
أساسيات نظم قواعد البيانات
Fundamentals Of
Database Systems

الفصل الأول

منهجيات
حِفظ وإسترجاع البيانات إلكترونيا

Data Storing & Retrival Approaches

المبحث الأول : منهج "الملفات المرتبطة بالتطبيقات".

المبحث الثاني: منهج "قواعد البيانات".

مقدمة :

إن كلمة : **البيانات Data** تعني – بكل بساطة – وصف مجرد لتمثيل الحقائق أو الأحداث أو الأشياء ، بشكل قابل للقياس والتسجيل ، سواء فى صورة رمزية Symbolic أو نصية Text أو رقمية.

فى اللغة الإنجليزية ، يُطلق لفظ : **Datum** على صيغة المفرد Singular ، أي على أي قطعة بيانات منفردة ، مثل : رقم التليفون الخاص بأحد العملاء ، أو شعبة التخصص لأحد الطلاب ، أو المبلغ الاجمالي لإحدى فواتير الشراء. أما لفظ : **Data** فيطلق على صيغة الجمع Plural ، أي على تشكيلة من قطع البيانات ذات الصلة أو المعنى الضمني Implicit Meaning.

إن كافة نظم المعلومات الإلكترونية EIS التي توجد في التنظيمات المعاصرة ، وهى التي كانت تسمي في الماضي "نظم المعلومات المبنية علي الحاسبات CBIS" تقوم في صميمها وجوهرها علي عملية حفظ وإسترجاع البيانات إلكترونيا. وينطبق ذلك سواء كان النظام محل الإهتمام هو نظام المعلومات الإدارية MIS أو نظام دعم القرارات DSS أو نظام معالجة العمليات والأنشطة TPS .... فالمشكلة مع أي نظام من هذه النظم كانت – ولسوف تظل – هي : كيف يمكن حفظ وإسترجاع البيانات اللازمة لتلبية احتياجات المستخدمين المختلفين بالشكل الذى يحقق أعلي مستويات الكفاءة والسرعة؟

فـــي هذا الخصوص ، عرف تاريخ تكنولوجيا المعلومات منهجيتان مختلفتان لحل مشكلة حفظ وإسترجاع البيانات إلكترونيا ، هما :

- ⇦ **منهج الملفات المرتبطة بالتطبيقات** The Application-Oriented Approach.
- ⇦ **منهج قواعـــد البيانـــات** The DataBase Approach.

وتـــبعا لذلك ، فإن الدراسة في هذا الفصل تنقسم إلى مبحثين ، يختص كل مبحث مـــنهما بعـــرض أحـــد هذين المنهجين ، علي الترتيب. هذا العرض يوضح مضمون أو طبـــيعة كل منهج ، ويتعرف علي خصائصه المـــيزة ، ويستخلص أهم المزايا ونواحي القصور التى أسفرت عنها تلك الطبيعة والخصائص.

* * *

١ المبحث الأول

منهج الملفات المرتبطة بالتطبيقات

The Application – Oriented Approach

لقد تواجد هذا المنهج العتيق ، من الناحية التاريخية ، مع بداية العهد بالتشغيل الإلكتروني للبيانات Data Processing. واشتهر لدى الكتّاب فى البداية باسم : The Traditional File Processing Approach ... أي المنهج التقليدي لمعالجة الملفات. ثم أصبح يعرف فى الوقت الحاضر باسم أكثر دلالة على جوهره ، هو : Application-Oriented Approach ، أى **منهج الملفات المرتبطة بالتطبيقات.**

إن العيوب الجسيمة التى اقترنت بهذا المنهج وصاحبت تطبيقه كانت السبب فى اندثاره فى الوقت الحاضر ، لكن قيمته الحقيقية ظلت فى أنه هو الذى وضع الأساس ، ومهد الطريق ، لظهور المنهج المعاصر : منهج قواعد البيانات.

أولا) جوهر المنهج :

يقوم هذا المنهج التقليدي على فكرة جوهرية ، هى تبعيّة البيانات لبرامج التطبيقات التى تستخدمها : Application/Data Dependence. بمعنى : أن كل برنامج

تطبيقي Database Application يستقل فى ذاته بملفات البيانات اللازمة لتشغيله لتصبح جزءا لا يتجزأ من هيكله ... حتى ولو كان بعض هذه البيانات عاما أو مشتركاً ، يتكرر حرفيا فى ملفات البيانات التابعة لأكثر من برنامج تطبيقي آخر فى نفس الوقت.

على سبيل المثال ، يحتاج المسجل (أو مسئول شئون الطلاب) فى إحدى الكليات الى الاحتفاظ بملفات بيانات توضح القيد الدراسي لكل طالب وموقفه التعليمي ، بينما يحتاج المحاسب فى نفس الكلية الى بيانات عن المصروفات الدراسية المستحقة على كل طالب والمبالغ المتحصلة منها. وعلى الرغم من أنهما يهتمان كلاهما ببيانات خاصة بنفس الطلاب ، إلا أن كلا منهما يحتفظ فى إدارته بملفات بيانات منفصلة بالإضافة الى برنامج تطبيقي خاص لمعالجة واسترجاع تلك البيانات.

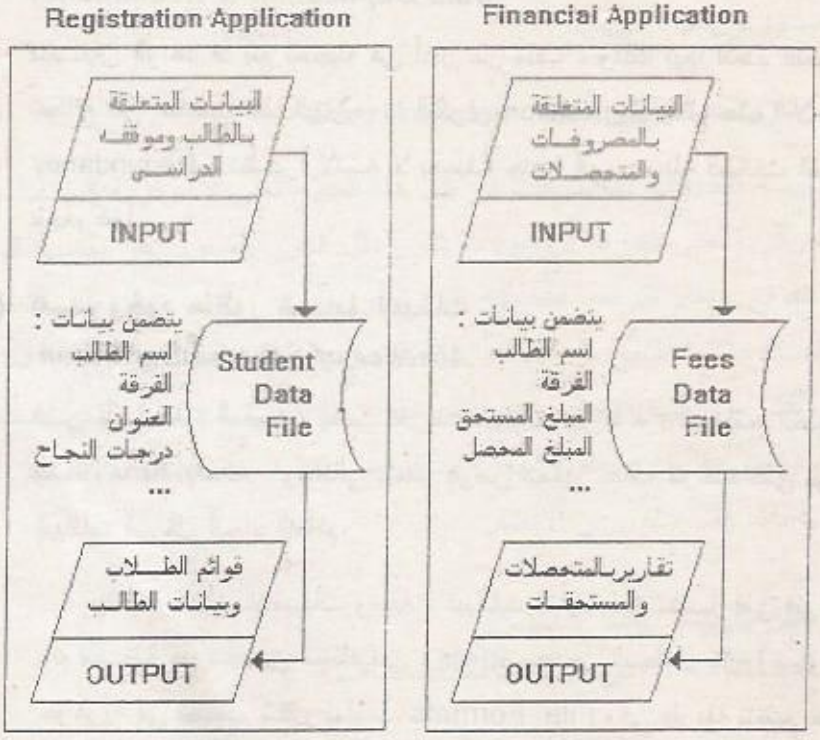
ولا شك أن السبب الرئيسي فى هذا التشتت والانفصال هو أن كلا منهما (أي مسجل الكلية ومحاسبها) يحتاج الى بعض قطع أو مفردات البيانات التى قد لا توجد طرف المستخدم الآخر. فمثلاً :

- إن بيانات درجات وتقديرات النجاح تعتبر هامة بالنسبة للمسجل ، ولكنها قد لا تعتبر ذات قيمة أو منفعة ، ولا تخدم أي هدف ، بالنسبة للمحاسب.
- وبالعكس ، فإن بيانات الرسوم الدراسية المستحقة على كل طالب ، واشتراكات الطلاب فى السيارات ، وبيانات الإعفاءات من المصروفات تعتبر بيانات حيوية بالنسبة للمحاسب ، ولكنها لا تحمل أية قيمة أو أهمية بالنسبة للمسجل!

لكن استقلال كل تطبيق ببياناته ، واحتوائه عليها بداخله ، وبشكل منفصل عن البيانات المتاحة فى التطبيقات الأخرى ، قد ترتب عليه ظهور مشكلة جوهرية ، هى : تكرار بعض البيانات Duplication of data فى التطبيقات المختلفة بداخل نفس التنظيم. فبالرجوع الى المثال السابق ، يتضح أن بيانات مثل : اسم الطالب ، وفرقته الدراسية، وشعبة التخصص ، يتم تخزينها مرتين :

- مرة أولى ، فى ملف بيانات الطلاب التابع لبرنامج التسجيل.
- ومرة أخرى ، فى ملف بيانات المصروفات التابع للبرنامج المالي.

وإتماما للرؤية ، فإن الشكل التالي ، شكل رقم (١-١) ، يعرض نموذجاً مبسطاً لنتائج تطبيق منهج الملفات المرتبطة بالتطبيقات فى هذه الحالة. ومنه ، يلاحظ أن كل برنامج تطبيقي يستقل بملف أو ملفات البيانات الخاص به ، رغم أن بعض قطع البيانات يتكرر تسجيلها فى الملفات بين تطبيق وآخر.



شكل رقم (١-١)

نموذج مبسط لتطبيق منهجية الملفات المرتبطة بالتطبيقات

ثانيا) خصائص منهج الملفات المرتبطة بالتطبيقات

إن بناء منهج الملفات المرتبطة بالتطبيقات على مفهوم تبعية الملفات للتطبيقات التى تستخدمها قد أورثه مجموعة من الخصائص التى انتهت به الى التراجع والاندثار. من أهم هذه الخصائص نذكر ما يلى :

(١) تكرار وحشوْ البيانات بغير ضرورة
Data Duplication & Redundancy

فالبيان الواحد قد يتم تسجيله فى أكثر من ملف ، وذلك تبعا لتعدد التطبيقات التى تحتاج الى استخدام هذا البيان. ذا التكرار Duplication يمثل حشواً لا فائدة منه Redundancy ، نظرا لأنه لا يضيف جديدا الى حصيلة البيانات المخزونة فى مجموعها.

(٢) عدم وجود منظور شامل للبيانات
Absence of Schema Definition

فى ظل هذا المنهج ، يعتبر تعريف البيانات جزءا من البرنامج التطبيقي ذاته ، ومحكوما باحتياجاته ، وبالتالى يتعذر فرض معايير عامة موحدة على طريقة تمثيل البيانات فى كل أنحاء النظام.

ونظرا لأن التطبيقات وملفات البيانات التابعة لها تنشأ على فترات متباعدة وبواسطة مبرمجين مختلفين ، فإنه يصبح من المحتمل كثيراً وجود اختلافات جوهرية فى تصميم شكل الملفات File Formats وفى طريقة تنظيم حفظ الملفات وأساليب الوصول إليها واسترجاعها ... من تطبيق الى آخر. وبالتالى فإن هذا المنهج يفتقد الى وجود تعريف أو منظور شامل للبيانات على مستوى المنشأة أو التنظيم ككل.

(٣) تعـقد مشكلات حماية وأمن النظام *Security Problems*

مـن المفـروض أن لا يسمح النظام لكل مستخدم بالوصول الى كافة البيانات ، أو بإجـراء كافـة العمليات عليها (الإطلاع ، الإضافة ، التحديث ، الحذف ...) ، بل يجـب وضـع قيود تحدد لكل مستخدم ما الذى يستطيع الوصول إليه من البيانات ومـدى سـلطته فـى إجـراء تعديلات عليها ، وذلك تبعا لاختصاصاته الوظيفية ومسـتواه فـى الهيكل التنظيمي. ولكن ، نظرا لأن التطبيقات الجديدة تضاف الى الـنظام بشـكل منفصـل ، وعلى فترات متباعدة ، فإنه يصبح من العسير تنميط وتعمـيم مثل هذه القيـود.

(٤) عُـزلة وانفصال البيانـات *Data Isolation*

تبعـيـة البـيانات للتطبـيقات فى ظل هذا المنهج تعنى **لا مركزية البيانات**. هذه اللامركزية تعنى بدورها أن البيانات تكون مثل الجزر المنعزلة عن بعضها البعض وتـتقوقع بداخل البرامج التطبيقية التى تحتوى عليها. وكلما تزايد عدد التطبيقات القائمـة ، كلما أصبح من الصعب تلبية الاحتياجات المستجدة من البيانات المتناثرة بين أكثر من تطبيق ، وتعذر وتـأخر تطوير التطبيقات الجديدة اللازمة لاستخراج التشكيلات Views المطلوبة منها ، وذلك نظرا لكثرة وتعـقد التفاعلات التى يجب أن يأخذها المبرمجون فى الاعتبار.

ثالثا) مـزايا وعـيوب منهـج الملفات المرتبطة بالتطبيقات

ولعل من أهم العيوب ونواحي القصور التى عاني منها منهج الملفات المرتبطة بالتطبيقات ، ما يلي :

(١) سوء استخدام الموارد المتاحة
Inefficient use of Recourses

فتسجيل البيان الواحد أكثر من مرة فى أكثر من تطبيق ، وما يعنيه من الحشو والتكرار ، ينتج عنه الكثير من مظاهر سوء استغلال الموارد ، نذكر منها :

- إنفاق تكلفة إضافية غير مبررة اقتصاديا تتمثل فى قيمة الجهد والوقت الضائع فى إعادة إدخال وتحديث نفس قطع البيانات عدة مرات فى التطبيقات المختلفة.
- تبديد جزء لا يستهان به من الطاقة التخزينية المتاحة لحفظ الملفات ، دون مبرر.

(٢) عدم تجانس البيانات
Data Inconsistency

إن تكرار تسجيل وتحديث نفس البيان فى أكثر من تطبيق يكون مصحوبا فى الغالب بعدم تطابق النسخ المختلفة من هذا البيان بين تطبيق وآخر ، الأمر الذى قد يترتب عليه اتخاذ قرارات غير سليمة أو قرارات متضاربة. ويرجع عدم التجانس فى البيانات على هذا النحو الى عدة أسباب ، منها :

- حدوث أخطاء عند إدخال البيان نفسه فى أحد التطبيقات دون الأخرى.
- اختلاف توقيت عمليات تحديث البيان نفسه بين التطبيقات المختلفة.

(٣) عدم المرونة في تصميم وتطوير التطبيقات
Inflexibility in Application Development

وهذه المشكلة لها مظاهر ومسببات متعددة :

- فتبعية البيانات للتطبيقات تعنى فى الوقت ذاته أن كل تطبيق يكون مقيدا بالعمل فقط مع البيانات التى يفصح عنها هيكله. ويترتب على ذلك أن

إجراء أي تغيير فى هيكل أي ملف من ملفات البيانات سوف يتطلب إجراء تعديلات فى البرنامج الذى يستخدم هذا الملف ، كما أن تعديل أي برامج تطبيقي من البرامج القائمة سوف يستتبع غالبا إجراءا تعديل مناظر فى هيكل ملفات البيانات التابعة له.

- كما أن عزلة البيانات ، وعدم وجود معايير موحدة تحكم طريقة تمثيلها ، وتباين أشكال تصميمها من تطبيق الى آخر ... يجعل من الصعب على المبرمج كتابة تطبيقات جديدة لاسترجاع تشكيلة بيانات محفوظة داخل عدة ملفات تابعة لتطبيقات مختلفة.

- فضلا عن أن الوقت المستغرق فى إنشاء تطبيقات جديدة يصبح طويلا بدرجة لا يستهان بها ، وذلك بسبب عدم وجود منظور شامل للبيانات المتاحة على مستوى التنظيم ككل.

(٤) انخفاض معدل الأداء الشامل للنظام
Low Overall Performance

إن الأثر النهائي لكافة نواحي القصور السابقة ، من مظاهر للإسراف والضياع وسوء استخدام الموارد ، ومن تباطؤ وتعقد وعدم مرونة عمليات تصميم وتطوير البيانات والتطبيقات ، بالإضافة الي عدم توافق وتكامل البيانات... إنما ينعكس بدرجات متفاوتة من الشدة على مقاييس الأداء الشامل للنظام ، وذلك في صور متعددة ، نذكر منها :

- ارتفاع الأعباء المالية لإدارة وصيانة النظام.
- البطء المتزايد في الإنجازات كلما كبُر النظام وأصبح أكثر تعقيداً.
- تناقض وعدم صحة القرارات المبنية على النظام.

... لـتكون المحصلة النهائية لذلك كله هى عجز النظام بأسره عن تحقيق التوازن المنشود بين تكلفة تشغيله وصيانته ، وبين المنافع المرجوة منه.

ويتضـح مـن ذلـك كلـه ، أن إغراق هذا المنهج في الدوران حول محور تبعية وانعزالـية البـيانات ، وافـتقاره الى منظور شامل لهيكل البيانات ، والي معايير موحدة لتصميم الملفات ، قد دفع بـه الى الذبول والانحسار مع عالم التطبيق.

ومـع ذلك ، ومع الإعتراف بهذه العيوب الجوهرية ، فإنه يمكننا أن نحسب لهذا المنهج التقليدى عددا من الحسنات القليلة ، نوجزها فيما يلي :

- أن هـذا المـنهج يسمح – بكل سهولة – بالمحافظة علي سرية البيانات ، ولكن بالمعـنى الضـيق لهذه الكلمة. ويرجع ذلك إلى أن البيانات لا تخضع للمشاركة بواسطة مستخدمين آخرين.
- وتظـل قيمة هذا المنهج الحقيقية هى أنه مهد الطريق لنشأة وتطور منهج قواعد البـيانات ... وهـو الـذى وضع نصب عينية منذ البداية كيف يمكن تجنب هذه العثرات والمشكلات والعيوب.

* * *

المبحث الثاني

منهج قواعد البيانات

The DataBase Approach

إن المنهجية الحديثة السائدة فى الوقت الحاضر كوسيلة لحل مشكلة حفظ وإسترجاع البيانات إلكترونيا – بكفاءة وسرعة – هى المعروفة إصطلاحيا باسم : **منهج قواعد البيانات The Database Approach**. فى استعراضنا لجوهر وخصائص ومزايا وعيوب هذا المنهج ، سوف نتوخى نفس الإطار المتبع فى المبحث السابق ، وذلك حرصا على توفير عامل القابلية للمقارنة.

أولا) جوهر منهج قواعد البيانات

يقوم منهج قواعد البيانات على منطق واضح ومحدد ، هو أن تجنّب ظاهرة تكرار البيانات وما يقترن بها من مشكلات وعيوب لحقت بالمنهج السابق إنما يتحقق عن طريق تطبيق مبدأ : استقلالية البيانات عن التطبيقاتApplication/Data Independence . هذا المبدأ يقضى بأن برامج التطبيقات يجب أن تكتب بشكل مستقل عن أية ملفات محددة للبيانات . وأن البيانات اللازمة لكافة هذه التطبيقات يجب تخزينها فى مستودع شامل وبشكل مستقل عن برامج التطبيقات!

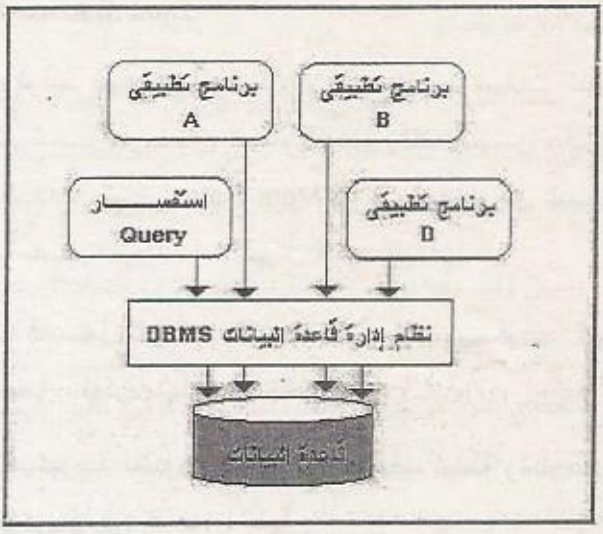
إن التطبيق العملي لمبدأ استقلالية البيانات عن التطبيقات يقتضي العمل على توفير دعـامتيـن أساسيتين ، تتحلّق حولهما كافة مفاهيم وأساليب وتقنيات منهج قواعد البيانات ... وبدون أي منهما أو كلتاهما لا تقوم له قائمة. هاتان الدعامتان هما :

(١) تخزين كافة البيانات اللازمة لتلبية احتياجات كافة الاستفسارات وبرامج التطبيقات ، فـي وعـاء عـام : Common Pool ، أو في مستودع شامل : Repository ، يـتـم فيـه تسجيل البيان الواحد مرة واحدة. وتكون حصيلة البيانات المخزونة فـي هـذا المسـتودع متاحة لخدمة احتياجات كافة المستخدمين ... على سـبيل المشاركة Shareable Base. هذا الوعاء الشامل هو الذى يشار إليه باسم : **قاعدة البيانات Data Base** .

(٢) إخضـاع هـذا المسـتودع الشامل ، أي قاعدة البيانات ، لرقابة وسيطرة برنامج تطبيقي وسيط ومتخصص في حفظ وإسترجاع البيانات إلكترونيا ، هو الذى يطلق عليـه: **نظـام إدارة قاعـدة البيـانات Data Base Management System** ، واختصاره DBMS. المهمة الرئيسية لهذا البرنامج الوسيط ، هى توفير الواجهة الـتى تكفل التفاعل بين استفسارات وبرامج تطبيقات المستخدمين من جهة ، وبين البيانات المخزّنة فى قاعدة البيانات من جهة أخرى.

وإتمامـا للفائدة ، فإن الشكل التالى ، شكل رقم (١-٢) ، يصور لك مضمون مبدأ استقلالية البيانات والتطبيقات في منهج قواعد البيانات ، ومنه يتضح ما يلي :

- أن كافة البيانات اللازمة لتلبية احتياجات المستخدمين المختلفين يتم تخزينها ، مرة واحدة ، فى قاعدة البيانات.
- أن نظـام إدارة قـاعدة البـيانات هو يقوم بعمليات حفظ وإسترجاع ومعالجة هذه البـيانات ، وهوالـذى يعمـل على تمكين المستخدمين من الوصول الى البيانات المخزونة فى القاعدة عن طريق الاستفسارات وبرامج التطبيقات.



شكل رقم (١-٢)

استقلالية البيانات والتطبيقات
فى ظل منهج قواعد البيانات

ثانيا) خصائص منهج قواعد البيانات

هـناك مجموعة من الخصائص البارزة التى تميز منهج قواعد البيانات عن منهج الملفات المرتبطة بالتطبيقات. هذه الخصائص تتمثل ، بإيجاز ، فيما يلي:

١) وحدة البيانات وعدم تكرارها *Data Un-Redundancy*

فالبيان الواحد يسجل **مرة واحدة فقط** فى كل أنحاء القاعدة ... مهما كان عدد التطبيقات أو الاستفسارات المستخدمة له أو المبنية عليه. وبالتالى فإن هذا المنهج يستبعد كافة صور تكرار وحشو البيانات غير المرغوب فيه ، وما يرتبط بها من سلبيات ومشكلات.

(٢) وجود وصف شامل لهيكل البيانات
Data Schema Definition

كما يتميز منهج قواعد البيانات بأن النظام لا يتضمن فقط البيانات المتعلقة بوصف الأحداث التى تدخل فى نطاق اهتمام القاعدة ، لكنه يتضمن علاوة عليها نوعاً آخر من البيانات ، يسمى Meta-Data ، وهى "**بيانات عن البيانات**" . هذه البيانات الوصفية تقدم تعريفا شاملا لهيكل قاعدة البيانات.

التعريف الشامل للبيانات على هذا النحو فى منهج قواعد البيانات يخضع لعدة اعتبارات هامة ، نذكر منها بوجه خاص :

أ – أن هذا التعريف سوف يتضمن معلومات شاملة ومتنوعة عن **البيانات** المسموح بتخزينها فى القاعدة ، منها :

- هيكل كل ملف File Structure من ملفات البيانات ، والعلاقات بينها.
- نوع وشكل وعنوان ، كل مفردة بيانات Data type and Format .
- تعريف القيود والتحفظات Constraints الواردة على عمليات استرجاع وتحديث البيانات ، والتى توضع بغرض ضمان صحة وتوافق وتكامل محتويات القاعدة فى كل وقت.

ب– أن الوصف الشامل لهيكل القاعدة الذى ينعكس فيه هذا التعريف يطلق عليه اصطلاحيا اسم : **منظور البيانات** Data Schema.

ج – أن اللغة المستخدمة فى كتابة هذا المنظور ووصفه للنظام هى لغة من اللغات الخاصة بنظم إدارة قواعد البيانات ، هى **لغة تعريف البيانات** DDL .

د – أن الوصف الناتج عن هذا التعريف سوف يتم تخزينه فى ملف خاص من ملفات النظام داخل قاعدة البيانات ، يطلق عليه اسم : **قاموس البيانات** Data Dictionary ، أو **كتالوج النظام** System Catalog.

(٣) قابليـة البيانات للمشـاركة
Data is Shareable

مـــن أبرز الخصائص الأساسية المميزة لمنهج قواعد البيانات أن البيانات المخزنة فى قاعدة البيانات تكون قابلة للمشاركة ، سواء من جانب تطبيقات مختلفة ، أو من جانب مستخدمين مختلفين (فى شكل استفسارات Query).

وينبـــنى علـــى قابلـــية البيانات للمشاركة خاصيتان إضافيتان على جانب كبير من الأهمية المنهجية ، هما :

أ) **قدرة النظام على استخراج تشكيلات متنوعة من نفس البيانات المخزنة :**

فـــى ظـــل قابلـــية البيانات للمشاركة ، من المتوقع أن يحتاج كل تطبيق من التطبيقات ، أو كل مستخدم من المستخدمين ، على حدة ، إلى منظور مختلف Perspective أو **تشـــكيلة** خاصـــة View من البيانات المخزنة فى القاعدة ، هـــى وحدها التى تتناسب مع احتياجاته. لهذا السبب فإن من خصائص منهج قواعد البيانات قدرة النظام الفائقة على استخراج تشكيلات متعددة ومتباينة من نفـــس البـــيانات المخـــزنة بالقاعدة ، حسب احتياجات المستخدمين ، دون أن يشـــــغل أذهانهم بتفاصيل معقدة ومملة عن كيفية تخزين ومعالجة واسترجاع هذه البيانات.

ويلاحظ أن محتويات كل **تشكيلة** من تشكيلات البيانات المستخرجة :

- قـــد تقتصر **فقط** على استرجاع مجموعة جزئية Subset من البيانات المخزّنة ، سواء كانت هذه البيانات محفوظة أصلا فى ملف واحد أو فى عـــدد مـــن ملفات قاعدة البيانات. ومثال ذلك : حالة استرجاع تشكيلة بيانات تضم معاً : اسم مندوب البيع والقسم التابع له من ملف الموظفين Employee Data File ، بالإضافة الى : مقدار راتبه الشهري من ملف المرتبات Salary Data File.

- وقـد تحتوى على **بيانات افتراضية** Virtual Data يتم استخلاصها عند الطلب من بيانات مخزّنة فى تلك الملفات. ومثال ذلك : استخراج مقدار العمولـة الـتى يسـتحقها هـذا المندوب (بيان غير مخزّن من قبل) ، بالاعـتماد علـى جملة صفقات البيـع التى أبرمها خلال فترة معينة ، (بـيانات مخزنة فعلا فى ملف المبيعات Sales File ، أو ملف طلبيات البيـع المنفـّذة Sales Orders).

ب) **إمكانية العمل بنظام تعـدد المستخدمين Multi-Users :**

يقصد بنظم التشـغيل متعددة المستخدمين Multi-User Systems ، كما يدلّ علـيه اسـمها ذاتـه ، قـدرة النظام على السماح لمجموعة من المستخدمين المختلفيـن بالوصول الى قاعدة البيانات ومعالجة محتوياتها فى ذات الوقت. بـيد أن هـذه القدرة لن تتحقق إلا فى ظل منهج قواعد البيانات ، حيث أنها تتطلـب إدمـاج كافـة البيانات اللازمة لمثل هذه العمليات الآنيـّة فى قاعدة بـيانات وحـيدة A Single Database ، يتم تصميمها وفقا لمفاهيم وتقنيات هذا المنهج بالذات.

ومـن الجديـر بالذكـر فى هذه الحالة ، أنه يجب أن يتضمن النظام برنامج متخصـص فى الرقابة الآنيّة أو الفورية Concurrency Control Software .[1] هذا البرنامج سوف يعمل على مراقبة المستخدمين المتعددين الذين يحاولون تحديث نفس قطع البيانات ، وذلك من أجل ضمان صحة التعديلات وعدم تعارضها.

---

(١) للتوسع فى هذه النقطة ، راجع بوجه خاص :

- A. Bernstein, V. Hadzilacos, & N. Goodman, ***Concurrency Control and Recovery in Database Systems,*** (Readings, MA : Addison-Wesley Publishing Co., 1987).

ومـــثال ذلـــك ، عندما يحاول مسئولو بيع التذاكر ، فى عدد من المكاتب المتباعدة جغرافـــيا التابعة لإحدى شركات الطيران ، حجز مقعـــد معين بالطائرة عن رحلة معيـــنة لعدة ركاب مختلفين فى نفس الوقت. إذ أنه بدون وجود برنامج للرقابة الآنيـــة ، يصبح من المحتمل جداً حجز المقعد نفســـه لأكثر من راكب واحد على نفس الرحلة.

(٤) احتياطات تأمين وحماية البيانات
Security Restrictions

فـــى ظـــل مركـــزية البيانات Centralization وقابليتها للمشاركة فى منهج قواعد البـــيانات ، يكـــون من الضروري التحقق من أن الوســـيلة الوحيدة للوصول الى البـــيانات هى ، وفقط هـــي ، الوصول من خلال القنوات الصحيحة أو الشرعية. لذلـــك ، فإن من الخصائص المميزة لهذا المنهج أنه يسمح بتعريف احتياطات دقيقة وكافـــية لحماية وتأمين البيانات ضد محاولات الوصول أو التعديل غير المشروع من جانب أشخاص غير مصرح لهم بذلك.

وتشمل هذه الاحتياطات بوجه خاص :

- وضـــع قواعـــد واضحة وصريحة لتقييد حـــرية المستخدمين المختلفين فى الوصول الى تشكيلات البيانات ، ومراقبة الالتزام بتطبيق تلك القواعد.
- تقـــييد نوع العمليات التى يمكن لمستخدم معين إجراؤها على كل قطعة من البيانات لكل مستخدم على حدة ، (الاسترجاع ، التحديث ، الحذف...).

(٥) إمكانية المحافظة علي تكامل البيانات
Integrity Restrictions

أيضـــا ، من أهم الخصائص المميزة لنظم إدارة قواعد البيانات فى ظل هذا المنهج أنهـــا تســـمح بتعريف ، وفرض الالتزام بتطبيق ، العديد من الاشتراطات والقيود

والتحفظات التى تعمل على مراقبة كل عملية من عمليات معالجة البيانات ، للتحقق من سلامتها ، وبالتالى ضمان المحافظة علي تكامل البيانات المخزونة في القاعدة.

من أهم صور هذه القيود والتحفظات ، ما يلي :

- عدم السماح بتخزين قطعة بيانات جديدة أو معدّلة بأحد ملفات القاعدة إلا إذا كانت من نفس نوع البيان المحدد مقدما Data Type (بيانات نصّية Text ، أو رقمية Numbers ، أو تاريخ Date ، أو صورة Graph ..). فلا يسمح النظام مثلا بإدخال حروف أبجدية فى حقل مخصص لتسجيل التواريخ ، أو إدخال حروف أبجدية فى حقل مخصص لتسجيل أرقام.
- عدم السماح بإضافة سجل جديد الى أحد ملفات القاعدة إلا إذا كان هذا السجل مرتبطاً بسجل آخر تم تسجيله من قبل فى أحد الملفات المرتبطة. ومثال ذلك : رفض النظام تسجيل خصم نقدي على قيمة إحدى فواتير المبيعات إلا إذا كانت هذه الفاتورة مقيدة فعلا بملف العميل وقام بتسديدها فى المهلة المحددة للتمتع بهذا الخصم.

ثالثا) مزايا وعيوب منهج قواعد البيانات

في الحقيقة ، إن المزايا التى تتحقق للمنشآت من تطبيق منهج قواعد البيانات تجلُّ عن الحصر. ويمكننا تلخيص أهم هذه المزايا فيما يلي :

(١) حُسن استخدام الموارد المتاحة
Efficient Use of Resources

إن منهج قواعد البيانات هو الحل الإقتصادى الأمثل لمشكلة حفظ وإسترجاع وإدارة البيانات إلكترونيا ، فى المنشآت الكبيرة أو الصغيرة علي السواء. من أهم مظاهر حسن إدارة الموارد وكفاية استخدامها فى ظل تطبيق هذا المنهج ما يلي :

أ) **أن البـيـــان الواحد يسجل مرة واحدة فى كل أنحاء القاعدة.** وذلك ، مهما تعـــددت الاستفســـارات وبـــرامج التطبيقات التى تحتاج إليـــه. وبالتالى فإنه يســـاعد علي تجنب الكثير من مظاهر سوء استخدام الموارد الملازمة لمنهج الملفات المرتبطة بالتطبيقات ، ومنها :

- تجنب الوقت الضائع فى تسجيل وتحديث نفس البيان أكثر من مرة.
- تجنب الفاقـــد فى الطاقة التخزينية ، نظرا لاختصار حجم الملفات.
- تجنـــب تضارب نفس البيان بين تطبيق وآخر بسبب أخطاء الإدخال أو اختلاف توقيت عمليات التحديث.

ويتـــرتب على النتائج عدة مزايا إضافية ، نذكر منها :

- توفير سرعة أكبر فى الوصول الى البيانات المطلوبة.
- تحقيق اقتصاد ملحوظ فى تكلفة تشغيل وإدارة القاعدة ككل.

ب) **الاستفادة بوفورات الحجم الكبير Economics of Scale.** ويرجع ذلك إلى أن مركـــزية البيانات داخل قاعدة واحدة على مستوى المنشأة ككل إنما تسمح بتحقيق المزايا التالية :

- انخفـــاض ملموس في التكلفة الاستثمارية ، حيث أن تكلفة إقتناء الأجهزة والتجهيزات اللازمة للنظام ككل تقل كثيرا عن التكلفة التى كانت لتتحملها المنشـــأة فيما لو تُرك لكل إدارة من الإدارات أن تقتنى التجهيزات اللازمة لتشـــغيل تطبيقاتها الخاصة بشكل منفصل ، كما كان يحدث فى ظل منهج الملفات المرتبطة بالتطبيقات.
- انخفـــاض ملمـــوس فـــى الأعباء الخاصة بإدارة وصيانة قاعدة البيانات ، نظرا لتوفير الوقت الضائع في تكرار تسجيل نفس البيانات.
- انخفاض ملموس فى تكاليف التشغيل الكلية للنظام.

ج) **تحقيق التوازن بين الاحتياجات المتعارضة.** فمن مظاهر حسن استخدام وإدارة الموارد المتاحة أيضا أن منهج قواعد البيانات يسمح بتكوين نظرة شاملة عن احتياجات التنظيم ككل من البيانات ، نظرة لا تقتصر على الرؤية الفردية المحدودة والضيقة لاحتياجات كل مستخدم على حدة – فى عزلة عن غيره من المستخدمين. وبالتالى ، فإنه يمكن من تصميم هيكل قاعدة البيانات علي النحو الذى يوفر أفضل مستوى من الخدمة الشاملة ، والتضحية بما عدا ذلك من احتياجات قليلة الفائدة أو الأهمية.

إن المنطق الذى يبرر هنا التضحية ببعض الاحتياجات يستند الى مبدأ اقتصاديات المعلومات. فالبيانات تعتبر أصلاً من الأصول An Asset ... ولا يجوز من الناحية الاقتصادية أن تكون تكلفة اقتناء البيانات اللازمة للوفاء باحتياجات معينة أكبر من المنفعة المترتبة على توفير هذه البيانات. وكلما كانت بيانات معينة خاضعة للمشاركة من جانب عدد كبير من المستخدمين ، كلما كانت قيمتها أكبر.

(٢) تجنُب عدم توافق البيانات
Avoiding Inconsistency

عندما يتكرر نفس البيان أكثر من مرة – كما كان هو الحال فى ظل منهج الملفات المرتبطة بالتطبيقات – فإن فرصة حدوث **عدم التوافق** فى البيانات المخزَّنة Data Inconsistency تكون كبيرة. ويرجع ذلك الى سببين رئيسيين :

- اختلاف توقيت عمليات تحديث هذا البيان فى التطبيقات المختلفة .
- حدوث أخطاء فى تحديث ذات البيان فى أحد الملفات دون الأخرى.

هذه المشكلة وآثارها السلبية يتم استبعادها تماما فى ظل منهج قواعد البيانات ، لأن فلسفة هذه المنهج تمنع من الأصل حدوث التكرار غير الضروري

للبيان الواحد ، مهما تعددت التطبيقات المستخدمة له ، وبالتالي فانه يضمن تجانس وتوافق البيانات ... وصحة ما يبنى عليها من قرارات .

(٣) مرونة وسرعة تعديل النظام
System Flexibility & Timely

في التنظيمات المعاصرة ، تتميز احتياجات المستخدمين بأنها قابلة للتغيير مع مرور الزمن ، كنتيجة لتغير الظروف المحيطة بالمنشأة ، أو للتوسعات المستمرة في الأنشطة. الاستجابة لهذه التغيرات تتطلب غالبا أحد أو بعض التصرفات التالية : إجراء تعديلات فى التطبيقات القائمة ، تصميم وتطوير تطبيقات جديدة مستحدثة ، إدخال تعديلات طفيفة أو جوهرية على هيكل قاعدة البيانات تتضمن إضافة ملفات جديدة أو توسيع مشمول الملفات القائمة.

من المزايا البارزة لمنهج قواعد البيانات أنه يمكن من تنفيذ أي تعديلات جديدة في البيانات أو التطبيقات بسرعة ومرونة :

- فعندما يتم تصميم وإنشاء وتشغيل قاعدة البيانات في إحدى التنظيمات ، فإن الوقت اللازم لتصميم تطبيقات جديدة أو تطوير التطبيقات القائمة بالفعل ينخفض بدرجة كبيرة ، وذلك بسبب التسهيلات العديدة التي يوفرها وجود قاعدة البيانات ونظام إدارة قاعدة البيانات DBMS. ويقدّر الخبراء أن طول الفترة التى يستغرقها إنشاء أي تطبيق جديد باستخدام تلك التسهيلات ينخفض بنسبة تتراوح بين ٧٥% و ٨٥% من الوقت اللازم لإنشاء أو تطوير ذات البرامج التطبيقية فى ظل منهج الملفات المرتبطة بالتطبيقات!!
- كما أن استقلالية البيانات عن التطبيقات توفر المرونة الكاملة فى إجراء مثل هذه التعديلات ، حيث أنه يمكن تعديل هيكل البيانات دون أن تتأثر بذلك برامج التطبيقات القائمة بالفعل.

ومـــن جهـــة أخـــرى ، فإنه يمكن تعديل برامج التطبيقات القائمة وتطوير برامج تطبيقات جديدة دون أن تتأثر بذلك البيانات المخزنة فى القاعدة.

ويترتب على مرونة تعديل البيانات والتطبيقات وما يقترن بها من إنخفاض جوهرى فى وقت تطوير التطبيقات الجديدة ، المزايا التالية :

- ضمان سرعة الاستجابة للتغيرات البيئـــية ، والتكيف معها.
- تلبية الاحتياجات المستحدثة فى الوقت المناسب Timely Response.
- انخفاض تكلفة تصميم واختبار وتوثيق التطبيقات الجديدة.

(٤) ارتفاع المعدل الشامل لأداء النظام
Good Overall Performance

إن تقييم الأداء الشامل لنظام قاعدة البيانات يتم على محورين : الكفاية Efficiency والفعالية Effectiveness :

- **الكفـــايـــة :** تقاس الكفاية بمدى قدرة النظام على تحقيق أهدافه بأقل تكلفة. وبالـــتالى فـــإن مقاييس الكفاية تركز على قياس العلاقة بين مدخلات النظام (الوقـــت – الجهـــد – الـــتكلفة) ومخرجاته (خدمة الاحتياجات من البيانات). وواضـــح أن منهج قواعد البيانات يسمح بتصميم نظم أكثر كفاية ، نظرا لما يحققه من وفورات اقتصادية ناتجة عن حسن استغلال الموارد المتاحة.
- **الفعالـــيـــة :** كما تقاس الفعالية بمدى قدرة النظام على تحقيق معدلات الأداء المخططـــة ، فى الوقت المحدد لها. وبطبيعة الحال ، فإن فاعلية منهج قواعد البـــيانات فى تحقيق الهدف النهائي لنظم المعلومات الإلكترونية ، وهو تيسير وصـــول المســـتخدمين المختلفين الى البيانات المطلوبة فى الوقت المناسب ، تكـــون أفضـــل وأقوى كثيرا من فاعلية منهج الملفات المرتبطة بالتطبيقات ، الذى يزداد فشلا كلما تزايدت أحجام البيانات وتعقدت احتياجات المستخدمين.

والآن ، قـد يتساءل المرء : إذا كان منهج قواعد البيانات يكفل تحقيق كافة المزايا السابقة ، أفليس لهذا المنهج من عيوب ؟

فـي الواقـع إن هـذا المنهج لا يسلم من النقد. إلا أن أهم هذه الإنتقادات وأولاها بالإهـتمام هو **صعوبة** – وليس استحالة – المحافظة علي سرية البيانات في ظل خضوع تلـك البـيانات للمشـاركة من جانب عدد كبير من المستخدمين. إن علاج هذه المشكلة يتطلب وضع قيود أو ضوابط علي حرية كل مستخدم علي حدة ، وذلك من زاويتين :

- ما هي بالتحديد **البيانات** التي يسمح له النظام بالوصول اليها؟
- وما هي بالتحديد **عمليات معالجة البيانات** التي يسمح لـه النظام بإجرائها علي تلك البيانات .

وذلـك بالطبع ليس بالأمر الهين. لكنه يجب أن يتحقق ... وإلا فإن منهج قواعد البيانات بأسره يصبح عديم الفائدة ...

* * *

الفصل الثاني

المفاهيم الأساسية لنظم قواعد البيانات

Basic Concepts of Database Systems

المبحث الأول : مفهوم "نظام قاعدة البيانات".

المبحث الثاني: مفهوم "نظام إدارة قاعدة البيانات".

المبحث الثالث: مفهوم "قاعدة البيانات".

مقدمة :

إن لكل علم من العلوم مفاهيمه ومصطلحاته الفنية الخاصة به Terminology. فـــلا يصح أن تؤخذ المسميات بالمعاني الدارجة في اللغة العامة ، لأن المعني الاصطلاحي تختلف دلالته من علم الي آخر. فكلمة " **أصول** " في المحاسبة مثلا تعنى "ممتلكات المشروع وحقوقه طرف الغير" بينما تعني نفس الكلمة في العلوم الشرعية " الأب أو الجد أو العصب العاصب"! والحقيقة أن مفتاح فهم أي علم يتوقف علي فهم المدلول الاصطلاحي لمفاهيمه الأساسية.

وعلـــيه فسوف تكون نقطة بدايتنا في دراسة نظم قواعد البيانات هي التعرف على المعنى الاصطلاحي لثلاثة من المفاهيم الأساسية المرتبطة بالموضوع ، وما يلحق بها من مفاهيم فرعية.

هذه المفاهيم هي :

- نظام قاعدة البيانات Data Base System [DBS] .
- نظام إدارة قاعدة البيانات Data Base Management System [DBMS].
- قاعدة البيانات Data Base [DB].

إن كل مفهوم من هذه المفاهيم يختلف عن غيره في المضمون والدلالة والمؤديات. ولا يجوز للدارسين الخلط بينها أينما تمت الإشارة والإسناد الي أي منها في كل أنحاء هذا المؤلف ، أو في كل موقف تعامل مع نظم قواعد البيانات.

إن الفهم الصحيح للمعاني الإصطلاحية لهذه المفاهيم الثلاثـة ، والوقوف بوضوح علـى الفـروق القائمـة ، بينها دون لبس أو إبهام ... يعتبر في الواقع هـو نقطة البداية الصـحيحة للخـوض في غمار موضوع يتسـم – بطبيعته – بدرجات عالية من التشابك والتعقـيد ، ألا وهو موضوع " **نظم قواعد البيانات** ".

فـي دراستنا لهذه المصطلحات ، سوف نركز علي معرفة ثلاثة أبعاد لكل مفهوم ، وعرضـها بشـكل مقـارن يسمح بتمييزها وفهم نواحى الإختلاف القائمة بينها ، وبالتالي ترسيخ هذه المصطلحات في ذهن الدارس منذ البداية. هذه الأبعاد هي :

- الجوهر أو الطبيعة.
- الوظيفة أو الوظائف الرئيسية.
- العناصر أو المكونات الأساسية.

* * *

١ المبحث الأول

مفهوم : نظام قاعدة البيانات
The Database System Concept

تعرفنا في الفصل السابق على منهج قواعد البيانات ، وأوضحنا أنه هو المنهج السائد حالياً في حل مشكلة حفظ واستخدام البيانات ، وأنه يقوم في جوهره على تحقيق استقلالية البيانات عن التطبيقيات ، مما يتطلب حفظ كافة البيانات في مستودع شامل هو **قاعدة البيانات** DB ، وإخضاع هذه القاعدة لرقابة وسيطرة برنامج تطبيقى متخصص في إدارة البيانات ، هو **نظام إدارة قاعدة البيانات** DBMS ....

الآن الي المفهوم الأشمل والأعمّ ، وهو مفهوم: **نظام قاعدة البيانات** Database System الذى يعتبر أول المفاهيم الأساسية ذات الدلالة الصطلاحية في فهم ودراسة قواعد البيانات. إنه المفهوم الذى يجمع فى طياته كلا من قاعدة البيانات ونظام إدارة قاعدة البيانات!

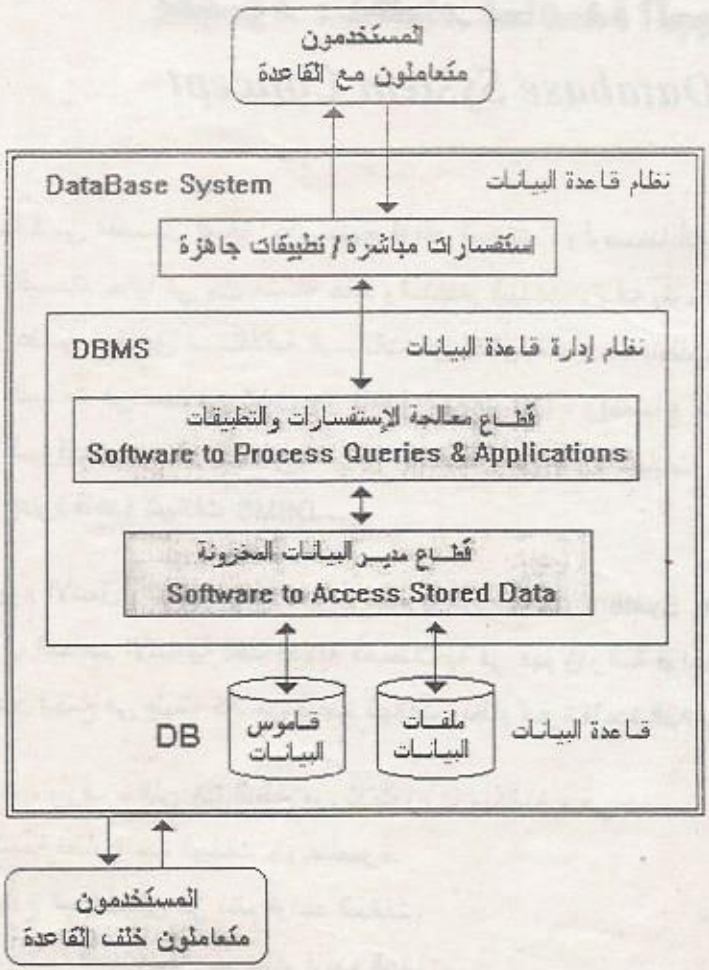
فى هذا المبحث ، سوف نناقش هذا النظام من ثلاث زوايا متكاملة ، هي :

- ❑ طبيعة نظام قاعدة البيانات ، وعناصره.
- ❑ أنواع المستخدمين في نظم قواعد البيانات.
- ❑ تفاعلات المستخدمين مع نظام قاعدة البيانات.

... وإليك تفاصيل ما أجمله البيان.

أولا) طبيعة وعناصر نظام قاعدة البيانات

يمكن تعريف نظام قاعدة البيانات Base System Data [DBS] بانـــه : البيئة Environment أو الإطار العام General Framework الذى تحدث بداخله عمليات حفظ واسترجاع البيانات . طبقا لمقتضيات منهج قواعد البيانات.



الشكل رقم (١/٢) : عناصر نظام قاعدة البيانات DBS

فـنظام قاعدة البيانات DBS هو الإطار الذى يمثل الترجمة العملية لمنهج قواعد البيانات ، ويكـون الهـدف النهائي لهذا الإطار هو : تيسير تلبية احتياجات المستخدمين من البيانات المخزونة فى قاعدة البيانات ، بكفاءة وسرعة.

وكمـا يتضح من الشكل السابق ، رقم (٢-١) ، فإن هذا النظام (أو الإطار العام) يتكون من ثلاثة مستويات أو طبقات متتابعة ومتفاعلة معا ، هي ... من أسفل إلى أعلي :

- قاعدة البيانات DB ، بما تحتوى عليه من بيانات فعلية وبيانات وصفية.
- نظام إدارة قاعدة البيانات DBMS ، بشقيه (قطاع المعالجة – قطاع التخزين).
- تفاعلات المستخدمين مع النظام ، فى شكل استفسارات مباشرة وتطبيقات جاهزة.

وهكـذا يتضـح لـنا أن مفهـوم **نظام قاعدة البيانات** DBSهو مفهوم أكثر اتساعاً وشـمولاً مـن مفهوم قاعدة البيانات **DB** ، ومفهوم **نظام قاعدة البيانات DBS** ... إذ أنه يضـم فـي طياته كلا منهما ، بالإضافة إلى تفاعلات المستخدمين مع النظام. ونظرا لأننا سـوف نخصـص مبحث مستقل لكل من قاعدة البيانات ونظام إدارة قاعدة البيانات ، فإن التركـيز فـي المبحـث سـوف ينصـب علي تناول المستوى الأعلي ... وهو تفاعلات المستخدمين مع النظام.

ثانيا) مستخدمو نظام قاعدة البيانات

نظام قاعدة البيانات هو وسيلة لغاية ... هذه الغاية هى تلبية احتياجات معروفة أو مـتوقعة لطوائف محددة من المستخدمين Users. فما هى فئات وطوائف هـؤلاء المسـتخدمين ؟ وما هى احتياجاتهم التى يعمل علي تلبيتها نظام قاعدة البيانات ؟ وكيف تتفاعل كل طائفة منها مع النظام ؟

فى هذا الخصوص ، يمكن التفرقة بين نوعين من المتعاملين مع القاعدة :

- **المـــتعاملون مـــع القاعدة** : وهم الذين يتفاعلون بشكل مباشر وبطريقة مستمرة مع البيانات المخزّنة في قاعدة البيانات.
- **المتعاملون خلف القاعدة** : وهم الذين لا يهتمون بالبيانات المخزونة بالقاعدة في حد ذاتها لكن أعمالهم تكون ضرورية من أجل تشغيل وإدارة تلك القاعدة.

وفيما يلي مناقشة سريعة لطبيعة واختصاصات ومهام كل مجموعة ، وطوائفها المختلفة.

❑ المجموعة الأولي :
المتعاملون مع قاعدة البيانات
Actors on the Scene

تشـــمل هـــذه المجموعة : الأشخاص الذين تنطوي وظائفهم أو اختصاصاتهم على الاستخدام المباشر لقاعدة البيانات Day-to-Day Use ذاتها والتفاعل المستمر مع البيانات المخزونة فيها ... سواء لغرض تعريف وإنشاء القاعدة ، أو تسجيل وتحديث بياناتها ، أو استرجاع ومعالجة تلك البيانات.

وتضم هذه المجموعة أربع طوائف مختلفة من المتعاملين ، تتمثل في :

- ⇦ مدير قاعدة البيانات Database Administrator (DBA) ، ومعاونوه.
- ⇦ مخططو تطبيقات قاعدة البيانات Application Managers.
- ⇦ المستخدمون الخبراء أو المتمرسون Sophisticated End Users.
- ⇦ المستخدمون من غير الخبراء أو المتخصصين Naive End Users.

ولنـــا على هذه الطوائف ، منذ البداية ، مَلحـــظتَيْن :

الأول : هو أن كل طائفة منها تختلف عن الطوائف الأخرى ، وذلك من حيث :

- ➢ طبيعة الاحتياجات أو المهام المطلوب تنفيذها على قاعدة البيانات.
- ➢ الوسائل المستخدمة في نقل هذه المطالب الى نظام إدارة قاعدة البيانات.
- ➢ مستوى الخبرة والدراية بإمكانيات ولغـــات نظام إدارة قاعدة البيانات.

الثانـي : هو أن هذه الطوائف لا تستطيع الوصول الى قاعدة البيانات واسترجاع وتحديث محتوياتها إلا عن طريق :

- إصدار الاستفسارات المباشرة ، أو تشغيل برامج التطبيقات الجاهزة.
- توجيه هذه الاستفسارات والتطبيقات الي النظام المهيمن على قاعدة البيانات، أي الى نظام إدارة قاعدة البيانات DBMS .

... ويوضح الشكل التالى (رقم ٢-٢) طوائف المتعاملين مع قاعدة البيانات :

(شكل رقم ٢-٢)

المتعاملون مع قاعدة البيانات

والآن ، هيا بنـا نتعرف بإيجاز على طبيعة ووظائف واحتياجات كل طائفة من الطوائف الأربع المذكورة ، مع بيان الأداة أو اللغة المستخدمة في التفاعل مع النظام.

١) مديـــر قاعـدة البيانـات
Database Administrator

فـي بيئة تشغيل نظام قاعدة البيانات متعدد المستخدمين Multi-Users ، يكون هناك تزاحم آني بين المستخدمين على موارد قاعدة البيانات. هذه الموارد تتمثل فى :

- قاعدة البيانات ، وما تحتوى عليه من بيانات فعلية.
- برامج التطبيقات الجاهزة اللازمة لتنفيذ عمليات معالجة البيانات.
- خدمات وتسهيلات نظام إدارة قاعـدة البيانـات.

إن مدير قـــاعدة البيانات : Database Administrator [DBA] هو المشرف العام المسئول عن إدارة هذه الموارد ، وعن بسط الرقابة والسيطرة المركزية على نظام قاعدة البيانات ، بكل عناصره : من البيانات ، وبرمجيات ، وتفاعلات للمستخدمين مع النظام.

❑ الوظائف الرئيسية لمدير قاعدة البيانات :

تتلخص أهم اختصاصات مدير قاعدة البيانات فيما يلي :

➢ تعريف منظور قاعدة البيانات Schema Definition :

إن مـــن أهـــم واجبات مدير القاعدة تعريف المنظور الشامل لقاعدة البيانات ، أي تعـــريف هيكل البيانات ، والعلاقات التى تربط بينها ، والقيود أو التحفظات الواردة عليها مـــن أجـــل ضـــمان صـــحة وتكامل البيانات المخزنة بها. كما يقوم مدير قاعدة البيانات بتصـــميم هـــياكل التخزيـــن المناسبة لحجم وطبيعة قاعدة البيانات ، وطرق الوصول الى البيانات المخزونة بالقاعدة. (سوف نناقش موضوع **المنظور** بالتفصيل في الفصل الثالث).

➢ تنظيم ومراقبـــة شرعية الوصول الى البيانات :

حيـــث يدخـــل في إختصاصات مدير قاعدة البيانات عملية تنظيم ومنح التصاريح الرســـمية المـــبرمجة للمســـتخدمين المختلفين ، والتى نظم حق كل منهم في الوصول الى بيانات معينة Access-Authorization ومعالجتها.

فبالنســـبة لكل شخص يكون مسموحا له بالتعامل مع القاعدة ، يجب تحديد ما يلي تبعا لمستواه الوظيفي وإختصاصاته الإدارية وحدود سلطاته ومسئولياته :

- ما هي البيانات التي يسمح له بالوصول اليها والإطلاع عليها.
- ما هي عمليات المعالجة التي يسمح له بإجرئها علي تلك البيانات.

كما يكون من واجب مدير قاعدة البيانات DBA تحديث هذه التراخيص دوريا ، والتنسيق بينها ، ومراقبة استعمال المستخدمين لها.

➢ **وضع القيود اللازمة لضمان تكامل وتجانس البيانات :**

كمـا يقـوم مدير قاعدة البيانات – صـراحة – بتحديد القيود Constraints التى تستهدف المحافظة على صحة وتكامل وتجانس وسرية البيانات المخزونة فى القاعدة. فمثلا : مـن القيـود التي قد يضعها مدير قاعدة البيانات بهدف ضمان صحة البيانات إشـتراط "عدم زيادة عدد ساعات العمل الإضافي ٧ ساعات يوميا" ... فإذا حاول أحد المستخدمين فيما بعد – عن قصد أو غير قصد – إدخال قيمة تزيد عن هذا الحد ، فإن نظـام إدارة قاعدة البيانات سوف يرفض تسجيل هذه القيمة ، وبالتالي فإن هذا القيـد أو الشـرط سـوف يكفـل صحة بيانات سجلات الساعات الإضافية الموجودة في القاعدة وخلوها من الأخطاء.

➢ **إدخال التعديلات الضرورية علي المنظور أو القيود أو تراخيص الوصول :**

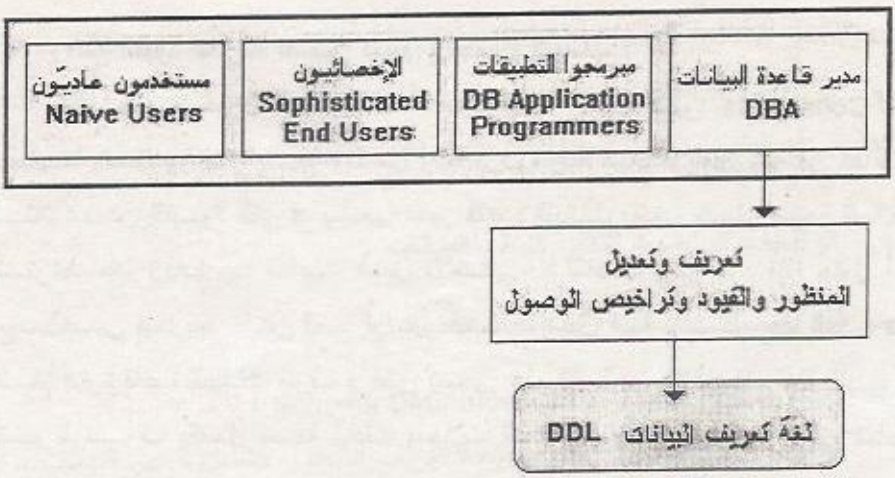
إن نظـام قـاعدة البيانات ليس نظاما مغلقا ، لكنه نظام مفتوح علي البيئة ، يأخذ منها ويعطيها ... ويتأثر بالتغيرات التي تحدث فيها. لذلك فإنه من أهم واجبات مدير قاعدة البيانات توجيه المبرمجين الى اتخاذ اللازم نحو تكويد وإدخال التعديلات المطلوبة علي كل من منظور البيانات – القيود والتحفظات – تراخيص الوصول ، بحيث تتشمى مع الظروف المتغيرة.

❑ **وسـيلة التفاعلات مع النظام :**

إن تنفيذ الوظائف السابقة يتطلب كتابة **تعليمات** لوصف المنظور أو تعريف القيود والـتحفظات أو تحديد تراخيص الوصول للمستخدمين المختلفين. هذه التعليمات هى التي يتم حفظها في ملف خاص من ملفات النظام يسمي : قاموس البيانات Data Dictionary.

أمـا اللغـة التي تستخدم في كتابة وتكويد هذه التعليمات فتسمي : " لغة تعريف البيانات " Data Definition Language [DDL].

والشكل التالي يلخص علاقة مدير قاعدة البيانات بالنظام :



الشكل رقم (٣/٢) :
علاقة مدير قاعدة البيانات بنظام قاعدة البيانات

٢) مخطّطو برامج التطبيقات
DB Application Programmers

بعد أن يقوم محللو النظام System Analysts بالتعرف علي احتياجات المستخدمين النهائيين ، ويضعوا تصوراتهم عن برامج التطبيقات اللازمة للوفاء بها ، يقوم **مبرمجو التطبيقات** DB Application Programmers بتنفيذ هذه التصورات في شكل برامج تطبيقات فردية Database Applications. وعلي ذلك ، فإن هذه الفئة هى التي تملك الخبرة والدراية الكافية لتصميم تطبيقات قواعد البيانات.

❑ **الوظيفة الرئيسية :**

الوظيفة الرئيسية لمبرمجو تطبيقات قاعدة البيانات هي القيام بكافة الأعمال وتنفيذ كافة المراحل ، اللازمة لتصميم التطبيقات الجاهزة ، من تكويد وإختبار وتصحيح Debug وتوثيق ، بحيث تصبح هذه التطبيقات جاهزة للإستخدام بسهولة ويسر.

ويلاحظ أن هذه التطبيقات الجاهزة تتميز بصفتين أساسيتين :

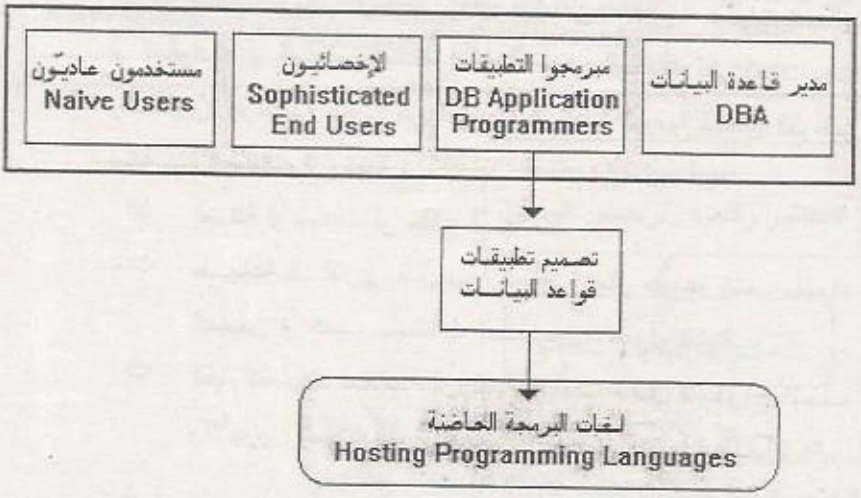
- إن هذه التطبيقات يجب أن تغطي -قدر الإمكان - كافـة عمليات المعالجة المُعتادة أو المألوفـة أو المتوقعة لبيانات القاعدة ، وهى العمليات التي تحدث بصفة روتينية ومـتكررة ورتيـبة كجزء من الأنشطة الوظيفية اليومية لبعض الموظفين. ومن أمثلة هذه العمليات الروتينية :
  - إضافة أو تحديث أو حذف السجلات.
  - طـباعة الـتقارير الدوريـة الجاهزة ،مثل طباعة كشف بأسماء ووظائف العاملين أو كشف بممتلكات المشأة من الأصول الثابتة.
  - القيام بعمليات المعالجة الروتينية للبيانات ، مثل استخراج كشوف المرتبات والأجور الأسبوعية ، أو إستخراج ملخص المبيعات الشهرى.

- أن هـذه التطبيقات تمثل الوسيلة الوحيدة التي تمكن **المسـتخدمين العاديين** أو غير الخـبراء Parametric-Users من التفاعل مع قاعدة البيانات. إذ أن أعضاء هذه الطائفـة من المتعاملين مع القاعدة لا يملكون الخبرة أو الدراية الكافية التي تسمح لهـم بالـتعامل مع نظام إدارة قاعدة البيانات وجها لوجه وبشكل مباشر عن طريق كتابة الاستفسارات Queries.

❑ اللغـة المستخدمة في تصميم التطبيقات الجاهزة :

يعـتمد مبرمجو تطبيقات قاعدة البيانات في تصميم هذه التطبيقات علي نوع من لغات البرمجة الوسيطة ، يسمي : اللغات الحاضنة Hosting Language. تتميز هذه اللغـات – مثل لغة C ولغة Pascal – بقدرتها علي احتواء تعليمات مكتوبة بلغة معالجة البيانات DML. وبالتالي فإنها تمكن المبرمج من تصميم واجهة التطبيق التي يعتمد عليها المستخدم User Interface ، ومن إدراج التعليمات التي توضح لنظام إدارة قاعدة البيانات ما هى عملية المعالجة المطلوب تنفيذها علي البيانات.

والشكل التالي يلخص ما تقدم :

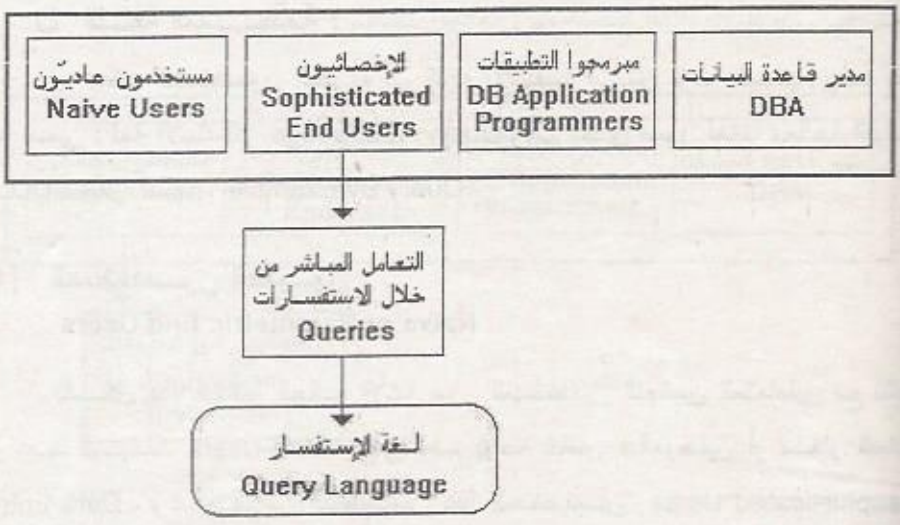


الشكل رقم (٤/٢) :
علاقة مبرمجو التطبيقات بنظام قاعدة البيانات

٣) المستخدمون الخبراء أو الأخصائيون
Sophisticated Users

تشمل هذه الطائفة من المستخدمين الأشخاص الذين تتوافر لديهم الخبرة والدراية الكافية بنظام إدارة قاعدة البيانات المستخدم وإمكانياته ، وبخاصة لغات معالجة البيانات التي يعترف بها ، والتسهيلات التي يوفرها من أجل تنفيذ أعمال معينة ، مثل ربط قاعدة البيانات مع صفحة الوب ، أو تصميم تقرير بطريقة مرئية Visual. ومن أمثلة هؤلاء المستخدمين : المهندسين ، والعلماء ، والمحللين ، والمستخدمين المتمرسين وغيرهم من الإخصائيين أو الخبراء.

والشكل التالي يصور علاقة هذه الطائفة بنظام قاعدة البيانات :



الشـكل رقم (٢-٥) :
علاقة المستخدمون الخبراء بنظام قاعدة البيانات

هـذا ، ويدخل أيضا ضمن نطاق هذه الطائفة ، المستخدمون المتخصصون فى كـتابة برامج تطبيقات راقية لا تدخل فى نطاق معالجة البيانات التقليدية ، مثل برامج قواعد المعرفة Knowledge base ، ونظم الخبرة Expert Systems .

❑ **التفاعلات مع النظام :**

هـؤلاء المستخدمون الأخصائيون أو الخبراء يمكنهم التعامل وجها لوجه وبشكل مباشـر مـع نظـام إدارة قاعدة البيانات ، وذلك عن طريق كتابة الاستفسارات المباشرة Queries ... وبخاصة في الأحوال التالية :

- عمليات الاستفسارات المعقدة التي تهدف إلى استخراج بيانات مستمدة من عدة جداول للبيانات وتنطوى علي عمليات حسابية متعددة.
- عمليـات المعالجة غير العادية والتى لا توجد بالنسبة لها تطبيقات جاهزة سابقة التصميم

❑ **اللـــغة المســـتخدمة :**

يعتمد المستخدمون الخبراء في كتابة الاستفسارات من لوحة المفاتيح رأسا على لغة تسمي : لغة الاستفسار Query Language وهى إحدى صور لغات معالجة البيانات DDLs ، ومن أمثلها : Query By Example.

٤) **المستخدمـــون العاديـــون**
Naïve or Parametric End Users

تشـــكل هذه الطائفة الجانب الأكبر من المستخدمين النهائيين المتعاملين مع نظـــم قواعـــد البـــيانات End-Users. وهى تضم بوجه خاص : الموظفين أو مدخلو البيانات Data-Entry ، وغـــيرهم من المستخدمين غير المتخصصين Unsophisticated Usres الذيـــن يكلفـــون بأداء عمليات إدخال وتحديث البيانات اليومية المألوفة أو المعتادة ، على أســـاس روتيني ومتكرر. هؤلاء المستخدمون لا يملكون الخبرة والدراية الكافية اللازمة للتعامل المباشر مع قاعدة البيانات عن طريق كتابة الاستفسارات ، لكنهم يعتمدون في هذا الخصوص علي برامج التطبيقات الجاهزة التي سبق إعدادها لهذا الغرض.

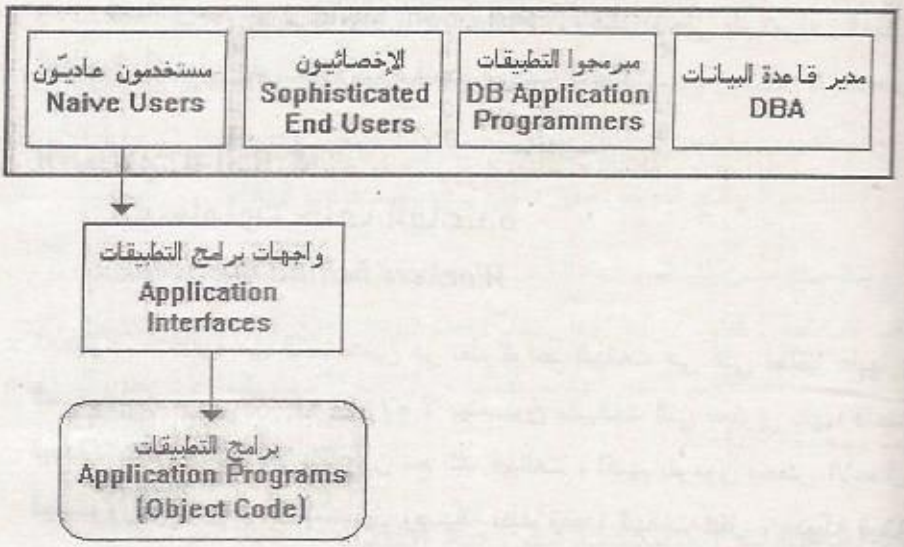
❑ **التفاعلات مع النظام :**

وعلـــي ذلك فإن تفاعلات المستخدمون العاديون مع نظام قاعدة البيانات تتم من خـــلال التطبـــيقات الجاهزة ، وهى تطبيقات تغطى – كما سبق بيانه – عمليات المعالجة الروتينية العادية ، مثل إدخال وتحديث بيانات السجلات وطباعة التقارير الجاهزة.

❑ **الأداة المستخدمة في هذا التفاعل :**

فـــي الـــتعامل مع نظام قاعدة البيانات ، لا يمكن لهذه الطـــائفة استخدام لغات الاستفســـار ، وإنمـــا يعـــتمد أعضاؤها في القيام بعمليات معالجة البيانات المطلوبة علي استخدام ما يسمي : واجهة التطبيقات الجاهزة : Application Interfaces. وهي التي يمكن من خلالها لأى مستخدم استدعاء التطبيق المناسب أو تنفيذ العملية المطلوبة.

والشكل التالي يوضح علاقة المستخدمين العاديين بالنظام.



شكل رقم (٢-٦) :

علاقة طائفة المستخدمون العاديون بنظام قاعدة البيانات

من أهم أنواع واجهات العمل التي يتم تصميمها لهذا الغرض ما يلي :

- واجهات النماذج الجاهزة Form-Base Interfaces :

تستخدم الواجهات من هذا النوع لإظهار نماذج ذات تصميم خاص Forms تستخدم لغرض إدخال أو استرجاع بيانات أحد السجلات ، أو طباعة تقارير سابقة التجهيز Reports ، أو تنفيذ استفسارات محددة مقدما. ومن أمثلتها قائمة الإختيار الرئيسية في تطبيقات نظام Access والتى يطلق عليها اسم : لوحة التبديل Swichboard.

- واجهات قوائم الاختيار Menu-Based Interfaces :

تستخدم هذه الواجهات فى تنفيذ بعض العمليات التى تنطوي على بدائل متعددة ، مثل : طباعة تقرير غير سابق التجهيز. ويقوم هذا النوع بتزويد المستخدم بقائمة

مـــن الاختيارات Options ، تسمى : Menu. ومن خلال هذه القائمة ، يستطيع المســتخدم أن يحدد الاختيار المناسب للعملية المطلوبة ، وهذا الاختيار قد يستدعى بدوره قائمة فرعية أخرى Pull-Down Menu .. وهكذا ، حتى يتم تعريف العملية المطلوبة بأكملها ، فيقوم نظام إدارة قاعدة البيانات بتنفيذها.

❑ المجموعة الثانية : المتعاملون خلف القاعـدة

Workers behind the Scene

المجموعـــة الثانية من المستخدمين في نظم قواعد البيانات هى التي أطلقنا عليها : المـــتعاملون خلف القاعدة. وهؤلاء لا يهتمـــون بالبيانات التي تحتوى عليها قاعدة البيانات بشكل مباشر ولا يتفاعلون مع تلك البيانات ، لكنهم يقومون ببعض الأعمال الضـــرورية مـــن أجـــل تشـــغيل وصيانة نظام قاعدة البيانات ككل ، وتهيئة البيئة البرمجية التي يعمل فيها هذا النظام.

وتشمل طائفة المتعاملون خلف القاعدة ، بوجه خاص ، الفئات التالية :

١) مصممو برامج نظام إدارة قاعدة البيانات DBMS Designers :

وهم المختصون بتصميم المكونات البرمجية المختلفة Modules لنظام إدارة قاعدة البـــيانات ذاته ، أو تصميم هذا النظام بأسره ، كحزمة متكاملة من حزم البرمجيات الجاهزة Software Package. مع ملاحظة أن الميل الغالب في الوقت الحاضر هو الاعتماد علي شراء الحزم التجارية الجاهزة ، مثل نظام Access.

٢) مصممو برامج المنافع العامة أو خدمات الدعم DBMS Utilities :

ويقصـــد بهـــم : المبرمجون الذين يقومون بتصميم برامج خدمات الدعم أو المنافع الإضـــافية اللازمة للمساعدة فى تشغيل واستخدام نظام قاعدة البيانات ، ومن أبرز

أمثلتها : برامج النسخ الاحتياطي لملفات البيانات Backup Tools وبرامج مراقبة وتقييم أداء النظام Performance Monitoring Tools.

٣) مسئولو تشغيل وصيانة النظام Operators & Maintenance Personnel:

وهم الفنيون والمهنيون الذين يتولون مسئوليات تشغيل Running وصيانة الأجهزة والملفات والبرامج الموجودة فى داخل إطار نظام قاعدة البيانات.

ويتضـح مـن ذلـك ، أنه بالرغم من أن أعمال هذه المجموعة من المستخدمين ، على تنوعها المختلفة ، تعتبر لازمة لجعل نظام قاعدة البيانات قابلا للاستخدام والتشغيل الآمن – إلا أنهـا لا تسـتخدم قاعدة البيانات بذاتها ، ولا تتناول محتوياتها لأغراض استرجاع وتحديث ومعالجة البيانات.

ثالثا) تفاعلات المستخدمين مع النظام
DB-Users' Interactions

المسـتخدمون النهائيون – من خبراء وعاديين – يتفاعلون مع البيانات المخزونة فى القاعدة من خلال نوعين من الأدوات ، هما :

- الاستفسارات المباشرة أو التفاعلية Interactive Queries ،
- التطبيقات الجاهزة الملحقة بنظام قاعدة البيانات DB Applications .

وعليه فإن الاستفسارات وبرامج التطبيقات تمثل أحد المستويات التي يتكون منها أي نظام من نظم قواعد البيانات.

وفيما يلي نبذة موجزة عن كل نوع.

١) الاستفسارات المباشرة Queries :

الاستفسار Query – بالمعنى الاصطلاحي– هو الاستعلام عن بيانات معينة مخزنة فى قاعدة البيانات. هذه الاستفسارات يتم كتابتها بواسطة المستخدمون الخبراء أو

المبرمجيـــن Programmer وغـــيرهم من المستخدمين الذين تتوافر لديهم الخبرة والدراية الكافية بإمكانيات وتسهيلات نظام إدارة قاعدة البيانات DBMS Facilities ... ومـــن ثم فإنهم يستطيعون التعامل مع قاعدة البيانات بشكل مباشر، عن طريق إصدار مثل هذه الاستفسارات.

وبطبيعة الحال ، يكون اللجوء الى الاستفسارات كطريقة للتفاعل مع قاعدة البيانات عـــندما يتطلب الأمر معالجات غير عادية أو معقدة ، لا تغطيها برامج التطبيقات الجاهزة المـــتاحة. ويـــتم فى هذه الحالة بناء وتعريف الاستفسار من لوحة المفاتيح رأسا : **لـــغة الاستفسار** Database Query Language.

(٢) برامـــج التطـــبيقات الجاهـــزة
DB Application Programs

التطبيقات الجاهزة في أي نظام لقاعدة البيانات Database Applications ما هى إلا بـــرامج ضمنية صغيرة متخصصة فى القيام بعمليات المعالجة الروتينية لبيانات القاعدة ، ويتولى إعدادها **مبرمجو التطبيقات** Application-Programmers. وقد أوضـــحنا مـــن قبل أن هذه التطبيقات يتم إعداده للاستخدام من جانب المستخدمين العاديين Naïve or Parametric Users ، وذلك من أجل تيسير تعاملاتهم اليومية مـــع القاعدة. وبطبيعة الحال ، يتخصص كل تطبيق منها بتنفيذ عملية معينة من العمليات الروتينية أو المعروفة مقدما ، التى تُجرى عادة على بيانات القاعدة.

* * *

٢ المبحث الثاني

مفهوم : نظام إدارة قاعدة البيانات

The Concept of Database Management System [DBMS]

يتضح لنا من مطالعة عناصر نظام قاعدة البيانات DBS – والمعروضة في الشكل رقم (١/٢) – أن **نظام إدارة قاعدة البيانات** DBMS هو العنصر الثاني من عناصر نظام قاعدة البيانات ... العنصر الذي يقوم بدور حيوي فى إدارة البيانات المخزونة في القاعدة ، وتمكين المستخدمين من الوصول إلي هذه البيانات واسترجاعها ومعالجتها.

من الضروري للدارسين والمتعاملين مع نظم قواعد البيانات أن يكون واضحا فى الذهن الفروق الجوهرية الدقيقة التي توجد بين مفهوم **نظام قاعدة البيانات** DBS ومفهوم **نظام إدارة قاعدة البيانات** Data Base Management System. إذ أن كل مفهوم منهما إنما يعنى شيئا مختلفا ... فلا يجوز الخلط بينهما ، أو اعتبارهما مترادفين.

في هذا المبحث ، سوف نسلط الأضواء على بعض جوانب نظام إدارة قاعدة البيانات ، وذلك فقط بالقدر الذى يكفى لتكوين الخلفية المناسبة لدى القارئ عن طبيعة هذا النظام ووظائفه ، والكيفية التى يقوم بها بأداء تلك الوظائف.

حيث تركز المناقشة علي تناول الجوانب التالية :

- ❑ طبيعـــة ووظائف نظام إدارة قاعدة البيانات.
- ❑ لغات البرمجة المستخدمة فى نظم قواعد البيانات.
- ❑ عناصر ومكونات نظام إدارة قاعدة البيانات.

أولا) طبيعة ووظائف نظام إدارة قاعـدة البيانـات

يمكـــن لـــنا تعـــريف **نظام إدارة قاعدة البيانات** DBMS بأنه : برنامج تطبيقى Application Software متخصص في إدارة قواعد البيانات ، يتضمن حزمة من البرمجيات المسئولة عن : تعريف وإنشاء هيكل قاعدة البيانات ، وإدخال وتحديث البـــيانات التـــي تدخـــل فـــي نطـــاق هذه القاعدة ، وتلبية احتياجات المستخدمين المختلفين من هذه البيانات بكفاءة Efficiently وفاعلية Effectiveness.

ومـــن هـــذا التعريف المبســـّط ، يمكن لنا تفهـــُم : طبيعة الدور الذى يقوم به نظام إدارة قاعدة البيانات ، ونوعية الوظائف الرئيسية التى تقـــع فى صميم اختصاصه وتحقق الغاية من وجوده ... الأمر الذى نوضحه الفقرات التالية.

(١) طبيعة نظام إدارة قاعدة البيانات كواجهة بينيـــّة DBMS as an Interface Engine

من التعريف السابق لنظام إدارة قاعدة البيانات DBMS ، يتضح لنا هـــذا النظام هـــو حـــزمة من البرمجيات المتخصصة في إدارة قواعد البيانات. هذا النظام يمثل في الواقع **واجهة بينية** An Interface أو **بيـــئة عمل وسيطة** ، تتواجد بين كل من :

- ⇦ تفاعلات المستخدمين المتمثلة في الاستفسارات والتطبيقات الجاهزة ، من جهة.
- ⇦ والبيانات المخزونة في القاعدة من جهة أخرى.

هذه الواجهة الوسيطة هى التى سوف يتمكن المستخدمون المختلفون – عن طريقها ، ومن خلالهـا – من حفظ البيانات المقصودة بطريقة منظمة ، والوصول إلى البيانات المطلوبة واسترجاعها ومعالجتها ... بكفاءة وسرعة.

مـن أجل توضيح هذه النظرة ، نفترض أن إحدى المنشآت التجارية التى تعمل فى مجـال تسويق مجموعة من المنتجات لديها خمسة تطبيقات تواجدت على فترات مختلفة ، ويخـتص كل تطبيق منها بالتعامل مع تشكيلة من البيانات المناسبة لتلبية بعض احتياجات المستخدمين. هذه البرامج التطبيقية هى :

- برنامج المنتجات : **Products** ، ويختص بتسجيل وتحديث وعرض قائمة تشكيلة المنتجات التى تتعامل فيها المنشأة.
- بـرنامج الأسعار : **Price-Changes** ، ويختص بتسجيل وتحديث وعرض قائمة التغيرات فى أسعار المنتجات.
- برنامج المخزون السلعى : **Stock** ، ويختص باستخراج قائمة الأرصدة الموجودة حاليا بالمخازن من كل منتج.
- بـرنامـج إعادة تموين المخازن : **Re-Order** ، ويختص باستخراج تقرير دوري عـن الأصناف التي اقتـربت أرصدتها الفعلية من النفـاذ ، أو التى تجاوزت حد إعادة الطلب ، ويجب إصدار أمر توريد كميات جديدة منها قبل نفاذ رصيدها.
- برنامج الدليل : **Catalogue** ، ويختص بطباعة كتالوج المنتجات والأسعار.

الجـدول التالي ، شكل رقم (٢-٧) ، يتضمن بيان أهم نوعيات أو مفردات البيانات اللازمة للوفاء باحتياجات مستخدمي كل تطبيق من هذه التطبيقات على حدة ، منظورا إليه فى عزلة عن باقي التطبيقات. مع ملاحظة أنه فى ظل منهجية قواعد البيانات ، **لن يستقل** كـل بـرنامج تطبيقـي بالبيانات اللازمة لتنفيذه ، وإنما يجب أن يتم تخزين كافة قطع أو مفردات البيانات اللازمة لتلبية احتياجات كافة التطبيقات فى وعاء شامل أو مستودع واحد ، هو : **قاعدة البيانات DB**.

إسم البرنامج Prorgam Name	البيانات المطلوبة لتنفيذ البرنامج Data Items Required by each Program
Products	Product-No , Product description , Price , Quantity-in-Stock , Re-Order-Level.
Price	Product-No , Price.
Catalogue	Product-No , Product description , Price.
Stock	Product-No , Quantity-in-Stock.
Re-Order	Procuct-No , Product Description, Product-in-Stock , Re-Order-Level.

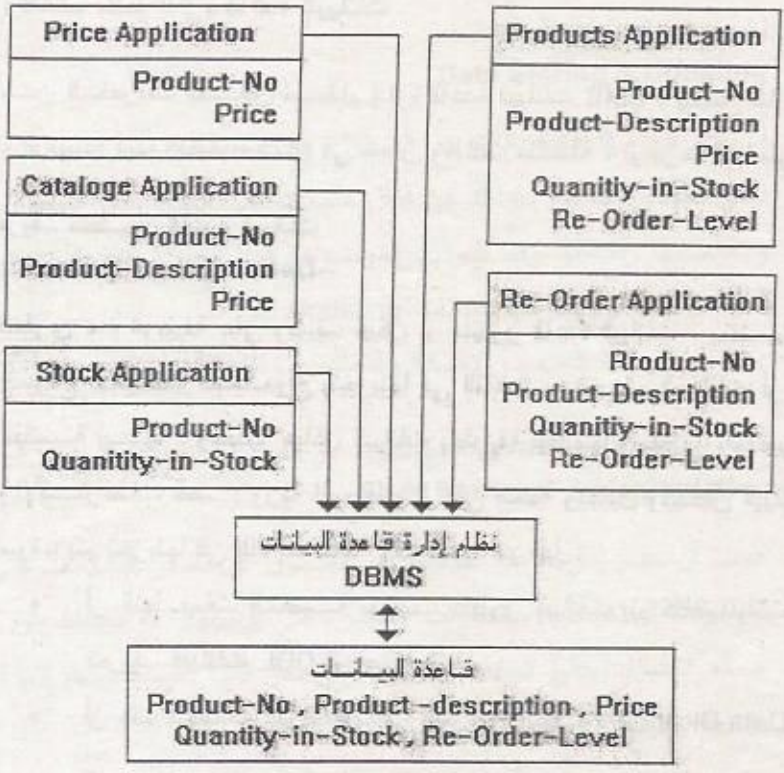
شكل رقم (٢-٧)

البيانات اللازمة لبعض تطبيقات قواعد البيانات الجزئية في منشأة تجارية

وتبعا لذلك ، إذا تأملنا البيانات السابقة ، سوف نكتشف سريعا أن قطع البيانات المطلوب تخزينها في قاعدة البيانات ، على اختلاف استخداماتها ، تتمثل فقط في خمسة مفردات Data Items ، تكون محلّ مشاركة من كافة التطبيقات ، وهي :

- رقم المنتج **Product-No**.
- مواصفات المنتج **Product Description**.
- سعر المنتج **Price**.
- رصيد الصنف في المخازن **Quantity-in-Stock** .
- مستوى إعادة الطلب **Re-Order-Stock**.

في ضوء ما تقدم ، فإن الشكل المعروض في الصفحة التالية ، شكل رقم (٢-٨) ، يعرض عليك نموذجا مبسطا لهيكل نظام قاعدة البيانات ، متضمنا البيانات التي تحتوى عليها القاعدة ، والدور الذي يلعبه نظام إدارة قاعدة البيانات كواجهة عمل بينيّة :



الشكل رقم (٢-٨)

نموذج مبسّط لدور DBMS كواجهة عمل بينية

ويتضح من هذا الشكل أن نظام إدارة قاعدة البيانات DBMS هو الواجهة البينية التي تتوسط ، وتحقق التفاعل التبادلي ، بين كل من :

- قاعدة البيانات ذاتها DB ، بما تحتوى عليه من بيانات فعلية.
- وتطبيقات قاعدة البيانات الجاهزة ، التي تسعى الي الوصول الي قاعدة البيانات لغرض تحديث أو استرجاع أو معالجة محتوياتها.

وبدون هذه الواجهة البينية ، لن يقوم نظام قاعدة البيانات من الأساس.

(٢) وظائف نظام إدارة قاعدة البيانات

من التعريف السابق لنظام إدارة قاعدة البيانات BMS ، يتضح لنا أيضا أن الوظائف الرئيسية لهذا النظام ، تتمثل في خمس وظائف متكاملة ، نوجزها فيما يلي :

أ - تعريف منظور قاعدة البيانات
Defining the DB Schema

تنطوي هذه الوظيفة على وصف هيكل أو منظور قاعدة البيانات ، متضمنا: حصر أنواع البيانات المسموح بتخزينها فى القاعدة ، وتعريف العلاقات أو الروابط القائمة بينها ، وتحديد هياكل البيانات وطريقة تنظيمها وحفظها ، وتعيين القيود والاشتراطات الضرورية للمحافظة علي صحة وتكامل وتجانس البيانات التي سوف يتم تخزينها في تلك القاعدة. وقد علمت من قبل:

- أن التعليمات الخاصة بوصف منظور البيانات يتم كتابتها باستخدام لغة تعريف البيانات DDL الخاصة بالنظام.
- أن هذه التعليمات يتم حفظها في قاموس البيانات Data Dictionary.

ب - إنشاء هيكل قاعدة البيانات
Constructing the Database

ويقصد بهذه الوظيفة : التشييد الفعلي لهيكل قاعدة البيانات ، وتعتبر جزءا لا يتجزأ من الوظيفة السابقة. ففي قاعدة البيانات العلاقية Relational-Database مثلا ، تنطوى هذه الوظيفة علي تصميم جداول البيانات ، وتحديد خصائص الحقول التي يحتوى عليها هيكل كل جدول منها ، وتحديد ما يسمي حقول المفاتيح Key Fields واستخدام هذه الحقول في الربط بين سجلات الجداول المختلفة بعلاقات الارتباط المناسبة ، بحيث تشكل هذه العناصر في مجموعها نسيجا متكاملا ... وبذلك تصبح القاعدة جاهزة لاستقبال البيانات.

ج – تخـزين وتحـديث البيانـات
Data Storing & Updating

فـنظام إدارة قاعدة البيانات هو المسئول عن استقبال وتخزين البيانات التي تدخل فـي نطـاق القاعدة ، وذلك في شكل سجلات Records ، مع التحقق من صحتها وتجانسـها وتكاملها وفقا للقواعد الموضوعة لهذا الغرض. كما أنه المسئول عن تنفيذ عمليات تحديث هذه البيانات Updating لكي تعكس التغيرات فى عالم الواقع ... بما فى ذلك الحذف والإضافة والتصحيح والإحلال.

د – تلبية احتياجات المستخدمين المختلفين
Data Retrieval & Manipulation

كمـا أن نظام إدارة قاعدة البيانات هو المسئول عن تفسير التعليمات التي تتضمنها التطبيقات الجاهزة والاستفسارات المباشرة التي يوجهها له المستخدمون ، وتطبيق هـذه التعليمات علي البيانات المطلوبة من أجل تلبية احتياجات هؤلاء المستخدمين من هذه البيانات ، وتقديمها لهم في الشكل المطلوب.

هـ – توفير أدوات تصميم تطبيقات قواعد البيانات
Application Development Tools

عـلاوة علي ما تقـدم ، فإن نظم إدارة قواعد البيانات الكبيرة أو المتطورة – مثل نظـام Access ونظـام FoxPro ونظام Oracle – توفر للمبرمجين وغيرهم من المسـتخدمين الخـبراء بيـئة تصميم شاملة ، تعاونهم في تصميم تطبيقات متكاملة لقواعد البيانات ، بما في ذلك أدوات تصميم الجداول Tables، والنماذج Forms ، والاستفسارات Queries ، والتقارير الجاهزة Reports... الخ.

لكـن القـيام بهذه الوظائف يتطلب توجيه تعليمات إلى نظام إدارة قاعدة البيانات تكون مكتوبة باللغة أو اللغات التي يفهمها هذا النظام ... **فما هي هذه اللغات ؟**

ثانيا) لغـات قواعـد البيانـات

لغـات قواعـد البيانات هى اللغات التي تستخدم في كتابة التعليمات بالشكل الذى يفهمـه نظام ادارة قاعدة البيانات ، وبالتالي يستطيع تفسير وتنفيذ تلك التعليمات. إن كل نظام له لغاته الخاصة به ، والتى قد لا يفهمها نظام غيره.

تنقسم لغات قواعد البيانات إلى ثلاثة أنواع ، هي :

- لغـات تعريف منظور البيانات Data-Definition Languages.
- لغـات معالجة البيانات Data-Manipulation Languages.
- لغـات التكامل الشاملة Comprehensive Integrated Languages.

ونقدم فيما يلي شرحا موجزا لطبيعة ووظيفة وتصنيفات كل نوع من هذه اللغات :

(١) لغـات تعـريف البيانـات
Data-Definition Languages

أوضـحنا من قبل أن أول واجبات مدير نظام قاعدة البيانات DBA هو تعريف مـنظور قـاعدة البـيانات ، الـذى يتضمن وصف أنواع البيانات المسموح بتخزينها ، والعلاقات القائمة بينها. والقيود والتحفظات الخاضعة لها ، ومسارات تخزينه وحفظها ، والتشكيلات المطلوبة لتلبية احتياجات المستخدمين منها...

اللغـة المستخدمة في كتابة وتعريف منظور البيانات هى **لغة تعريف البيانات** Data Definition Language ، وتسمى اختصارا DDL. وينـتج عن عملية ترجمة Compilation التعليمات المكتوبة بهذه اللغة وتفريعاتها مجموعة من جداول النظام التى يـتم تخزينها فى قاموس البيانات Data Dictionary. هـذا القاموس يلعب دورا حيويا وبالغ الأهمية فى نظام إدارة قاعدة البيانات ، لأن النظام سوف يرجع إليه ، ويستشيره ، قبل تنفيـذ أيـة عملية من عمليات قراءة أو استرجاع أو معالجة البيانات الفعلية.

❑ أنواع لغات تعريف البيانات :

في نظم إدارة قواعد البيانات الكبيرة ، تنقسم لغة تعريف البيانات إلى ثلاثة أنواع :

- **لغـــة VDL** : وهـــى التي تصف أنواع تشكيلات البيانات View اللازمة لتلبية احتياجات المستخدمين المختلفين (المنظور الخارجي).
- **لغـــة DDL** : ويقتصر دورها في هذه الحالة علي وصف ما هي قطع البيانات التـــي يتم تخزينها في قاعدة البيانات لوصف الأحداث الداخلة في نطاقها (المنظور المنطقي).
- **لغـــة SDL** : وهي التي تصف كيفية تخزين هذه البيانات بالفعل داخل وسائط التخزين (المنظور المادى أو الداخلي).

(٢) لغـات معـالجة البيانـات
Data-Manipulation Languages

يُقصد بمعالجة البيانات Data Manipulation – بمعناها الواسع – كافة العمليات المتعلقة بحالـــة البيانات الفعلية Database Instance المخزونة بقاعدة البيانات ، والتى يجريها المستخدمون المختلفون للنظام ، سواء فى شكل استفسارات مباشرة Queries أو من خلال التطبيقات الجاهزة. وتشمل هذه العمليات بوجه خاص :

- استرجاع Retrieve بيانات مخزّنة فى قاعدة البيانات.
- إضافـــة Insertion بيانات جديدة الى محتويات قاعدة البيانات.
- حـــذف Deletion بيانات مخزّنة بالفعل داخل قاعدة البيانات.
- تعـــديل Modification أو تحديث Updating بيانات مخزّنة بقاعدة البيانات.

ويلاحـــظ أنـــه بمجـــرد الانتهاء من ترجمة منظور البيانات ، وطرح نظام قاعدة البيانات للاستخدام الفعلي Populating Database ، فإن نظام إدارة قاعدة البيانات يجب أن يتضمن وســـيلة تسمح بمعالجة استفسارات وتطبيقات المستخدمين.

وهكذا فإن لغة قواعد البيانات المستخدمة لهذا الغرض هى اللغة المعروفة باسم : لغة معالجة البيانات Data-Manipulation Language ، وإختصارها الرمز DML. هذه اللغة هى التى تجعل نظام إدارة قاعدة البيانات قادرا على تمكين المستخدمين من الوصول الى البيانات المخزّنة ، وإجراء عمليات المعالجة المطلوبة عليها.

❑ أنواع لغات معالجة البيانات :

تنقسم لغات معالجة البيانات إلى نوعين : لغات إجرائية ، وغير إجرائية :

أ - لغات معالجة البيانات غير الإجرائية Non-Procedural DMLs

وهى لغات معالجة البيانات التى تتطلب من المستخدم أن يحدد فقط ما هى البيانات التى يريدها والعملية المطلوب تنفيذها عليها ، دون أن يشغل باله بكيفية الحصول على تلك البيانات. وهى من لغات المستوى الرفيع High-Level.

تستخدم هذه اللغات بإحدى طريقتين :

الأولى : في التعامل المباشر مع قاعدة البيانات :

فبعض لغات معالجة البيانات تسمح بإمكانية الاعتماد عليها ، فى حد ذاتها ، للتفاعل المباشر مع قاعدة البيانات Stand-Alone Interactive، وإجراء عمليات معالجة معقدة على محتوياتها من البيانات. ويطلق علي مثل هذه اللغات اسم : **لغة الاستفسار** Query Language.

الثانية : من داخل برامج التطبيقات الجاهزة :

كما يمكن أن تستخدم لغات معالجة البيانات غير الإجرائية في تصميم تطبيقات قواعد البيانات ، وذلك من خلال إحدى لغات البرمجة الحاضنة التي تسمح بإحتواء تعليماتها بشكل ضمنى.

ب - لغات معالجة البيانات الإجرائية

Procedural DMLs

وهي لغات معالجة البيانات التى تتطلب من المستخدم أن يحدد : **ما هى** البيانات التى يحتاجها ، **وكيف** نحصل على هذه البيانات ، بالإضافة إلى وصف وتعريف **عملية** المعالجة المطلوب تنفيذها علي تلك البيانات.

هذا النوع يعتبر من لغات المستوى الأدنى Low-Level ، التى يجب تضمين تعليماتها من خلال اللغات الحاضنة Hosting Languages المستخدمة في تصميم تطبيقات قواعد البيانات الجاهزة. وقد كانت هذه اللغات تستخدم بوجه خاص مع نظم قواعد البيانات القديمة ، المبنية على النموذج الهرمي أو النموذج الشبكي Network Model.

(٣) لغات التكامل الشاملة

Comprehensive Integrated Languages

تتجه نظم إدارة قواعد البيانات الحديثة إلى عدم استخدام عدة لغات منفصلة لتعريف منظور هيكل البيانات DDL ومعالجة البيانات DML ... ولكنها تعتمد بدلا من ذلك على لغة واحدة متكاملة ، تستطيع الوفاء بكافة الوظائف التى تقوم بها تلك اللغات التقليدية مجتمعة.

هذه اللغة الموحدة أو الشاملة تستطيع في نفس الوقت القيام بما يلي :

- تعريف المنظور المنطقي للبيانات ،
- وتعريف الهيكل المادي للبيانات ،
- وتصميم وتنفيذ الاستفسارات وبرامج التطبيقات اللازمة لمعالجة البيانات.

ويطلق العلماء على هذا النوع من لغات قواعد البيانات اسم : **لغات التكامل الشاملة** Comprehensive Integrated Languages.

ولعــل مــن أكثر لغات التكامل الشاملة استخداما وشيوعا فى الوقت الحاضر ، اللغات التالية : (١)

- لــغة الاستفسار القياسية [SQL] Standard Query Language.
- لــغة الاستفسار [QUEL] QUEry Language.
- لــغة الاستفسار بالأمثلة [QBE] Query By Example.

علــى سبيل المثال ، فإن لغة **SQL** تحتوى على أوامر لتعريف منظور البيانات، وأوامــر للــتفاعل المباشر مع النظام من خلال تصميم وتنفيذ استفسارات المستخدمين ، وأوامر لمعالجة وتحديث البيانات.

كما أن لغة **SQL** تتضمن :

⇦ إمكانيات راقية لتعريف وتطبيق الفهارس Indexes علي السجلات التي تحتوى عليها ملفات البيانات المرتبطــة

⇦ وتسهيلات متنوعة لإدراج تعليماتها Embedding SQL Statements ، الخاصة بمعالجة البيانات، ضمن لغات البرمجة عامة الغرض المستخدمة لتصميم تطبيقات قواعد البيانات ، مثل لغة C ولــغة Pascal.

وفــى النهاية ، تجدر الإشارة الى أن نظام إدارة قاعدة البيانات العلاقية Access 2000 ، محــل الاهــتمام فى هذا الكتاب ، يسمح من داخل بيئته باستخدام لغة SQL فى تصميم الاستفسارات.

---

(١) فى شرح تفصيلي مقارن لهذه اللغات ، ارجع بوجه خاص الى :

- Ramez Elmasri, & Shamkant B. Navathe, Fundamentals of Database Systems, 2nd ed., ***Opt. Cit.***, pp. 185-258, &
- S. Sannan, & G. Otten, ***SQL - The Standard Handbook,*** (Englewood Cliffs, NJ: McGraw Hill International, 1993.

ثالثا) مكوّنات نظام إدارة قاعدة البيانات

بالعودة مرة أخرى الي الشكل رقم (٢-١) ، يتضح أن أي نظام لقاعدة البيانات يجب أن يحتوى بالضرورة على نظام لإدارة قاعدة البيانات DBMS. وينقسم نظام إدارة قاعدة البيانات الى قطاعين متفاعلين من المكونات Components :

- ⇦ الأول : هو قطاع معالجة الاستفسارات والعمليات ،
- ⇦ والثاني : هو : قطاع إدارة التخزين (إدارة تخزين البيانات).

ونقدم فيما يلي شرحاً موجزاً لوظيفة كل قطاع منهما ، والمكونات التى يتألف منها ، وعلاقات التفاعل القائمة فيما بين هذه المكونات معا ، ومع باقي عناصر نظام قاعدة البيانات ككل.

(١) قطاع معالجة الاستفسارات والعمليات

Query/Transactions Processor

القطاع الأول فى نظام إدارة قاعدة البيانات هو القطاع المعروف باسم : قطاع معالجة الاستفسارات والعمليات. هذا القطاع هو الذى يواجه المستخدمين ، ويتضمن مجموعة من البرمجيات اللازمة لاستقبال وتفسير التعليمات التي تنطوي عليها استفسارات وبرامج تطبيقات المستخدمين ، وتنفيذها على البيانات المخزنة في قاعدة البيانات.

❑ **الوظائف الأساسية لقطاع المعالجة :**

يقوم هذا القطاع بالوظائف التالية :

١) استقبال برامج التطبيقات الجاهزة واستفسارات المستخدمين المباشرة.
٢) تفسير ما يحتوى عليه التطبيق أو الاستفسار من تعليمات.
٣) تنفيذ هذه التعليمات علي البيانات المطلوبة، وذلك بمعاونة ، ومن خلال ، قطاع إدارة تخزين البيانات.

❑ الخصائص العامة لقطاع المعالجة :

يتميز قطاع معالجة الاستفسارات والعمليات بالخصائص التالية:

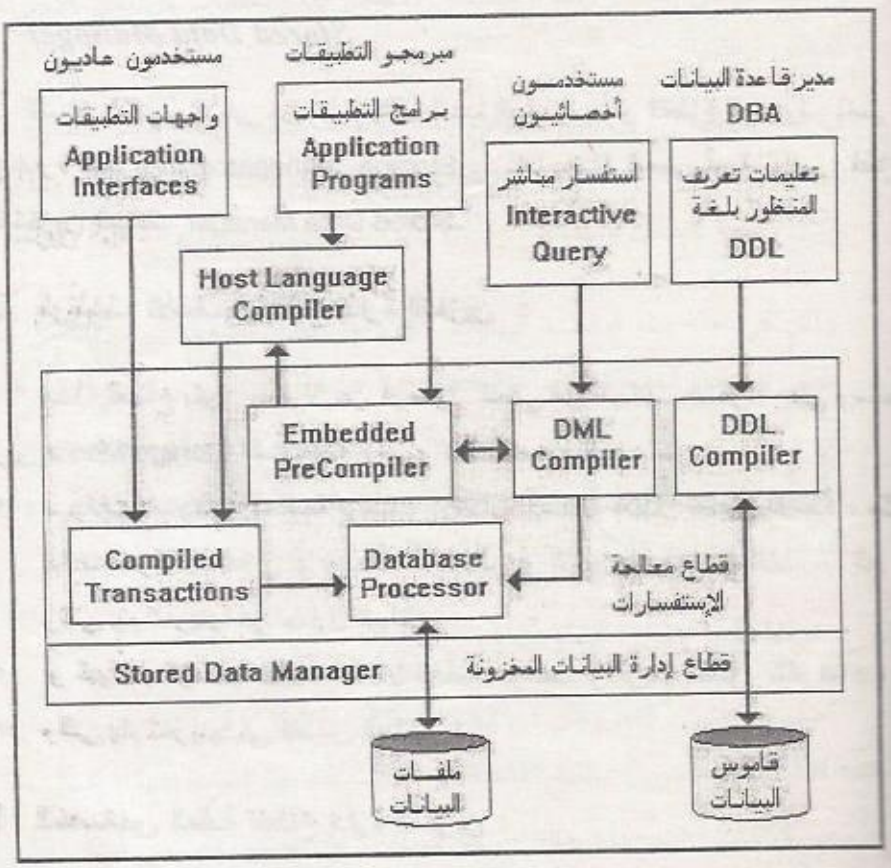
- أنـــه هو القطاع الذى يواجه المستخدمين المختلفين لقاعدة البيانات ، سواء كانوا مـن المستخدمين العاديين الذين يستخدمون برامج تطبيقات قاعدة البيانات فى تنفيذ العمليات الروتينية اليومية ، أو كانوا مستخدمين أخصائيين يتفاعلون مع القاعدة من خلال تصميم الاستفسارات المعقدة ، أو كان هؤلاء المستخدمين من المبرمجين الذين يتولون عملية تصميم برامج تلك التطبيقات.
- أنـــه يمـثل ذلك الجزء من نظام إدارة قاعدة البيانات المسئول عن استقبال وتفسير وتنفيذ عمليات معالجة البيانات الواردة فى تلك الاستفسارات وبرامج التطبيقات ، أي المسئول عن القيام بعمليات استرجاع وتحديث البيانات المخزّنة فى القاعدة .
- وهـذا القطاع لا يتعامل مع البيانات المخزنة على وسائط التخزين بشكل مباشر ، وإنما يتحقق له ذلك عن طريق ومن خلال القطاع الثاني لنظام إدارة قاعدة البيانات وهو قطاع إدارة تخزين البيانات.

❑ مكونات قطاع المعالجة :

يتضـمن الشـكل المعروض في الصفحة التالية – شكل رقم (٢-٩) – المكونات الأساسـية لقطـاع معالجة العلميات والاستفسارات ، وعلاقات التفاعل القائمة بينها وبين باقى عناصر النظام ، وهي تشمل :

- **معالج قاعـدة البيانات** Database Processor : وهو المسئول الأوحد عن التنفيذ الفعلي للعمليات المطلوبة علي البيانات المعنية.
- **مـترجم لغة معالجة البيانات** DML Compiler : وهو الذى يقوم بترجمة الأوامـر المكتوبة بلغة DML أو لغة الاستفسار إلى تعليمات يفهمها معالج قاعدة البيانات.

- المترجم التمهيدى للغـــة المعالجة الضمنية Embedded DML Compiler : وهو الـــذى يقـــوم بترجمة الأوامر الضمنية المكتوبة بلغة معالجة البيانات DML الواردة بداخـــل التطبـــيقات الجاهـــزة التي يتم تصميمها باستخدام إحدى اللغات الحاضنة ، وذلك بالتعاون مع مترجم اللغة الحاضنة Hosting Language Compiler.



الشكل رقم (٢-٩)
مكونات قطاع معالجة الاستفسارات والعمليات
في نظام إدارة قاعدة البيانات

- **مترجم لـغة تعريف البيانات** DDL Compiler : وهو الذى يقوم بترجمة التعليمات الخاصة بوصف منظور قاعدة البيانات ، وحفظها في جداول خاصة بداخل قاموس البيانات Data Dictionary.

(٢) قطاع إدارة تخـزين البيانات

Stored Data Manager

الشـقّ الثاني والأخير من نظام إدارة قاعدة البيانات ، هو القطاع المعروف باسم : **قطاع إدارة التخزين** Storage Management ... كما يسميه البعض أحيانا باسم : **قطاع إدارة تخزين البيانات** Stored Data Manager.

❑ الوظائف الأساسية لقطاع إدارة التخزين :

هـذا القطاع يكون مسئولا عن الوصول الفعلي الى البيانات المخزّنة على وسائط التخزين Storage Media المستخدمة ، سـواء أكانت هذه البيانـات :

- ⇦ بـيانات السجلات الخاصة بوصف الأحداث المتعلقة بمجال تطبيق القاعدة ، مثل بيانات حركة المخازن أو مبيعات الأصناف أو نتائج امتحانات الطلاب ... الخ ، والتى يتم تخزينها في جداول البيانات.
- ⇦ أو البيانات الوصفية Meta Data الخاصة بوصف وتعريف منظور تلك القاعدة ، والتى يتم تخزينها فى قاموس البيانات.

❑ الخصائص العامة لقطاع إدارة التخزين :

يتمـيز قطاع إدارة تخزين البيانات بالخصائص التالية :

- أنـــه هـو القطاع الذى يواجه وسائط التخزين المادى لبيانات القاعـدة (أقراص التخزين المغناطيسي مثلا).

- أنـــه يمثل ذلك الجزء من نظام إدارة قاعدة البيانات المسئول عن كافة عمليات : التخزيـن المـادى لبـيانات القـاعدة بأماكنها الصحيحة داخل وسائط التخزين – والوصـول الفعلـي الى البيانات المطلوبة لتنفيذ الاستفسارات والعمليات فى مواقع تواجدهـا الفعلـي على تلك الوسائط ، واسترجاع هذه البيانات وحدها بالذاكرة ... وذلـك حـتى يتمكـن قطاع معالجة الاستفسارات والعمليات من تنفيذ الإجراءات المطلوبة عليها.

- أن قطاع إدارة تخزين البيانات يقوم بتلك الوظائف بالتعاون مع نظام التشغيل الذى يسـتخدمه الحاسب Operating Systems (مثل نظام Windows 2000) ، وذلك من خلال واجهة للتعامل بينهما.

- ولكن في حالة نظم قواعـد البيانات ، فإن دور نظام التشغيل سوف يقتصر غالبا علـى توفـير الخدمات الأساسية Most Basic Services ، ويتم بناء نظام قاعدة البـيانات على هذا الأساس. وتبعا لذلك فإن تصميم قطاع إدارة التخزين فى أى نظـام لإدارة قواعـد البـيانات يجب أن يأخذ فى الاعتبار ضرورة وجود واجهة للتعامل بين قطاع إدارة التخزين وبين نظام تشغيل الحاسب محل الاهـتمام.

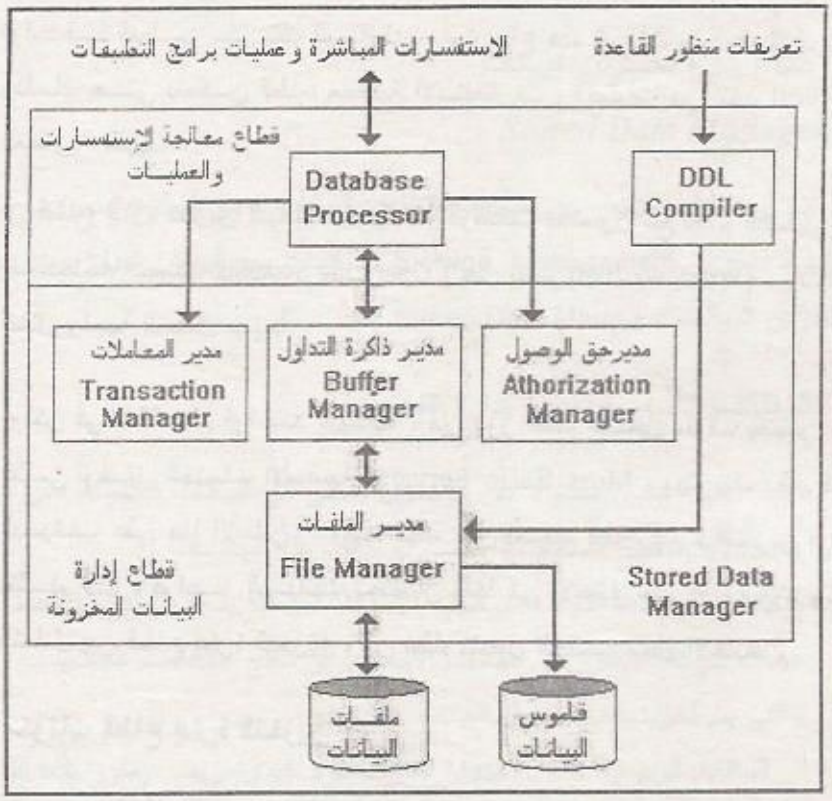
❑ مكونات قطاع إدارة التخزين :

يتضـمن الشكل المعروض في الصفحة التالية – شكل رقم (٢-١٠) – المكونات الأساسية لقطاع إدارة تخزين بيانات القاعدة ، وعلاقات التفاعل القائمة بينها ، تشمل :

- **مدير الملفات** File Manager : وهو المسئول عن عملية تخصيص أماكن التخزين المناسبة للملفات ، بالتعاون مع نظام التشغيل.

- **مديـر ذاكرة تداول البيانات** Buffer Manager : وهو الذى ينظم الاحتفاظ المؤقت بالبيانات المطلوبة بالذاكرة ، لتفادى تكرار عمليات الوصول إلى أقراص التخزين.

- **مديـــر المعـــاملات** Transaction Manager : وهو المسئول عن عملية تخصيص أماكن التخزين المناسبة للملفات علي الديسك ، بالتعاون مع نظام التشغيل.



الشكل رقم (٢-١٠)

مكونات قطاع معالجة الاستفسارات والعمليات
في نظام إدارة قاعدة البيانات

- **مديـــر ذاكـــرة تداول البيانـــات** Buffer Manager : هذا المكون هو الذى يراقب إستكمال إجراءات تنفيذ أي عملية من عمليات معالجة البيانات حتى نهايتها ، بحيث إذا حـــدث خلل أو توقف أثناء التنفيذ فإنه يقوم بإعادة البيانات الى الحالة التي كانت

عليها قبل البدء فى تنفيذ هذه العملية غير المستكملة ، تحت شعار "إما الكل أو لا شيء" ... وذلك من أجل توافق البيانات المخزونة في القاعدة.

أيضا ، وفي الأحوال التي يحاول فيها عدد من المستخدمين تحديث ذات البيانات في ذات الوقت ، فإن مدير المعاملات يقوم بفرض نوع من الرقابة الآنية Concurrency-Control على مثل هذه العمليات ، وذلك من أجل منع حدوث أي تضارب أو تعارض بينها ، حفاظا علي توافق البيانات.

- مدير التكامل وتراخيص الوصول Authorization & Integrity Manager : وهو المسئول عن مراقبة الإلتزام بتطبيق تراخيص الوصول التي سبق تحديدها بمعرفة مدير قاعدة البيانات DBA للمستخدمين المختلفين ، فلا يسمح لأى مستخدم بالوصول إلى بيانات معينة أو إجراء عمليات معالجة علي هذه البيانات إلا في حدود الترخيص الممنوح لهذا المستخدم.

كما أنه يقوم بمراقبة هذه العمليات أثناء التنفيذ ، للتحقق من إلتزامها بتطبيق الاشتراطات والقيود Constraints الموضوعة من قبل ، لضمان صحة وتجانس وتكامل البيانات المخزونة في القاعدة.

(ثامنا) تصنيف نظم إدارة قواعد البيانات

هناك سبعة أسس مختلفة لتصنيف نظم إدارة قواعد البيانات.[1] من أبرز هذه الأسس ، ما يلي :

---

[1] في شرح تفصيلي مقارن لهذه الأسس ، راجع :

- Ramez Elmasri, & Shamkant B. Navathe, Fundamentals of Database Systems, 2nd ed., ***Opt. Cit.***, pp. 34-35.

- **عـــدد المستخدمين** Number of Users :
  - ⇦ نظم المستخدم الفردى Single-User Systems.
  - ⇦ نظم تعدد المستخدمين Multi-User Systems.

- **عـــدد المواقـــع** Number of Sites :
  - ⇦ نظم مركزية Centalized Systems (أي أن قاعدة بياناتها محفوظة في حاسب واحد).
  - ⇦ نظـــم موزعة Distributed DBMS (وهى تلك التي تتوزع بيانات علي حدة حاسبات مرتبطة معا بشبكة Computer Network).

- **نموذج البيانـــات** Data Model :

والمقصود هنا هو "نموذج البيانات" الذى يتأسس عليه نظام إدارة قاعدة البيانات. ويعتبر هذا المعيار في الواقع هو المعيار الأساسي Main Criterion لتصنيف تلك النظم. وتبعا له : فإن غالبية نظم إدارة قواعد البيانات الجاهزة يتم تصنيفها علي أساس أحد النماذج الثلاثة التالية :

- ⇦ النموذج العلاقي Relational Model.
- ⇦ النموذج الشبكي Network Model.
- ⇦ النموذج الهرمي Hierarchical Model.

... هـــذا ، وسوف نعود إلى تناول موضوع نماذج البيانات Data Models بالتفصيل في الفصل الثالث من هذا الكتاب.

* * *

المبحث الثالث

مفهوم : قاعدة البيانات

The Concept of Database

من حيث المبدأ ، يمكن تعريف قاعدة البيانات Database بأنهـا : **أي تجمّع منظم لبيانات متجانسة** A collection of Related Data **أو مترابطة** ، سـواء تم تسجيل هذه البيانات يدويـاً أو إلكترونيـاً.

كـن هذا التعريف المبدئي لقاعدة البيانات لا يكفى لإبراز الطبيعـة الفنية الخاصة التى تتسـبها فى إطار نظم قواعد البيانات ومنهجية قواعد البيانات ... إنه تعريف غير محدد لمعالم ، وفضفاض ، الى الحد الذى قد يجعلنا نعتبر – بالخطأ – أن الجريدة اليومية تعتبر اعدة بيانات! أوليست تنطوي على تجميع منظم لبيانات متجانسة؟

(أ) خصائص قاعدة البيانات

فـي الحقيقة : ليس كل **تجميع منظم للبيانات** دليل علي أننا أمام " قاعدة بيانات " بالمفهوم الاصطلاحي. فحتى يكتسب هذا التجميع صفة قاعدة البيانات ، يجب أن يتوافر له مجموعة من الخصائص ، لا تغـنى إحداها عن وجود الأخريات.

هذه الخصائص نوجزها فيما يلي :

- **أن تغطي القاعدة أحد مجالات الواقع العملي بالتحديد.**

فيجب أن تتعلق القاعدة بوصف جانب محدد بدقة أو جزء معين من عالم الواقع : Some Aspect of the Real World ، وتنعكس فيها أية تغيرات تحدث في هذا العالم الصغير Miniworld. فلا توجد قاعدة بيانات مطلقة ، أو شاملة جامعة!! بل يجب أن تدور تلك القاعدة وجودا وعدما حول مجال أن موضوع محدد ، مثل المخازن – الامتحانات – المبيعات ، أو حتى شركة من الشركات ... الخ.

- **أن يكون للقاعدة مصدرا معلوماً تتدفق منه البيانات.**

فالبيانات هي مدخلات القاعدة التي تتدفق إليها من بيئة النظام ، وهى التي تصف الأحداث التي تدخل في نطاق هذه القاعدة. وكما تحتم تعيين هذا المجال بدقة ، فإنه يتحتم كذلك تحديد المصادر التي تتدفق بهذه البيانات الواردة منه. فمثلا : إن مصدر البيانات التي تصف عملية شراء معينة هو "أمر التوريد" ، ومصدر البيانات التي تصف سداد القيمة المستحقة للمورد هو "شيك المدفوعات" . إن معلومية المصدر هو الذى يضمن أن تكون البيانات متجانسة. وهذا التجانس هو الذي يسمح بالربط بينها ، مثل البيانات المالية فى منشأة تجارية ، أو بيانات المرضى فى إحدى المستشفيات. فكل قطعة بيانات منها يمكن ربطها بالأخريات. أما البيانات المتنافرة أو العشوائية ، فإن تمثل جزرا معزولة أو مفككة ، ولا تصلح كمدخلات لقاعدة بيانات سليمة.

- **أن تهدف القاعدة الي خدمة غايات مقصودة ومحددة Specific Purposes.**

فقاعدة البيانات ليست هدفا في حد ذاتها ، بل وسيلة لتحقيق هدف. هذا الهدف هو تلبية احتياجات معروفة أو متوقعة لبرنامج تطبيقي معين أو أكثر ، أو لطائفة محددة من المستخدمين المحتملين ، وذلك علي سبيل المشاركة. وبدون معرفة هذه الاحتياجات بدقة ، لن يكون تصميم القاعدة سليما أو ناجحا.

- أن تكون البيانات محفوظة بشكل منظم Organized Data .

وذلـك على النحو الذى يسـاعد المستخدمين المختلفين على القيام بعمليات البحث والاستعلام عن البيانات المطلوبة ، واسترجاع تلك البيانات أو تحديثها أو تلخيصها أو طباعـتها عـند الحاجة إليها ، بكفاءة وسـرعة. هذا الحفظ المنظم ليس عملية إجـتهادية ، وإنما يتم وفقا لمقتضيات منهج قواعد البيانات ، وباستخدام أحد النماذج المعـروفة لوصف وتصميم هيكل القاعدة ، مثل : نموذج قاعدة البيانات العلاقية – محل الإهتمام في هذا الكتاب.

ونخلص من ذلك الى طرح التعريف الشامل التالي لقاعدة البيانات :

> قـاعدة البيانات DB هي : **وعاء أو مستودع شامل** Repository ، يحتوى على تجميع منظم لبيانات متجانسة ، يكون لها بالضرورة مصدر معلوم لتدفق البيانات ومسارات محددة للتفاعل مع أحداث الواقع الذى تمثله هذه القاعدة ، وتعمل على تلبية احتياجات معروفة أو متوقعة لمجموعات محددة من المستخدمين على سبيل المشاركة.

ثانيا) أنـواع قواعـد البيـانات

يمكن تصنيف قواعد البيانات علي أسس عديدة أهمها :

- التصنيف بحسب **درجة الميكنة** : إلى قواعد يدوية وأخرى إلكترونية.
- التصنيف بحسب **حجم القاعدة** : إلى قواعد صغيرة أو مكتبية وقواعد كبيرة.

وفيما يلي نبـذة موجزة عن كل نوع.

(١) التصنيف بحسب درجة ميكنة قاعدة البيانات

تنقسم قواعد البيانات ، من زاوية استخدام أو عدم استخدام تكنولوجيا المعلومات ، الي نوعين : قواعد بيانات يدوية ، وقواعد بيانات إلكترونية.

أ – قاعدة البيانات اليدوية Manual Database

تمثل قاعدة البيانات اليدوية الشكل التاريخي أو النمط التقليدي القديم لتصميم قواعد البيانات ، وهى التى يتم إنشائها والتعامل معها فى كل مراحلها بالجهود الإنسانية المباشرة ، دون وساطة الحاسبات أو برامج إدارة قواعد البيانات.

من أبرز أمثلة قواعد البيانات اليدوية : بطاقات الكتب والمجلات التى تحتفظ بها إحدى المكتبات Library Card Catalog. هذه القاعدة قد تتضمن مئات الآلاف من بطاقات المعلومات ، مصنفة فى مجموعات على أسس متنوعة مثل : عنوان الكتاب ، اسم المؤلف ، موضوع الكتاب..الخ. ويتم حفظ كل مجموعة علي حدة ، مع ترتيب بطاقاتها على أساس أبجدي Alphabetic Order ، وذلك لغرض تيسير عمليات البحث والاستعلام اليدوي.

ب– قاعدة البيانات الإلكترونية Computer-Based Database

المقصود بقاعدة البيانات الإلكترونية هو : قاعدة البيانات المبنية على تكنولوجيا المعلومات Information Technology ... أى التى تعتمد فى إنشائها واستخدامها علي التفاعل بين كل من العنصر البشرى والعناصر المادية التي تتكون منها تكنولوجيا المعلومات ، وتشمل : الحاسبات والأجهزة الملحقة – شبكات الحاسب – البرمجيات Computer Software ، وبخاصة نظام إدارة قاعدة البيانات DBMS.

هذا النوع من قواعد البيانات هو النوع محل الإهتمام في هذا المؤلف ، علي إعتـبار أنـه يمثل حجر الزاوية الذى يتأسس عليه أي نظام إلكتروني للمعلومات ، سـواء كان نظاما للمعلومات الإدارية MIS ، أو لدعم القرارات DSS ... أو كان تطبيقامن تطبيقات معالجة العمليات والإنشطة TPS.

ملحوظة : من الآن فصاعدا : يجب عليك ملاحظة أن "قاعدة البيانات الإلكترونية" هى النوع المقصود حيثما تذكر كلمة "قواعد بيانات" في كل أنحاء هذا المؤلف.

(٢) التصنيف بحسـب : حجـم ، ودرجة تعقيـد قاعدة البيانات

كما تختلف قواعد البيانات من حيث : حجمها DataBase Size ، ودرجة تعقيدها DataBase Complexity. ومن هذه الزاوية ، يمكن لنا تقسيم قواعد البيانات الى نوعين مختلفين : قواعد بيانات صغير الحجم – وقواعد بيانات متوسطة أو كبيرة الحجم.

– قواعد البيانات صغيرة الحجم
Small-Size Database

وهـى التى تقتصر على عدد محدود من السجلات ، وتتضمن هياكل بيانات بسيطة وغير معقدة ، وذات أحجام صغيرة ... ويستخدمها عادة مستخدم واحد فقط أو عدد محدود من المستخدمين. ومن أمثلتها :

- قواعـد البـيانات الشخصية ، مثل قاعدة بيانات المكتبة المنزلية أو قاعدة بيانات شرائط الفيديو ... الخ.
- قواعـد البيانات الفردية فى المنشآت الصغيرة ، مثل قاعدة بيانات الأدوية في إحدى الصيدليات ، أو قاعدة بيانات الأصناف في مركز بيع قطع غيار السيارات ، أو قاعدة تحصيل إشتراكات الأعضاء في أحد النوادي.

ب- قواعد البيانات المتوسطة أو كبيرة الحجم
Large or Medium-Size Database

وهى قواعد البيانات الضخمة ، التى يستخدمها عدد كبير من المستخدمين (من ٢٥ مستخدم الى مئات المستخدمين) ، والتى تنطوي على تخزين مئات الآلاف من سجلات البيانات في قواعد ذات هياكل معقدة ... وتنطوى علي التعامل اليومي مع المئات من الاستفسارات والتطبيقات الجاهزة.

مثل هذه القواعد يتم تصميمها ، فى أغلب الأحوال ، للعمل كجزء من نظام شامل للمعلومات على مستوى المنظمة بأسرها ، وبخاصة :

- فى الأجهزة الحكومية المركزية ، كالوزارات ومصلحة الضرائب.
- في مؤسسات الأعمال الكبرى ، كالجامعات والشركات التجارية والمصانع والبورصات والمستشفيات
- في الشركات ذات الفروع المتعددة ، كالبنوك وشركات التأمين والطيران.

ثالثا) أنواع البيانات المخزونة في القاعدة

كما أوضحنا من قبل عند عرض منهجية قواعد البيانات ، فإن البيانات التى يتم تخزينها فى نظام قاعدة البيانات لا تقتصر فقط على البيانات الفعلية الخاصة بالأحداث التى أنشئت القاعدة من أجل تسجيلها ومعالجتها ، ولكنها تشتمل أيضا على نوع آخر من البيانات ، هو البيانات الوصفية Meta Data الخاصة بتعريف ووصف هيكل أو منظور قاعدة البيانات.

وعلي ذلك ، فإن البيانات المخزونة في أي قاعدة بيانات ، مهما كان نوعها تنقسم – بحسب طبيعتها – إلى نوعين : بيانات فعلية Actual Data – وبيانات وصفية Meta Data . وفيما يلي وصفاً موجزا لطبيعة وخصائص كل نوع.

(١) البيانـات الفعـلية Actual Data :

البـيانات الفعلـية هـي البيانات التي تم جمعها وتسجيلها في القاعدة من أجل تمثيل ووصـف الأحـداث الـتى تدخل فى مجال تطبيق تلك القاعدة. ومن أمثلتها بيانات العملاء أو بيانات الموردين أو بيانات المنتجات ... الخ.

هذه البيانات تعتبر هي جوهر نظام قاعدة البيانات ذاته ، وهي مبرر الوجود الذى تم من أجله إنشاء هذا النظام. وتتميز هذه البيانات بصفتين :

- أن البيانات الفعلية هي التي سوف تخضع لعمليات المعالجة المختلفة ، بما في ذلك عمليات التخزين والاسترجاع والتخليص والتحديث والطباعة ...، وذلـك وفقـا لاحتياجات المستخدمين المختلفة التي تتبلور فى : الاستفسار المباشرة أو التطبيقات الجاهزة.

- أن كافـة نماذج البيانات Data-Models تتفق معاً علي ضرورة حفظ هذه البـيانات في شكل سجلات Records ، لكنها تختلف عن بعضها البعض بشأن كيفية تجميع وربط هذه السجلات معا في علاقات ذات معني. في الـنموذج العلاقـي مثلا ، يتم حفظ كل مجموعة متجانسة من السجلات – مـثل سجلات المنتجات أو سجلات طلبيات البيع – في شكل جدول بيانات Data-Table ... الأمر الذى سوف نتناوله بالتفصيل في الفصل القادم.

(٢) البيـانات الوصـفيّة Meta Data :

البـيانات الوصفية Meta Data ، هي **بيانات عن البيانات** ! فهى لا تصف الأحداث الداخلـة في مجال القاعدة ، لكنها تصف هيكل أو منظور قاعدة البيانات التي تستخدم في تسجيل تلك البيانات.

وتتميز البيانات الوصفية بثلاث خصائص هامة ، هي :

- إنها تتضمن البيانات الخاصة بوصف "منظور أو هيكل قاعدة البيانات" Data Schema ، والتى تشمل بوجه خاص الأنواع التالية :
  - ما هي البيانات التى يجب تخزينها فى القاعدة ، والعلاقات القائمة بينها والقيود والتحفظات المفروضة عليها (المنظور المنطقى).
  - ما هى طريقة تخـزين واسـترجاع تلك البيانات على وسائط التخزين المستخدمة (المنظور المادى).
  - مـا هى التشكيلات المختلفة من البيانات Data-Views اللازمة لتلـبية احتياجات المستخدمين المختلفين (المنظور الخارجى).
- أن هـذه البيانات الوصيفة يتم تسجيلها في شكل **تعليمات** Codes مكتوبة بلغة تعريف البيانات DDL الخاصة بنظام إدارة قاعدة البيانات المستخدم.
- أن هـذه التعليمات يتم حفظها داخل ملف خاص من ملفات النظام يطلق عليه اسم : **قاموس البيانات** Data Dictionary.

ويلاحـظ أنه في حالة نظم قواعد البيانات الكبيرة ، قد تشتمل البيانات المخزونة في قاعدة البيانات – بالإضافة الي جداول البيانات وقاموس البيانات – على نوعين آخرين من البيانات الوصفية اللازمة لتحسين أداء النظام ، هما :

- **الفهارس Indices** ، وهى جداول تكميلية لفرز السجلات أو البيانات علي أساس أو أسـس معيـنة (ترتيـب تصاعدى أو تنازلي مثلا) ، الأمر الذى يكفل سرعة الوصول إلى البيانات المطلوبة التى تحتوى على قيم معينة.
- **البيانات الإحصائية Statistical-Data** ، وهي التى تتضمن مؤشرات عن أحوال البيانات المخزنة في قاعدة البيانات واستخداماتها الحالية. ومن أهم الاستخدامات الحـيوية لهـذه المؤشرات الإحصائية أنها تعمل علي مساعدة نظام إدارة قاعدة البيانات DBMS في اختيار أفضل طريقة لتنفيذ الاستفسار الحالي.

❑ أهمية ووظيفة قاموس البيانات :

إن قـاموس البيانات يعتبر بحق من أخطر عناصر قاعدة البيانات ... إذ أنه بدون التصـميم والتنفيذ الجيد لهذا القاموس ، فإن كفاءة أداء النظام بأسره تصبح عسيرة المنال. كمـا أن فقـدان أو تلف هذا القاموس يعنى في الواقع فقدان إمكانية الوصول إلى البيانات المخزونة واسترجاعها ، وبالتالي توقف النظام بأسره عن العمل.

ويفيـد قاموس البيانات فى خدمة غرضين :

الأول : أنـه قـبـل أن يمكن نظام إدارة قاعدة البيانات DBMS من تنفيذ أي محاول من جانـب أحد المستخدمين لإسترجاع ومعالجة أي بيان مخزون في القاعدة ، سواء كانـت في شكل استفسار مباشر أو من خلال تطبيق جاهز ، فإن هذا النظام يلجأ أولا إلي قاموس البيانات ، لاستشارته بشأن عوامل متعددة أهمها :

- هل البيان المطلوب يعتبر من البيانات المخزونة فعلا في القاعدة ؟
- هل من حق هذا المستخدم بالذات الوصول إلى هذا البيان أو القيام عملية المعالجة المطلوبة عليه ؟
- كيف يمكن الوصول إلى مكان وجود هذا البيان وإسترجاعه ؟

الثاني : أن المستخدمين الخبراء Sophisticated Users ، سواء كانوا من المبرمجين أو الإخصـائيين ، قـد يلجأون – عند الضرورة – إلى قاموس البيانات ، من أجل تكوين فكرة وافية وموضوعية عن الهيكل الحالي لقاعدة البيانات.

* * *

الفصل الثالث

أساليب
وصف هيكل أو بنية قاعدة البيانات

Abstracution Of Database Structure

المبحث الأول : نماذج تمثيل البيانات.

المبحث الثاني: المنظور ثلاثي المستويات.

مقدمة :

إن إحـــدى الخصـــائص الرئيسية لمنهج " قواعد البيانات" هى : الاهتمام بتعريف هيكل البيانات The Database Structure ، أي وصف البنية الداخلية لقاعدة البيانات Database Architecture. إن العملية التى تستهدف تـــقديم مثل هذا الوصف بأسلوب منهجي تسمي : **نمذجة البيانات** Data Modeling.

وهناك أسلوبان مختلفان لوصف قاعدة البيانات :

الأول : هـــو التمثـــيل النظرى لهيكل البيانات باستخدام ما يعرف باسم : **نماذج البيانات** Data Models.

الثانى : هو وصف البنية الداخلية للقاعدة باستخدام ما يسمي: **المنظور ثلاثى المستويات** The Three-Levels Architecture.

هذان الأسلوبان ليسا بديلان ... فلا تغنى أحدهما عن وجود الأخر. كما أن كل أسلوب منهما ينطوى علي عـــدة **مستويات من التجريد** Abstration Levels ، تعمل علي وصف نفس البيانات ولكن من زوايا مختلفة وبدرجات مختلفة من التعقيد والتفاصيل الفنيـــة.

إن فهـــم الكيفية التي تعمل بها كل طريقة منهما علي تجريد ووصف هيكل أو بنية قـــاعدة البـــيانات سوف تمكننا من تناول مجموعة أخرى من المفاهيم الأساسية المرتبطة ذات الصلة الوثيقة بنمذجة البيانات.

وتبعا لذلك ، فإن الدراسة في هذا الفصل سوف تنقسم إلى مبحثين متكاملين ، هما :[1]

- نمـــاذج تمثيـــل البيانـــات **Data Models**.
- المنظور الثلاثى المستويات **The Three-Schema Architecture**.

* * *

(١) للتوسع فى دراسة موضوع **نمذجـــة البيانـــات** ، أنظر بوجه خاص :

- S. Abiteboul, R. Hull, & V. Vianu, ***Foundation of Databases,*** (Readings, Massachusetts: Addison-Wesley Publishing Co., 1995);
- W. Kim, Ed., ***Modern Database Systems***, (Readings, MA: ACM Press/Addison-Wesley, 1995);
- P. O'Neil, ***Database: Principles, Programming, Performance***, (San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1995);
- Abraham Silberschatz, Henry F. Korth, & S. Sudarshan, ***Database System Concepts***, (New York: The McGraw-Hill Companies, Inc.,1997); &
- M. Stonebraker, Ed., ***Readings in Database Systems,*** 2nd ed., (San Mateo, CA: Morgan Kaufmann Publishers, Inc., 1994).

١ المبحث الأول

نمـاذج تمثيل البيانات
Data Models

نظراً لأن مستخدمي نظام قاعدة البيانات ينقسمون الي طوائف متعددة ، منها مدير قـــاعدة البيانات DBA ، المبرمجين ، والمستخدمين الخبراء ، والمستخدمين العاديين ، وغيرهم كثيرين ... ونظرا لأن هذه الطوائف تختلف عن بعضها الـــبعض في مستوى الخبرة والدراية بنظم قواعد البيانات ولغاتها وإمكانياتها واشـــتراطاتها ، فإن غالبيـــة هؤلاء المستخدمين لا يفهمون التفاصيل الفنية شديدة التعقيد التي ينطوى عليها هيكل البيانات.

ويصبح المطلوب هو البحث عن وسيلة لأخرى لتجريد البيانات علي عدة مستويات متدرجة من السهولة أو التعقيد ، بحيث يسهل فهمها من جانب المستخدمين المختلفين. لقد وجد العلماء الحل الأمثل لهذه المشكلة فى نماذج البيانات : Data Models . إن **نمـــاذج البـــيانات** مـــا هى إلا أدوات نظرية لوصف هيكل البيانات ، وذلك بمستويات مختلفة من التجريد bstractuion Levels ... هذه المستويات التي تقدم وصفا مفهوما لهيكل القاعدة ، مع حجب التفاصيل المعقدة الخاصة بالبيانات المخزونة في القاعدة ، تلك التفاصيل التي لا يحتاج اليها معظم المستخدمين.

في هذا المبحث ، سوف نناقش **نماذج البيانات** من ثلاث زوايا متكاملة :[1]

- ❑ طبيعـــة وأهميـــة نماذج البيانات.
- ❑ خصائص وأنواع نمـــاذج البيانات.
- ❑ نموذج **الكيان–العلاقة** E-R Model.
- ❑ استخدام نموذج الكيان– العلاقة في وصف قاعدة البيانات.

* * *

(١) الفـكر المتاح حافل بعدد ضخم من المراجع التى تناقش موضوع **نماذج البيانات** ، انظر منها بوجه خاص :

- A. H. Khalil, ***Database Systems,*** (Cairo, Egypt: Ain Shams Library, 1998);
- C. Batini, S. Ceri, & S. Navathe, ***Database Design: An Entity-Relationship Approach***, (Redwood City, CA: The Benjamin/ Cummings Publishing Co., 1992);
- Ramez Elmasri, & Shamkant B. Navathe, ***Fundamentals of Database Systems,*** 2nd ed., (Redwood City, CA: The Benjamin/ Cummings Publishing Co., 1994) ; &
- J. D. Tsichritzis, & F. Lochovsky, ***Data Models,*** (Englewood Cliffs, NJ: Prentice-Hall, 1982).

أولا) طبيعة وأهمية نماذج البيانات

يمكن تعريف **نموذج البيـــانات A Data Model** ، بأنـــه : مجموعة من المفاهيم والأدوات الـــنظـــرية Conceptual Concepts & Tools ، التى يمكن استخدامها لغرض وصف هيكل قاعدة البيانات ، وذلك بشكل مستقل تماما عن أي نظام من نظم إدارة قواعد البيانات DBMS. علما بأن المقصد بهيكل قاعدة البيانات Database Structure هـــو : أنواع البيانات والعلاقات والقيود التي تخضع لها عمليات حفظ وإسترجاع هذه البيانات.

بهذا المفهوم ، تصبح أهم سمة من سمات **نموذج البيانات** هي أنه يمكن تعريفه في مـــرحلة مبكرة من مراحل تصميم نظام قاعدة البيانات ، بل وغالبا من قبل أن يتم اختيار نظـــام إدارة قـــاعدة البيانات المناسب ذاته. لأن نموذج البيانات يقدم وصفا لهيكل القاعدة محل التصميم لا يتقيد بأى نظام من تلك النظم.

❑ أهمية نماذج البيانات

إن وجود نموذج يصف الهيكل الشامل للبيانات في المرحلة التمهيدية لتصميم أي نظام لقواعد البيانات إنما يحقق العديد من الفوائد والمزايا ، نذكر منها ما يلي :

١) أن وجود **نموذج** لوصف البيانات في هذه المرحلة المبكرة سوف يمكن محلل النظام System Analyst مـــن التركيز علي وصف الخصائص العامة Data Properties للبـــيانات المطلـــوب تخزينها ، وذلك دون التعرض لخصوصيات نظام معين لإدارة قواعـــد البـــيانات DBMS. وهذا الوصف يساعد محلل النظام علي إنجاز المراحل التالية لعملية تصميم النظام محل الاهتمام بسهولة وسرعة.

٢) أن وجـــود **نموذج** لوصف البيانات سوف يجعل من السهل مقارنة وتقـــييم النظم المخـــتلفة لإدارة قواعـــد البيانات ، من أجل اختيار النظام الأصلح الذى يـــتوافق مع كل من :

- طبيعة البيانات الداخلة في نطاق القاعدة وخصائصها.
- احتياجات مستخدمي نظام قاعدة البيانات تحت التصميم .

وذلك بالمستوى المرغوب فيه من الكفاءة والفاعلية.

٣) كما أن وجود **نموذج** لوصف البيانات يجعل من اليسير - فيما بعد - تحويل هذا النموذج أو ترجمته الى ما يقابله من منظور للبيانات Data Schema ، وبخاصة على مستوى المنظور الخارجي والمنظور المنطقي.

ثانيا) أنواع نماذج البيانات

Categories of Data Models

يتضمن الفكر المتاح عددا كبيرا من أنواع نماذج البيانات. وقد اهتم الكتّاب والعلماء بتصنيف هذه النماذج الى مجموعات متميزة ، وذلك على أسس مختلفة. بيد أن التصنيف المناسب لأغراض هذا الكتاب هو ذلك الذى يقسّم نماذج البيانات الى ثلاثة مجموعات متجانسة. هذه المجموعات هى :

- النماذج المادية Physical Models.
- النماذج النظرية أو المنطقية Conceptual or Logical Models.
- النماذج التمثيلية أو التنفيذية Representational or Implementation Models.

في الفقرات التالية ، تركز الدراسة علي تقديم فكرة موجزة عن طبيعة ، ومستوى تمثيل ، نماذج كل مجموعة.

١) النماذج المادية لتمثيل البيانات
Physical Models

النماذج المادية تعتبر من نماذج المستوى الأدنى Low-Level Data Models، التى تقدم مفاهيم لوصف التفاصيل الفنية الدقيقة والمعقدة الخاصة بكيفية تخزين البيانات

الخاصـــة بالقاعدة وتنظيم هياكلها. هذه المفاهيم موجهة أصلا الى المتخصصين فى تصـــميم قواعد البيانات ، وليس للمستخدمين العاديين. ومن أمثلة هـــذه المفاهـــيم : شـــكل الســـجل Record Format ، وترتيـــب السجلات Records Ordering ، ومسارات الوصول الى البيانات Data Paths ... الخ.

وفى الواقع ، إن النماذج المادية المتاحة قليلة ، نذكر منها :

- نموذج الذاكرة – الإطار : Frame-Memory Model
- نموذج التوحـــيد : Unifying Model.

٢) **النماذج المنطقية أو النظريـــة**
Logical or Conceptual Models

النماذج المنطقية أو النظرية لتمثيل البيانات تعتبر – بطبيعتها – من نماذج المستوى الأعلى High-Level Data Models ، التى تقـــدم مفاهيم سهلة الفهم لوصف قاعدة البـــيانات ، مفاهـــيم تتناسب مع الطريقة التى ينظر بها الكثيـــر من المستخدمين إلى بيانات القاعدة.

من أمثـــلة هذه المفاهيم – ذات المعانى الإصطلاحية – ما يلي :

- مفهوم الكيانات Entities.
- مفهوم العلاقات Relationships.
- مفهوم الخصائص أو السمات Attributes.
- مفهوم القيود والتحفظات Constraints ... الخ.

ويمكن القول بأن النموذج الذى يعتبر من أبرز النماذج المنطقية ، وأكثرها عمومية واســـتخداما فى تمثيل البيانات ، النموذج المعروف باسم : **نموذج الكيان–العلاقـــة** Entity-Relationship Model. وســـوف نعود الى مناقشة هذا النموذج بالتفصيل في المبحث القـــادم.

٣) النماذج التنفـــيذية لتمثيل البيانات
Implementation Models

تحتل النماذج التنفيذية لتمثيل البيانات ، وتسمى أيضا النماذج التمثيلية أو التجسيدية : Representational Models ، مـــنطقة الوسط بين النوعين السابقين. فهي تقـــدم مفاهـــيم لوصـــف البيانات من النوع الذى يمكن فهمه بواسطة المستخدمين النهائيين لقـــاعدة البيانات ، لكنها فى الوقت نفسه لا تعزل نفسها تماما عن الكيفية التى يتم بها تنظـــيم وتخزين البيانات. لهذا الســـبب ، فإن النماذج التنفيذية تمثل النماذج الأكثر استخداما وشيوعا في وصف بنيـــة قواعد البيانات فى الوقت الحاضر.

ومـــن أهم النماذج التابعـــة لهذه الفئـــة ثلاثة نماذج هي : النموذج العلاقي ، والنموذج الشبكى ، والنموذج الهرمي ... علما بأن بعض العلماء يضيف اليها نوعا رابعا ، هو النموذج الشيئى Object-Oriented Model. (١)

وفيما يلي عرض موجز لطبيعة وخصائص كل نوع.

❑ النمـــوذج العـــلاقـــى
Relational Model

يقوم النموذج العلاقي بوصف هيكل قاعدة البيانات على أنـــه مجموعة من جداول البيانات Data Tables المترابطة معا ، ويستخدم هذه الجداول لتمثيل كل من ســـجلات البيانات ، والعلاقات Relationships القائمة بين هذه البيانات. ونظرا لأن هذا النموذج هو محور الدراسة الأساسى في هذا الكتاب ، وسوف نخصص لدراسته الفصل القادم بأكمله ... فإننا سوف نكتفي هنا بتقديم فكرة مبسطة عن كينونته.

---

(١) راجع بوجه خاص :

- Ramez Elmasri, & Shamkant B. Navathe, Fundamentals of Data Base Systems, ***Op. Cit.***, p. 34.

في النموذج العلاقي لتمثيل البيانات ، نجد أن :

- هـــناك **جدول** بيانات Data Table مستقل لتسجيل بيانات المفردات المتعلقة بكل موضوع أو كيان Entity علي حدة.
- كل جدول من جداول القاعدة يكون له بالضرورة اسما مميزا يدل عليه.
- يـــتكون هيكل كل جدول من عدة **حقول للبيانات** Fields ، تخصص لتسجيل بـــيانات متجانســـة لوصف خصائص الموضوع الذى يمثله الجدول ، مثل : بيانات الموردين ، وذلك في شكل **سجلات** Records.
- أن **القـــيم** Values المسجلة فى بعض هذه الحقول سوف تستخدم **كأساس للربـــط** بين سجلات البيانات فى الجداول المرتبطة. ويطلق علي هذا النوع من الحقول اسم : **حقول المفاتيح** Key Fields.

الشكل التالى ، شكل رقم (رقم ٣-١) ، يعرض نموذجا لقاعدة بيانات علاقية بسيطة، تتكون من جدولين للبيانات :

- جدول بيانات العملاء Customer-Information Table
- جدول أرصدة الحاسبات الحسابات Account-Balance Table.

مـــع ملاحظـــة أن الربط بين عميل معين في الجدول الأول وبين رصيد حسابه في الجدول الثاني يتم علي أساس القيم المدرجة **بحقل المفتاح** الموجود في الجدولين وهو حقل : رقم **الحساب** Account-Number.

ومن تأمل هذا النموذج البسيط لتمثيل البيانات ، يتضح لنا ما يلي :

- أن العمـــيل Smith ، مـــثـــلا ، لـــه حساب واحد برقم A-215. إن رصيد حساب هذا العميل فى جدول الأرصدة هو 700 جنيه.
- أن العميل Jonson لـــه حسابين مختلفين هما : رقم A-101 ورقم A-201. الرصـــيد المقابل لكل حساب منهما فى جدول أرصدة الحسابات هو 500 و 900 جنيه ، على الترتيب.

Customer-Information Table

Customer-Name	Customer-Street	Customer-City	Account-Number
Jonson	Alma	Palo Alto	A-101
Smith	North	Rye	A-215
Hayes	Main	Harrison	A-102
Turner	Pultnam	Stamford	A-305
Jonson	Alma	Palo Alto	A-201
Lindsay	Park	Pittsfield	A-222

Account Balance Table

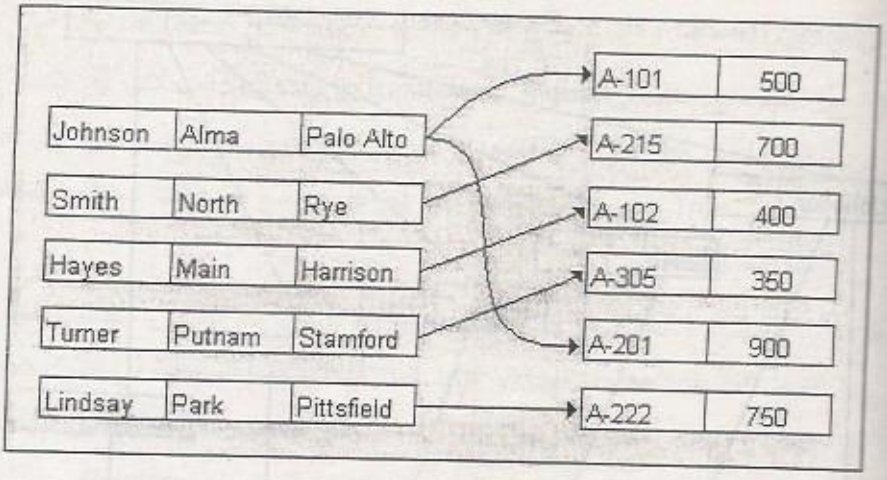
Account-Number	Balance
A-101	500
A-215	700
A-102	400
A-305	350
A-201	900
A-222	700

شكل رقم (٣-١)

نموذج مبسط لقاعدة بيانات علاقية Relational Database

- ❑ النمـــوذج الشـــبكيّ
  Network Model

فـــي هـــذا الـــنموذج ، يتم تمثيل البيانات على أنها مجموعة من **السجلات** Records (بالمعـــنى الـــوارد فى لغة Pascal). كما يتم تمثيل العلاقات القائمة بين البـــيانات ، باســـتخدام كـــل من : السجلات ، **والروابط** Links ، على الترتيب. أما الدلالـــة علـــي وجـــود هذه العلاقات وإظهارها فيتم عن طريق استخدام ما يسمي : **المؤشـــرات** Pointers. والشكل التالي ، يعرض عليك مخططا للنموذج الشبكي ، باستخدام نفس البيانات السابقة :



شكل رقم (٣-٢)

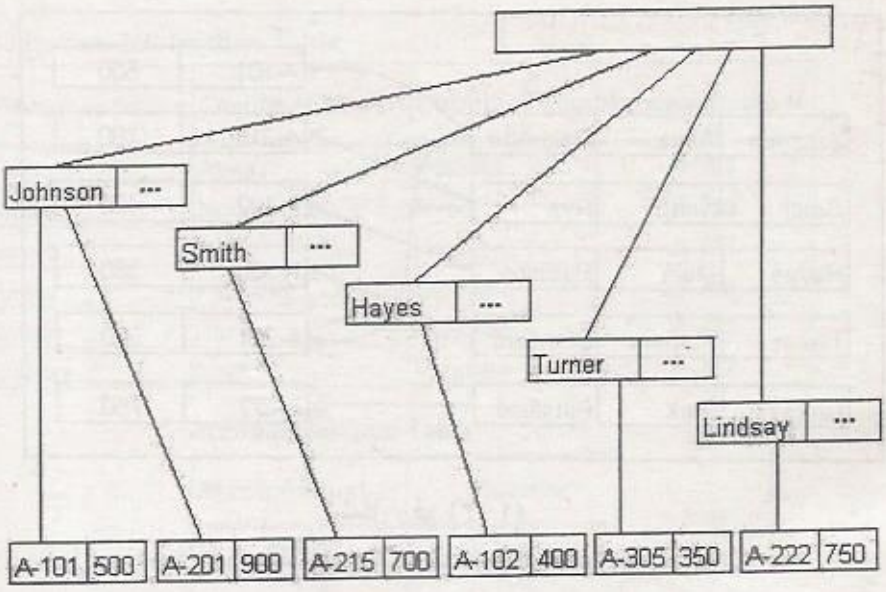
نموذج مبسط لقاعدة بيانات شبكة Network Database

- **النموذج الهرمي أو المتدرج**
  Hierarchical Model

يتشـــابه الـــنموذج الهرمـــي مع النموذج الشبكي فى أنهما يمثلان البيانات والعلاقات القائمة بين البيانات باستخدام السجلات Records والروابط Links. لكن جوهر الإختلاف بينهما يتمثل فيما يلي :

- ⇦ بيـــنما يقوم النموذج الهرمي بتنظيم السجلات في شكل شجرة Tree أو مجموعة من الأشجار المتداخـــلة ذات الفروع المتشابكة ...
- ⇦ فـــإن الـــنموذج الشـــبكي يتعامل مع السجلات كما لو كانت مناظر عشوائية Arbitrary Images.

والشـــكل التالـــي يعـــرض مخططاً مبسطاً للنموذج الهرمي ، باستخدام نفس البـــيانات السابقة. ومنه يتضح أن : كل عميل تربطه بحسابه رابطة واحدة Link ، باستثناء العميل Jonson الذى له حسابين فيتفرع من سجله إليهما رابطتان.



شـــكل رقم (٣-٣)

نموذج مبسط لقاعدة بيانات هرمية Hierarchical Database

وقبل أن نترك هذه النقطة ، يكون من المناسب أن نوضح نواحي الشبه ونواحي الإختلاف بين هذه النماذج الثلاث.

أما عن **نواحي الشبه** ، فتتمثل في أمرين :

- أن الـــنماذج الثلاثة – العلاقي والشبكي والهرمي – تنتمي إلى فصيلة واحدة هي النماذج التنفيذية ، وهى التي تقدم مفاهيم قابلة للفهم لوصف البيانات ... دون أن تعزل نفسها تماما عن كيفية تخزين هذه البيانات.
- أن الـــنماذج الثلاثة – العلاقـــي والشبكي والهرمي – تقوم بتسجيل البيانات على هيئة سجلات Record Structures. لذلك فإنها ، توصف أحيانا بأنها : النماذج المبنيـــة على السجلات : Record-Based Data Models.

أما عن **نواحي الإختلاف** ، فإن النماذج الثلاثة تختلف عن بعضها البعض من زاويتين :

- أن النموذج العلاقـــى Relational Model يقوم بتخزين السجلات المتجانسة في شكل جداول Tables ، أما النموذج الهرمي فإنه يقوم بتنظيم السجلات علي هيئة أشـــجار Tree ، بيـــنمـــا يتعامل النموذج الشبكي مع السجلات المتناثرة كمناظر عشوائية Images.
- أن الـــنموذج العلاقـــى Relational-Model يقـــوم بالربط بين سجلات الجدوال المخـــتلفة باستخدام علاقات الإرتباط المبنية علي القيم المسجلة في حقول المفتاح Key Fields ، أنما النموجين الهرمي والشبكى فيعتمدان في الربط بين السجلات وإظهار العلاقات علي إستخدام الروابط Links والمؤشرات Pointers.

ثالثا) نمـوذج الكيـان - العـلاقـة

Entity-Relationship Model

فـــي الفقـــرات التالية سوف نلقى الضوء علي نموذج **"الكيان – العلاقة"** ، حتى يتضـــح لنا الدور الذى تلعبة النماذج فى تبسيط عملية تمثيل ووصف هيكل قاعدة البيانات. إن تركـــيزنا علـــي هذا النموذج بالتحديد يرجع إلى أنه يقدم بذاته الخلفية الفلسفية للنموذج الأساسي محل الإهتمام في هذا الكتاب ، وهو **النموذج العلاقي** Relational Model.

١) طبيعة نموذج الكيان – العلاقة :

نمـــوذج **"الكيان–العلاقة"** ، ويرمز له بالرمز E-R Model ، هو أحد أنواع النماذج المنطقية أو النظرية : Conceptual Models لتمثيل البيانات. ويقوم هذا النموذج على فكرة النظر الى عالم الواقع Real World على أنه يتكون من : مجموعة من الأشـــياء أو الأشخاص أو الوحدات الاعتباريـــة ، تسمي : **كيانـــات** Entities ، ومجموعة من **العلاقـــات** Relationships التى تربط بين هذه الكيانات.

وقـد نشـأ هذا النموذج من أجل تيسير عملية تمثيل قاعـدة البيانات الخاصة بأي مشروع أو مؤسسة أو وحدة تنظيمية مستقلة ، وذلك عن طريق السماح بتحديد مواصفات مـا يسمى **بمنظـور المشروع أو التنظـيم** : The Enterprise Schema. ويقصد بـه المنظور الذى يمثل الهيكل المنطقي الشامل لقاعدة البيانات.

ويمكن **القول** بأن الفائدة الكبرى من استخدام هذا النموذج فى تمثيل هيكل البيانات ، تتمـثل فى أنـه يرسم بدقة وبساطة **خريطة تفصيلية** للأشياء والأحداث والتفاعلات التى تحـدث فـى الحياة اليومية بالمنظمة محل الاهتمام ، ويترجم هذه الخريطة التفصيلية الى شكل بيانى يسهل فهم دلالته ، يسمي E-R Diagram.

٢) المفاهيم الأساسية في نموذج الكيان– العلاقة
The Basic Concepts of E-R Model

هناك ثلاثة مفاهيم رئيسية يعتمد عليها نموذج "الكيان– العلاقة" ، هى :

- الكيانات **Entities**.
- العلاقات **Relationships**.
- الخصائص أو السـمات **Attributes**.

وذلك بالإضافة الى مجموعة القيود **Constraints** التى يجب أن يلتزم بها مضمون قاعدة البـيانات ، بهدف ضمان صحة وتوافق البيانات Data Consistency . وفيما يلي نقـدم شـرحا موجزا لطبيعة كل مفهوم من هذه المفاهيم الثلاثة ، وما يترتبط به من مصطلحات إضافية ... مع بيان الدور الذى يلعبه فى تمثيل ونمذجـة قاعدة البيانات.

المفهوم الأول : الكيـانات Entities

المقصـود بالكيان An Entity ، بالمعني الاصطلاحي للكلمة ، هو عبارة عن أحد الأشياء Things أو الموضوعات Objects أو الوحدات Units

التى لها وجود فعلى (مادى ملموس) أو وجود اعتباري (معنوى أو غير ملمـوس) فى عالم الواقع الذى تمثله قاعدة البيانات ، والذي يكون قابلا للتمييز عن كافة الأشياء أو الموضوعات الأخرى في هذا العالم.

على سبيل المثال :

- فإن كل موظف فى المشروع يمثل **كياناً** مستقلا وقابلا للتمييز ، وكل طالب في المعهد يمثل **كيانا** وقابل للتمييز. وفي الحالتين ، فإن هذا الكيان له وجود مادى ملموس.
- أيضـــا ، فإن كل قرض من القروض التى يمنحها أحد البنوك لعملائه المختلفين يمكن بالمثل اعتبار أنه يمثل كياناً مستقلا وقابلا للتمييز ، وذلك على الرغم من أن القرض يعتبر وحدة اعتبارية أو معنوية ، ليس لها وجود مادي ملموس.

ويطلـــق مصطلح **مجموعـــة الكيان** Entity-Set علي : جملة المفردات Items أو حصـــيلة الأشـــياء أو الوحدات المتجانسة Collection ، التى تكون حتما من نفس النوع ، والتي تحمل نفس الخصائص أو السمات.

والأمثلة على ذلك عديدة :

- فـــالموظفون فـــى كل أنحاء المشـــروع يمثلون مجموعة كيان واحدة ، ويحمل أعضاؤها ذات الخصائص ، ويمكن أن تسمي **Employee** : Entity Set.
- كمـــا أن جميع العملاء المتعاملين مع أحد البنوك يمثلون مجموعة كيان واحدة ، يمكن أن يطلق عليها اسم **Customer** : Entity Set.
- وبالمـــثل ، فإن كافة القروض من أي نوع التى يمنحها هذا البنك لعملائه تمثل مجموعة كيان متميزة ، يمكن أن تسمى **Loan** : Entity Set.

هذا ، ويلاحظ أنه يمكن اعتبار أن كل موظف ، أو عمـــيل ، أو قـــرض بمفـــرده إنمـــا يمثل فى حـــد ذاته امتدادا An Extension لمجموعته الأصلية.

المفهوم الثاني : الخصائص أو السـمات Attributes

إن كل كيان من الكيانات بذاته Single Entity يتم وصفـه والتعبير عنه فى حدود مجموعة من الخصائص أو السمات A Set of Attributes.

إن هـذه الخصائص تستخدم – فى مجموعها- كأساس لتعـريف هذا الكيان ، وتميـيزه عـن غيره من الكيانات أو المفردات الأخرى في مجموعة الكيان. بمعنى أن تمثـيل كيان معين أو التعبير عنه يكون فى حـدود مجموعة الخصائص التى يتصف بها. كما أن القيم المرتبطة بهذه الخصائص Attribute Values (مثل : إسم العامل – عنوانه – تاريخ ميلاده) سوف تصبح هي مفردات البيانات التى يتم تسجيلها عنه بقاعدة البيانات.

ويلاحـظ أن مجموعة الخصائص أو السمات Attributes التى يتم تحديدها لأي كيـان من الكيانات ، يجب أن تتميـز بصفتيـن متلازمتيـن :

الأولى : **التماثل فى النوعية أو الطبيعة :**

فـالخواص أو السمات هى حقائق وصفية : Descriptive Properties ، يجب أن يحوزها كل عضو فى المجموعة.

على سبيل المثال :

- ⇦ فـإن كـل عميل فى مجموعة العملاء : Customer يمكن وصفه فى حدود : اسمه ، رقم حسابه ، محل الإقامة ، المدينة أو الحي السكنى.
- ⇦ وكل قرض فى مجموعة Loan يمكن وصفه فى حدود : رقم القرض ، وقيمته بالجنيه ، وتاريخ القرض.

هـذا يعـنى أن قاعدة البيانات يجب أن تقوم بجمع وتخزيـن نفس النوعية من الخصـائص ، وذلك فيما يتعلق بكل عضـو من أعضاء نفس مجموعة الكيان Entity Set.

ثانيـــاً : التبايـــن فى القـــيمة :

فكل خاصية من الخواص سوف تتفاوت -غالبا- فى قيمتها Attribute Value من عضو الى آخر فى المجموعة. فمثلا :

- أسماء العملاء فى مجموعة Customers لا تكون متماثلة غالبا.
- كمـــا أن أرقام جلوس الطلاب فى مجموعة Students يجب أن تكون مختلفة بالضرورة.
- كما تختلف غالبا قيمة القرض المقدم لكل عميل في مجموعة Loans.

ويلاحـــظ أيضا أن القيم المسموح بها لخاصية معينة تسمي اصطلاحيا باســـم : **النطاق Domain** ، أو **مجموعة القيم Value Set** لهذه الخاصية. فمثلا :

- يمكـــن تعـــريف نطاق خاصية "عدد ساعات العمل العادية" للعامل فى مجموعة Workers بأحد المصانع على أنه يشمل أي رقم من الأرقام الموجبة بحد أقصى ٤٠ ساعة فى الأسبوع.
- كما يمكن تعريف نطاق خاصية "أجر الساعة" على أنه يتضمن أي رقم مـــن الأرقام الموجبة يتراوح بين حدين : حد أدنى معين يمثل أقل أجر تحـــدده نقابة العاملين لفئة العمال التى ينتمي إليها العامل فى الساعة ، وحـــد أقصـــى معين يمثل أقصى أجر لساعة العمل تسمح به اللائحـــة الداخلية للمنشأة.

أنـــواع الخصائص
Types of Attributes

توجد عدة أنواع من الخصائص في نموذج الكيان- العلاقة ، من أهمها :

- الخصائص البسيطة والخصائص المركبة.
- الخصائص ذات القيمة الوحيدة وذات القيم المتعددة.

⇦ الخصائص المخزونة والخصائص المشتقة أو المستخلصة.

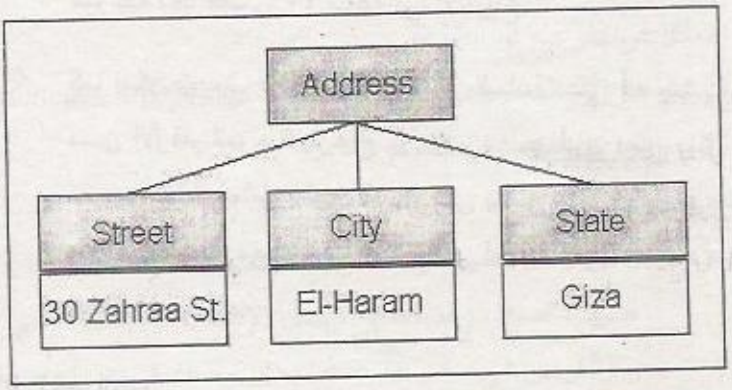
وفيما يلي شرحا موجزاً للمقصود بكل نوع :

❑ **الخصائص المركبة Composite Attributes:**

الخاصية المركبة هى التي يمكن تجزئتها إلى أجزاء أصغر Subparts ، يمثل كل جـزء مـنها فـي حد ذاته خاصية فرعية ذات معنى. فمثلا : إذا تضمن خاصية العنوان Address لعامل معين القيم التالية :

Address = 30 Zahraa St., El-Haram, Giza

فإننا نكون أمام خاصية مركبة ، حيث أن العنوان يمكن تجزئته إلى عدة خصائص منفصلة ، كما في الشكل التالي :



شكل رقم (٣-٤)

تجزئة الخاصية المركبة إلى عدة خصائص بسيطة

إن المـيزة الكـبرى للخاصية المركبة تتمثل في أنها تضم معا Concatenate قيم الخصائص البسيطة التي تندرج تحتها. وبالتالي فإنها تفيد في وصف المواقف التي

لا تتطلـــب الإشارة إلى أحد مكوناتها بذاته وإنما الإشارة إلى هذه المكونات كوحدة واحدة. والعكس بالعكس.

- **الخصائص البسيطة Simple Attributes :**

الخصـــائص البســـيطة هـــى التي لا يمكن تجزئتها Un-Divisible. ويطلق عليها اصطلاحيا اسم Attributes **Atomic**. ومن أمثلتها خاصية City وخاصية State في الشكل السابق.

- **الخصائص ذات القيمة الواحدة Single-Valued Attributes :**

إن غالبية الخصائص تنطوى علي وجود قيمة واحدة لكل كيان. ومثال ذلك أن كل طالـــب له عمر واحد ، وكل فاتورة لها تاريخ محدد ، وكل كتاب له عنوان مميز. هذا النوع من الخصائص يطلق عليه : وحيد القيمة **Singe-Valued**.

- **الخصائص ذات القيم المتعددة Multi-Valued Attributes :**

في بعض الأحوال ، تتضمن الخاصية الواحدة عدة قيم بالنسبة لنفس الكيان. ومثال ذلك خاصية لـــون Color السيارة: فبينما تكون معظم السيارات ذات لون واحد ، فـــإن هناك أيضا بعض السيارات متعددة الألوان ، مثل سيارات التاكسي في مدينة القاهرة (اللون الأبيض واللون الأسود).

ومثال آخر : هو خاصية **الدرجة الجامعية** College Degree للموظف ، حيث أن الأشخاص المختلفين يمكن أن يكون لديهم عدد مختلف من الدرجات الجامعية – أي قيم متعددة لنفس الخاصية :

- فقد يوجد شخص لا يحمل درجة جامعية.
- بينما يوجد شخص آخر يحمل درجة جامعية واحدة.

- ومن الممكن أيضا أن يوجد شخص ثالث يحمل درجتين جامعيتين أو أكثر. مثل هذا النوع من الخصائص هو الذى يطلق عليه اصطلاحيا اسم : متعدد القيم Multi-Valued.

- ❑ الخصائص المشتقة أو المستخلصة Derived Attributes
- ❑ والخصائص المخزونة أو المخزّنة Stored Attributes :

في بعض الأحوال ، قد تكون قيمة خاصة معينة مرتبطة بقيمة خاصية أخرى (أو عدة خصائص أخرى). ومثال ذلك خاصة العمر Age وخاصية تاريخ الميلاد Birthdate لشخص معين. إذ أن عمر هذا الشخص (قيمة خاصية Age) يمكن تحديده علي أساس الفرق بين تاريخ الميلاد وتاريخ اليوم الحالي.

وفي هذه الحالة :

- توصف خاصية Age بأنها : **خاصية مشتقة** Drived Attribute ، حيث أن قيمتها يتم اشتقاقها أو استخلاصها من خاصية تاريخ الميلاد Birthdate لهذا الشخص بالذات.
- وبالطبع لن يمكن معرفة عمر الشخص في هذه الحالة إلا إذا كان قد سبق تخزين تاريخ ميلاده. وبالتالي ، فإن خاصية تاريخ الميلاد Birthdate يطلق عليها هنا اسم : **خاصية مخزنة** Stored Attribute.

وتفيد هذه التفرقة في إرساء مبدأ هام في تصميم قواعد البيانات ، هو : أن مدخلات قاعدة البيانات لا يجوز أن تتضمن قيم الخصائص المشتقة ، إذ أن نظام إدارة قاعدة البيانات هو الذى يقوم – عند الحاجة – بحساب أو استخلاص تلك القيم من واقع القيم المخزنة المرتبطة بها.

من أمثلة القيم المشتقة أو المستخلصة (المحسوبة) التي تنطبق عليها هذه القاعدة ، أي لا يجوز إعتبارها مدخلات إلى القاعدة ، ما يلي :

- **ثمـــن بيع السلعة** ، حيث تستخلص قيمته من الخصائص المخزنة : الكمية المباعة Quantity ، سعر بيع الوحدة Unit-Price.
- **الأجـــر المســـتحق للعامل بالقطعة** ، حيث تستخلص قيمته من الخصائص المخـــزنة : عـــدد الوحدات المنتجة Production-Unites ، ومعدل الأجر للقطعة Wage-Rate.

المفهوم الثالث : **العلاقـــات Relationships**

المفهـــوم الثالـــث الذى يتأسس عليه نموذج "الكيان–العلاقة" هو مفهوم : **العلاقـــــة** Relationship. فى المعني الاصطلاحي ، تعتبر العلاقـــة همزة الوصل Association أو الصلة التى تربط بين كيانات متنوعـــة معا. أحد أنواع العلاقة هو : "العلاقة الثنائية" Binary Relationship ، وهى التى تربط بين كيانيـــن مختلفين.

على سبيل المثال :

- فإنه يمكننا تعريف علاقة ثنائية تربط بين "عميل " معين ، وبين "القرض" الخاص بهذا العميل.
- كما يمكننا تعريف علاقة ثنائية تربط بين "أمر توريد" محدد ، وبين "المورد" الذى قام بتوريد مشمول الطلبية الواردة فى هذا الأمر.
- ويمكن تعريف علاقة ثنائية تربط بين "طالب" محدد ، وبين "الشعبة" المقيد بها.

ويرتـــبط بمفهـــوم العلاقـــة مفهـــوم آخـــر هـــو : **مجموعـــة العلاقة Relationship Set أو حـــزمة العلاقة** . ويقصد به : سلسلة العلاقات التى تكون من نفس النوع.

على ســـبيل المثال :

⇦ إن حـــزمة العلاقة التى تربط بين مجموعة عملاء البنك وبين مجموعة القروض التى حصلوا عليها من هذا البنك يمكن تعريفها بأنها مجموعة علاقـــة ، تسمى : The Relationship Set : Borrows.

⇦ وبالمـــثل فإن حزمة العلاقات التى تربط أوامر التوريد الصادرة بالموردين الذين قـــاموا بتنفيذ هذه الطلبيات تعتبر مجموعة علاقة ، ويمكن أن يطلق عليها باسم : The Relationship Set : **Supplies**.

٣) الأسس الفلسفية لنموذج الكيان–العلاقـــة
Basic Philosophy of E-R Model

إن وظيفة نموذج الكيان– العلاقة هي العمل علي تمثيل الهيكل المنطقي للبيانات. هـــذا التمثـــيل يتأســـس علـــي مجموعة من الاعتبارات والأفكار النظرية المستمدة من – والمبنية علي – المفاهيم الثلاثة السابقة ... نوجزها فيما يلي :

- أن قـــاعدة البيانات تتكون من حشـــد متنوع Collections من عدة مجموعات كيانات **Entity Sets** قابلة للتمـــييز ، وأن كل مجموعة منها تشتمل على عدد مفتوح من الأعضاء من نفس النـــوع.
- أن تمييز كل مجموعة كيـــان عن غيرها من المجموعات في القاعدة إنما يتحقق عن طريق إســـناد أو تعيين مجموعة من الخصائص أو الســـمات **Attributes** لكـــل عضـــو فيها Entity ، على أن تكون هذه الخصائص متماثلة فى النوع بين كافـــة أعضـــاء نفس المجموعة ، لكن قيمها الفعلية **Values** هى التى يمكن أن تتفاوت – وغالبا ما تتفاوت– بين أعضاء المجموعة الواحدة ، لكن هذا التفاوت يجب أن يكون فى نطاق معين **Domain** ، مسموح به.
- أن الكـــيانات المختلفة لا تقف معـــزولة عن بعضها البعض ، لكنها تترابط معها بعلاقات متنوعة **Relationships Sets** .

٤) التعبيـــر البيـــاني عن النمـــوذج
The E-R Model Diagram

مـــن الناحية العملية ، يلجأ مصممو قواعد البيانات الى التعبير عن الهيكل الشامل لقاعـــدة البيانات محل الإهتمام فى صورة مخطط بياني Graph ، باستخدام أحد الأدوات المفيدة ، تسمى : **The E-R Diagram**.

هذا المخطط البياني يعتمد علي مجموعة من الرسومات- المتعارف عليها – للدلالة علي المفاهيم السابقة ، تتلخص فيما يلي :

- **المستطيلات** Rectangles : كل مستطيل منها يمثل مجموعة كيان Entity Set.
- **الأشـــكال البيضـــاوية** Ellipses : وتمثل الخواص أو السمات Attributes التى يتصف به كل عضو فى مجموعة الكيان.
- **المعيـــنات** Diamonds : وتمثـــل العلاقات Relationships القائمة بين أعضاء الكيانات المختلفة.
- **الخطوط** Lines : لربط الخصائص بالكيانات ، والكيانات بالعلاقات.

ويلاحظ أنه من الضرورى أن يتم تسمية كل عنصر من هذه العناصر بنفس اسم الكيان أو العلاقة أو الخاصية التى يمثلها.

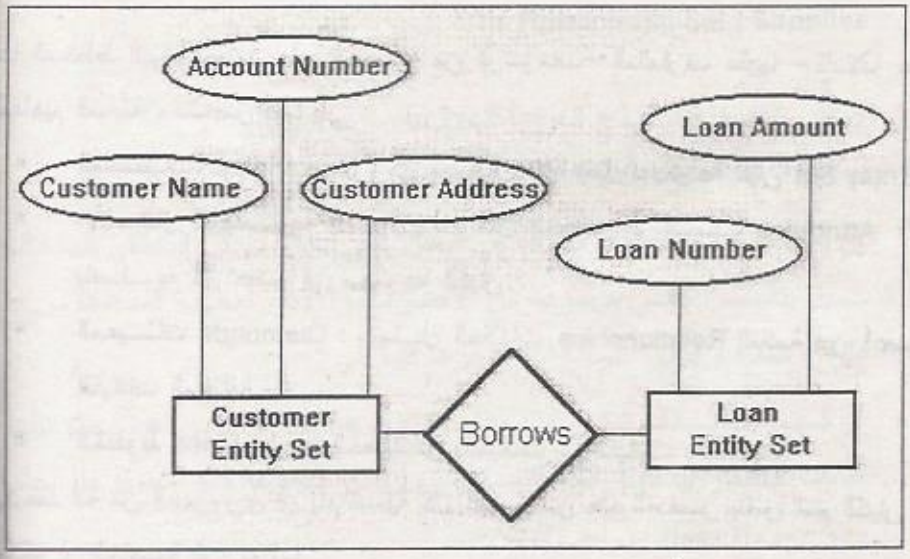
الشكل التالى ، شكل رقم (٣-٥) ، يعرض نموذجا لكيفيـــة استخدام هذه الأداة فى التعبـــير البيانـــي عن قطاع محدود من المنظور المنطقي الشامل لقاعدة البيانات فى أحد البنوك. هذا النموذج تبسيطي ، ويتضمن تعريف مجموعتين فقط من الخصائص التعريفية Attributes لمجموعتين مختلفتين من الكيانات Entity Sets التى يحتوى عليها عالم الواقع فى هذا البنك ، ويتضمنها بالضرورة الهيكل الشامل لقاعدة البيانات.

هاتان المجموعتان من الكيانات ، هـــما :

- مجموعة العمـــلاء : **Customer** Entity Set.

- مجموعة القروض : **Loan** : Entity Set.

كمـــا يتضمن هذا النموذج تعريف مجموعة علاقات **Relationship Set** واحدة ، وهى بالضـــرورة علاقـــات مـــن نفس النوع ، تربط بين أعضاء مجموعة العملاء ومجموعة القروض ، يطلق عليها اسم : **Borrows** : The Relationship Set.



شكل رقم (٣-٥)

التعبير البياني عن نموذج الكيانات والعلاقات E-R Diagram

وبطبـــيعة الحـــال ، فإنه من خلال هذه المجموعة العلاقات : Borrows ، يمكن ربط أى عمـــيل فـــى مجموعة العملاء : Customer ، بالقرض الذى حصل عليه ضمن مجموعة القـــروض Loan. ومن الجدير بالذكر – وإن لم يكن ظاهرا فى الشكل البياني السابق – أن نمـــوذج الكيان العلاقة يتضمن علاوة على ما تقدم مجموعة من **القيود Constraints** التى يجب أن تلتزم بها قاعدة البيانات.

من أهم هذه القيود ، ما يسمى اصطلاحيا باسم : **Mapping Cardinalities** ، أي عـــدد التـــبادلات البينيـــة الرئيسية أو الأساسية ، وهى التى تحدد عدد الأعضاء من كل كيان من الكيانات التى ترتبط فيما بينها بعلاقة محددة.

ومثال ذلك : أنه يمكن لكل عميل فى البنك :

- أن يأخذ قرضاً واحـــداً ،
- أو يأخذ أكثر من قرض ،
- أو لا يأخذ أي قرض علي الإطلاق.

* * *

٢ المبحث الثاني

المنظور ثلاثى المستويات

The Three-Levels Schema

سبق أن أوضحنا – عند مناقشة منهج قواعد البيانات – أن من أهم خصائص هذا المنهج : استقلالية البيانات عن التطبيقات ، وقابلية البيانات للمشاركة من جانب المستخدمين المختلفين ، قاموس البيانات Data Dictionary أو كتالوج النظام DBMS Catalog لحفظ منظور قاعدة البيانات Schema.

في هذا المبحث القصير ، سوف نلقى المزيد من الضوء علي موضوع : **منظور البيانات** Data Schema . وتشمل مناقشتنا :

- ❑ طبيعة وأهمية منظور البيانات.
- ❑ مستويات التجريد في المنظور الثلاثي.
- ❑ حالة إيضاحية عن المنظور ثلاثى المستويات.

علاوة علي ذلك ، فإن فهم **"منظور البيانات"** ، بمستوياته ودلالاته ومؤدياته ، سوف يساعدنا علي تفسير مجموعة إضافية من المفاهيم المرتبطة بقواعد البيانات ، تشمل بوجه خاص : حالة البيانات – الاستقلالية المادية – الإستقلالية المنطقية... الأمر الذى نتطرق اليه بالتفصيل في نهاية هذا المبحث.

أولا) طبيعة وأهمية منظور البيانات

يقصد **بمنظور البيانات** Data Schema : مجموعة التعليمات التي تصف **البنية الأساسية لقاعدة البيانات** Database Structure ... ذلك الوصف الذى يتضمن: أنواع البيانات المسموح بتخزينها Data Types في القاعدة ، والعلاقات القائمة بين هذه البيانات Relationships ، والقيود المفروضة على استرجاع وتحديث البيانات Constraints ، وذلك بالإضافة الى معلومات عن الطريقة التى يتم بها تخزين واسترجاع تلك البيانات.

في ضوء هذا المفهوم ، فإن منظور البيانات يتميز **بالخصائص** العامة التالية :

١) أن منظور البيانات هو **تعريف** قاعدة البيانات الجديدة لنظام إدارة قاعدة البيانات المستخدم ، أي أنه يمثل الوصف الفنى لبنية القاعدة A Database Discription.

٢) أن هذا الوصف يتم تحديده بدقة في مرحلة تصميم قاعدة البيانات ، ويكون من غير المتوقع تعديله – في المدى القريب علي الأقل.

٣) أن منظور البيانات يتم كتابته علي شكل **تعليمات** باستخدام لغة خاصة بنظام إدارة قاعدة البيانات المستخدم ،هى : لغة تعريف البيانات DDL.

٤) هذه التعليمات يتم ترجمتها Compiled وحفظها – كبيانات وصفية Meta-Data – في كتالوج النظام DBMS Catalog أو قاموس البيانات Data Dictionary.

٥) أن نظام إدارة قاعدة البيانات سوف يرجع إلى هذا المنظور عندما يحتاج إلى ذلك ، وبخاصة قبل تنفيذ أي محاولة من أحد المستخدمين لتخزين أو إسترجاع أو معالجة بعض البيانات المخزونة في القاعدة.

ونخلص من ذلك إلى أن تعريف المنظور الصحيح لنظام إدارة قاعدة البيانات هو مسألة في غاية الأهمية ، ويجب أن يتم تصميم ووصف هذا المنظور بعناية فائقة.

ثانيا) إطار المنظور الثلاثى لبنية القاعدة

The Three-Levels Schema Architecture

يقوم إطار المنظور الثلاثي المستويات علي تعريف ووصف بنيـــة قاعدة البيانات علـــى ثلاثة مستويات مختلفة من تجريد البيانات Data Abstraction Levels ، تتمثل – بالترتيب – فيما يلي :

- **المنظور الخارجي** External Schema.
- **المنظور المنطقي** Conceptual or Logical Schema.
- **المنظور الداخلي** أو المادي Physical or Internal Schema.

هـــذه المســـتويات الثلاثـــة تهتم بوصف **نفس هيكل البيانات** ، ولكن من وجهات نظر مختلفة تتراوح ما بين نهايتين : هما :

- في المستوى الأعلى : الخارجي ... التركيز على طريقة فهم المستخدم للبيانات.
- وفـــي المستوى الأدنى : الداخلي ... التركيز على تفاصيل كيفية تخزين وإسترجاع البيانات على وسائط التخزين.

---

(١) في دراسة تفصيلية مبكرة لهذا الإطار الثلاثي ، ارجع بوجه خاص الى :

- D. Tsichritzis, & A. Klug, Ed., ***The ANSI/x3/SPARC DBMS Framework***, (New York: AFIPS Press, 1978).

وللتوسع في دراسة هذا الإطار ، ارجع بوجه خاص الى المراجع الآتية :

- C. J. Date, ***An Introduction to Database Systems***, 6th. ed., (Readings, Massachusetts: Addison-Wesley Publishing Co., 1995), pp. 28-51, &
- D. R. Howe, ***Data Analysis for Data Base Design***, 2nd. ed., (London: Edward Arnold, A Division of Hodder & Stoughton, 1989), pp. 24-40.

❑ مستويات التجريد في المنظور الثلاثى :

فيما يلي نقدم شرحا موجزا لكل مستوى :

١) المنظـــور الداخلي أو المادي للبيانات
The Internal Schema

هـــذا المـــنظور يمـــثل أدنى مستويات التجريد ، ويسميه البعض باسم : المنظور المـــادي The Physical Schema. وهو وثيق الصلة بالتخزين المادي للبيانات ، إذ أنه يصـــف **كيف** How يتم تخزين البيانات بالفعـــل على وسيط التخزين المستخدم ، ويصف التفاصـــيل الدقـــيقة لتخزين البيانات ومسارات الوصول إليها. فبيانات الاســـم والمرتب الخاصـــة بـــأحد العاملين مثلا يتم وصفها كقطاعات Blocks لأماكن تخزين متعاقبة على وسائط التخزين.

٢) المنظـــور المنطقي للبيـــانات
The Logical Schema

المســـتوى الأعلـــى التالى من مستويات التجريد يتمثل فى المنظور المنطقي أو النظري The Logical or Conceptual Schema. إنه المنظور الذى يركز على وصف مـــا هي What البيانات التى يتم تخزينها بالفعل بداخل قاعدة البيانات ، وما هى العلاقات Relationships الـــتى تربط بين هذه البيانات. وفي الحقيقة ... إن هذا المنظور هو الذى يخـــتص بوصـــف **الهـــيكل الشامل أو المجمَّع** لقاعدة البيانات الذى يخدم احتياجات كافة المستخدمين كوحدة. وعند هذا المستوى بالتحديد :

- يتم إستبعاد كافة صور التكرار غير الضروى في البيانات Data Duplication . أي تطبـــيق مـــبدأ : أن البيان الواحد يسجل مرة واحدة فقط في كل أنحاء قاعدة البيانات ، ما تعددت استخداماته في تلبية احتياجات المستخدمين.
- كما يتم حجب التفاصيل الفنية المعقدة لهياكل التخزين المادية للبيانات ، والتركيز فقـــط على وصف أنواع البيانات ، والعلاقات القائمة بين البيانات ، بالإضافة الى

بـيان العمليات التى يجريها المستخدمون على تلك البيانات User Operations ، والقيود المفروضة على تنفيـذ تلك العمليات Constraints.

٣) المنظور الظاهـري أو الخارجي للبيانات
The External Schema

يمـثل المنظور الخارجي The External Schema أعلى مستويات التجريد فى وصـف بنـية نظـام قاعدة البيانات. ويطلق عليه البعض : مستوى التشكيلات الفردية للبيانات Individual User Views . وفيه يتم وصف تشكيلة البيانات View التى تناسـب كل مستخدم على حدة من المستخدمين المختلفين لقاعدة البيانات. فكل مستخدم يرغب فقط فى الوصول الى تشكيلة البيانات Data View التى تلـبي احتياجاته ، أي الى جزء فقط من مضمون قاعدة البيانات هو الذى يمثل رؤيته لها.

❑ التعبير البيانى عن المنظور الثلاثي :

يوضـح الشـكل الـتالى ، رقم (٣-٦) ، الإطار العام للمنظور الثلاثي لبيئة قاعدة البيانات. ومنه تتضح لنا مجموعة من الاعتبارات الهامة ، نوجزها فيما ما يلي :

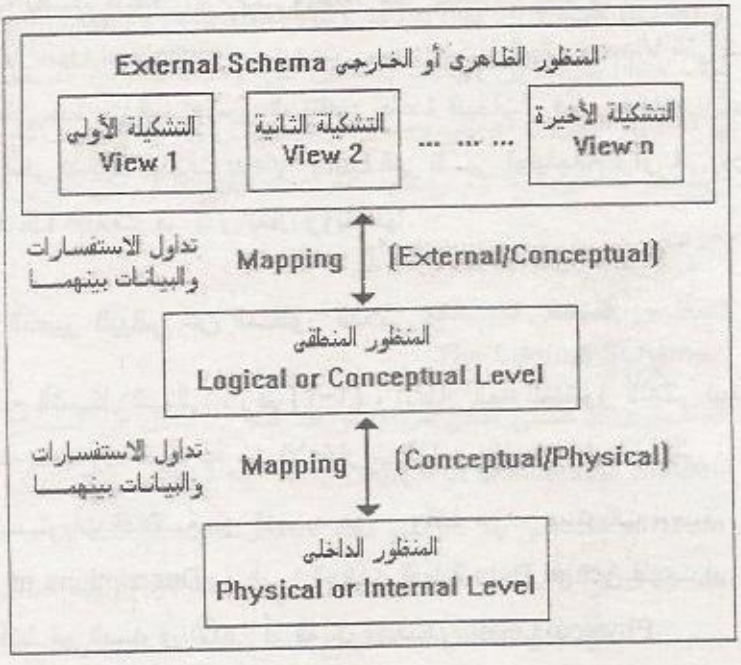
- أن المسـتويات الثلاثـة من التجريد هى – وفقط هي – مستويات لوصف البيانات Descriptions of Data ... أمـا البيانات الفعلية Actual Data فتوجد فى مستوى واحد فقط هو المستوى الأدنى أو المستوى المادي Physical Level.

- ان كل نظام لقاعدة البيانات يجب أن يتضمن : منظورا ماديا واحدا فقط ، ومنظورا منطقـيا واحـدا فقط ، بينما يمكن – بل ويجب بالضرورة – أن يتضمن العديد من أنماط المنظور الخارجي (مستوى تشكيلات البيانات المطلوبة للمستخدمين).

- أن هـناك مفهومـا آخر يرتبط بمستويات التجريد الثلاثة ، هو ما يطلق عليه اسم **Mapping** ، أي الـتداول البينىّ أو التناظرى. والمقصود بالتداول البينـي هو :

عمليات التحويل Transformation التى يجريها النظام على الاستفسارات والنتائج (أي البيانات الناتجة عن الاستفسار) ، فيما بين مستويات التجريد الثلاثة لبنية القاعدة ، أي :

- بين المستوى المادي والمستوى المنطقي ، من جهة.
- وبين المستوى المنطقي ومستوى التشكيلات ، من جهة أخرى.



شكل رقم (٣-٦)

نموذج المنظور الثلاثي لهيكل قاعدة البيانات

❑ فوائد المنظور الثلاثي :

المنظور الثلاثى المستويات لوصف بنية قاعدة البيانات يخدم في الواقع عدة أغراض ، نذكر منها ما يلي :

١) توفير عدة مستويات لتعريف البيانات ، تختلف عن بعضها البعض فى درجة التعقيدات والتفاصيل الفنية التى تحتوى عليها ، وبالتالى سوف يجد كل مستخدم الوصف الذي يتناسب مع درجة خبرته ودرايته بالنظام.

٢) توفير إطار فكرى متكامل Theoretical Framework ، يصلح كأساس لشرح هياكل الكثير من الأنواع المختلفة لنظم قواعد البيانات ، وبخاصة تلك القائمة على أساس النموذج العلاقى Relational System.

٣) أنه يقدم المنهج الأمثل لتصميم وتحليل أي نظام من نظم إدارة قواعد البيانات DBMS ، والتعرف على ميكانيكيات العمل التي يطبقها وعلي التفاعلات القائمة بين العناصر الرئيسية لهذا النظام ، ومن ثم فإنه يوفر إمكانية الحكم الموضوعي على مدى كفاءة أداء هذا النظام.

ثالثا) حالة إيضاحية عن المنظور الثلاثي

An Example of the Three-Levels Schem

عند هذه النقطة ، يكون من المناسب أن نقدم مثالا توضيحيا ، يساعد على توضيح كيفية تطبيق المستويات والمفاهيم السابقة على نظام قاعدة بيانات محدودة لشئون العاملين. يشمل هيكل النظام ثلاثة برامج تطبيقية (1, 2, 3) ، وقاعدة بيانات.

(١) فى المستوى الخارجي External Level :

يوجد منظور خارجي لكل تطبيق على حدة ، يتضمن كل منظور منها وصفا مبسطا لتشكيلة محددة من البيانات المخزّنة فى القاعدة ، أي وصفا لمجموعة من قطع البيانات Data Items المخزّنة ، والتى تناسب احتياجات هذا المستخدم بالذات. فمثلا :

- التطبيق أو البرنامج الأول **Program-1** ، الذى يعرض بعض بيانات العامل ، سوف يقابله المنظور الخارجي External-Schema A .

- والـــبرنامج الثانـــي **Program-2** ، الذى يفترض أنه يعرض بيانات عن مرتب العامل ، يقابله المنظور الخارجي External-Schema B .. وهـــكذا

ولغـــرض تمثـــيل البيانات المطلوبة فى المستوى الخارجي ، سوف يتضمن كل مـــنظور وصفا لتشكيلة البيانات الملائمة لاحتياجات المستخدم ، يشمل بوجه خاص : قطع البيانات Data Items المطلوبة ، وشكل كل بيان Format of Data Item ، والتسلسل أو الترتيب الذى سوف تظهر به هذه البيانات للمستخدم. فمثلا :

- المنظور الأول **External-Schema A** ، سوف يتضمن المعلومات الآتية :
  - Employee-No : نوعه نصي Text ، ويتكون من ٦ حروف.
  - Employee-Name: نوعه نصي ، ويتكون من ٣٠ حرف.
  - Employee-Address : نوعه نصي ، ويتكون من ٣٠ حرف.
- والمنظور الثاني **External-Schema B** ، سوف يتضمن المعلومات الآتية :
  - Employee-No : نوعه نصي Text ، ويتكون من ٦ حروف.
  - Employee-Department : نوعه نصي ، ويتكون من ٤ حروف.
  - Employee-Salary: نوعه رقمي/عشري ، ويتكون من ٧ حرف.

وهـــكذا...

(٢) فى المستوى النظري أو المنطقيّ Conceptual Level :

عند هذا المستوى ، يوجد منظور منطقي واحد للبيانات Conceptual Schema يشـــمل **كافـــة** البـــيانات اللازمة للتطبيقات الثلاثة مجتمعة. فكل قاعدة بيانات يكون لها بالضرورة منظور منطقي واحد فقط.

إن تمثـــيل البـــيانات فى ظل هذا المنظور ، سوف يتضمن وصفا لجميع مفردات البـــيانات المطلوب تخزينها فى قاعدة البيانات من أجل تلبية احتياجات **كافة** التطبيقات

المستخدمين المختلفين ، مع تحديد هيئة كل بيان Format ، بالإضافة الى أسماء الملفات التي سوف تحتوى علي مفردات البيانات المذكورة ، والعلاقات التى تربط بين هذه البيانات.

كما يشمل المنظور المنطقي أيضا القيود والاشتراطات Constraints التى يلزم مراعاتها عند تنفيذ عمليات تخزين واسترجاع ومعالجة هذه البيانات ... ومنها على سبيل المثال :

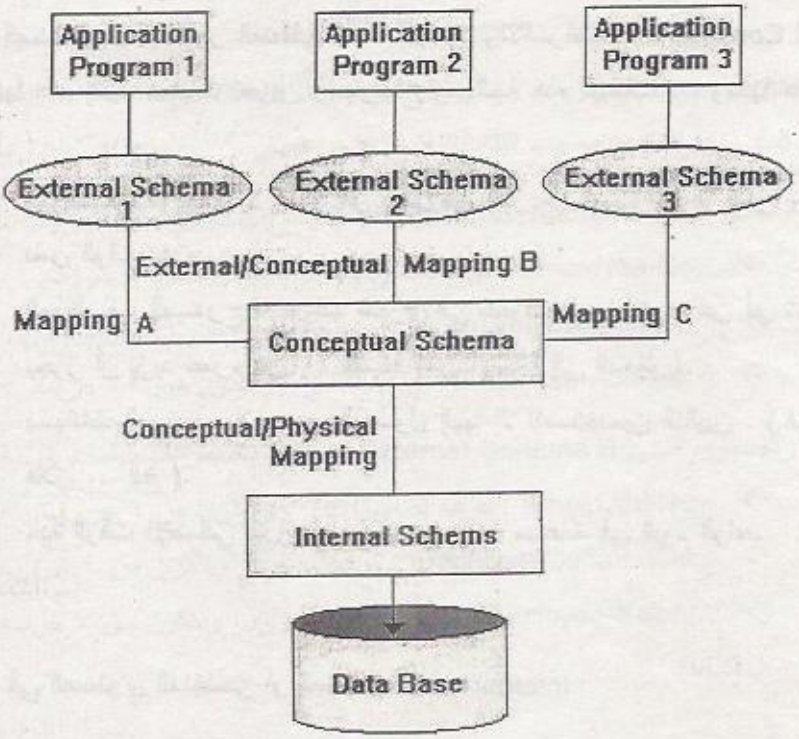
- الموظف الواحد لا يجوز أي يعمل فى أكثر من قسم واحد أو إدارة واحدة فى نفس الوقت.
- الموظف المدرج له مرتب شهري فى كشوف المرتبات – عن أي شهر – لا يجوز أن يزيد عمره عن ٦٠ سنة (سن الإحالة إلى المعاش).
- بيانات المرتب لا يسمح بالوصول إليها إلا للمستخدمين التاليين : (فلان ...و فلان ... الخ).
- مدة الوقت الإضافي لكل عامل لا تزيد عن ٨ ساعات فى اليوم الواحد.

وهكذا...

(٣) فى المستوى الداخلي أو المادى Internal Level :

فى بنية قاعدة البيانات الحالية ، كما فى أي قاعدة بيانات أخرى ، سوف يتواجد بالضرورة منظور مادي واحد للبيانات Physical Schema ، يشمل وصف كيفية تخزين واسترجاع البيانات المطلوب فى داخل وسائط التخزين. إن هذا الوصف سوف يركز على تحديد تفاصيل فنية دقيقة مثل : السجلات Records – حقول البيانات Fields – حجم وحدات تخزين البيانات Block-sizes معبرا عنها بالبايت Byte ، شكل الفهارس Indexes المستخدمة لغرض إعادة ترتيب وفرز السجلات ، وسيط التخزين المستخدم Storage Media فى حفظ البيانات... وغير ذلك تفاصيل طريقة تخزين البيانات.

الشكل التالى يعرض مخطط بياني يوضح كيف يتم التعـبير عن هيكل نظام قاعدة البيانات المفترض فى المثال السابق ، باستخدام المنظور الثلاثي المستويات :



شكل رقم (٣-٧)

المنظور الثلاثي لهيكل نظام قاعدة البيانات

لاحـظ كيف تندمج كافة تشكيلات بيانات المستخدمين على المستوى الخارجي فى مـنظور منطقـي واحد يصف النسيج الموحد والمتكامل للبيانات المطلوب تخزينها داخل القـاعدة والتى تخدم كافة الاحتياجات ... وكيف أن هذا المنظور المنطقي ينعكس بدوره فى منظور مادي واحد ، يركز على تفاصيل طريقة تخزين واسترجاع هذه البيانات.

(ثانيا) مفاهيم إضافية مرتبطة بالموضوع

More Related Concepts

في ضـوء مـا تقدم ، يمكننا الآن أن نتعرف علي المعنى الإصطلاحي لبعض المفاهيم الإضـافية المرتبطة بالموضوع ، والتى تعتبر في غاية الأهمية بالنسبة لدراسة نظم قواعد البيانات. هذه المفاهيم هى :

- ⇦ حالة البيانات ، والفرق بينها وبين منظور البيانات.
- ⇦ الاستقلالية المادية ، والاستقلالية المنطقية للبيانات.

وفيما يلي شرح موجز لهذه المفاهيم :

(١) حالـة البيـانات Data Instance
ومنظور البيانات Data Schema

هناك فرق بين كل من : وصف البيانات Data Description المسموح بتخزينها في قاعدة البيانات ، وبين حالة البيانات المخزّنة The Database Itself داخل القاعدة.

فعـندما نقـوم بتعريف قاعدة البيانات الجديدة لنظام إدارة قاعدة البيانات ، فإننا نقوم فقـط بوصـف مـنظور هذه القاعدة ، وحفظ هذا الوصف فى الكتالوج أو قاموس البيانات. وعند هذه النقطة تكون القاعدة خالية من أي بيانات فعلية Empty State.

وعـندما نبدأ في إدخال البيانات إلى القاعدة لأول مرة ، فإن هذه العملية تنتهي بحالـة للبيانات تسمي "**الوضع المبدئى للبيانات**" Initial State. ومنذ هذه اللحظة ، فإن إجراء أي عملية علي البيانات المخزونة في القاعدة – مثل إضافة أو حذف أحد السجلات أو تحديـث بعـض البـيانات – سوف يترتب عليه في نهايتها الحصول علي حالة جديدة لقاعدة البيانات Another Database State.

هـــذا التميـيز قد ترتب عليه ظهور مفهومين مختلفين للبيانات ، هما : مفهوم **حالـــة البيانات** Data Instance ، ومفهوم **منظور البيانات** Data Schema. وفيما يلي نوضح معنى كل مفهوم ، وخصائصه.

ونخلص من ذلك إلى أن : البيانات الفعلية المخزّنة فى قاعدة البيانات تتغير حالتها – بيـن لحظة وأخرى – مع حدوث كل عملية جديدة من عمليات تحديث البيانات. ويطلق اصطلاح **حالة البيانات** Database Instance أو اصطلاح **الوضع الراهن للبيانات** Database State على المضمون الفعلي للبيانات المخزّنة بالقاعـدة في لحظة معينة بالذات ، تعتبر بالضرورة لحظة سكون افتراضـيّة.

هذا يعنى أن حالة أو وضع البيانات المخزونة في القاعدة يكون فى تغيـرٍ مستمر نتيجة لتفاعل المستخدمين وتحديثهم المستمر لتلك البيانات. كما أن حالة قاعدة البيانات قبل أي عملية تحديث (مثل تسجيل فاتورة بيع) سوف تختلف عن حالة قاعدة البيانات بعد إتمام تنفيذ هذه العملية.

والنقطة ذات الأهمية هنا ، هي أن نظام إدارة قاعدة البيانات يكون مسئولا – جزئيا – عـن ضمان أن حالة البيانات بعد أي عملية تكون حالة صحيحة Valid State ، وذلك بمعـنى أنهـا حالة تتوافق مع الهيكل وتخضع للقيود السابق تحديدها في منظور البيانات. هذا يؤكد ، مرة أخرى ، الأهمية الفائقة للتحديد الدقيق والصحيح لهذا المنظور منذ البداية.

(٢) الاستقلالية المنطقية Logical Data Independance والاستقلالية المادية Physical Data Independandce

سـبق أن أشـرنا – فى الفصل الأول من هذا الباب – الى مفهوم : اسـتقلالية البيانات عن التطبيقات : Application/Data Independence . وأوضحنا أنه يعتبر من أهم الخصائص المميزة لمنهج قواعد البيانات. والآن ، وفى ضوء فهمنا للمنظور الثلاثي

لبنية قاعدة البيانات ، يمكننا أن نعيد صياغة هذا المفهوم الاصطلاحي بالشكل الذى يجعله قابلا للتطبيق العملي.

إذ يمكـــن لنا القول بأن المقصود **باستقلالية البيانات** Data Independence هو: القـــدرة علـــى تعديل تعريف منظور البيانات فى أحد المستويات الثلاثة للتجريد ، دون أن يتأثر تعريف منظور البيانات فى المستوى الأعلى التالى له مباشرة.

وتبعا لذلك ، فإنه يمكن التمييز بين مستويين مختلفين من استقلالية البيانات :

- **الاســـتقلالية الماديـــة للبيانات Physical Data Independence :**

يقصد بها : القدرة على تعديل **المنظور المادي** Physical Schema لبنيـــة قاعدة البـــيانات ، دون أن يتطلب ذلك إجراء تعديلات تابعة فى المنظور المنطقي ، أو فى المنظور الخارجي لتلك القاعدة. يلاحظ أن التعديلات على المستوى المادي تكون قليلة عموما ، وتستهدف غالبا تحسين الأداء.

- **الاســـتقلالية المنطقـــيّة البيانـــات Physical Data Independence**

أما الاستقلالية المنطقية للبيانات فيقصد بها : القدرة على تعديل **المنظور المنطقي** لبنية قاعدة البيانات Logical Schema ، دون أن يتسبب ذلك فى تغيير المنظور الخارجي للبيانات ، أو فى إعادة كتابة أو تعديل برامج التطبيقات الموجودة. مثل هذه التعديلات تكون ضرورية عند الرغبة فى توسيع نطاق قاعدة البيانات ، كما فى حالة إضافة نوع جديد من السجلات Record Type أو من مفردات البيانات Data Items.

فى الواقع : إن تحقيق الاستقلالية **المنطقية** للبيانات يكون أصعب من تحقيق استقلاليتها المادية ويرجع ذلك الى أن برامج التطبيقات تعتمد بشـــدة على الهيكل المنطقي للبيانات التى تتعامل معها.

الفصل الرابع

نموذج قـاعدة البيانـات العلاقيـة

The Relational Database Model

المبحث الأول : إطـار نمـوذج قـاعدة البـيانات العلاقيـة.

المبحث الثانـي : جـداول البيانـات.

المبحث الثالـث : حقـول المفاتيـح.

المبحث الرابـع : علاقـات الإرتبـاط.

المبحث الخامس : القيـود والتحفـظات.

مقدمة :

لقـد نجح النموذج العـلاقي Relatonal Database Model في أن يعمـّد نفسه علـي القمـــة ، بإعتباره يمثل في الوقت الحاضر النموذج الأساسى لتمثيل البـيانات ، والإطـار المنطقـي لتصميم تطبيقات قواعد البيانات في مختلف المجالات ، وبخاصة في المجالات التجارية.

إن هدفـنا الأساســي مـن دراســة هذا النموذج ليس هو الإغراق في تفاصيل النظريات والأسـاليب والتقنـيات المرتبطة بها ، وإنما هو التركيز علي ما يكفى منها لجعل القارئ يكتسـب المهارات الأساسية لتحليل وتصميم قواعد البيانات في المجالات التجارية ، مثل نظـم مراقـبة المخـازن – تحليل المصروفات – تسجيل وتحليل طلبيات البيع – تسجيل ومتابعة الأصول الثابتة ... الخ.

مـن أجـل تحقـيق هذا الهدف ، تنقسـم المادة العلمية الواردة إلى خمسة مباحث مـترابطة : المبحـث الأول منها يلقى المزيد من الضوء علي الملامح العامة لنظم قواعد البـيانات العلاقـية ، أما المباحث الأربعة الأخرى فتركز علي مناقشة العناصر الأساسية التي يقوم عليها نموذج قاعدة البيانات العلاقية ، وهى : جداول البيانات Data Tables– وحقـول المفاتيحKey Fields – وعلاقات الإرتباط Relationships – ثم أخيرا القيود والتحفظات Constraints.

هذه المباحث هي : (١)

المبحث الأول : **إطار نموذج قاعدة البيانات العلاقية.**

المبحث الثاني : **جـــداول البيانـــات Data Tables.**

المبحث الثالث : **حقـــول المفاتيح Key Fields.**

المبحث الرابع : **علاقات الارتباط Relationships.**

المبحث الخامس : **القيـــود والتحفظات Constraints.**

وننـــبه مـــنذ البداية إلى أن قاعدة البيانات التى لا تهتم بتعريف ، وفرض الالتزام بتطبـــيق ، **القيود والتحفظات** ، إنما تعتبر قاعدة هشـــة في تماسكها ، وغير صحيحة في بنـــيانها. ونقول ذلك لأن هذا العنصر بالذات لا يلقى الإهتمام الواجب من جانب بعض مصـــممى تطبيقات قواعد البيانات. إن قاعدة البيانات من هذا النوع سرعان ما تعانى من وجود علاقات متفسخة أو مبتورة ، ومن عجزها عن منع المستخدمين (عن جهل أو سوء نيـــه) من انتهاك وإهدار تكامل وتوافـــق البيانات المخزونة بها... ومن ثم فإنها تصبح ، إن عاجلاً وإن آجلاً، غير صالحة للاستخدام فيما وُضعت من أجلها. **فالقيود والتحفظات** هـــي عنصر لا يقل في أهميته عن العناصر الثلاثة الأخرى للقاعدة العلاقية : الجداول ، حقول المفتاح ، علاقات الارتباط.

* * *

(١) يعود النموذج العلاقي إلي دراسة بحثية قام بها Codd في الستينيات :

- E. F. Codd, "A Relatiional Model for Large Shared Data Banks", ***Communications of the CAM,*** (Volume 13, Number 6, June 1970), pp. 377-387.

١ المبحث الأول

إطار نموذج قاعدة البيانات العلاقية

The Framework of Relational DB

لقد أوضحنا في الفصل السابق أن النموذج العلاقي ينتمي إلى مجموعة **النماذج التنفيذية** لتمثيل البيانات، وهي نماذج تتميز بتقديم مفاهيم سهلة لوصف هيكل البيانات دون أن تعزل نفسها تماما عن كيفية تخزين هذه البيانات. كما أوضحنا نواحي الشبه ونواحي الإختلاف بينه وبين النموذج الشبكي والنموذج الهرمي.

في هذا المبحث ، سوف نركز على شرح بعض الملامح الرئيسية للإطار العام للنموذج العلاقي : Relational Database Model. هذه الملامح تتضمن :

- ❑ تعريف قاعدة البيانات العلاقية.
- ❑ الأنظمة التجارية للقاعدة العلاقية.
- ❑ مثال تطبيقي لقاعدة بيانات علاقية.

---

(١) في دراسة وافية للنموذج العلاقي ، راجع بوجه الخاص :

- P. Atzeni, & De Antonellis, ***Relational Database Theory,*** (Menlo Park, California : The Benjamin/Cummings Publishing Co., 1993).

أولا) تعريف قاعدة البيانات العلاقية

ويمكن تعريف قاعدة البيانات العلاقية بأنها :

مجموعة من **جداول البيانات** Data Tables ، المرتبطة معــا **بعلاقات ارتباط** Relationships ، مبنية علي حقول عامة أو مشتركة تسمي **حقول المفاتيح** Key Fields ، وذلك تحت سيطرة مجموعة من **القيود** و**التحفظات** Constraints التي تستهدف المحافظة علي صحة وتجانس وتكامل وسرية البيانات.

ويتضح من هذا التعريف أن العناصر الرئيسية التى تقوم عليها قاعدة البيانات العلاقية ، أربعة عناصر ، هى :

- جداول البيانات Data Tables.
- حقول المفاتيح Key Fields.
- علاقات الإرتباط التي تقوم بين محتويات هذه الجداول Relationships.
- القيود والتحفظات Constraints.

ومــن رأينا أن السر الكامن وراء تسمية هذا النموذج : **بالنموذج العلاقي** هو أن كافــة هذه العناصر تنبثق جميعها من مفهوم **العلاقة** Relationship :

- فالبــيانات الــواردة عــن أحد سجلات جدول البيانات ، هي مجرد حقائق أو قيم Values لوصــف خصائص معينة لإحدى مفــردات الكيان أو الموضوع Entity الذي تمثله قاعدة البيانات ، (كأحد أذون الصرف أو أحد الموردين... الخ). ومن ثــم فإن هذا القطع من البيانات مــا هي إلا مجموعة من القيم Values المرتبطة معــا بعلاقــة.

- وبالتالي فإن **جداول البيانات** ما هى إلا مجموعات من العلاقات التي تصف خصائص الكيانات التي تمثلها.
- كما أن القيم الموجودة فى **حقل المفتاح** فى أحد الجداول ، تعتبر هى همزة الوصل التي تربط العلاقات الممثلة في هذا الجدول بالعلاقات الممثلة في جدول آخر أو أكثر.
- وفى النهاية فإن كافة الجداول ترتبط ببعضها البعض فى شبكة من العلاقات لتكون النسيج الشامل لقاعدة البيانات.

أن العلاقات هي المفهوم السحري الذى يستخدم لوصف كل عناصر هيكل قاعدة البيانات العلاقية. وبهذا المنطق ، فإن النموذج العلاقى يكون بمقدوره أن يجمع Collect وينظم Organize ويسترجع Retrieve بيانات قد تبدو متنوعة فى طبيعتها ، لكنها تكون فى النهاية مرتبطة ببعضها بطريقة أو بأخرى.

ثانيا) الأنظمة التجارية للقواعد العلاقية

إن سوق البرمجيات التجارية الجاهزة حافل الآن بالكثير من برامج ونظم إدارة قواعد البيانات العلاقية Relational-Database Products :

- **فعلى مستوى قواعد البيانات الضخمة** : تشمل هذه البرامج بوجه خاص :
  - IBM's DB2.
  - Ingers.
  - Oracle.
  - Sybase.

- Informix.
- MicroSoft SQL Server ...

➢ **وعلى مستوى قواعد البيانات الصغيرة** التي تعمل مع نظم الحاسبات الشخصية : فإن البرامج الأكثر عمومية فى وانتشارا فى الوقت الحاضر ، تشمل :

- Ashton-Tate dBase III Plus, IV .
- Borland Visual dBase.
- MicroSoft Visual FoxPro.
- MicroSoft Access XP ....

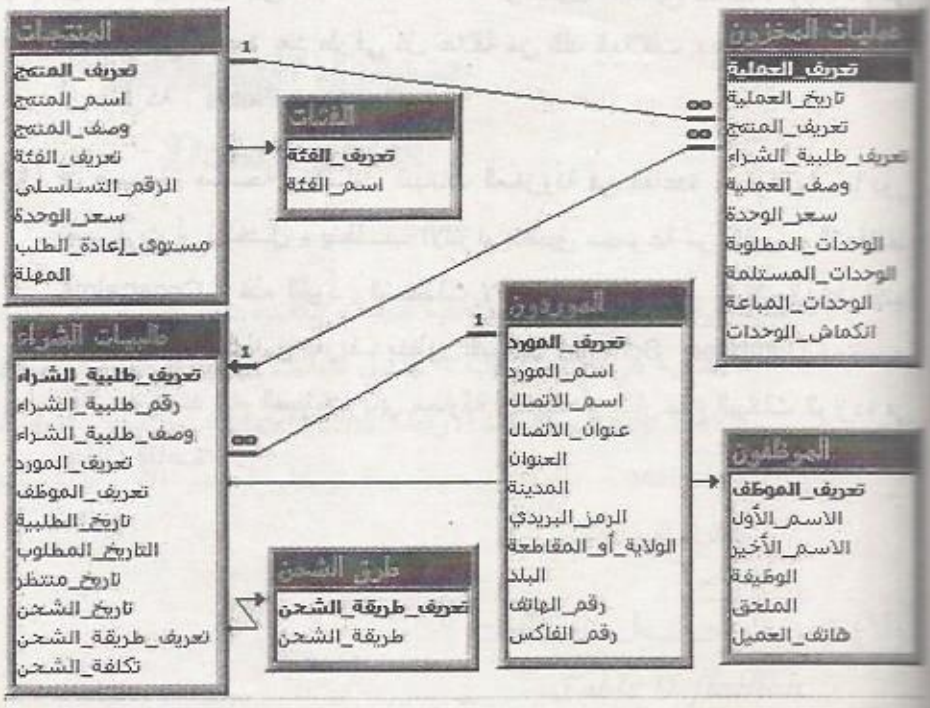
ثالثا) نموذج تطبيقي لقاعدة بيانات علاقية

والآن ، وقبل أن نغوص في التفاصيل الفنية المرتبطة بالموضوع ، فإن الشكل التالي يعرض المخطط البياني لجانب من هيكل قاعدة بيانات علاقية بإحدى شركات تسويق المنتجات الجاهزة ، تم تصميمها باستخدام نظام Access. هذه القاعدة خاصة بنظام مراقبة المخزون السلعى : Inventory Database.

بتأمل هذا الشكل ، يتضح لنا كيفية تطبيق فلسفة النموذج العلاقي في تصميم قواعد البيانات ... وآية ذلك ما يلي :

(١) إن قطع البيانات التي يتضمنها هيكل القاعدة يتم حفظها في ٧ **جداول بيانات** Tables ، يختص كل منها بتمثيل كيان أو موضوع An Entity مستقل ومنفصل وقابل التمييز. هذه الموضوعات أو الكيانات هي : المنتجات – فئات المنتجات – طلبيات الشراء أو أوامر التوريد – طرق الشحن – الموردون – الموظفون

المختصون (مندوبو المشتريات) – عمليات المخزون (الوارد والمنصرف والفاقد ...الخ). إن بعض هذه الكيانات يكون له وجود مادي ملموس مثل : المنتجات والموردون ، بينما بعضها الآخر له وجود اعتباري وغير ملموس مثل : طرق الشحن ، وحركة المخزون.



شكل رقم (٤-١)

مخطط بياني لهيكل قاعدة بيانات علاقية

قاعدة مراقبة المخزون السلعي

(٢) أن هـناك **علاقـات ارتـباط** Relationships تربط بين بيانات هذه الجداول. وبعـبارة أكثر دقـة ، فإن هناك عدة أنواع من علاقات الارتباط (علاقة واحد-بواحد ، وعلاقة واحد-بكثير) ، تعمل علي ربط سجلات كل جدول منها بسجلات جدول آخر في ذات الوقت.

(٣) أن الأساس الذى تبنى عليه هذه العلاقات هو **حقول المفاتيح** Key Fields ، وهى الحقول التي تتواجد عند طرفي كل علاقة من تلك العلاقات ، وتعتبر حقول عامة أو مشتركة : Common Fields.

(٤) أن ضـمان صـحة وتكـامل البيانات المخزونة فى القاعدة ، وتوافقها معاً دون تضـارب أو تداخـل ، يتطلـب الالتزام بتطبيق مجموعة من **القيود والتحفظات** Constraints. هذه القيود والتحفظات لا تظهر صراحة فى الشكل السابق لكنها تعتـبر جزءا هاما من تعريف منظور البيانات Database Schema ، وتسرى تلقائـيا عـند قيام المستخدم بأي محاولة لتحديث أو استرجاع البيانات الواردة فى جداول القاعدة .

* * *

٢ المبحث الثاني

جـداول البيـانات
Data Tables

أوضـحنا في المبحث السابق أن قاعدة البيانات العلاقية تقوم على أربعة عناصر أساسـية هي على الترتيب : جداول البيانات Tables ، وحقول المفاتيح Key Fields ، وعلاقات الارتباط Relationships ، والقيود والتحفظات Constraints. هـذا المبحـث يركـز على العنصر الأول منها ، وهو **جداول البيانات**.

أولا) تعريف جـدول البيـانات
Table Definition

جداول البيانات Data Tables هى : وحدات البناء Building Blocks الأولي فى قاعدة البيانات العلاقية. إنها الأوعية المستخدمة في التخزين الفعلي للبيانات المطلوب تخزينها في القاعدة ، وذلك بطريقة تسمح بتعريف ما يوجد بينها من علاقات ، وبالتالى فإنها تمثل مظهر الوجود الفعلي Foundation لقاعدة البيانات. وعليك أن تتذكـّر دائما

أن تمثيل البيانات في **جداول** هو الخاصية الأساسية التي تميز نموذج قاعدة البيانات العلاقية عن غيره من النماذج التنفيذية الأخرى لتمثيل البيانات ، مثل : النموذج الشبكي Network Model والنموذج الهرمي Hierarchical Model.

ويمكن تعريف جدول البيانات في النموذج العلاقي بأنه :

وعاء إلكتروني لحفظ بيانات متجانسة ، تتعلق بوصف خصائص مفردات أحد الموضوعات أو الكيانات الموجود في مجال تطبيق القاعدة Entity ، حفظاً منظماً في شكل سجلات Records.

ثانيا) خصائص جداول البيانات

Characteristics of Tables

من التعريف السابق ــ يتضح لنا أنه يشترط بالضرورة أن يتوافر في **جدول البيانات** Table مجموعة من الخصائص والشروط ... التي لا يجوز إغفال إحداها أو التغاضي عنه. هذه الخصائص والشروط هي :

(١) أن يختص الجدول بتمثيل كيان قابل للتمييز
A Single Table for A Different Entity

فالجدول الواحد يجب أن يمثل بالضرورة كيانا واحدا فقط Single Entity من الكيانات التى لها وجود مادي أو اعتباري فى عالم الواقع الذى أنشئت قاعدة البيانات لتمثيله (أشياء ، أشخاص ، وحدات ...). ولا يجوز مطلقا التغاضي عن شرط "**الجدول الواحد للكيان الواحد**" ، لأنه المبدأ الذى يتوقف عليه التصميم الصحيح والكفء لقاعدة البيانات العلاقية.

على سبيل المثال : فإن هيكل قاعدة بيانات مراقبة المخزون السابقة يتضمن سبعة جداول منفصلة ، يختص كل منها بتمثيل كيان منفرد ، له وجود مادي أو معنوي. هذه الأشياء

أو الوحـدات أو الأشخاص هى : المنتجات- فئات المنتجات – طلبيات الشراء (أو أوامر الـتوريد) – طـرق الشحن – الموردون – ا لموظفون (مندوبو المشتريات) – عمليات أو حـركة المخـزون. ورغم أن هذه الوحدات ترتبط ببعضها البعض فإنها تعتبر كيانات مخـتلفة لأغـراض تصميم قاعدة البيانات العلاقية ، ويجب أن ينفـرد بتمثيل كل منها جدول مستقل.

(٢) أن يكون لكـل جـدول اسـماً مميزاً ... A Unique Name

يجـب أن يكـون لكل جدول اسما فريدا ومميزا وغير قابل للتكرار علي مستوى القاعدة و يستخدم للإشارة الى هذا الجدول بالذات. فلا يجوز أن يتواجد جدولان للبيانات يحمـلان نفس الاسم فى كل أنحاء القاعدة ، على الإطلاق. ويفضل أن يعكس هذا الاسم طبـيعة البـيانات المخزونة فى الجدول ، أو نوع الكيان الذى يمثله. كما يلاحظ أنه كلما كان اسـم الجدول قصيرا ، كلما كان ذلك أفضل وأيسـر لأغراض البرمجة.

(٣) أن يحتوى الجدول بالضرورة على حقل مفتاح رئيسي
A Primary Key Field

إن حقـل المفتاح الرئيسي Primary Key Field – كما سوف يتضح لنا بالتفصيل بعد قليل – هو الحقل الذى يحتوى على قيم فريدة Unique وغير قابلة للتكرار. وبالتالى فإنه يمكن الاعتماد على هذا الحقل في القيام بوظيفتين أساسيتين :

⇦ **العمـل كمميـز وحيد للسجلات Identifier المختلفة التى يحتوى عليها الجدول.** فحقـل رقـم-جلوس-الطالب ، أو حقل الرقم –القومي-للمواطن ، أو حقل الرقم-الكـودي-للصـنف ... تصلح جميعها لأن تعتبر حقل المفتاح الرئيسي فى الجدول الـذى يحتوى عليها ، نظرا لأن من المؤكد أنها سوف تتضمن بيانات أو قيم غير قابلة للتكرار. وبالتالي يمكن استخدامها في تمييز واسترجاع أي سجل عند الطلب.

⇦ **التعمـــل كأســـاس لبناء علاقة الإرتباط Relationship مع أي جدول آخر مرتبط.**

فكمـــا إتضح لنا من معاينة النموذج المعروض في المبحث الأول عن هيكل قاعدة مراقبة المخزون السلعي بالشكل رقم (٤-١) ، فإن أي علاقة أرتباط بين جدولين يكون أحد أطرافها بالضرورة حقل المفتاح الرئيسي في أحد هذين الجدولين.

لهذه الأسباب ، يعتبر توافر حقل المفتاح الرئيسي شرطا أساسيا من شروط تصميم جداول قـــاعدة البـــيانات العلاقية ، نظرا لأنه يعتبر الأداة الوحيدة التى تعتمد عليها قاعدة البيانات العلاقية فى الربط بين بيانات الجداول المختلفة التى تحتوى عليها.

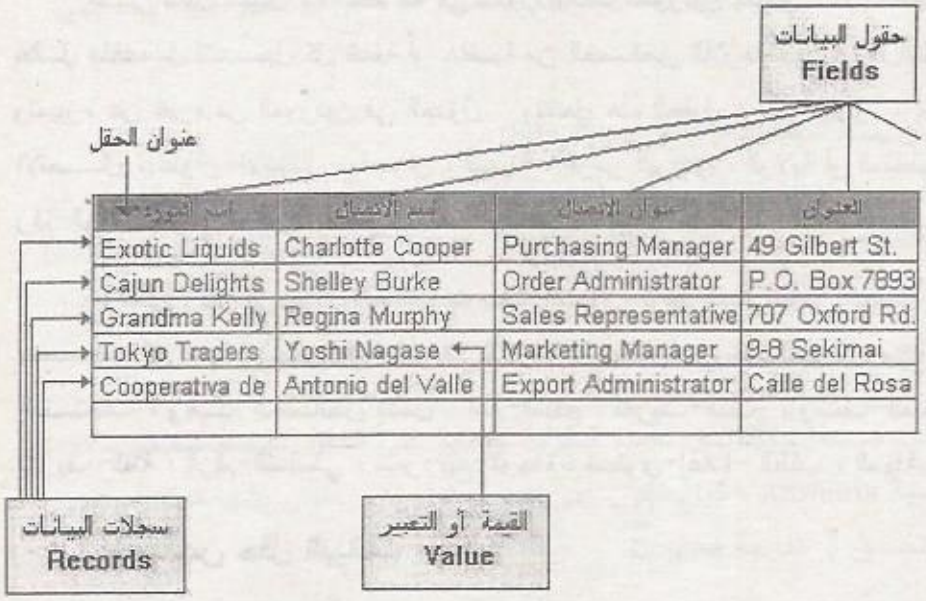
ثالثا) التصميم الداخلي لهيكل الجدول :

مـــن الناحـــية المنطقـــية ، يتـــخـــذ التصميم الداخلي لهيكل جدول البيانات شكل **المصفوفة** Matrix ، أي مجموعة من الأعمدة Columns والصفوف Rows. فى المفهوم الاصطلاحي لقواعد البيانات :

- يطلق مصطلح : **حقول البيانات** Data Fields على **الأعمدة** ... التى يتكون منها هيكل الجدول ،
- كمـــا يطلق مصطلح : **سجلات البيانات** Data Records على **الصفوف** ... التى يحتوى عليها هذا الجدول.

لتوضيـــح ذلك ، فإن الشكل التالى ، شكل رقـــم (٤-٢) ، يتضمن بعض محتويات جدول **الموردين** المشـــار إليه في قاعدة مراقبة المخزون السلعي سابقة الذكر. ومنه يتضح أن هناك ثلاثـــة مفاهيم أساسية مرتبطـــة بالتصميم الداخلي لهيكل جدول البيانات ، هـــي :

- حقل البيانات Data Field
- سجلات البيانات Data Records
- القيمة Value أو التعبير Expression.



شكل رقم (٤-٢)

الهيكل المنطقي الداخلي لجدول البيانات

فـي الفقـرات التالية ، نوضح المقصود بكل من : الحقول – السجلات – القيم ، مع بيان أهم الخصائص التي يتميز بها كل مفهوم منها عن غيره من المفاهيم.

(١) حقول البيانات

يمكـن تعريف **حقل البيانات** Field ، بأنه عبارة عن : أحد أعمـدة Columns المصفوفة التي تمثل الهيكل الداخلي لجدول البيانـات. ويختص كل حقل بيانات بتسجيل قطعة واحدة من البيانات ، وهذه القطعة تصف أحد خصائص الكيان Entity Attributes الذى يمثلـه جدول البيانـات.

علـــى سبيل المثال ، نلاحظ أنه في جدول بيانات **الموردين** السابق ، تم تخصيص **حقـــل منفصـــل** لتســـجيل كل صفة أو خاصية من الخصائص اللازمة لتعريف كل مورد وتمييزه عن غيره من الموردين فى الجدول. وتشمل هذه الحقول : اسم-المورد ، اسم-الاتصـــال ، عنوان-الاتصال ، العنوان ، المدينة ، الرمز البريدي ، الولاية أو المقاطعة ، رقم-الهاتف ، رقم-الفاكس ... الخ.

وبالمثل : فإنه فى جدول **المنـــتجات** بذات القاعدة ، توجد مجموعة من الحقول التى يخصـــص كل منها لتسجيل قطعة البيانات اللازمة لوصف إحدى خصائص كل منتج من المنـــتجات ، وهـــذه الخصائص تشمل : اسم-المنتج ، تعريف-المنتج ، وصف-المنتج ، تعريف-الفئة ، الرقم-التسلسلي ، سعر-بيع-الوحدة ، مستوى-إعادة- الطلب ، المهلة.

❑ خصائص حقل البيانات

يتوجب بالضرورة أن يتوافر فى كل حقل من حقول جدول البيانات مجموعة من الخصـــائص والشـــروط الهامـــة ، حتى يكون تصميم هذا الجدول سليما من وجهة نظر النموذج العلاقي.

ونوجز هذه الشروط فيما يلي :

أ - **أن يكون الحـــقل وحدة غير قابلة للتجزئة Atomic.**

فلا يجوز أن يتضمن قطعاً مختلفة من البيانات (كالعنوان ورقم التليفون معا ...) أو يتضـــمن بـــيانات قابلـــة للتجزئة (كالعنوان متضمنا المدينة أو الحي ، والاسم متضمنا اللقب). والقاعدة هنا هى : قطعة بيان واحدة فى الحقل الواحد.

ب - **أن يتحدَّد لكل حقـــل في الجدول اسما أو عنوانا Field Name مميزا.**

هذا الاسم يجب أن يكون مميزا ومختلفا عن أسماء باقي الحقول التي يحتوى عليها هـــيكل الجدول ، ويستخدم في الدلالة على هذا الحقل أو محتوياته بالتحديد في كل

علمـيات الاسترجاع والمعالجة. مع ملاحظة أنه يمكن استخدام نفس الإسم لتمييز حقل آخر في جدول آخر.

ج – أن يتـم تحديد نوع الحقل Data Type.

إن نوع الحقل هو الذي يحدد طبيعة البيانات التي سوف يُسمح مستقبلا بتخزينها في داخل هذا الحقل. من هذه الأنواع : حقل نصوص Text ، حقل أرقام Numbers حقـل تواريـخ Date ، حقل عملات Currency ، حقل برمجة شـيئية OLE ... الخ. ولهذا الموضوع عودة في الباب الثاني من هذا الكتاب.

... هـذا ، ويُطلـق علماء النظرية العلاقية على **الحقل** اسما اصطلاحيا ، هو : الخاصـية **Attribute** ، كما يطلقون على **عدد الحقول** التى يحتوى عليها هيكل أي جدول هو : **العُـمق** أو **الدرجة** Degree.

(٢) سجـلات البيانات

سـجل البيانات Data Record هـو أحد صفوف المصفوفة ، ويتضمن كافة قطع البيانات المسجلة في حقول الجدول المختلفة ، وتخص **مفردة واحدة** من مفردات الكيان الذي يمثله هذا الجدول. وهكذا : فإن أي سجل في جـدول بيانـات معين سوف يعـبر عن : سلسـلة من **الخصائص Attributes** المطلوب جمع وتخزين بيانات عنها ، والتى تستخدم فى مجموعها للتعرف على ، وتمييز ، كل مفردة عن غيرها من مفردات الكيان أو الموضوع الذي يمثله هذا الجدول.

على سـبيل المثال ، فإن أي سجل في جدول بيانات **الموردين** السابق يتضمن فى واقع البيانات اللازمة لوصف مجموعة من **الخصائص** التى تميز هذا المورد بالذات عن غيره من الموردين ، وتشـمل :

- **اسم-المورد** ، ويوضح الاسم التجاري أو الرسمى للشركة.
- **اسم-الاتصال** ، ويوضح اسم الموظف المسئول الذى تبعث باسمه المراسلات الخاصة بهذا المورد بالذات.
- **عنوان-الاتصال** ، ويوضح الوظيفة التى يشغلها هذا الموظف ، أو القسم أو الإدارة التابع لهـا.
- العنوان ، المدينة ، الرمز البريدي ، الولاية أو المقاطعة ، رقم الهاتف ، رقم الفاكس ... الى غير ذلك من البيانات.

❑ خصائص سجل البيانات

يشترط أن تتوافر فى كل سجل جديد يتم إضافته الي جدول البيانات مجموعة من الخصائص أو الشروط ، مجتمعة. هذه الشروط هى :

أ - **أن يكون لكل مفردة من مفردات الكيان الذى يمثله الجدول سجلا واحدا فقط Only One Record** : فلا يجوز أن يتضمن جدول الموردين السابق مثلا أكثر من سجل واحد لنفس المورد ، بل يجب أن يكون لكل مورد سجل واحد فقط يدل.

ب - **أن تكون بيانات السجل الواحد كافية لتمييز المفردة المعنية عن باقي المفردات فى الجدول.** إذ أنه يجوز أن تتشابه **بعض** بيانات سجل معين مع غيره من السجلات ، إلا أن كافة بيانات السجل الواحد **مجتمعة** يجب أن تصلح لتمييز المفردة التى تمثلها عن باقي المفردات فى الجدول. وبدون ذلك ، يكون تصميم هيكل الجدول ذاته غير سليم.

ج - **لا يجوز أن يكون السجل فارغا من أي بيانات Null Record**. فوجود سجل فارغ بأكمله يتعارض مع مفهوم قاعدة البيانات لأنه يكون سجل وهمي ولا يمكن ربطه بغيره من السجلات في الجداول المرتبطة. ولكن لا يمنع الأمر ، بطبيعة الحال

مـن أن يحتوى السجل على بعض الحقول الفارغة نظرا لأن بعض البيانات قد لا تكـون معـروفة ، علـى الأقـل فى الوقت الحاضر. ولكن : يشترط فى جميع الأحوال أن لا يكون حقل المفتاح الرئيسي لهذا السجل خاليا من أي بيانات.

ويلاحـظ أخـيرا أنـه لا وجود لمفهوم **رقم السجل** Record Number في قاعدة البـيانات العلاقـية ، لأنها لا تعتمد في استرجاع وتخزين السجلات على ترتيبها المادي Physical Order. هذا ، ويُطلق علماء نظرية قواعد البيانات العلاقية على **السجل** اسما اصطلاحيا هو : Tuble ، كما يطلقون على **عدد السجلات** التى يحتوى عليها الجدول في أي لحظة اسم : Cardinality.

(٣) القيم المسجلة في الحقول

القـيمـة Value ، فـي مفهـوم النموذج العلاقي ، هي قطعـة البيانات الواحدة المسجلة عند نقطـة تقاطع Intersection عمود معين مع صف معين داخل المصفوفـة. ونظـرا لأن العمـود هو أحد **الحقول** التى تصف خاصية محددة ، كما أن الصف هو أحد **السجلات** التى تتضمن البيانات اللازمة لوصف وتمييز إحدى المفردات ، فإنه يمكن القول بـأن المقصود **بالقيمة Value** ، هو : **قطعة البيانات** التى تصف إحدى خصائص مفردة معيـنة بـالذات من مفردات الكيان الذى يمثله الجدول. من أمثلـة القيمة : اسم مورّد محدد ، أو كمية أمر توريد معين ، أو رقم شيك مسحوب ...

❑ خصائص القيمـة

يشترط في القيمة التي تسجل في حقل البيانات بأي جدول الشروط التالية :

أ – **أن تكون القيمة من نفس نوع حقل البيانات** : Data Type. القيمة والحقل يجب أن يكونا من نفس النوع. فلا يجوز مثلا تسجيل تعـبير نصي Text في حقل من

مـن أن يحتوى السجل على بعض الحقول الفارغة نظرا لأن بعض البيانات قد لا تكـون معـروفة ، علـى الأقـل فى الوقت الحاضر. ولكن : يشترط فى جميع الأحوال أن لا يكون حقل المفتاح الرئيسي لهذا السجل خاليا من أي بيانات.

ويلاحـظ أخـيرا أنـه لا وجود لمفهوم **رقم السجل** Record Number في قاعدة البـيانات العلاقـية ، لأنها لا تعتمد في استرجاع وتخزين السجلات على ترتيبها المادي Physical Order. هذا ، ويُطلق علماء نظرية قواعد البيانات العلاقية على **السجل** اسما اصطلاحيا هو : Tuble ، كما يطلقون على **عدد السجلات** التى يحتوى عليها الجدول في أي لحظة اسم : Cardinality.

(٣) القيم المسجلة في الحقول

القـيمـة Value ، فـي مفهـوم النموذج العلاقي ، هي قطعـة البيانات الواحدة المسجلة عند نقطـة تقاطع Intersection عمود معين مع صف معين داخل المصفوفـة. ونظـرا لأن العمـود هو أحد **الحقول** التى تصف خاصية محددة ، كما أن الصف هو أحد **السجلات** التى تتضمن البيانات اللازمة لوصف وتمييز إحدى المفردات ، فإنه يمكن القول بـأن المقصود **بالقيمة Value** ، هو : **قطعة البيانات** التى تصف إحدى خصائص مفردة معيـنة بـالذات من مفردات الكيان الذى يمثله الجدول. من أمثلـة القيمة : **اسم** مورِّد محدد ، أو **كمية** أمر توريد معين ، أو **رقم** شيك مسحوب ...

❑ خصائص القيمـة

يشترط في القيمة التي تسجل في حقل البيانات بأي جدول الشروط التالية :

أ – **أن تكون القيمة من نفس نوع حقل البيانات** : Data Type. القيمة والحقل يجب أن يكونا من نفس النوع. فلا يجوز مثلا تسجيل تعـبير نصي Text في حقل من

نوع التواريخ Date Type ، كما لا يجوز تسجيل تعبيـــر منطقي (True/ False) في أحد الحقـــول من النوع الرقمي Numbers or Numeric Type.

ب - **أن تكـــون القـــيمة فى الحدود المسموح بها.** أي تكون القيمة المراد تسجيلها فى حقـــل البيانات من القيم الواردة فى ما يسمى **بالنطاق Domain**. النطاق الخاص بحقـــل معيـــن ، هـــو مستودع عام Pool يضم كافة القيـــم المسموح بها لوصف الخاصـــية التي يمثلها هذا الحقـــل. فلا يجوز أن يتضمن السجل قيمة تخرج عن حدود هذا النطاق. ومن أمثلة ذلك :

- نطـــاق حقل **كمـــية-الصنف** فى جدول طلبيات الشراء هو : أي رقم موجب يتراوح مثلا بين رقم ١ (الحد الأدنى) ورقم ١٠٠٠ (الحد الأقصى).
- نطاق حقل **رقم-التليفون** فى جدول الموردين هو : أي أرقام موجبة مكونة مـــن ٧ مـــنازل ، أو ١٠ منازل (إذا أخذنا في الإعتبار الرقم الكودي للبلدة) دون زيادة أو نقصان.

ج - **أن جـــواز تكرار القيمة معلق علي شرط.** فمن الجائز أن تكرر القيمة داخل أحد الحقـــول بالجدول ، كأن تكون الكمية المباعة من الصنف واحدة في أكثر من طلبية أو يحصـــل أكثر من طالب على نفس تقدير المادة. ولكن يشترط أن لا يكون هذا الحقـــل هو حقل المفتاح الرئيسي Key Field لهذا الجدول. فمن الشروط الأساسية لحقـــل المفـــتاح الرئيسي هو أن تكون القيم الواردة به غير قابلة التكرار ، وبالتالي تصلح كمميز وحيد للسجلات الموجودة بهذا الجدول ...

❑ توفيق المسميات

والآن : إن إزدحام المسميات علي نحو ما ورد في هذا المبحث قد يدعو إلى الخلط أو التشويش. لهذا السبب ، وإتماماً للفائدة ، يوضح الجدول التالي ، شكل رقم (٤-٣) ،

مقـابلـة بيـن المصطلحات العلمية لنظرية قواعد البيانات العلاقية ، فيما يتعلق بعنصر الجـداول ، وبين المصطلحات الموازية لها التي يشيع استخدامها حاليا بين الممتهنين في الممارسة التطبيـقية.

Formal Relational Terms	Informal Equivalent Terms	المقابل باللغة العربية
Relation	Table	جدول بيانات
Tuple	Record	سجل بيانات
Cardinality	Number of Rows	عدد السجلات بالجدول
Attribute	Field	حقل بيانات
Degree	Number of Columns	عدد الحقول بالجدول
Domain	Pool of Legal Value	القيـم المسموح بها

شكل رقم (٤-٣)

بعض المصطلحات المتقابلة الخاصة بقواعد البيانات العلاقية

* * *

٣ المبحث الثالث

حقـــول المفـــاتيح
Key Fields

حقـــول المفاتـــيح هي العنصر الثاني من العناصر الرئيسية التي تقوم عليها قـــاعدة البيانات العلاقية. وتعتبر هذه الحقول هي **الأساس الذي تبني عـلـيه علاقات الارتباط** ، اللازمة لربط السجلات فى الجداول المختلفة لتشكل معا نسيجا متماسكا وقابلا للاستخدام المشترك.

أولا) تعريف حقل المفتـاح
Key Fields Definition

إن أي علاقـــة إرتـــباط بين جدولين تتأسس بالضرورة علي وجود نوعين من حقول المفتاح :

- حقل **المفتاح الرئيسي** PK في الجدول الأول الذى تبدأ منه علاقة الإرتباط ،
- وحقل **المفتاح الخارجي** FK في الجدول الثاني المستهدف بعلاقة الإرتباط.

وعلـــي ذلك فإن حقـــول المفتاح بهذا المعنى تعـــتبر : **حقول عامة أو مشتركة** Common Fields ، أي مكـــررة بـــنفس المواصفات في أكثر من جدول (وغالبا بنفس الاســـــم). وهى تمثل في الحقيقة الحقول الوحيدة التي يُسمح بتكرارهـــا Duplicated بداخل قاعدة البيانات ، في ظل مفهوم القاعدة العلاقية.

علـــى سبيل المثال : إذا رجعنا الي الشكل رقم (٤-١) ، الذى يتضمن المخطط البيانى لهيكل قاعدة بيانات مراقبة المخزون السلعي ، سوف نلاحظ ما يلي :

- أن جـــدول **المنـــتجات** يحتوى على حقلين مختلفين من حقول المفتـــاح ، منهما : حقل : **تعريف-المنتج** ، الذي يربطه بجدول : عمليات المخزون.
- وفـــى الوقـــت ذاتـــه ، فإن جدول **عمليات المخزون** ، يحتوى بالمثل على حقلين مختلفيـــن من حقول المفتاح ، منها : حقل : **تعريف-المنتج** ، الذى يربطه بجدول **المنتجات**.

وبناء علي هذه العلاقة بين الجدولين ، إذا قمنا بإصدار طلب شراء (أو أمر توريد) جديد لأحد المنتجات ، ووصلت الطلبية الى المخازن بالفعل ، فإن بيانات وصف هذا المنتج لا يـــتكرر تســـجيلها فـــي جدول عمليات المخزون ، وإنما يستخدم حقل الربط : **تعريف-المنتج** ، للإشارة الى هذه البيانات فى جدول المنتجات.

ثانيا) أنـواع حقول المفاتيح

Types of Key Fields

تنقســـم حقول المفتاح فى القاعدة العلاقية الى نوعين رئيسيين ، هما : **المفتاح الرئيسي** Primary Key ، و**المفتاح الخارجي** Foreign Key.

وينقسم المفتاح الرئيسي بدوره إلى نوعين ، وذلك علي أساس مكونات هذا المفتاح :

- مفتاح رئيسى يتكون من حقل مفرد Single-Field PK.
- مفتاح رئيسي يتكون من حقلين أو أكثر Composite Key.

وإليك مفهوم كل نوع من أنواع حقول المفتاح ، واستخداماته ، بإيجاز :

(١) حقول المفاتيح الرئيسية

Primary Key Fields

في مفهوم النموذج العلاقي ، إن كل جدول من جداول قاعدة البيانات العلاقية يجب فى الواقع أن يحتوى على **حقل مفتاح رئيسي**. القيم المسجلة في هذا الحقل سوف تستخدم – أساسا كمميز للسجلات المختلفة التى يحتوى عليها الجدول عن بعضها البعض بشكل فريد وغير متكرر Unique Identifier بحيث يمكن استرجاع أي سجل منها عند الطلب بسهولة. وبعبارات شرّاح النظرية العلاقية : إن هذا الحقل يوفر آلية عناوين الوصول على مستوى السجلات: Tuple-Level Addressing Mechanism .

ويلاحظ أن القرار الذي يتخذه مصمم القاعدة بشأن ما هو الحقل الذي سوف يُتخذ كمفتاح رئيسي في كل جدول من جداول القاعدة يعتبر في الواقع من أهم القرارات التى يتوقف عليها نجاح تصميم قاعدة البيانات العلاقية بأكملها.

☐ خصائص حقل المفتاح الرئيسي

هناك ثلاثة خصائص لحقول المفتاح الرئيسية ، يجب التحقق من وجودها جميعها فى حقل المفتاح الرئيسي بكل جدول ، وذلك كشرط لتصميم قاعدة بيانات علاقية سليمة. هذه الشروط هى :

أ - **إن وجـــود حقـــل المفـــتاح الرئيسي فى أي جدول يعتبر ضرورة حتمية.** فلا يجـــوز أن يحـــتوى هـــيكل قاعدة البيانات على أحد الجداول بدون حقل مفتاح رئيسي، وإلا فإن هذا الجدول سوف يكون مقطوع الصلة بالجداول الأخرى في القاعدة ، وبالتالي فإن استرجاع البيانات المخزونة بها لن يتم بكفاءة.

ب - **يجب أن يحتوى حقل المفتاح الرئيسي وبالضرورة على قيم غير قابلة للتكرار حاضـــرا ومستقبلا.** فلا يجوز على الإطلاق أن يتضمن هذا الحقل قيما متماثلة أو متكررة Duplicated Values ، بمعني أن يكون سجلين أو أكثر من سجلات الجدول تحمل نفس القيمة. وإلا فإن هذا الحقل يفقد مبرر وجوده ، ويفشل في القـــيام بوظيفته ، وهى : العمل كمميز فريد Unique Identifier للسجلات التى يحتوى عليها جدول البيانات عن بعضها البعض.

على سبيل المثال ، فإن حقولا مثل : رقم جلوس الطالب فى الامتحان، أو الرقم التأمينـــي للموظف ، أو الرقم الكودي للصنف فى المخازن ... تصلح جميعها لأن تـــتخذ كحقـــل مفتاح رئيسي فى الجداول التى تتضمنها ، لأنه لا ينتظر أن يحمل نفس الرقم طالبان أو موظفان أو صنفان في نفس الوقت.

ج - **لا يجـــوز أن يحـــتوى جدول البيانات على أكثر من حقل مفتاح رئيسي واحد.** بمعـــنى أنـــه لا يجوز تعريف أكثر من حقل واحد فقط من الحقول التي يحتوى علـــيها هـــيكل أي جـــدول بيانات بوصفه حقل المفتاح الرئيسي لهذا الجدول. فالجدول الواحد له حقل مفتاح رئيسي واحد.

فـــإذا حـــدث وكان جدول البيانات يحتوى على أكثر من حقل واحد يتوافر فيها شـــروط حقـــل المفـــتاح الرئيسي ، أي يحتوى على قيم غير متكررة وبالتالي

يصـــلح لأن يستخدم كمميز فريد للسجلات ، فإن هذه الحقول سوف يطلق عليها اســـما اصطلاحيا هو : Candidate Keys. وعلى مصمم قاعدة البيانات فى هـــذه الحالـــة أن يختار من بينها ذلك الحقل الذى يتم تعريفه على أنه هو حقل المفـــتاح الرئيسي Primary Key لهذا الجدول ... أما الحقول الأخرى فتعامل معاملة الحقول الأخرى العادية ، وتوصف علي سبيل التذكرة بأنها : **حقول غير مميـــزة** Non-Identifier.

ومصداقاً لذلك ، فإنه بالرجوع إلى مخطط هيكل قاعدة بيانات مراقبة المخزون ، المعـــروض فـــي الشكل رقم (٤-١) بصدر هذا الفصل ، يمكنك ملاحظة أن هيكل كل جـــدول مـــن جـــداول البيانات السبعة يبدأ بحقل مفتاح رئيسي هو الحقل المكتوب **بخط ثـــقيل** Bold Typeface. نذكر منها علي سبيل المثال :

- حقل : **تعريف-المنتج** ، فى جدول المنتجات.
- حقل : **تعريف-طلبية-الشراء** ، فى جدول طلبيات الشراء.
- حقل : **تعريف-الفئة** ، فى جدول فئات الصنف.
- حقل : **تعريف-الموظف** ، في جدول الموظفين.

☐ المفتاح الرئيســـي المركـــب

Composite Key Fields

فـــي بعـــض الحالات ، قد يواجه مصمم قاعدة البيانات بمشكلة أن كافة الحقول الـــتى يحـــتوى عليها هيكل أحد الجداول من النوع الذى سوف يحتوى على قيم متكررة وغـــير مـــتـــفردة. وبالتالي : فلن يصلح أي حقل منها للاستخدام بمفرده كحقل مفتاح رئيسي ، يعمل كمميز وحيد للســـجلات في هذا الجدول.

ونظـــرا لأن القواعـــد العامـــة تقضى بضرورة أن يكون لكل جدول حقل مفتاح رئيســـي Primary Key ، فإن أحد الحلول المطروحة لحل هذه المشكلة تكون باستخدام المفتاح المركب Composite Key.

يـــتكون **المفـــتاح المركب** من حقلين مختلفين (أو أكثر) ، يتم اختيارهما من بين الحقول الموجـــودة فعـــلا داخل هيكل الجدول ، على أن يتميز هذان الحقلان **مجتمعان** بصفتين متلازمتيـــن :

- أنـــــه إذا نظـــرنا الـــى كل حقل منهما بمفرده ، سوف نجد أنـــه يحتوى فى حد ذاتـــه على قيـــم متكررة Duplicate Values ، وبالتالي : فإنه لا يصلح بمفرده كمميز للسجلات.
- إلا أنـــه إذا نظرنا إلى هذان الحقلان **كوحدة** واحدة ، سوف نجد أن القيم المركبة الواردة بهما تكون متـــفرّدة Unique وغير متكررة لجميع السجلات. وبالتالي فإن هذين الحقلين كوحدة واحدة يصلحان للعمل كمميز فريد للسجلات الواردة فى هذا الجدول.

وتـــبعـــا لذلـــك ، فإن **المفتاح المركب** سوف يصبح بمثابة **حقل المفتاح الرئيسي** لهـــذا الجـــدول ، ويعـــامل على هذا النحو في قاعدة البيانات التي تضم هذا الجدول. وبطبيعة الحال ، يجب أن يكون حقل المفتاح الخارجي المقابل له – إن وجد– علي نفس الشكل ... الأمر الذي يعني إضافة الكثير من التعقيد الي هيكل القاعدة.

لذلـــك : فـــإن الحل الأفضل لهذه المشكلة فى رأينا هو استخدام ما يمسي : حقل الترقـــيم التلقائي Auto-Number Fiels ، الأمر الذى سوف نتناوله في حينه عند دراسة نظام قواعد إدارة قواعد البيانات العلاقية Access في الباب الثاني.

(٢) حقـول المفاتيح الخارجية

Foreign Key Fields

إذا كنت قد استوعبت حقا مفهوم المفتاح الرئيسى فإنه يصبح من السهل عليك أن تـفهُم المقصود بهذا النوع من حقول المفتاح: **المفتاح الخارجي** Foreign Key Field. يمكـن القول ببساطة أن هذا الحقل يمثل الوجه الآخر للعملة ، لأنه هو الحقل المقابل لحقل المفتاح الرئيسي في الجدول الآخر المستهدف بعلاقة الارتباط.

إن حقـــل المفتاح الخارجي بأي جدول للبيانات لا ينشأ من فراغ ، بل يجب أن يقـابلـه ، ويفضل بنفس الاسم ، حقل مفتاح رئيسي في جدول آخر يراد الارتباط معه بعلاقـــة. وعلـى ذلك : فإن **حقل المفتاح الخارجي** في أي جدول من جداول قاعدة البيانات هو حقل يحتوى على قيـم مسجلة أصلا في **حقل المفتاح الرئيسي** لجدول آخر مرتبط به داخل نفس القاعدة.

على سبيل المثال ، وبالرجوع مرة أخرى الى شكل رقم (٤-١) الخاص بمخطط هـيكل قاعدة بيانات مراقبة المخزون ، فإنه يمكن لنا ملاحظة أنه يوجد حقلان خارجيان ضمن حقول جدول **عمليات المخزون** ، هما:

- حقـــل : **تعريف-المنتج** ، الذى يعتبر فى الوقت ذاته هو حقل المفتاح الرئيسي لجدول **المنتجات**.
- وحقـل : **تعـريف-طلبية-الشراء** ، الذى يعتبر بالمثل هو حقل المفتاح الرئيسي لجدول **طلبيات الشراء**.

وهكـذا يكـون الأمر بالنسبة لكل جدولين مرتبطين فى القاعدة ، فهذه العلاقة تقوم دائما على حقل مفتاح رئيسي في أحد الجدولين ، يقابله حقل مفتاح خارجي في الجدول الآخر.

❑ خصائص حقل المفتاح الخارجي :

في ضوء المفهوم السابق لحقل المفتاح الخارجي ، يمكن القول بأن هذا النوع من حقول المفتاح يتميز بالخصائص التالية :

أ - **الشـــرط الوحـيد لحقل المفتاح الخارجي هو أن يطابق حقل المفتاح الرئيسي المقابل له (على الطرف الآخر من علاقة الارتباط) وذلك من ثلاثة زوايا مختلفة هـي :**

- **نـوع البيانات المسموح بتخزينها Data Type.** فإذا كان نوع البيانات في حقل المفتاح الرئيسي هو رقم Number أو ترقيم تلقائي AutoNumber فإن نوع البيانات المسموح بتخزينه في حقل المفتاح الخارجي المقابل لـه يجب أيضا أن يكون من نوع : رقم Number .

- **نطـاق القـيم المسموح بتخزينه Domain.** والسبب في ذلك هو أن أيـة قيمة يراد تسجيلها في حقل مفتاح خارجي يجب بالضرورة أن تكون مؤشرا Reference ، أو يجـب أن تشير إلى قيمة مماثلة لها تماما ، تكون مسجلة مـن قبل في حقل المفتاح الرئيسي بالجدول المرتبط. وبدون ذلك ، سوف تتفسّخ العلاقات بين جداول البيانات التي تتكون منها القاعـدة.

- **اسـم الحقل أو عنوانه Field Name.** من القواعد التي نوصي بها دائما ضـرورة أن يحـرص مصمم قاعدة البيانات العلاقية على أن يعطى نفس الاسـم لكـل من حقل المفتاح الرئيسي وحقول المفاتيح الخارجية التى تشير إليه فى كل أنحاء القاعدة. فهذا الإجراء سوف تترتب عليه مزايا متعددة نذكر منها :

- أنـــه يساعد كثيرا على تفهم طريقة تصميم هيكل قاعدة البيانات.
- كمـــا أنـــه يسهل عملية الوصل Join بين الجداول المرتبطة ، وهي التي تحـــدث عند استخراج **تشكيلة من البيانات المنتقاة** التي تتواجد في بعض حقول أكثر من جدول واحد من هذه الجداول ، لغرض الإجابة على بعض استفسارات المستخدمين.

ب – **وفـــيما عـــــدا ذلك ، فإن حقل المفتاح الخارجي يختلف في خصائصه عن حقل المفتاح الرئيسي المقابل له** ، وذلك من ثلاث زوايا ، كما يلي :

- **أن وجـــود حقل المفتاح الخارجي اختياري وليس إجباري.** إن وجود حقل المفتاح الرئيسي في أي جدول هو وجود إجباري ، لأنه الأساس الذي يعتمد علـــيه فـــي تمييـــز سجلات الجدول وفي بناء علاقة الارتباط مع الجداول الأخـــري. أما حقل المفتاح الخارجي فلا ضرورة لوجوده في جدول معين إلا إذا كان هذا الجدول مستهدف بعلاقة ارتباط.

- **يجـــوز وجـــود أكثر من حقل مفتاح خارجي في الجدول الواحد.** فبينما لا يجـــوز أن يحـــتوى جـــدول بيانات معين إلا على حقل مفتاح رئيسي واحد فقـــــط ، فإن هذا الجدول يمكن أن يتضمن أكثر من حقل مفتاح خارجي ، طالمـــا أن الســـجلات التى يحتوى عليها هذا الجدول ترتبط بالسجلات التى تحتوى عليها الجداول المستهدفة.

فكمـــا يتضح من هيكل قاعدة بيانات مراقبة المخزون المشار اليها، يحتوى جـــدول **طلبيات الشراء** على حقل مفتاح رئيسي واحد ، وهو حقل **تعريف-طلبـــية- الشـــراء** ، فإنـــه يحتوى فى ذات الوقت على ثلاثة حقول مفاتيح

خارجية ، هى بالتحديد : حقل: **تعريف-المورد** وحقل : **تعريف-الموظف** ، وحقل: **تعريف-طريقة-الشحن**. وهذه الحقول الخارجية هي التى تربط السجلات الواردة فى جدول طلبيات الشراء بالسجلات الأخرى الموجودة فى كل من : جدول الموردون – جدول الموظفون – جدول طرق الشحن ، على الترتيب.

- **أن القيم الواردة بحقل المفتاح الخارجي قابلة للتكرار**. فعلي العكس من حقل المفتاح الرئيسي ، الذي يشترط أن تكون القيم الواردة به غير قابلة للتكرار في الحاضر أو المستقبل ، فإن حقل المفتاح الخارجي يسمح بذلك. وسوف نرى فيما بعد أن هذا الوضع يتحقق بالضرورة في ظل علاقة الارتباط من نوع واحد الي كثير One-To-Many.

وفى الواقع ، إن التطبيق السليم لاستخدام كل من حقول المفاتيح الرئيسية وحقول المفاتيح الخارجية فى تصميم جداول قاعدة البيانات العلاقية ، يعتبر من أهم خطوات بناء وتعريف هيكل القاعدة ، لأنه هو وحده الذى يسمح **بالربط الصحيح** بين سجلات البيانات المتناثرة فى جداول القاعدة ، عن طريق القيم المسجلة فى حقول المفاتيح المعنية بتلك الجداول.

* * *

المبحث الرابع

علاقـــات الارتبـــاط
Relationships

إن مفهوم **علاقات الارتباط** Relationships ، يشير الى : **همزة الوصل** ، أو **الصلة** ، أو **المسار** الذي يربط بين سجلات جدولين من جداول القاعدة ، علي أســـاس القيم المسجلة بكل من : حقل المفتاح الرئيسي في الجدول الأول الذي تـــبدأ منه علاقة الارتباط وحقل المفتاح الخارجي المقابل له في الجدول الثاني المستهدف بعلاقة الارتباط.

ففي نموذج قاعـــدة بيانات مراقبة المخزون الســـابقة ، أنظر الشكل رقم (٤-١) ، توجـــد علاقـــة ارتـــباط واحـــدة على الأقل بين كل جدول وبين أحد أو بعض الجداول الأخـــرى. فمثلا : توجد علاقة ارتباط بين جدول طلبيات الشراء وجدول الموردين ، كما توجد علاقة ارتباط جدول المنتجات وجدول عمليات المخزون . وفى الحقيقة ، إن القوة الحقيقـــية لقـــاعدة البـــيانات العلاقية إنما تكمن فى قدرة هذه القاعدة على استخدام المفاتيح الرئيســـية Primary Keys والمفاتـــيح الخارجية Foreign Keys بغرض إقامة شبكة من علاقـــات الارتـــباط ، الواضحة والصحيحة ، بين كافة الجداول التى يحتوى عليها هيكل قاعدة البيانات.

وفـــي الحقـــيقة ، إن نـــوع علاقة الارتباط القائمة بين سجلات جدولين من جداول البيانات لهو الذى يحدد شكل تشكيلات البيانات User View التى سوف يمكن استرجاعها مســتقبلا من هذين الجدولين. وهناك – من الناحية النظرية – ثلاثـــة أنواع من علاقات الارتباط. هـــذه الأنواع هى :

- علاقة واحد–بواحد : One-to-One Relationship.
- علاقة واحد–بكثير : One-to-Many Relationship.
- علاقة كثير–بكثير : Many-to-Many Relationship.

ونظـــرا لأن مـــن الأهمـــية بمكان فهم طبيعة هذه الأنواع ، وظروف تطبيقها ، من أجل التصميم الناجح لهيكل قاعدة البيانات العلاقية ، فإننا نوضح فيما يلي: المقصود بكل نوع ، وظروف ومتطلبات تطبيقـــه.

أولا) العـلاقـة واحـــد-بواحـــد

One-To-One Relationship

العلاقـــــة من نوع **واحد–بواحد** هي أبسط أنواع علاقات الارتباط التي قد توجد بيـــن سجلات جداول قاعدة البيانات ، لكنها في الوقت ذاته أقل الأنواع استخداما وشيوعا. تـــتواجد **علاقة واحد–بواحد** One-to-One Relationship عندما يوجد سجل **واحد فقـــط** عند طرفي علاقة الارتباط : أي أن **كل سجل** في الجدول الأول سوف يقابـــله **سجل واحد فقـــط** فـــي الجـــدول الثاني المرتبط معـــه بهذه العلاقـــة. فكأن السجلين المرتبطين معا بعلاقـــة واحد–بواحد هما في الأصل سجل واحد ، ويتعلق بوصف خصائص مفردة واحدة لنفس الكيان ، لكن هذه الخصائص تم فصلها في جدولين لأسباب تبرر ذلك.

لكـــن المشكلة مع هذا النوع من علاقات الارتباط هي أن شكل التصميم الناتج عنه يمـــثل خروجاً سافراً على أحد شروط بناء الجداول في القاعدة العلاقية ، وهو الشرط بأن

يضـم الجدول الواحد كافة البيانات المتعلقة بنفس الكيان أو الموضوع الذى يمثله الجدول ... لذلك ، يجب أن يكون هناك ما يبرر هذا الخـروج!!

❑ ظـروف تبـرير استخدام علاقـة واحـد - بواحـد

القاعدة العامة فى التصميم السليم لقاعدة البيانات العلاقية هى : ضرورة أن يتضمن الجـدول الواحد كافة الخصائص التى تصف بشكل كامل **الكيان** الذى يمثله الجدول ، ولا يصح فصل هذه الخصائص فى جدولين مختلفين أو أكثر. ومع ذلك ، فقـد تتواجد بعض الظروف التي يستحسـن أو يكون من الضرورى في ظلها استخدام علاقة **واحد-بواحد** ، وذلك على سبيل الاستثناء من القاعدة العامة السابقة.

هذه الظروف قليلة في التطبيق العملي ، ولا يجوز التوسع فيها. ولعل من أهم هذه الظروف والمبررات ما يلي :

- الرغبة في المحافظة على سرية البيانات
- الرغـبة في تحسين كفاءة التعامل مع جداول البيانات التي تحتوى علي عدد ضخم من الحقول.

وإليك البيان بإيجاز ...

(١) المحافظة على سرية البيانات.

إن أحـد أهـم المبررات العملية الوجيهة لتطبيق مثل هذا الفصل في بيانات كان يمكـن فـي الأصل إدماجها معاً في جدول واحد ، هو : المحافظة على سـرّية البيانات. فقـد يرى المصمـمّ أن بعض بيانات الجدول الأصلي سرية ولا يجوز أن يطلع عليها كل من يُسمح له بالوصول الي هذا الجدول واسترجاع بياناته. لذلك ، يتم فصل هذه البيانات

الســرية فــي جدول آخر مستقل ، مع ربطه بالجدول الأصلي عن طريق علاقة من نوع واحــد- بواحــد. وبهــذا الشكل من التصميم يستطيع مصمم قاعدة البيانات قصــر حق الوصـــول Authorized Access إلى بيانات الجدول الجديد على شريحة معينة أو عدد محدود من المستخدمين دون سواهم.

والأمثلة على هذا الموقف كثيرة في واقع الحياة العملية :

- ⇦ فالكثير من البنوك وشركات القطاع الخاص لا ترى حرجا من إطلاع فئات كثير من مستخدمي قاعدة البيانات على البيانات الشخصية للعاملين بها ، بينما تعتبر أن بيانات المرتب أو المكافآت التى يحصل عليها كل موظف من الأســرار التي لا يســمح بــالإطلاع علــيها إلا لأفــراد محددين بالذات. وبالتالي فإنه يصبح من الضــروري في مثل هذه الحالات فصل هذين النوعين من البيانات في جدولين ، مــع ربطهمــا معا بعلاقة **واحد-بواحد** ، مع تقييد حرية الوصول الي أي منهما علي النحو المطلوب.

- ⇦ والمنطق ذاته ينطبق أيضا على بعض البيانات المالية الحساسة الخاصة في بعض المنشــآت. ومن أمثلتها : **بيانات العملاء** ، مثل : حدود الائتمان المسموح بها ، أو الحــد الأقصى لفترة الســماح ، أو نسبة الخصم التي يحصلون عليها تــنزيلا مــن قوائم الأسعار الرسمية. ومن ثم فإن فصل هذه البيانات فى جدول مستقل عــن بــيانات العمــلاء الشخصية الأخرى ، سوف يجعل من الممكن قصر حق الوصــول الــى تلك البيانات الحساسة على أشخاص معينين بالذات ، بينما يسمح للمستخدمين الآخرين بالوصول فقط الى بيانات العميل الشخصية الأخرى التى لا تحمــل مــثل هذه الحساسية ، مثل : الاسم التجارى ، والعنوان ، ورقم التليفون ورقــم الفاكس ، والسجل التجارى ... وما إلى ذلك.

❑ حالـة توضيحية :

مـن منطلق الرغبة في المحافظة على سرية البيانات ، إذا فرضنا أننا نريد فصل حقل **المرتب** في جدول مستقل عن باقي الحقول التي يحتوى عليها هيكل جدول **الموظفين** في قاعدة بيانات شئون العاملين بإحدى المنشآت ، فإن ملامح الشكل المقترح للتصميم في هذه الحالة تتمثل فيما يلي :

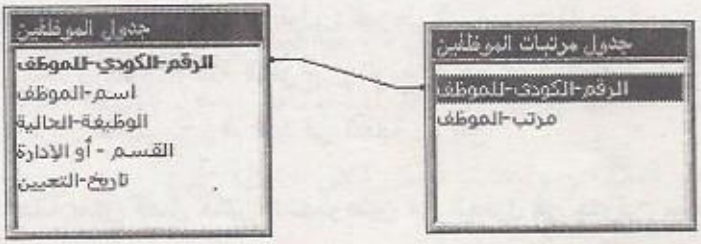
أ - يـتم إنشـاء جـدول جديد ، يطلق عليه : مرتبات الموظفين ، ويحتوى هيكل هذا الجدول على حقلين ، على الأقل ، هما :

- حقل **المرتب** ، المطلوب استبعاده من جدول **الموظفين**.
- حقـل مفـتاح خارجـي ، يقـابل حقل المفتاح الرئيسي بجدول الموظفين ، ويفترض هنا أنه حقل : **الرقم-الكودي-للموظف**.

ب - يـتم الربط بين الجدولين بعلاقة من نوع واحد-بواحد ، على أساس حقل : **الرقم-الكودي-للموظف**.

ج - عـنـد تعريف القيود والتحفظات الخاصة بقاعدة البيانات ، يتم قصر حق الوصول إلى جدول **مرتبات الموظفين** على الفئات المناسبة.

والشكل التالي يصور هذا الشكل من التصميم :



شكل رقم (٤-٤)

تصوير العلاقة من نوع واحد-بواحد

(٢) تجزئة الجدول الذي يحتــوى علي عدد ضخم من الحقول

الظرف الثاني من ظروف تفضيل استخدام علاقة **واحد-بواحد** ، هو ظرف يتعلق بحالــة خاصة ، تتمثل فى وجود جدول يحتوى هيكله علي عدد ضخم من الحقول. ومن أجــل الرغــبة فــي تخفيف مشكلات التعامل مع هذا الجدول الثقيل (مثل صعوبة تصميم شاشــات الإدخال – أو إلقاء عبء كبير علي الذاكرة أثناء التعامل مع الجدول ... الخ) ، فــإن من المرغوب فيه أن يتم تجزئة هذا الجدول الي جدولين مع الربط بينهما بعلاقة من نوع واحد-بواحد.

❑ حالة توضيحية

ومــثال ذلك جدول بيانات الموظفين في أي مؤسسة ، إذ أنه يحتوى بالطبيعة علي عدد ضخم جداً من الحقول. هذه الحقول يمكن في الواقع تصنيفها الي مجموعتين:

- **البيانات الشخصية** ، وتشمل حقول : الاسم – واللقب – عنوان السكن – المدينة أو الحــي – المحافظة – تاريخ الميلاد – محل الميلاد – رقم بطاقة تحقيق الشخصية – جهــة استخراجها – تاريخ استخراجها – الحالة الاجتماعية – عدد الأولاد ... الخ.
- **البــيانات الوظيفــية** ، وتشمل حقول : المؤهل الدراسي – تاريخ الحصول عليه – جهــة الحصول عليه – سنة التخرج – تاريخ التعيين – الوظيفة الحالية – الدرجة المالية – الرقم التأميني – رقم القيد في النقابة ... الخ.

وفــي هــذه الحالة يمكن فصل هاتين المجموعتين من الحقول في جدولين مستقلين مع الربط بينهما بعلاقة من نوع واحد بواحد :

- جدول **البيانــات الشخصية** ، يتضمن مجموعة حقول البيانات الشخصية.

- جدول **البيانات الوظيفية** ، ويتضمن مجموعة حقول البيانات الوظيفية.

ومـن المقـترح أن يكون حقل الربط المشترك هو حقل **الرقم-الكودي-للموظف**. وتكـون علاقـة الارتباط التي تقوم بين سجلات هذين الجدولين ، تأسيسا على هذا الحقل المشـترك ، هـي علاقـة من نوع **واحد-بواحد** ، نظرا إلى أن سجل البيانات الشخصية الخاصـة بموظف معين في الجدول الأول ، سوف يقابله سجل واحد فقط للبيانات الوظيفة الخاصـة بـنفس الموظف في الملف الثاني. هذا يعنى أن بيانات سجلات الجدول الثاني مكملـة لسـجلات الجـدول الأول ، حيـث تتعلـق جميعها في الجدولين معـاً بوصف الخصائص الشاملة لنفس الكيان أو الموضوع ، وهـو : **الموظف** المختص.

ثانيا) عـلاقـة : واحـد- بكثيـر

One-To-Many Relationship

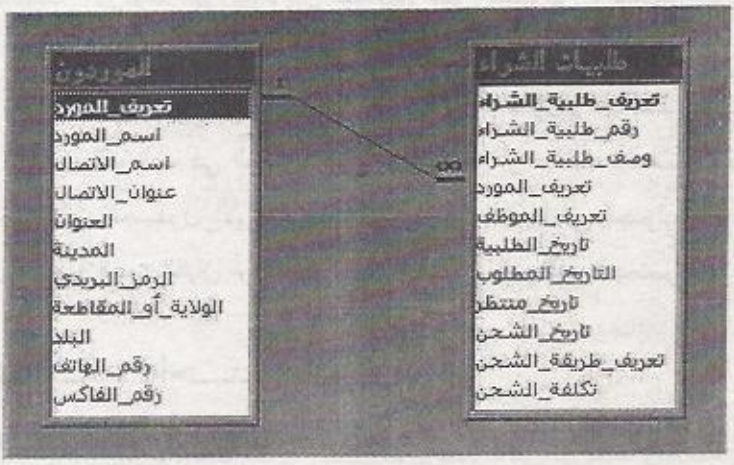
علاقـة **واحـد-بكثـير** هي أكثر أنواع العلاقات استخداما وشيوعا فى تطبيقات قواعـد البيانات العلاقية. وفيها يكون من المتوقع أن أي **سـجل واحد** فى الجدول الأول سوف يرتبـط **بأكثر من سـجل** فى الجدول الآخر المرتبط.

الأمـثلة على هذا النوع من علاقات الإرتباط عديدة ... ففي قاعدة بيانات مراقبة المخـزون المشار في المبحث الأول من هذا الفصل ، توجد هناك مثـلا عـلاقة ارتباط من نوع **واحد-بكثير** بين كل من : جدول **الموردين** وجدول **طلبيات الشراء**.

هذه العلاقة يصورها لنا الشكل التالي ، شكل رقم (٤-٥). بمناظرة هذا الشكل ، تتضح لنـا الحقائق التالية :

- من المنتظر أن كل مورد له سجل بجدول **الموردين** سوف يكون له عادة أكثر من أمر توريد واحد مسجل بجدول **طلبيات الشراء**.

- أن حقـــل المفـــتاح الرئيسي **تعريف-المورد** هو الذى يربط بين المورد المعين وبين أوامر التوريد المتعددة الخاصة به ، أي بين واحد-وكثير.



شكل رقم (٤-٥)

نموذج لعلاقة واحد-بكثير One-to-Many Relationship

كمـــا يلاحـــظ من الشكل ذاته أن هناك أعرافا معينة يتم تطبيقها فى تمثيل هذا النوع من علاقـــات الارتباط : **علاقة واحد بكثير** One-to-Many Relationship. حيث يتم تمييز طرفي خط الارتباط الموصل بين الجدولين كما يلي :

- فـــي ناحـــية رأس العلاقة ، أي **جدول الموردين** ، وهو الجدول الذى يحتوى على حقل المفتاح الرئيسي ، يُكتب رقم [1] للدلالة على " **واحد** ".
- وفى الناحية المقابلة المستهدفة بالعلاقة ، أي جدول طلبيات الشراء ، تُكتب علامة ما لا نهاية، للدلالة على " **بكثيـــرين** ".

وذلـــك بالإضافة إلى ما أوضحناه من قبل من أن حقل المفتاح الرئيسي يكتب بالخط الثقيل Bold ، لتمييزه عن غيره من الحقول ، أو يكتب بجواره الرمز **PK** .

ونظـرا لأن من المهم تمييز اتجاه علاقة واحد-بكثير ، فإن بعض الشُراح (وبعض المـراجع المتصلة بالموضوع) يميلون إلى التمييز في وصف الجدولين المرتبطين ، وذلك علي النحو التالي :

- وصف الجدول الأول (ناحية الواحد) بأنـه : **الجدول الرئيسي** Master Table في العلاقـة.
- ووصـف الجدول الآخر المرتبط معه (ناحية الكثيرين) بأنه : **جدول الصفقات أو المعاملات** Transaction Table.

ثالثا) العلاقـة : كثيـر- بكثيـر

Many-To-Many Relationship

علاقـة كثيـر-بكثير Many-to-Many Relationship ، تعتبر في الواقع أعقـد أنـواع علاقـات الارتـباط بين الجداول التي يصادفها مصمم قاعدة البيانـات العلاقية. وتوجـد هذه العلاقـة عندما يكون كل سجل فى الجدول الأول من طرفيّ العلاقة مرتبطا بأكـثر مـن سجل فى الجدول الآخر، وفى نفس الوقت يكون كل سجل فىّ الجدول الثاني مرتبطا بالمثل بأكثر من سجل فى الجدول الأول.

❑ أمثلة من الواقع العملي

إن الواقـع العملي فى الكثير من التنظيمات حافل بالمواقف التى تتواجدُ فيها مثل هذه العلاقات التبادلية بين جداول البيانات. ومن أمثلة المواقف التي تتواجد فيها :

➢ **قاعدة بيانات الإنتاج بأحد المصانع :**

فـى العادة ، سوف تحتوى على جدول **للمنتجات** وجدول **للخامات** الداخلة فى صنع هـذه المنـتجات. وهنـا : نجد أن كل منتج فى جدول المنتجات يرتبط بأكثر من

مادة فى جدول الخامات ، لأن صنع المنتج يتطلب عادة خامات متعددة . وفى الوقت ذاته ، نجد أن كل مادة من المواد فى جدول الخامات ترتبط بأكثر من منتج من المنتجات الواردة فى جدول المنتجات ، نظرا لأن الخامة الواحدة سوف تدخل عادة فى صنع أكثر من منتج من المنتجات المتجانسة ، وإن يكن بمقادير مختلفة.

وهكذا تكون العلاقة بين السجلات فى كل من جدول المنتجات وجدول الخامات هى علاقة من نوع **كثير-بكثير**.

- **قاعدة بيانات الطلبيات الواردة من العملاء بمنشأة للتسويق.**

فى قاعدة تسجيل ومتابعة بيانات الطلبيات الواردة من العملاء بإحدى شركات التسويق ، من الشائع وجود جدولين ترتبط سجلاتهما بعلاقة من نوع كثير-بكثير : جدول **للمنتجات** وآخر **لفواتير المبيعات**. سوف نجد أن كل منتج فى الجدول الأول يرتبط بأكثر من فاتورة بيع فى الجدول الثاني ، لأن المبيع منه كثير . كما أن كل فاتورة فى الجدول الثاني قد تحتوى على أكثر من منتج بالجدول الأول. بمعنى : أن المنتج الواحد يدخل عادة فى عدة فواتير بيع مختلفة ، كما أن الفاتورة الواحدة قد تتضمن عدة منتجات مختلفة.

ومن ثم فإن علاقة السجلات فى كل من جدول طلبيات البيع وجدول المنتجات ، تكون علاقة من نوع **كثير-بكثير**.

❑ تمثيل العلاقة كثير- بكثير

إن المشكلة مع العلاقة من نوع كثير-بكثير هى أن نموذج القاعدة العلاقية لا يستطيع التعبير عنها في صورتها البحتة. هذا لا يعنى إستحالة تمثيل هذه العلاقة ، وإنما يكون الحل هو تحويل العلاقة **كثير-بكثير** إلى علاقتين من نوع **واحد-بكثير** وذلك عن طريق إضافة **جدول آخر وسيط Intermediate Table** .

ولتوضـيح هـذا الحـل ، نعود الى نموذجنا الافتراضي عن قاعدة بيانات مراقبة المخـزون. ولنا أن نتصور أن العلاقة الأصلية التى ترتبط – من الناحية الواقعية – بين جدول **المنتجات** وجدول **طلبيات الشراء** كانت في الأصل علاقة **كثير–بكثير**. حيث أن :

- كـل منتج فى **جدول المنتجات** ، يمكن أن يصدر بشأنه أكثر من طلب شراء واحد فى جدول طلبيات الشراء.
- وكـل طلبـية فـى **جـدول طلبيات الشراء** ، قد تتضمن شراء أكثر من منتج من المنتجات الواردة فى جدول المنتجات.

إن التمثيل الصحيح للعلاقة بين هذين الجدولين يتطلب إتخاذ الإجراءات التالية :

١ – إنشاء جدول ثالث وسـيط ، نختار له اسما يعكس علاقة الوساطة التى سوف يقوم بها بين جدولي العلاقة الأصلية ، هو جـدول : **عمليات المخـزون**.

٢ – تعريف حقل مفتاح رئيسي لهذا الجدول للعمل فيما بعد كمميز وحيد لسجلاته ، هو حقل : **تعريف–عملية–المخزون**.

٣ – إضافة حقلين من حقول المفتاح الخارجية Foreign Keys إلى هيكل هذا الجدول :

- حقـل : **تعريف–المنتج** ، ويقابل المفتاح الرئيسي لجدول المنتجات.
- حقـل : **تعريف–الطلبية** ، وهو يقابل حقل المفتاح الرئيسي لجدول طلبيات الشـراء.

٤ – إضافة أي حقول أخرى عادية إلى هيكل الجدول ، للتعبير عن خصائص كل عملية علي حدة ، مثل : الكميات المطلوبة – الكميات المستلمة – سعر الوحدة ... الخ.

الشكل التالي ، شكل رقم (٤–٦) ، يصور نتيجة تطبيق هذه الإجراءات. ويتضح منه أن علاقـة **كثـير–بكثير** ، التى كانت قائمة من الناحية العملية بين كل من جدول المنتجات وجـدول طلبيات الشراء ، قد تحولت من الناحية المنطقية الى علاقتين مختلفتين من نوع **واحـد–بكثيـر** . هاتان العلاقتان هما :

- علاقــة واحد–بكثير بين كل من : **جدول طلبيات الشراء وجدول عمليات المخزون** وهــى التى تسمح لنــا بالحصول على إجابة أي استفســار عن المنتجات المتعددة الداخلة فى أي طلبيــة معينة من الطلبيات بجدول المنتجــات.
- وعلاقة واحد–بكثير بين : **جــدول المنتجات وجدول عمليات المخزون** ، وهى التي عــن طــريقها نتحصل على إجابة أي استفسار عن بيانات الطلبيات المتعددة التي صدرت بشــأن منتج معين.

شكل رقم (٤–٦)

تحويل علاقة كثير–بكثير Many-to-Many

الى علاقتين من نوع : واحد–بكثير One-to-Many

* * *

٥ المبحث الخامس

القيـود والتحفظات

Constraints

العنصـر الرابع من العناصر الأساسية المرتبطة بتصميم قواعد البيانات العلاقية هو عنصر: **القيـود والتحفظات** Constraints ، وهي التى يتم فرضها ومراقبة الالـتزام بتطبيقها فى عمليات استرجاع وتحديث البيانات ، وذلك من أجل ضمان صـحة وتناسـق وتكامل وسرية البيانات المخزونة في القاعدة. وتقـع مسئولية تعريف هذه القيود والتحفظات على عاتق مدير النظام DBA.

فـى الفقـرات التالية ، سوف نلقى الضوء – بإيجاز – على أربعة أنواع من هذه القيـود والتحفظات ، هى :

- قيـود نطاق القيـم Domain Constraints.
- قيـود حقول المفاتيـح Key Constraints.
- قاعدة تكامل الكيانات Entity-Integrity Constraint.
- قاعدة التكامل الإشاري أو المرجعي Referential Integrity Constraints.

أولا) قيــود نطــاق القــيم

Domain Constraints

الــنطاق Domain ، كمــا أوضــحنا من قبــل ، هو مجموعة القيــم المسموح باستخدامها فى وصف خاصية معينة Attribute ، أي التى يُسمح بتسجيلها فى حقل معين مــن حقول جدول البيانات. هــذا النوع من القيود يجرى تحديده على مستوى **كل حقــل** من الحقول التى يتكون منها هيكل جدول البيانات ، كل على حدة.

من أهــم أنواع هذه القيــود ، أربعة أنواع ، يمكن أحيانا الجمع بينها ، هــي :

- قيــود نوع البيانات Data-Type Constraints.
- قيــود خواص البيانات Data-Properties Constraints.
- قيــود صحة البيانات Data-Validation Constraints.
- قيــود القيــم المحصورة Enumerated-Values Constraints.

وفيمــا يلي مناقشــة سريعة لمفهوم ، وصور تطبيق ، كل نوع.

١- قيــود نــوع البيانات
Data Type Constraints

قيود نوع البيانات من أهم صور قيود النطاق ، حيث أنها تحدد طبيعة البيانات التي يُســمح للمســتخدم بتســجيلها مستقبلا فى حقول البيانات. هناك أنواع معيارية سائدة فى أغلــب نظــم قواعد البيانات العلاقية ، نذكر منها : حقل النصوص Text، حقل رقمــي Number ، حقــل مذكرات Memo ، حقل تواريخ Date ، حقل وقت أو زمن Time ، حقــــل منطقي Logical ، حقل عملات Currency ، حقل برمجة شيئية OLE ... وما الي ذلك.

٢- قيـود خواص البيانات
Data-Properties Constraints

هـــذا النوع من القيـود يتعلـق بوصف وتعريف خواص معينة للبيانات التى يتم تسجيلها فى كل حقل ، نذكر منها بوجه خاص ما يلي :

- **حجـــم البيانات** Field Size ، أي عدد الحروف التي يتسع لها مقاس الحقل ، ولا يمكن تخزين ما يزيد عليه ، إن وُجد.
- **الصـــورة أو الشكل** Data Format ، الذي سوف تتخذه هذه البيانات ، سواء فى أثناء إدخالها لغرض التخزين ، أو عند عرضها للمستخدم كنتيجة للاستفســار.
- **حتمية الاستيفاء** Required ، بحيث لا يقبل النظام تخزين السجل الجديد برمته ما لم يقـم المستخدم بتحديد قيمة لهذا الحقل.

لـنأخذ مثالا واحدا على هذا النوع من القيود ، هـو : أحجام البيانات التي يمكن تخزينها فـي **الحقـول الرقمية** Number Data Types ... إذ تشمل هذه الخاصية وحدها تشكيلة واسـعة من البدائل ، نوضحها فى الشكل التالي ، شكل رقم (٤-٧) :

Data Type	Can Handel Numbers	
	From	To
Byte	0	255
Integer	-32,768	32767
Long Integer	-2,147,483,648	2,147,483,647
Single	-3.402823 E38	-1.401296 E-45
	1.401296 E-45	3.402823 E38
Double	-4.94065645841247 E-324	-1.79769313486232 E308
	1.79769313486232 E308	4.94065645841247 E324

شكل رقم (٤-٧) : أحجام الحقول الرقمية Numeric Data Types

٣- قيــود التحقق من صحة البيانات
Data-Validation Constraints

هذا النوع من قيود نطاق البيانات يتعلــق بتعريف القواعد التي يجب الالتزام بها عــند إدخال البيانات في حقل معين ، بهدف التحقق من صحة هذه المدخلات. فالبرنامج ســوف يفحــص القــيمة المطلوب إدخالها في الحقل ، ولا يُسمح بتخزينها إلا إذا كانت مســتوفية لهذه الشروط ، بأكملها ، ويتم الفحص بشكل تلقائي أثناء عملية الإدخال ، وذلك دون أن يطلب المستخدم ذلك ، أو حتى يشعر بحدوثه.

وفيما يلي نماذج لأهم صور قيود التحقق من صحة المدخلات .

- استخدام **الحدود** Limits Constraints ، كما فى اشتراط أن يكون تاريخ استحقاق الكمبيالة أو السند الإذني لاحقا لتاريخ اليوم الحالي ، كحد أدنى : **[()Date<]**.
- استخدام **المدى** Sub-Range Constraints ، كما فى اشتراط ضرورة أن يقــع سن العامل أو الموظف بين ١٨ سنة و ٦٥ سنة.
- استخدام **الصــورة** Picture Constraints ، كما فى اشــتراط أن يتكون الرقــم الكودي للعامل من حرفين أبجديين يعقبهما ثلاثة أرقام **[AA999]**).

٤- قيــود القــيم المحصورة
Enumerated-Data List Constraints

هــذا الــنوع من القيود هو صورة أخرى من صور قيود النطاق ، حيث يتطلب تعــــريف لائحــة أو قائمة بكافة القيم المسموح بإدخالها فى حقل معين ، على سبيل الحصــر ، ولا يُــترك للمستخدم بعد ذلك إن يختار إحدى هذه القيم دون سواها. إن الجدول الرقيب Lookup Table ، الذى عرضناه من قبل كموقف يبرر تطبيق علاقة **واحد-بواحد** ، يعتبر بحق من أبــرز صور قيود النطــاق.

ثانيا) قيـود حقـول المفـاتيـح

Key-Field Constraints

فـي مـرحلة تصمـيم هيكل جدول البيانات ، يجب تطبيق ثلاثة قيود أو قواعد هامـة ، والعمـل على مراقبة الالتزام بتطبيقها فيما بعـد فى كافة أحوال الوضع الراهن للبيانات Data Instance. ونوجز أبرز هذه القيود فيما يلي :

- **أن تشكل خصائص كل سجل بذاتها مفتاحـا شـاملا : Super Key**

إن جـدول البيانات فى القاعدة العلاقية هو مجموعة **Set** من السجلات. ونظرا لأن كـل عنصر فى المجموعة ، بالمفهوم الرياضي ، يجب أن يكون متميزا Distinct ، كذلـك فـإن كـل سجل فى الجدول يجب أن يكون متميزا عن باقي السجلات ، بحيث لا يحـتمل أن يحـتوى سـجلان مختلفان على نفس القيم فى جميع الحقول أو الخصائص. ويعـنى ذلـك ، أن هـذا القـيد يقضـى بضرورة أن يحتوى هيكل كل جدول على تلك المجموعـة مـن الخصـائص (الحقول) التى يستحيل معها أن تتماثل كافة القيم الخاصة بوصـف سجلين مختلفين من سجلات هذا الجدول. ويطلق العلماء على هذه المجموعة من الخصائص اسـما اصطلاحيا ، هو : المفتاح الشامل Super-Key لمنظور الجدول.

- **أن يتضمن هيكل الجدول حقلا واحداً فقط كمفتاح رئيسي Primary Key**

حقـل المفتاح الرئيسي هو الحقل الذى يتضمن قيما فريدة وغير متكررة. ومن ثـم فإنه الحقل الذى يمكن الاعتماد عليه كمميز وحيد Unique Identifier للسجلات التى يحـتوى عليها هذا الجدول. وقـد أوضحنا من قبل أن هيكل الجدول قد يتضمن أكثر من حقـل واحد يتضمن قيما فريدة وغير متكررة ، ويمكن من حيث المبدأ أن يعمل كل حقل

مـــنهم بذاته كمميز للسجلات ، ويطلق العلماء على هذه الحقول اسم Candidate Keys. فـــي هـــذه الحالـــة ، يجب أن يختار القائم بتصميم قاعدة البيانات حقل واحد فقط من هذه الحقول للعمل كمفتاح رئيسي ، بينما تظل الحقول الأخرى مجرد خصائص للســـجلات.

- **أن يكون حقل المفتاح الرئيسي قابـــلاً للثبات فى الزمـــن Time-Invariant**

ويقضى هذا القيـــد بضرورة أن يستمر حقل المفتاح الرئيسي في التمسك بخاصية الانفـــراد والتميز Uniqueness في المستقبل ، مهما أضيفت سجلات جديدة إلى مضمون جـــدول البـــيانات. ويســـاعد على تطبيق هذا القيد منذ البداية أن يتفهـــم المصمم معنى الخاصـــية التى يعبر عنها كل حقل. على سبيل المثال ، فإننا لا نستطيع أن نختار ، ولا يجوز أن نختار ، حقل اســـم-الطالب كمفتاح رئيسي ، لأنه لا يوجد ضمان بأنه لن يلتحق بالمعهد طالبان يحملان نفس الاسم الثلاثي أو حتى الرباعي!!

ولا شـــك فى أن الالتزام بتطبيق القواعد السابقة لا يعنى فقط سلامة تصميم هيكل كل جدول من جداول قاعدة البيانات ، ولكنه سوف يكفل أيضا تماسك العلاقات التى يتم تعريفها للربط بين هذه الجداول.

ثالثا) قاعـدة تكامُـل الكيانـات

Entity-Integrity Constraint

تقضـــى قـــاعدة تكامل الكيانات ، بوصفها نوعا من أنوع القيود والتحفظات في قـــاعدة البـــيانات العلاقـــية ، بأنـــه لا يجوز أن يحتوى حقل المفتاح الرئيسي بأي جدول للبيانات على قيمـــة فارغة : Null Value. ويرجع ذلك حقيقة أن القيم المسجلة في حقل المفـــتاح الرئيسي بأي جدول هي التي تعتمد عليها القاعدة في التعرف على أي سجل من

سـجلات هـذا الجـدول ، وفى ربط هذا السجل بغيره من الحقول المسجلة في الجداول الأخـرى المرتـبطة بهذا الجدول. وبالتالي : فإن احتواء حقل المفتاح الرئيسي على قيم فارغة يعنى أننا لن نستطيع التعرف على بعض السجلات التى يحتوى عليها هذا الجدول أو ربطها بسجلات الجداول المرتبطة به.

رابعا) قاعدة التكامل الإشاري أو المرجعـي

Referential Integrity Constraints

فـي الوقـت الذي تنطبق فيه القيود السابقة (قيود النطاق ، قيود الحقول ، قيود البيانات) على مستوى الجدول الفردي Table–Individual فإن قاعـدة التكامل الإشاري أو المرجعـي تنطبق على العلاقة بين جدولين مختلفين ، وتستهدف المحافظة على توافق واتسـاق البيانات الخاصة بسجلات هذين الجـدولين.

في المعـنى الاصطلاحي الدقيق ، يُقصد بمفهوم : **التكامل الإشـاري أو المرجعي** Referential Integrity – عـدم احتواء قاعدة البيانات على أي قيـم واردة في حقل المفتاح الخارجي بأحد الجداول لا يكون لها قيم مناظرة فى حقل المفتاح الرئيسي بالجدول الآخـر المرتبط بالعلاقة ، وهو بالطبع الحقل الذى يشـير إليه – أو يرجع لـه – حقل المفتاح الخارجي.

علـى سبيل المثال : وبالنظر إلى قاعدة بيانات مراقبة المخزون ، التي قتلناها بحثا حـتى الآن !! ، إفـترض أنـنا أصدرنا طلب شراء جديد ، وعند تسـجيله في جدول طلبـيات الشـراء ، أدرجـنا بالخطأ قيمـة غير صحيحة في حقـل المفتاح الخارجي : تعـريف–المورد ، وسمح النظام بذلك. فإن ذلك يعنى أنـه لن يكون هناك قيمة مناظرة لهـذه القيمة الخاطئة ضمن النطاق المسموح به لحقل المفتاح الرئيسي بجدول الموردين ،

وهــو نفسه حقل : **تعريف-المورد**. وهكذا تضيع الصلة المفترضة بين هذين السجلين ، أي أمر التوريد والمورد الذي نفـّذه!!.

فــي هذه الحالة ، يُطلق على السجل المشار إليــه فى جدول طلبيات الشراء اسم : **ســجل لقيــط Orphan Record** ، لأنــه سجل ليس لــه **والدين Parent** فى الجدول المســتهدف بالعلاقــة ، وهو : جدول الموردين. ولا شك أن السماح بقبول تسجيل مثل هذه السجلات اللقيطة فى جداول القاعدة سوف يعنى انتهاك تكامل البيانات فى القاعدة .

وعلــى ذلــك ، فإن قاعدة **التكامل الإشاري أو المرجعي** هي القاعدة الحاكمة التى يجــب تطبــيقها فــى تصميم قــاعدة البيانات العلاقية ، والتى يجب أن يعمل نظام إدارة قــاعدة البــيانات DBMS علــى ضمان الالتزام بها وتطبيقها في مواجهة كافة محاولات اســترجاع وتحديــث البــيانات التي يقوم بها مستخدمو القاعدة ، لأنها الضمان الوحــيد لتحقيق هدف المحافظة على تكامل البيانات المخزنة في تلك القاعدة.

وإتماما للفائدة ، سوف نعرض فيما يلي لمحات عن كيفية تطبيق تلك القاعدة.

❑ تطبيق قاعدة التكامل الإشاري :

فــى التطبــيق العملي لقاعدة التكامل الإشاري ، يجب على مصمم قاعدة البيانات Database Designer أن يحدد ما الذى يجب أن يقوم به النظام ، أو رد فعــله ، فى حالة اكتشاف محاول إدخال سجل لقيط ، أو محاولة تعديل أو حذف إحدى القيم المســجلة فعــلا وبشــكل سليم فى أحد حقول المفتاح الخارجية أو حقول المفتاح الرئيسية بالجداول المرتبطة؟

إن **الاحــتمالات** المــتاحة أمام المصمم فى هذه الحالة هى أن يكون رد فعل النظام واحدا من أمــرين ، لا ثالث لهما :

- أن يـتم رفض العملية تماما ، وإعادة قاعدة البيانات الى الوضع السابق على محاولة تنفيذ هذه العملية Initial Database State.
- أو يقـوم النظام بقبول هذه العملية ، ولكن مع اتخاذ إجراءات تعويضية أو تكميلية لضمان صحتها.

ولسـوف يـتوقف الاختيار السليم بين هذا البديل أو ذاك على عاملين ، هما :نوع العملـية المطلوب تنفيذها على البيانات ، وعلى ظروف الوضع الراهن للبيانات المخزونة حاليا فى القاعـدة. لتوضيح ذلك ، سوف نعرض فيما يلي كيفية تطبيق قاعدة التكامل المرجعي أو الإشاري على عمليتين مختلفتين من هذا النوع ، همـا : عملية **حـذف** قيمة مخزونة فى حقل المفتاح الرئيسى بجدول معين ، وعملية تعديل أو **تحديث** لتلك القيـمة :

١- **عمـلية الحذف Delete :**

الفكـرة هـنا تصير بسيطةً وسهلة الفهم ، عندما ننظر الى مهمة المصمِّم فى هذا الخصوص على أنها محاولة منـه لإيجاد الإجابة الصحيحة للسـؤال التالى :

مـا الـذي يحدث إذا حاول المستخدم القيام بعملية حـذف لأحد السجلات الموجـودة فـى جـدول الهـدف The Target Table ؟ مثل حذف أحد سـجلات الموردين من جدول **الموردين** ؟ وهو كما أوضحنا من قبل يحـتوى على حقل مفتاح رئيسى ، هو حقل **تعريف-المورد** ، الذى يقابله حقـل خارجي مناظر فى جدول طلبيات الشراء.

وهـنا ، سـوف تكـون الإجابـة السليمة ، هى : جعل النظام يقوم باختيار **أحـد** التصـرفات الـثلاثة التـالية – دون سواها – وذلك بنـاءاً على تقييمـه **للوضع** الراهن البيانات :

- **القبول المقيد Restriction** : فيتم قبول العملية بشرط أن يتحقق النظام من عــدم وجــود أي ســجلات لطلبــيات تشير الى هذا المورد وتتضمن القيمة المطلوب حذفها فى حقل المفتاح الخارجي المقــابل بجدول طلبيات.

- **الرفــض Rejection** : أو يتم رفض تنفيذ العملية ، فى حالة عدم تحقق هذا الشرط ، ووجود طلب شراء واحد على الأقل يتضمن تلك القيمة.

- **التعــمــيم Cascades** : أو يــتم تنفــيذ عملية الحذف على سجل المورد ، وعلــى كافة السجلات المرتبطة به والمشار إليها فى حقل المفتاح الخارجي بجدول طلبيات الشراء.

٢- **عمــلية التحديث Updating** :

وبالمثــل ، تكون مهمة المصمِّم فى تعريف كيفية تطبيق قاعدة التكامل المرجعي أو الإشاري على محاولة المستخدم إجراء عملية تحديث البيانات ، أي تغيير القيم المســجلة فى حقل المفتاح الرئيسي بأحد جداول البيانات المرتبطة ، هــي البحث عن الإجابة الصحيحة للســؤال التالي :

مــا الــذى يحــدث إذا حاول المستخدم القيام **بعملية تحديث Updating** لإحــدى القيــم المسجلة فى حقل المفتاح الرئيسي بجدول معين مرتبط ؟

مــثــل : تعديل القيمة المسجلة فى حقل : **تعريف-المورد** وتخص سجل أحد الموردين ، بجدول **الموردين** ؟

مــرة أخــرى ، فــإن الإجابة السليمة سوف تكون – كما اتضح من مناقشة عملية الحــذف – هــي : جعل النظام يقوم باتخاذ أحــد التصرفات التاليــة ، فى ضوء تقييمه لحالة البيانات أو وضعها الراهن :

- **القبول المقيد Restriction** : حيث يتم قبول عملية التعديل بشرط أن يتحقق الـــنظام مـــن عـــدم وجود أي طلبيات تشير الى هذا المورد فى حقل المفتاح الخارجـــي المقـــابل بجدول طلبيات الشراء ، وبالتالى تتضمن القيمة الأصلية المطلوب تعديلها.

- **الـــرفـــض Rejection** : أو يتم رفض تنفيـــذ عملية التحديث القيمة ، وذلك فـــى حالـــة اكتشاف عدم تحقق هذا الشرط ، ووجود طلب شراء واحد على الأقـــل يتعلق بهذا المورد فى جدول طلبيات الشراء المرتبط ، ويتضمن تلك القيمـــة.

- **التعـــمـــيم Cascades** : أو يتم تنفيـــذ عملية التحديث المطلوبة على القيمة المسجلة بخصوص هذا المورد فى حقل المفتاح الرئيسي بجدول الموردين ، مـــع تنفـــيـــذ نفس التحديث على كافة القيم المقابلة لها التى تشـــير الى هذا المـــورد ذاته ضمن حقل المفتاح الخارجي بجدول طلبيات الشـــراء.

وهكـــذا يتضح من كل ما تقدم الى أي حد يرتبط مفهوم التكامل الإشاري بمفهوم حقول المفاتيح الخارجية. بل ويمكن القول بأن كل من مفهوم التكامل الإشاري أو مفهوم حقـــول المفـــتاح الخارجية ، إنما يعمل على تعريف المفهوم الآخر ويتكامل معـــه. فلا يمكن شرح ما هو المقصود بالمفتاح الخارجي بدون التطرق الي مفهوم التكامل الإشاري ... كمـــا لا يكون هناك محلا لتطبيق مفهوم التكامل الإشاري بدون وجود حقول المفاتيح الخارجية.

* * *