الجدول في قاعدة البيانات العلائقية.
- فمثلاً لكل طالب يجب أن نسجل الرقم، الاسم، تاريخ الميلاد، المرحلة... ولمنتج معين يكون الرقم، الوصف، الحجم، اللون...
- ويرمز للصفة بشكل بيضاوي يحتوي على اسم الصفة وتربط الصفة مع الكيان بواسطة خط مستقيم.



مجال القيم Domain

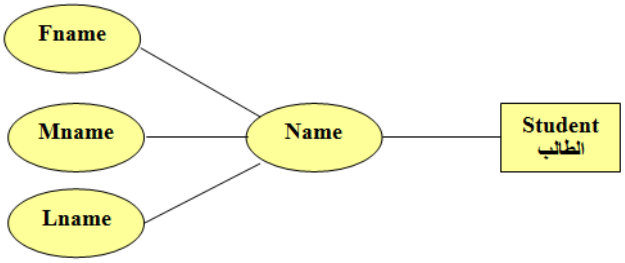
- لكل صفة يوجد هناك مجال للقيم (domain).

- فمثلا رقم الطالب يجب أن يكون عدد صحيح من عشر خانات، واسم الطالب يجب أن يحتوي علي قيم رمزية بطول 30 حرف، والمعدل يجب أن يحتوي علي عدد كسري ما بين الصفر والرقم 3...
- وإن الصفة (أو مجموع الصفات) التي تم اختيارها كمفتاح رئيسي (primary key) تمثل كأي صفة ولكن يوضع خط تحت الاسم.



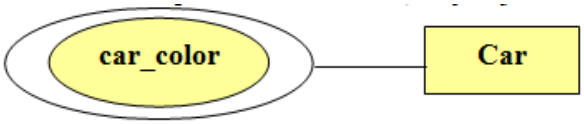
الصفات البسيطة والمركبة Simple and Composite Attributes

- Simple attributes cannot be fragmented, such as: Stud ID, gender, and birth…
- Composite attributes are a characteristics are that can defragment such as: Name (first name, second name), and address (city, town, street, house number).
- الصفات البسيطة هي التي لا يمكن تجزئتها مثل: رقم الطالب، الجنس، وتاريخ الميلاد...
- أما الصفات المركبة فهي التي يمكن تجزئتها مثل: الاسم يمكن أن يجزأ إلى (الاسم الأول، الثاني، اسم العائلة)، ويجزأ العنوان إلى (المدينة، الحي، الشارع، رقم المنزل).
- ويرمز للصفة المركبة بشكل بيضاوي ترتبط معه أشكال بيضاوية أخرى، ويحتوي كل منها على اسم الصفة الفرعية، وترتبط الصفات الفرعية مع الصفة الرئيسية بواسطة خط مستقيم.

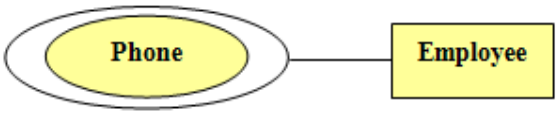


صفات وحيدة أو متعددة القيم Single – valued or Multiple – valued Attributes

- Single – valued attributes is contain a single value such as (car number, date of manufacture).
- Multiple – valued attributes are have multiple values ,such as the color of the car (there could be the color of the roof, body sides).
- الصفات التي تحتوي علي قيمة واحدة مثل (رقم السيارة ، تاريخ الصنع).
- أما الصفات التي تحتوي على عدة قيم مثل لون السيارة (فيمكن أن يكون هنالك لون للسقف ، الجسم ، الجوانب).



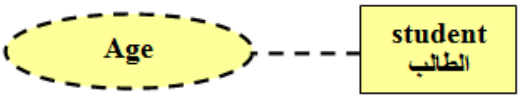
- ومثال آخر: يمكن أن يكون للزبون عدة أرقام هاتف (المحمول، المنزل، العمل..)



- ويرمز للصفة متعددة القيم بشكل بيضاوي داخل شكل بيضاوي آخر يحتوي علي اسم الصفة وترتبط الصفة مع الكيان بواسطة خط مستقيم.

الصفات المشتقة Derived Attributes

- Derived attributes can be derived from other attributes.
- هي الصفات التي يمكن اشتقاقها من صفات أخرى.
- ويرمز لها بشكل بيضاوي متقطع يحتوي على اسم الصفة وترتبط مع الكيان بخط مستقيم متقطع أيضاً.
- على سبيل المثال: عمر الطالب يمكن معرفته من خلال الفرق بين التاريخ الحالي وتاريخ الميلاد.



- مثال آخر: يمكن حساب معدل الطالب من خلال:

(Sum(Mark) / Count(Subjects)) (مجموع الدرجات / عدد المواد)

- ويمكن حساب تكلفة الفاتورة:

(Unit_price * Quantity) (سعر الوحدة X عدد الوحدات)

- ويمكن حساب الخصم من خلال:

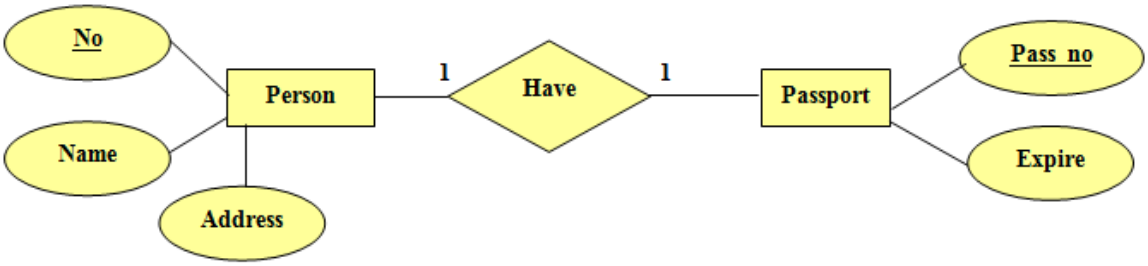
(1 – نسبة الخصم) X (سعر الوحدة X عدد الوحدات)

(1 - Discount_percent) * (Unit_price * Quantity)

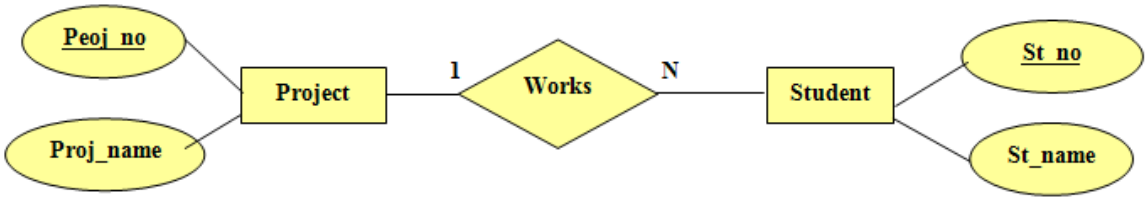
العلاقات Relationship

- The relationship (R) between a set of entities (E1, E2, ... En) represents the links between these entities, and each instance in R is a union between related entities, therefore the unit represents a single row from each participating entity in this relationship.
- العلاقة (R) بين مجموعة من الكيانات (E1,E2, ...En) تمثل الارتباطات بين هذه الكيانات، وكل وحدة (Instance) في العلاقة (R) هي عبارة عن اتحاد بين الكيانات المرتبطة، بحيث أن هذه الوحدة تمثل بصف واحد من كل كيان مشارك في هذه العلاقة.

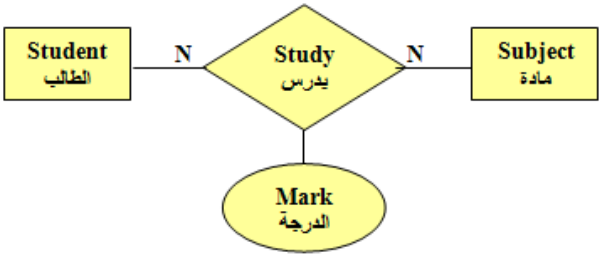
- ويرمز لها بشكل معين (Diamond) يحتوي على اسم الرابط أو العلاقة.
- كما يوجد لكل علاقة نسبة الارتباط (Cardinality Ratio) تبين مقدار التشارك ما بين الكيانات إما (1:1) أو (N:1) أو (N:N).
- على سبيل المثال كل شخص لديه جواز سفر واحد، وجواز السفر يعود لشخص واحد، وهذه العلاقة تمثل علاقة 1-1 ويمكن تمثيلها كما في الشكل:



- مثال آخر: كل طالب يعمل على مشروع واحد، والمشروع يعمل عليه أكثر من طالب، وهذه العلاقة تمثل علاقة 1-N ويمكن تمثيلها كما في الشكل:



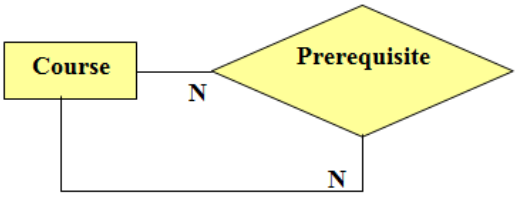
- ومثال آخر: كل طالب يدرس مادة واحدة أو أكثر، والمادة يدرسها مجموعة من الطلاب. وهذه تمثل علاقة N-N ويمكن رسمها كما في الشكل:



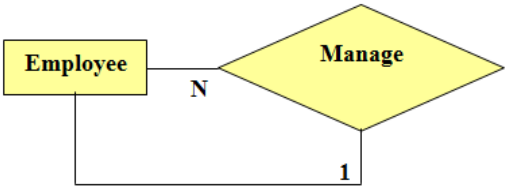
- مع ملاحظة أن العلاقة يمكن أن يكون لها صفات أيضاً، وهذه هي صفات علاقة الكيانين مع بعضهما، أي تمثل صفة مشتركة.

تمثيل علاقة الكيان مع نفسه Recursive

- وهي تمثيل ارتباط الكيان بنفسه، فمثلاً لو فرضنا أن المقرر الدراسي يمكن أن يكون لديه متطلب سابق أو أكثر (وهذا المتطلب هو عبارة عن مقرر).



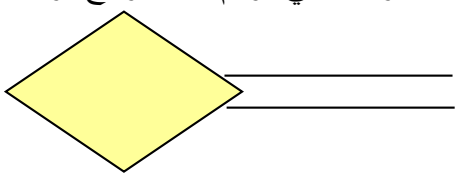
- وكذلك يجب أن يكون للموظف مدير واحد فقط (والمدير بدوره هو أيضا موظف).



أنواع القيود على العلاقات Participation Constraints

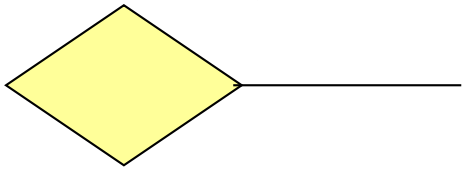
1. اشتراك كلي (Total Participation):

- كل كيان يجب أن يرتبط بوحدة (Instance) في العلاقة.
- يسمى هذا القيد بقيد «ارتباط الوجود» (Existence Dependency)، أي أن وجود وحدة من كيان ما يستلزم ارتباطها بوحدة من كيان آخر.
- يتم تمثيل قيد الاشتراك الكلي برسم خط مزدوج يربط الكيانات المرتبطة بهذه العلاقة مثل:

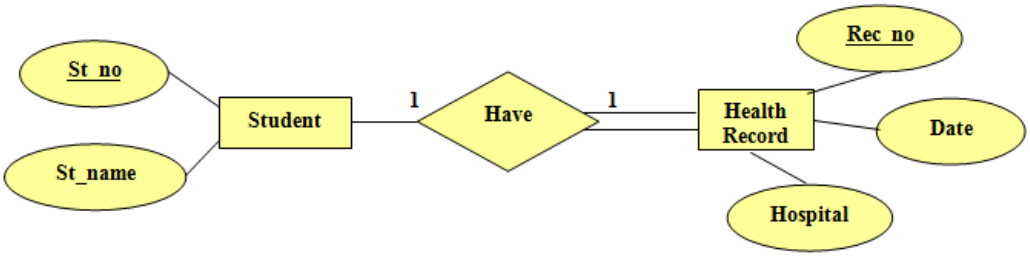


2. اشتراك جزئي (Partial Participation):

- بعض الكيانات ترتبط ببعض الوحدات (Instances) في العلاقة.
- يتم تمثيل قيد الاشتراك الجزئي برسم خط مفرد يربط الكيانات المرتبطة بهذه العلاقة مثل:



- **مثال:**

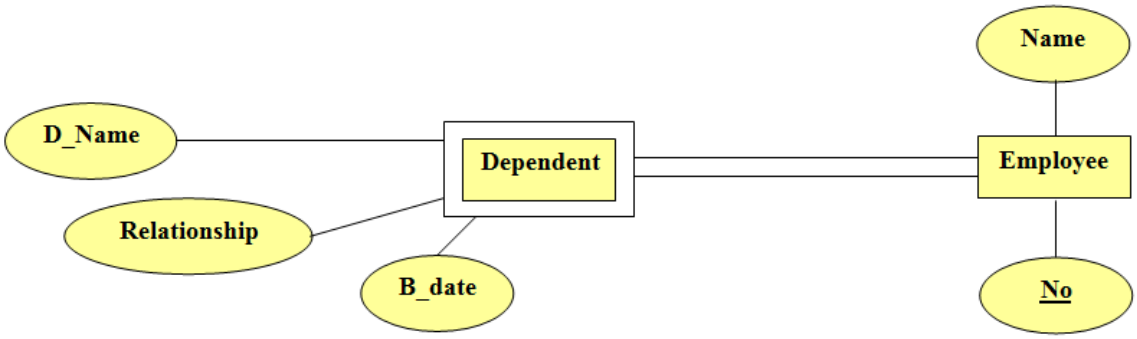


ملاحظات:

- لكل طالب سجل واحد (نوع العلاقة 1).
- السجل يكون لطالب واحد (نوع العلاقة 1).
- يمكن أن يكون بعض الطلبة ليس لديهم سجلات (اشتراك جزئي).
- كل سجل لابد وأن يكون يتبع طالب معين (اشتراك كلي).

الكيانات الضعيفة Weak Entities

- Weak entities are a no independent entities in a system, therefore their existence depends on the existence of another entity.
- هي عبارة عن كيانات لا توجد مستقلة بذاتها في النظام، أي أن وجودها يعتمد على وجود كيان أخر.
- فمثلاً لو فرضنا أن مؤسسة ما تسجل معلومات عن أسماء الأشخاص التابعين للموظف مثل الأبناء، الزوجة أو الوالدين، فوجود معلومات التابع مرتبطة بوجود الموظف.
- وفي هذه الحالة يتم اختيار المفتاح الرئيسي للكيان الرئيسي مع صفة من صفات التابع (مثل الاسم) لتشكل مفتاحا رئيسيا للكيان التابع و يوضع تحته خط متقطع.
- ويرمز للكيان الضعيف بمستطيل داخل مستطيل يحتوي علي اسم الكيان الضعيف، ويرتبط مع الكيان الرئيسي بخطين مستقيمين (يعني أن وجود الكيان الاول شرط لوجود الكيان الأخر وليس بالضرورة للكيانات الضعيفة فقط).



أمثلة لتشكيل نموذج ER

1- Suppose you have a (Trading management System), this company has many of departments and their data as follows (Department name - Department number - Phone). And has a number of employees who work in different departments and their data as follows (Employee

name - Employee ID - Title - Salary). The work date is recorded for every employee in his department.

Propose ER diagram to this system?

الحل:

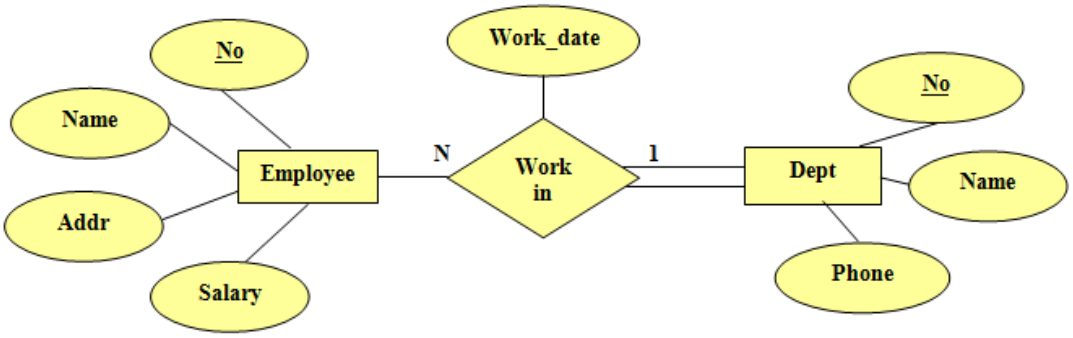
تحديد الكيانات:

القسم (اسم القسم – رقم القسم – هاتف)

الموظف (اسم الموظف – الرقم الوظيفي – العنوان – الراتب).

تحديد العلاقات:

علاقة عمل الموظفون في الاقسام.



ملاحظات:

- الموظف يعمل في قسم واحد والقسم به عدة موظفون (1:N).
- يمكن أن يكون بعض الموظفون ليس لديهم اقسام (اشتراك جزئي).
- لا يمكن أن يكون القسم إلا وبه موظفون (اشتراك كلي).
- "تاريخ العمل" هي صفة للعلاقة "Work in" ولذلك اضيفت لها.

2- Suppose you have a (Learning management System), this system have a number of students those registered to many courses, The registrar of learning system is write the year and the class number when registering a student for any course.

Propose ER diagram to this system?

الحل

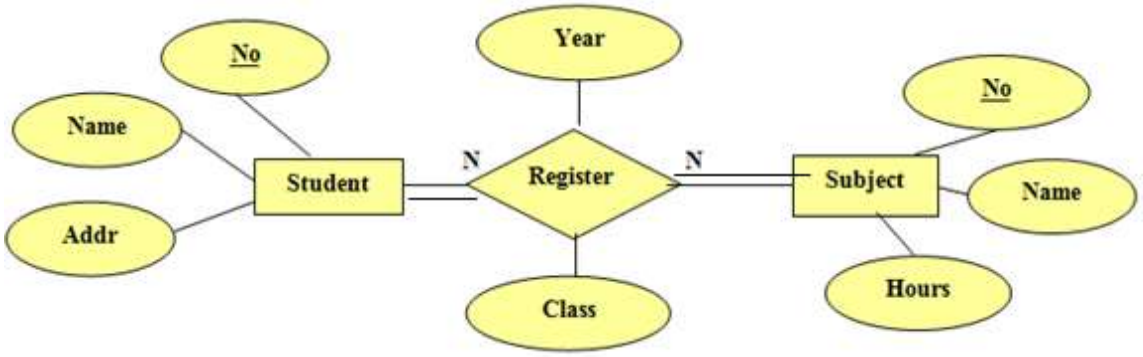
تحديد الكيانات:

الطالب (اسم الطالب – الرقم الجامعي – العنوان)

المقرر (اسم المقرر – رقم المقرر – عدد الساعات).

تحديد العلاقات:

علاقة تسجيل الطالب لمقرر.



ملاحظات:

- الطالب يمكن أن يسجل مجموعة من المقررات (نوع العلاقة N).
- المقرر يسجله مجموعة من الطلبة (نوع العلاقة N).
- لا يمكن أن يكون بعض الطلبة ليس لديهم مقررات (اشتراك كلي).
- لا يمكن أن تكون المقررات غير مسجل فيها طلبة (اشتراك كلي).
- "السنة، والشعبة" هي صفات للعلاقة "يسجل" ولذلك اضيفت لها.

3- Suppose we have a (Library management System), this library contains many Books that composed by Authors, every author can compose one or more book. These books are published by one Publisher that can publish one or more book.
Propose ER diagram to this system?

الحل:

الكيانات:

الكتاب (رقم الكتاب، عنوان الكتاب)

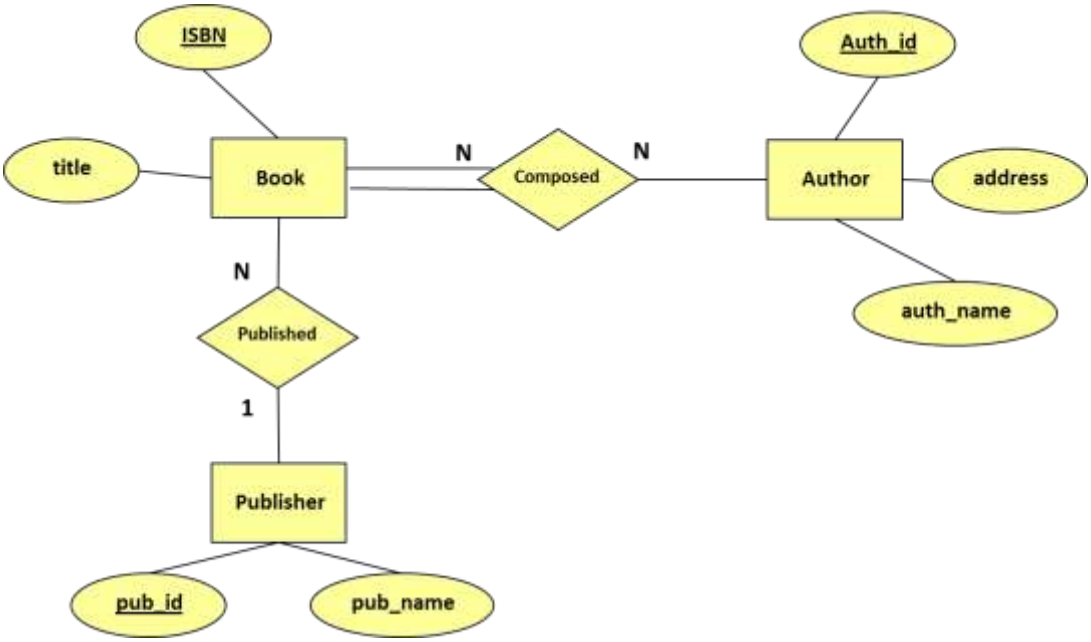
المؤلف (رمز المؤلف، اسم المؤلف، العنوان)

الناشر (رمز الناشر، اسم الناشر)

العلاقات:

علاقة الكتاب مع المؤلف

علاقة الكتاب مع الناشر



ملاحظات:

- الكتاب يؤلفه مؤلف واحد او أكثر من مؤلف بالتعاون، ويمكن للمؤلف تأليف أكثر من كتاب (N:N).
- يُنشر الكتاب في دار نشر واحد، ودار النشر يمكنه ان ينشر أكثر من كتاب (N:1).
- لا يمكن للكتاب إلا وبه مؤلف (اشتراك كلي)، ويمكن أن يكون المؤلف ليس لديه كتاب(اشتراك جزئي).
- يمكن ان يكون الكتاب غير منشور (اشتراك جزئي)، ويمكن لدار النشر أن لايحتوي على كتب (جزئي)

4- Suppose we have a (Company management System), this company contains many Employees that are belongs to Departments, and works on Projects. One department have many employees, and every employee is works to one project, as well as every one project is worked by many employees.
Propose ER diagram to this system?

الحل:

الكيانات:

الموظف (رمز الموظف، اسم الموظف)

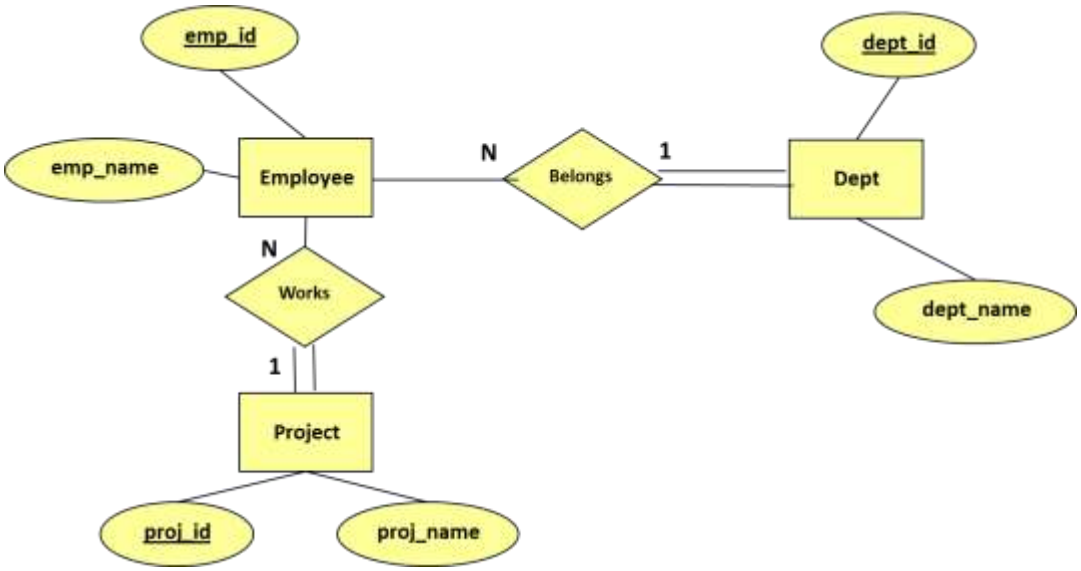
القسم (رمز القسم، اسم القسم)

المشروع (رمز المشروع، اسم المشروع)

العلاقات:

علاقة الموظف مع القسم

علاقة الموظف مع المشروع



ملاحظات:

- الموظف ينتمي لقسم واحد والقسم به عدة موظفون (1:N).
- الموظف يعمل على مشروع واحد والمشروع يعمل عليه عدة موظفون (1:N).
- يمكن أن يكون بعض الموظفون ليس لديهم اقسام (اشتراك جزئي). ولا يمكن أن يكون القسم إلا وبه موظفون (اشتراك كلي).
- يمكن أن يكون بعض الموظفون ليس لديهم مشاريع (اشتراك جزئي). ولا يمكن أن يكون المشروع إلا ويعمل عليه موظفون (اشتراك كلي).

تحويل مخطط ER إلى مخطط قاعدة البيانات Mapping ERD to DB schema

مخطط قواعد البيانات Database Schema

- DB Schema describes the database graphically in preparation for construction in the form of tables in Database Management System DBMS, and it resulting from mapping ER diagram by using Mapping Algorithm.

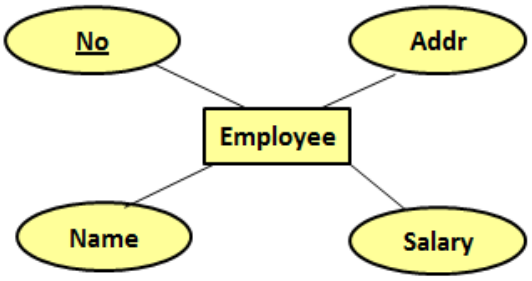
- هو مخطط يصف قاعدة البيانات بشكل رسومي تمهيداً لبنائه على شكل جداول في نظام إدارة قواعد البيانات DBMS، وينتج من عملية إخضاع مخطط ER لخوارزمية التحويل.
- تتم عملية تحويل مخطط ER بتطبيق مجموعة من الخطوات البسيطة التي تسمى خوارزمية التحويل.
- ويتم تطبيق هذه الخوارزمية بصورة كاملة، مع تجاوز بعض الحالات التي لم تظهر في نموذج ER.
- وبنهاية تطبيق هذه الخوارزمية يجب أن نحصل على مخطط قاعدة البيانات العلائقي، والذي يمثَّل بصورة رسومية.

خوارزمية التحويل Mapping Algorithm:

1. تحويل الكيانات القوية.
2. تحويل الكيانات الضعيفة.
3. تحويل العلاقات الثنائية من النوع 1:1.
4. تحويل العلاقات الثنائية من النوع N:1.
5. تحويل العلاقات الثنائية من النوع N:N.
6. تحويل الصفات متعددة القيم.

1. **تحويل أنواع الكيانات القوية:**

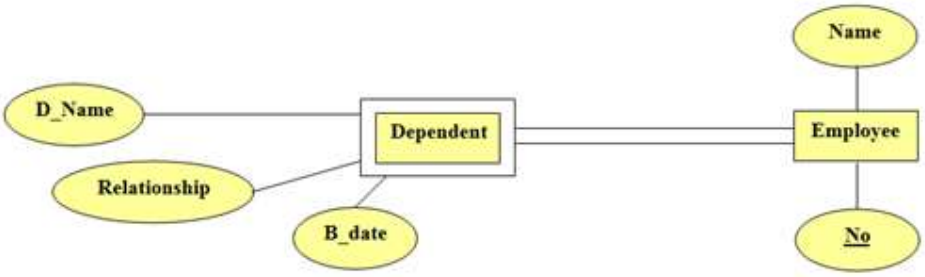
- يتم هنا تحويل جميع الكيانات القوية، أي الكيانات غير الضعيفة، بإنشاء جدول يتكون من الحقول التي تقابل صفات ذلك الكيان.
- ويتم تحديد أحد مفاتيح الكيان، وتسميته بالمفتاح الرئيسي Primary Key (PK).
- وإذا كانت الصفة التي تمثل المفتاح من النوع المركب (Composite Attributes) فإن المفتاح الرئيسي سيكون مجموعة الحقول التي تنشأ من الصفة المركبة.

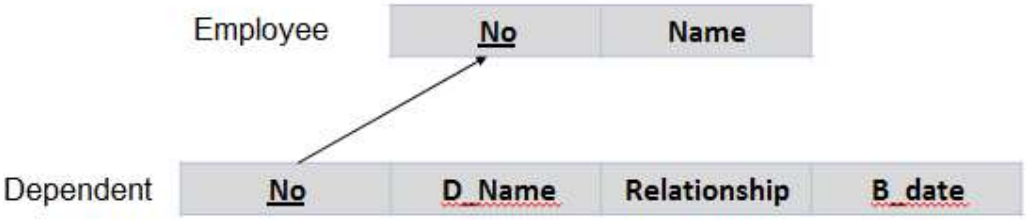


Employee	Emp_Id	Name	Addr	Salary

2. **تحويل أنواع الكيانات الضعيفة:**

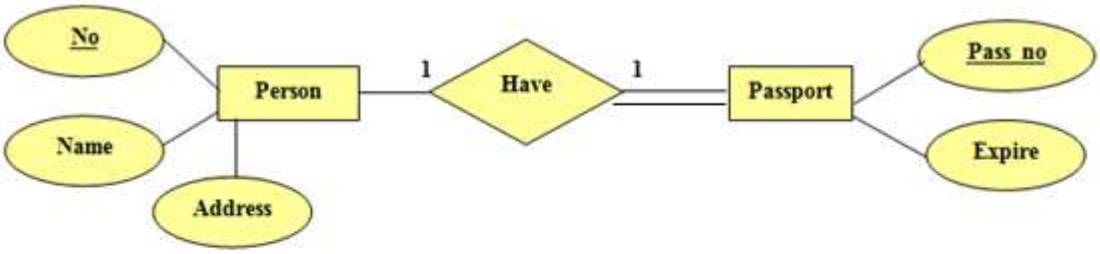
- يتم تحويل كل واحدة من الكيانات الضعيفة بإنشاء جدول يتكون من الحقول التي تقابل صفات ذلك الكيان، كما يجب إضافة المفتاح الرئيسي للكيان القوي الذي يتبعه ذلك الكيان الضعيف.

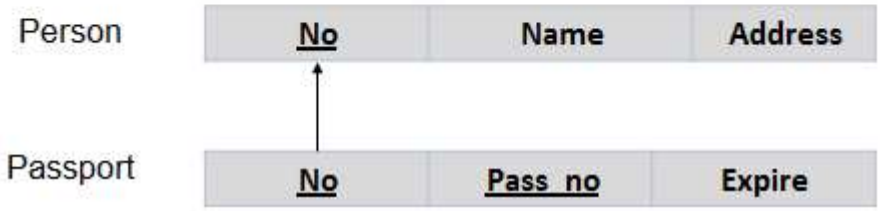




3. تحويل العلاقات الثنائية من النوع 1:1

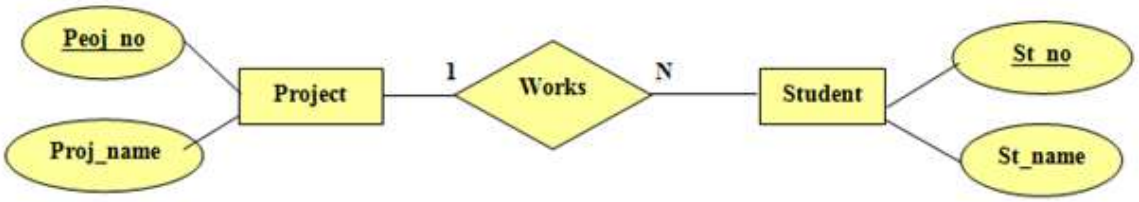
- إذا كانت العلاقة بين الكيانين علاقة (One-to-One) فإن عملية التحويل تتم وفق عدة خيارات اشهرها خيار يسمى **بطريقة المفتاح الأجنبي**، وفيها يتم إضافة المفتاح الرئيسي PK لأحد الجدولين إلى الجدول الآخر كمفتاح اجنبي FK .
- ويفضل أن يكون الجدول الذي يحتوي على المفتاح الأجنبي، هو الجدول الذي يكون نوع قيد اشتراكه في العلاقة من نوع الاشتراك الكلي (Total Participant).





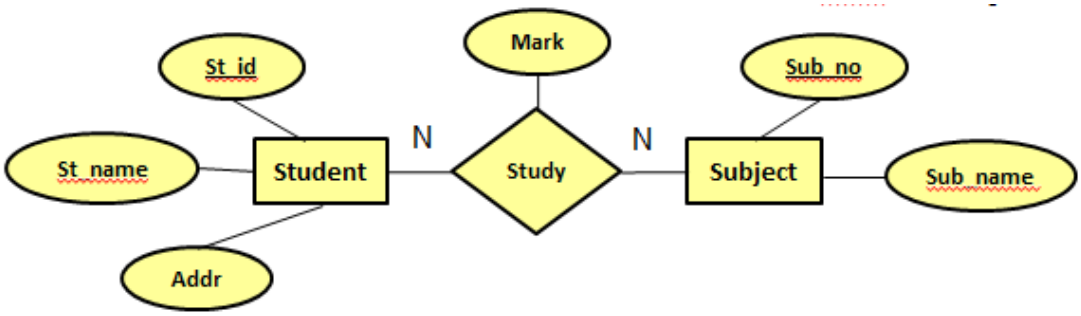
4. تحويل العلاقات الثنائية من النوع N:1

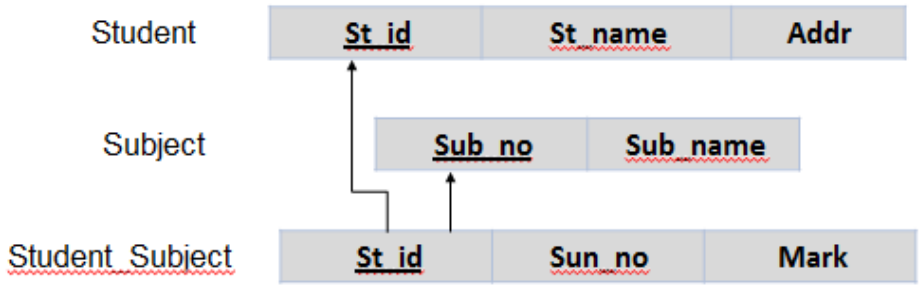
- يتم هنا إنشاء جدولين لتمثيل الكيانين المرتبطين، على أن يتم تطبيق **طريقة المفتاح الأجنبي** السابقة، وذلك بإضافة المفتاح الرئيسي PK للجدول من جهة العلاقة (1) إلى الجدول الآخر المرتبط بالعلاقة (N)، وبغض النظر عن نوع الاشتراك.



5. تحويل العلاقات الثنائية من النوع N:N

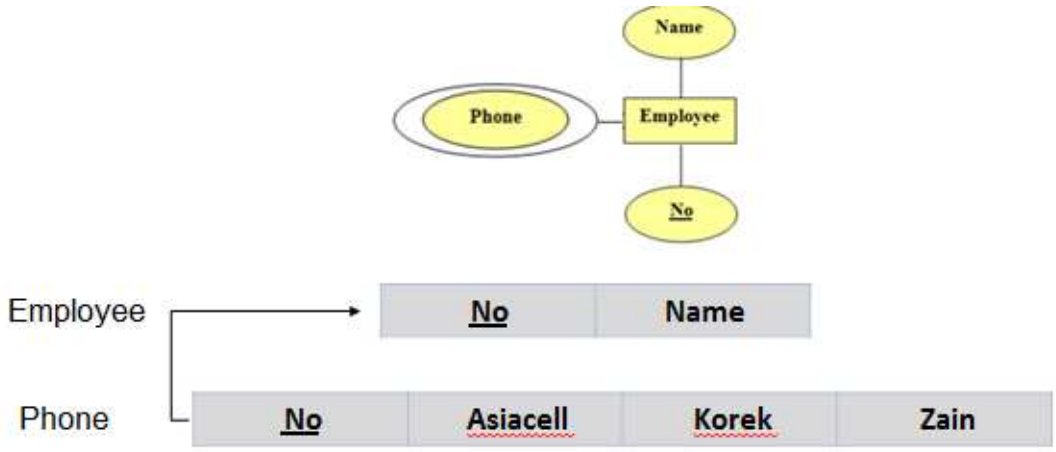
- في هذا النوع من العلاقات يتم استحداث جدول جديد، فيكون الناتج من هذه العلاقة ثلاثة جداول، جدولين لتمثيل الكيانين المرتبطين بالعلاقة، ويضم الجدول الثالث حقلين كمفتاحين أجنبيين يمثلان المفتاحين الرئيسيين في الجدولين.
- ويمكن إضافة أي حقل آخر يكون له مغزى، كأن تكون العلاقة لها صفة بذاتها، فتتحول الصفة إلى حقل في الجدول الجديد.



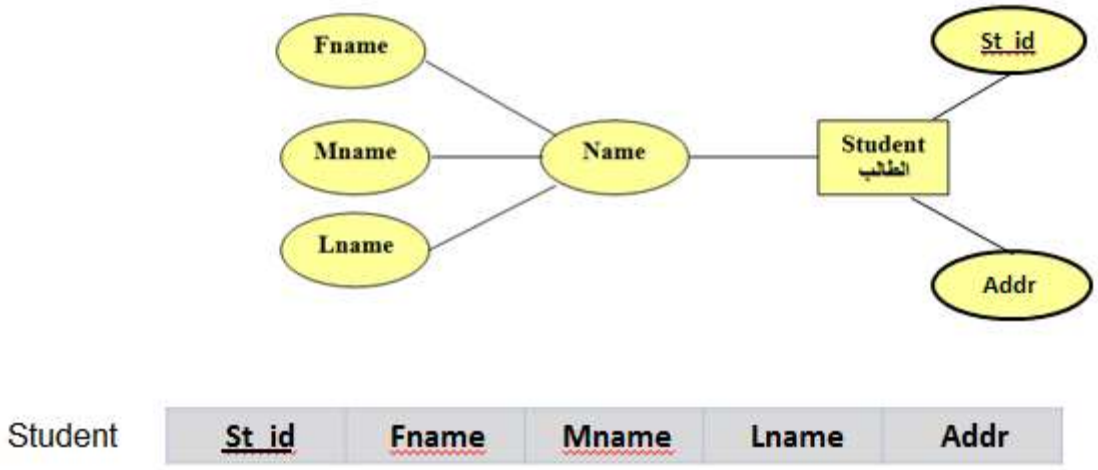


6. تحويل الصفات متعددة القيم

- يتم في هذه الحالة عادة إنشاء جدول جديد يضم الصفة المتعددة القيم كحقل، ويضاف إلى الجدول مفتاح أجنبي FK يكون ممثلاً للمفتاح الرئيسي في الجدول الناتج من الكيان الذي يحتوي على الصفة متعددة القيم.



- أما الصفات المركبة فتتحول إلى صفات بسيطة، فحقول عادية كما في الشكل ادناه:



- والصفات ذات القيم المشتقة تلغى من الجدول، لأنها صفات قابلة للاشتقاق من صفات أخرى، فلا داعي لوجودها، ولكن يمكن اضافتها فيما بعد في نظام DBMS.

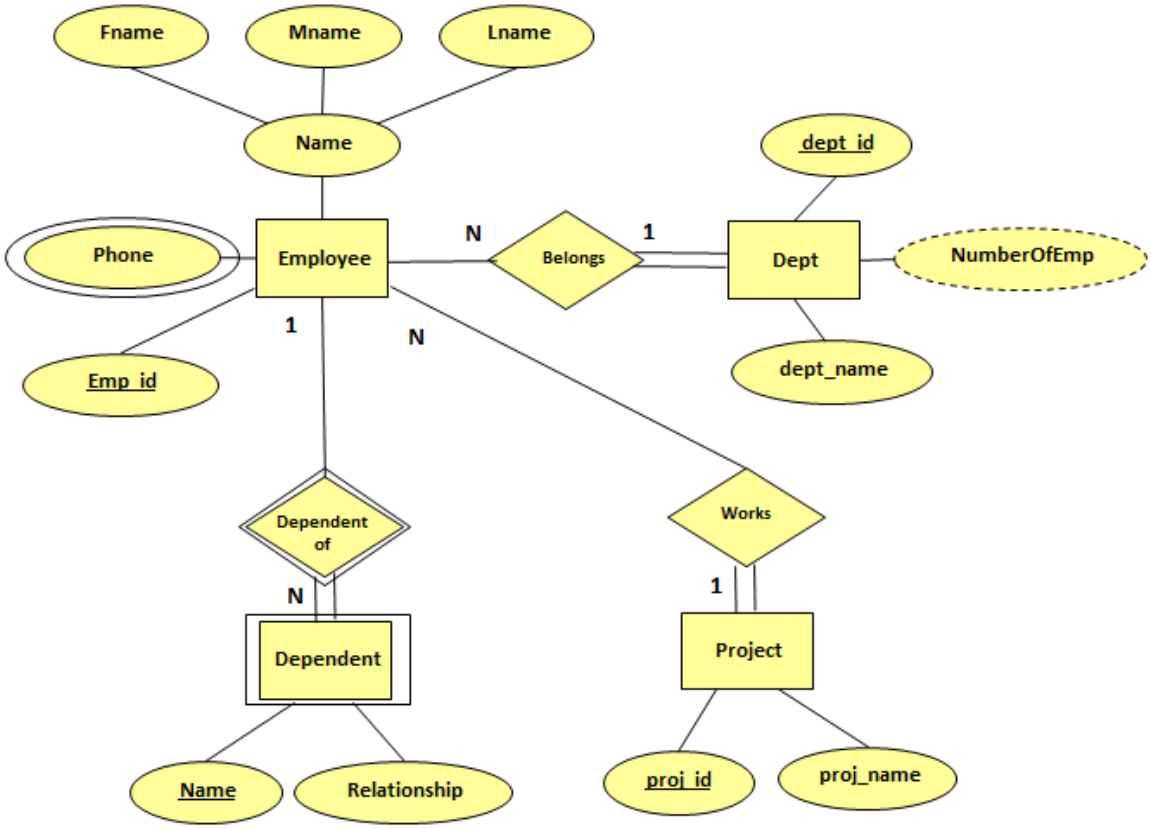
مثال توضيحي

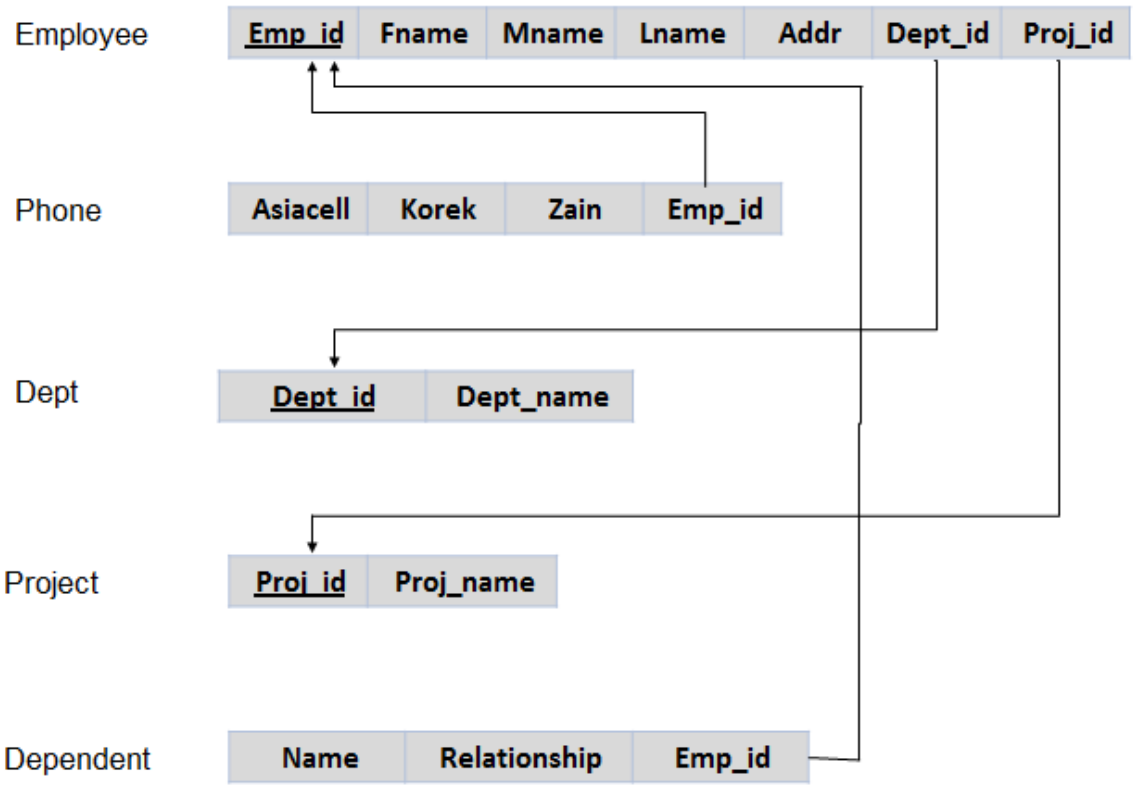
قاعدة بيانات موظفين

Suppose you have a (Company Management System), this company contains many employees that are belongs to departments. Each one department contains many employees. Some of these employees are works

to one project, and the one project can be worked by many employees. And one employee can have dependents those are dependent on him.

Propose ER diagram to this system?
And mapping it to DB Schema?





الصيغ المعيارية Normal Forms

Normalization is the process of a clearance the database of inappropriate reputation of data by depending on the inference rules and functional dependency, although the process of developing the database design in the normal forms is a key building block for a state of art design of the database.

الصيغة المعيارية: هي عملية تخليص قاعدة البيانات من التكرار غير المناسب للبيانات بالاعتماد على قوانين الاستنتاج والاعتمادية الوظيفية، وإن عملية وضع تصميم قاعدة البيانات في الصيغة المعيارية يشكل لبنة أساسية في عملية التصميم الجيد لقاعدة البيانات، وتتم هذه العملية على عدة مراحل مثل:

- الصيغة المعيارية الأولى 1NF First Normal Form
- الصيغة المعيارية الثانية 2NF Second Normal Form
- الصيغة المعيارية الثالثة 3NF Third Normal Form

- **مشاكل تكرار البيانات (Data Anomalies)**

- نلاحظ في الجدول التالي أن معلومات الموظف والقسم الذي يعمل فيه موجودة في جدول واحد، ونتيجة لذلك فأن تكرار بعض البيانات مثل اسم وموقع القسم في كل سجل سوف يؤدي إلى عدة مشاكل.

Employee_department

Empno	Ename	Job	Salary	Deptno	Dname	Loc
10	Smer	Clerk	300	1	Accounting	Baghdad
20	Ali	Manager	500	1	Accounting	Baghdad
30	Khalid	Salesman	400	2	Sales	Mosul
40	Saeed	Salesman	450	2	Sales	Mosul
50	Salem	Clerk	350	3	Operation	Basrah

1. **مشكلة الإضافة Insertion Anomaly**: وهي عدم القدرة على إضافة قيمة بيانية في أحد السجلات؛ لعدم وجود قيمة بيانية أخرى في مكان آخر. حيث أننا لا نستطيع أن نضيف قسماً جديداً إلا إذا كان القسم يحتوي على موظف؛ لأن المفتاح الرئيسي للجدول هو رقم الموظف.
2. **مشكلة التعديل Updating Anomaly** : عند التعديل على قيمة بيانية في أحد المواقع، يجب تغيير جميع القيم المشابهة لها في المواقع الأخرى. حيث إذا قمنا بتعديل موقع (loc) القسم فلا بد من إجراء عملية التعديل لجميع الموظفين في هذا القسم؛ وإلا ستؤدي هذه العملية إلي عدم توافقية البيانات، أي نفس رقم القسم ولكن أكثر من موقع، وكذلك إذا تمت عملية تغيير موقع القسم عند أي موظف عن طريق الخطأ، فإنه لو قمنا بعملية استرجاع لجميع الموظفين الذين يعملون في هذا القسم فإن هذا الموظف لن يظهر بين الموظفين.
3. **مشكلة الحذف Deletion Anomaly**: وهي مشكلة ترابط البيانات التي تحذف بأكملها عند حذف قيمة بيانية متصلة بها. حيث إن القسم 3 يحتوي على موظف واحد فقط، ولو قمنا بحذف هذا الموظف فإن معلومات القسم 3 سوف تختفي من الجدول.

- **الاعتمادية الوظيفية Functional Dependency (FD)**

- Functional Dependency is the adoption of one attributes on the value of another attribute(s).Where through the identification of this dependency we will be able to determine where it should be placed, and this consequently leads to put the data in the right place and get rid of the data replication.

- وهي اعتماد قيمة إحدى صفات الكيان على قيمة صفة (صفات) أخرى، ويرمز لها بالرمز ⟶

- حيث من خلال تحديد هذه الاعتمادية سوف نستطيع أن نحدد المكان الذي يجب أن توضع فيه الصفة، وهذا بالتالي يؤدي إلى وضع البيانات في المكان الصحيح والتخلص من عملية تكرار البيانات.
- **مثال1 : A ⟶ B**
- يعني أن B تعتمد اعتماداً وظيفياً على A ، ونستطيع أن نقول هنا أن قيمة A تحدد قيمة B. وأن A ترجع لنا قيمة واحدة فقط لـ B.
- ويمكن ايضاً أن نقول أننا نستطيع أن نصل إلى البيانات في العمود B بمعرفتنا للبيانات في العمود A.
- **مثال2:** لكل موظف اسم واحد فقط ولكل موظف قسم واحد يعمل فيه إذاً:

FD1 : Empno ⟶ Ename
FD2 : Empno ⟶ Deptno

- أي يمكن معرفة اسم الموظف من رقمه وكذلك القسم.
- ويمكن أن نعيد كتابة هذه الاعتمادية على الشكل التالي:

FD1 : Empno ⟶ Ename , Deptno

- **قواعد الاستنتاج Inference Rules**
- Inference Rules is a set of rules that are used in the process of identifying the functional dependency.
- هي عبارة عن مجموعة من القواعد تستخدم في عملية تحديد الاعتمادية الوظيفية وتتلخص هذه القواعد كما يلي:
- **1- قاعدة الانعكاسية (Reflexive Rule)** : إذا كانت y جزء من x (y محتواه في x) فإن x تحدد y

X ≥ Y : X ⟶ Y

- **مثال:**

Empno ⟶ Deptno

2- قاعدة الزيادة أو الإضافة (Augmentation Rule) : إذا كانت x تحدد y فإن إضافة z إلي x تعني أنه بالإمكان إضافة z إلى y

{ X ⟶ Y } |= XZ ⟶ YZ

- **مثال:**

Empno ⟶ Deptno
Empno,Dname ⟶ Deptno,Dname

3- قاعدة التعدي (Transitive Rule) : إذا كانت x تحدد y وكانت y تحدد z فإن x تحدد z

$$\{ X \rightarrow Y , Y \rightarrow Z \} \models X \rightarrow Z$$

- **مثال:**

Book → Author
Author → Auth_Addr
Book → Auth_Addr

4- قاعدة الاتحاد (Union Rule): إذا كانت x تحدد y و x تحدد z إذاً x تحدد yz

$$\{ X \rightarrow Y , X \rightarrow Z \} \models X \rightarrow YZ$$

- **مثال:**

Empno → Ename
Empno → Gender
Empno → Ename,Gender

5- قاعدة التقسيم (Decomposition Rule) : وهي عكس قاعدة الاتحاد، إذا كانت x تحدد yz إذاً x تحدد y و تحدد z

$$\{ X \rightarrow YZ \} \models X \rightarrow Y , X \rightarrow Z$$

- **مثال:**

Empno → Ename,Gender
Empno → Ename
Empno → Gender

- **العلاقة غير المعيارية Unnormalized Form (UNF)**

- Unnormalized form is a relationship that contains a duplicate set of data, that is, more than value in a cell.
- هي العلاقة التي تحتوي على مجموعة مكررة من البيانات، أي وجود أكثر من قيمة بيانية في داخل الخلية، أمثلة:

Customer

Cno	Cname	Address
10	Smer Mohammed Ali	12 abc Mosul
20	Ahmed Ali Salem	343 xyz Baghdad
30	Khalid Saad Ahmed	23 Abc Anbar
40	Noor Ahmed Mohammed	67 xyz Dohok

Employee

Eno	Ename	Proj_code	Hours	Deptno	Dname
210	Ali	P1 P2 P3	12 20 40	10 20 20	Research Operation Operation
201	Salem	P1 P3	30 15	10 20	Research Operation
305	Ali	P2 P3	40 20	20 20	Operation Operation

- **الصيغة المعيارية الأولى (1NF) First Normal Form**

- The table will be in the 1NF if all table columns contain simple data (Atomic), that is, any intersection of column with row gives only one value.

- إن الجدول يكون في الصيغة المعيارية الأولى إذا كانت جميع أعمدة الجدول تحتوي على بيانات بسيطة أو مفردة (Atomic) غير مركبة، أي إن تقاطع العمود مع الصف يعطي قيمة واحدة فقط.
- **مثال** : الاسم (يتم تقسيمه إلى الاسم الأول، الاسم الثاني، اسم العائلة)، العنوان (يتم تقسيمه إلى المدينة، الشارع، رقم الشارع) وكل واحد في عمود مستقل.
- أي أن في 1NF هناك قاعدتان أساسيتان:

1- أن لا يكون هناك صفة متعددة القيم : بمعنى أن يكون داخل العمود أكثر من بيان.

2- أن لا يكون هناك صفة مركبة : بمعنى أن يقسم العمود إلى قسمين أو اكثر.

- **مثال 1:** في الجدول التالي معلومات الزبون، ونلاحظ أن الاسم يحتوي على ثلاثة قيم بيانية وكذلك العنوان، فبالتالي لا نستطيع أن نخزن قيمة واحدة فقط في عمود الاسم أو العنوان.

Customer

Cno	Cname	Address
10	Smer Mohammed Ali	12 abc Mosul
20	Ahmed Ali Salem	343 xyz Baghdad
30	Khalid Saad Ahmed	23 abc Anbar
40	Noor Ahmed Mohammed	67 xyz Dohok

- ولوضع الجدول في الصيغة المعيارية الأولى 1NF يجب تقسيم الأعمدة المركبة إلى أعمدة بسيطة:

Cno	Fname	Mname	Lname	Sno	Street	City
10	Smer	Mohammed	Ali	12	Abc	Mosul
20	Ahmed	Ali	Salem	343	Xyz	Baghdad
30	Khalid	Saad	Ahmed	23	abc	Anbar
40	Noor	Ahmed	Mohammed	67	xyz	Dohok

- **مثال 2:** يمثل الجدول التالي سجل ساعات العمل Hours لموظف في عدد من المشاريع Projects والقسم الذي يشرف على تنفيذ المشروع.

Employee

Eno	Ename	Proj_code	Hours	Deptno	Dname
210	Ali	P1	12	10	Research
		P2	20	20	Operation
		P3	40	20	Operation
201	Salem	P1	30	10	Research
		P3	15	20	Operation
305	Ali	P2	40	20	Operation
		P3	20	20	Operation

- كما هو مبين فأن عدداً من الأعمدة تحتوي على أكثر من قيمة مثل رمز المشروع وساعات العمل والاقسام، ولتحويله يجب أن نقوم بتقسيم الجدول على النحو التالي:

Eno	Ename	Proj_code	Hours	Deptno	Dname
210	Ali	P1	12	10	Research
210	Ali	P2	20	20	Operation
210	Ali	P3	40	20	Operation
201	Salem	P1	30	10	Research
201	Salem	P3	15	20	Operation
305	Ali	P2	40	20	Operation
305	Ali	P3	20	20	Operation

- ولكن توجد مشكلة في هذا الجدول وهي ايجاد المفتاح الرئيسي، إذ أصبح رقم الموظف لا يصلح أن يكون مفتاحاً رئيسياً PK، وذلك لأن من شروطه هي عدم التكرار.
- سوف نقوم الأن باستخدام الاعتمادية الوظيفية FD لمحاولة إيجاد المفتاح الرئيسي للجدول:

FD1 : Eno ⟶ Ename

- حيث أن الرقم يحدد الاسم، ولكل موظف رقم واحد. ولا يمكن أن نقول أن الاسم يحدد الرقم لوجود اسماء متشابهة.

FD2 : Proj_code ⟶ Deptno

- حيث أن لكل مشروع قسم واحد يشرف عليه.

FD3 : Deptno ⟶ Dname

- حيث أن لكل قسم اسم واحد.
- أما بالنسبة لبقية العناصر فمثلاً اسم الموظف لا يحدد شيئاً لأنه يوجد أكثر من موظف اسمه Ali فالأسم لا يحدد الرقم، وكذلك فأن Ali يعمل في أكثر من مشروع.
- وكذلك رمز المشروع لا يحدد عدد الساعات ولا الموظفين الذين يعملون فيه، فالمشروع P1 يعمل فيه أكثر من موظف وبساعات مختلفة.
- اما بالنسبة للقسم فلا يحدد الموظفين ولا المشاريع، فمثلاً القسم 20 يشرف على أكثر من مشروع وهذه المشاريع يعمل عليها اكثر من موظف.

- ففي هذه الحالة يجب علينا القيام بمحاولة جديدة لإيجاد المفتاح الرئيسي PK من خلال مفتاح مركب من اكثر من صفة.
- نقوم بربط رقم الموظف مع رمز المشروع:

FD4 : <u>Eno</u>, Proj_code ⟶ Ename
FD5 : Eno, <u>Proj_code</u> ⟶ Deptno
FD6 : <u>Eno, Proj_code</u> ⟶ Hours
FD7 : <u>Deptno</u> ⟶ Dname
FD8 : Eno, Proj_code ⟶ Ename, Hours, Deptno, Dname

- FD4 و FD5 تنطبق مع FD1 و FD2 حيث ان رقم الموظف وحده يحدد الاسم وكذلك رمز المشروع يحدد القسم.
- اما بالنسبة لـ FD6 فأنها تنطبق لأن رقم الموظف ورمز المشروع يحددان عدد الساعات لعمل الموظف في ذلك المشروع.
- وبالتالي نكون قد حصلنا على مفتاح رئيسي لهذا الجدول وكذلك قمنا بوضعه في الصيغة المعيارية الأولى.

ملاحظة: عندما يكون الجدول بالصيغة غير المعيارية UNF او 0NF نقوم بتحويل العمود ذو القيم المركبة أو متعددة القيم إلى قيم بسيطة بطريقتين:

- إما بتحويل العمود الواحد إلى عدة أعمدة،
- أو بعمل علاقة جديدة تحوي القيم المتعددة أو المركبة، بالإضافة إلى مفتاح أجنبي يمثل الجدول الأصلي بالطبع.

- **الصيغة المعيارية الثانية (2NF) Second Normal Form**

- The table will be in 2NF:

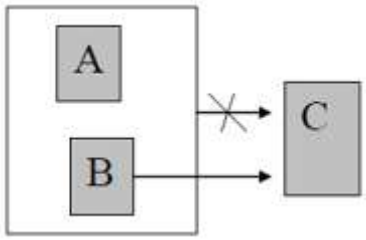
1. The table in the 1NF.
2. The table does not contain a partial dependency.

- إن الجدول يكون في الصيغة المعيارية الثانية إذا:

1. كان في الصيغة المعيارية الأولى.
2. لا يحتوي الجدول على اعتمادية جزئية.

- **الاعتمادية الجزئية Partial Functional Dependency**:

- Partial Functional Dependency is some of the columns (attributes) functionally dependent on the part of the primary key (composite).
- هي أن تعتمد بعض الأعمدة (الصفات) اعتماداً وظيفياً على جزء من المفتاح الرئيسي (المركب)، كما في الشكل:



- نلاحظ أن A, B تحدد C أي أن C تعتمد اعتماد وظيفياً على A,B، وكذلك أن B تحدد C اي أن C تعتمد اعتماداً وظيفياً على B، وفي هذه الحالة يمكن ان نقول أن هذا الجدول يحتوي على اعتمادية جزئية.
- يوضح الجدول الآتي درجات الطلاب: **Stud**

Sno	Cno	Grades
10	1	70
10	2	88
20	1	100
20	2	94

- فمثلا لو أردنا أن نعرف درجة الطالب رقم 10 ستظهر لنا درجتان وهما 70-88.
- وهذا يدل على أن الدرجات لا تعتمد اعتماداً وظيفياً كاملاً **Full Functional Dependency** على رقم الطالب، بل أنها تعتمد أيضاً على رقم المادة.
- ويمكن توضيحها كالآتي:

Sno , Cno ⟶ Grades

- اذن فالدرجات تعتمد وظيفياً كاملاً على رقم الطالب + رقم المادة.
- ومثلاً في حالة أضفنا إلى الجدول اسم الطالب واسم المادة: **Stud**

Sno	Cno	Grades	St_name	Course
10	1	70	Ahmed	DB
10	2	88	Ahmed	OS
20	1	100	Salem	DB
20	2	94	Salem	OS

- **مثال:**

Sno , Cno ⟶ St_name

- هنا خطأ لأن رقم الطالب يكفي لنعرف اسمه. وهذه الحالة تسمى اعتماد وظيفي جزئي.
- إذن فإن اسم الطالب يعتمد وظيفياً على رقم الطالب فقط. وهو اعتماد وظيفي جزئي.
- **مثال أخر:**

Sno , Cno ⟶ Course

- هنا خطأ لأن رقم المادة يكفي لنعرف اسمها. وهذه الحالة تسمى اعتماد وظيفي جزئي.
- في الصيغة المعيارية الثانية لا ينفع أن يكون هناك صفة تعتمد اعتماد وظيفي جزئي على المفتاح الرئيسي، ولكن يجب أن يكون اعتماداً وظيفياً كاملاً كما في حالة الدرجات فهي تعتمد على المفتاح المركب (رقم الطالب + رقم المادة).
- اما في حالة اسم الطالب، اسم المادة فهو اعتماد وظيفي جزئي.
- الحل في هذه الحالة أن نضيف جدول جديد لكل صفة مع المفتاح الرئيسي المعتمدة عليه اعتماداً وظيفياً كاملاً . وتحذف الصفات من الجدول الأساسي.

1) ننشأ جدول جديد به المفتاح الرئيسي + الصفة المعتمدة عليه اعتمادية وظيفية كاملة.

2) نحذف الصفة من الجدول الأساسي.

Sno	Cno	Grades	St_name	Course
10	1	70	Ahmed	DB
10	2	88	Ahmed	OS
20	1	100	Salem	DB
20	2	94	Salem	OS

Sno	St_name
10	Ahmed
20	Salem

Sno ⟶ St_name

- هنا يعتمد اسم الطالب على رقمه اعتماداً وظيفياً كاملاً.

1) ننشأ جدول جديد به المفتاح الرئيسي + الصفة المعتمدة عليه اعتمادية وظيفية كاملة.

2) نحذف الصفة من الجدول الأساسي.

Sno	Cno	Grades	St_name	Course
10	1	70	Ahmed	DB
10	2	88	Ahmed	OS
20	1	100	Salem	DB
20	2	94	Salem	OS

Cno	Course
1	DB
2	OS

Cno ⟶ Course

- هنا يعتمد اسم المادة على رقمها اعتماداً وظيفياً كاملاً.

والأن هل الجدول التالي هو في الصيغة المعيارية الثانية 2NF؟

Eno	Ename	Proj_code	Hours	Deptno	Dname
210	Ali	P1	12	10	Research
210	Ali	P2	20	20	Operation
210	Ali	P3	40	20	Operation
201	Salem	P1	30	10	Research
201	Salem	P3	15	20	Operation
305	Ali	P2	40	20	Operation
305	Ali	P3	20	20	Operation

- وللإجابة على ذلك، نجيب على السؤالين التاليين:

1- هل الجدول في الصيغة المعيارية الاولى 1NF؟

- نعم، لأنه لا توجد هناك قيم متكررة، كل عمود يحتوي على قيمة واحدة فقط.

2- هل توجد هناك اعتمادية جزئية؟

ولمعرفة ذلك يجب أن نحدد الاعتمادية الوظيفية

FD1 : Eno ⟶ Ename
FD2 : Proj_code ⟶ Deptno, Dname
FD3 : Eno, Proj_code ⟶ Ename, Deptno, Hours

- المفتاح الرئيسي للجدول هو Eno, Proj_code ولكن Nno يحدد Ename إذاً هناك اعتمادية جزئية، وكذلك Proj_code يحدد Deptno, Dname وهذه اعتمادية جزئية أخرى.
- وللتخلص من هذه المشكلة يجب أن نقوم بتقسيم الجدول إلى جداول بحيث يضم كل منها الجزء من المفتاح والأعمدة التي تعتمد عليه ونبقي فقط المفتاح المركب مع الاعمدة التي تعتمد عليه.

1- نقوم بنقل اسم ورقم الموظف إلى جدول جديد ونبقي نسخة من رقم الموظف في الجدول الأصلي (لأنه جزء من المفتاح الرئيسي).

2- نقوم بنقل رمز المشروع ورقم القسم واسم القسم إلى جدول جديد ونبقي نسخة من رمز المشروع في الجدول الأصلي (لأنه جزء من المفتاح الرئيسي).

Eno	Proj_code	Hours
210	P1	12
210	P2	20
210	P3	40
201	P1	30
201	P3	15
305	P2	40
305	P3	20

Eno	Ename
210	Ali
201	Salem
305	Ali

Proj_code	Deptno	Dname
P1	10	Research
P2	20	Operation
P3	20	Operation

- **الصيغة المعيارية الثالثة Third Normal Form (3NF)**

- The table will be in the 3NF if:

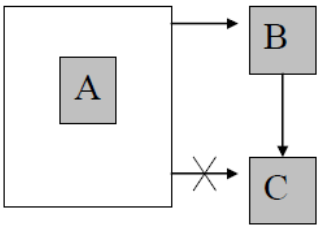
1. The table in the 2NF.
2. The table does not contain a transitive dependency.

- إن الجدول يكون في الصيغة المعيارية الثالثة إذا:

1. كان في الصيغة المعيارية الثانية.
2. لا يحتوي على اعتمادية متعدية.

- **الاعتمادية المتعدية Transitive Functional Dependency:**

- Transitive Functional Dependency is some of the columns (attributes) functionally dependent on the non primary key attribute.
- هي أن تعتمد بعض الأعمدة (الصفات) اعتماداً وظيفياً على صفة غير المفتاح الرئيسي.



- نلاحظ أن A تحدد B,C أي أن B,C تعتمد اعتماد وظيفياً على A، وكذلك أن B تحدد C اي أن C تعتمد اعتماداً وظيفياً على B، وفي هذه الحالة يمكن ان نقول أن هذا الجدول يحتوي على اعتمادية متعدية.
- بمعنى انه هناك صفة تعتمد على صفة أخرى والصفة الأخرى تعتمد على المفتاح الرئيسي، في هذه الحالة تكون الصفة التي في المنتصف هي صفة وسيطة.
- يعني أنها تعتمد بطريقة غير مباشرة على المفتاح الرئيسي، أي أن الصفة لا تعتمد على المفتاح الرئيسي مباشرة، وإنما تعتمد على صفة هي التي تعتمد على المفتاح الرئيسي.
- نلاحظ في الجدول التالي أن رقم القسم يعتمد مباشرة على المفتاح الرئيسي (رقم الموظف)، بينما اسم مدير القسم يعتمد على رقم القسم وليس على المفتاح الرئيسي.

Eno	Deptno	Dept_manager
10	1	Ali
35	2	Hassan
55	3	Noor
20	4	Ahmed

Eno → Deptno → Dept-manager
Deptno → TFD

- في هذه الحالة يكون الحل:

1) نحذف الصفة المتعمدة من الجدول الأساسي.

2) ننشأ جدول جديد به الصفة المعتمدة مع الصفة الوسيطة، ونعين الصفة الوسيطة مفتاح رئيسي.

Eno	Deptno	Dept_manager
10	1	Ali
35	2	Hassan
55	3	Noor
20	4	Ahmed

Deptno	Dept_manager
1	Ali
2	Hassan
3	Noor
4	Ahmed

- والأن هل الجداول التالية في الصيغة المعيارية الثالثة 3NF؟

Eno	Proj_code	Hours
210	P1	12
210	P2	20
210	P3	40
201	P1	30
201	P3	15
305	P2	40
305	P3	20

Eno	Ename
210	Ali
201	Salem
305	Ali

Proj_code	Deptno	Dname
P1	10	Research
P2	20	Operation
P3	20	Operation

- وللإجابة نجيب على التساؤلين التاليين:

1- هل الجداول في الصيغة المعيارية الثانية 2NF؟

- نعم، وذلك لعدم وجود اعتمادية جزئية.

2- هل توجد اعتمادية متعدية؟

- لمعرفة ذلك يجب تحديد الاعتمادية الوظيفية لكل جدول:
- **الجدول الأول:**

FD1 : Eno → Ename

- لا توجد اعتمادية متعدية.

- **الجدول الثاني:**

FD2 : Eno, Proj_code ⟶ Hours

- لا توجد اعتمادية متعدية.
- **الجدول الثالث:**

FD1 : Proj_code ⟶ Deptno, Dname
FD2 : Deptno ⟶ Dname

- المفتاح الرئيسي Proj_code يحدد Deptno, Dname وفي نفس الوقت فأن Deptno يحدد Dname أي أن هناك اعتمادية متعدية.
- وللتخلص من هذه المشكلة نقوم بتقسيم الجدول إلى جداول بحيث يضم كل منها الاعمدة التي تعتمد على بعض، ونبقي المفتاح مع الأعمدة التي تعتمد عليه وحده فقط مع ابقاء المحدد الجديد Deptno.
- أي نقوم بنقل رقم واسم القسم إلى جدول جديد ونبقي نسخة من رقم القسم في الجدول الأصلي:

Eno	Proj_code	Hours
210	P1	12
210	P2	20
210	P3	40
201	P1	30
201	P3	15
305	P2	40
305	P3	20

Eno	Ename
210	Ali
201	Salem
305	Ali

Proj_code	Deptno
P1	10
P2	20
P3	20

Deptno	Dname
10	Research
20	Operation

- **Ex: Convert the following UNF table to 3NF form?**

UNF

St_no	St_name	C_no	Cource	Degree
30	Ahmed	1 2	DB OS	70 80
33	Khaled	1 2	DB OS	62 75
35	Noor	1 2	DB OS	40 50

للتحويل إلى 1NF يجب أن تحتوي كل خلية على قيمة بيانية واحدة:

1NF

St_no	St_name	C_no	Cource	Degree
30	Ahmed	1	DB	70
30	Ahmed	2	OS	80
33	Khaled	1	DB	62
33	Khaled	2	OS	75
35	Noor	1	DB	40
35	Noor	2	OS	50

FD1: St_no ⟶ St_name
FD2: C_no ⟶ Cource
FD3: St_no, C_no ⟶ Degree

اعتمادا على النتائج السابقة يتم وضع مفتاح رئيسي للجدول مكون من رقم الطالب ورقم المادة.

- للتحويل إلى 2NF يجب التحقق من الشرطين:

1- الجدول في الصيغة المعيارية الاولى 1NF.

2- لا توجد هناك اعتمادية جزئية بين الحقول.

- الشرط الاول تحقق لان البيانات مفردة، أما الشرط الثاني فانه لم يتحقق وذلك لأن الحقول غير المفتاحية St_name و Course تعتمد على جزء من المفتاح وليس على كل المفتاح وهذا واضح من خلال الاعتمادات الوظيفية:

FD1: St_no ⟶ St_name
FD2: C_no ⟶ Cource

- لتحويلها الى 2NF نقوم بتقسيم الجدول الى علاقات اعتمادا على الاعتمادات الوظيفية التي جعلت من الجدول ليس في الصيغة المعيارية الثانية وهي:

St_no ⟶ St_name
C_no ⟶ Cource

وتكون النتيجة كما يلي:

FD1: St_no ⟶ St_name
FD2: C_no ⟶ Cource
FD3: St_no, C_no ⟶ Degree

2NF

St_no	St_name
30	Ahmed
33	Khaled
35	Noor

C_no	Cource
1	DB
2	OS

St_no	C_no	Degree
30	1	70
30	2	80
33	1	62
33	2	75
35	1	40
35	2	50

- للتحويل إلى 3NF يجب التحقق من الشرطين:

1. الجدول في الصيغة المعيارية الثانية.
2. لا يحتوي الجدول على اعتمادية متعدية.

- لمعرفة ذلك يجب تحديد الاعتمادية الوظيفية لكل جدول:

الجدول الأول: **FD1 : St_no ⟶ St_name**

- لا توجد اعتمادية متعدية.

الجدول الثاني: **FD2 : C_no ⟶ Course**

- لا توجد اعتمادية متعدية.

الجدول الثالث: **FD3 : St_no, C_no ⟶ Degree**

لا توجد اعتمادية متعدية. إذن تحقق شروط 3NF فتبقى الجداول كما هي.

لغة الاستفسارات المهيكلة Structured Query Language (SQL)

- Structured Query Language (SQL) is a language that dealing with databases, and all relational database applications are rely on it. SQL is working by sending a request to the database engine and get response from it which retrieves a set of results.

- تُعرَّف لغة الاستعلام المهيكلة SQL بأنها لغة التعامل مع قواعد البيانات، وتعتمد عليها كافة التطبيقات التي تتعامل مع قواعد البيانات العلائقية. وتعمل SQL بمبدأ توجيه طلب إلى محرك قاعدة البيانات والحصول على جواب منه والذي يسترجع مجموعة النتائج.
- توفر SQL مجموعة من الأوامر ويمكن تقسيمها إلى قسمين رئيسيين:

1. **لغة معالجة البيانات Data Manipulation Language (DML)**

2. لغة تعريف البيانات Data Definition Language (DDL)

1. تشمل لغة معالجة البيانات (DML) على الأوامر التالية:

- SELECT : يستخدم لاستخراج البيانات من قاعدة البيانات.
- INSERT : يستخدم لإضافة بيانات جديدة.
- UPDATE : يستخدم للتعديل على البيانات.
- DELETE : يستخدم لحذف البيانات.

2. لغة تعريف البيانات Data Definition Language (DDL)

- CREATE TABLE يستخدم لإنشاء الجداول.
- ALTER TABLE يستخدم للتعديل على جدول منشأ سابقاً.
- DROP TABLE يستخدم لحذف جدول منشأ سابقا.
- CREATE INDEX : يستخدم لتكوين المفاتيح.

أمر SELECT:

- يعتبر أمر SELECT من أشهر أوامر اللغة وأكثرها استخداماً، وهو يستخدم لاستعادة وانتقاء مجموعة من البيانات من قاعدة البيانات وذلك باسترجاع جدول يحتوي مجموعة البيانات المطلوبة.
- **الصيغة العامة لأمر SELECT:**

SELECT [field1, field2, ..] FROM [table_name];

- **ملاحظات:**
- تُستخدم إشارة * كبديل لأسماء الحقول.
- يُستخدم تعبير DISTINCT لاستعادة جميع السجلات مع إلغاء التكرار في السجلات المعادة.
- يُستخدم التعبير ORDER BY لترتيب السجلات المُعادة ترتيباً تصاعدياً أو تنازلياً حسب التعبير المرافق المستخدم: ASC للترتيب التصاعدي أو DESC للترتيب التنازلي.
- في حال الرغبة باستخدام أسماء بديلة لحقول جدول القيم المعادة نستخدم التعبير AS.

Ex: Suppose you have a table contains students' data:

Stud

St_no	St_name	Age	Avg	Gender	Address
1	Ahmed	22	70	T	Mosul
2	Khaled	20	65	T	Baghdad
3	Noor	21	80	F	Mosul
4	Mohammed	23	75	T	Anbar
5	Ali	22	60	T	Baghdad
6	Reem	24	90	F	Mosul
7	Ahmed	25	80	T	Mosul

1- Query to select student no. and name?

SELECT St_no, St_name
FROM Stud

الناتج:

St_no	St_name
1	Ahmed
2	Khaled
3	Noor
4	Mohammed
5	Ali
6	Reem
7	Ahmed

2- Query to select all table data?

SELECT *
FROM Stud

الناتج:

St_no	St_name	Age	Avg	Gender	Address
1	Ahmed	22	70	T	Mosul
2	Khaled	20	65	T	Baghdad
3	Noor	21	80	F	Mosul
4	Mohammed	23	75	T	Anbar
5	Ali	22	60	T	Baghdad
6	Reem	24	90	F	Mosul
7	Ahmed	25	80	T	Mosul

3- Query to select students' names without reputation?

SELECT DISTINCT St_name
FROM Stud

الناتج:

St_name
Ahmed
Ali
Khaled
Mohammed
Noor
Reem

4- Query to select students' names and ages ordered by name ascending?

```
SELECT St_name, Age
FROM Stud
ORDER By St_name ASC
```

الناتج:

St_name	Age
Ahmed	22
Ahmed	25
Ali	22
Khaled	20
Mohammed	23
Noor	21
Reem	24

- **الكلمة المفتاحية WHERE**
- تستخدم الكلمة المفتاحية WHERE مع أمر SELECT لاستعادة مجموعة من السجلات التي تحقق شرط أو مجموعة من الشروط.
- يمكن للعبارة الشرطية أن تتضمن عمليات مقارنة مثل (= , < > , < , > , => , =<).
- تقبل الكلمة المفتاحية WHERE أكثر من شرط يفصل بينها عمليات منطقية مثل AND أو OR ويمكن أن يسبق الشرط العملية NOT لنفيه.
- **الكلمة المفتاحية LIKE**
- تُستخدم الكلمة المفتاحية LIKE ضمن العبارة الشرطية، كشرط لوجود مثيل. غالباً ما تُستخدَم هذه الكلمة مع إشارة (%)، التي تضاف إلى القيمة التي نبحث عن مثيلاتها، كبديل عن أي رقم من الأرقام أو الأحرف.
- **الكلمة المفتاحية BETWEEN**
- تُستخدم الكلمة المفتاحية BETWEEN ضمن العبارة الشرطية، كشرط لوجود قيمة محصورة بين قيمتين محددتين.

5- Query to select students' names for males and those lives in Baghdad?

```
SELECT St_name
  FROM Stud
  WHERE  Gender = .T.  AND Address = 'Baghdad'
```

الناتج:

St_name
Khaled
Ali

6- Query to select students' names and ages for names those ends with 'd' character only?

```
SELECT St_name, Age
   FROM Stud
   WHERE  St_name LIKE '%d'
```

الناتج:

St_name	Age
Ahmed	22
Khaled	20
Mohammed	23
Ahmed	25

7- Query to select students' no. , names, and addresses for those ages between 20 and 22?

```
SELECT  St_no, St_name, Address
   FROM Stud
   WHERE  Age  Between  20  AND  22
```

الناتج:

St_no	St_name	Address
1	Ahmed	Mosul
2	Khaled	Baghdad
3	Noor	Mosul
5	Ali	Baghdad

- هناك بعض الدوال الحسابية التي تستخدم مع أمر Select مثل:

الجمع SUM( )، أكبر MAX( )، أصغر MIN( )، المعدل AVG( )، العدد COUNT( )

8- Count the average of students' ages?

```
SELECT  AVG (age)
   FROM Stud
```

الناتج:

Avg_age
22.43

9- Query to select students' names, averages, and with the addition two degrees to the average?

```
SELECT  St_name, Avg, Avg+2 AS AvgPlus
   FROM Stud
```

St_name	Avg	AvgPlus
Ahmed	70	72
Khaled	65	67
Noor	80	82
Mohammed	75	77
Ali	60	62
Reem	90	92
Ahmed	80	82

الناتج:

الكلمة المفتاحية GROUP BY:

- تستخدم الكلمة المفتاحية GROUP BY لعملية تجميع البيانات حسب حقل معين .
- وتستخدم معها الدوال الحسابية التي تم ذكرها مسبقاً مثل SUM، AVG، COUNT..
- فهذه الدوال تحتاج دائماً إلى تثبيت حقل يتم الفرز من خلاله يسمى حقل التجميع grouping_column، ويتعين هذا الحقل عن طريق أمر Group by.
- **الصيغة العامة :**

```
SELECT grouping_column, function(column)
FROM table
GROUP BY grouping_column;
```

10- Count the number of males and the number of females?

```
SELECT Gender,  COUNT(Gender)
 FROM Stud
GROUP BY Gender
```

Gender	Cnt_Gender
F	2
T	5

الناتج:

Ex: Suppose you have a table contains employees' data:

emp

emp_no	emp_name	salary	sepc	dept
1	Mohammed	3500	Manager	Computer
2	Ahmed	2000	Programmer	Computer
3	Noor	3000	Analyst	Reseach
4	Reem	2500	Designer	Computer
5	Ali	2000	Programmer	Reseach
6	Hassan	3000	Analyst	Operation
7	Rana	2500	Programmer	Computer

11- Count the number of employees in every department?

```
SELECT Emp.dept, COUNT(emp_no)
 FROM emp
 GROUP BY Emp.dept
```

الناتج:

dept	Cnt_emp_no
Computer	4
Operation	1
Reseach	2

ملاحظة: يمكن كتابة **COUNT(*)** **بدلا من COUNT(emp_no)**

12- Count the sum of salaries in every department?

```
SELECT Emp.dept, SUM(salary)
 FROM emp
 GROUP BY Emp.dept
```

الناتج:

dept	Sum_salary
Computer	10500
Operation	2000
Reseach	5000

تستخدم الكلمة المفتاحية HAVING مع أمر GROUP BY لغرض وضع شرط على النتائج، ويمكن أن تستخدم مع الدوال الحسابية.

13- Count the average of salaries in every department only for the average is greater than 2500?

```
SELECT Emp.dept, AVG(salary)
 FROM emp
 GROUP BY Emp.dept
HAVING   AVG(Emp.salary) > 2500
```

الناتج:

dept	Sum_salary
Computer	2625
Operation	3000

أمر INSERT

- يستخدم هذا الامر لإضافة بيانات جديدة إلى الجدول.
- **الصيغة العامة لأمر INSERT:**

INSERT INTO table_name
VALUES (value1, value2,....)

- ويمكن ايضا تحديد الحقول المطلوب إضافتها فقط وتكون كالتالي:

INSERT INTO table_name (column1, column2,...)
VALUES (value1, value2,....)

Ex: Suppose you have a table contains students' data:

Stud

St_no	St_name	Age	Avg	Gender	Address
1	Ahmed	22	70	T	Mosul
2	Khaled	20	65	T	Baghdad
3	Noor	21	80	F	Mosul
4	Mohammed	23	75	T	Anbar
5	Ali	22	60	T	Baghdad
6	Reem	24	90	F	Mosul
7	Ahmed	25	80	T	Mosul

Add new student record?

INSERT INTO Stud
VALUES (8, 'Salim', 22, 80, T, 'Mosul')

- **الناتج:**

St_no	St_name	Age	Avg	Gender	Address
1	Ahmed	22	70	T	Mosul
2	Khaled	20	65	T	Baghdad
3	Noor	21	80	F	Mosul
4	Mohammed	23	75	T	Anbar
5	Ali	22	60	T	Baghdad

6	Reem	24	90	F	Mosul
7	Ahmed	25	80	T	Mosul
8	Slaim	22	80	T	Mosul

أمر UPDATE

- يستخدم هذا الامر لتعديل البيانات في الجدول.
- **الصيغة العامة لأمر UPDATE:**

```
UPDATE table_name
SET column_name = new_value
WHERE column_name = value
```

- ويمكن ايضا تحديد الحقول المطلوب إضافتها فقط وتكون كالتالي:

```
INSERT INTO table_name (column1, column2,...)
VALUES (value1, value2,....)
```

Ex: Update the last student record?

```
UPDATE Stude
SET Age = 25, Gender = .F. , Address = 'Baghdad'
WHERE St_no = 9
```

- الناتج:

St_no	St_name	Age	Avg	Gender	Address
1	Ahmed	22	70	T	Mosul
2	Khaled	20	65	T	Baghdad
3	Noor	21	80	F	Mosul
4	Mohammed	23	75	T	Anbar
5	Ali	22	60	T	Baghdad
6	Reem	24	90	F	Mosul
7	Ahmed	25	80	T	Mosul
8	Slaim	22	80	T	Mosul
9	Noor	25	75	F	Baghdad

أمر DELETE

- يستخدم هذا الامر لغرض حذف البيانات من الجدول.
- **الصيغة العامة لأمر DELETE:**

```
DELETE FROM table_name
WHERE column_name = value
```

- ملاحظة : لحذف جميع الصفوف نستخدم الصيغة التالية:

```
DELETE FROM table_name
```

- Or

DELETE * FROM table_name

Ex: Delete the first student record?

DELETE FROM Stud
WHERE St_no = 1

الناتج:

St_no	St_name	Age	Avg	Gender	Address
2	Khaled	20	65	T	Baghdad
3	Noor	21	80	F	Mosul
4	Mohammed	23	75	T	Anbar
5	Ali	22	60	T	Baghdad
6	Reem	24	90	F	Mosul
7	Ahmed	25	80	T	Mosul