المملكة العربية السعودية
جامعة الطائف

نظم قواعد بيانات

996066015

١٤٣٩-٢٠١٨

الجزء الأول

أصول تصميم قواعد البيانات

FUNDAMENTALS OF DATABASE DESIGN

الفصل الأول

مدخل إلى قواعد البيانات

Introduction to Databases

منذ فترة انتشرت قواعد البيانات (databases) و تطبيقات نظم المعلومات (Information Systems Applications) التى تعمل على قواعد البيانات و أصبحت عنصراً جوهرياً فى تسيير أمور الحياة اليومية فى المجتمع المعاصر. فنجد أننا نستخدمها فى الأنشطة المختلفة الرياضية و التعليمية و الإدارية و الطبية و الزراعية و السياسية و التجارية و الاستثمارية المختلفة.

١-١ تمهيد Preface

يصادف معظمنا أنشطة عديدة تشمل بعض الاتصالات مع قاعدةٍ ما للبيانات. مثال على ذلك تسجيل المواليد و الوفيات و النتائج الدراسية و وثائق السفر و تحقيق الشخصية و عند إيداع أو سَحْب وديعة مصرفية أو حَجْز فى الفنادق أو خطوط طيران أو صرف رواتب أو شراء سلع أو دفع فاتورة هاتف أو كهرباء أو مياه، فإننا نستخدم أحد قواعد البيانات.

كافة الأنشطة السابقة تدخل فى نطاق التطبيقات التقليدية لقاعدة البيانات لأن أغلب المعلومات المتناولة إما أن تكون كلاميةً أو رقمية. و توجد حالياً تطبيقات

متقدمة لقواعد البيانات مثل استخدام تطبيقات الذكاء الاصطناعى فى التنقيب عن البيانات وتطبيقات قواعد البيانات فى التجارة الإلكترونية (البيع و الشراء عبر الإنترنت). هناك أيضاً تطبيقات لقواعد البيانات ذات الوسائط المتعددة لتخزين الصور و أفلام الفيديو و الصوتيات.

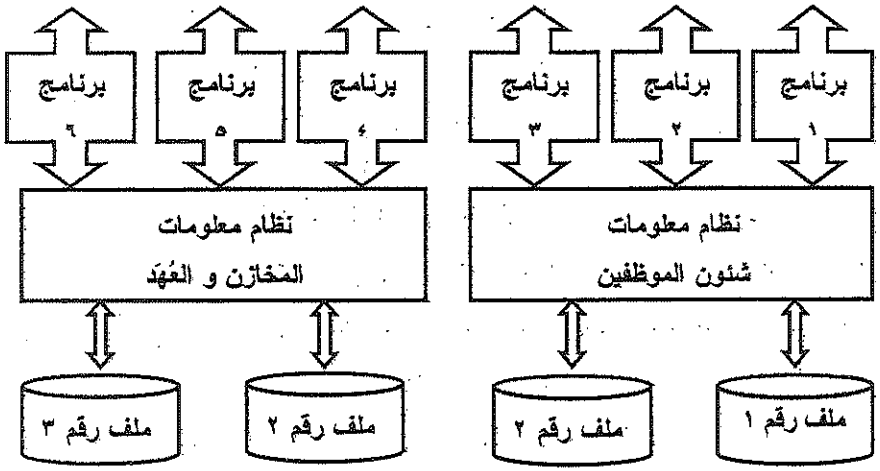
تلعب قواعد البيانات دورا حساسا فى كل المجالات التى تستخدم الحاسبات تقريبا مثل الصناعة و الزراعة و التجارة والهندسة و الطب والقانون والتعليم و المكتبات. حيث نخزن البيانات ذات الصلة ببعضها البعض فى قاعدة البيانات، مثل أسماء و رموز المقررات الدراسية التى ندرسها و كذلك عدد الساعات المعتمدة و الدرجات التى نحصل عليها فى الامتحانات و مواعيد تلك الامتحانات.

١-٢ نظم ملفات البيانات Data File Systems

ظل مبرمجو نظم المعلومات لفترة طويلة يبنون برامجهم مع استخدام على ملفات البيانات و قد أدت الملفات دوراً هاماً فى تخزين المعلومات. صاحب استخدام الملفات بعض المشاكل والعيوب التى تمت مواجهة بعضها و التعايش مع البعض الآخر مع بذل الكثير من الجهد و الوقت لجعل تأثير تلك المشاكل فى أضيق الحدود.

١-٢-١ معالجة ملفات البيانات Data File Processing

فى نظام معالجة ملفات البيانات، كل برنامج يُصمَّم لأداء غرض معين وله الملفات الخاصة به دون وجود إطار عام يربط جميع البرامج و/ أو يسمح بإضافة برامج جديدة بسهولة و ربما تكررت البيانات في أكثر من ملف. الشكل ١-١ يبين علاقة ملفات البيانات بالبرامج.



شكل ١-١ : نظام معالجة ملفات البيانات .

١-٢-٢ أنواع ملفات البيانات Data Files Techniques

المبرمجون يستخدمون عدة أنواع لملفات تخزين البيانات. كلٌ منها له استخدامات و كذلك طريقة استخدام خاصة به تبعاً لطريقة تخزين و استخراج للبيانات المُخزّنة به. فيما يلى نذكر أنواع البيانات بإيجاز شديد :

- **ملف تتابعى (Sequential File)** : فى هذا النوع يتم تخزين سجلات البيانات بشكل تتابعى بنفس ترتيب وصولها للملف بعد آخر سجل موجود. لاسترجاع السجلات المخزّنة تجرى قراءة السجلات من أول سجل فى الملف إلى آخر سجل أو إلى السجل المطلوب بشكل تتابعى.

- **ملف عشوائى (Random File)** : فى هذا النوع يتم التخزين بترتيب عشوائى مع معرفة موقع (عنوان) كل سجل عن طريق معرفة المسافة بين موقع كل سجل من بداية الملف (**الإزاحة offset**). وعند الحاجة إلى قراءة أى سجل تتم قراءته مباشرة دون الحاجة إلى قراءة أى سجل آخر و ذلك باستخدام مقدار الإزاحة لكل سجل.

- **ملف مفهرس (Indexed File)** : يختلف هذا النوع عن النوعين السابقين حيث يستخدم فهرس أشبه بفهرس الكتاب. نستطيع عن طريق الاطلاع على فهرس الكتاب و عن طريقه يمكننا الوصول إلى أى موضوع نرغب فى الاطلاع عليه. كذلك يتم عمل فهرس للسجلات المُدْخَلة باستخدام

أحد حقول السجل. بناءً على طريقة التخزين تلك نستطيع استخراج أى سجل مباشرة باستخدام مفتاحه أى فهرسه.

١-٢-٣ مشاكل استخدام الملفات File Problems

كما رأينا فى الشكل ١-١ أن كل تطبيق يستخدم ملفات خاصة به و ربما تستخدم عدة تطبيقات ملفين متشابهين يحتويان على نفس البيانات مثل البيانات الأساسية للموظف التى يستخدمها نظام معلومات شئون الموظفين و نظام معلومات رواتب الموظفين. ينشأ عن ذلك مشاكل عديدة نحصر منها ما يلى :

- **تكرار البيانات (UnControlled Redundancy)**

يحدث تكرار لنفس البيانات فى أكثر من ملف أو حتى فى نفس الملف مما يسبب ضياع حيز التخزين و الجهد و الوقت اللازمين لذلك.

- **عدم تجانس أو توافق البيانات (Incompatible Data)**

نظراً لإدخال نفس المعلومة فى عدة أماكن فربما يتم إدخالها فى مكان ما بشكل مختلف عن الأماكن الأخرى. كذلك من الممكن أن نجرى تعديلاً أو حذفاً لأحد السجلات فى ملف دون الآخر. كل هذا يسبب عدم التجانس أو التوافق بين البيانات.

- **عدم المرونة (Inflexibility)**

تتميز نظم معالجة ملفات البيانات بعدم مرونتها فى القابلية للتعديل عند طلب تعديل شكل أو حجم أو نوع البيانات و تتطلب جهداً و وقتاً كبيراً و بالتالى تكلفة عالية.

- **الافتقار إلى المواصفات القياسية (Poor Standards)**

فى العادة لا يلتزم المبرمجون بنفس المواصفات القياسية من ناحية مسميات و أطوال حقول البيانات أو القيود المفروضة عليها لأنها ليست إجبارية من النظام أو من مدير النظام.

- **معدل منخفض لإنتاج البرامج (Low Productivity)**

نظراً لأن المبرمج فى نظم الملفات يقوم بتصميم و استخدام كل سجل مع فتح و غلق كل برنامج للملفات أكثر من مرة مع قيام المبرمج بكتابة جمل لكل شئ داخل برامجه فإن معدل إنتاج البرامج يكون بطيئاً نسبياً.

- **مشاركة محدودة (Limited Sharing)**

تكون مشاركة البرامج و كذلك المستخدمين فى النظم و ملفات البيانات محدود جداً.

- **صعوبة الصيانة (Complicated Maintenance)**

نظراً لوجود وصف لبيانات و سجلات الملفات داخل كل برنامج فعند إجراء أى تعديل يلزم تعديل جميع البرامج التى تتعرض للجزء المعدل ثم

إعادة الترجمة و ما يمكن أن يصاحب تلك الخطوات من صعوبات و أخطاء.

- **أمن و سرية البيانات محدودة (Limited Data Security)**

تتضمن نظم معالجة ملفات البيانات إمكانية تأمين البيانات و حمايتها من المستخدمين الغير مُصَرَّح لهم بتناولها و لكن على نطاق محدود لملف بالكامل مثلاً و ليس لسجل أو حقل أو عنصر بيانات.

١-٣ نظام قواعد البيانات Database System

لقد نشأت قواعد البيانات و نظم قواعد البيانات من أجل إيجاد بديل لملفات البيانات و نظم معالجتها بحيث يحل كافة المشكلات و القيود و الصعوبات التى يواجهها المستخدمون فى تعاملهم مع الملفات. و لقد نجحت بالفعل أن تطيح ملفات البيانات بشكل كبير دامغ فى نظم المعلومات التى تُبْنَى الآن على قواعد البيانات.

١-٣-١ البيانات و المعلومات و قواعد البيانات
Data and Information and Database

البيانات (data) هى كافة البيانات المطلوب إدخالها والاستعلام عنها. كل بيان يمثل عنصر مستقل مثل : (اسم المريض، رقم الغرفة، عنوان طبيب .....).

المعلومات (Information) هى البيانات التى تمت معالجتها و وضعها فى صورة ملائمة و مفهومة للمستخدم.

قواعد البيانات هى أسلوب محدد لتنظيم المعلومات يبسط كيفية إدخالهـا وتعـديلها واستخراجها إمّا بنفس الشكل المدخل أو مجمعة في صورة إحصائيات أو تقارير أو شاشات استعلام، مع التحكم في كل صغيرة وكبيرة.

تصميم قاعدة بيانـات يشمل تحديد أنواع البيانات والتراكيب والقيود على البيانـات فى قاعدة البيانات.

بناء قاعدة بيانات فهو عملية تخزين البيانات نفسها فى وسط تخزين يـتحكم فيـه DBMS.

معالجة قاعدة بيانات تتضمن وظائف مثل الاستعلام من قاعدة البيانات لاسـتخراج بيانات معينة و تعديل قاعدة البيانات لتعكس التغيرات فى العالم المصغر ثم إنتـاج تقارير من البيانات.

عند تصميم قاعدة يجب أن نلم بمجموعات المستخدمين لها و التطبيقات المحصورة سلفا والتى يهتم بها هؤلاء المستخدمين. بتعبير آخر فإن قاعدة البيانـات لها مصدر تشتق منه البيانات و درجة معينة من الاتصال مع أحداث فـى العـالم الواقعى ولها جمهور من المهتمين التشطين بمحتويات قاعدة البيانات.

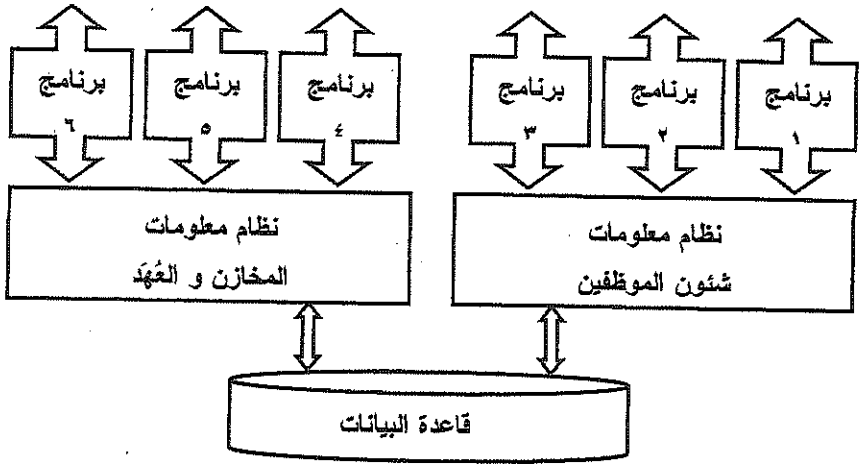
١-٣-٢ قواعد البيانات و نظم إدارة قواعد البيانات

Database and Database Management Systems

يمكن إنتاج ومعالجة قاعدة باستخدام الحاسب الآلى بواسطة مجموعة من بـرامج التطبيقات المكتوبة خصيصا لهذا الغرض أو بواسطة **نظم إدارة قواعـد البيانـات (Database Management Systems (DBMS))** مثـل Access أو Oracle أو Power Builder أو Informix أو Sybase أو غير ذلك.

نظام إدارة قاعدة البيانات هو مجموعة البرامج التى يمكن استخدامها فـى إنشاء ومعالجة قاعدة بيانات ما. أى أن نظام DBMS هو نظام برامجـى متعـدد الأغراض يسهل تعريف وبناء ومعالجة قواعد البيانات للتطبيقات المختلفة.

من الممكن أن يتم تصميم قاعدة بيانات واحدة تستخدمها كافة البـرامج و التطبيقات التى تخص جهةٍ ما أو تخصيصها لتطبيق معين. تتميز قاعـدة البيانـات بأن تخزين أى بيان يتم فى مكان واحد فقط تتأثر به كافة البرامج و التطبيقات التى تتناول قاعدة البيانات. الشكل ١-٢ يبين ذلك.



شكل ١-٢ : نظام معالجة قواعد البيانات .

فى الغالب نُطوِّر قاعدة البيانات بحيث تستوعب بيانات جزء من العالم الواقعى و تعكس كافة التغييرات التى تحدث فيه. يجب أيضاً أن تكون البيانات ذات صلة و مترابطة و تشكل معلومات منطقية أى تؤدى معانى مفيدة. حيث يتم تصميم وبناء قاعدة البيانات و تعبئها بالبيانات لغرض محدد.

يمكن أن تكون قاعدة البيانات فى أى حجم وذات درجات مختلفة من التركيب و التعقيد. مثال على ذلك قائمة المدرسين فى مدرسةٍ ما، تتكون من مئات السجلات و كل سجل له تركيب بسيط. من ناحية أخرى نجد قائمة سجلات مواطنى دولةٍ ما فى مصلحة الأحوال المدنية ممكن أن تحتوى على ملايين

السجلات للمواطنين. تهتم قاعدة البيانات بالبيانات الموجودة فى مصلحة الأحوال المدنية حيث يتم تسجيل بيانات المواطنين حال الولادة و عند استخراج بطاقة تحديد الشخصية أو جواز السفر و بطاقات إقامة الأجانب و ربطها بتأشيرات السفر و عقود الزواج. هذه الكمية الكبيرة من المعلومات يجب أن ترتب وتتم أدارتها بحيث يتمكن المستخدمين من البحث عن أو تعديل البيانات حسب حاجتهم.

١-٣-٣ مميزات استخدام قواعد البيانات

Database Advantages

كما رأينا فى الشكل ١-٢ أنه من المكن أن تستخدم كافة البرامج أو التطبيقات قاعدة بيانات واحدة تضم كافة البيانات التى تحتاجها البرامج و التطبيقات فى شكل متكامل موحد. فمثلاً، تستخدم عدة تطبيقات قاعدة بيانات واحدة تحتوى مرة واحدة فقط على البيانات الأساسية للموظف التى يستخدمها نظام معلومات شئون الموظفين و نظام معلومات رواتب الموظفين. يترتب على استخدام قواعد البيانات فوائد متعددة نذكر منها ما يلى :

- **ندرة تكرار البيانات (Minimal Data Redundancy)**

نظراً لاستخدام قاعدة بيانات واحدة فأى بيان لا يتم تسجيله أكثر من مرة. فقط يحدث تكرار لعدد محدود من حقول البيانات بشكل يتحكم فيه مصمم قاعدة البيانات من أجل الربط بين البيانات بعضها البعض. لا يحدث تكرار لنفس البيانات فى أكثر من ملف أو حتى فى نفس الملف ما يمنع ضياع حيز التخزين و الجهد و الوقت اللازمين لذلك.

- **تجانس أو توافق البيانات (Compatible Data)**

يترب على عدم تكرار نفس البيانات داخل قاعدة بيانات واحدة عدم وجود أى بيانات غير متجانسة. ذلك لأن إدخال المعلومة أو تعديلها أو حذفها يتم فى نفس قاعدة البيانات و تتأثر به كافة التطبيقات التى تتناول القاعدة. كل هذا يسبب تجانس و توافق البيانات.

- **توفر المرونة (Flexibility)**

تتميز نظم معالجة قواعد البيانات بالمرونة الكبيرة و القابلية للتعديل عند طلب تعديل شكل أو حجم أو نوع البيانات و تتطلب جهداً و وقتاً بسيطاً جداً و بالتالى تكلفة منخفضة.

- **توفر المواصفات القياسية (High Standards)**

فى العادة يضع مصمم قاعدة البيانات قيوداً على البيانات و على علاقتها ببعضها البعض. هذه القيود يفرضها النظام على جميع المتعاملين مع قاعدة البيانات مما يضمن توفر مواصفات قياسية عالية من ناحية مسميات و أطوال حقول البيانات و القيود المفروضة عليها لأنها إجبارية من النظام.

- **معدل عالى لإنتاج البرامج (High Productivity)**

نظراً لأن المبرمج فى نظم قواعد البيانات يتناول قاعدة البيانات مباشرة و يركز جهده على بناء النظام من شاشات إدخال أو استعلام و تقارير فإن

معدل إنتاج التطبيقات يكون عالياً نسبياً بسبب سهولة إنتاج البرامج المبنية على قواعد البيانات.

- **مشاركة كبيرة (Wide Sharing)**

توفر نظم قواعد البيانات مشاركة كبيرة مع تعدد مستخدمى النظم. كما توفر برامج التحكم المتزامن التى تسيطر و توجه عمل المستخدمين لقواعد البيانات بحيث لا تتضارب أو تتفاوت البيانات.

- **سهولة الصيانة (Complicated Maintenance)**

نظراً لأن التطبيقات تتناول نفس قاعدة البيانات فإن إجراء أى تعديل يتم فى موضع واحد فى قاعدة البيانات بسهولة و يسر و تحت مسئولية المختص.

- **أمن و سرية البيانات عالية جداً (Strong Data Security)**

تتضمن نظم قواعد البيانات إعطاء صلاحيات لبعض المستخدمين أو مجموعات المستخدمين مع التحكم فى نوع الاستخدام لكل عنصر فى قاعدة البيانات و عند أى مستوى داخل قاعدة البيانات. هذا يؤمن البيانات تأميناً عالياً ضد المستخدمين الغير مصرح لهم بتناول أجزاء معينة فى قاعدة البيانات.

- **تكامل البيانات (Data Integration)**

يتم تصميم قاعدة البيانات بحيث تتكامل البيانات المجزأة معا لوجود علاقات رابطة بين الكيانات المجزأة و كلاهما يمثلان أجزاء مستقلة مترابطة من قاعدة البيانات.

- **تجزئة البيانات (Data Granularity)**

تصميم قاعدة البيانات فى شكل بيانات مُجَزَّأة يتيح اكبر قدر من التحكم فى عناصر البيانات و السجلات المختلفة.

- **تحديث فورى للبيانات (Immediate Update of Data)**

تصميم قاعدة البيانات بالشكل القياسى المتكامل و توحيد مصدر و مَصَب المعلومات التى تتناولها كافة التطبيقات يتسبب فى أن أى تحديث سواء كان تعديل أم إضافة أم حذف لبيانات معينة، متاحاً و فورياً لكافة التطبيقات التى تتناول قاعدة البيانات.

- **استعادة البيانات والنُسَخ الاحتياطية للبيانات (Data Recovery & Backup Independence)**

توفر نظم إدارة قاعدة البيانات برامج لتوفير النسخ الاحتياطية من قاعدة البيانات. هذا بالإضافة إلى وجود برامج و تقنيات تقوم باستعادة البيانات

عند حدوث أى أعطال غير تدمير البيانات. و حتى فى حالة تدمير البيانات يمكن الاستعانة بالنسخة الاحتياطية.

- **استقلالية البيانات (Data Independence)**

تصميم قاعدة البيانات بحيث تتكامل البيانات المجزأة معاً و بشكل منفصل تماماً عن التطبيقات التى تتناولها يجعل عملية صيانة التطبيقات الموجودة أو حتى بناء تطبيقات جديدة تتناول نفس قاعدة البيانات تتم بعيداً عن تلك القاعدة و تتأثر بها و لا تؤثر فى هيكلها التركيبى. يمكن كذلك تواجد قاعدة البيانات على جهاز خادم و أى نظم تعمل على أجهزة أخرى يمكنها تناول تلك التطبيقات. كذلك عند تعطل النظم التى تعمل عليها التطبيقات لا تتأثر قاعدة البيانات بذلك.

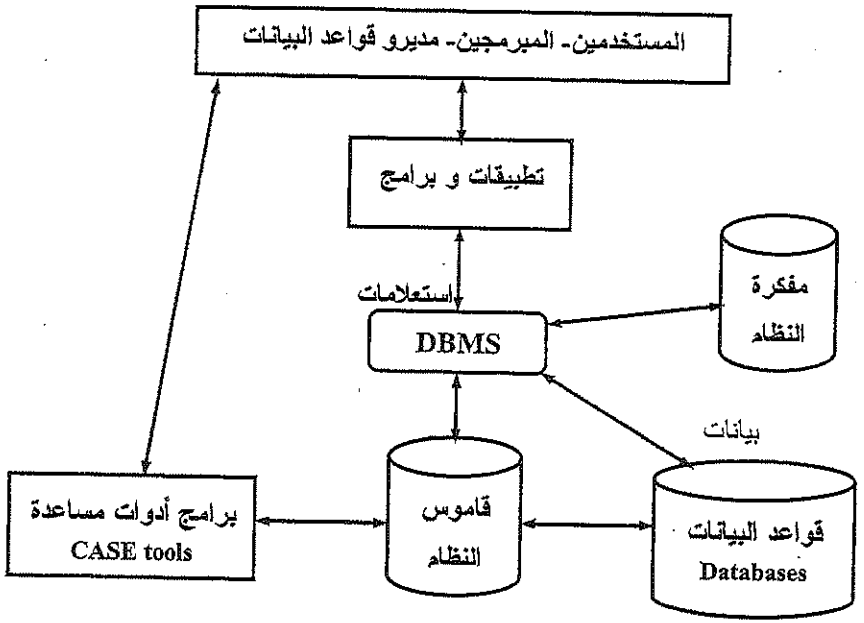
١-٤ بيئة قواعد البيانات Database Environment

يوضّح الشكل ١-٣ بإيجاز بيئة قاعدة البيانات. و كما نرى فإن الشكل يبين تنوع القائمون على قواعد البيانات و أن جميع التعاملات مع قواعد البيانات تتم من خلال برامج و تطبيقات أو أدوات مساعدة تمكننا من التعامل مع برمجيات DBMS.

نرى أيضاً فى الشكل ١-٣، مفكرة النظام (system log) التى يتم تسجيل كافة التعاملات التى تجرى على قواعد البيانات مما يفيد فى استعادة البيانات المفقودة. كذلك يوجد قاموس البيانات الذى يحتوى على مسميات كافة عناصر البيانات المستخدمة فى قاعدة البيانات و فهارس و تراكيب البيانات والقيود المفروضة على كل هؤلاء.

١-٥ القائمون على قواعد البيانات Database Peoples

يعمل من خلال بيئة قاعدة البيانات فريق كبير يتضمن نوعيات و تخصصات مختلفة للقائمين عليها،حيث يوجد من يقوم على تصميم و تطوير و إدارة و صيانة و تشغيل و استخدام قاعدة البيانات. نتعرّف على القائمون على قواعد البيانات من خلال التخصصات والوظائف.



شكل ١- ٣ : بيئة قواعد البيانات و القائمون عليها.

١-٥-١ إدارة قاعدة البيانات Database Administrator

يوجد فى أى شركة أو هيئة أو وزارة لديها قاعدة بيانات فريق عمل مسئول عن إدارة قاعدة البيانات يرأس الفريق مدير قاعدة البيانات (Database Administrator (DBA).

يكون DBA و فريقه مسئولين عن التحكم فى كافة إمكانيات قواعد البيانات و استخداماتها و كذلك إمكانيات واستخدامات نظام إدارة قاعدة البيانات. تتضمن مسئوليات الإدارة الترخيص بالصلاحيات لكافة المتعاملين مع قواعد البيانات أو البيانات. كذلك عليهم مراقبة عمل القاعدة و أدائها و كذلك أى تجاوزات أو انتهاكات تحدث من قبل المستخدمين. أيضاً من أهم مهامهم تنظيم عملية النسخ الاحتياطى للبيانات.

١-٥-٢ إنتاج قاعدة البيانات و تطبيقاتها
Development of Database and its Applications

هذا الجانب يتضمن جانباً هاماً من الوظائف التى نحتاجها لإنتاج قاعدة البيانات و تطبيقاتها. يشمل عدة وظائف تبدأ بالتحليل ثم التصميم ثم التطوير و التشغيل:

- **تحليل النظم (System Analysis)**

مسئولية محلل النظم (system analyst) هى تحليل متطلبات الجهة التى سوف تستخدم النظام و من ثم تحديد كافة البيانات و علاقاتها ببعضها

البعض. كذلك يوفر التحليل نماذج و أشكال و أنواع البيانات و القيود المفروضة عليها.

- **تصميم قاعدة البيانات (Database Design)**

عملية تصميم قاعدة بيانات التى يقوم بها مصمم قاعدة البيانات (database designer) هى تحديد البيانات التى ستخزن فى قاعدة البيانات لاختيار التراكيب المناسبة لتمثيل و تخزين المعلومات.

حيث يحصل مصمم قاعدة البيانات على نتيجة مرحلة التحليل و عن طريق الاتصال بكل مستخدمى قاعدة البيانات المتوقعين مستقبلا لكى يفهم متطلباتهم ومن ثم الوصول إلى تصميم يحقق متطلباتهم.

بعد تجميع كافة البيانات السابقة و طلبات المستخدمين المتعددة و المختلفة على مصمم قاعدة البيانات أن يضع كل هذه الأفكار فى رؤية واحدة تضمن شمولها لكافة البيانات المطلوبة من قِبَل جميع المستخدمين مع سهولة تخزينها أو استرجاعها بأشكال مختلفة حسب متطلبات التشغيل.

- **تطوير قاعدة البيانات (Database Development)**

عملية تطوير و إنشاء و اختبار قاعدة البيانات قبل وضعها فى مرحلة التشغيل تقع فى الغالب على أحد أعضاء فريق إدارة قاعدة البيانات لضمان سلامة مكونات قاعدة البيانات.

- **تطوير تطبيقات قاعدة البيانات (Database Applications)**

يقوم **المبرمج (programmer)** و**مطور التطبيقات (application developer)** بتطوير نظم المعلومات التى تتناول قاعدة البيانات من خلال برامج التطبيق. غالباً ما توفر نظم DBMS آلية تسهل للمبرمجين و المطورين إنشاء تطبيقاتهم مثلما تفعل ACCESS أو ORACLE فى Developer2000 و كافة نظم DBMS الأخرى.

١-٥-٣ تناول قاعدة البيانات Accessing Database

فى الغالب يتناول قاعدة البيانات مستخدمون ذوى خبرات متفاوتة فبعضهم لا علاقة لهم و لا دراية بقواعد البيانات أو كيفية تصميمها أو تطويرها أو تشغيلها و البعض الآخر ربما تكون معرفتهم بالحاسب سطحية و بسيطة، بينما هناك جانب منهم على دراية و خبرة كبيرة فى التعامل مع قواعد البيانات. نطلق على الجميع **المستخدمون النهائيون (end users)**.

يشمل المستخدمون النهائيون كافة المتعاملين مع قواعد البيانات مثل القائمين على حجز مقاعد الطيران أو حجز الفنادق و تسجيل بيانات الطلبة و الأعمال الإدارية من أجل الحصول على التقارير و الإحصائيات. كذلك هناك صرافوا البنوك و موظفى المطارات و الموانئ الحدودية.

غالبية نظم DBMS توفّر الأدوات و البرامج المساعدة التى تجعل استخدام المستخدم النهائى لتطبيقات قاعدة البيانات سهلة و مريحة دون الحاجة إلى تدريب كبير أو تخصص فى الحاسب الآلى.

١-٥-٤ تشغيل و صيانة قاعدة البيانات

Database Operating and Maintenance

ضمن العاملين فى بيئة قاعدة البيانات **المشغلون** القائمون بالتشغيل الفعلى (**operators**) و كذلك **مسئولو الصيانة** (**maintenance personnel**) وهم مسئولو الصيانة لبيئة البرمجيات و المكونات الصناعية لنظام قاعدة البيانات.

١-٦ أسئلة

١. أذكر بعض استخدامات قواعد البيانات فى حياتنا اليومية.

٢. أذكر مع التوضيح أنواع ملفات البيانات المتعارف عليها.

٣. وضّح المشاكل الناشئة من استخدام ملفات البيانات.

٤. عرّف البيانات والمعلومات، قواعد البيانات و نظم قواعد البيانات.

٥. أذكر مميزات استخدام قواعد البيانات.

٦. ما المقصود بـ DBMS؟ و ما هى وظيفته؟

٧. بين وظيفة كلاً من : قاموس البيانات و مفكرة النظام فى DBMS.

٨. حدد دور كلاً مما يلى فى بيئة قاعدة البيانات :
مدير قاعدة البيانات، مصمم قاعدة البيانات، مطوّر قاعدة البيانات، مشغّل قاعدة البيانات.

الفصل الثاني

مبادئ قواعد البيانات العلاقية

Basics of Relational Databases

قاعدة البيانات هى أسلوب محدد لتنظيم المعلومات يبسط كيفية الإدخال والإخراج فى إطارات مختلفة مع التحكم في كل كبيرة وصغيرة وخفض حجم البرمجة. فى الماضى القريب كانت قواعد البيانات المتعارف عليها هى قواعد البيانات الشبكية (Network Database) و قواعد البيانات ذات التسلسل الهرمى (Hierarchical Database). و ظلّت هذه الأنواع مستخدمة إلى وقت قريب. بعد ذلك ظهرت قواعد البيانات العلاقية و ظلّ المبرمجون يستخدمون الثلاثة أنواع لقواعد البيانات بالإضافة إلى استخدام بعض الجهات لملفات البيانات التقليدية.

لكن بعد فترة ظهرت نظم DBMS متميزة، لإدارة قواعد البيانات العلاقية (Relational Database Management Systems(RDBMS)). و نظراً لقوة نظم RDBMS و كفاءتها و تلبيتها لحاجة المبرمجين مع توفير أدوات مساعدة قوية لهم، فقد طغت RDBMS و أصبحت هى النوع الوحيد المستخدم و أصبح ما عداها تراثاً و تاريخاً.

تعتمد قاعدة البيانات العلاقية فى تصميمها على المفاهيم الطبيعية الموجودة فى بيانات نموذج العالم المُصَغَّر الذى تمثله هذه القاعدة. أساس قواعد البيانات

العلاقية هو العلاقات الرابطة بين البيانات و التى تعتبر الجزء الأهم و الذى يمثل أغلب التعاملات مع قاعدة البيانات.

٢-١ نموذج قاعدة بيانات بسيطة

Simple Database Model

نموذج قاعدة البيانات (Database Model) هو نموذج يبين لنا صورة كاملة لنظام معلومات و وظائفه و القيود الموجودة فيه، و يركز على التكامل بين البيانات.

الآن نعرض مثالا ربما يكون مألوفا لدى أغلب القراء : وهو قاعدة بيانات مبسطة للمستشفى (HOSPITAL) لمعالجة معلومات و بيانات المرضى و الغرف المُنَوَّمين بها و كذلك الأدوية المنصرفة لهم و المصانع المنتجة لها. سوف نستخدم الكلمة ملف كمكان لتخزين البيانات مؤقتاً حتى نتعرَّف على مفاهيم و أساليب تصميم قاعدة البيانات. الشكل ٢-١ يعرض مكونات قاعدة بيانات المستشفى و عينة بيانات قليلة لقاعدة للبيانات المذكورة.

وقاعدة البيانات منظمة فى أربعة ملفات كل منها يخزن سجلات بيانات من نفس النوع ويمكن أن تكون تلك السجلات مرتبة أو غير مرتبة. من الرسم نجد أن ملف PATIENT يخزن بيانات عن كل مريض، و يخزن ملف MEDICINE

بيانات الأدوية، بينما ملف ROOM فيخزن بيانات عن غرف تنويم المرضى و نخزن بيانات علاج المرضى فى ملف TREATED_BY.

PATIENT (مريض)

Pno (رقم المريض)	Name (الاسم)	Sex (الجنس)	Room# (رقم الغرفة)	Doctor (الطبيب)
٦٧٦	خديجة	٢	١٠٠	سيف
٤٥٤	ناصر	١	٢٠٠	محمد
٨٩٨	خالد	١	٣٠٠	دعاء
٩٥٥	آلاء	٢	١٠٠	عزة

MEDICINE (دواء)

Mno (رقم الدواء)	Mname (اسم الدواء)	Factory (المصنع المنتج)
S103	Abc1	المصرية
S363	Abc2	السعودية
S325	Abc3	الدولية

ROOM (غرفة)

R# (رقم الغرفة)	Phone (رقم الهاتف)	Bed# (عدد الأسرة)
١٠٠	٢٣٢	٢
٢٠٠	٤٥٣	٢
٣٠٠	٥٤٥	١

TREATED_BY(يُعَالَج بواسطة)

Patient# (رقم المريض)	Medicine# (رقم الدواء)	Q(الكمية المنصرفة)
٦٧٦	S103	٣
٤٥٤	S363	٢
٨٩٨	S325	٤

شكل ٢-١ : نموذج مبسط لقاعدة بيانات لمستشفى.

لتعريف قاعدة البيانات يجب أن نحدد تركيب السجلات التى يمكن تخزينها فى كل ملف بتحديد الأنواع المختلفة لعناصر البيانات المخزنة فى كل سجل. ففى الشكل ٢-١ نرى أن سجل كل مريض يحتوى على بيانات لتمثيل رقم المريض و اسمه و رقم الغرفة التى يوجد بها و كذلك اسم الطبيب المعالج.

كذلك يحتوى سجل كل دواء على بيانات لتمثيل رقم الدواء و اسمه و اسم المصنع الذى أنتجه. بينما يحتوى سجل كل غرفة على رقم الغرفة و رقم الهاتف و عدد الأسرة الموجودة فى تلك الغرفة. أخيراً يحتوى كل سجل فى TREATED_BY على بيانات تمثيل رقم المريض و رقم الدواء و كميته.

يجب أيضا أن نحدد نوع بيانات كل عنصر بيانات داخل السجل مثل تحديد أن اسم المريض هو سلسلة من الحروف الهجائية (String) و رقم المريض

يكون عدداً صحيحاً (Integer) و غيرها من أنواع البيانات مثل حرف واحد (Char) أو نستخدم أيضا ترميزاً (Coding) لتمثيل عنصر البيانات كما سنرى فى الأمثلة التى نتعرض لها خلال أجزاء هذا الفصل.

لبناء قاعدة بيانات HOSPITAL فإننا نخزن البيانات لتمثيل كل مريض و دواء و غرفة كسجل فى الملف المناسب. لاحظ أن السجلات فى الملفات المختلفة يمكن أن تكون ذات صلة ببعضها البعض مثلا سجل "محمد" فى ملف PATIENT ذو صلة بسجل TREATED_BY الذى يحدد كمية الدواء المنصرفة للمريض محمد. و نجد بالمثل الكثير من العلاقات فى أغلب قواعد البيانات التى تتضمن أنواعا عديدة من السجلات يربط بين بعضٍ منها عدة علاقات رابطة (Relationships).

تتضمن معالجة قواعد البيانات الاستعلام و التعديل. أمثلة على الاستعلام : " أستخرج قائمة بالمرضى المنومين فى الغرفة رقم ١٠٠ " و "أستخرج أسماء المرضى الذين يعالجهم الطبيب سيف و عولجوا بالدواء S103 ". أمثلة على التعديلات : "**عدّل اسم الطبيب المعالج للمريض محمد إلى شادى**" و "**أنشأ غرفة جديدة رقمها ٤٠٠ بها ٤ أسرة و هاتفها٨٩٨**". هذه الاستعلامات و التعديلات العامة يجب تحديدها بدقة فى لغة نظام قاعدة البيانات قبل أن نتمكن من تنفيذها.

٢-٢ مفاهيم أساسية Basic Concepts

نتعرَّف الآن على المفاهيم الأساسية المتعارَف عليها فى قواعد البيانات العلاقية. حيث نتعرَّف أيضاً على المصطلحات و الرموز المستخدمة فى نموذج تصميم قاعدة البيانات.

٢-٢-١ تعاريف أساسية Basic Definitions

يجب على دارس قواعد البيانات الإلمام الكامل بالتعاريف التالية :

- **البيانات (data)**

هى أى حدوث للبيانات التى تصف أى كائن.

- **البيانات وصفية (metadata)**

هى البيانات التى تصف البيانات المخزنة وصفاً دقيقاً. ويُطلَق عليها metadata (data about data).

- **الكيان (Entity)**

هو وحدة معلومات تمثل فئة أو مجموعة من الأشياء أو الكائنات أو الأنشطة. هذه الوحدة لها مواصفات (خصائص) تصفها و تخصها و تُعبِّر عن مجموعة الكائنات التى تنتمى إليها. هذه المجموعة هى أمثلة أو حالات

أو نماذج أو كائنات تتبع هذا الكيان و كل منها عبارة عن قيم للخصائص التى تصف هذا الكيان. فى أغلب الأحوال يكون الكيان اسماً مفرداً.

أمثلة على الكيانات : أنظر إلى الكيانات التى تعلو أسمائها الجداول الموجودة فى الشكل ٢-١ (ROOM و PATIENT و MEDICINE). أمثلة لكيانات أخرى : STUDENT (طالب) و CAR (سيارة) و COURSE (مقرر دراسى) وACCOUNT (حساب) و GAME (لعبة) و INSTRUCTOR (مدرس) و COLLEGE(كلية) و LICENCE (رخصة) و غير ذلك.

- **العلاقة الرابطة (Relationship)**

هى العلاقة التى تربط بين الكيانات و تمثل علاقة رابطة فى العالم المصغر الذى تمثله قاعدة البيانات. و تهتم قواعد البيانات بشكل كبير جداً بالعلاقات الرابطة بين الكيانات لأنها تعبر عن الروابط بين البيانات فى الواقع. فى أغلب الأحوال تكون العلاقة الرابطة فعلاً مضارعاً أو فعلاً مبنياً للمجهول.

على سبيل المثال إذا كان لدينا الكيانات STUDENT (طالب) و INSTRUCTOR (مدرس) و COURSE (مقرر دراسى)، يمكن أن يكون لدينا علاقات رابطة كثيرة نذكر منها ثلاثة علاقات رابطة : العلاقة الرابطة STUDY (يَدْرِس) تربط بين الطالب و المقررات الدراسية التى

يدرسها. العلاقة الرابطة الثانية هى TEACH (يُدَرِّس) و تربط بين المدرس و المقررات التى يُدَرِّسها. أما العلاقة الرابطة الثالثة فهى GUIDE (يرشد) و تربط بين المدرس المرشد الأكاديمى و الطلبة الذين يرشدهم.

- **الخاصية أو الحقل (Attribute)**

هى صفة تصف كيان معين و قيمتها هى أحد مكونات سجلات البيانات. أمثلة على الخاصية : Student_No (رقم الطالب) و Student_Name (اسم الطالب) و Address (العنوان).

- **عنصر البيانات (data item)**

هو أقل وحدة بيانات مثل قيمة خاصية. مثال على ذلك قيمة الخاصية Student_No (رقم الطالب) هى عنصر البيانات "٤٢٢٠٤٤٤" وقيمة الخاصية Student_Name (اسم الطالب) هى عنصر البيانات "سيف الله خالد ناصر السيد" و قيمة خاصية Address (العنوان) هى "ص ب ٢٣٧ بريدة، القصيم، السعودية".

- **عنصر بيانات مجمع (data aggregate)**

هو عنصر بيانات يمكن يتكون من عناصر بيانات بسيطة أصغر مثال على ذلك قيمة الخاصية الاسم Name "سيف الله خالد السيد" يمكن تقسيمها إلى قيم لثلاث عناصر بيانات بسيطة كما يلى :

اسم الخاصية المجمعة: الاسم (Name)

أسماء حقول:	First	Mid	Family
مثال بيانات:	سيف الله	خالد	ناصر السيد

كذلك العنوان Address يمكن تقسيمه إلى أربعة حقول هى رقم صندوق البريد (POBox) و الحى (Town) و المدينة (City) و الدولة (Country) كما يلى :

اسم الخاصية المجمعة: العنوان (Address)

أسماء حقول:	POBox	Town	City	Country
مثال :	٢٣٧	بريدة	القصيم	السعودية

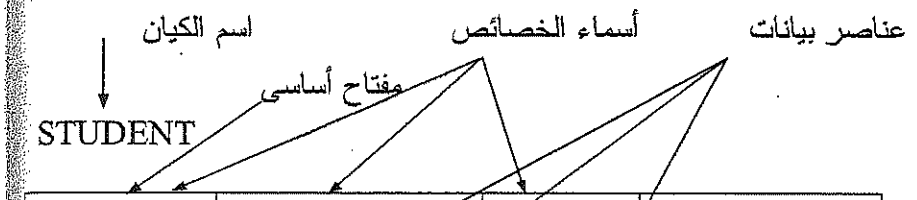
- **سجل (Record)**

هو تجميع لعناصر بيانات. هذا التجميع هو أحد أمثلة أو حالات الكيان مثل بيانات طالب أو مقرر دراسى أو مدرس، فكلٌ منهم يمثل سجلاً تابعاً لكيان معين. كل قيمة من قيم السجل ما هى إلا عنصر بيانات عبارة عن قيمة

لخاصية من خصائص الكيان. جميع السطور الموجودة فى جدول الكيان STUDENT الموجود فى شكل ٢-٢ هى سجلات.

٢-٢-٢ المفتاح (Key)

هو خاصية واحدة (أو عدة خصائص مجتمعة) من خصائص الكيان نستخدمه لاختيار سجل أو أكثر من سجلات ذلك الكيان. يوجد ثلاث أنواع من المفاتيح :



StudentNo	StudentName	Address	.....
30003333	آلاء الله خالد ناصر	القاهرة	سجل ١
30000323	دعاء الله خالد ناصر	القصيم	سجل ٢
33434304	سيف الله خالد ناصر	الاسكندرية	سجل ٣

شكل ٢-٢ : رسم توضيح الكيان و الخصائص و السجلات.

- **المفتاح الأساسى (Primary Key)**

المفتاح الأساسى لكيان هو أحد خصائص هذا الكيان و قيمته تكون وحيدة فى كل سجل و لا تتكرر (unique) فى أى سجل آخر من نفس الكيان. كذلك لا يمكن ترك قيمة خاصية المفتاح الأساسى فراغاً فى أى سجل. نميز هذه الخاصية بوضع خط متصل أسفل الخاصية كما نرى فى الشكل ٢-٣. من الممكن أن يتكون المفتاح الأساسى من خاصيتين أو أكثر و نضع تحتهم جميعاً خطاً واحداً متصلاً.

كما نرى فى الشكل ٢-٣، الخاصية Student_No هى المفتاح الأساسى لكيان STUDENT و الخاصية Dept_No (رقم القسم) هى المفتاح الأساسى للكيان DEPARTMENT (القسم).

- **المفتاح الخارجى (Foreign Key)**

المفتاح الخارجى لكيانٍ ما هو خاصية عادية ضمن خصائص هذا الكيان و موجود كخاصية مفتاح أساسى فى كيان آخر. نميز هذا المفتاح بوضع خط متقطع أسفل اسم الخاصية. الخاصية Dept_No فى الشكل ٢-٣ تُعَد مفتاحاً خارجياً للكيان STUDENT لأنها مفتاحاً أساسياً للجدول DEPARTMENT.

الجدير بالذكر أنه من غير الضرورى أن يكون اسما الحقلين الذين يمثلان رقم القسم فى الكيانين متطابقين، لكن الأساس هو المفهوم الذى يمثلانه حتى إن اختلف الاسمان مثلاً (Dept_No و #D).

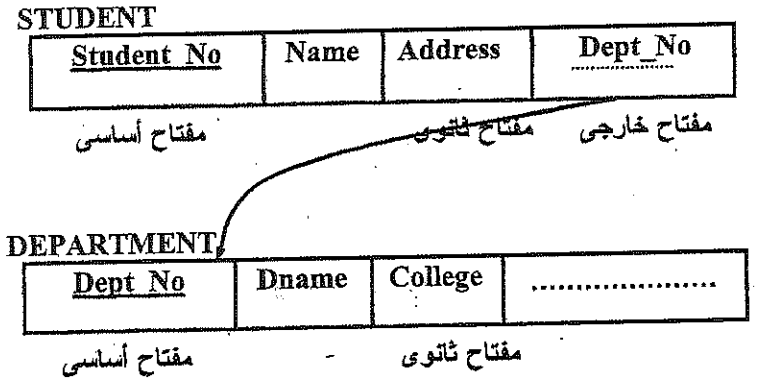
بينما من الضرورى وجود مفتاح أساسى لأى كيان فى قاعدة البيانات و إلا فلن يمكننا منع وجود سجلات متكررة. لكن ليس من الضرورى وجود مفتاح خارجى لكل جدول. الجدول DEPARTMENT لا يوجد به مفتاح خارجى.

من المهم أن تعلم أيضاً أن المفتاح الخارجى يُستخدم للربط بين الكيانات لاستخراج بيانات ذات صلة من كيانين مختلفين. مثال "استخرج اسم القسم و الكلية التى يدرس بها الطالب سيف". لتنفيذ هذا الاستعلام يجب أن نستخرج رقم القسم التابع له الطالب من سجل الطالب بالكيان STUDENT. بعد ذلك نستخدم رقم القسم هذا كمفتاح أساسى للجدول DEPARTMENT و نستخرج بيانات القسم و الكلية من السجل الذى مفتاحه ذلك الرقم.

- **المفتاح الثانوى (Secondary Key)**

المفتاح الثانوى لجدولٍ ما هو أى خاصية يمكن استخدامها لاختيار سجلات معينة من بين السجلات الموجودة فى الكيان. مثال على ذلك الخاصية

Address مفتاح ثانوى فى الكيان STUDENT. كذلك الخاصية College مفتاح ثانوى للكيان DEPARTMENT.



شكل ٢-٣ : رسم يوضح المفاتيح الأساسية و الخارجية و الثانوية و العلاقات بين كلٍ منهما

٢-٢-٣ نماذج البيانات Data Models

نموذج البيانات هو وصف للبيانات عن أشياء أو أنشطة أو أحداث في مكان ما لجعل البيانات منظمة ومفهومة. توجد ثلاثة أنواع (تصنيفات) لنماذج البيانات :

- **نموذج بيانات خارجى (External Data Model)**

نموذج البيانات الخارجى هو نموذج مثل ويصف البيانات المطلوبة فى جهة حكومية أو تجارية أو صناعية أو أهلية أو جهة خدمات. نستطيع أن نطلق على هذا النموذج **مرحلة التحليل** (غير مرتبط بالأجهزة أو البرامج).

- **نموذج بيانات مفاهيمى (Conceptual Data Model)**

نموذج البيانات المفاهيمى هو نموذج يُعرّف ويصف بدقة كافة متطلبات و رؤى المستخدمين فى قاعدة البيانات على أن يتم ذلك من خلال وصف موحد. نستطيع أن نطلق على هذا النموذج **مرحلة التصميم** (غير مرتبط بالأجهزة أو البرامج).

- **نموذج بيانات داخلى (Internal Data Model)**

نموذج البيانات الداخلى هو نموذج يركز على تمثيل وإنشاء قاعدة البيانات المصممة على جهاز الحاسب باستخدام الأدوات سواء كانت برمجية أم أجهزة (Software/Hardware). يمكننا أن نطلق على هذا النموذج **مرحلة التنفيذ** (مرتبط بالأجهزة أو البرامج).

٢-٢-٤ رموز الرسم البيانى Graph Notation

هناك رموز نستخدمها فى مجال قواعد البيانات. هذه الرموز موحدة و تكفى للتعبير عن معظم التعاريف و المفاهيم التى ذكرناها كأساسيات لقواعد البيانات العلاقية. يحتاج الدارس أن يفهم تصميمات قواعد البيانات الموجودة أو يقوم بتصميم قاعدة

البيانات التى يهتم بها عند احترافه لعمل مصمم قواعد بيانات. نضع هذه الرموز القياسية بين يدى الدارس فى الجدول ٢-١:

٢-٣ الروابط فى قواعد البيانات العلاقية

Relational Database Associations

قواعد البيانات العلاقية ترتكز بشكل أساسى على الروابط بين عناصر البيانات أو بين الكيانات أو بين سجلات البيانات. فى الحقيقة أن أحد أهم أسباب نجاح قواعد البيانات العلاقية هى تمثيلها للروابط المختلفة التى تيسر نتيجة لذلك إمكانيات استعلام سهلة و قوية.

جدول ٢-١ : الرموز القياسية المستخدمة فى قواعد البيانات

المفهوم	الرمز
الكيان STUDENT	STUDENT
العلاقة الرابطة STUDY	STUDY
الخاصية Student_Name	Student_Name
المفتاح الأساسى Student_No	Student_No

(ملاحظة: لا يظهر فى الرسم البيانى أى مفتاح إلا المفتاح الأساسى فقط).

٢-٣-١ نسب المشاركة فى العلاقة الرابطة (Cardinality)

أى رابطة بين عناصر البيانات هى بالأساس تربط عنصر بيانات معين إما بعنصر بيانات معين أو بعدة عناصر بيانات. كذلك فإن العلاقة الرابطة بين الكيانات (أو السجلات) هى بالأساس تربط كياناً (أو سجلاً) معيناً بكيان (أو سجل آخر) أو بعدة كيانات (أو سجلات) أخرى.

يوجد مفهوم على قدر كبير من الأهمية يتحكم فى كافة أنواع الروابط المذكورة. هذا المفهوم هو **Cardinality** و هو يعبِّر عن نسبة مشاركة فى العلاقة أو الرابطة سواء بين عنصر و آخر أو كيان و آخر. نعنى بنسبة المشاركة هى عدد العناصر أو السجلات المشاركة فى الرابطة.

تحدد الكاردينالتى مفهومين مرافقين للرابطة :

- **اختيارى (Optional)** أى أنه يمكن أن تكون المشاركة صفراً.
- **إجبارى (Mandatory)** أى أنه لابد أن تكون المشاركة بعنصر واحد على الأقل.

فى الأجزاء التالية سوف نوضح مفهوم الكاردينالتى فى جميع أنواع الروابط.

٢-٣-٢ الروابط بين عناصر البيانات Associations of Data items

هناك ثلاثة أنواع من الروابط بين عناصر البيانات نوردها فيما يلى :

- **رابطة واحدة (One Association)**

هذه الرابطة بين عنصرين تعنى أن كل عنصر بيانات من خاصيةٍ ما يقابلها عنصر بيانات واحد من العنصر الثانى. مثال على ذلك عنصر البيانات (قيمة الخاصية) من Student_No (رقم الطالب) يرتبط بعنصر البيانات من Student_Name (اسم الطالب). أى أن كل رقم طالب يقابله اسم طالب واحد فقط كما نرى فى الشكل ٢-٢(أ).

- **رابطة متعددة (Many Association)**

هذه الرابطة بين عنصرين تعنى أن كل عنصر بيانات من خاصيةٍ ما يقابلها عناصر بيانات متعددة من العنصر الثانى. مثال على ذلك عنصر البيانات (قيمة الخاصية) من Student_No يرتبط بعنصر البيانات من Course# (رقم مقرر دراسى). أى كل رقم طالب يقابله أكثر من رقم مقرر دراسى كما نرى فى الشكل ٢-٢(ب).

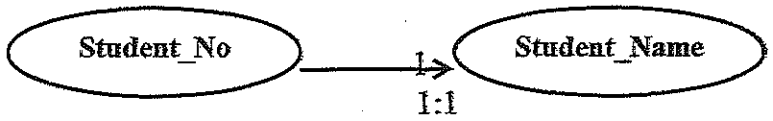
- **رابطة كاردينالتى (Cardinal Association)**

نوع الرابطة هذا يتداخل مع كل الرابطة الواحد و الربطة المتعددة. مثلاً، مع الرابطة الواحدة تحدد هل نسبة المشاركة هى 0:1 أى من صفر إلى واحد مثال على ذلك الرابطة بين عنصر البيانات من Bed# (رقم السرير) و عنصر البيانات من Patient# (رقم المريض) كما فى الشكل ٢-٤(ج)، أو 1:1 كما فى الشكل ٢-٤(أ) بين رقم الطالب و اسمه.

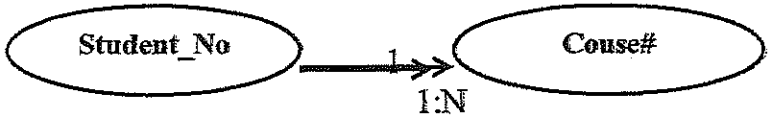
أما مع الرابطة المتعددة فتحدد هل نسبة المشاركة هى 0:N أى من صفر إلى عدد N مثل العلاقة الرابطة بين عنصر البيانات من Room# (رقم الغرفة) و عنصر البيانات من Patient# (رقم المريض) كما فى الشكل ٢-٤(د)، أو 1:N أى من ١ إلى عدد N كما هو الحال بين

عنصر البيانات من #Student و عنصر البيانات من #Course فى الشكل ٢-٤(ب).

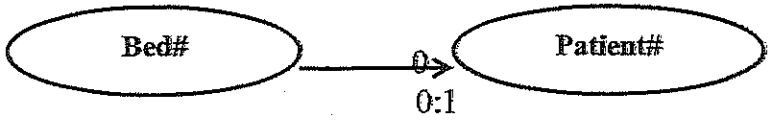
(أ)



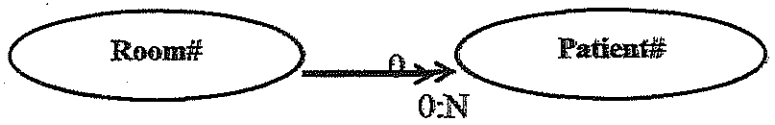
(ب)



(ج)



(د)



شكل ٢-٤ : أنواع الروابط بين عناصر البيانات.

٢-٣-٣ أنواع العلاقات الرابطة بين الكيانات (السجلات)

Types of Associations between Entities (Records)

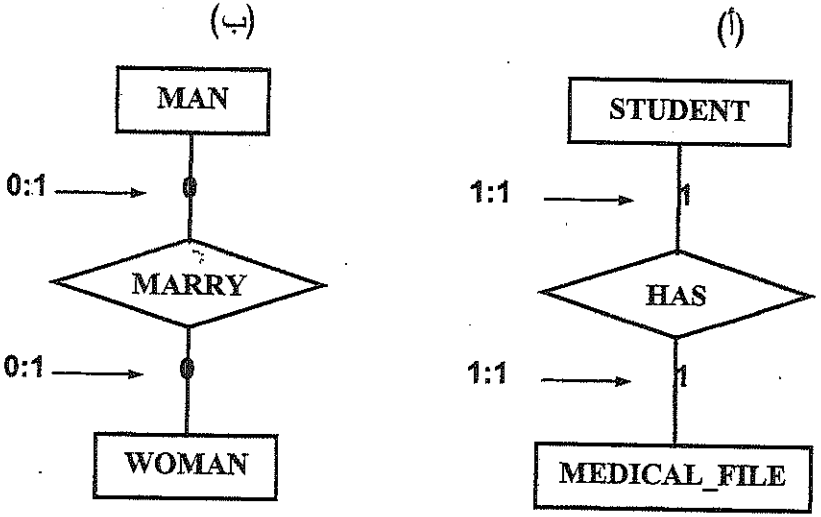
العلاقات الرابطة بين الكيانات (السجلات) هى أهم ما يميز قاعدة البيانات العلاقية. تتوقف قاعدة البيانات التى نصممها و ننفذها بشكل كبير على أنواع العلاقات الربطة. لدينا ثلاث أنواع للعلاقات الرابطة بين الكيانات (السجلات) نوردها فيما يلى :

- علاقة **سجل واحد مع سجل واحد (one-to-one)**

هذه العلاقة تعنى أن أى سجل من الكيان الأول يرتبط مع سجل واحد على الأكثر من الكيان الثانى، و كذلك يرتبط أى سجل من الكيان الثانى مع سجل واحد على الأكثر من الكيان الأول. الشكل ٢-٥ يعرض مثالين من هذا النوع.

المثال الأول نراه فى الشكل ٢-٥(أ) باستخدام العلاقة الرابطة HAS (يمتلك)، من النوع ONE-TO-ONE. هذا المثال يبين أن كل طالب من الكيان STUDENT له ملف طبى واحد فقط (و لابد من وجود ملف طبى واحد على الأقل) على الأكثر من الكيان MEDICAL_FILE (أى أن الكاردينالتى 1:1 جهة الكيان MEDICAL_REPORT). كذلك كل ملف طبى يخص طالب واحد فقط الأكثر(و لابد من وجود طالب على الأقل). (أى أن الكاردينالتى هى 1:1 جهة الكيان STUDENT).

المثال الثانى الذى نراه فى الشكل ٢-٥(ب)عن الزواج طبقاً للشريعة المسيحية باستخدام العلاقة الرابطة MARRY (يتزوج)، من النوع ONE-TO-ONE. هذا المثال يبين أن كل رجل من الكيان MAN يمكن أن يتزوج امرأة واحدة فقط من الكيان WOMAN أو لا تزوج أبداً (لذلك فالكاردينالتى هى 0:1 ناحية الكيان WOMAN). كذلك كل امرأة من كيان WOMAN يمكن أن تتزوج رجلاً واحداً فقط من الكيان MAN أو لا تتزوج أبداً (لذلك فإن الكاردينالتى هى 0:1 ناحية الكيان MAN).

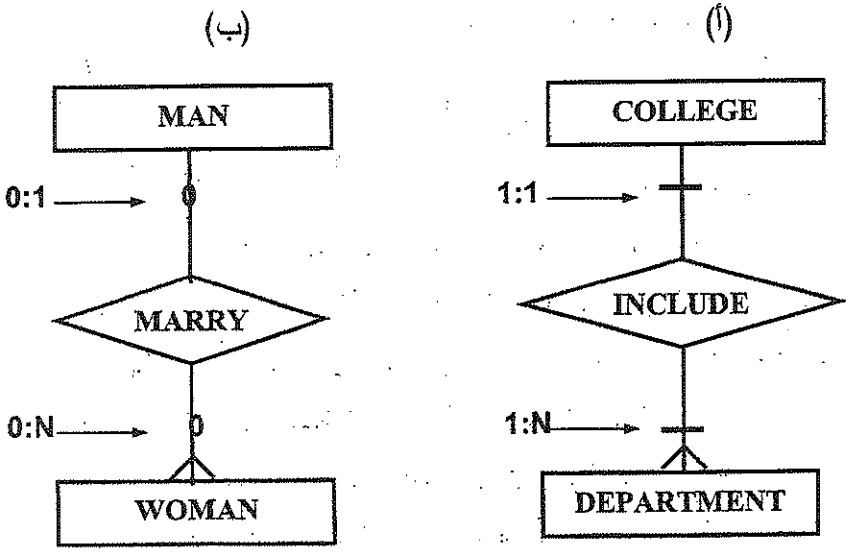


شكل ٢-٥ : العلاقة ONE-TO-ONE.

- **علاقة سجل واحد مع عدة سجلات (one-to-many)**

هذه العلاقة تعنى أن أى سجل من الكيان الأول يمكن أن يرتبط مع عدة سجلات من الكيان الثانى، بينما يرتبط أى سجل من الكيان الثانى مع سجل واحد على الأكثر من الكيان الأول. الشكل ٢-٦ يعرض مثالين من هذا النوع.

المثال الأول نراه فى الشكل ٢-٦(أ) يستخدم العلاقة الرابطة INCLUDE (يتضمّن)، من النوع ONE-TO-MANY. هذا المثال يبين أن سجل كل كلية من الكيان COLLEGE يرتبط بعدة أقسام (سجلات) من الكيان DEPARTMENT و قسم واحد على الأقل (أى أن الكاردينالتى 1:N جهة الكيان DEPARTMENT). كذلك فإن أى سجل لقسم لابد أن يوجد ضمن كلية واحدة على الأكثر و على الأقل لابد أن يوجد فى كلية واحدة فقط (أى أن الكاردينالتى هى 1:1 جهة الكيان COLLEGE).



شكل ٢-٦ : العلاقة ONE-TO-MANY.

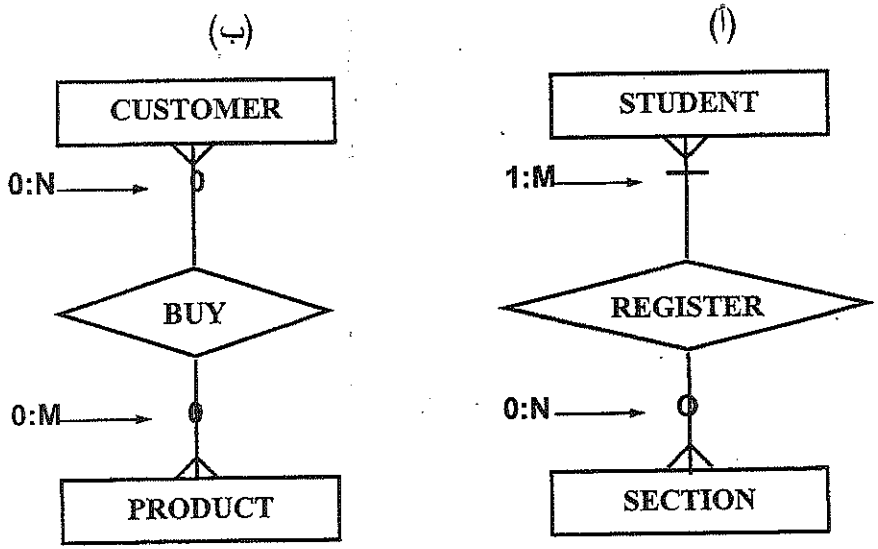
المثال الثانى نراه في الشكل ٢-٦(ب)عن الزواج طبقاً للشريعة الإسلامية. هذا المثال يبين أن سجل أى رجل من الكيان MAN يمكن أن يتزوج أكثر من سجل لامرأة من الكيان WOMAN أو لا يتزوج أبداً (لذلك فالكارديناليتى هى 0:N ناحية الكيان WOMAN). بينما أى سجل لامرأة من كيان WOMAN يمكن أن تتزوج رجلاً واحداً فقط من الكيان MAN أو لا تتزوج أبداً (لذلك فإن الكارديناليتى هى 0:1 ناحية الكيان MAN).

- **علاقة عدة سجلات مع عدة سجلات (many-to-many)**

هذه العلاقة تعنى أن أى سجل من الكيان الأول يرتبط مع عدة سجلات الكيان الثانى، و كذلك يرتبط أى سجل من الكيان الثانى مع عدة سجلات من الكيان الأول. الشكل ٢-٧ يعرض مثالين من هذا النوع.

المثال الأول نراه فى الشكل ٢-٧(أ) يستخدم العلاقة REGESTER (يُسجِّل) من النوع MANY-TO-MANY. هذا المثال يبين أن كل طالب من الكيان STUDENT يمكنه أن يُسجِّل فى عدة شعب دراسية أو لا يسجل أى شعبة دراسية من الكيان SECTION (أى أن الكارديناليتى 0:N جهة الكيان SECTION). كذلك فإن سجل أى شعبة دراسية فى الكيان SECTION يمكن أن يسجل بها عدة طلبة و لابد من وجود طالب على الأقل. (أى أن الكارديناليتى هى 1:M جهة الكيان STUDENT).

المثال الثانى الذى نراه فى الشكل ٢-٧(ب)يستخدم العلاقة BUY (يشترى) من النوع MANY-TO-MANY. هذا المثال يبين أن سجل أى عميل (زبون) من الكيان CUSTOMER يمكن أن يشترى عدة منتجات و ربما لا يشترى أى منتج من الكيان PRODUCT (لذلك فالكارديناليتى هى 0:M ناحية الكيان PRODUCT). كذلك أى سجل منتج من الكيان PRODUCT يمكن أن يشتريه عدة عملاء و قد لا يشتريه أى عميل من الكيان CUSTOMER (لذلك فإن الكارديناليتى هى 0:N ناحية الكيان CUSTOMER).



شكل ٢-٧ : العلاقة MANY-TO-MANY.

٢-٣-٤ درجة العلاقة الرابطة بين الكيانات (السجلات)

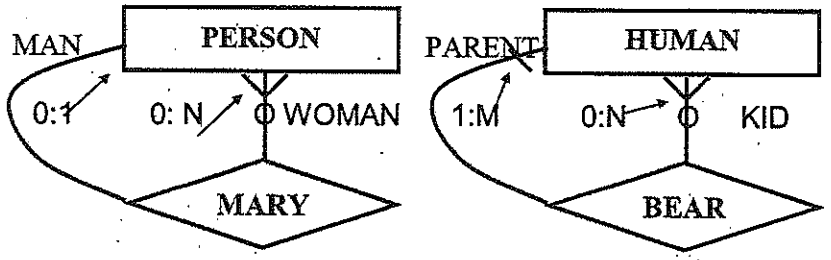
Degree of Association between Entities (Records)

تحدد درجة العلاقة الرابطة عدد الكيانات المشاركة فى العلاقة الرابطة أو بمعنى آخر عدد الكيانات التى تربط بينها العلاقة الرابطة. الجدير بالذكر أنه أياً كانت درجة العلاقة الرابطة فلابد أن تكون ذات نوع من الأنواع الثلاثة التى ذكرناه سابقاً. يوجد لدينا ثلاث درجات للعلاقة الرابطة نوردها فيما يلى :

- **علاقة أحادية (unary relationship)**

تكون درجة العلاقة الرابطة أحادية إذا كانت علاقة رابطة بين الكيان ونفسه. أى تربط بين سجلات من نفس الكيان. فى هذه الحالة يمكن أن يلعب أى سجل دورين بحيث يبدو السجل كأنه ينتمى إلى كيانين من نفس النوع. الشكل ٢-٨ يعرض مثالين لعلاقة درجتها أحادية.

(ب) (أ)



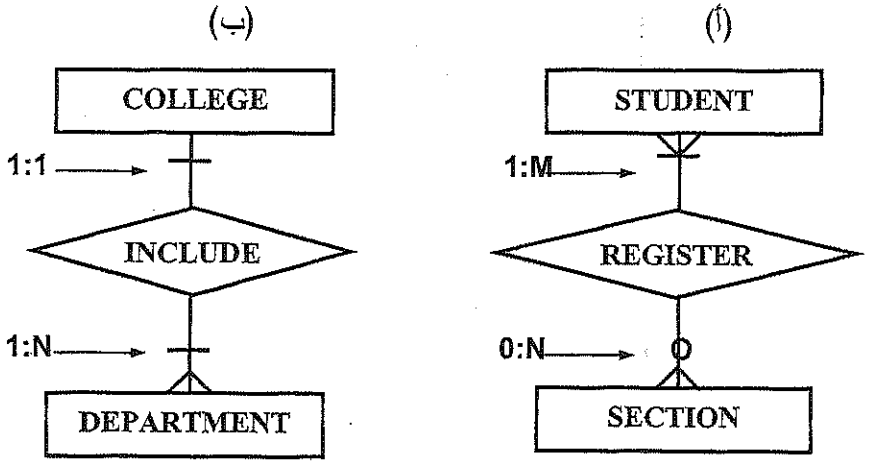
شكل ٢-٨ : علاقات رابطة أحادية.

المثال الأول الموجود فى الشكل ٢-٨(أ) يبين العلاقة الرابطة الأحادية BEAR (يُنْجِب) التى تربط بين سجلات من نفس نوع الكيان HUMAN (إنسان). كما نرى فى الرسم فإن سجلاً يلعب دور PARENT (الوالد أو الوالدة) يمكن يرتبط (ينجب) عدة سجلات تلعب دور KID (الابن أو الابنة) من نفس نوع الكيان HUMAN أو لا ينجب إطلاقاً (لذلك فإن الكاردينالتى ناحية الابن تكون 0:N). أما السجل الذى يلعب دور الابن فبالضرورة ينجبه والد و والدة. (لذلك فإن الكاردينالتى من ناحية الأب أو الأم تكون 1:M). هنا M تساوى اثنين على الأكثر. أما 1 فتعنى أنه ممكن أن ينجب إنسان واحد إنساناً أخر (مثل سيدنا آدم و سيدتنا حواء عليهما السلام و كذلك سيدتنا مريم و سيدنا عيسى عليهما السلام).

المثال الثانى الموجود فى الشكل ٢-٨(ب) يبين العلاقة الرابطة الأحادية MARRY (يتزوج) التى تربط بين سجلات من نفس نوع الكيان PERSON (شخص). كما نرى فى الرسم فإن سجلاً يلعب دور الرجل يمكن يرتبط (يتزوج) من عدة سجلات تلعب دور المرأة من نفس نوع الكيان PERSON أو لا يتزوج أبداً (لذلك فإن الكاردينالتى ناحية سجل المرأة تكون 0:N). أما السجل الذى يلعب دور المرأة فيمكنه أن يربط (تتزوج) من سجل يلعب دور الرجل أو لا تتزوج أبداً. (لذلك فإن الكاردينالتى من ناحية سجل الرجل تكون 0:1).

- **علاقة ثنائية (binary relationship)**

تكون درجة العلاقة الرابطة ثنائية إذا كانت علاقة رابطة بين كيانين. أى تربط بين سجلات من كيانين مختلفين. الشكل ٢-٩ يعرض مثالين لعلاقة درجتها ثنائية.



شكل ٢-٩ : علاقات رابطة ثنائية.

هذان المثالان من ضمن الأمثلة التى درسناها فى الشكلين هما نفس المثالين الذين استخدمناهما فى الشكلين ٢-٧(أ) و ٢-٦(أ) و ينطبق عليهم ما سبق أن ذكرناه مع هذين المثالين، مع التركيز على أنهما مثالين لعلاقات رابطة درجتها ثنائية.

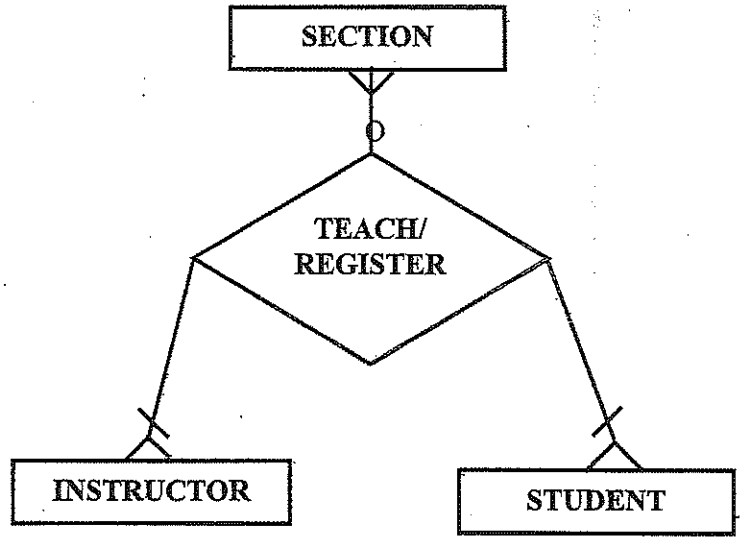
- **علاقة ثلاثية (ternary relationship)**

تكون درجة العلاقة الرابطة ثلاثية إذا كانت علاقة رابطة بين ثلاث كائنات. أى تربط بين سجلات من ثلاث كيانات مختلفة. فى اغلب قواعد البيانات لا نستخدم العلاقة الثلاثية و نحولها إلى علاقتين ثنائيتين أو أكثر. الشكلين ٢-١٠ و ٢-١١ يعرضان مثالين للعلاقة الثلاثية.

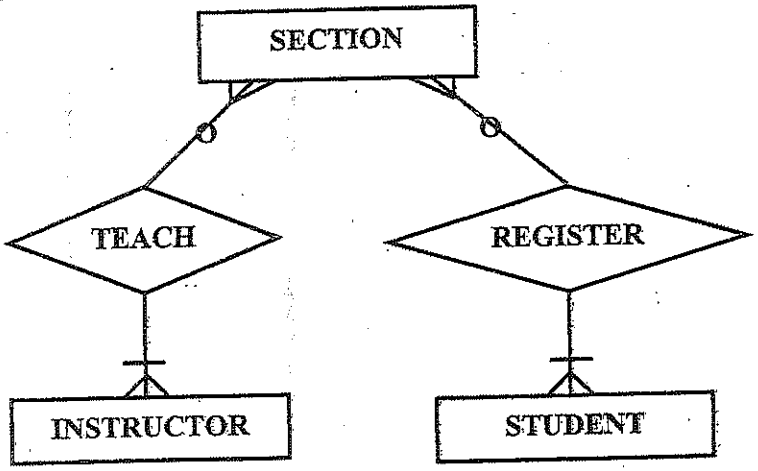
الشكل ٢-١٠(أ) يعرض مثال لعلاقة ثلاثية و تحويل كل منهما إلى علاقات ثنائية. فنجد أن العلاقة الثلاثية TEACH/STUDY و التى تربط بين سجلات من الكيانات الثلاث STUDENT و INSTRUCTOR و SECTION(شعبة دراسية). و يمكن أن نقرأ علاقة من هذا النوع بهذا الشكل "الطالب على يَدْرِس مقرر قواعد البيانات الذى يُدَرِّسه المدرس محمد". الشكل ٢-١٠(ب) يوضح كيفية تحويل العلاقة الثلاثية إلى علاقتين ثنائيتين هما TEACH و STUDEY ليسهل تنفيذهما.

الشكل ٢-١١(أ) يعرض مثال لعلاقة ثلاثية و تحويل كل منهما إلى علاقات ثنائية. فنجد أن العلاقة الثلاثية PRODUCE/BUY (ينتج/يشترى) و التى تربط بين سجلات من الكيانات الثلاث PRODUCT (سلعة) و CUSTOMER و FACTORY (مصنع). و يمكن أن نقرأ علاقة من هذا النوع بهذا الشكل "العميل سعد يشترى السلعة

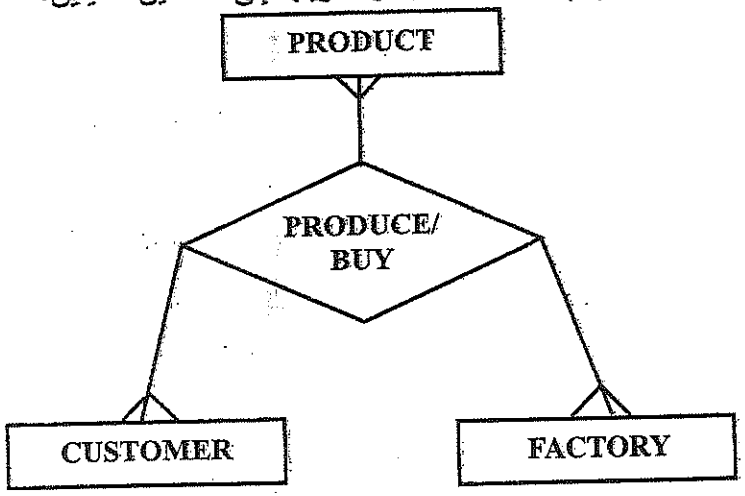
سكر التى ينتجها المصنع العربى". الشكل ٢-١١(ب) يوضح كيفية تحويل العلاقة الثلاثية إلى علاقتين ثنائيتين هما PRODUCE و BUY ليسهل تنفيذهما.



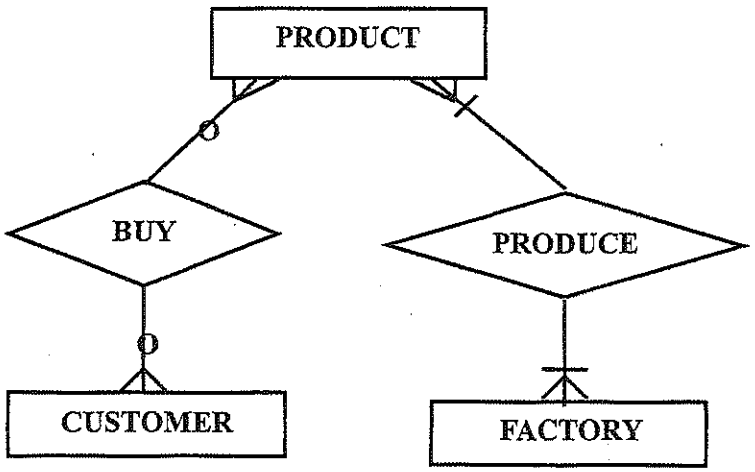
شكل ٢- (أ)١٠ : علاقة ثلاثية و تحويلها إلى علاقتين ثنائيتين.



شكل ٢-١٠(ب) : علاقة ثلاثية و تحويلها إلى علاقتين ثنائيتين.



شكل ٢-١١(أ) : علاقة ثلاثية و تحويلها إلى علاقتين ثنائيتين.



شكل ٢-١١ : علاقة ثلاثية و تحويلها إلى علاقتين ثنائيتين.

٢-٤ النموذج العلاقى Realtional Model

لبناء قواعد بيانات يُفضل مبرمجو و مصممو قواعد البيانات استخدام **نموذج البيانات العلاقى (Relational Data Model)**. هذا النموذج مبنى على العلاقات بين الكيانات و البيانات. كما نجد أن أغلب البرامج و النظم المستخدمة فى مجال قواعد البيانات قد صُمِّمَت لبناء هذا النوع.

٢-٤-١ مميزات النموذج العلاقى

Advantages of Relational Model

يتميز النموذج العلاقى عن غيره من نظم قواعد البيانات للأسباب التالية :

- له الأدوات و الخصائص التى تميزه عن غيره و الخاصة به.
- يمثل منطقياً كافة الكيانات و العلاقات و خصائصهما.
- يُعَد تطويراً و امتداداً لشكل ملف البيانات التقليدى.
- يستخدم المفاتيح الأساسية و الخارجية للربط بين الكيانات.
- اعتماداً عليه بُنِيَت خصائص قواعد البيانات التى تجعلها لا تتأثر بمشاكل الصيانة.
- على أساس استخدامه بُنِيَت عليه كافة اللغات.
- يمكن تطبيق كافة العمليات الحسابية و المنطقية على مكوناته.

٢-٤-٢ نموذج الكيان و العلاقة الرابطة

Entity_Relationship Model

أحد أشهر و أهم طرق تمثيل و تصميم قواعد البيانات هو نموذج الكيان و العلاقة الرابطة **(Entity- Relationship Data Model (ERD))**. نموذج ERD هو النموذج الذى يتم استخدامه لإنشاء قواعد البيانات على الحاسب الآلى و له قواعد و أشكال محددة تصف كافة الكيانات الموجودة فى تطبيق معين و العلاقات الرابطة بين تلك الكيانات و خصائصهما و كذلك القيود المفروضة على كلٍ منهما. أى أنه يمثل تصميم قاعدة البيانات.

الأشكال ٢-٥ و٢-٦ و ٢-٧ و ٢-٨ و ٢-٩ و ٢-١٠ و ٢-١١ التى درسناها خلال هذا الفصل تعرض أمثلة بسيطة لنماذج ERD و سنقصها فقط

تحديد خصائص كل كيان أو علاقة رابطة. هذه الأمثلة بسيطة و ليست إلا أمثلة تدريبية أو أجزاء من نماذج ERD أكبر. فى الفصل التالى ندرس عملية التصميم باستخدام نموذج ERD بشكل أعمق و تفصيل أكبر.

٢-٥ أسئلة

١. أذكر درجات العلاقة الرابطة بين السجلات مع مثالين لكل نوع.

٢. أذكر أنواع العلاقة الرابطة بين السجلات مع مثالين لكل نوع.

٣. عرّف نموذج البيانات Data Model، ثم اشرح بإيجاز أنواع نماذج البيانات.

٤. عرّف النموذج العلاقى وأذكر مميزاته.

٥. أذكر مع الشرح بإيجاز مفهوم المفتاح و أنواعه.

٦. ما هو نموذج ERD؟

٧. عرّف كلاً من : data aggregate و data item و entity و record و metadata و relationship و attribute .

٨. عرف Cardinality مع ذكر أمثلة.

الفصل الثالث

نموذج الكيان و العلاقة الرابطة

Entity Relationship Diagram (ERD)

بإمكان مصممى قواعد البيانات العلاقية استخدام طريقتان لتصميم قاعدة البيانات. الطريقة الأولى هى استخدام **الرسم البيانى للكيان و العلاقة الرابطة (Entity Relationship Diagram (ERD))**. فى هذا الفصل و الفصل التالى نركز على استخدام نموذج ERD لتصميم قواعد البيانات العلاقية. أما الطريقة الثانية فهى طريقة **تطبيع قواعد البيانات (Database Normalization)**، و سوف نركّز عليها فى الفصل الخامس.

أصعب شىء يواجه من يتعلم تصميم قواعد البيانات هو كيفية التفكير فى التصميم و إنجازه بشكل جيد يلائم متطلبات التطبيق. و لقد درسنا المفاهيم فى الفصل السابق و الآن جاء دور التطبيق بالتدرج من تطبيقات بسيطة فى هذا الفصل إلى تطبيقات بالمستندات إلى تطبيقات كبيرة فى الفصل التالى. يتم ذلك من خلال أمثلة كثيرة و متعددة تغطى الكثير من التطبيقات حتى نصل بالدارسة و الدارس إلى مستوى متقدم يجعله فى أول طريق احتراف لتصميم قواعد البيانات.

٣-١ أسلوب تصميم نموذج الكيان و العلاقة الرابطة
Entity-Relationship Design Technique

يتكون تصميم ERD من مجموعة من **الكيانات (Entity)** تربط بعضها ببعض **علاقات رابطة (Relationship)**. لكلٍ من الكيانات و العلاقات الرابطة خصائص تخص كلاً منهم.

فى الغالب تكون السجلات التى تتبع الكيانات عبارة بيانات شبه ثابتة و نادراً ما تحتاج إلى التعديل و يُطلَق عليها **بيانات استاتيكية (static data)**. علينا أن نحدد هنا خصائص كل كيان و الخاصية التى تميز كل سجل يتبع الكيان و لا تتكرر لتكون هى **المفتاح الأساسى (primary key)**.

أما السجلات التى تصف العلاقات الرابطة فهى عبارة عن بيانات تتجدد و تتغير و تُضاف و تُحذَف بشكل متواصل و يُطلَق عليها **بيانات ديناميكية (dynamic data)**. يجب أن نحدد لكل علاقة رابطة الخصائص التى تساعد على وصف العلاقة الرابطة بين كل كيانين و كذلك نوع العلاقة **(one-to-one** أو **one-to-many** أو **many-to-many)**.

بعد تحديد الكيانات و خصائصها و العلاقات الرابطة بين الكيانات و خصائصها نبدأ فى الرسم البيانى لنموذج الكيان و العلاقة الرابطة الذى يضم كل ما وصلنا إليه من كيانات و علاقات رابطة و خصائص كلٍ منهما. القاعدة العامة أن

نموذج ERD يتم رسمه أو قراءته من اليسار إلى اليمين و من أعلى إلى أسفل. و لكن يمكن قراءته و رسمه من أى اتجاه دون أن يؤثر ذلك على المعانى التى يمثّلها. و نضع خطاً تحت كل مفتاح أساسى.

آخر خطوة هى تحديد نوع كل علاقة رابطة ونسبة المشاركة (0 أو 1).

فى بداية دراستنا للتصميم نقدم أمثلة هى فى الأساس تطبيقات مصغَّرة أو بالأحرى هى أجزاء من قاعدة بيانات كبيرة لنهيئ الدارسة و الدارس لفهم تصميم قواعد البيانات فى أول خطوة فى هذا المجال.

٣-٢ تطبيق قاعدة بيانات الكلية المُصغَّر
Simple College System

صمم ERD لقاعدة بيانات الكلية. نهتم هنا بتخزين بيانات **الطلبة** من رقم جامعى واسم و عنوان و تخصص و المقررات التى **يسجلها** الطالب و الفصل الدراسى و العام الدراسى الذى سجل فيه المقرر و نتيجة اختباره فى كل مقرر. كذلك نهتم بتسجيل بيانات **أعضاء هيئة التدريس** من رقم و اسم و هاتف و القسم الذى يعمل به و المقررات التى **يُدَرِّسونها** و الفصل الدراسى و العام الدراسى اللذان تم تدريس كل مقرر دراسى. كل ذلك مع تسجيل رمز و اسم كل **مقرر دراسى** و عدد الساعات المعتمدة.

التصميم : (فى الشكل ٣-١)

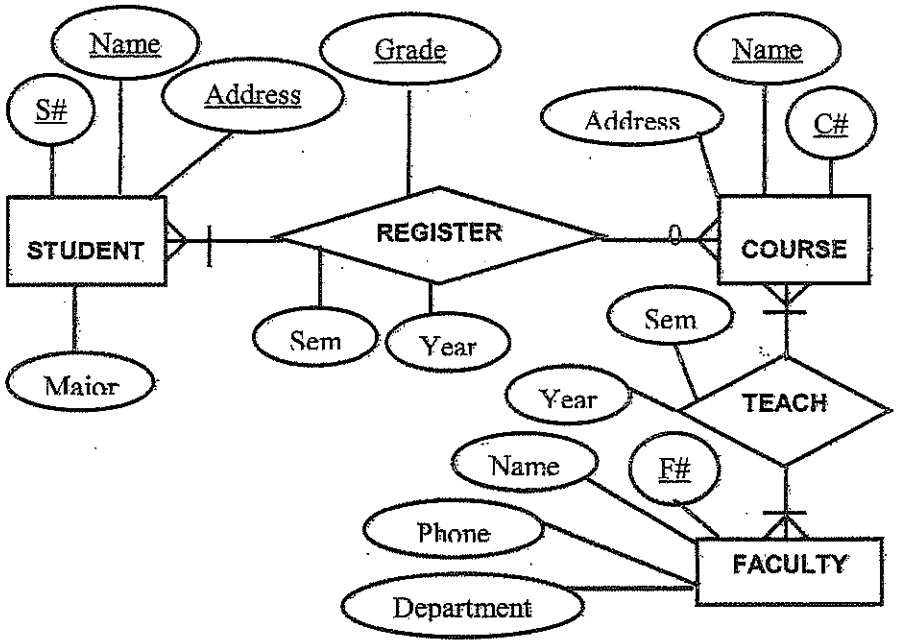
أولاً الكيانات : نذكر الدارس أن الكيان هو وحدة معلومات لها خصائص تصفها و تخصها و أنها تكون أسماء. و عندما نختار اسماً لكيان علينا أن نفكر فى السجلات التى يمكن أن تنتمى إليه هو فقط. بالبحث فى الوصف السابق نجد أن الكيانات (قد جعلنا أسماء الكيانات الثالثة بخط ثقيل لنميزها لك فى هذا المثال) و خصائصها التى تخصها دون غيرها تكون كما يلى :

- الطالب (STUDENT) وله الخصائص: الرقم الجامعى (S#) و اسم الطالب (Name) و العنوان (Address) و التخصص (Major). أما المقررات التى سجّلها (درسها) أو متى درسها أو نتيجته فيها فهى متغير و ليس مكانها هنا ولها أماكنها التى سنعرفها حالاً.
- عضو هيئة التدريس (FACULTY) وله الخصائص : رقم عضو هيئة التدريس (F#) و اسمه (Name) و هاتفه (phone) و القسم الذى يعمل به (Department). أما المقررات التى يدرسها و متى يدرسها فهى متغيرة و ليس مكانها هنا.
- المقرر الدراسى (COURSE) و له الخصائص : رمز المقرر (C#) و اسم المقرر (Title) و عدد الساعات المعتمدة (Credit). أما من يُدَرِّس و من يَدْرُس و متى يحدث ذلك فهو فى العلاقات الرابطة.

ثانياً العلاقات الرابطة : نذكر الدارسة و الدارس أن العلاقات الرابطة تكون فى الغالب عبارة عن فعل يمثل العلاقة بين كيان و نفسه (وهذا نادر) أو كيانين (و هذا هو الغالب) و لا نفكر فى ربط ثلاث كيانات معاً. لأنه لدينا ثلاث كيانات فهناك على الأكثر ثلاث علاقات رابطة و ربما أقل.

علينا أن نبحث بين كلمات وصف التطبيق عن علاقة كل كيان بالكيانين الآخرين و لا نتخيّل أو نفترض علاقة من أنفسنا أو حتى من الواقع الذى يمثله التطبيق، بل نلتزم بما هو مطلوب. بالبحث نجد علاقتين فقط (و قد جعلنا العلاقتين الرابطتين بخط ثقيل و تحت الاسم خط) و هما :

- يُسجِّل (REGISTER) و هى علاقة تربط الطالب بالمقررات التى يسجّلها للدراسة و لها الخصائص : الفصل الدراسى (Sem) و العام الدراسى (Year) والنتيجة أو التقدير (Grade). نعلم هنا أن كل طالب يمكنه تسجيل عدة مقررات (0:N ناحية المقرر). أما المقرر الدراسى فيمكن أن يسجله عدة طلبة و على الأقل طالب واحد أى 1:M ناحية الطالب. أى أن هذه العلاقة نوعها many-to-many.



شكل ٣-١ : نموذج ERD لتطبيق الكلية.

- يُدَرّس (TEACH) و هى علاقة رابطة بين عضو هيئة التدريس و المقرر الدراسى وخصائصها: الفصل الدراسى (Sem) و العام الدراسى (Year). كل عضو هيئة تدريس يمكن أن يُدَرّس عدة مقررات و على الأقل مقرر واحد (1:M ناحية المقرر). كذلك المقرر الدراسى يمكن أن يدرسه عدة

أعضاء هيئة تدرس و على الأقل عضو واحد (1:N ناحية عضو هيئة التدريس). أى أن هذه العلاقة نوعها many-to-many.

٣-٣ تطبيق قاعدة بيانات المستشفى المُصغَّر

Simple Hospital System

ارسم ERD اللازم لتمثيل بيانات المرضى فى أحد المستشفيات و الأطباء المعالجون مشتملا على رقم المريض واسمه ورقم الغرفة المقيم بها ورقم تليفون الغرفة عدد الأسرَّة بها و كمية واسم و رقم الدواء المنصرف له و كذلك رقم الطبيب واسمه و تليفونه وتخصصه و تاريخ دخول المريض للتنويم.

<u>التصميم</u> : (فى الشكل ٣-٢)

<u>أولاً الكيانات</u> : بنفس الطريق نستخرج الكيانات وهى أربعة :

- المريض (PATIENT) و له الخصائص : رقم المريض (P#) و اسم المريض (Pname).
- الطبيب (DOCTOR) و له الخصائص : رقم الطبيب (D#) و اسم الطبيب (Dname) و تليفونه (Phone) و تخصصه (Prof).
- الدواء (MEDICEN) و له الخصائص :رقم الدواء (M#) و اسم الدواء (Mname).

- الغرفة (ROOM) و له الخصائص : رقم الغرفة (R#) و رقم تليفون الغرفة (Rphone) و عدد الأسرة (Beds).

ثانياً العلاقات الرابطة :

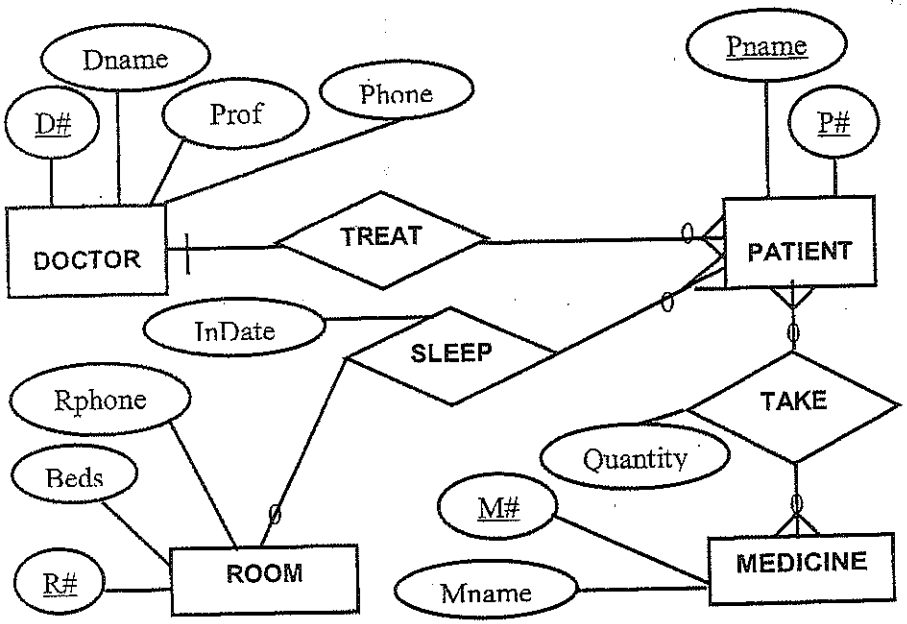
- يُعالِج (TREAT) و هى علاقة رابطة بين الطبيب و المريض الذى يعالجه. فى العادة يكون كل مريض تابع لطبيب واحد و واحد فقط (حتى و إن فحصه العديد من الأطباء)، لذلك تكون الكاردينالتى (1:1) ناحية الطبيب. أما الطبيب فيتبعه عدة مرضى و يمكن ألا يتبعه أى مريض (مثل طبيب الأشعة و التخدير). لذلك تكون الكاردينالتى (0:N) ناحية المريض. هذه العلاقة one-to-many.

- ينام فى (SLEEP) و هى علاقة رابطة بين المريض و الغرفة. و حيث أنه يمكن لكل غرفة أن يقيم بها عدة مرضى أو لا يقيم بها أى مريض، لذلك فالكاردينالتى تكون (0:N) ناحية المريض. أما كل مريض فيمكن أن ينام فى غرفة معينة واحدة أو لا يتم تنويمه (مريض عيادة خارجية)، لذلك فالكاردينالتى (0:1) ناحية الغرفة. هذه العلاقة one-to-many. هذه العلاقة الربطة لها خاصية InDate (تاريخ التنويم).

- يصرِف (Take) و هى علاقة رابطة بين المريض و الدواء. كل مريض يمكن أن يصرف عدة أدوية أو لا يصرف دواءً على الإطلاق، لذلك فالكاردينالتى تكون (0:N) ناحية الدواء. كذلك كل دواء يمكن أن يصرفه عدة مرضى أو لا يصرفه أى مريض، لذلك فالكاردينالتى تكون (0:M)

ناحية المريض. هذه العلاقة من النوع many-to-many. هذه العلاقة لها خاصية واحدة هى كمية الدواء المنصرفة لكل مريض (Quantity).

لا توجد هنا علاقة رابطة بين الطبيب و الدواء أى لا نسجل بيانات تحدد من أمر بصرف الأدوية (إلا إذا طُلِبَ ذلك). كما لا نسجل بيانات أو سجلات علاقة رابطة فى المثال السابق بين عضو هيئة التدريس و الطالب إلا إذا كان هناك إرشاد أكاديمى و طُلِبَ أن تتضمنه قاعدة البيانات.



شكل ٣-٢ : نموذج ERD لتطبيق المستشفى.

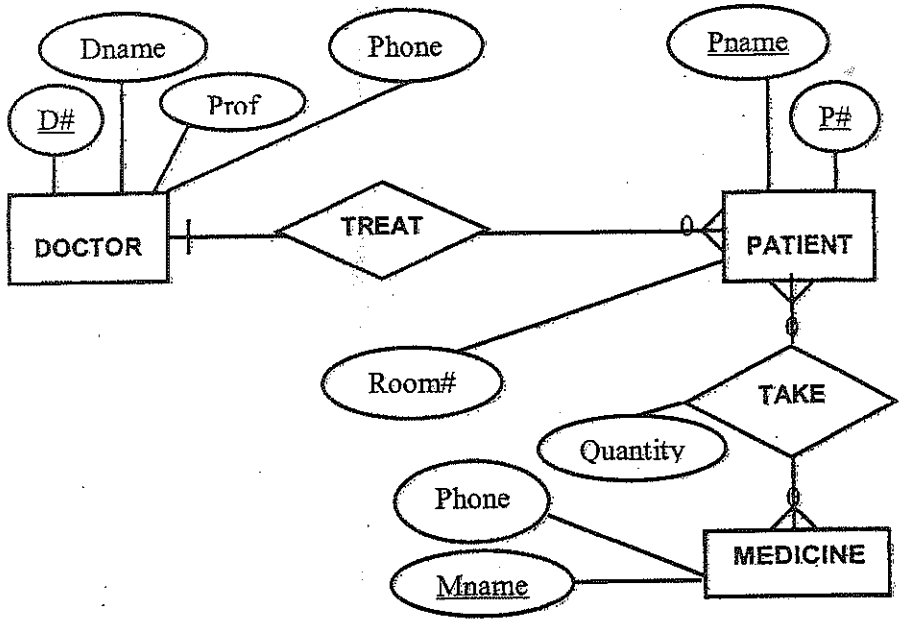
٣-٤ تطبيق قاعدة بيانات المستشفى مع تعديل المتطلبات

Modified Hospital Database System

أرسم ERD اللازم لتمثيل بيانات المرضى فى أحد المستشفيات و الأطباء المعالجون مشتملا على رقم المريض واسمه ورقم الغرفة المقيم بها و كمية واسم و نوع الدواء المنصرف له و كذلك رقم الطبيب واسمه و تليفونه وتخصصه.

التصميم : (فى الشكل ٣-٣)

لاحظ أن نفس أن وصف هذا التطبيق هو نفس وصف التطبيق السابق مع فارق واحد أن وصف التطبيق الحالى لم يذكر بالنسبة لغرفة التنويم إلا رقمها فقط و تجاهل تليفون الغرفة و عدد الأسرة بها و تاريخ التنويم بالغرفة. لهذا السبب فإن التصميم يكون هو نفس التصميم الموجود فى الشكل ٣-٢ و لكن لم تعد هناك حاجة لكيان الغرفة (ROOM) و العلاقة الرابطة (SLEEP) بين المريض و الغرفة. و تزيد خاصية رقم الغرفة (Room#) لكيان المريض. بناءً على ذلك يصبح التصميم الجديد كما نراه فى الشكل ٣-٣.



شكل ٣-٣ : نموذج ERD لتطبيق المستشفى بعد التعديل.

تنبيه هام :

يمكن أن نطبق نفس فكرة التعديل السابق على أى تصميم بحذف بعض الخصائص مما يسبب حذف كيانات و علاقات رابطة. كذلك يمكن إجراء الفكرة بالعكس أى بإضافة خصائص جديدة ينشأ عنها كيانات و علاقات جديدة.

٣-٥ تطبيق الاستعارة المكتبية Library System

ارسم ERD اللازم لقاعدة بيانات المكتبة فيما يخص بيانات المستعير من رقم واسم وكذلك وظيفته و جهة عمله و تاريخ ميلاده و نوع الاستعارة و مدة الاستعارة و تاريخها و كذلك بيانات الكتاب كرقم و عنوان و دار النشر و سنة النشر و عدد صفحات الكتاب و المؤلفين و عناوينهم الإلكترونية.

التصميم : (فى الشكل ٣-٤)

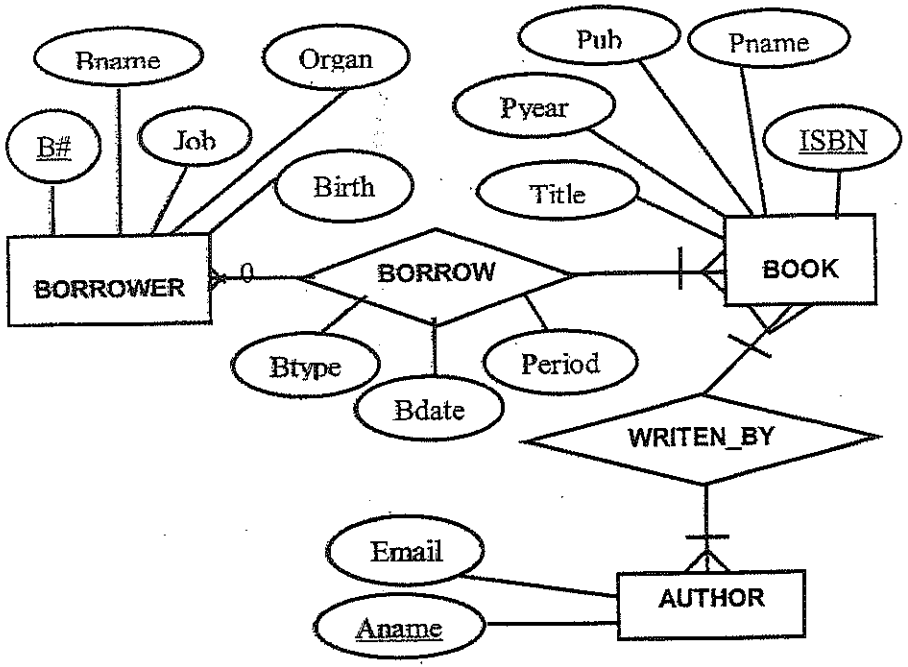
أولاً الكيانات :

- المستعير (BORROWER) و له الخصائص : رقم (B#) و اسم (Bname) و عمله (Job) و جهة عمله (Organ) و تاريخ ميلاده (Birth).
- الكتاب (BOOK) و له الخصائص : رقم (ISBN) و عنوان (Title) و دار النشر (Pub) وسنة النشر (Pyear) وعدد الصفحات (Pages).
- المؤلف (Author) و له الخصائص : المؤلف (Aname) و عنوانه الإلكترونى (Email).

ثانياً العلاقات الرابطة :

- يستعير (BORROW) و هى علاقة رابطة بين المستعير و الكتاب و لها الخصائص : نوع الاستعارة (Btype) و مدة الاستعارة (Period) و

تاريخ الاستعارة (Bdate). يمكن لأى مستعير أن يستعير عدة كتب و على الأقل كتاب واحد ليصبح مستعيراً، لذلك فالكارديناليتى تكون (1:N) ناحية الكتاب. أما الكتاب فيمكن أن يستعير عدة مستعيرين و قد لا يستعيره أحد، لذلك تكون الكارديناليتى (0:M) ناحية المستعير. هذه العلاقة نوعها many-to-many.



شكل ٣-٤ : نموذج ERD لتطبيق الاستعارة المكتبية.

- كُتِبَ بواسطة (WRITEN_BY) و هى علاقة رابطة بين المؤلف و الكتاب. يمكن لأى مؤلف أن يكتب عدة كتب و بالطبع على الأقل كتاب واحد ليصبح مؤلفاً، لذلك تصبح الكارديناليتى (1:N) ناحية الكتاب. بالنسبة لأى كتاب، يمكن أن يشترك فى تأليف عدة مؤلفين و على الأقل مؤلف واحد، لذلك فالكارديناليتى تصبح (1:M) ناحية المؤلف.

٣-٦ تطبيق أوامر الشراء للعملاء

Customer Orders System

ارسم ERD اللازم لقاعدة بيانات لتخزين البيانات الخاصة بأمر الشراء الذي يرسله العميل و يهتم في هذه العملية برقم العميل و اسمه و عنوانه و أرقام أوامر الشراء و تاريخ صدورها و كذلك الكمية المطلوبة من العميل في أمر الشراء من كل صنف و رقم كل صنف و اسمه و سعره و الجهة المصنعة للصنف.

التصميم : (فى الشكل ٣-٥)

أولاً الكيانات :

- أمر الشراء (ORDER) وله الخصائص : رقم أمر الشراء (#Order) و تاريخه (Odate).

- العميل (CUSTOMER) و له الخصائص : رقم العميل (#Cust) و اسمه (Cname) و عنوانه (Address).

- الصنف (ITEM) و له الخصائص : رقم الصنف (#Item) و اسم الصنف (Description) و سعره (Price) و الجهة المصنعة (Factory).

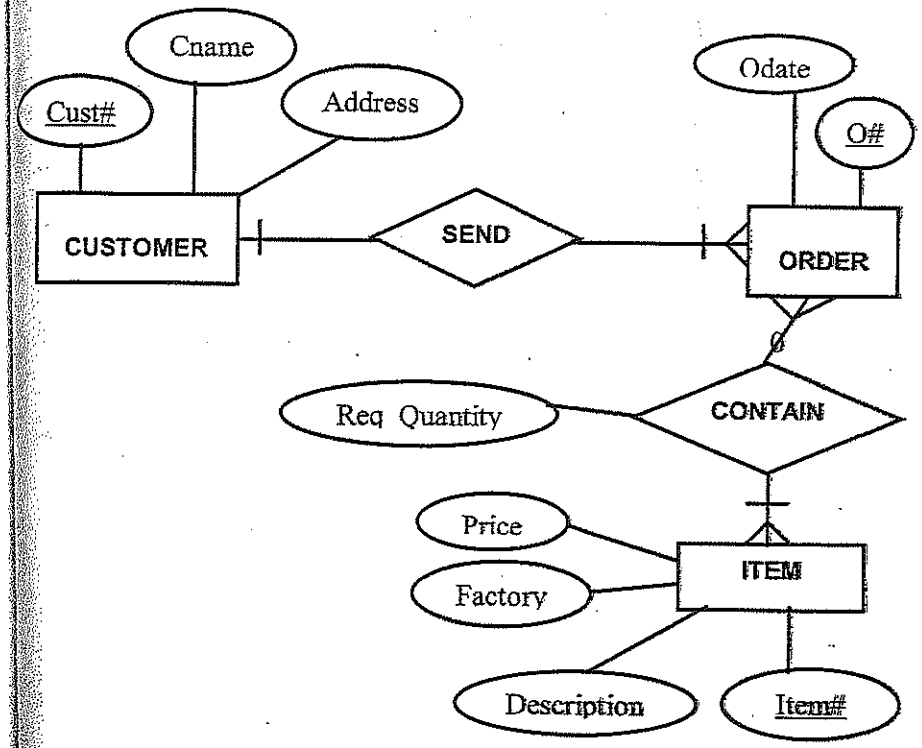
هنا الكمية التى يطلبها العميل فى أمر الشراء من كل صنف واضح أنها بين أمر الشراء و الصنف أى تخص العلاقة بينهما. ذلك لأننا لا نستطيع وضع هذه الكمية مع الصنف لأنها مع الصنف لا تحدد أمر الشراء الذى طلب الصنف. كذلك لا نستطيع وضعها مع أمر الشراء لأنها لا تعبر عن كمية الصنف المطلوب فى أمر الشراء. لذا فمكان الكمية تلك هو العلاقة بين أمر الشراء و الصنف.

ثانياً العلاقات الرابطة :

- يرسل (SEND) و هى علاقة رابطة بين العميل و أمر الشراء. نظراً لأن أى عميل يستطيع أن يرسل عدة أوامر شراء و على الأقل أمر واحد حتى يصبح عميلاً، فإن الكارديناليتى تصبح (1:N) ناحية أمر الشراء.

بالنسبة لأمر الشراء فلا يمكن أن يرسله إلا عميل واحد على الأكثر و على الأقل، لذلك تصبح الكارديناليتى (1:1) ناحية العميل. هذه العلاقة الرابطة نوعها one-to-many.

- يحتوى على (CONTAIN) و هى علاقة رابطة بين أمر الشراء و الصنف و لها الخاصية : الكمية المطلوبة من كل صنف فى أمر الشراء (Req_Quantity). كل أمر شراء يمكن أن يحتوى على عدة أصناف و على الأقل صنف واحد، لذلك تصبح الكاردينالتى (1:N) ناحية الصنف.



شكل ٣-٥ : نموذج ERD لتطبيق أوامر الشراء للعملاء.

بالنسبة لأى صنف يمكن أن يوجد فى أكثر من أمر شراء و ربما لا يطلبه أى أحد فى أمر الشراء، لذلك تصبح الكاردينالتى (0:N) ناحية أمر الشراء. هذه العلاقة الرابطة نوعها many-to-many.

٣-٧ تطبيق سوق الأوراق المالية

Stock Exchange System

مطلوب رسم ERD لسوق الأوراق المالية خاصة بعملية تداول الأسهم . حيث يتم التعامل على أسهم عدة شركات، كل سهم له كود و السعر الاسمى و إجمالى عدد أسهم شركة من هذا السهم ورقم و اسم الشركة وعنوانها. علماً بأنه يمكن أن يكون لكل شركة أكثر من سهم. كذلك يتم تسجيل بيانات المستثمر الذى يمكن أن يمتلك أى عدد من أى نوع من الأسهم مع تسجيل بيانات كافة عمليات الشراء التى قام بها من سعر شرائه للسهم و تاريخ الشراء و عدد الأسهم المشتراة و كذلك بيانات عمليات البيع التى قام بها من سعر بيعه للسهم و تاريخ البيع و عدد الأسهم المباعة إلى جانب بيانات المستثمر الأساسية من رقم تحقيق الشخصية و نوعه واسمه عنوانه.

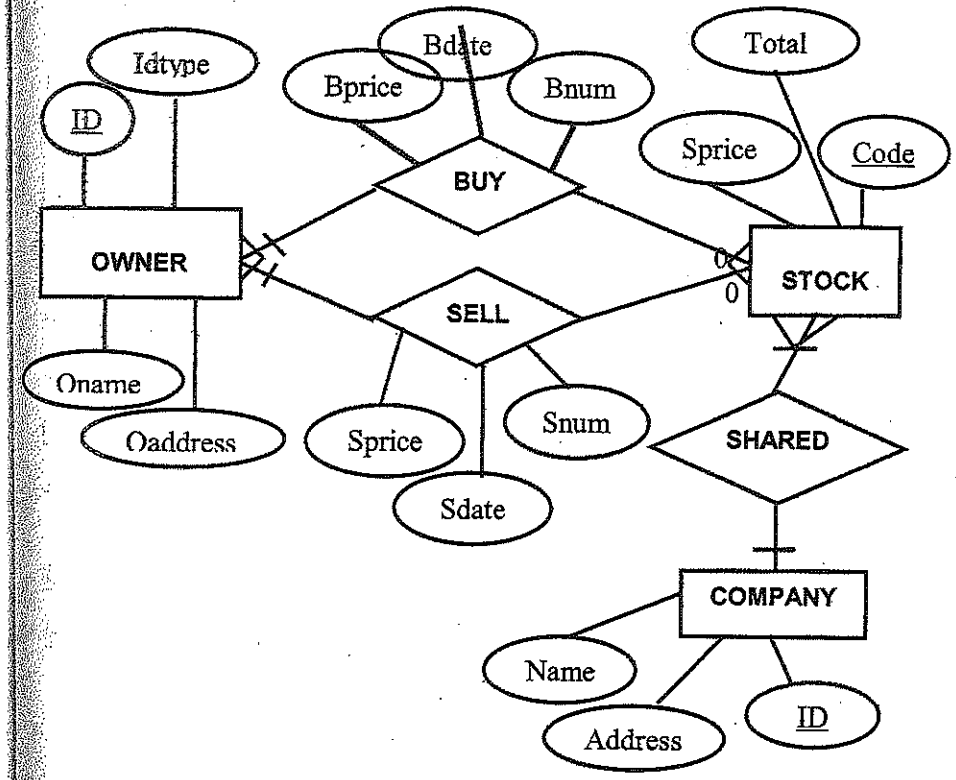
التصميم : (فى الشكل ٣-٦)

أولاً الكيانات :

- السهم (STOCK) و له الخصائص : كود السهم (Code) و السعر الاسمى (Sprice) و إجمالى الأسهم (Total).

- الشركة التى يتبعها السهم (COMPANY) و له الخصائص : رقم الشركة (C#) و اسم الشركة (Name) و عنوانها (Address).

- مالك السهم (OWNER) و له الخصائص : رقم تحقيق الشخصية (ID) و نوعه (Idtype) و اسمه (Oname) و عنوانه (Oaddress).



شكل ٣-٦ : نموذج ERD لتطبيق شراء و بيع

ثانياً العلاقات الرابطة :

- يشترى (BUY) و هى علاقة رابطة تربط بين المستثمر و السهم و خصائصها : عدد الأسهم التى اشتراها (Bnum) و سعر الشراء (Bprice) و تاريخ الشراء (Bdate). حيث أن كل عميل يمكنه شراء أى عدد من الأسهم من سهم معين أو لا يشترى على الإطلاق، لذلك فإن الكاردينالتى تصبح (0:N) ناحية السهم. و حيث أن كل سهم يمكن أن يشتريه أى عدد من العملاء و على الأقل عميل واحد، فإن الكاردينالتى تصبح (1:M) ناحية العميل. هذه العلاقة many-to-many.

- يبيع (SELL) و هى علاقة رابطة بين المستثمر و السهم و خصائصها : عدد الأسهم المباعة (Snum) و سعر البيع (Sprice) و تاريخ البيع (Sdate). حيث أن كل عميل يمكنه بيع أى عدد من الأسهم من سهم معين أو لا يبيع على الإطلاق، لذلك فإن الكاردينالتى تصبح (0:N) ناحية السهم. و حيث أن كل سهم يمكن أن يبيعه أى عدد من العملاء و على الأقل عميل واحد، فإن الكاردينالتى تصبح (1:M) ناحية العميل. هذه العلاقة many-to-many.

- يُشارك به فى (SHARED) و هى علاقة رابطة بين السهم و الشركة. حيث أن أى سهم يتبع شركة واحدة على الأقل و على الأكثر، فإن الكاردينالتى تصبح (1:1) ناحية الشركة. و حيث أن أى شركة ممكن أن

يكون لها أكثر من سهم و سهم واحد على الأقل، لذلك تصبح الكاردينالتى (1:N) ناحية السهم.

٣-٨ تطبيق حسابات العملاء فى المصارف

Bank Accounts Systems

ارسم ERD لتصميم قاعدة بيانات تتبع بيانات عملاء المصارف على مستوى الدولة من رقم تحقيق الشخصية و نوعها و الاسم و العنوان و الهاتف وكذلك حسابات كل عميل من رقم و نوع عملة ورصيد. علماً بأنه يمكن لأى عميل أن يفتح أكثر من حساب فى نفس المصرف و يمكن أن يكون الحساب مشتركاً. نحتفظ لكل مصرف برقم و اسم و عنوان و اسم مدير عام المصرف و كذلك بكل سحب أو إيداع يقوم به العميل على الحساب من قيمة و تاريخ.

التصميم : (فى الشكل ٣-٧)

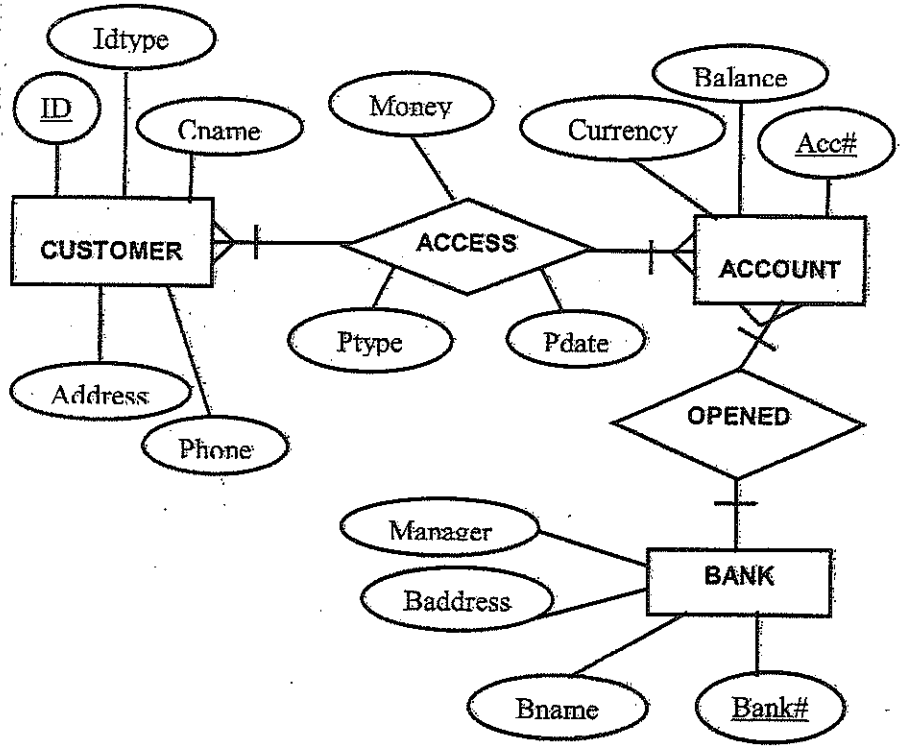
أولاً الكيانات :

- العميل (CUSTOMER) و له الخصائص : رقم تحقيق الشخصية (ID) و نوع (Idtype) و الاسم (Cname) و العنوان (Address) و الهاتف (Phone).
- الحساب (ACCOUNT) و له الخصائص : رقم الحساب (Acc#) و نوع العملة (Currency) و الرصيد (Balance).

- المصرف (BANK) و له الخصائص : رقم المصرف (Bank#) و اسم المصرف (Bname) و اسم المدير (Manager) و عنوان المصرف (Baddress).

ثانياً العلاقات الرابطة :

- يتناول الحساب (ACCESS) و هى علاقة رابطة بين العميل و الحساب و لها الخصائص : نوع العملية (سحب/إيداع)(Ptype) و قيمة المبلغ (Money) و تاريخ العملية (Pdate). (تضم هذه العلاقة فتح الحساب و السحب و الإيداع). حيث أن أى عميل يمكن أن يفتح/يسحب/يودع فى أكثر من حساب و على الأقل حساب واحد، تصبح الكاردينالتى (1:N) ناحية الحساب. و حيث أن كل حساب يمكن أن يكون مشتركاً (متناوَلاً) بين أكثر من عميل و على الأقل عميل واحد، تصبح الكاردينالتى (1:M) ناحية العميل.



شكل ٣-٧ : نموذج ERD لتطبيق حسابات عملاء المصارف.

- حساب مفتوح فى مصرف (OPENED) و هى علاقة رابطة بين الحساب و المصرف. حيث أن كل عميل يمكنه فتح أكثر من حساب فى نفس المصرف و حساب واحد على الأقل ليصبح عميلاً، تصبح الكارديناليتى (1:N) ناحية الحساب. و حيث أن أى حساب يتم فتحه فى مصرف واحد على الأكثر و على الأقل، تصبح الكارديناليتى (1:1) ناحية المصرف.

٣-٩ أسئلة

١. ارسم ERD اللازم لقاعدة بيانات لتخزين البيانات الخاصة بأمر الشراء الذي يرسله العميل و يهتم في هذه العملية برقم العميل و اسمه و عنوانه و أرقام أوامر الشراء و تاريخ صدورها و كذلك الكمية المطلوبة من العميل في أمر الشراء من كل صنف و رقم كل صنف و اسمه و سعره و الجهة المصنعة للصنف و عنوانها و تاريخ انتهاء الصلاحية و الفواتير الصادرة للعميل بما فيها من رقم الفاتورة و تاريخ الفاتورة و الكمية الموردة من كل صنف للعميل و إجمالى كل فاتورة.

٢. ارسم ERD لتصميم قاعدة بيانات تتبع بيانات عملاء المصارف على مستوى المملكة من رقم الحفيظة أو الإقامة و الاسم و العنوان و الهاتف وكذلك حسابات قروض كل عميل من رقم و نوع وقيمة و رصيد القرض. علماً بأنه لا يمكن لعميل أن يحصل إلا على قرض واحد فى نفس الوقت. و نحتفظ لكل مصرف برقم و اسم و عنوان و اسم مدير عام المصرف و كذلك بكل عملية سداد من قيمة و تاريخ.

٣. ارسم ERD اللازم لقاعدة بيانات المكتبات فيما يخص بيانات المستعير من رقم واسم وكذلك وظيفته و جهة عمله وعنوانها و تاريخ ميلاده و نوع الاستعارة و مدة الاستعارة و تاريخها و كذلك بيانات الكتاب كرقم و عنوان و دار النشر و سنة النشر و عدد صفحات الكتاب و المؤلفين و عناوينهم

الإلكترونية. مع الاحتفاظ بالمعلومات الخاصة بالمكتبة التى يُستعار منها الكتب من اسم و عنوان و هاتف.

٤. ارسم ERD اللازم لتسجيل مخالفات المرور. حيث يتم تحرير مخالفة للمخالف مبيناً بها رقم المخالفة و تاريخها و نوعها و قيمة الغرامة ومكان المخالفة كذلك رقم تحقيق الشخصية المخالف و نوعها و جهة عمله و عنوانها مع تسجيل رقم الشرطى الذى حرر المخالفة.

٥. ارسم ERD لتسجيل مبيعات السيارات فى صالة عرض سيارات. حيث يُحتفَظ ببيانات السيارة من رقم و نوع و سنة صنع و لونها و كذلك بيانات صاحب السيارة من رقم تحقيق الشخصية و نوعها واسمه و هاتفه و جهة عمله و عنوانها. عندما يقوم بشرائها مشترى معين يتم تسجيل بيانات المشترى (مثل البائع) و بيانات عقد البيع من رقم و تاريخ البيع و القيمة المدفوعة و طريقة الدفع و الموظف الذى حرر العقد.

الفصل الرابع

تصميم قواعد البيانات العلاقية

Relational Database Design

فى هذا الفصل سوف نكمل ما بدأناه فى الفصول الثلاثة السابقة لنقترب بالدارسة و الدارس من مرحلة احتراف تصميم قواعد البيانات العلاقية. و سوف نبدأ مرحلة تصميم نموذج ERD لتطبيقات قواعد البيانات العلاقية الكبيرة بعد التعرُّف على كيفية استخدام نموذج ERD لتصميم قواعد البيانات.

٤-١ تصميم قواعد البيانات باستخدام نموذج ERD

Database Design by ERD Model

الخطوة الهامة نحو تصميم قاعدة البيانات هى تصميم نموذج ERD لتطبيق ما. الخطوة التالية هى تحويل كافة الكيانات و العلاقات الرابطة الموجودة فى هذا النموذج إلى **علاقات (Relations)**. هذه العلاقات تتحول أو تأخذ الاسم جداول لنستخدمها مباشرة فى إنشاء قاعدة البيانات باستخدام أحد نظم DBMS. خلال هذا التحويل نلتزم بكافة الرموز و القيود الموجودة على الرسم لنحصل على تصميم قاعدة البيانات. يجب أن نحدد المفاتيح الأساسية و الخارجية لكافة العلاقات.

٤-١-١ تحويل الكيان إلى علاقة

Mapping Entity to Relation

يتم تحويل جميع الكيانات الموجودة فى نموذج ERD لأى تطبيق إلى علاقات. اسم العلاقة هو نفس اسم الكيان ثم نضع أمامه أسماء كافة خصائص الكيان بين قوسين. يجب ألا ننسى أن نضع خطاً تحت المفتاح الأساسى لكل كيان. نطبّق ذلك على نموذج ERD الموجود فى شكل ٣-٧ و الخاص بتطبيق حسابات العملاء. نستطيع تحديد العلاقات الخاصة بالكيانات فى هذا النموذج كما يلى :

CUSTOMER (ID, Idtype, Cname, Address, Phone)
ACCOUNT (Acc#, Balance, Currency)
BANK (Bank#, Bname, Baddress, Manager)

٤-١-٢ تحويل العلاقة الرابطة إلى علاقة

Mapping Relationship to Relation

أى علاقة رابطة موجودة فى نموذج ERD لتطبيق ما نقوم بتحويلها إلى علاقة مثلما فعلنا مع الكيانات. تأخذ العلاقة الناشئة نفس اسم العلاقة الرابطة فى ERD.

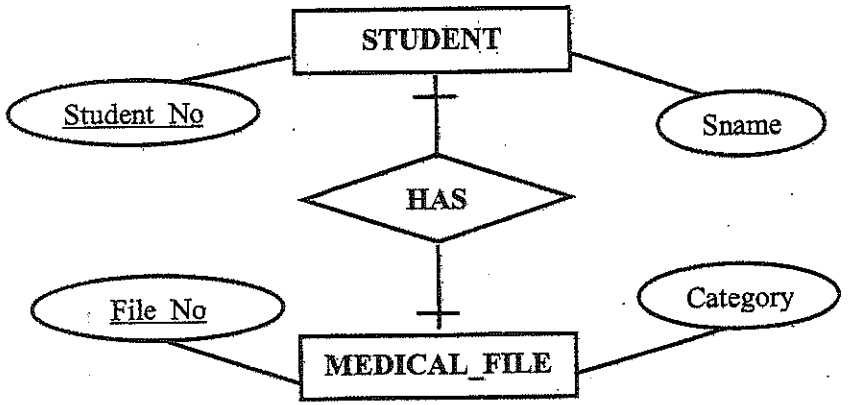
أما خصائص أى علاقة ناتجة من علاقة رابطة فتتكون من الخصائص الموجودة فى نموذج ERD بالإضافة إلى مفتاحى الكيانين اللذين تربط بينهما العلاقة الرابطة.

بعد ذلك علينا أن نحدد مفتاحاً أساسياً للعلاقة الناشئة و تختلف طريقة تحديد المفتاح تبعاً لنوع العلاقة الرابطة. يتم التحديد كما يلى :

- **العلاقة الرابطة one-to-one**

اختيار مفتاح العلاقة الناتجة عن علاقة رابطة من النوع one-to-one يكون اختيارياً لأحد مفتاحى الكيانين اللذين تربط بينهما العلاقة الرابطة.

على سبيل المثال العلاقة الرابطة HAS الموجودة في الشكل ٤-١ من النوع one-to-one و تربط الكيانين STUDENT و MEDICAL_FILE. يتم تحويل العلاقة الرابطة HAS إلى علاقة تأخذ أحد الاختيارين الموجودين فى شكل ٤-١.



- HAS(Student_No, File_No)
- HAS(Student_No, File_No)

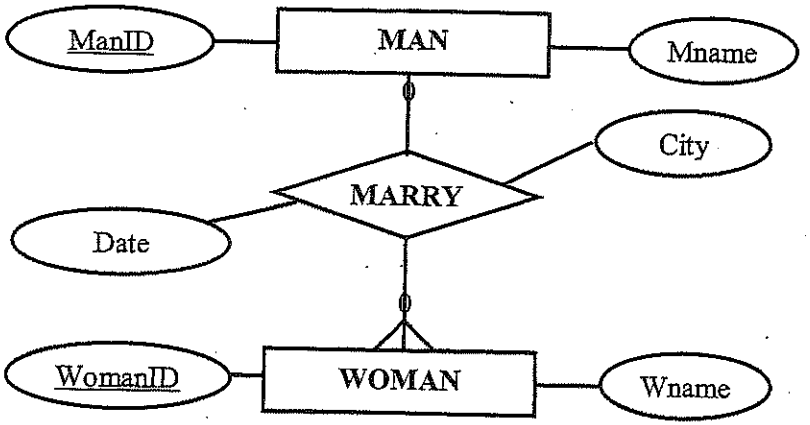
شكل ٤-١ : علاقة ONE-TO-ONE.

في الاختيار الأول اخترنا الخاصية Student_No (مفتاح الكيان STUDENT) مفتاحاً للعلاقة HAS. أما الاختيار الثانى فقد اخترنا الخاصية File_No مفتاحاً للعلاقة HAS. و علينا بعد أن نختار أحد الاختيارين لنكمل الحل بما له من تبعات.

- **العلاقة الرابطة one-to-many**

اختيار المفتاح الأساسى للعلاقة الناتجة عن علاقة رابطة من النوع one-to-many يكون إجبارياً. المفتاح الأساسى للكيان ناحية many فى العلاقة الربطة، لابد أن نتخذه مفتاحاً أساسياً للعلاقة الناشئة عن العلاقة الرابطة من هذا النوع.

على سبيل المثال العلاقة الرابطة MARRY (يتزوج) الموجودة فى الشكل ٤-٢ من النوع one-to-many و تربط الكيانين MAN و WOMAN.



- MARRY(WomanID, ManID, Date, City)

شكل ٤-٢ : علاقة ONE-TO-MANY.

يتم تحويل العلاقة الرابطة MARRY إلى علاقة تتضمن المفتاحين الأساسيين للكيانين MAN و WOMAN بالإضافة إلى خاصيتى العلاقة الرابطة : Date (تاريخ الزواج) و City (محل الزواج). يكون المفتاح الأساسى للعلاقة الناشئة هو مفتاح الكيان ناحية many و هو WomanID. العلاقة الناشئة نراها فى شكل ٤-٢.

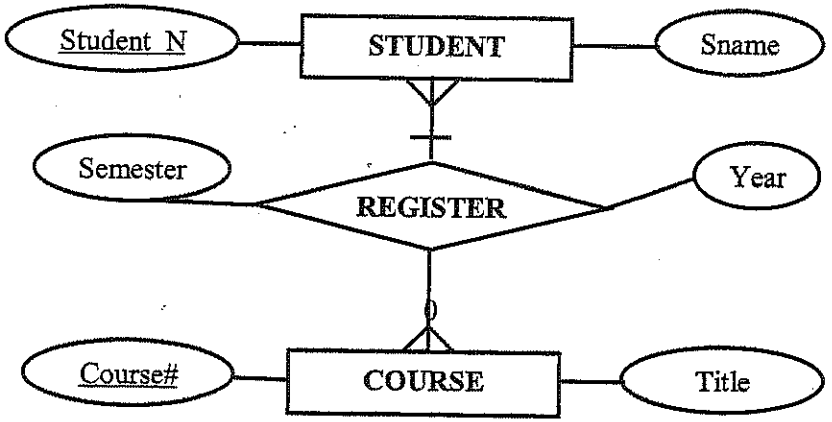
- **العلاقة الرابطة many-to-many**

اختيار مفتاح العلاقة الناتجة عن علاقة رابطة نوعها many-to-many يكون إجباريا. فلابد أن يكون المفتاح مُكوَّناً من المفتاحيين الأساسيين للكيانين اللذين تربط بينهما العلاقة الرابطة. كذلك تضم العلاقة الناشئة خصائص العلاقة الرابطة إلى جانب المفتاحيين.

على سبيل المثال العلاقة الرابطة REGISTER الموجودة فى الشكل ٤-٣، من النوع many-to-many و تربط الكيانين STUDENT و COURSE. يتم تحويل العلاقة الرابطة REGISTER إلى علاقة نراها فى شكل ٤-٣.

كما نرى فإن المفتاح الأساسى للعلاقة الناشئة يتكون من كلٍ من المفتاح الأساسى للكيان STUDENT و هو (Student#) و المفتاح الأساسى للكيان COURSE و هو (Course#). فنضع تحتهما معا خطاً

متصلاً. أيضاً، تحتوى العلاقة الناشئة على خاصيتى العلاقة الرابطة و هما Semester (الفصل الراسى الذى سجّل فيه الطالب المقرر) و Year (العام الدراسى).



- REGISTER(Student#, Course#, Semester, Year)

شكل ٤-٣ : علاقة MANY-TO-MANY.

نذكر الدارس أن العلاقة REGISTER من النوع many-to-many. هذا لأن كل طالب يمكن أن يسجل عدة مقررات و كذلك أى مقرر يمكن أن سجله عدة طلبة.

٤-١-٣ مثال تحويل نموذج ERD إلى قاعدة بيانات

Example of Mapping ERD Model To Database

بعد أن تعرّفنا على كل ما يتعلق بتحويل مكونات نموذج ERD إلى علاقات (Relations). نطبّق ذلك على المثال الذى درسناه فى التطبيق ٣-٨ فى الفصل السابق حيث نرى نموذج ERD لتطبيق حسابات العملاء المصارف فى الشكل ٣-٧. نتيجة تحويل الكيانات و العلاقات الرابطة إلى علاقات تُشكّل قاعدة بيانات تتكون من خمس علاقات، الثلاث علاقات الأولى منها تخص الثلاث كيانات و العلاقتان الأخيرتان تخصا العلاقتان الرابطتان حسب ما هو مبين فى نموذج ERD. علاقات قاعدة البيانات تكون كما يلى :

أولاً : علاقات من الكيانات

(1) CUSTOMER (ID, Idtype, Cname, Address, Phone)

(2) ACCOUNT (Acc#, Balance, Currency)

(3) BANK (Bank#, Bname, Baddress, Manager)

ثانياً : علاقات من العلاقات الرابطة

(4) ACCESS (ID, Acc#, Ptype, Pdate, Money)

(5) OPENED (Acc#, Bank#)

نلاحظ أن العلاقة الناشئة عن العلاقة الرابطة ACCESS تتكون من المفتاح الأساسى لكلٍ من الكيانين اللذين تربطهما و تحتهم معاً خط متصل،

بالإضافة إلى خصائص العلاقة الرابطة : Ptype و Pdate و Money. المفتاح الأساسى لهذه العلاقة يتكون من خاصيتين (ID,Acc#) لأن العلاقة الرابطة ACCESS من النوع many-to-many.

أما العلاقة الناشئة عن العلاقة الرابطة OPENED فتتكون فقط من المفتاح الأساسى لكل من الكيانين اللذين تربطهما لعدم وجود خصائص للعلاقة الرابطة. المفتاح الأساسى لهذه العلاقة يتكون من حقل واحد و هو مفتاح الكيان ناحية many و هو (Acc#) لأن العلاقة الرابطة OPENED من النوع one-to-many.

أخر ما نرغب فى تحديده فى قاعدة البيانات تلك هو تحديد المفاتيح الخارجية لكل جدول بوضع خط متقطع أسفل منها. و نذكّر الدارس أن المفتاح الخارجى لجدول A هو حقل عادى فى الجدول A و لكنه مفتاحاً أساسياً فى جدول آخر B.

يوجد فى قاعدة البيانات محل دراستنا مفتاحاً خارجياً واحداً فى العلاقة OPENED رقم ٥ و هو Bank# لأنه مفتاحاً أساسياً فى العلاقة BANK رقم ٣ فى قاعدة البيانات. و قد وضعنا أسفله خط متقطع.

٤-٢ تحسين قاعدة البيانات Database Optimization

يمكن تنفيذ قاعدة البيانات التي بنيناها حالاً لتطبيق حسابات عملاء المصارف باستخدام أحد نظم DBMS. كما يمكننا عمل ذلك لقاعدة بيانات أى تطبيق نستنتجه من نموذج ERD لذلك التطبيق.

مع ذلك يمكننا إجراء **تحسين (Optimization)** لقاعدة بيانات أى تطبيق و ذلك باستخدام بعض القواعد للوصول إلى قاعدة البيانات المُثلى (Optimized Database). نقوم بذلك عن طريق دمج العلاقات التى لها نفس المفتاح الأساسى فى علاقة واحدة تضم خصائص كل منهم بالإضافة إلى المفتاح الأساسى المشترك.

بتطبيق ذلك على قاعدة بيانات حسابات عملاء المصارف التى درسناها فى المثال السابق نحصل على قاعدة البيانات المُحَسَّنة التالية :

(1) ➔ (1) CUSTOMER (ID, Idtype, Cname, Address, Phone)

(2)&(5) ➔ (2) OPPENED_ACCOUNT(Acc#,Balance,Currency,Bank#)

(3) ➔ (3) BANK (Bank#, Bname, Baddress, Manager)

(4) ➔ (4) ACCESS (ID, Acc#, Ptype, Pdate, Money)

كما نرى فإن العلاقات (١) و (٣) و (٤) ظلّت كما هى دون تغيير. بينما تم دمج العلاقتين (٢)BANK و (٥)OPENED فى علاقة واحدة جديدة رقم (٢)

OPENED_ACCOUNT مفتاحها الأساسى هو نفس المفتاح الأساسى لكل من العلاقتين ٢ و٥.

بالتدقيق نجد أن العلاقة OPENED ناتجة من العلاقة الرابطة OPENED من النوع one-to-many. لذلك فقد أمكن دمجها مع العلاقة الناشئة عن الكيان ناحية many. أما العلاقة ACCESS فهى ناتجة عن علاقة رابطة من النوع many-to-many، و لا يمكن دمجها مع أى علاقة أخرى.

٤-٣ الكيانات الضعيفة Weak Entities

تضم تطبيقات قاعدة البيانات عدة أنواع من الكيانات، نستهدف منها نوعين هامين هما الكيان القوى و الكيان الضعيف.

٤-٣-١ الكيان القوى مقابل الكيان الضعيف

Strong Entity versus Weak Entity

- **كيان قوى (مستقل و مسيطر)**

الكيان القوى هو كيان مستقل له خصائصه و من ضمنها خاصية المفتاح الأساسى التى تميز أى سجل ينتمى له عن باقى السجلات التى تنتمى إليه. جميع الكيانات التى مرّت علينا خلال هذا الكتاب هى كيانات قوية.

- **كيان ضعيف (تابع و خاضع)**

الكيان الضعيف هو كيان تابع لكيان آخر قوى و مسيطر. لا يوجد مفتاح أساسى مميز لسجلات الكيان الضعيف بل نستخدم المفتاح الأساسى للكيان المسيطر كمفتاح أساسى للكيان الضعيف بالإضافة إلى خاصية من خصائص الكيان الضعيف يُسمَّى **المفتاح الجزئى (partial key)**. هذا المفتاح الجزئى يميز سجلات الكيان الضعيف عن بعضها البعض. نوضّح الكيان الضعيف و العلاقة الرابطة بينه و بين الكيان القوى المسيطر بوضع اسميهما فى مستطيل و معين مزدوجا الحدود الخارجية على الترتيب، كما نرى فى الشكل ٤-٤.

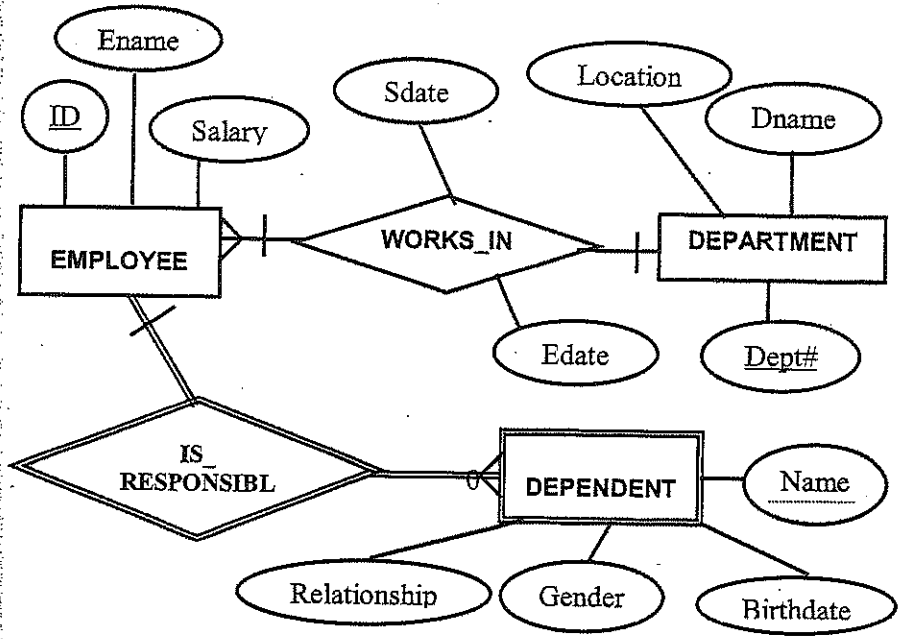
٤-٣-٢ نموذج ERD للكيان الضعيف

ERD Model of Weak Entity

الشكل ٤-٤ يعرض نموذج ERD للعاملين بأقسام أحد الشركات. كما يوجد فى الشكل كيانين قويين هما EMPLOYEE (الموظف) و DEPARTMENT (القسم) و تربط بينهما علاقة رابطة WORKS_IN (يعمل فى) من النوع one-to-many.

نرى أيضاً الكيان الضعيف DEPENDENT (تابع/مُعال) الذى يرتبط بالكيان القوى EMPLOYEE (موظف) عن طريق العلاقة الرابطة IS_REPONSIBLE (مسئولاً عن). هذه العلاقة الرابطة من النوع one-to-

many. حيث أن كل تابع/معال يكون تابعاً لموظف واحد على الأكثر و على الأقل، لذلك تكون الكاردينالتى (1:1) ناحية الموظف. بينما كل موظف يمكن أن يكون مسئولاً عن عدة تابعين أو لا يعول أحداً على الإطلاق، لذلك فإن الكاردينالتى تكون (0:N) ناحية التابع/المُعَال. لاحظ وجود المفتاح الجزئى (Name) و تحته خط متقطع.



شكل ٤-٤ : نموذج ERD للعاملين فى شركة يحتوى على كيان ضعيف.

٤-٣-٣ تحويل ERD للكيان الضعيف إلى قاعدة بيانات

Mapping ERD Model of Weak Entity to Database

الآن علينا أن نحوّل نموذج ERD هذا إلى قاعدة بيانات ثم نقوم بعمل أى تحسينات ممكنة عليها. مثلما فعلنا فى ٤-١-٣ و ٤-٣. قاعدة البيانات لنموذج ERD الموجود فى الشكل ٤-٤ تكون كما يلى :

أولاً : علاقات من الكيانات

(1) EMPLOYEE (ID, Ename, Salary)

(2) DEPARTMENT (Dept#, Dname, Location)

(3) DEPENDENT (ID,Name, Relationship, Gender, Birthdate)

ثانياً : علاقات من العلاقات الرابطة

(4) WORKS_IN (ID, Dept#, Sdate, Edate)

لاحظ أن الخاصية Dept# فى العلاقة WORKS_IN موجودة فى العلاقة DEPARTMNET كمفتاح أساسى. لذلك فإن Dept# يُعتبَر مفتاحاً خارجياً فى العلاقة WORKS_IN و نضع أسفله خط متقطع.

لاحظ أيضاً أننا لا نحتاج إلى تمثيل العلاقة الضعيفة IS_RESPONSIBLE التى تربط بين الكيان القوى (EMPLOYEE) و الكيان الضعيف (DEPENDENT).

ثالثاً : تحسين قاعدة البيانات

يوجد لدينا فى نموذج ERD علاقة رابطة من النوع one-to-many و هى WORKS_IN. هذه العلاقة رقم(٤) مفتاحها الأساسى هو ID (ناحية many) وهو نفس المفتاح الأساسى للكيان EMPLOYEE رقم (١)، لذلك يمكن دمجهما لتصبح قاعدة البيانات كما يلى :

(1)&(4)➔(1) EMPWORKS(ID, Ename, Salary, Dept#, Sdate, Edate)
(2) ➔(2) DEPARTMENT (Dept#, Dname, Location)
(3) ➔(3) DEPENDENT (ID,Name, Relationship, Gender, Birthdate)

٤-٤ العلاقة الربطة ISA ISA Relationship

فى بعض التطبيقات يظهر لدينا عدة كيانات تربط ببعضها البعض بعلاقة رابطة ذات طبيعة خاصة تختلف عما درسناه من قبل. هذه العلاقة الرابطة هى علاقة بين كيانين أحدهما طبقة أعلى (أصل أو أب) و الأخرى طبقة أسفل (فرع أو ابن) متفرعة من الطبقة الأصل.

٤-٤-١ مفهوم العلاقة الرابطة ISA

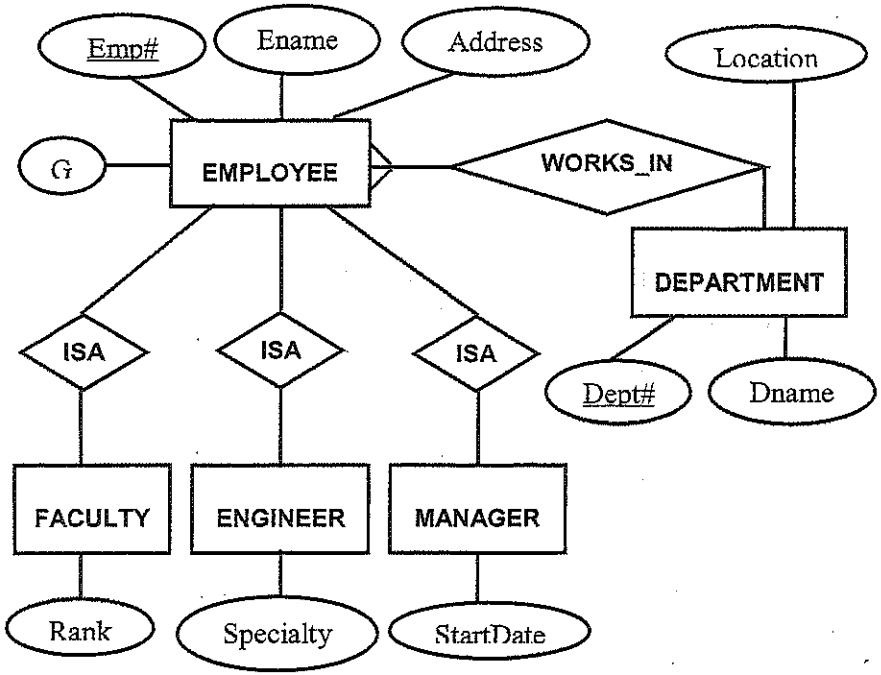
Concept of ISA Relationship

العلاقة الرابطة بين طبقة فرعية و طبقة أصل يُطلَق عليها العلاقة الرابطة IS_A أو اختصاراً ISA. واضح أن هذه العلاقة الرابطة معناها "هذا يمكن أن يكون" أو

"هذا يكون" و نستخدمها بالقول "أن هذا الكيان الطبقة الأصل يكون طبقة فرعية و العكس. أمثلة للعلاقة الرابطة ISA نراها فى الشكل ٤-٥.

الشكل ٤-٥ يعرض مثالاً لنموذج ERD لتطبيق قاعدة بيانات موظفى الأقسام بكليةٍ ما. كما نلاحظ فى الشكل تواجد العلاقة الرابطة ISA ثلاث مرات لتربط بين EMPLOYEE (الموظف) من جهة و كلٍ من FACULTY (عضو هيئة التدريس) و ENGINEER (مهندس) و MANAGER (مدير) من جهة أخرى.

العلاقات الرابطة ISA تعنى أن الموظف يمكن أن يكون فى نفس الوقت أحد ثلاث احتمالات (حسب الرسم) إما عضو هيئة تدريس أو مهندساً أو مديراً. و نلاحظ أن كل وظيفة (كيان) من الوظائف المحتملة لها خصائص تخصها فقط دون غيرها من الوظائف. فوظيفة عضو هيئة التدريس لها الخاصية Rank (المرتبة العلمية) و وظيفة المهندس لها خاصية Specialty (التخصص الهندسى) بينما وظيفة المدير فلها خاصية StartDate (تاريخ بدء عمله كمدير).



شكل ٤-٥ : نموذج ERD يحتوى على العلاقة الرابطة ISA.

بناءً على ما سبق فإن أى موظف يتم توزيع بياناته على كيانين : الأول يضم الخصائص العامة للموظف { Emp# (رقم الموظف) و G (نوع الجنس) و Ename (اسم الموظف) و Address (عنوان الموظف)} و الجزء الثانى يضم الخصائص الخاصة لكل وظيفة. مما يوفر حيز التخزين.

نفهم من ذلك أن الفصيلة الأصل EMPLOYEE تم تقسيمها إلى ثلاث أنواع من الوظائف أو التخصصات :FACULTY و ENGINEER و EMPLOYEE أو العكس. أى أن الوظائف الثلاث يمكن تجميعها فى الوظيفة العامة و هى الموظف. نقوم بالتجميع أو التقسيم حسب الحاجة و حسب حجم البيانات فى التخصصات (الفصائل الفرعية).

٤-٤-٢ تصميم قاعدة بيانات تحتوى على ISA

Designing a Database has ISA

الشكل ٤-٥ يعرض نموذج ERD لقاعدة بيانات الموظفين و علينا الآن أن نحوله إلى قاعدة بيانات.

أولاً : علاقات من الكيانات

الكيانات الموجودة خمس كيانات : EMPLOYE و DEPARTMENT و FACULTY و ENGINEER و MANAGER. الكيانين الأوليين يتم تحويلهم كما تعودنا إلى علاقتين. أما الكيانات الثلاث الأخيرة فهى جديدة بالنسبة لنا. يتم تحويل أى كيان منهم – مرتبط بعلاقة رابطة ISA مع كيان آخر – إلى علاقة مفتاحها الأساسى هو نفس المفتاح الأساسى للكيان الموجود على طرف ISA الآخر. و مع المفتاح الأساسى نضع الخصائص الخاصة. بناءً على ذلك نحصل على العلاقات التالية :

(1) DEPARTMENT (Dept#, Dname, Location)

(2) EMPLOYEE (Emp#, Ename, Address, G)

(3) FACULTY (Emp#, Rank)

(4) ENGINEER (Emp#, Specialty)

(5) MANAGER (Emp#, StartDate)

ثانياً : علاقات من العلاقات الرابطة

لدينا علاقة رابطة واحدة و هى WORKS_IN تربط بين الموظف و القسم الذى يعمل به و هى علاقة نوعها one-to-many. **يوجد لدينا ثلاث علاقات ISA و لا نمثلها فى قاعدة البيانات.** بالتالى يكون لدينا العلاقة التالية :

(6) WORKS_IN (Emp#, Dept#)

ثالثاً : تحسين قاعدة البيانات

بالنظر إلى قاعدة البيانات التى استنتجناها من نموذج ERD الموجود فى الشكل ٤-٥، نجد أن العلاقات ٢ و ٣ و ٤ و ٥ و ٦ لهم نفس المفتاح و هو Emp#، إلا أننا **لن ندمج** إلا العلاقتين ٢ (EMPLOYEE) و ٦ (WORKS_IN). يمكن دمج العلاقات الثلاث الأخرى معهم **فقط** إذا أردنا أن نلغى التقسيم أو التخصص. تكون قاعدة البيانات بعد التحسين كما يلى :

(1) ➔(1) DEPARTMENT (Dept#, Dname, Location)

(2)&(6)➔(2)EMPLOYEE_WORKS_IN(Emp#, Ename, Address, Dept#, G)

(3) ➔(3) FACULTY (Emp#, Rank)

(4) ➔(4) ENGINEER (Emp#, Specialty)

(5) ➔(5) MANAGER (Emp#, StartDate)

٤-٥ تصميم قواعد بيانات من مستندات

Designing Database from Documents

فى تطبيقات قواعد البيانات التى درسناها فى الفصل الثالث بنينا نماذج ERD و بالتالى قواعد البيانات بناءً على معطيات وصفية أى وصف للموضوع. فى أغلب الأحيان يختلط الوصف أو ينفصل مع/عن البيانات الموجودة فى مستندات لدى الجهة صاحبة التطبيق. معالجة مستند مثل فاتورة المشتريات الموجودة فى شكل ٤-٦، تختلف عن الوصف و هذا ما سنعالجه الآن.

٤-٥-١ البيانات الغير ممثلة فى قواعد البيانات

Non-Represented Data

عندما نطّلع على مُستَنَدٍ ما به بيانات كثيرة، فمن ضمن هذه البيانات نجد أن هناك أنواع كثير لا يتم تسجيلها فى قاعدة البيانات. لذلك علينا أن نحدد أولاً أنواع البيانات التى لا يتم تسجيلها لنقوم بعد ذلك بدراسة باقى البيانات الموجودة فى المستند و بناء نموذج ERD للمستند و من ثمّ نبنى قاعدة البيانات. البيانات التى لا يتم تسجيلها فى قاعدة البيانات تقع فى الأنواع التالية :

- **الشعارات أو اسم الجهة صاحبة المستند.** مثال على ذلك "شعار الجامعة مع اسم عمادة القبول و التسجيل".
- **الرقم المسلسل أو رقم كل صفحة مطبوعة أو تاريخ الطباعة** : مثلما يظهر الرقم المسلسل فى كشف الأصناف فى فاتورة أو أمر شراء و كذلك

الرقم المسلسل للطالب فى كل كشف لطلبة الشعب الدراسية. و أيضاً رقم الصفحة المطبوعة فى كشف الطلبة و تاريخ اليوم الذى تم فيه طباعة نتيجة أو سجل طالب.

- **البيانات التى يمكن اشتقاقها أو حسابها من بيانات أخرى.** فمثلاً إذا كان لدينا فاتورة مشتريات تحتوى على مجموعة من الأصناف التى اشتراها العميل متضمنة عدد الوحدات من كل صنف و سعر كل وحدة و إجمالى سعر شراء وحدات كل صنف. هذا الإجمالى لا داعى لتخزينه لأن الحاسب يقوم بحسابه بسرعة. مع ملاحظة أن الإجمالى العام للفاتورة أو المعدل التراكمى ينبغى تخزينه بمجرد حسابه لأول مرة.

- **الملاحظات و التوقيعات و التعليقات :** مثل يتم التسليم بمقر العميل، التحميل و التنزيل على نفقة العميل أو توقيع الموظفين لا يتم تسجيله فى قاعدة البيانات.

٤-٥-٢ تطبيق مستندى Documental Application

نقدم الآن مثالاً لتطبيق فاتورة المشتريات. الشكل ٤-٦ يعرض مستند فاتورة المشتريات. يحتوى هذا المستند على مزيج من البيانات بعضها لن يتم تخزينه أو معالجته مثل :

سلسلة الأصول العلمية

1/1	Osol ElElm Group	
date : 5/11/2003	KNA	ش البطل/ مكة المكرمة
10:33:22	شعارنا العلم و النجاح	ت و ف ٣٨٠٠٠٥٠

Customer# :3434 **Customer Name** : Khaled ElSayed **Phone**: 3343444
Invoice# : 8989 **Invoice Date** : 3/9/2003 **Order** : 6767-15/8/2003

No	Item#	Description	Unit price	Quantity	Subtotal
1	101	Introduction	35	2	70
2	102	Fundamental of C	35	3	105
3	103	Fundamental of C++	35	5	175
4	214	Fundamental of Struct	35	7	245
5	325	Fundamental of DB	50	3	150
6	363	Fundamentals of AI	50	5	250
	التوصيل لمقر العميل مجاناً			Total	1145

شكل ٤-٦ : فاتورة المشتريات كمستند.

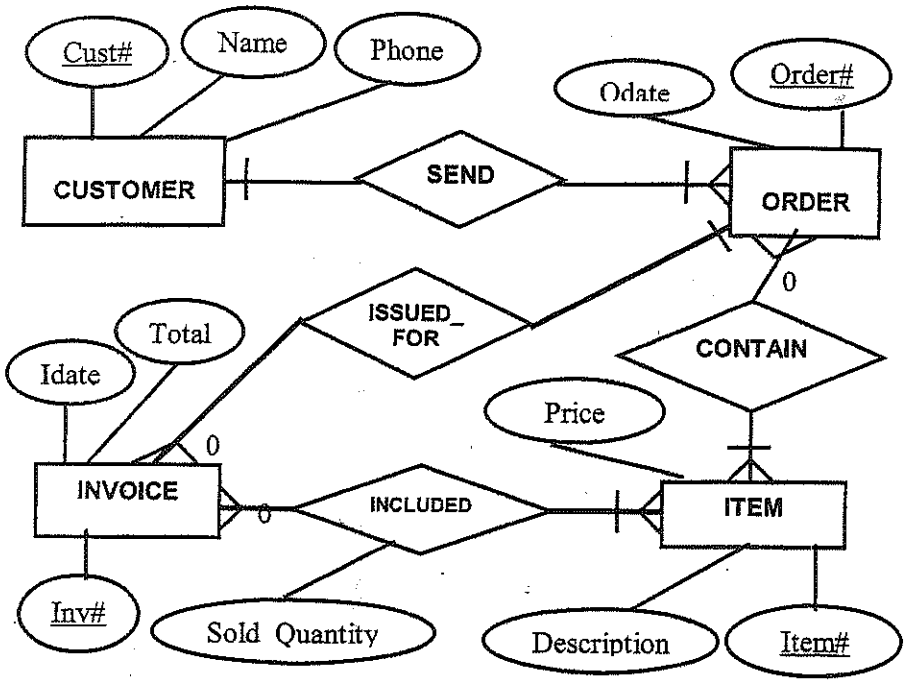
- اسم الشركة و شعارها باللغة العربية أو الإنجليزية.
- ملاحظات و إعلانات مثل "شعارنا العلم و النجاح" و " التوصيل لمقر العميل مجانا".
- عنوان و هاتف و فاكس الشركة.
- تاريخ و زمن الطباعة و رقم الصفحة و الرقم المسلسل لكل صنف No.
- إجمالى الثمن الفرعى لكل صنف subtotal.

باقى البيانات الموجودة فى الفاتورة يجب معالجتها فى قاعدة البيانات. يفحص هذه البيانات نستطيع اكتشاف الكيانات CUSTOMER و ORDER و INVOICE و ITEM و خصائصها موجودة فى الشكل ٤-٧.

لم يتبقى أى بيانات يمكن معالجتها إلا Quantity (الكمية المبُاعة من كل صنف فى الفاتورة) و أفضّل أن نسمى الخاصية Sold_quantity. هذه الخاصية لا يمكن وضعها مع الصنف أو الفاتورة أو العميل لأنها لا تعبّر عن صفة أو خاصية لأي منهم. إنما هى تخص العلاقة بين الفاتورة و الصنف حسب المعنى الذى ذكرناه.

علينا الآن رسم نموذج ERD للكيانات الأربعة التى عثرنا عليها مع تخمين العلاقات الرابطة بين بعضها البعض. نموذج ERD لمستند الفاتورة نراه فى الشكل ٤-٧. كما نرى فى الشكل لقد استنتجنا العلاقات الرابطة التالية :

- SEND (يرسل) و هى علاقة رابطة بين العميل (CUSTOMER) و أمر الشراء (ORDER). نظراً لأن كل عميل يمكنه إرسال عدة أوامر شراء و على الأقل أمر شراء واحد، فإن الكاردينالتى تصبح (1:N) ناحية أمر الشراء. أما أمر الشراء فلا يمكن إن يرسله إلا عميل واحد و واحد فقط، لذلك تصبح الكاردينالتى (1:1) ناحية العميل. أى أن هذه العلاقة one-to-many.



شكل ٤-٧ : نموذج ERD لتطبيق مستند فاتورة المشتريات.

- CONTAIN (يحتوى على) و هى علاقة رابطة بين أمر الشراء (ORDER) و الصنف (ITEM). نظراً لأن كل أمر شراء يمكن أن يحتوى على عدة أصناف و على الأقل صنف واحد، أى أن الكاردينالتى هى (1:N) ناحية الصنف. أما الصنف فيمكن أن يحتويه عدة أوامر شراء

أو لا يحتويه أى أمر على الإطلاق، أى أن الكاردينالتى تصبح (0:M) ناحية أمر الشراء. لذلك هذه العلاقة many-to-many.

- INCLUDED (يتضمن) و هى علاقة رابطة بين الصنف (ITEM) و الفاتورة (INVOICE) و لها الخاصية Sold_Quantity (الكمية المباعة). نظراً لأن كل صنف يمكن أن يوجد فى أكثر من فاتورة أو لا يوجد فى أى فاتورة على الإطلاق، فإن الكاردينالتى تصبح (0:N) ناحية الفاتورة. أما الفاتورة التى يمكن أن تحتوى على عدة أصناف و صنف واحد على الأقل، فإن الكاردينالتى تصبح (1:M) ناحية الصنف. لذلك فالعلاقة نوعها many-to-many.

- ISSUED_FOR (صدرت بسبب أو من أجل) و هى علاقة رابطة بين الفاتورة (INVOICE) و أمر الشراء (ORDER). حيث أن كل فاتورة تصدر بناءً عن أمر شراء واحد و واحد على الأقل، فإن الكاردينالتى تصبح (1:1) ناحية أمر الشراء. أما أمر الشراء فيمكن أن تتبعه (تصدر من أجله) عدة فواتير و ربما لا تصدر أى فاتورة، فإن الكاردينالتى تصبح (0:N) ناحية الفاتورة. هذه العلاقة نوعها one-to-many .

بعد ذلك علينا أن نحول نموذج ERD إلى قاعدة بيانات تحتوى على العلاقات التالية :

(1) CUSTOMER (Cust#, Name, Phone)

(2) ORDER (Order#, Odate)

(3) INVOICE (Inv# , Idate, Total)

(4) ITEM (Item#, Description, Price)

(5) SEND (Order#, Cust#)

(6) CONTAIN (Order#, Item#)

(7) INCLUDED (Inv#, Item#, Sold_Quantity)

(8) ISSUED_FOR (Inv#, Order#)

نستطيع دمج العلاقتين ٢ و ٥ لأن مفتاحهما الأساسى هو Order#. كذلك يمكننا دمج العلاقتين ٣ و ٨ لأن مفتاحهما الأساسى هو Inv#. بعد تطبيق عمليتى الدمج المذكورتين نحصل على قاعدة البيانات المُحَسَّنة التالية :

(1) ➔(1) CUSTOMER (Cust#, Name, Phone)

(2)&(5)➔(2) SEND_ORDER (Order#, Odate, Cust#)

(3)&(8)➔(3) INVOICE_ISSUED_FOR (Inv# , Idate, Total, Order#)

(4) ➔(4) ITEM (Item#, Description, Price)

(6) ➔(5) CONTAIN (Order#, Item#)

(7) ➔(6) INCLUDED (Inv#, Item#, Sold_Quantity)

٤-٦ تطبيق مسندات المستشفى

Hospital Documents Application

تصدر مستشفى الملك فهد التخصصى التقرير التالى عن المرضى المتعاملين مع أقسام المستشفى فى التخصصات المختلفة و غرف التنويم و الأدوية المنصرفة لكل مريض و كمياتها فى الشكل ٤-٨. صمم قاعدة بيانات لهذا التقرير.

KING FAHD HOSPITAL

Tel.: 322222 Fax: 343434 P.O.BOX 323232 Buraidah, ElQassim, Saudi Arabian

Page 1/5

date : 3/10/1424 Specialty : Heart

Dno	Doctor_ Name	Patient#	Patient_ Name	Patient_ Work	Work_ Address	Room#	Medicine		Q
							M#	Name	
323	Khaled	656A	Ali	KSU	Riyadh	B2121	23	ABC	10
		232B	Saad	ASU	Cairo	C2323	16	XYZ	5
		434C	Lolo	KAU	Jeddah	F4466	18	DFD	3

شكل ٤-٨ : مستند تقرير المستشفى.

الحل

علينا أولاً استبعاد البيانات التى لا يتم معالجتها أو تسجيلها و هى كالآتى :

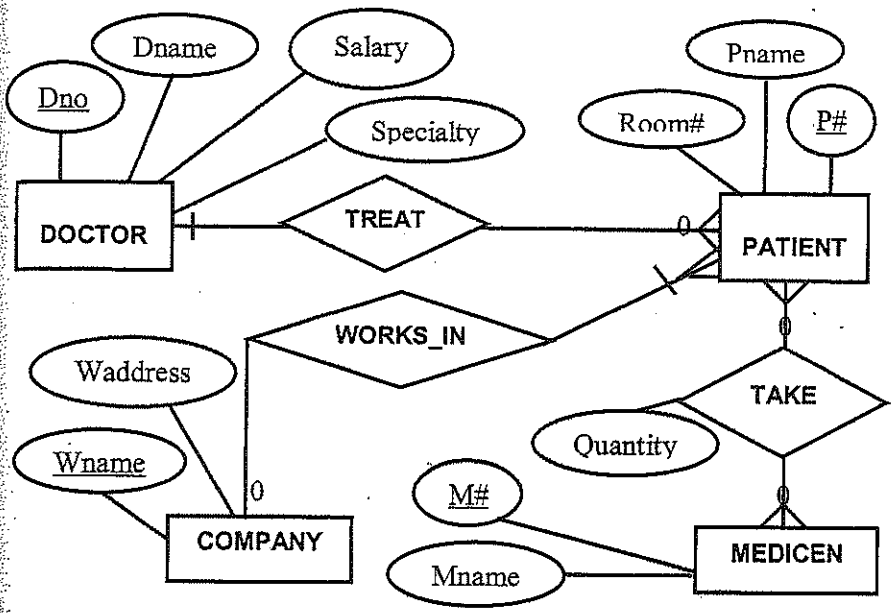
- اسم المستشفى و الشعارات و الأهلة.
- عنوان و هاتف و فاكس المستشفى.

- رقم الصفحة و تاريخ الطباعة.

بعد ذلك نحدد الكيانات و خصائصها و تكون على النحو التالى :

- DOCTOR (الطبيب) وله الخصائص : Dno (الرقم) وDname (الاسم) و Specialty (التخصص).
- PATIENT (المريض) وله الخصائص : P# (الرقم) و Pname (الاسم) و Room# (رقم الغرفة).
- COMPANY (جهة عمل المريض) و له الخصائص : Wname (اسم جهة العمل) و Waddress (عنوان جهة العمل).
- MEDICINE (الدواء) و له الخصائص : M# (رقم الدواء) و Mname (اسم الدواء).

بعد ذلك علينا البدء فى رسم نموذج ERD كما هو مُبَيَّن فى الشكل ٤-٩. يحتوى التطبيق على الكيانات الأربع، ثم علينا أن نفكّر فى العلاقات الرابطة بينهما. العلاقات الرابطة تكون على النحو التالى :



شكل ٤-٩ : نموذج ERD لتطبيق المستشفى.

- TREAT (يعالج) و هى علاقة رابطة بين الطبيب والمريض. حيث أن الطبيب يشرف على علاج عدة مرضى و يمكن ألا يشرف على علاج مريض بعينه، فإن الكاردينالتى تصبح (0:N) ناحية المريض. أما المريض فغالباً ما يتبع طبيب واحد و واحد فقط، لذلك تصبح الكاردينالتى (1:1) ناحية الطبيب. هذه العلاقة الرابطة نوعها One-to-many.

- WORKS_IN (يعمل فى) و هى علاقة رابطة بين المريض و جهة عمله. حيث أن أى مريض يعمل فى جهة عمل واحدة أو لا يعمل على الإطلاق، فإن الكاردينالتى تصبح (0:1) ناحية جهة العمل. أما جهة العمل فغالباً ما يعمل بها العديد و على الأقل واحد فقط، لذلك تصبح الكاردينالتى (1:N) ناحية المريض. هذه العلاقة الرابطة نوعها one-to-many.

- TAKE (صرف) و هى علاقة رابطة بين المريض و الدواء و لها الخاصية Q (كمية الدواء التى صرفها المريض). حيث أن أى مريض يمكن أن يصرف عدة أنواع من الأدوية و ممكن ألا يصرف أى دواء، أى أن الكاردينالتى تصبح (0:N) من ناحية الدواء. أما الدواء يمكن أن يأخذه عدة مرضى أو لا يأخذه أى مريض، أى أن الكاردينالتى تصبح (0:N) من ناحية المريض. هذه العلاقة الرابطة نوعها many-to-many.

بتحويل نموذج ERD إلى قاعدة بيانات نحصل على العلاقات التالية :

(1) DOCTOR (Dno, Dname, Specialty, Salary)
(2) PATIENT (P#, Pname, Room#)
(3) WORK (Wname, Waddres)
(4) MEDICINE (M#, Mname)
(5) TREAT (P#, Dno)
(6) WORKS_IN (P#, Wname)
(7) TAKE (P#, M#)

بعد ذلك نقوم بإجراء OPTIMZATION أى تحسين قاعدة البيانات فنحصل على قاعدة بيانات تحتوى على العلاقات الخمسة التالية :

(1) ➔ (1) DOCTOR (Dno, Dname, Specialty, Salary)

(2)&(5)&(6) ➔ (2) PATIENT (P#, Pname, Room#, Dno, Wname)

(3) ➔ (3) WORK (Wname, Waddres)

(4) ➔ (4) MEDICINE (M#, Mname)

(7) ➔ (5) TAKE (P#, M#)

٧-٤ تطبيق مصارف الدولة Bank System

ارسم ERD لتصميم قاعدة بيانات لبيانات عملاء مصرف على مستوى الدولة من رقم إثبات الشخصية و نوعها و الاسم و العنوان و كذلك حسابات كل عميل من رقم و رصيد و نوع العملة. كذلك نهتم بالعمليات التى تتم على أى حساب سواء كانت سحباً أم إيداعاً وقيمة المبلغ و رقم و تاريخ العملية و رقم الماكينة أو رقم الصرّاف. علماً بأنه يمكن لأى عميل أن يفتح أى عدد من الحسابات فى نفس فرع المصرف و كذلك الفروع الأخرى كما يجوز أن يكون الحساب مشتركاً بين أكثر من عميل.

كذلك يوجد للمصرف العديد من الفروع و نحتفظ لكل مصرف برقم و اسم المصرف و اسم مدير المصرف أما بالنسبة للفروع فهناك رقم الفرع داخل المصرف و عنوانه و اسم مدير الفرع.

تضم قاعدة البيانات معلومات عن قروض العملاء (رقم القرض و قيمته و تاريخه و نوع العملة) و رقم عملية السداد و القيمة المسددة و تاريخها. علما بأن العميل يحق له الحصول على أى عدد من القروض من كافة الفروع دون علاقة بين ذلك و حساباته إن كان له حسابات. ثم حول ERD إلى Optimized Relations.

الحل

علينا أن نبحث أولاً عن الكيانات و خصائصها. تكون الكيانات بالشكل التالى :

- CUSTOMER (العميل) و له الخصائص التالية : ID# (رقم إثبات الشخصية) و Idtype (نوع إثبات الشخصية) و Name (الاسم) و Address (العنوان).
- ACCOUNT (الحساب الجارى) و له الخصائص التالية : Acc# (رقم الحساب) و Currency (نوع العملة) و Balance (الرصيد).
- APROCESS (عملية على الحساب الجارى) و له الخصائص التالية : AP# (رقم العملية) و Type (نوع العملية) و Adate (تاريخ العملية) و (Avalue) قيمة المبلغ و M# (رقم الماكينة أو الصرّاف).
- LOAN (حساب القرض) وله الخصائص التالية : L# (رقم القرض) و Lvalue (قيمة القرض) و Ldate (تاريخه) و Currency (نوع العملة).
- LPROCESS (عملية على حساب القرض) و له الخصائص التالية : LP# (رقم العملية) و Date (تاريخ العملية) و (Value) قيمة المبلغ.

- BANK (المصرف) و له الخصائص التالية : B# (رقم المصرف) و Bname (عنوان المصرف) و Bmanager (اسم مدير المصرف).
- BRANCH (الفرع) و له الخصائص التالية : Br# (رقم الفرع) و Braddress (عنوان الفرع) و Brmanager (اسم مدير الفرع). لاحظ أن هذا الكيان ضعيف يعتمد على الكيان القوى (BANK) و يتكون مفتاحه الأساسى من رقمه كفرع (Br#) بالإضافة إلى رقم المصرف التابع له (B#).

علينا الآن أن نبحث عن العلاقات الرابطة و خصائصها. فنجد أن العلاقات الرابطة تكون بالشكل التالى :

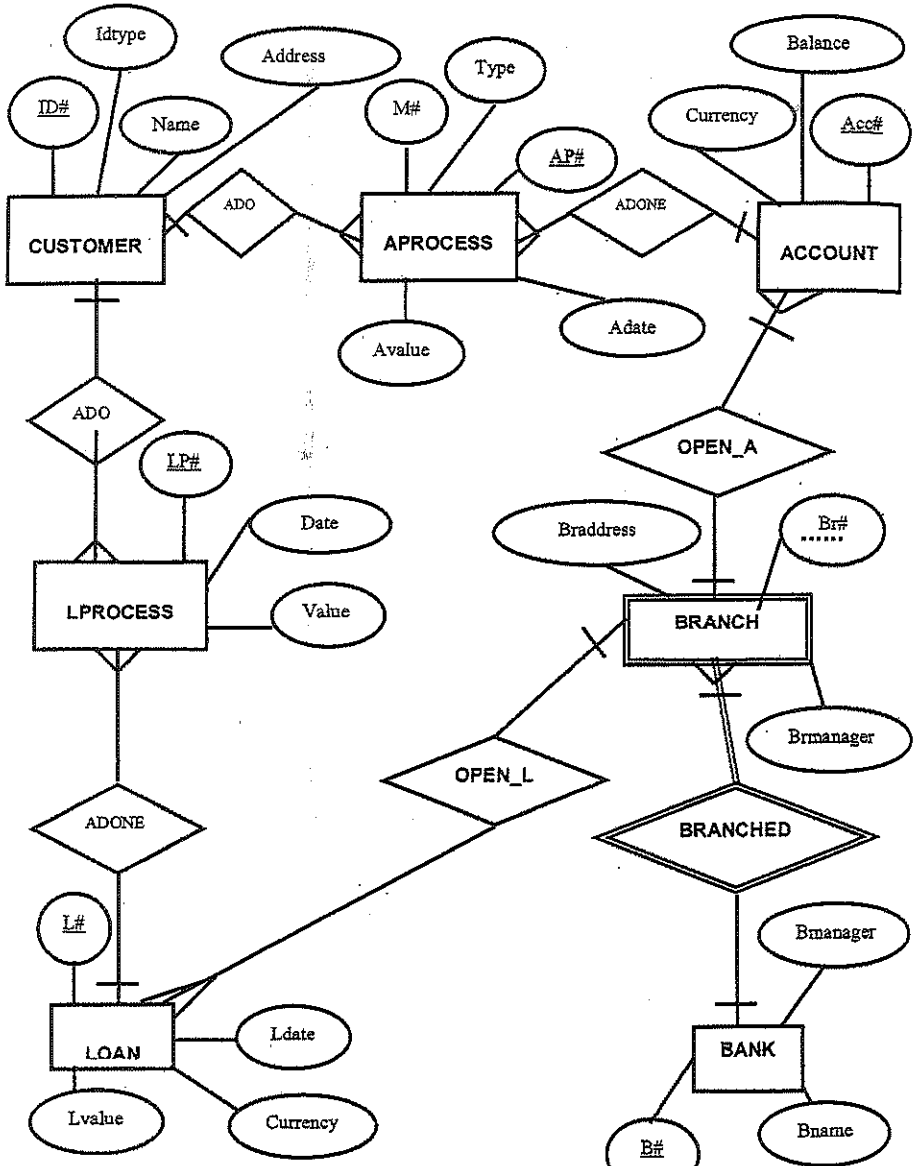
- ADO (يُجرى بعملية على حساب جارى) و تربط بين العميل و العملية. لأنه يمكن للعميل أن يُجرى أكثر من عملية أو لا يجرى أى عملية، تصبح الكاردينالتى (0:N) ناحية العملية. أما العملية فلابد أن يقوم بها عميل وعميل واحد على الأقل، فتصبح الكاردينالتى (1:1) ناحية العميل.
- ADONE (تمت العملية على حساب جارى) و تربط بين العملية و الحساب الجارى. لأنه يمكن إجراء أكثر من عملية على الحساب الجارى أو لا تُجرى أى عملية، تصبح الكاردينالتى (0:N) ناحية العملية. أما العملية فلابد أن تقع على حساب جارى واحد، فتصبح الكاردينالتى (1:1) ناحية العميل. هذه العلاقة الرابطة نوعها one-to-many.
- LDO (يُجرى بعملية على حساب قرض) و تربط بين العميل و العملية. لأنه يمكن للعميل أن يُجرى أكثر من عملية أو لا يجرى أى عملية، تصبح الكاردينالتى (0:N) ناحية العملية. أما العملية فلابد أن يقوم بها عميل وعميل واحد على الأقل، فتصبح الكاردينالتى (1:1) ناحية العميل.
- LDONE (تمت العملية على حساب قرض) و تربط بين العملية و الحساب الجارى. لأنه يمكن إجراء أكثر من عملية على حساب القرض أو

لا تُجرى أى عملية، تصبح الكاردينالتى (0:N) ناحية العملية. أما العملية فلابد أن تقع على حساب قرض واحد، فتصبح الكاردينالتى (1:1) ناحية العميل. هذه العلاقة الرابطة نوعها one-to-many.

- OPEN_A (فتح حساب جارى) و هى علاقة رابطة بين الحساب الجارى للعميل من جهة و فرع المصرف المفتوح فيه الحساب من الجهة الأخرى. نظراً لأنه يمكن فتح عدة حسابات جارية فى فرع واحد و حساب واحد على الأقل، فإن الكاردينالتى تصبح (1:N) ناحية الحساب الجارى. أما بالنسبة للحساب الجارى فلا يمكن فتحه إلا فى فرع مصرف واحد و واحد فقط، لذا تصبح الكاردينالتى (1:1) ناحية فرع المصرف. هذه العلاقة الرابطة نوعها one-to-many.
- OPEN_L (فتح حساب قرض) و هى علاقة رابطة بين حساب قرض العميل من جهة وفرع المصرف المفتوح فيه حساب القرض من الجهة الأخرى. نظراً لأنه يمكن فتح عدة حسابات قروض فى فرع واحد أو ل يفتح أى حساب قرض، فإن الكاردينالتى تصبح (0:N) ناحية حساب القرض. أما بالنسبة لحساب القرض فلا يمكن فتحه إلا فى فرع مصرف واحد و واحد فقط، لذا تصبح الكاردينالتى (1:1) ناحية فرع المصرف. هذه العلاقة الرابطة نوعها one-to-many.
- BRANCHED (متفرع من) و هى علاقة رابطة ضعيفة تربط بين الكيان الضعيف فرع المصرف و الكيان القوى المصرف. نظراً لأن المصرف يتفرع منه غالباً عدة فروع و على الأقل فرع واحد (الفرع الرئيسى)، فإن الكاردينالتى تصبح (1:N) ناحية فرع المصرف. بالنسبة لفرع المصرف فلابد أن يكون متفرعاً من مصرف واحد و واحد فقط، أى أن الكاردينالتى تصبح (1:1) ناحية المصرف. هذه العلاقة الرابطة نوعها one-to-many.

بناءً على ما سبق يكون نموذج ERD للمصرف كما فـي الشـكل ٤-١٠. نقوم بعد ذلك بتحويل نموذج ERD للمصارف إلى قاعدة بيانـات نحصـل علـى القواعد التالية :

(1) CUSTOMER (ID#, Idtype, Name, Address)
(2) ACCOUNT (Acc# , Balance, Currency)
(3) APROCESS(AP#, Type, Adate,Avalue,M#)
(4) LOAN (L# , Lvalue, Ldate, Currency)
(5) LPROCESS (LP#, Date, Value)
(6) BANK (B# , Bname, Bmanager)
(7) BRANCH (B#, Br# , Braddress, Brmanager)
(8) ADO (ID#, AP#)
(9) ADONE(AP#, Acc#)
(10) LDO (ID#, LP#)
(11) LDONE (LP#, L#)
(12) OPEN_A (Acc#, B#, Br#)
(13) OPEN_L (L#, B#, Br#)



شكل ٤-١٠ : نموذج ERD لتطبيق مصارف الدولة.

نستطيع أن نستخدم أسلوب تحسين قواعد البيانات لنحصل على قاعدة بيانات مُحَسَّنة تحتوى على القواعد التالية :

(1) ➔(1) CUSTOMER (ID#, Idtype, Name, Address)

(2)&(12)➔(2) OPEN_ACCOUNT (Acc# , Balance, Currency,B#,Br#)

(3)&(8)&(9)➔(3)ADOPROCESS(AP#,Type,Adate,Avalue,M#,ID#,Acc#)

(4)&(13)➔ (4) OPEN_LOAN (L#,Lvalue,Ldate,Currency, B#, Br#)

(5)&(10)&(11) ➔(5) LPROCESS (LP#, Date,Value, ID#, L#)

(6) ➔ (6) BANK (B# , Bname, Bmanager)

(7) ➔ (7) BRANCH (B#, Br# , Braddress, Brmanager)

٨-٤ أسئلة

١. مطلوب تصميم قاعدة بيانات(ERD) للكلية بحيث تهتم ببيانات الطلبة من رقم جامعي و اسم و عنوان و تخصص و بيانات المقررات الدراسية من رقم المادة و اسمها و عدد الساعات المعتمدة والشعب الدراسية فى كل مقرر من رقم و عدد الطلبة المسجلون فيها و رقم قاعة المحاضرات التى تتواجد فيها كل شعبة و الفصل و العام الدراسي الذي تم فيه التسجيل و كذلك نتيجة الامتحان و تهتم قاعدة البيانات أيضا بأعضاء هيئة التدريس من رقم و اسم و وظيفة و القسم التابع له و كذلك أسماء و أماكن الأقسام داخل الكلية و الفصل الدراسي الذي يقوم عضو هيئة التدريس فيه لشعب مقرر معين. كذلك تسجل قاعدة البيانات الإرشاد الأكاديمى من قِبَل عضو هيئة التدريس للطلبة.

٢. مطلوب تصميم قاعدة بيانات لتمثيل بيانات الحاجزين –رقم و اسم وتليفون – على رحلة طيران التى تقوم فى موعد ما فى تاريخ محدد من مطار القيام إلى مطار الوصول مع بيان أن الحجز يكون مؤكدا إذا سجل الراكب رقم التذكرة فى حجزه المشتمل على –رقم الحجز وتاريخ الحجز وآخر موعد للتذاكر– وكذلك بيانات الطائرة التى ستقوم بالرحلة –رقم و موديل وتاريخ الصنع والشركة المالكة وعنوانها ورقم تليفونها و المدير المسئول وعدد المقاعد واسم قائد الطائرة.

٣. ارسم ERD و حوله إلى Optimized Database للجدول التالى :

Date :25/11/1423 Department : Computer Science

S#	SName	Course#	Cname	CLab#	Labhours	Package
417	Sayfaullah	325	DBase1	L325	2	Access
		363	AI	L363	2	Java
		426	Dbase2	L426	3	Oracle

٤. مطلوب تصميم قاعدة بيانات لسوق الأوراق المالية خاصة بعملية تداول الأسهم. حيث يتم التعامل على أسهم عدة شركات كل سهم له كود و اسم الشركة وعنوانها و المدير المسئول و هاتفها و السعر الأسمى و تاريخ القيد و عدد الأسهم المقيدة فى سوق الأوراق المالية و السعر الحالى وسعر الفتح فى بداية اليوم و سعر الغلق فى نهاية اليوم و العملة المقيد بها السهم.

كذلك يتم تسجيل بيانات المستثمر الذى يمكن أن يمتلك أى عدد من أى نوع من الأسهم مع تسجيل بيانات كافة عمليات الشراء التى قام بها من سعر شرائه للسهم و تاريخ الشراء و عدد الأسهم المشتراة وشركة الوساطة (السمسرة) التى قامت بتنفيذ عملية الشراء. كذلك بيانات عمليات البيع التى قام بها من سعر بيعه للسهم و تاريخ البيع و عدد الأسهم

المباعة و شركة الوساطة التى قامت بتنفيذ عملية البيع إلى جانب بيانات المستثمر الأساسية من رقم إثبات الشخصية و نوعها و اسمه وهاتفه.

٥. ارسم ERD اللازم لقاعدة بيانات لتخزين البيانات الخاصة بعملية إصدار فاتورة لأحد العملاء وهذه الفاتورة تصدر بناء على أمر الشراء الذى يرسله العميل ويهتم فى هذه العملية برقم العميل واسمه وعنوانه وأرقام أوامر الشراء وتاريخ إصدارها وكذلك الكمية المطلوبة من العميل فى أمر الشراء من كل صنف وكذلك الكمية الموردة فعلا ورقم كل صنف واسمه و سعر الوحدة منه والكمية الموجودة فى المخزن من كل صنف. أيضاً يهتم البرنامج برقم الفاتورة وتاريخها وطريقة الدفع (آجل/نقدا). كما تهتم قاعدة البيانات بالمصنع الذى أنتج الصنف و عنوانه و هاتفه. ثم حول ERD إلى Optimized Relations.

٦. ارسم ERD للتقرير التالى و احصل منه على قاعدة بيانات مُحَسَّنة.

Buraidah Hospital Doctor Report

Date :25/11/1423 Specialty : Heart Medicin

D#	Doctor	Dphone	Patient-code	Patient Name	Procedure	Charge
10	Khaled	332323	455a	K. Elboghdady	X_ray	56.88
			88g	L. Khalil	Examination	787

٧. ارسم ERD واحصل منه على Optimized relations للجدول التالى:

(بـــفرض أن أى مدرب يدرب مركز لعب واحد)

Player اللاعب	Position مركز اللعب	Coach المدرب
Ali	FB	Majed
Sayed	G	Fahd
Khaled	FB	Peter
John	T	Nasser
Walid	FB	Majed

٨. مطلوب تصميم قاعدة بيانات تهتم ببيانات الفرق والمباريات الرياضية فـــى دورى كرة القدم. الفريق به عدد من اللاعبين، لا يشاركون جميعا فى كـــل مباراة. المطلوب أن نهتم باللاعبين المشاركين فى كل مباراة كـــل فريـــق، مركز لعبه فى المباراة ونتيجة المباراة. حاول تصميم مخطـــط ER لهـــذا التطبيق، بادئاً بالافتراضات التى وضعتها.

الفصل الخامس

تطبيع قواعد البيانات

Databases Normalization

درسنا فى الفصلين الثالث و الرابع كيفية تصميم قاعدة بيانات عن طريق رسم نموذج ERD من تحليل وصفى للتطبيق أو تحليل مستندات التطبيق. ثم بعد ذلك حوّلنا الرسم إلى علاقات قاعدة البيانات و منها إلى علاقات **قاعدة بيانات مُحَسَّنة (Optimized Database)**. فى هذا الفصل نتعلم أسلوب مختلف للحصول على قاعدة البيانات المُحَسَّنة مباشرة. هذا الأسلوب نسميه تطبيع البيانات الذى يُزيل عيوب البيانات لتصبح فى شكل طبيعى معين.

٥-١ مشاكل البيانات Data Anomalies

بعيداً عن الكيانات و العلاقات الرابطة بينهما و خصائص الجميع، سننظر هنا إلى بيانات تطبيق معين كوحدة واحدة أى علاقة عامة واحدة تضم كافة خصائص أو حقول التطبيق و تُسَمَّى **علاقة عامة (General Relation)**. ثم نتعرَّف على المشاكل التى تنشأ عند استخدام هذه العلاقة سواء عند الإضافة أو الاستخراج أو التعديل أو الحذف.

سنناقش هذه المشاكل من خلال العلاقة STUDENT_REPORT التى نراها فى الشكل ٥-١. حيث تتضمن هذه العلاقة خصائص من وثيقة تخرج طالب. و نعتبر هذه العلاقة فى شكل غير طبيعى أى أن استخدامها فى قواعد البيانات غير ملائم و يسبب مشاكل كثير، لذا فهى غير طبيعية.

علاقة تقرير الطالب **STUDENT_REPORT**

رقم	طالب	عنوان	رمزه	مقرر	ساعات	رقمه	مدرس	عمل	تقدير
St#	Name	Address	Crs#	Title	Hours	Inst#	Iname	Work	Grade
5	Fahd	Qassim	cs101	Intro	3	7	Ali	CS	A
5	Fahd	Qassim	cs102	C	4	12	Gamil	math	B
5	Fahd	Qassim	cs103	C++	4	2	Rashid	CS	C+
5	Fahd	Qassim	cs325	DB1	3	2	Khaled	CS	B+
5	Fahd	Qassim	cs426	DB2	3	3	Tarek	CS	B

شكل ٥-١ : علاقة عامة غير طبيعية.

٥-١-١ تكرار البيانات Data Redundancy

نستطيع بمجرد النظر إلى العلاقة STUDENT_REPORT أن هناك بيانات متكررة كثيرة و هى :

- بيانات الطالب يتكرر تخزينها مع تسجيله لكل مقرر.
- تتكرر بيانات كل مقرر مع كل طالب يُسَجِّل ذلك المقرر.
- تتكرر بيانات كل مدرس مع كل مقرر يُسجله طالب ما.

هذا التكرار يسبب مشاكل عديدة إلى جانب استهلاك حيز التخزين و وقت إدخال البيانات و وقت القائمين على إدخال البيانات. أيضاً يؤثر ذلك على سرعة معالجة البيانات و استهلاك الأجهزة. نتعرف على المشاكل فى الأجزاء التالية.

٥-١-٢ مشاكل إدخال البيانات Insertion Anomalies

تتعدد مشاكل إدخال البيانات فى علاقة مثل STUDENT_REPORT. نذكر فيما يلى أمثلة من هذه المشاكل :

- عندما نريد إدخال بيانات مقرر جديد لم يُسجِّله أى طالب. لا نستطيع إدخال بيانات أى مقرر لم يُسجِّله طالب واحد على الأقل.

- عندما نرغب فى إدخال بيانات عضو هيئة تدريس جديد لم يقم بعد بالتدريس أو يرتبط عمله بالأبحاث العلمية و لا يباشر تدريس أى مقررات. لا يمكنا إدخال بيانات مدرس لم يدرِّس مقرر درسه طالب واحد على الأقل.

- عند إدخال بيانات أى طالب جديد أو حذف الفصل الدراسى و لم يُسجِّل بعد أى مقررات دراسية من الخطة، نضطر أن نترك فراغاً فى الحقول الخاصة بالمقررات و بعضو هيئة التدريس.

- نظراً لإدخال بيانات معينة أكثر من مرة، من الممكن أن يحدث إدخال خطأ لبعض أو كل هذه البيانات فى موضع من مواضع الإدخال سواء كان ذلك عمداً أو سهواً.

٥-١-٣ مشاكل حذف البيانات Deletion Anomalies

عملية حذف البيانات يتخللها و يَنجُم عنها مشاكل عديدة ترتبط بعملية الإدخال و مشاكلها و نوضحها فيما يلى :

- عند حذف بيانات طالبة/طالب (أو طلبة) ما و هى الطالبة الوحيدة (أو الطلبة) المُسجِّلة لمقرر معين، يتم حذف بيانات هذا المقرر و نفقد بياناته.
- عند حذف بيانات مدرسة/مدرس (أو مدرسين) تدرس مقرر (أو مقررات) معين و تُسجِّله طالب أو طالب واحد يتم حذف بيانات المقرر و الطالبة و نفقد بياناتهما.
- عند نحذف سجل طالبة/طالب أو مقرر أو مدرس سوف يكون بالتأكيد لدينا سجلات أخرى ربما ننسى أو لا نستطيع حذفها.

٥-١-٤ مشاكل التعديل Updating Anomalies

أيضاً تتضمن عملية تعديل البيانات و ينجم عنها مشاكل عديدة منها :

- عند تعديل بيانات مقرر أو طالب أو مدرس، يجب أن نجرى نفس التعديلات فى كافة مواضع تخزين تلك البيانات.
- ربما يحث خطأ فى تعديل البيانات فى موضع تخزين معين دون مواضع أخرى سهواً أو عمداً. نتيجة ذلك تصبح غير متوافقة.

٥-٢ التطبيع و الاعتماد الوظيفى للبيانات

Normalization and Functional Dependency

تطبيع البيانات (Normalization) هو عملية تفكيك (تحليل / تجزئ) تراكيب البيانات الكبيرة (الموجودة فى علاقة واحدة عامة) إلى تمثيل بسيط (عدة علاقات بسيط) طبقاً لاعتما د البيانات على بعضها البعض.

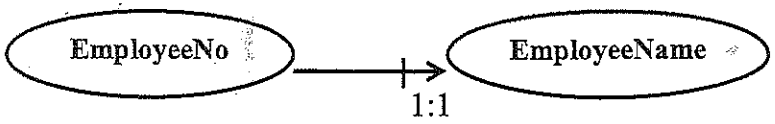
الاعتماد الوظيفى للبيانات (Data Functional Dependency) هو قيد اعتماد/ارتباط خاصية أو مجموعة من الخصائص على خاصية أو مجموعة أخرى من خصائص قاعدة البيانات. و هو صفة نفهمها من معنى أو مدلول الخصائص التى تشكِّل قاعدة البيانات تفيد بأن وجود خصائص يرتبط بوجود خصائص أخرى.

كما درسنا فى الفصل الثانى فى ٢-٣-٢، الروابط بين عناصر البيانات و أن هناك رابطة مفردة بين عنصريين (أى أن كل عنصر يقابله عنصر) مثل رقم الموظف و اسم الموظف كما فى الشكل ٥-٢(أ) و رقم الموظف و رقم القسم كما فى الشكل ٥-٢(ب). لقد لاحظنا أن نسب المشاركة (الكاردينالتى) التى تجعل المشاركة فى الرابطة المفردة تكون (1:1) فى الحالة الأولى أو (1:0) فى الحالة الثانية.

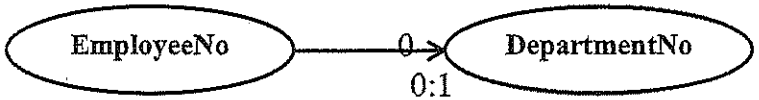
درسنا أيضاً فى نفس الفصل الرابطة المتعددة بين عنصريين (أى أن كل عنصر يقابله عدة عناصر من العنصر الثانى) مثل رقم الغرفة و رقم السرير كما فى الشكل ٥-٢(ج) أو رقم المدرس و رقم المقرر فى الشكل ٥-٢(د). لقد لاحظنا أن نسب المشاركة (الكاردينالتى) التى تجعل المشاركة فى الرابطة المتعددة تكون (0:N) فى الحالة الأولى أو (1:N) فى الحالة الثانية.

إذا تأملنا علاقة عامة مثل العلاقة STUDENT_REPORT التى مررنا عليها فى الشكل ٥-١، لوجدنا عدة أنواع لاعتماد البيانات على بعضها البعض. فيما يلى نتعرَّف على أهمها.

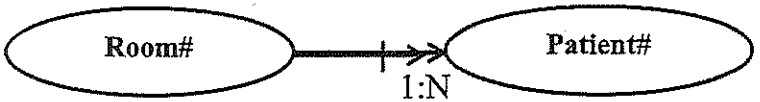
(أ)



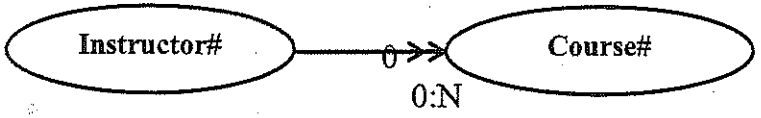
(ب)



(ج)



(د)



شكل ٥-٢ : أنواع الروابط بين عناصر البيانات.

٥-٢-١ الاعتماد الجزئى Partial Dependence

الاعتماد الجزئى للبيانات يُقْصَد به اعتماد خاصية ما أو مجموعة من الخصائص على خاصية أو مجموعة أخرى من الخصائص فى نفس الوقت الذى تعتمد فيه خاصية أخرى أو مجموعة ثانية على خاصية أو مجموعة مختلفة من الخصائص.

على سبيل المثال إذا طبّقنا ذلك على العلاقة العامة STUDENT_REPORT لوجدنا مثلاً أن الخصائص (المقرر و عدد ساعاته و رقم المدرس) تعتمد على (رمز المقرر) فى الوقت الذى يعتمد (التقدير) على كل (رمز المقرر و رقم الطالب) معاً.

٥-٢-٢ الاعتماد الانتقالى Transitive Dependence

الاعتماد الانتقالى للبيانات يُقْصَد به اعتماد خاصية ما أو مجموعة ما من الخصائص على خاصية ما، A مثلاً، فى نفس الوقت الذى تعتمد فيه الخاصية A على خاصية أو مجموعة خصائص أخرى.

على سبيل المثال إذا طبّقنا ذلك على العلاقة العامة STUDENT_REPORT لوجدنا أن (اسم المدرس و القسم الذى يعمل به) يعتمدان على (رقم المدرس) فى الوقت الذى يعتمد فيه (رقم المدرس) على (رمز المقرر).

٥-٣ أشكال تطبيع البيانات (1NF, 2NF,3NF)

Data Normalization Forms (1NF, 2NF, 3NF)

ندرس هنا كيفية التعرف على عيوب الموجودة فى بيانات علاقة عامة مثل STUDENT_REPORT ثم كيف نزيل عيوب البيانات بطريقة منتظمة و

مرحلية. بهذا الأسلوب نستطيع أن نحصل كنتيجة لكل مرحلة على شكل تطبيع معين إلى أن نصل مستوى تطبيع متقدم.

٥-٣-١ شكل التطبيع الأول First Normal Form (1NF)

أول مراحل التطبيع نبدأ بالبحث عن أول و أهم عيوب البيانات فى العلاقة العامة و هى **وجود مجموعات بيانات متكررة (Repeated Group)**. فإن وجدنا مجموعات بيانات متكررة، نحدد للقاعدة العامة درجة أو شكل تطبيعها و هى **درجة التطبيع صفر (Zero Normal Form (0NF))** أو نقول أنها **غير طبيعية (UnNormalized)**. المطلوب فى هذه الحالة هو إزالة عيب تكرار البيانات.

بعد إزالة عيب مجموعات تكرار البيانات أو عند عدم وجوده تكون درجة التطبيع للعلاقة هى شكل تطبيع أول (First Normal Form (1NF)) أو ربما شكل تطبيع ثانى (Second Normal Form (2NF)) أو ربما شكل تطبيع ثالث (Third Normal Form (3NF)) أو شكل تطبيع آخر. يتوقف ذلك وجود باقى العيوب التى سوف نتعرَّف عليها تباعاً من خلال دراسة اعتماد البيانات.

لنرى الآن كيفية نطبق المفاهيم السابقة على العلاقة العامة STUDENT_REPORT لنحدد درجة تطبيعها و نحسّنها إن أمكن. أولاً نتأمل العلاقة STUDENT_REPORT التى تجمع كافة الخصائص كما يلى:

علاقة تقرير الطالب **STUDENT_REPORT**

تقدير	عمل	مدرس	رقمه	ساعات	مقرر	رمزه	عنوان	طالب	رقم
Grade	Work	Iname	Inst#	Hours	Title	Crs#	Address	Name	St#

كما سبق أن أشرنا و بمجرد النظر إلى الجدول الذى يحوى أمثلة لبيانات العلاقة العامة STUDENT_REPORT نجد هنا مجموعة بيانات متكررة تضم كلاً من St#(رقم الطالب) و Name (اسم الطالب) و Address (عنوان الطالب). لذلك فإن درجة تطبيع العلاقة العامة STUDENT_REPORT هى 0NF أو أنها غير طبيعية (UnNormalized)، و علينا إزالة هذا العيب.

قبل التعرُّف على كيفية إزالة عيب مجموعات التكرار نسأل أنفسنا سؤالاً هاماً و هو هل أو بالأحرى كيف يمكننا التعرُّف على مجموعات التكرار فى علاقة دون وجود بيانات؟ الإجابة هى نعم يمكننا التعرُّف على وجود مجموعات التكرار من دون وجود بيانات. أما كيف فهذا يتم بأن نجرِّب بأنفسنا إدخال سجلات فى هذه العلاقة لنفس العنصر الرئيسى بها و هو هنا الطالب لنحدد وجود مجموعات تكرار أم لا.

لكى نزيل مجموعة البيانات المتكررة نقوم بفصل خصائص تلك المجموعة فى علاقة جديدة خاصة بها مع الاحتفاظ بالحقل الجَوْهَرى (المفتاح الرئيسى) مع ما يتبقى من خصائص فى العلاقة العامة. نحصل من جرّاء هذا على العلاقتين اللتين نراهما فى الشكل ٥-٣.

3NF ⟵ طالب (1)STUDENT

عنوان	طالب	رقم
Address	Name	St#

تقرير الطالب **STUDENT_REPORT**

تقدير	عمل	مدرس	رقمه	ساعات	مقرر	رمزه	رقم
Grade	Work	Iname	Inst#	Hours	Title	Crs#	St#

شكل ٥-٣ نتيجة إزالة مجموعة التكرار.

العلاقتين السابقتين STUDENT و STUDENT_REPORT ممكن أن تكون درجة تطبيع أياً منهم 1NF أو 2NF أو 3NF أو غير ذلك. هذا ما سوف نعرفه عند تفحُّصها فى الأجزاء التالية بحثاً عن عيوب البيانات الأخرى.

٥-٣-٢ شكل التطبيع الثانى Second Normal Form (2NF)

مراحل التطبيع الباقية تدرس و تهتم باعتماد البيانات على بعضها البعض. فى ثانى مرحلة تطبيع نبدأ بالبحث فى جميع العلاقات الموجودة لدينا عن ثانى عيب من عيوب البيانات و هو **الاعتماد الجزئى للبيانات (Partial Dependency of Data)**. فإن وجدنا اعتماداً جزئياً فى علاقةٍ ما تكون درجة أو شكل تطبيعها هى **شكل التطبيع الأول** أو **درجة التطبيع** "1" (First Normal Form(1NF)). المطلوب فى هذه الحالة هو إزالة مشكلة الاعتماد الجزئى فى البيانات.

بعد إزالة عيب الاعتماد الجزئى للبيانات أو عند عدم وجوده فى الأساس تكون درجة التطبيع للعلاقة هى شكل التطبيع الثانى (Second Normal Form (2NF)) أو ربما شكل التطبيع الثالث (Third Normal Form (3NF)) أو شكل تطبيع آخر. يتوقف ذلك وجود باقى العيوب التى سوف نتعرّف عليها تباعاً.

قبل تطبيق المفاهيم السابقة نذكّر الدارس أن الاعتماد الجزئى للبيانات هو اعتماد خاصية أو مجموعة خصائص على خاصية ما أو مجموعة أخرى من الخصائص فى نفس الوقت الذى تعتمد فيه الخاصية أو مجموعة الخصائص الثانية على خاصية أو مجموعة مختلفة من الخصائص.

أى أنه لكى تكون درجة تطبيع علاقة ما 2NF يجب أن تكون ألا تحتوى تلك العلاقة على مجموعة تكرار و أن تعتمد خصائص هذه العلاقة اعتماداً كلياً على مفتاحٍ (خاصية) ما.

بالنسبة للعلاقة الأولى STUDENT تحتوى ثلاث خصائص، تعتمد خاصية Name و كذلك خاصية Address على المفتاح St#. لذلك فهى تخلو من مشكلة الاعتماد الجزئى، لذا فدرجة تطبيعها 2NF أو 3NF أو غير ذلك. كذلك مشكلة الاعتماد الانتقالى غير موجودة لعدم اعتماد المفتاح St# على أى خاصية أخرى. بناءً على ما سبق فإن العلاقة STUDENT درجة تطبيعها 3NF. و نطلق عليها مجازاً علاقة طبيعية أو Normalized. تأخذ هذه العلاقة الرقم ١.

أما العلاقة STUDENT_REPORT فنجد أن مشكلة الاعتماد الجزئى موجودة. حيث أن Grade (تقدير الطالب فى مقرر معين) تعتمد على كلٍ من الخاصيتين St# و Crs#. بينما باقى الخصائص تعتمد على (ترتبط) رقم المقرر Crs#. لإزالة مشكلة الاعتماد الجزئى تلك نفصل كل مجموعة جزئية مع ما تعتمد عليه و يُصبِّح مفتاحاً لها، فنحصل من العلاقة STUDENT_REPORT على العلاقتين اللتين نراهما فى الشكل ٥-٤.

STUDENT_COURSE و STUDENT_INSTRUCTOR علاقتين يمكن أن تكون درجة تطبيع كل منهما 2NF أو 3NF أو غير ذلك على حسب وجود مشكلة الاعتماد الانتقالى فى أيٍ منهما من عدمه. هذا سنعرفه فى الجزء التالى.

الطالب و المقرر
(2)STUDENT_COURSE ← 3NF

رقم	رمزه	تقدير
St#	Crs#	Grade

المقرر و المدرس
COURSE_INSTRUCTOR

رمزه	مقرر	ساعات	رقمه	مدرس	عمل
Crs#	Title	Hours	Inst#	Iname	Work

شكل ٥-٤ : نتيجة إزالة الاعتماد الجزئى.

٥-٣-٣ شكل التطبيع الثالث Third Normal Form (3NF)

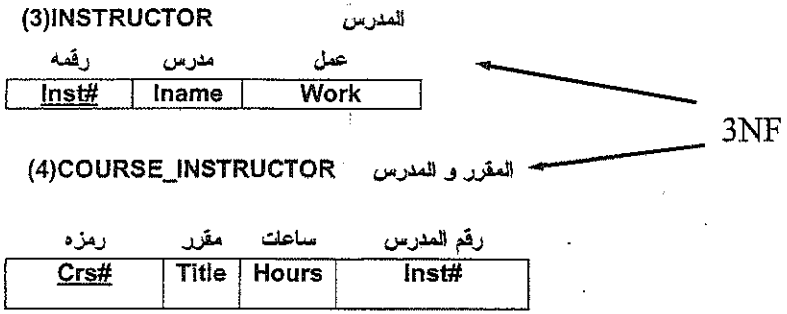
المرحلة الثالثة من مراحل التطبيع تهتم بالبحث عن **مشكلة تواجد الاعتماد الانتقالى (Transitive Dependence)**. فإن وجدنا اعتماد جزئى فى علاقةٍ ما فإن شكل تطبيعها أو درجة تطبيعها "٢" **(Second Normal Form (2NF))**. المطلوب فى هذه الحالة هو إزالة عيب الاعتماد الانتقالى للبيانات.

بعد إزالة عيب الاعتماد الانتقالى للبيانات أو عند عدم وجوده أصلاً تكون درجة التطبيع للعلاقة هى شكل التطبيع الثالث (Third Normal Form (3NF)) أو شكل تطبيع آخر أعلى من ذلك. يتوقف ذلك وجود باقى العيوب التى سوف نتعرَّف عليها تباعاً. نطلق مجازاً على العلاقة التى درجة تطبيعها 3NF علاقة طبيعية (Normalized).

أى أنه لكى تكون درجة تطبيع علاقة ما 3NF يجب أن تكون ألا تحتوى تلك العلاقة على مجموعة تكرار و أن تعتمد خصائص هذه العلاقة اعتماداً كلياً على مفتاحٍ (خاصية) ما و لا يوجد أى اعتماد انتقالى بين خصائص تلك العلاقة.

بالنظر إلى العلاقة STUDENT_COURSE نجد أن الخاصية Grade يعتمد على كلٍ من St# و Crs# كمفتاح، دون وجود أى اعتماد للمفتاح على خاصية أخرى. لذلك فإن درجة تطبيع العلاقة STUDENT_COURSE هى "٣" أى 3NF أو طبيعية (Normalized) و تأخذ الرقم ٢.

أما العلاقة COURSE_INSTRUCTOR فنجد أن بها الخاصيتين Work (عمل المدرس) و Iname (اسم المدرس) يعتمدان على Inst# (رقم المدرس)، بينما يرتبط رقم المدرس برقم المقرر Crs#. أى أن العلاقة تحتوى على مشكلة الاعتماد الانتقالى. لإزالة مشكلة الاعتماد الانتقالى نفصل كل مجموعة مع ما ترتبط به أو تعتمد عليه فنحصل على العلاقتين اللتين فى الشكل ٥-٥.



شكل ٥-٥ : نتيجة إزالة الإعتماد الانتقالى.

بعد هذه المراحل الثلاث حصلنا على علاقات طبيعية لقاعدة البيانات لا تتضمن مشاكل البيانات التى ذكرناها فيما سبق. قاعدة البيانات التالية نهائية و مُحسَّنة آلياً و لا تحتاج إلى أى تحسينات (لاحظ المفتاح الخارجى Inst#) :

(1) STUDENT (St#, Name, Address)
(2) STUDENT_COURSE (St#, Crs#, Grade)
(3) INSTRUCTOR (Ints#, Iname, Work)
(4) COURSE_INSTRUCTOR (Crs#, Title, Hours, Inst#)

٥-٤ شكل تطبيع بويس و كودد (BCNF)

Boyce-Codd Normal Form

درجات التطبيع الثلاثة السابق تعريفها (1NF, 2NF, 3NF) هى أكثر أشكال التطبيع استخداماً. ويوجد شكل تطبيع جديد هو شكل تطبيع بويس و كودد-Boyce Codd Normal Form (BCNF)) لإزالة أى شذوذ فى علاقة من شكل التطبيع الثالث 3NF. و يُعتَبَر شكل التطبيع BCNF أكثر تقدماً و شمولية عن 3NF، حيث أننا نجد أن أى علاقة درجة تطبيعها BCNF تكون درجة تطبيعها 3NF أيضاً و العكس غير صحيح.

نستطيع القول أن درجة تطبيع علاقة ما BCNF إذا كان كل مُحَدِّد فى العلاقة مفتاحاً مرشَّحاً. افترض مثلاً أن لدينا العلاقة ITEM الخاص بالأصناف المعروضة فى متجر و الموجودة فى الشكل ٥-٦(أ). هذه العلاقة ذات شكل تطبيع ثالث. و لكننا نرى أن العملة تعتمد على البلد المنتج للصنف. حيث تختلف العملات من بلد لآخر من حيث العملة الرئيسية و العملات الصغيرة. لذلك يتم عمل علاقتين ITEM و CURRENCY كما فى الشكل ٥-٦(ب). كلا العلاقتين في شكل تطبيع BCNF.

(أ)

رقم الصنف Item#	وصف الصنف Description	البلد المنتج Producing_country	العملة Currency	سعر Price

(ب)

Producing_country	Currency

Item#	Description	Producing_country	Price

شكل ٥-٦ : (أ) علاقة ITEM فى شكل تطبيع 3NF.
(ب) العلاقتين ITEM و CURRENCY فى شكل تطبيع BCNF.

٥-٥ شكل التطبيع الرابع

Forth Normal Form (4NF)

شكل التطبيع الرابع Fourth Normal Form (4NF) مبنى على تواجد الاعتماد متعدد القيم (Multivalued Dependency). الاعتماد متعدد القيم يمنع وجود عدة قيم لخاصيتين أو أكثر من خصائص العلاقة. فعند وجود خاصية متعددة القيم فى علاقةٍ ما نحتاج إلى تمثيل جميع القيم التى تمثلها.

على سبيل المثال العلاقة EMPLOYEE الموجودة فى الشكل ٥-٧(أ)، درجة تطبيعها 3NF و لكنها تحتوى على خاصيتين متعددتى القيم و هما Project# (رقم المشروع) و Car# (رقم السيارة)، حيث أنه من الممكن أن يعمل الموظف بأكثر من مشروع، و من الممكن أيضاً أن يمتلك الموظف أكثر من سيارة. لكى نحل المشكلة الموجودة فى العلاقة EMPLOYEE نقسم هذه العلاقة إلى علاقتين كما ترى فى الشكل ٥-٧(ب).

بعد دراستنا لأشكال التطبيع المختلفة نستطيع أن نقول أن أشكال التطبيع الثلاثة الأولى 1NF و 2NF و 3NF هى الأكثر تواجداً و استخداماً بينما شكلى التطبيع BCNF و 4NF و شكل تطبيع آخر هو 5NF فإنهم نادرين و قلّما نستخدمهما.

EMPLOYEE موظف

اسم الموظف EmpName	المشروع Project	رقم السيارة Cart#
Ali	P1	Kna545
Ali	P2	XYZ323
Ali	P1	XYZ323
Ali	P2	Kna545

شكل ٥-٧ (أ) : العلاقة EMPLOYEE التى تحتوى على خاصيتين متعددتى القيم.

EMP_PROJ موظف-مشروع

رقم الموظف EmpName	المشروع Project
123	P1
123	P2

MP_CAR موظف - سيارة

اسم الموظف EmpName	رقم السيارة Cart#
123	Kna545
123	XYZ323

شكل ٥-٧ (ب) : نتيجة تنفيذ تطبيع 4NF على العلاقة EMPLOYEE التى تحتوى على خاصيتين متعددتى القيم.

٥-٦ تطبيق توريد و تصنيع المنتجات

حدد أولاً درجة تطبيع العلاقة العامة التالية ثم قم بالخطوات اللازمة لتصبح طبيعية.

CUSTOMER_REPORT

رقم العميل Cust-No	اسم العميل Cust-Name	العنوان Cust-address	الهاتف Cust-Phone	رقم الصنف Item-No	وصف الصنف Description	سعر الوحدة Price	الكمية المباعة Quantity	المصنع المنتج Produce Factory
101	Walee d	mail	0020	2201 2345 2202	Sugar Oil 42lit sweet	14 130 40	100 350 950	Elhenaky Elsafy Tusaly

الحل

نظراً لوجود مجموعات متكررة فإن درجة التطبيع 0NF أى غير طبيعية (Unnormalized). نطبّق مراحل التطبيع الثلاث كما يلى :

- **أولاً : إزالة المجموعات المتكررة**

بمجرد النظر إلى العلاقة المعطاة CUSTOMER_REPORT نجد أن الخصائص (Cust_no,Cust_Name,Cust_Phone, Cust_phone) تشكّل مجموعة تكرار يجب إزالتها وفصلها فى علاقة منفصلة مع الاحتفاظ بالمفتاح Cust_No مع باقى خصائص العلاقة، كما يلى :

(1) CUSTOMER

Cust_ No	Cust_ Name	Cust_ Addres s	Cust_ Phone

← 3NF

CUSTOMER_ITEM

Cust_ NO	Item_ No	Description	Price	Quantity	Produce_ Factory

- **ثانياً : إزالة الاعتماد الجزئى**

الآن نتأمل العلاقة CUSTOMER فنجد أن الخصائص الثلاث الأخيرة تعتمد اعتماداً كلياً على الخاصية الأولى Cust_No، لذلك لا يوجد اعتماد جزئى. أى أن درجة تطبيعها ربما تكون 2NF أو 3NF و ليست 1NF.

الآن نتأمل العلاقة CUSTOMER_ITEM فنجد أن الكمية المباعة من كل صنف تعتمد على رقم العميل (Cust_No) و رقم الصنف (Item_NO). كما أن الخصائص Decription و Price و Produc_Factory تعتمد على Item_No. هذا يُشكِّل مشكلة اعتماد جزئى يجب حلها بفصل كل مجموعة مع ما تعتمد عليه (مفتاحها) فنحصل على العلاقتين التاليتين :

(2) CUSTOMER_ITEM

Cust_No	Item_No	Quantity

← 3NF

(3) ITEM_FACTORY

Item_No	Description	Price	Produce_Factory

- **ثالثاً : إزالة الاعتماد الإنتقالى**

الآن علينا أن نفحص العلاقات الثلاث أرقام ١ و ٢ و ٣ لنبحث عن وجود أى اعتماد انتقالى. لا يمكن العثور على أي اعتماد انتقالى كما ترى فى العلاقات الثلاثة. لذلك فإن العلاقات الثلاث التالية درجة تطبيعها 3NF و تشكِّل علاقات طبيعية لقاعدة البيانات.

(1)CUSTOMER (Cust_No, Cust_Name, Cust_Address, Cust_Phone)

(2)CUSTOMER_ITME(Cust_No, Item_NO, Quantity)

(3)ITEM_FACTORY(Item_No,Description,Price,Produce_Factory)

٥-٧ تطبيق تصنيع الأصناف

حدد أولاً درجة تطبيع العلاقة العامة التالية ثم قم بالخطوات اللازمة لتصبح طبيعية. (ملحوظة: هذا التطبيق جزء من تطبيق ٥-٦ بعد تعديل عليك إكتشافه).

ITEM_FACTORY

رقم الصنف	وصف الصنف	سعر الوحدة	الكمية المورَّدة	المصنع المنتج	عنوان المصنع
Item -No	Description	Price	Supplied Quantity	Producing Factory	Factory Address
2201	Sugar	14	100	Elhenaky	ElMadina
2345	Oil 42lit	130	350	Elsafy	ElQassim
2202	sweet	40	950	Tusaly	ELKahera

الحل

بتحليل العلاقة ITEM_FACTORY لا نجد مجموعات متكررة. لذلك فإن درجة التطبيع أعلى من 0NF أى ربما تكون 1NF أو 2NF أو 3NF. نطبّق مرحلتين التطبيع للبحث عن الاعتماد الجزئى و الانتقالى :

- **أولاً : إزالة الاعتماد الجزئى**

علينا الآن أن نبحث فى العلاقة ITEM_FACTORY عن أى اعتماد جزئى. نجد أن الكمية المورَّدة (Sup_Quantity) تعتمد على رقم الصنف (Item_No) و المصنع المنتج (Producing_Factory). بينما تعتمد جميع الخصائص ما عدا Sub_Quantity على Item_No فقط. كل هذا يُشكِّل مشكلة اعتماد جزئى يجب إزالتها عن طريق فصل كل مجموعة جزئية على ما تعتمد عليه (مفتاح). يعنى ذلك أن العلاقة Item_Factory ذات درجة تطبيع 1NF لوجود اعتماد جزئى بها و لذلك نحصل منها على العلاقتين التاليتين :

(1) ITEM_FACTORY

Item-No	Producing_Factory	Sup_ Quantity

← 3NF

(2) ITEM_PRODUCTION

Item-No	Description	Price	Producing Factory	Factory Address

- **ثانياً : إزالة الاعتماد الانتقالى**

الآن نفحص العلاقة الأولى ITEM_FACTORY للبحث عن الاعتماد الانتقالى. لا نجد أى اعتماد انتقالى بهذه العلاقة، لذلك فدرجة تطبيعها 3NF أى طبيعية (Normalized).

أما العلاقة الثانية ITEM_PRODUCTION فنجد فيها الخاصية Factory_Address تعتمد على Producing_Factory فى الوقت الذى تعتمد Producing_factory على Item_No. معنى ذلك وجود اعتماد انتقالى. نزيل مشكلة الاعتماد الانتقالى بفصل كل مجموعة الاعتماد الانتقالى بالشكل التالى :

(2) FACTORY

Producing_Factory	Factory_Address

← 3NF

(3) ITEM

Item_No	Description	Price	Producing Factory

بناءً على ذلك فإن قاعدة البيانات التى استخلصناها باستخدام التطبيع تتضمن العلاقات الثلاث التالية :

1) ITEM_FACTORY (Item_no, Producing_factory, Sup_Quantity)

(2) FACTORY (Producing_Factory, Factory_Address)

(3) ITEM (Item_No, Description, Price, Producing_Factory)

٥-٨ تطبيق المشروعات

حدد أولاً درجة تطبيع العلاقة العامة التالية ثم قم بالخطوات اللازمة لتصبح طبيعية.

DEPT_PROJECT

رقم القسم	القسم	رقم المشروع	اسم المشروع	ميزانية المشروع	بداية المشروع	نهاية المشروع	مدير المشروع
Dept#	Dept_ Name	Proj_ No	Proj_ Name	Proj_ Budget	Start _date	End_ date	Pmanage r

الحل

نظراً لوجود مجموعات متكررة فإن درجة التطبيع 0NF أى غير طبيعية (Unnormalized). نطبّق مراحل التطبيع الثلاث كما يلى :

- **أولاً : إزالة المجموعات المتكررة**

بمجرد النظر إلى العلاقة المعطاة DEPT_PROJECT ،نجد أن الخصائص Dept# و Dept_Name تشكّل مجموعة تكرار يجب إزالتها وفصلها فى علاقة منفصلة مع الاحتفاظ بالمفتاح Dept# مع باقى خصائص العلاقة، كما يلى :

(1) DEPARTMENT

Dept#	Dept_Name

← 3NF

(2) DEPT_PROJECT

Dept#	Proj_No	Proj_ Name	Proj_ Budget	Start_ date	End_ date	Pmanager

- **ثانياً : إزالة الاعتماد الجزئى و الانتقالى**

بفحص العلاقتين السابقتين DEPARTMENT وDEPT_PROJECT نكتشف عدم وجود أى اعتماد جزئى أو انتقالى بهما، لذلك فدرجة تطبيع العلاقتين هى 3NF أى طبيعية (normalized).

- ثالثاً : إزالة الخاصيات متعددة القيم

فى حالة تبعية المشروع لأكثر من قسم أو أن القسم يشرف على أكثر من مشروع، أى أن العلاقة DEPT_PROJECT تحتوى على خاصيتين متعددتى القيم (Dept# و Proj_No) . يمكننا الحصول على درجة تطبيع 4NF لهذه العلاقة بتقسيمها إلى العلاقتين التاليتين :

(2) DEPT_PROJECT

Dept#	Proj_No

← 4NF

(3) PROJECT

Proj_No	Proj_ Name	Proj_ Budget	Start_ date	End_ date	Pmanager

٥-٩ أسئلة

١. حول التقرير التالى إلى علاقة عامة بعد استبعاد عناصر البيانات التى لا يتم تسجيلها. بعد ذلك بين درجة Normalization للبيانات ا لتالية ثم اجعلها Normalized.

Date :25/11/1423 **Department : Computer Science**

S#	SName	Course#	Cname	CLab#	Labhours	Package
417	Sayfaullah	325	DBase1	L325	2	Access
		363	AI	L363	2	Java
		426	Dbase2	L426	3	oracle

٢. حول التقرير التالى إلى علاقة عامة بعد استبعاد عناصر البيانات التى لا يتم تسجيلها. بعد ذلك بين درجة Normalization للبيانات ا لتالية ثم اجعلها Normalized.

Buraidah Hospital Doctor Report

Date :25/11/1423 **Specialty : Heart Medicin**

D#	Doctor	Dphone	Patient-code	Patient_Name	Procedure	Charge
10	Khaled	332323	455a	K. Elboghdady	X_ray	56.88
			88g	L. Khalil	Examination	787

٣. حول فاتورة المشتريات التالية إلى علاقة عامة بعد استبعاد حقول البيانات التى لا يتم تسجيلها، ثم حدد درجة تطبيعها و قم بخطوات التطبيع اللازمة لجعلها طبيعية.

قواعد بيانات
د/ طارق حلمي

الفصل الثامن

متطلبات قاعدة البيانات و إدارتها

Database Requirements & Administration

يوجد فى الأسواق نظم عديدة تخدم فى مجال قواعد البيانات أو بالأجرى فى مجال نظم إدارة قواعد البيانات ((Database Management Systems DBMS)). و قد برز منها عدة نظم أثرت حياتنا بالعديد من التطبيقات المفيدة فى جميع المجالات. لكى نستخدم تلك النظم فى إنشاء التطبيقات المختلفة نحتاج إلى التعَرُّف على كيفية إدارة قواعد البيانات و كذلك تحديد متطلباتها.

٨-١ نظم إدارة قواعد البيانات (DBMS)

Database Management Systems (DBMS)

نظام DBMS لإدارة قواعد البيانات هو نظام برامجى يُستخدَم لإنشاء و تعديل و حذف جداول قاعدة البيانات ثم التحُكم فى إدخال و تعديل و حذف البيانات و كذلك

التحكم فى التطبيقات المبنية على تلك الجداول. من أهم مسئوليات DBMS هى التحكُّم فى صلاحيات تناول المستخدمين لقواعد البيانات.

٨-١-١ منتجات DBMS DBMS Products

تتوفَّر نظم DBMS على مختلف الأجهزة و نظم التشغيل. أغلب الشركات توفِّر دائماً إصدارات من نظم DBMS على كلٍ من نظم الحاسبات الشخصية و شبكات الحاسبات و الأجهزة الكبيرة.

من أشهر نظم إدارة قواعد البيانات : مايكروسوفت أكسس (MSAccess) على الحاسبات الشخصية و أوراكل (Oracle) على الحاسبات الشخصية و شبكات الحاسب و الأجهزة الكبيرة. هناك أيضاً نظم قوية مثل : Power Builder و Sybase و Informix و غيرهم.

٨-١-٢ مكونات نظم DBMS DBMS Components

يتكون نظام DBMS من عدة مكونات أو بالأحرى برامج نظم فرعية تتكامل معاً لتوفِّر لمصممى و مبرمجى و مستخدمى و مديرى قواعد البيانات كافة الإمكانيات و التسهيلات الضرورية لإنشاء قواعد بيانات عالية الجودة و كذلك التطبيقات المبنية على تلك القواعد. فيما يلى نقدِّم بعضاً من أهم البرامج الفرعية التى يتكوَّن منها نظام DBMS.

- **محرك DBMS (DBMS Engine)**

هو الجزء المركزى الهام فى نظام DBMS، و يوفِّر الوسائل اللازمة للتعامل مع مستودعات تعريف البيانات و قاعدة البيانات و تنظيم وظائف كافة مكونات نظام DBMS. يتلقى هذا المحرِّك الطلبات و الأوامر من المستخدمين و البرامج و يتولى تنفيذها. ينظم هذا المحرك أيضاً استخدام الذاكرة و حواجز الذاكر. كما يُحدد أيضاً موقع البيانات على قرص البيانات و يُصدر أوامر التخزين و الاسترجاع إلى نظام التشغيل.

- **برنامج الاتصال بمكونات DBMS**

يوفِّر هذا البرنامج وسائل اتصال المستخدمين و التطبيقات مع مكوِّنات DBMS المختلفة مثل : لغات برمجة DBMS (مثل PL\SQL) و لغة الاستعلام عن البيانات (DDL) و لغة تناول البيانات (DML) و وسيط للغات التقليدية (مثل C و C++) و وسيط لإظهار الرسوم و الوسائط المتعددة (Multimedia) و نماذج الشاشات (Forms) و التقارير (Reports).

- **برنامج أمن البيانات (Data Security)**

يتيح هذا البرنامج الحماية و التحكم فى البيانات و المستخدمين.

- **برنامج عمل النسخ الاحتياطية (Backup)**

هذا البرنامج يوفِّر وسيلة تخزين نُسَخاً من كامل البيانات أو جزء منها من أجل توفير نسخة بديلة عند تعرُّض الجهاز أو البرامج أو البيانات لأى مشكلة.

- **برنامج استعادة البيانات (Recovery)**

يقوم هذا البرنامج باستعادة البيانات المفقودة بسبب أعطال البرمجيات أو الأجهزة أو وسائل نقل البيانات.

- **برنامج لإدارة تكامل و اتساق قاعدة البيانات**

يهتم هذا البرنامج بالقيود المفروضة على الجداول و العلاقات بين الجداول و التحقق من تنفيذها. و من أهم وظائفه ضبط تعاملات المستخدمين لقاعدة البيانات المتزامنة (فى نفس الوقت) فيما يُعرَف بالتحكم المتزامن فى البيانات.

٨-٢ إدارة قواعد البيانات Database Administration

يقع على كاهل إدارة قاعدة البيانات (مدير و موظفين) العديد من المسئوليات التى تتكامل من أجل ضمان إنتاج و تشغيل قاعدة بيانات صحيحة متكاملة فعالة تؤدى الوظائف المنوطة بها.

إدارة البيانات (Data Administration) هى عملية إدارية عالية المستوى مسئولة عن إنشاء نماذج البيانات و كذلك معالجة تعاريف البيانات المشتركة.

إدارة قاعدة البيانات (Data Administration) هى عملية فنية مسئولة عن التصميم الفعلى لقواعد البيانات و كذلك السرية و الأداء و النسخ الاحتياطى لقواعد البيانات.

مدير قاعدة البيانات ((Database Administration (DBA) هو شخص يُفتَرَض أن يكون ذو مهارات شخصية عالية لتعامله مع مختلف الأفكار و وجهات النظر ويحتاج إلى سياسة و حكمة لكى يستطيع أن يتعامل مع أربعة مجموعات : المستخدمين (من أجل الاحتياجات و السياسات) و الإدارة (من أجل تحقيق الأهداف و الخطط فى ظل ميزانية محددة) و العمليات (النواحى القياسية و الإرشادات الفنية) و مُطَوِرى التطبيقات (من أجل المتطلبات الفنية و السياسات الموحدة).

إنشاء قواعد البيانات

إنشاء قاعدة بيانات تمر بعدة مراحل تبدو مستقلة أحياناً و متشابكة أحياناً أخرى. تبدأ من التخطيط ثم تحديد المتطلبات ثم التصميم و التنفيذ والتشغيل و الصيانة. و قد تعرَّفنا على أغلب هذه المراحل ونتعرَّف الآن على كيفية تحديد متطلبات قاعدة البيانات.

٨-٣ تحديد متطلبات قواعد البيانات

Database Requirements Definitions

تأتى عملية تحديد متطلبات قاعدة البيانات ضمن عملية تصميم قاعدة البيانات التى تنصب على إنشاء هياكل لقاعدة البيانات طبقاً لاحتياجات المستخدم من البيانات.

٨-٣-١ تمهيد Preface

تحديد متطلبات قاعدة البيانات هى عملية تعريف احتياجات المستخدمين (مع تنوعها) الحالية و المستقبلية (و هى ما سبق أن أطلقنا عليه نموذج البيانات الخارجى).

يلى ذلك تحويل هذه المتطلبات إلى نموذج يجمع كافة المتطلبات فى شكل موحد يبين الكيانات و العلاقات الرابطة بينهما (نموذج البيانات المفاهيمى). تأتى بعد ذلك مرحلة التصميم القابل للتنفيذ بوضع النموذج المفاهيمى فى شكل يتناسب مع نظام DBMS الذى سيتم التنفيذ الفعلى النهائى بما يتضمنه من برمجيات و أجهزة معينة (نموذج البيانات الداخلى).

٨-٣-٢ خطوات تحديد متطلبات قاعدة البيانات

Phases of Database Requirement Definitions

عملية تحديد متطلبات قاعدة البيانات تمر بعدة مراحل هى :

- **تحديد هدف و مجال قاعدة البيانات** هو تحديد الهدف الأساسى للتطبيق و ارتباطه بالخطة العامة للجهة صاحبة التطبيق (أى تحديد عنوان دقيق جزئى و ليس عام يحدد اسمه أنواع البيانات المتوقع استخدامها).

- **اختيار أساليب تحديد المتطلبات** التى تحدد الوسائل اللازمة لإظهار رؤى المستخدمين من مطبوعات و استفسارات من خلال نماذج مناسبة.

- **تحديد طلبات المستخدمين** هى حصر و جمع البيانات تركّز على طلبات المستخدمين التى يُشكِّل كل منها جزء من البيانات المطلوبة لعمل معين أو مهمة محددة، مع معرفة كيفية تناول هذه البيانات و من يستعملها و كيف يستعملها و متى يستعملها و كيفية تشكيل البيانات.

- **نمذجة البيانات** بعمل نموذج يوضّح الكيانات و العلاقات الرابطة بينها و عناصر البيانات و المفاتيح الأساسية.

- **نمذجة القيود على البيانات** بتحديد كافة القيود الأساسية على البيانات و الكيانات و العلاقات الرابطة بينهما.

٨-٣-٣ تحديد القيود على قاعدة البيانات

Defining Constraints on Database

كما سبق أن أشرنا فى السطور السابقة إلى ضرورة نمذجة القيود على قاعدة البيانات و هى تقع تحت ثلاثة أنواع :

- **قيود على عنصر أو حقل البيانات**

تُستَخدَم قيود المفاتيح (Key Constraints) كقيود على الحقول. فعندما نحدد أن خاصيةً ما أو حقلاً هو المفتاح الأساسى (Primary Key) فهذا يعنى أن عناصر البيانات التى يتم تخزينها فى هذا الحقل فريدة وحيدة لا تتكرر و لا يُمكن أن تُترَك فراغاً. كذلك نستطيع فرض قيود على خصائص معينة ألا تكون فراغاً أو تكون سلسلة حرفية طولها متغير أو ثابت أو أن تكون متغيراً رقمياً محدوداً بين عدة قيم أو بين قيمة دنيا و قيمة قصوى. كذلك يمكن أن ترتبط قيم حقل معين بقيم حقل آخر بأى علاقة ممكنة.

- **قيود على الكيانات**

القيود الخاصة هى **قيود مشاركة و وجود (Participation & Existence Constraints)**. يحدد ما لو كان وجود كيان معين يعتمد على ارتباطه بكيان آخر من خلال علاقة رابطة بينهما. هذه المشاركة تكون إما **كاملة (Total Participation)** أو **جزئية (Partial Participation)**.

على سبيل المثال، لا يتم فتح شعبة دراسية فى أى مقرر دراسى إلا فى حال تسجيل طالب أو طلبة فى تلك الشعبة، أى أن سجل الشعبة لا يمكن أن يتواجد إلا فى حالة سجل فى العلاقة الرابطة يُسَجِّل (REGISTER). هذا القيد يُعَد قيد مشاركة كاملة أو مشاركة وجود.

أما العلاقة يُدَرِّس (TEACH) فيها نجد أن جزءً من أعضاء هيئة التدريس يمكن أن يقوموا بتدريس مقرر معين (أى يرتبط جزء أو مجموعة سجلات أعضاء هيئة التدريس بسجل فى العلاقة COURSE من خلال سجل فى العلاقة TEACH بينما بعضهم لا يشاركون فى التدريس بل فى أعمال بحثية مثلاً). هذه المشاركة مشاركة جزئية.

- **قيود على العلاقات الرابطة بين الكيانات**

هذا النوع من القيود يختص بنسبة المشاركة من كل كيان فى العلاقة الرابطة و هى ما لأطلقنا عليه الكاردينالتى و هى تكون أحد المشاركات التالية : (0:1) أو (1:1) أو (0:N) أو (1:N). و قد سبق أن تعرَّفنا عليها بالتفصيل فى الفصول الثالث و الرابع و الخامس.

٨-٣-٤ متطلبات التشغيل Operating Requirements

من المهم جمع معلومات عن متطلبات المستخدمين لمعالجة البيانات، أهمها :

- **أمن البيانات** : تحديد صلاحيات تناول البيانات للمستخدمين.
- **حفظ البيانات** : كيف نحفظ البيانات و إلى متى؟
- **إعادة استخدام البيانات** : ما هى القيود الموضوعة لاستخدام البيانات و تحديد وقت استخدامها.
- **نمو قاعدة البيانات** : كيف تنمو قاعدة البيانات مستقبلاً.

٨-٤ أسئلة

١. ما المقصود بكلٍ من : DBA و DBMS.
٢. ما هى مهام DBA؟
٣. أذكر أمثلة لنظم DBMS الواسعة الانتشار.
٤. اشرح بإيجاز مكونات نظام DBMS.
٥. بين المقصود بعملية تحديد متطلبات قاعدة البيانات و مراحلها.
٦. ما الفرق بين إدارة قاعدة البيانات و إدارة البيانات؟
٧. أذكر خطوات تصميم قاعدة البيانات.

الفصل التاسع

قواعد البيانات الموزعة و التحكم المتزامن

Distributed Database & Concurrency Control

فى العهد القريب كان يتم إنشاء قاعدة البيانات من أجل مجموعة معروفة و شبه محددة من المستخدمين من خلال نهايات طرفية تتصل بالحاسب الكبير أو على نفس الجهاز. بعد التطور المُذهل فى مجال الاتصالات و الحاسبات الشخصية و شبكة الإنترنت، زاد تناول البيانات من قِبَل مستخدمين جُدُد فى بلدان شتى و غير محددين و ذوى اهتمامات شتى.

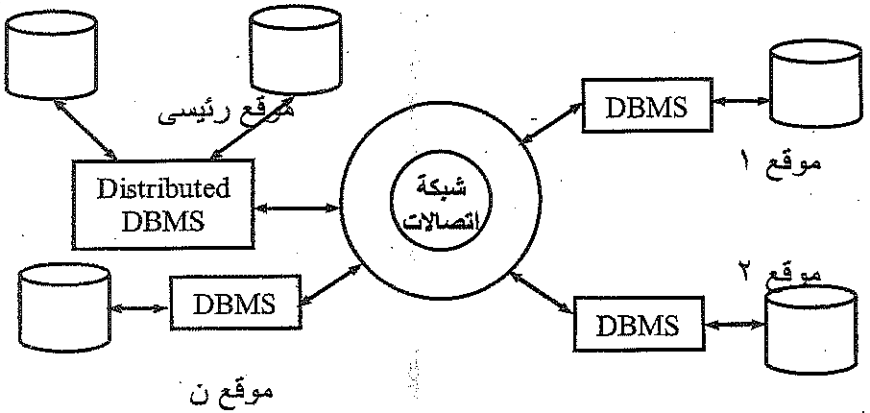
لقد ظهرت فى الآونة الأخيرة فكرة الحكومة الإلكترونية (Electronic Government). تقوم فكرة الحكومة الإلكترونية على عناصر كثيرة منها تسهيل إنهاء إجراءات المواطنين و الأجانب مع المكاتب الحكومية من خلال الإنترنت.

حيث يستطيع الفرد أن يتعرف على شروط الطلبات و يقدم طلبه و يحصل على نتيجة ذلك من خلال شبكة الإنترنت. كل ذلك أدى إلى زياد أعداد المتناولين لقواعد البيانات. و لقد حققت بعض الدول خطوات كثيرة في هذا الاتجاه و معظم الدول تسعى جاهدة لتحقيق أهدافاً كثيرة فى هذا الاتجاه.

٩-١ قاعدة البيانات الموزّعة Distributed Database

انتشرت قواعد البيانات فى حياتنا العامة بشكل واسع النطاق و أصبحت متاحة للعديد من المستخدمين فى نفس الوقت. فمثلاً، ماكينات الصرافة الموجودة فى الشوارع يستخدمها الكثيرون فى نفس الوقت لتناول قاعدة بيانات خاصة بحسابات المصارف. كذلك قاعدة بيانات حجز خطوط الطيران يستخدمها مئات الآلاف حول العالم لإجراء حجز أو تعديله أو حذف فى نفس الوقت.

قاعدة البيانات الموزّعة (Distributed Database) هى تجميع لعدة قواعد بيانات بينها علاقة منطقية متبادلة و توزع على شبكة حاسب آلى و نظام إدارة قاعدة بيانات موزعة مع جعل التوزيع واضحاً للمستخدم. و لقد أصبح شائعاً وجود مجموعة من ملفات قواعد البيانات المخزنة فى نقاط مختلفة من شبكةٍ ما ومعالجة الروابط بينهما عبر روابط معينة من خلال شبكة الإنترنت. الشكل ٩-١ يعطى مثال لقاعدة بيانات موزعة.



شكل ٩-١ : نموذج لقاعدة بيانات موزّعة.

٩-٢ إمكانيات قواعد البيانات الموزعة

Distributed Database Capabilities

قواعد البيانات الموزعة تجلب مميزات الحساب الموزع إلى ميدان إدارة قواعد البيانات. يتكون نظام الحساب الموزع (distributed computing system) من عدد من العناصر المعالَجَة وليست بالضرورة متجانسة و لكنها ترتبط ببعضها البعض بواسطة شبكة حاسب و تتعاون لإنجاز أعمال محددة موكلة إليها. الهدف العام من نظم الحساب الموزع هو تقسيم مشكلة كبيرة صعبة التنفيذ إلى أجزاء صغيرة ثم تحلها بكفاءة بأسلوب متناسق.

إمكانية تطبيق هذا الأسلوب تنجم عن سببين : (١) يتم تسخير قدرة أكبر من الحاسب لحل الأعمال المعقدة و (٢) كل عنصر معالج مستقل يمكن إدارته على حدة و يطور تطبيقاته الخاصة المستقلة.

قاعدة البيانات الموزعة يمكنها وصف النظم المختلفة التى تختلف من نظام لآخر فى نقاط عديدة. الشىء الرئيسى الذى يوجد فى كل هذه النظم عموماً هو أن البيانات و البرمجيات يتم توزيعها عبر عدة مواقع متصلة بشكلٍ ما من شبكات الاتصال. رغم صعوبتها و تعقيدها فإن قواعد البيانات الموزّعة تتمتّع بالمميزات التالية :

- مناسبة للجهات ذات المواقع الكثيرة الموزعة.
- لامركزية تنظيمية.
- تشغيل الاقتصادى.
- استقلالية الكبيرة للأجهزة
- استقلالية نظم DBMSs.
- استقلالية لنظم التشغيل و الشبكات.
- استقلالية المواقع.
- تجزئ الاستعلامات.
- شفافية موقع البيانات.
- شفافية الأداء.

- شفافية التكرار و التوزيع.
- استمرارية العمل.
- سهولة التوسع بإضافة مواقع جديدة.

٩-٣ شبكة حاسبات الخادم و التابع

Client-Server Network

لقد تم تطوير قواعد البيانات الموزّعة اعتماداً على التطوير الكبير فى شبكات الخادم و التابع. حيث يوجد **حاسب آلى كبير (الخادم Server)** ينظم و يخدم و يدير مجموعات **حاسبات أصغر (تابعين Clients)** من خلال **تقنيات شبكة تراسل البيانات (Data Communication Network)**. يتم توزيع الأدوار ما بين الخادم و التابعين بأكثر من طريقة.

فى أحد هذه الطرق نجد أن الحاسب الآلى الذى يعمل كخادم SQL يسمى **خادم التعاملات/المعاملات (Transaction Server)** أو **معالج قاعدة البيانات (Database Processor)**. بينما يُسمّى الحاسب الذى يعمل فى دور التابع **معالج التطبيق(Application Processor)**. التفاعل بين الخادم و التابع يمكن تحديده من قِبَل المستخدم عند مستوى الخادم أو خلال برامج خادم DBMS متخصص يعد جزءً من حزمة برامج DBMS.

على سبيل المثال ربما يَعرِف المستخدم أياً من البيانات تكون مخزنة عند كل خادم، يفتت طلب الاستعلام يدوياً إلى استعلامات جزئية على كل موقع و يسلم الاستعلامات الجزئية المنفردة إلى مختلف المواقع. الجداول الناتجة يمكن توحيدها علانية بواسطة استعلام مستخدم بعد ذلك على مستوى التابع. البديل أن تتوفر برامج عند التابع لتنفيذ تلك الأفعال آلياً.

التابع يكون مسئولاً عن توليد خطة تنفيذ موزعة لاستعلام أو معاملة متعددة المواقع و كذلك توجيه التنفيذ الموزع بإرسال الأوامر إلى الخادمين.تشمل تلك الأوامر الاستعلامات و المعاملات المحلية المطلوب تنفيذها مثل أوامر إرسال البيانات لتابعين أو خادمين آخرين. أى أن برمجيات التابع يجب أن تكون موجودة عند كل موقع بحيث يمكن استلام استعلامات متعددة المواقع.

٩-٤ التحكم المتزامن Concurrency Control

التزامن (concurrency) هو وقوع أحداث متزامنة أى فى نفس الوقت. فى مجال قواعد البيانات التى يستخدمه أكثر من مستخدم يطلب عدة مستخدمين قراءة أو تعديل أو إدخال بيانات نفس وحدة البيانات فى نفس الوقت. و نقصد هنا **بوحدة البيانات (Data Unit)** هو أى مجموعة بيانات موجودة فى إطار واحد يجمعها معاً. يتراوح حجم وحدة البيانات من عمود فى جدول إلى قاعدة البيانات بالكامل كما سنعرف فى ٩-٤-٣.

٩-٤-١ لماذا التحكم المتزامن؟

Why Concurrency Control?

نتيجة لطلبات المستخدمين المتزامنة يحدث تداخل بين العمليات التى يقوم بها المستخدمين. من الممكن أن يؤدى هذا التداخل إلى حدوث خلل فى البيانات المخزَّنة داخل قاعدة البيانات كما نرى فى الشكل ٩-٢ حول حجز سلع أو أصناف غذائية من المورد.

فى هذا الشكل يطلب كل موظف حجز معرفة عدد الوحدات المتاحة من سلعة غذائية معينة ثم يقوم بتوقيع الحجز على العدد المطلوب إن كان متاحاً أو يرفض إجراء الحجز إن كان غير ممكن.

لكننا نرى أن الموظف الأول قرأ عدد الوحدات المتاحة من السلعة رقم ٣١٢ فوجدها ٨ وقبل أن يقوم بحجز عدد الوحدات المطلوبة منه و هى ٦ وحدات، قام الموظف الآخر بقراءة نفس العدد المتاح من وحدات الصنف ٣١٢ (لازالت ٨ وحدات) و يرغب فى حجز ٣ وحدات.

تعامل رقم : ت١	تعامل رقم: ت٢
١. قراءة عدد وحدات السلع المتاحة من الصنف ٣١٢. (عدد الوحدات المتاحة =٨ وحدات)	١. قراءة عدد الوحدات المتاحة من الصنف ٣١٢. (عدد الوحدات المتاحة = ٨ وحدات)
٢. حجز ٦ وحدات من الصنف ٣١٢ (Number = 8-6= 2)	٢. حجز ٣ وحدات من الصنف ٣١٢ (Number = 8-3 = 5)
٣. تعديل عدد الوحدات المتاحة فى سجل الصنف ٣١٢	٣. تعديل عدد الوحدات المتاحة فى سجل الصنف ٣١٢
(عدد الوحدات المتاحة =٢ وحدة)	(عدد الوحدات المتاحة = ٥ وحدات)

العدد الفعلى الصحيح هو ٨-٦-٣ = -١ وحدة

شكل ٩-٢ : مشكلة التداخل بين التعاملات المتزامنة.

بعد ذلك قام الموظف الأول بحجز ٦ وحدات و أصبح عدد الوحدات ٢ وحدة. تلاه حجز الموظف الثانى لثلاث وحدات من الرقم الموجود عنده و هو ٨ فأصبح العدد المتاح ٥ وحدات. بعد ذلك قام البرنامج بتسجيل العدد المتبقى من

الوحدات مرة مع كل موظف (أولاً ٢ و ثانياً ٥ وحدات). ما نرى أنه فى الأصل لا يمكن إجراء الحجز الثانى فعلياً لعدم توفُّر العدد المطلوب. هكذا تحدث مشكلة للعملاء ساعة تسليم السلع !

من الممكن ذكر أمثلة عديدة للمشاكل التى يمكن أن تحدث نتيجة لتزامن تعاملات المستخدمين فى المصارف و خطوط الطيران و المبيعات و غير ذلك. لذلك لابد من استخدام ما يُعْرَف **بالتحكم المتزامن (Concurrency Control)**. نستطيع تعريف **التحكم المتزامن** بأنه **التحكم فى التناول المتزامن للبيانات بجعل كل مستخدم لقاعدة البيانات المشتركة يبدو و كأنه المستخدم الوحيد لقاعدة البيانات.**

نتيجة لذلك كان من الضرورى وجود تقنية فى نظام DBMS تختص بالتحكم المتزامن فى تعاملات المستخدمين أو بالأحرى تختص بالتحكم فى تعاملات المستخدمين المتزامنة. هذه التقنية تُسَمَّى **برنامج مدير التحكم المتزامن (Concurrency Control Manager)**. يختلف عمل مدير التحكم المتزامن من نظام DBMS لآخر حسب الطريقة المستخدمة فى ذلك.

٩-٤-٢ حجز (حجب) البيانات Data Locking

أغلب نظم DBMS تستخدم طرق لحجز البيانات أو حجبها لصالح المستخدم الذى يقوم بتناولها دون غيره من المستخدمين. و يستمر حجز البيانات إلى أن ينتهى ذلك المستخدم و يؤكّد تنفيذ تعامله مع قاعدة البيانات أو يتراجع عنه و يلغيه تماماً. عند ذلك يتم إلغاء الحجز عن تلك البيانات و تُصبح متاحة لجميع المستخدمين.

عملية حجز البيانات هذه تمنع أخطاء تحديث البيانات (إدخال أو تعديل أو حذف). دعنا نشبه عملية حجز وحدة بيانات لصالح مستخدم معين بحجز وحدة من صنف معين. حيث أنه لا يستطيع أى عميل ثان أن يأخذ هذه الوحدة إلا فى حال إلغاء الحجز من قِبَل العميل الأول. دعنا نطبّق عملية حجز البيانات على مثال حجز وحدات للسلع و الأصناف الغذائية، كما نرى فى الشكل ٩-٣.

تعامل رقم : ت١	تعامل رقم: ت٢
١. طلب حجز سجل الصنف ٣١٢	١. طلب حجز سجل الصنف ٣١٢.
٢. قراءة عدد الوحدات المتاحة من صنف ٣١٢. (عدد الوحدات المتاحة =٨ وحدات)	(غير متاح، عليك الانتظار)
٣. طلب حجز ٦ وحدات من الصنف ٣١٢ (Number = 8-6= 2)	
٤. تعديل عدد الوحدات المتاحة فى سجل الصنف ٣١٢ (Number =2)	
٥. إلغاء/رفع الحجز عن سجل صنف ٣١٢	٢. طلب حجز سجل الصنف ٣١٢.
	٣. قراءة عدد الوحدات المتاحة على صنف ٣١٢. (عدد الوحدات المتاحة = ٢ وحدة)
	٢. طلب حجز ٣ وحدات على الصنف ٣١٢ (Number =2-3 = 5)

طلب الحجز الثانى مرفوض بسبب قلة عدد الوحدات المتاحة عن المطلوب

شكل ٩-٣ : حل مشكلة التعاملات المتزامنة باستخدام حجز البيانات.

نرى فى الشكل ٩-٣، أن النظام رفض طلب الموظف الثانى لسجل الصنف ٣١٢ لأنه محجوز لصالح تعامل الموظف الأول و استمر ذلك حتى تم

الانتهاء من تعامل الموظف الأول تماماً. بعد ذلك أصبح متاحاً للموظف الثانى أن يحجز نفس السجل. بعد ذلك اكتشف النظام أن العدد المتاح غير كافٍ فأعتذر عن طلب الموظف الثانى.

٩-٤-٣ أنواع الحجز Locking Techniques

طريقة حجز البيانات التى رأيناها فى الشكل ٩-٣ قد تمنع أخطاء إدخال أو تعديل أو حذف البيانات لكنها لا تصلح فى جميع الأحوال. فربما أراد تعامل موظف الحجز الأول أن يستعلم عن عدد الوحدات المتاحة دون رغبة فى حجز فعلى لبعض الوحدات، بينما كان تعامل موظف الحجز الثانى أن يستعلم أيضاً أو أن يحجز فعلياً بعض الوحدات، فلما الانتظار؟

لذلك هناك نوعين من الحواجز أو الغوالق (Locks) التى يستطيع النظام أن يستخدمها لعمل الحجز أو الحجب المناسب على البيانات. نوعى الحاجز هما :

- **حاجز مشترك للقراءة (Shared Lock)**

يُستخدَم هذا الحاجز لحجز (Locking) وحدة بيانات للقراءة دون أن يحجب وحدة البيانات عن باقى المستخدمين. نُسمِّى هذه العملية عملية حجز مشترك أو حجز قراءة. مع هذا الحاجز يستطيع جميع المشتركين معه قراءة وحدة البيانات المحجوزة دون إجراء أى تحديث لها من قِبل أى مستخدم.

- **حاجز خاص للكتابة (Exclusive Lock)**

يُستخدَم هذا الحاجز لحجز وحدة بيانات للكتابة و يحجب وحدة البيانات عن باقى المستخدمين. نُسمِّى هذه العملية عملية حجز خاص أو حجز كتابة. مع هذا الحاجز يستطيع من قام بحجز وحدة البيانات أن يجرى أى تناول (استعلام إدخال و تعديل أو حذف) دون أن يتمكّن أى مستخدم آخر من التعامل بأى شكل مع وحدة البيانات تلك (لا يستطيع جميع المشتركين حتى قراءة وحدة البيانات المحجوزة) حتى ينتهى تنفيذ تعامل هذا المستخدم بالتأكيد أو الإلغاء و تحرير وحدة البيانات من الحاجز (Unlocking).

٩-٤-٤ أنواع وحدة البيانات المحجوزة Locked Data Items

قبل تطبيق أى تقنية للتحكم المتزامن باستخدام أحد نوعى الحاجز التى درسناها تواً علينا أن نحدد على أى وحدة بيانات. كما سبق أن عرفنا فى الجزء الثانى من هذا الكتاب أن نظام DBMS (من خلال SQL مثلاً) يُمكِّنا من التحكم فى كل جزء من قاعدة البيانات. نوع وحدة البيانات يحدد **مستوى الحجز (Blocking Level)** كما يلى :

- **عمود (حقل) فى جدول (Column)**

هنا يتم حجز عمود بيانات فى جدول معين مثل عمود الراتب Salary من الجدول EMPLOYEE.

- **سجل (سطر) فى جدول (Record)**

هنا يتم حجز سجل واحد فقط فى جدول معين مثل سجل رقم ٥ للموظف Khaled من سجلات الجدول EMPLOYEE. فى هذه الحالة تكون باقى السجلات متاحة لأى تعاملات.

- **جدول (Table)**

هنا يتم حجز كافة البيانات الموجودة فى جدول معين مثل الجدول EMPLOYEE. يحدث ذلك عند رغبتنا فى إجراء تحديث لجميع سجلات الجدول.

- **مخطط قاعدة البيانات (Schema)**

هنا يتم حجز مخطط قاعدة البيانات بالكامل مثل مخطط قاعدة البيانات COLLEGE. يحدث ذلك عند رغبتنا فى عمل نسخة احتياطية مثلاً.

٩-٤-٥ مشكلة التوقف التام Dead Lock Problem

تحدث مشكلة التوقف التام عندما يحجز تعامل معين م١ وحدة بيانات ب١ مثلاً، ثم يحجز تعامل آخر م٢ وحدة البيانات ب٢. أثناء تنفيذ التعاملَين يطلب التعامل م١ حجز وحدة البيانات ب٢ فيجدها محجوزة للتعامل م٢ فيبقى فى وضع انتظار (wait state) حتى ينتهى التعامل م٢. أثناء استكمال تنفيذ التعامل م٢ يطلب م٢ حجز وحدة البيانات ب١ فيجدها محجوزة للتعامل م١ فيدخل فى حالة الانتظار

أيضاً. و يَظَل التعاملين م١ و م٢ فى حالة انتظار لانتهاء التعامل الآخر إلى الأبد أو إلى أن يلغى مستخدم ما أحد التعاملين أيهما أقرب. هذا يُعرَف **بالتوقف التام (Dead Locking)**.

طرق التخلص من التوقف التام

يمكن أن يستخدم برنامج مدير التحكم المتزامن عدة طرق للتخلص من التوقف التام، نذكر منها :

- **منع حدوث التوقف التام**

على طريقة "الوقاية خير من العلاج"، يستطيع مدير التحكم المتزامن منع حدوث مشكلة التوقف التام من أساسها. فيقوم التعامل بحجز كافة وحدات البيانات التى يرغب فى حجزها عند بداية تنفيذ التعامل. مثلاً، يحجز التعامل م١ فى المثال السابق وحدتى البيانات ب١ و ب٢ من البداية إلى النهاية. لكن لإجراء ذلك يجب أن يكون بالإمكان معرفة كافة وحدات البيانات المطلوبة قبل بدء التنفيذ و هذا غير ممكن فى أغلب الأحوال.

- **إزالة التوقف التام**

هذه الطريقة أكثر استخداماً، حيث تسمح بحدوث التوقف التام و عندما تكتشف حدوث هذه المشكل تقوم بحلها أو إزالتها. يقوم مدير التحكم المتزامن بذلك عن طريق معالجة مصفوفة تحتوى فى كل سطر على التعامل (الفاعل) الذى

يستخدم **وحدة بيانات** معينة (المفعول به). ثم يقوم بفحص المصفوفة بصفة دورية ليكتشف حالات التوقف التام.

عند اكتشاف حالة توقف تام يقوم مدير التحكم المتزامن بإلغاء أحد التعاملَين (يُسمَى التعامل الضحية) اللذين يسببان حالة التوقف التام مع التراجع عن أى عملية قام بتنفيذها (Rollback) و بعد انتهاء التعامل الآخر يُعاد تنفيذ كافة عمليات التعامل الضحية (REDO).

٩-٤-٦ التحكم المتزامن دون استخدام الحواجز

Concurrency Control without Locks

توجد عدة طرق للتحكم المتزامن لا تستخدم الحواجز، بالتالى لا ينشأ معها مشكلة التوقف التام. هذه الطريق تستخدم ما يُعْرَف بالعلامة الزمنية. **العلامة الزمنية (Timestamp)** هى محدد وحيد ينشئه DBMS لتمييز المعاملة. يتم إعطاء قيم للعلاقات الوقتية طبقاً لترتيب تسليم المعاملات إلى النظام لتنفيذها. لذلك يمكن اعتبار العلامة الزمنية بمثابة وقت بدأ التعامل. طرق التحكم المتزامن التى تستخدم ترتيب العلامة الزمنية لا تستخدم الحواجز.

من طرق التحكم المتزامن التى تستخدم العلامة الزمنية، طريقة **التحكم المتزامن باستخدام تعدد النُسَخ** لقيم وحدات البيانات (**Multiversions Concurrency Control**). هذه الطريقة تحفظ القيم السابقة لوحدة البيانات

عندما تتغير قيمتها. يُعْرَف ذلك بطرق التحكم المتزامن متعدد النسخ (القيمة) لأنه تتم معالجة القيم المختلفة لعنصر البيانات.

فعندما تطلب معاملة ما التعامل مع عنصر يتم اختيار القيمة (النسخة) المناسبة للعنصر لمعالجة تسلسل جدول المعاملات المنفذة حالياً فى نفس الوقت إن كان ممكناً. الفكرة أن بعض عمليات القراءة التى ربما يتم رفضها فى بعض الطرق يمكن قبولها عن طريق قراءة النسخة الأقدم للعنصر. فعندما تكتب معاملة ما عنصر معين فإنها تكتب نسخة جديدة أما النسخة القديمة فتبقى و يحتفظ بها النظام.

٩-٥ أسئلة

١. ما المقصود بكلٍ من : الحكومة الإلكترونية و قاعدة البيانات الموزّعة و نظام الحساب الموزّع؟
٢. عرّف التزامن و التحكم المتزامن. ما المسئول عن التحكم المتزامن؟
٣. أذكر مميزات قاعدة البيانات الموزعة؟
٤. ما المقصود بوحدة البيانات و ما هى مستويات حجز البيانات؟
٥. ما هى شبكة الخادم و التابع؟ مما تتكون؟ ما دور كل منهما؟
٦. اشرح بإيجاز أنواع حجز البيانات؟
٧. وضح كيفية حدوث مشكلة التوقف التام و كيفية معالجتها؟
٨. ما المقصود بالعلامة الزمنية؟
٩. اشرح بإيجاز طريقة للتحكم المتزامن لا تستعمل الحواجز.

الفصل العاشر

استعادة البيانات و أمن قاعدة البيانات

Data Recovery & Database Security

ليس هناك أدنى شك فى أن قواعد البيانات تكتسب أهميتها من كيفية إدارتها للبيانات و تسهيل تخزين البيانات فى شكل ما و كذلك استرجاعها أو تعديلها أو حتى حذفها بأكثر من شكل و بأكثر من وسيلة. فى أغلب نظم المعلومات تصبح البيانات هى العنصر الأهم و الأغلى ثمناً و التى من أجلها كانت قواعد البيانات. لذلك كان لابد من أن يتضمن أى نظام DBMS على تقنيات و أساليب للحفاظ على البيانات و تأمينها. كذلك لابد من وجود تقنيات لاستعادة البيانات المفقودة نتيجة لأعطال البرامج أو الأجهزة.

١٠-١ استعادة البيانات Data Recovery

عندما يمرض الإنسان فإن عافيته تضعُف و يحتاج إلى العلاج و الدعاء حتى يشفى أو يتعافى من المرض، أى يستعيد عافيته و صحته. هذا هو الحال بالنسبة للبيانات، فقد نفقد بعض أو كل البيانات نتيجة لسبب أو لأكثر، و علينا أن نستخدم التقنيات المتاحة فى نظام إدارة قاعدة البيانات لاستعادة البيانات المفقودة و عودة قاعدة البيانات إلى حالة متجانسة و متماسكة (الحالة السابقة للعطل).

١٠-١-١ أسباب فقد البيانات Data Lost Reasons

فقد البيانات أو ضياعها يحدث نتيجة أسباب كثيرة. بعض هذه الأسباب ينجم عن خطأ بعضها بَشَرِى و البعض الآخر نتيجة خطأ فنى. نستطيع حصر أهم أسباب فقد البيانات فيما يلى :

- خطأ بشرى فى إدخال بيانات غير سليمة.
- عدم اكتمال تنفيذ بعض العمليات التى تجرى على البيانات.
- تعطُّل نظم البرامج.
- تعطل الأجهزة.
- تعطل خطوط نقل البيانات (الشبكات).
- فيروسات الحاسب.
- كوارث طبيعية.

يوفِّر نظام DBMS عدة تقنيات لمواجهة مثل هذه المشاكل و استعادة البيانات و إعادتها إلى الحالة السابقة للفقد أو الخطأ مباشرة.

١٠-١-٢ إمكانيات الاستعادة Recovery Capabilities

يوفِّر نظام DBMS العديد من الإمكانيات المتاحة التى تجعل عملية الاستعادة أمراً ممكناً فى أغلب أنواع فقد البيانات منها :

- **النسخ الاحتياطى (Backup)**

يوفِّر نظام DBMS إجراءً آلياً لعمل نسخة احتياطية لكامل قاعدة البيانات. يتم ذلك بعمل صورة كاملة لقاعدة البيانات بالكامل (Dump). على أن توضع هذه النسخة فى مكان آمن من كافة أشكال التلف أو الفقد أو السرقة.

- **مفكرة النظام (System Log or Journal)**

يوفرّ نظام DBMS ما نستطيع أن نطلق عليه مفكرة النظام. يقوم النظام بتسجيل كافة التعاملات مع قاعدة البيانات و كذلك تأثيرها على البيانات بصفة دورية. يمكن استخدام محتويات هذه المفكرة إما بإعادة تنفيذ بعض التعاملات (استعادة أمامية) أو بإلغاء بعض التعاملات (تراجع).

- **نقط الاختبار (Check Point)**

بصفة دورية يقوم نظام DBMS بتعطيل كافة العمليات و يضع ما يُعْرَف بنقاط الاختبار (و هو سجل يحتوى على المعلومات الهامة إعادة تشغيل

النظام) عدة مرات فى الساعة بعد التأكد من تنفيذ كافة التعاملات بشكل سليم، و تصبح هذه النقط هى نقط البداية الجديدة.

هذه الخاصية تمكّن نظام DBMS من تنفيذ كافة التعاملات بالكامل مع تعديل ملفات المفكرة دورياً، و فى حالة حدوث عطل فإننا نفقد عمل عدة دقائق فقط و تُعاد مرة أخرى.

- **برنامج إدارة الاستعادة (Recovery Manager)**

هو برنامج يقوم بإرجاع قاعدة البيانات إلى الحالة السليمة عند حدوث الأعطال ثم يعيد تنفيذ تعاملات البرامج و المستخدمين من المفكرة.

١٠-١-٣ طرق الاستعادة Recovery Techniques

بناءً على نوع فقد البيانات و إمكانيات الاستعادة يمكن استخدام أحد طرق الاستعادة التى نقدمها فيما يلى :

- **الاستعادة العكسية (Backward Recovery)**

تُستَخدَم الاستعادة العكسية لعمل تراجع عن الفعل (إلغاء تنفيذ التعاملات وإزالة تأثيراتها) أى UNDO و العودة إلى الحالة السابقة لقاعدة البيانات.

- **الاستعادة الأمامية (Forward Recovery)**

تُستَخدَم الاستعادة الأمامية بالبدء بآخر نسخة منن قاعدة البيانات و إعادة تنفيذ التعاملات الجديدة لتحدث التأثيرات المطلوبة فى قاعدة البيانات أى REDO. هذه الطريقة عكس طريقة الاستعادة العكسية.

- **إعادة التحميل و إعادة التشغيل (Restore & Rerun)**

تُستَخدَم طريقة إعادة التحميل و التشغيل لتعاملات السابقة للعطل بعد آخر نسخة احتياطية. حيث يجرى تحميل النسخة الاحتياطية (Restore) ثم إعادة تشغيل التعاملات (Rerun) التى تمت بعد عملية النسخ إلى وقت حدوث العطل.

- **سلامة و تكامل التعامل (Transaction Integrity)**

يجرى تعديل قاعدة البيانات عند تنفيذ التعاملات التى تغير سجلاً أو أكثر. و عند حدوث عطل أدّى إلى عدم اكتمال التعامل من بدايته حتى نهايته أثناء تنفيذ المعاملة يجب إجراء الاستعادة بتنفيذ الأمر Rollback أى التراجع عن كافة تأثيرات التعامل. فى حال اكتمال التنفيذ من بدء التعامل حتى نهايته، يجرى تأكيد تأثيرات التعامل فى سجلات قاعدة البيانات باستخدام الأمر Commit أى تأكيد التنفيذ.

١٠-١-٤ أنواع فقد البيانات Data Lost Types

يتراوح فقد البيانات ما بين إدخال قيم غير صحيحة لبيانات معينة إلى الفقد الكامل لبيانات قاعدة البيانات. هناك عدة أنواع رئيسية لفقد البيانات و لكل نوع الطريقة المناسبة للاستعادة، نذكر بعضها فيما يلى :

- **تسجيل بيانات غير صحيحة**

عند تعديل قاعدة البيانات بإدخال بيانات غير سليمة و تم اكتشاف الخطأ، يتم تصحيح أخطاء يدوياً إن كانت بسيطة. أما إن كانت كثيرة، فإننا نستطيع استخدام عدة تقنيات منها : الاستعادة العكسية أو تعامل تعويضى/بديل (Compensation Transaction) أو البدء من آخر نقط فحص.

- **التعاملات المُجهضة (الغير مكتملة)**

فى هذه الحالة نستطيع استخدام تقنية إلغاء/التراجع عن كافة نتائج التعاملات الغير مكتملة (Rollback).

- **فَنَاء قاعدة البيانات (Database Destruction)**

فى هذه الحالة لا يمكن إجراء أى تصحيح و نستطيع استخدام تقنية واحدة و هى : الاستعادة بإعادة تحميل النسخة الاحتياطية ثم تنفيذ كافة التعاملات أى استعادة أمامية.

- **تعطُّل النظام مع سلامة قاعدة البيانات (System Failure)**

فى هذه الحالة يتعطَّل النظام مع الاحتفاظ بقاعدة البيانات سليمة (database intact). نستطيع استخدام عدة تقنيات للاستعادة منها : إلغاء/التراجع عن أخر تعاملات أو البدء من آخر نقط فحص.

١٠-٢ أمن قواعد البيانات Database Security

فى الفترة الأخيرة أدى انتشار شبكة الإنترنت و زيادة عدد مستخدميها و تمكينها للمستخدمين من تناول قواعد بيانات من بُعْد إلى خطورة حدوث تناول غير مشروع لقواعد بيانات خطيرة مثل قواعد بيانات المصارف و الحكومات و السفارات و الجوازات و غيرها من المعلومات الحساسة التى تكلَّفت عليها أموالاً ضخمة.

١٠-٢-١ مسئولية أمن قواعد البيانات
Database Security Responsibility

كلما زادت التعاملات مع قاعدة البيانات و زاد عدد المستخدمين كلما زادت الحاجة إلى اتخاذ العديد من الإجراءات التى تكفل سلامة و أمن البيانات من أى مستخدم ليس له الحق فى تناول تلك البيانات. لذلك تلجأ الجهات صاحبة قاعدة البيانات إلى عمل تحكمات أمنية للتحكم فى كل كبيرة و صغيرة فى قاعدة البيانات.

أمن قاعدة البيانات هى حماية قاعدة البيانات من الاستخدام الخطأ أو الإضرار المتعمَد بالبيانات. هذه العملية تقع ضمن مسئوليات مدير قاعدة البيانات (DBA). حيث يستطيع استخدام العديد من الوسائل و السياسات لحماية قاعدة البيانات.

١٠-٢-٢ وسائل تأمين قواعد البيانات Database Security Tools

على مدير قاعدة البيانات استخدام كافة الوسائل التى يتيحها نظام DBMS من أجل حماية البيانات. من هذه الوسائل :

- **الجداول الافتراضية أو التخيلية (VIEW)**

حيث يتعامل المستخدمون مع جداول افتراضية أو تخيلية دون التعامل مع الجداول القاعدية الأصلية. بهذه الطريقة نقيد حرية التعامل مع البيانات دون تعطيل عمليات الاستعلام و لا يشعر المستخدم بذلك مطلقاً.

- **قواعد الترخيص بالصلاحيات**

يستطيع DBA أن يمنح ترخيصاً بصلاحيات معينة لمستخدمٍ ما أو مجموعة من المستخدمين. هذه الصلاحيات تحدد نوع تناول البيانات (SELECT أم INSERT أم UPDATE أم ALTER أم مجموعة منهما) و جزء قاعدة البيانات (قاعدة البيانات بالكامل أو جدول معين أو سجل أو عمود أو عنصر بيانات) الذى يُطبَّق عليه هذه الصلاحيات. هذه الصلاحيات تحدد المستخدمين و الأعمال المسموح لهم بأدائها.

- **برامج تحجيم المستخدمين**

هذه البرامج تضيف قيوداً إضافية على المستخدمين أو تقلل من الصلاحيات حسب اسم المستخدم (User Name) و استخدام كلمة مرور (Passwords) خاصة بكل مستخدم لنظام المعلومات.

- **برامج التشفير/الترميز**

هى البرامج التى تحول البيانات إلى شكل غير ممكن التعرُّف عليه إلا بعد إعادته إلى الشكل الأصلى من خلال إجراء فكر الترميز.

١٠-٣ أسئلة

١. ما المقصود باستعادة البيانات و نقاط الاختبار و أمن قواعد البيانات و مفكرة النظام؟

٢. ماذا نعنى بالعمليات التالية فى الاستعادة: UNDO و REDO و Rollback و Rollforward و Backup و Commit ؟

٣. حدد أسباب فقد البيانات.

٤. ما هى التسهيلات التى يوفرها نظام DBMS للاستعادة؟ مع الشرح بإيجاز.

٥. ما هى طرق الاستعادة المتعارف عليها؟ مع توضيح كيفية عمل كل طريقة.

٦. ما هى أنواع فقد البيانات؟

٧. على من تقع مسئولية أمن قواعد البيانات و كيف يمكن القيام بذلك.

٨. وضِّح مع الشرح بإيجاز وسائل تأمين قاعدة البيانات.