

قواعد البيانات (1)

تجميع وإعداد

دكتور عصام حلیم حسین

المحتوي

3 مقدمة
5 الجزء الأول: أساسيات قواعد البيانات
6 الفصل الأول: أساسيات قواعد البيانات
17 الفصل الثاني: برامج ولغات قواعد البيانات
31 الجزء الثاني: برنامج قواعد البيانات Microsoft Access
32 الفصل الثالث: مدخل إلى قواعد البيانات Microsoft Access
36 الفصل الرابع: تشغيل وإنهاء Access 2000
44 الفصل الخامس: إنشاء الجداول
59 الفصل السادس: إنشاء الاستعلامات واستخدامها Queries
77 الفصل السابع: تصميم النماذج واستخدامها (Forms)
83 الفصل الثامن: التقارير وبطاقات التسمية (Reports)
87 الفصل التاسع: الماكرو
89 الفصل العاشر: ربط الجداول
99 الجزء الثالث: لغة الإستعلام الهيكلية SQL

100 الفصل الحادي عشر: الاستعلام
118 الفصل الثاني عشر: العمليات علي المجموعات
124 الفصل الثالث عشر: جمل الاستعلام المتداخلة
132 الفصل الرابع عشر: دوال التجميع
147 الفصل الخامس عشر: الجداول
163 الفصل السادس عشر: المناظر
167 الفصل السابع عشر: ربط لغة الاستعلام الهيكلية مع لغات البرمجة
171 الجزء الرابع: ملحق رقم (1): لغة الإستعلام الهيكلية
187 تمارين

توصيف المقرر

أهداف المقرر: على اثر هذا المقرر يتمكن الطالب من: بعد انتهاء الطالب من دراسة هذا

المقرر ينبغي أن يكون ملما بما يلي:

يهدف هذا المقرر إلى تعريف الطالب بما يلي:

- المفاهيم الأساسية و التقنيات المتعلقة بنظم إدارة قواعد البيانات.
- الأساليب التقليدية و المتطورة لنمذجة البيانات.
- كيفية تصميم قواعد البيانات نظرياً (مفهوماً) و منطقياً (تطبيقاً).
- برنامج الاكسيس Access.
- لغة الاستعلام الهيكلية SQL.

اهداف المقرر:

- 1- المفاهيم الأساسية لقواعد البيانات.
- 2- عملية تطوير قاعدة البيانات.
- 3- النموذج العلائقي لقاعدة البيانات.
- 4- برنامج اكسيس Access.
- 5- نمذجة البيانات.
- 6- لغة الاستعلام الهيكلية SQL.

مقدمة

لقد دخل الحاسب الآلي في مجالات الحياة وظهر أثره في حل العديد من المشاكل التي تعاني منها الشعوب والأفراد. ومن تلك المشاكل القدرة على تخزين كم هائل من البيانات وما يترتب عليه من أسلوب حفظ واسترجاع وفهرسة تلك البيانات والتي تتطلب جهد وتكلفة تحتاج إلي وقت طويل وعلى سبيل المثال أسلوب حفظ بيانات الضباط بالقوات المسلحة والطلبة بالمعاهد والكليات العسكرية ومكتب تنسيق الكليات العسكرية ... الخ. ولقد كان الاعتماد حتى الآن على العنصر البشري فهو الذي يقع عليه العبء كله تقريباً وتتعدد مسؤوليته ابتداء من وضع استقبال البيانات ثم فهرستها لحفظها ناهيك عن العوامل الطبيعية التي تتعرض لها أوساط الحفظ الورقية وكما نرى نجد أن هذه العملية الروتينية تستغرق وقت طويل ويعتبر الوقت هو العامل الأساسي والحاسم لجميع الأعمال في هذا العصر.

قاعدة البيانات هي مجموعة من البيانات أو المعلومات التي يتم تنظيمها بحيث يسهل تحديد موقع معلومة محددة أو استرجاعها. فدفتر الهاتف يُعد مثلاً لقاعدة بيانات لم يتم تمثيلها على الكمبيوتر. يحتوي دفتر الهاتف على أسماء وعناوين وأرقام هواتف تم ترتيبها أبجدياً حسب اسم العائلة بحيث يسهل تحديد موقع معلومات عن شخص معين.

ويعتبر Microsoft Access نظام إدارة قواعد بيانات (DBMS) لإنشاء قواعد بيانات على الكمبيوتر واستخدامها. ونظام إدارة قواعد البيانات هو مجموعة متكاملة من البرامج يتم استخدامها لإنشاء معلومات في قاعدة بيانات وإدارتها. وتعتبر قواعد البيانات التي يتم إعدادها على الكمبيوتر أكثر فعالية من قواعد البيانات التي لا يتم إعدادها على الكمبيوتر (مثل دفتر الهاتف) حيث يمكن للمستخدمين إعادة تنظيم البيانات والبحث عن المعلومات بمئات الطرق. على سبيل المثال، إذا تم

تخزين دفتر هاتف كقاعدة بيانات في Access، سيمكنك البحث حسب العنوان أو الاسم الأول أو رقم الهاتف بدلاً من البحث حسب اسم العائلة فقط.

وكمثال آخر لتوضيح انواع قواعد البيانات كباقي اللغات المصممة للتعامل مع الحاسوب لغة SQL لها قواعد محددة. صممت لغة SQL لتمكن المستخدم من التفاعل المباشر مع الحاسوب. لذا فان بنية الأوامر في اللغة تمكن من استخدامها بشكل تفاعلي من اجل ايجاد إجابات سريعة عن الإستعلامات التي يحتاجها المستخدم. قامت شركة IBM باستخدام SQL في إنتاج عدد من نظم إدارة قواعد البيانات العلائقية مثل النظام System R والنظام System/38 والنظام SQL/DS وأخيراً النظام DB2. لكن النظام الذي حقق النجاح الأكبر هو النظام الذي اعتمده شركة Oracle وحمل أسماها والذي طرحته لأول مرة عام 1979.

الجزء الأول: أساسيات قواعد البيانات

الفصل الاول: اساسيات قواعد البيانات

مقدمة

سيتم في هذه الجزء وصف كيفية إعداد قواعد البيانات وكذا نظم إدارة قواعد البيانات (Database Management Systems DBMS) ومن المهم خلال قراءة لهذه المحاضرة الأخذ في الاعتبار أن غالبية المحاضرة تعالج قضية قواعد البيانات الخاصة بالبيانات الوصفية attribute data وليس البيانات الهندسية geometric data. ويمكننا تعريف قواعد البيانات كمجموعة مركبة (مهيكلة) structured collection من البيانات التي يمكن الدخول عليها (accessible) بطريقة منتظمة uniform way مما يجعلها عنصرا هاما في نظم المعلومات الجغرافية حيث يتم تنظيم البيانات باستخدام أنواع مختلفة من نظم إدارة قواعد البيانات DBMS .

▪ البيانات DATA :-

هي الأرقام أو الحروف أو الرموز أو الكلمات القابلة للمعالجة بواسطة الحاسب مثل: الرقم (65) أو كلمة بيانات.

▪ المعلومات Information :-

هي بيانات تم تنظيمها أو معالجتها لتحقيق أقصى استفادة منها.

مثال: الرقم (6) والرقم (5) إذا استخدمنا في عملية الضرب 6×5 أصبحا معلومة مفيدة.

▪ قواعدالبيانات Data base :-

هي تجميع لكمية كبيرة من المعلومات أو البيانات وعرضها بطريقة أو بأكثر من طريقة تسهل الاستفادة منها.

مثال : دليل الهاتف الذي يشتمل على أسماء وعناوين وأرقام هواتف سكان مدينة القاهرة يمكن أن نعتبره قاعدة بيانات وتحقيق الأستفاده من قاعدة البيانات هذه بإدخال رقم المشترك والحصول على أسمه وعنوانه أو إدخال أسم المشترك والحصول على رقم هاتفه وعنوانه وهكذا.

▪ نظم إدارة قواعد البيانات Database Management Systems :-

هي مجموعه من البرامج الجاهزة التي تقوم بتنفيذ جميع الوظائف المطلوبة من قاعدة البيانات.

مثال: بعد إضافة عملاء جدد لدليل الهاتف في مدينة القاهرة فإنك قد تحتاج لإعادة ترتيب أسماء المشتركين أبجدياً أو لترتيب عناوينهم، مثل هذا العمل من أحد وظائف إدارة قواعد البيانات.

▪ أهمية قواعد البيانات:-

أ- تخزين جميع البيانات بكافة الأنشطة لجهة ما بطرق متكاملة ودقيقه وتصنيف وتنظيم هذه البيانات بحيث يسهل استرجاعها في المستقبل.

ب- متابعة التغيرات التي تحدث في البيانات المخزنة وإدخال التعديلات اللازمة عليها، حتى تكون دائماً في الصورة الملائمة لاستخدامها فور طلبها.

ج- تخزين كم هائل من البيانات التي تتجاوز الإمكانيات البشرية في تذكر تفاصيلها ومن ثم إجراء بعض العمليات والمعالجات التي يستحيل تنفيذها يدوياً.

د- تساعد على تخزين البيانات بطريقه متكاملة، بمعنى الربط بين النوعيات المختلفة للبيانات المعبرة عن كافة الأنشطة.

هـ- تساعد على تحقيق السرية الكاملة للبيانات المخزنة بها بحيث لا تتاح أية معلومات لأي شخص ليس له الحق في الإطلاع عليها.

▪ وظائف قواعد البيانات:-

- أ- إضافة معلومة أو بيان جديد إلى الملف.
- ب- حذف البيانات القديمة والتي لم تعد هناك حاجة إليها.
- ج- تغيير بيانات موجودة تبعاً لمعلومات استحدثت .
- د- البحث والاستعلام عن معلومة أو معلومات محددة .
- هـ- ترتيب وتنظيم البيانات داخل الملفات .
- و- عرض البيانات في شكل تقارير أو نماذج منظمه .
- ز- حساب المجموع النهائي أو المجموع الفرعي أو المتوسط الحسابي لبيانات مطلوبة .

▪ أنواع قواعد البيانات:-

أ - من حيث الحجم:-

(1) مشروعات صغيرة:

(a) Access (b) Paradox (c) FoxPro (d) DBASEIII+/IV (e) R:BASE

(2) مشروعات كبيرة :

(a) Oracle. (b) SQL (c) DMS (Database Management System).

ب- من حيث طريقة العمل:-

(1) قواعد البيانات ذات شكل هرمي Hierarchy Databases

(2) قواعد بيانات شبكية Network Databases

(3) قواعد بيانات علائقية Relational Databases

▪ تنظيم قواعد البيانات داخل قاعدة البيانات:-

تخزن المعلومات المطلوبة لقواعد البيانات داخل (ملفات)، وتوضع هذه الملفات على أحد وسائط التخزين المساعدة مثل القرص المغناطيسي. وكل ملف عبارة عن جدول يشتمل على سطور وأعمده ، ويشتمل كل ملف على مجموعه من السجلات Records ويحتل كل سجل سطرًا داخل الملف ، ويقسم كل سجل إلى عدد من الحقول Fields .

-إذا أردنا إعداد دليل تلفونات لسكان مدينة المنيا، والملف المطلوب يشتمل على البيانات التالية:-

رقم المشترك	الاسم	العنوان	الهاتف
-1	حسين جمعه	الأبيض	4336754
-2	محمد حبيب	الابيض	2484354
-3	أحمد محمد مصطفى	الابيض	2505643

شكل (1) : يوضح كيفية تنظيم البيانات داخل جدول قاعدة البيانات.

هناك نوعين من نظم إدارة قواعد البيانات هما:

1- Hybrid systems ويتم فيها تخزين البيانات الهندسية (الإحداثيات) في قاعدة بيانات منفصلة عن قاعدة البيانات الوصفية وهي النوع الأكثر شيوعا في تطبيقات نظم المعلومات الجغرافية ويتم فيها ربط القاعدتين من خلال رقم منفرد (unique Id-number) يربط بين الأشكال الهندسية وبياناتها الوصفية وبعيدا عن هذا النوع من الربط الداخلي Internal linking فإنه من الممكن ربط هذا الرقم المنفرد مع قواعد بيانات خارجية من خلال الشبكات حيث قد يتم الربط في شبكة داخلية صغيرة Intranet أو شبكة المعلومات الدولية Internet.

2- Integrated systems ويتم فيها تخزين البيانات الهندسية والوصفية في نفس قاعدة البيانات ويمكن الربط مع مصادر البيانات الخارجية من خلال وسيط نظم إدارة قواعد البيانات RDBMS interface.

نموذج علاقات الكيانات ونظم إدارة قواعد البيانات Entity relationship (ER) model and DBMS

• تعتمد نمذجة قواعد البيانات على عدة مفاهيم وأكثر هذه المفاهيم استخداما هو نموذج علاقات الكيانات Entity relationship (ER) model وهو يعبر عن العلاقات بين الأشياء المطلوب عمل قاعدة بيانات لها أو بمعنى آخر يعبر عن هيكل قاعدة البيانات، وهو يتكون من ثلاثة عناصر تمثل رموز للأجزاء المختلفة التي تكون قاعدة البيانات:

1- نوع الكيان Entity type ويتحدد من خلاله نوع الشيء object الذي نتعامل معه.

2- نوع البيان الوصفي Attribute type وهو يصف أنواع الكيانات

3- نوع العلاقة relationship type وهو يحدد العلاقة بين الكيانات وبياناتها الوصفية.

• بالرغم من أن عملية نمذجة علاقات الكيانات قد تبدو جزءا سهلا من عملية نمذجة قواعد البيانات database modelling إلا أنها مجال هام جدا في عملية إنشاء قاعدة البيانات ويوضح الشكل رقم 8 مثلا صغيرا لنموذج علاقات الكيانات في قاعدة بيانات خاصة بإدارة شبكة الطرق حيث يظهر فيه عدد قليل من الكيانات التي يتم ربطها ببعضها من خلال بيانات وصفية معينة specific attributes ونلاحظ بالمثل أن كيان الطريق مرتبط ببيان وصفي رقم الطريق road number ومرتبب بكيان آخر هو رابط الطريق road link ويلاحظ أن درجة تعقيد النموذج تعتمد على مدى تعقيد قاعدة البيانات ، ومن خلال هذا النوع من نمذجة قواعد البيانات يمكن توضيح مدى القوة والضعف في طرق إنشاء قاعدة البيانات كما يمكن تحليلها من خلال الرسم.

• بعد الانتهاء من إعداد ال conceptual model يتم تنفيذ هيكل قاعدة البيانات داخل نظام

إدارة قواعد البيانات (البرنامج نفسه) وأثناء عمل ذلك لا بد من أخذ الآتي في الاعتبار:

1- مرونة عملية الإنشاء construction بدرجة كافية حتى يتم أخذ العمليات المختلفة في

نظام إدارة قواعد البيانات في الاعتبار.

2- هيكلية البيانات بما يسهل عملية استخراج (اشتقاق) البيانات.

3- يجب أن تراعي في عملية إنشاء قاعدة البيانات تقليل مخاطر الأخطاء داخل النظام فلا بد

ألا تعطى السماحية للمستخدم أن يدخل نوع خطأ من البيانات في جزء خطأ من النظام

كمثلا إدخال بيان نصي في مكان يقبل فقط البيانات الرقمية.

4- يجب تسهيل الدخول على قاعدة البيانات والتعامل معها من خلال إمكانيات البحث

الموجودة في نظام إدارة قواعد البيانات وهذا قد يشتمل على إنشاء واجهات interfaces

للمستخدمين الذين ليس لديهم مهارات في إدارة قواعد البيانات مما يصعب عليهم إستخراج واشتقاق البيانات.

• من المهم جدا وجود تعريف لفظي موحد standardized semantic للتأكد من إمكانية التواصل بين المستخدمين وكذا التأكد من ان الجميع يتحدث عن نفس الشيء فمعنى الطريق لسائق الحافلة أنه مكان يمكنه فيه قيادة الحافلة أما الطفل فقد يفكر في الطريق كمكان للعب كرة القدم بينما يفكر أشخاص آخرون في الطريق بشكل آخر ومن المهم جدا وجود مصطلح فني موحد (بإجماع الكل على تعريف محدد) لكل كيان في قاعدة البيانات التي نعمل عليها لتوفير إمكانية التواصل بين مستخدمي هذه القاعدة.

أنواع مختلفة من النماذج

• هناك أنواع مختلفة من هياكل قواعد البيانات database structures الشائع استخدامها وهي:

1- قواعد البيانات المتدرجة (هرميا) hierarchical databases

2- قواعد البيانات الشبكية network databases

3- قواعد البيانات المتصلة relational databases

أولا: قواعد البيانات المتدرجة hierarchical databases:

هيكل هذا النوع من قواعد البيانات يخلق شجرة بروابط بسيطة تمام تربط بين المستويات المختلفة ويسمح الهيكل بربط كيان واحد فقط من الأسفل بكيان واحد فقط من الأعلى ويوضح الشكل مثلا لهذا النوع من خلال قاعدة بيانات متدرجة لمكتبة بها مختلف الموضوعات والمؤلفين والكتب ومن

الواضح أنه يمكننا فقط ربط كتاب واحد بمؤلف واحد ومؤلف واحد بموضوع واحد مما يجعل قاعدة البيانات غير كفاء حيث يمكن أن يتم تأليف الكتاب من خلال أكثر من مؤلف كما يمكن أن يؤلف المؤلف عدة كتب في موضوعات مختلفة مما يجعل هذا النوع من قواعد البيانات غير شائع الاستخدام.

ثانيا: قواعد البيانات الشبكية network databases:

هيكل هذا النوع أكثر تعقيدا من سابقه حيث يمكن فيه ربط الكيانات المختلفة ببعضها بطريقة أكثر مرونة فالكتاب في قاعدة بيانات المكتبة يمكن ربطه بالعديد من المؤلفين ويمكن ربط المؤلفين بالعديد من الموضوعات وهكذا.

ثالثا: قواعد البيانات المتصلة relational databases:

هيكل هذا النوع أكثر مرونة من النوعين السابقين حيث يتم فيه تخزين الكيانات في جداول و يتم توصيف وربط (link/relate) الكيان (الجدول) بكيانات (جداول) أخرى في جداول أخرى ويوضح الشكل قاعدة بيانات المكتبة والتي يتم فيها تخزين كيان الموضوع subject كجدول به العديد من الموضوعات ثم يتم ربط هذه الموضوعات بجدول كيان المؤلف author والذي به قائمة من المؤلفين وهكذا

نموذج قاعدة البيانات المتصلة relational database model :

• هيكل قاعدة البيانات المتصلة هو أكثر الهياكل المستخدمة شيوعاً في برامج نظم المعلومات الجغرافية .

• يتم تنظيم هياكل قواعد البيانات المتصلة في جداول التي يتم تنظيمها بدورها في أعمدة تحوي معلومات مختلفة وأساسيات قواعد البيانات المتصلة هي أنه إذا احتوى عمود في جدول على نفس بيانات عمود آخر في جدول آخر فإنه يمكن ربط هذين الجدولين معاً والوصل بين المعلومات المخزنة بهما وهذا يسمى (حسب هياكل قواعد البيانات) العلاقة relation ويوضح الشكل ثلاثة جداول كل منها يحتوي على عمودين حيث يخزن الجدول الأول أسماء الأشخاص والمدن التي يعيشون فيها أما الثاني فيحتوي عموداً به أسماء المدن وآخر به أسماء الدول التي تقع هذه المدن داخلها وهذا التكرار للأعمدة التي تحتوي أسماء المدن يجعل من الممكن ربط الجداول المختلفة ببعضها أما الجدول الثالث فبه عمود يحتوي أسماء الدول وآخر به أسماء عواصم هذه الدول وهذا يجعل من الممكن ربط الجدول رقم 2 بالجدول رقم 3 لأن كل منهما به عمود يحتوي أسماء الدول المختلفة كما أنه يجعل ربط الجدولين 1 و 3 ممكناً لارتباط كل منهما بالجدول رقم 2 ، ويلاحظ أن استخدام المعلومات المتكررة في خلق علاقات بين الكيانات المختلفة في قواعد البيانات المتصلة هو نفس الشيء الحادث عند استخدام عمود Id-numbers في ربط البيانات الهندسية geometric data بالبيانات الوصفية attribute data.

• يتم التعامل مع قواعد البيانات المتصلة من خلال نظم إدارة قواعد البيانات المتصلة

• Relational Database Management Systems والتي يطلق عليها اختصارا RDBMS حيث يتم تنظيم البيانات في قاعدة البيانات في أعمدة وصفوف حيث لابد من تحديد عدد الأعمدة وكذا الصفوف ويمكن تسمية الأعمدة بأسماء fields, items, or tables مما قد يحدث ارتباك في المصطلحات أحيانا أما الصفوف فيمكن تسميتها objects, .post, records, or tuples.

• من المهم جدا عند خلق قاعدة بيانات جديدة تحديد هيكل بيانات الأعمدة fields المختلفة فمثلا لابد من تحديد نوع البيان المخزن وأي نوع من المعلومات ينبغي أن نسمح للمستخدم بإدخاله في العمود فمثلا إذا تم تعريف البيان كبيان نصي فإنه ينبغي أن نسمح للمستخدم بإدخال ASCII-text وتستخدم البيانات كبيانات نصية فقط بينما عند تعريفه كبيان رقمي يمكن للمستخدم تخزين أرقام وكذا استخدام العمود في الحسابات المختلفة ويلاحظ أن التعريف المحدد والحازم لكل عمود يقلل من الأخطاء الممكن حدوثها بسبب فصل أنواع البيانات data types المختلفة ومنع خلطها ببعضها (هذا فرق هام جدا بين نظم إدارة قواعد البيانات وبرامج spreadsheet مثل MS Excel التي يمكن فيها خلط أنواع البيانات المختلفة في نفس العمود) ، ومن الضروري أيضا تحديد حجم العمود بتحديد عدد الأحرف (أو الأرقام) characters المسموح بتخزينها فيه فمثلا قد يسمح بتخزين 50 حرفا في الأعمدة النصية text columns بينما في الأعمدة الرقمية فيمكن تخزين 30 وحدة رقمية digit مع 10 وحدات عشرية decimal ، كما يلزم أيضا تحديد نوع التخزين للقيم المختلفة (في الأعمدة الرقمية) كمثلا أن يتم تخزينها على هيئة binary, integer, or real.

• يتم هيكلية البيانات في جداول يتم ربطها معا وهذا الربط يخلق علاقات بين البيانات الوصفية المختلفة وحتى إذا لم يتم ربط جدولين معا بطريقة مباشرة فمن الممكن أن يكون بينهما علاقة

من خلال سلسلة من الجداول المرتبطة ببعضها مما يجعل من السهل دمج معلومات من الجدولين ولو نظرنا للارتباط بطريقة أعمق لوجدنا أن كل الأشياء وكذا الأدميين يرتبطون بطريقة أو بأخرى معا كما يوضح الشكل بأعلى.

- عند بناء الجداول في قواعد البيانات المتصلة يتم أخذ بعض المحددات في الاعتبار فمن المهم أن نخزن قيمة واحدة في كل خلية من خلايا الجدول حيث أنه ليس من الممكن مثلا تخزين عمر وعدد الناس في نفس الخلية كما لا يمكن مثلا تخزين العمر والإسم لنفس الشخص في نفس الخلية فكل خلية لابد أن تحوي قيمة منفردة unique تعبر عن صفة معينة للكيان entity المعبر عنه بالجدول.

- لابد أيضا من تلافي الاعتماد الوظيفي functional dependence ومعنى هذا أن نتلافى أن تكون قيم عمود معين بالجدول تعتمد على قيم عمود آخر سواء كان في نفس الجدول أو في جدول آخر (يتم تحويل القيم في العمود الأصلي من خلال خوارزم algorithm معين وتخزينها في عمود آخر يسمى calculated field وهذا يرفع المساحة المستخدمة من الذاكرة وبالتالي فهو طريقة غير كفاء حيث يمكن حساب هذه القيم بشكل مؤقت بدلا من تحميلها في الذاكرة وتخزينها فيها بشكل دائم).

- لابد أيضا من تلافي التكرار redundancy بقدر الإمكان عند إنشاء قاعدة البيانات وهذا التكرار يعني تخزين نفس المعلومات عدة مرات مما يجعل قاعدة البيانات أضخم وبالتالي أبطأ في استخراج البيانات منها وفي التعامل معها بشكل عام وعادة يتم تلافي هذا التكرار بتقسيم الجداول الضخمة لعدد من الجداول الصغيرة التي يتم ربطها من خلال الأعمدة المتكررة common columns (التي تحدثنا عنها في مثال الدول والعواصم) ويتضح هذا من خلال المثال الموجود بالشكل العلوي.

• استخدام الفهارس (مثلما يحدث في دليل التليفونات) سيزيد سرعة البحث في قاعدة البيانات حيث يمكن فهرسة البيانات (على سبيل المثال) أبجديا وهذه الفهرسة ستقلل وقت استخراج واشتقاق البيانات كما يمكن استخدام الفهارس نفسها في الربط بين الجداول المختلفة ومن هنا فإن الفهارس تجعل التعامل مع قاعدة البيانات كقوة بشكل أكبر لأنها تمنع التكرار duplication وتزيد من سرعة البحث وتحفظ سلامة المرجعية referential integrity لإمكانية استخدامها في الربط بين الجداول.

الربط والضم link and joins:

• تتعدد طرق ربط البيانات ومنطقيا لإغن العلاقة نوع من ثلاثة:

1- واحد لواحد one to one.

2- واحد لمتعدد one to many.

3- متعدد لمتعدد many to many.

أولاً: واحد لواحد one to one:

يربط هذا النوع من العلاقات بين الجداول عن طريق قيمة منفردة تظهر مرة واحدة في كل جدول والمثال يوضح وجود عمود اسمه "order" في كل جدول ويلاحظ أن رقم الأمر order number قيمة منفردة تظهر في الجدول مرة واحدة كما يلاحظ أن العلاقة تربط صف واحد في أحد الجدولين بصف واحد في الجدول الآخر.

ثانياً: واحد لمتعدد one to many:

يحدث هذا النوع من العلاقات عندما يمكن ربط صف واحد في أحد الجداول بعدة صفوف في جدول آخر وهذه الطريقة هي طريقة نموذجية لتلافي التكرار redundancy في قاعدة البيانات

ويوضح المثال جدولين الأول يحتوي عمود بأسماء الدول وعمود بأرقامها أما الجدول الثاني يحتوي عمود بأرقام الدول وآخر بالمدن المختلفة الموجودة في هذه الدول وحيث أن الدولة بها عدة مدن فإن رقمها يمكن تطبيقه على عدة مدن داخلها (عدة صفوف من الجدول تلائم دولة واحدة) من خلال جدول أسماء المدن، من هنا وبالنظر لقاعدة بيانات أكثر تعقيدا فإن الجدولين سيحتويان على معلومات أكثر يجب تخزينها بالطريقة الموضحة حيث سيحوي الجدول الأول معلومات (أعمدة) مثل عدد السكان والمساحة والعواصم الخاصة بالدول الموجودة به أما الجدول الآخر فإنه معلومات عن عدد السكان والمساحة ووصلات السكك الحديدية الخاصة بالمدن الموجودة به.

مما سبق فإن تقسيم الجدول باستخدام علاقة واحد لمتعدد one to many أفضل من تخزين كل شيء في نفس الجدول لأننا نكون قد تلافينا التكرار redundancy.

ثالثا: واحد لمتعدد many to many

هذا النوع من العلاقات لا يوجد به أي قيم منفردة في أي من أعمدة الجداول ويوضح المثال جدولين بهما معلومات عن حالة الطقس تم اقتباسها من محطة أرصاد ويحوي كل جدول رقم المحطة وسنة الرصد وشهر الرصد ثم معدلات تساقط الأمطار في أحد الجدولين ودرجات الحرارة في الآخر وفي هين الجدولين لا يوجد مؤشرات منفردة unique identifier وللربط بين القيم المنفردة (على سبيل المثال لنفس الشهر) لابد من استخدام عدة أعمدة (رقم المحطة وسنة الرصد وشهر الرصد).

- غالبا ما توجد لدينا مشكلة البيانات المفقودة missing data مما يسبب مشكلة عند التعامل مع البيانات الوصفية وعلاج ذلك هو ضم الجداول أو ربطها لمعرفة القيم المفقودة وعندما نريد ضم جدولين أو ربطهما فإن ذلك يمكن أن يتم من خلال إحدى طريقتين:

1- الضم الداخلي inner join.

2- الضم الخارجي outer join.

أولاً: الضم الداخلي inner join

عند ضم جدولين معا باستخدام هذا الأسلوب فإنه يمكن فقط ربط link الصفوف التي يمكن ربطها من خلال علاقة واحد لواحد وبالتالي فهو يأخذ فقط الصفوف التي بها قيم في كلا الجدولين ويتم حذف كل الصفوف الباقية من قاعدة البيانات الناتجة وهذه الطريقة تستخدم لحذف البيانات التي ليس لها صلة بموضوع الدراسة عند دمج قواعد البيانات ويلاحظ في المثال أننا نفقد بيانات من الجدول الأول (1 و 2) وبيانات من الجدول الثاني (6 و 7) لكن تصبح بعد ذلك قاعدة البيانات ذات هيكل مدروس.

ثانياً: الضم الخارجي outer join

عند ضم جدولين معا باستخدام هذا الأسلوب فإنه يتم الاحتفاظ بكل الصفوف من أحد الجدولين ثم إضافة الصفوف المتوافقة معها corresponding rows من الجدول الآخر في المثال تم أخذ كل الصفوف من الجدول الأول A ثم دمجها مع الصفوف المتوافقة من الجدول الآخر B وبالتالي فإن الصفوف الموجودة في الجدول الأول وليس لها صفوف متوافقة في الجدول الثاني سيظهر فيها بيانات مفقودة في الجزء الذي مصدره الجدول الثاني B (في هذه الحالة نفقد فقط القيم 6 و 7 من الجدول الثاني) ويلاحظ أنه طبقاً للغرض من قاعدة البيانات يمكننا أن نحدد نوع الضم الذي نستخدمه (داخلي/ خارجي) وإذا كان للجدولين نفس الصفوف (نفس المؤشرات بما يعني أن العمود No. به نفس القيم في الجدولين) فإن مشكلة البيانات المفقودة لن تحدث.

التنقيب عن البيانات أو كتالوج البيانات metadata:

• يمكن تعريف ال metadata بأنها المعلومات المخزنة عن البيانات (الموجودة في قاعدة البيانات) سواء كانت بيانات هندسية أو بيانات وصفية كما يمكن تعريفها بأنها معلومات توثيقية عن البيانات أو وصف أكثر تفصيلا للبيانات وتعد ال metadata موضوعا هاما جدا لأن توثيق قاعدة البيانات هام جدا لتعزيز المعايير المحددة الموحدة sustain certain standards لقاعدة البيانات وبالتالي فهي تضمن مستوى معين من الجودة والمرونة وكذا مستوى معين من إمكانية الاعتماد على قاعدة البيانات من خلال مستخدميها.

• أغلب الدول إما أن يكون لديها معايير محددة للبيانات التوثيقية metadata أو في طريقها لوضع هذه المعايير والأشكال التالية توضح أمثلة لبعض المعايير المقترنة من المعيار الأوروبي لل metadata:

1- إسم قاعدة البيانات واسم مالكها.

2- نظرة ووصف عام للبيانات الهندسية والبيانات الوصفية وكذا الاستخدام المقترح (المناسب) لقاعدة البيانات.

• ينبغي أن تحتوى البيانات التوثيقية metadata على المعلومات الخاصة بعلم الخرائط كالإسقاط projection ونظام الإحداثيات المستخدم coordinate system وكذا استدلال الخريطة datum.... إلخ وإذا لم تتوفر هذه المعلومات فمن الصعب معرفة نوع الإسقاط الذي نستخدمه للتعامل مع الخريطة (إذا تم اختيار إسقاط خاطئ فإن ذلك يحدث تشويها للخريطة) كما يصعب تحديد كيفية تحويل قواعد البيانات بين نظم الإحداثيات المختلفة.

• من المهم أيضا توثيق البيانات الخاصة بهياكل قواعد البيانات database structures المخزنة (سواء بيانات هندسية أو وصفية) وكذا المصطلحات الفنية وهيكل الملف file structure وكيفية تنسيق البيانات data format وذلك لتسهيل تبادل البيانات بين النظم المختلفة (سواء كانت برامج أو معايير standards).

• يجب أيضا توثيق البيانات بالامتداد المستقبلي المخطط له (بالنسبة للبيانات) بالإضافة إلى الامتداد الواقعي الذي يمثل كل من الامتداد المكاني (الإحداثيات والوحدات الإدارية... إلخ) والامتداد الزمني وبالتالي لابد أن ن فكر هل البيانات ملائمة للاستخدام (حاليا/ بعد شهرين/ بعد 6 أشهر فمثلا بعض قواعد البيانات مثل بيانات التعداد السكاني قد تكون صالحة لفترة محدودة مما يوجب علينا توضيح ذلك في البيانات التوثيقية metadata) وهل سيتم جمع بيانات أخرى مستقبلا وكذا ما هي تواريخ جمع البيانات وما هي فترة التحديث الدوري لها وهذه المعلومات ستساعد المستخدم لتحديد مدى ملائمة البيانات للاستخدام كما أنها فرصة ليعلم المستخدم عن الامتدادات المستقبلية لقاعدة البيانات.

• يمكننا اعتبار البيانات التوثيقية metadata كإفادات للجودة والتي تعد مكونا هاما من مكونات البيانات التوثيقية التي تحتوي على:

1- أصل قاعدة البيانات مع وصف لمصدر البيانات وعملية تطور القاعدة (معلومات عن المكان الذي استخرجت منه البيانات وكيف تم معالجتها processed قبل إدخالها لقاعدة البيانات فمثلا هل تم فحصها أو إجراء تحليلات إحصائية عليها).

2- الدقة الهندسية (منهجية تجميع البيانات الهندسية وما هي الدقة المستهدفة إذا تم جمع البيانات من خلال أجهزة مساحية متقدمة فإنها تكون أكثر دقة من البيانات الهندسية المستخرجة من الصور الجوية) والتي تؤثر على دقة الإحداثيات (إحداثي النقطة على

الخريطة يختلف عن إحداثيها في الطبيعة بمسافة 1 متر أم 10 متر) ويستخدم لقياس هذه

الدقة في بعض التطبيقات ما يطلق عليه (Root Mean Square (RMS).

3- دقة البيانات الوصفية (كما بالبيانات الهندسية فإن منهج جمع البيانات الوصفية سيؤثر على

الدقة) وهل تم جمع كل البيانات من الطبيعة بقياسات حقيقية أم تم تقديرها أو عمل

interpolation لها.

4- الدقة الزمنية لقاعدة البيانات (هل هي صالحة للاستخدام خلال فترة معينة وما هي هذه

الفترة هل هي شهر أم سنة... إلخ).

5- Logical consistency (كيف تم وضع البيانات معا وهل كل كيان متصل بالآخر وهل

كل كيان له علاقة بالآخرين) وهو ما يفصل طريقة إنتاج قاعدة البيانات والعمليات الفرعية

المستخدمة في ذلك ومن المهم أيضا في هذا العنصر تحديد نوع ال topology

(spaghetti topology of full polygon topology).

6- الكمال (التمام) completeness وتحدد ما إذا كانت كل المواصفات السابقة صحيحة

للمساحة الجغرافية (المرصودة في قاعدة البيانات) كلها أم هناك اختلافات في الجودة أو

الدقة الزمنية بين المناطق الجزئية وبعضها فقد تكون دقة الرسم في إحدى المناطق الجزئية

ممتازة ولكنها أقرب للكروكي في منطقة جزئية أخرى .

• ينبغي أن تحتوي البيانات التوثيقية على البيانات الإدارية الخاصة بقاعدة البيانات وذلك كخدمة

لمستخدمي قاعدة البيانات المحتملين في المستقبل لأن قاعدة البيانات يتم بيعها في أغلب

الحالات ولا يتم إعطاءها مجانا وبالتالي فإن هذه البيانات الإدارية تكون بمثابة الدعاية و

الإعلان للعملاء المحتملين في المستقبل وتساعدهم في الإستعلام عن البيانات وكذا في

كيفية طلب شراء هذه البيانات كما يتحدد من خلالها التنسيق format الذي يتم تسليم

البيانات على صورته وكذا الوسيلة media (عن طريق البريد الإلكتروني e-mail أو CD-ROM or diskette) وكذا الخدمات الإضافية التي يمكن لمنتج البيانات القيام بها مثل إمكانية عمل التحليلات المختلفة على هذه البيانات.

- هناك أيضا ما يسمى البيانات التوثيقية الخاصة بالبيانات التوثيقية metadata about metadata وهي هامة نظرا لأنه من المحتمل أن يكون التحديث مستمرا لقاعدة البيانات مما يظهر الحاجة لتحديث البيانات التوثيقية الخاصة بقاعدة البيانات لكننا لا بد أن نأخذ في اعتبارنا أن المهمة الرئيسية ليست هي عمل البيانات التوثيقية وأن مدى شمولية البيانات التوثيقية يجب أن تكون منطقية ليس مبالغا فيها حتى نمنع العاملين من الوصول لمرحلة إنتاج البيانات التوثيقية بدلا من قواعد البيانات والتي هي أساس العمل....التوازن مطلوب بين كل منهما.

الفصل الثاني: قواعد البيانات في سطور

تعتبر البيانات عن نصوص أو أرقام أو صور وفي بعض الأحوال أصوات، حيث يتم معالجة كل من هذه العناصر أو تخزينها في الحاسوب. ومما يذكر أن البيانات في حد ذاتها قد تكون بلا معنى. لذلك، سوف نحتاج إلى تفسيرها أو (معالجتها) لكي تصبح معلومات نستطيع فهمها وإدراكها.

لذلك، تعتبر المعلومات بيانات ذات مغزى معين.

ولكي نوضح الفرق بين مفهومي البيانات والمعلومات، مثلاً تشير كلمات مثل، خدمة العملاء و15000 و10 إلى بيانات. لكن إذا أمكن تفسيرها على أنها بيانات خاصة بشركة معينة، فقد تشير البيانات مثلاً إلى أن قسم خدمة العملاء به 10 موظفين ومتوسط الدخل السنوي للموظف هو 15000 دولار. عندئذ تصبح هذه البيانات معلومات. بطبيعة الحال، يمكن تفسير نفس هذه البيانات بشكل آخر مختلف.

من الجدير بالذكر أن البيانات تعتبر بمثابة المكونات الأساسية التي يقوم عليها أي عمل، وعندما يتم معالجتها ودمجها مع بعضها باستخدام تقنيات متعددة تظهر نتائج مختلفة.

أبسط تعريف لقاعدة البيانات هو:

مجموعة من البيانات المرتبطة مع بعضها والتي تختص بموضوع أو أكثر.

فيما يلي عدد من الأمثلة التي يمكن أن تشير إلى قاعدة بيانات:

- دليل الهاتف
- جدول مواعيد القطارات
- دفتر العناوين
- معلومات عن العملاء في شركة معينة

ما هي
البيانات؟

ما هي قاعدة
البيانات؟

الملف: مجموعة من السجلات المتعلقة بنفس الموضوع	مصطلحات أساسية
قائمة العملاء	
قائمة المنتجات	
مواعيد القطارات	
السجل: المعلومات الخاصة بعنصر معين	
العميل	
المنتج	
القطار	
الحقل: عناصر البيانات الفردية الخاصة بكل سجل	
العميل – الاسم والعمل والعنوان ورقم الهاتف	
المنتج – وصف المنتج وسعر التكلفة وسعر البيع	
القطار – الجهة والسائق وعدد المقاعد ونوع عربات القطار	
كذلك، يتم تحديد الحقول من خلال حجم ونوع المعلومات التي تحتوي عليها.	
مما يذكر أن قواعد البيانات تأتي في شكلين أساسيين:	
تستخدم قاعدة البيانات غير المفهرسة في حالة القوائم البسيطة، وقد تكون البيانات مكررة داخل قاعدة البيانات.	قاعدة غير المفهرسة
في قاعدة البيانات العلائقية، يتم وضع البيانات في عدد من الملفات المرتبطة مع بعضها وذلك للحد من تكرار البيانات. تتميز عمليات البحث والتحليل والتأمين للبيانات التي تتم من خلال قاعدة البيانات العلائقية بالسهولة واليسر.	قواعد البيانات العلائقية
افتراض مثلاً وجود شركة بها موظفين وعملاء وتقدم منتجات أو خدمات. ويجب على العميل أن يسجل طلب شراء للحصول على هذه المنتجات أو	

الخدمات.

قمنا بتحديد خمس ملفات أو جداول مختلفة هي:

- الموظف
- العميل
- المنتج
- طلب الشراء
- تفاصيل طلب الشراء

بطبيعة الحال، هناك طلبات شراء لأكثر من منتج. لذلك، فسوف نحتاج إلى جدول مرتبط من أجل تفاصيل طلب الشراء.

الجدول المحددة بأعلى لها علاقات ببعضها البعض:

- الموظف يبيع للعميل.
- العميل يطلب شراء منتج.
- يحتوي طلب الشراء على تفاصيل الشراء.

أنواع

العلاقات

عندما يقوم أحد العملاء بتقديم طلب للشراء، قد يكون هذا الطلب موجهاً لعدد من المنتجات. لمزيد من التوضيح، قد يكون هناك عدد من المنتجات في طلب شراء واحد. يعرف ذلك باسم علاقة جزء بمجموعة أجزاء وهي العلاقة الأكثر شيوعاً. هناك كذلك علاقة جزء بجزء الأقل شيوعاً. أما علاقة مجموعة أجزاء بمجموعة أجزاء، فغير مسموح بها في أكسيس: حيث يتم معالجتها كعلاقة جزء بمجموعة أجزاء من خلال إعداد جداول مرتبطة إضافية. ويوضح الشكل أدناه هذه العلاقات.

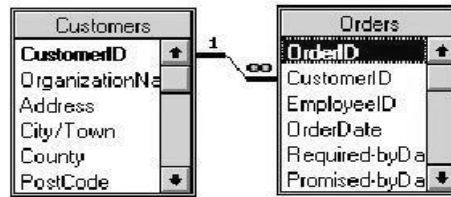
لاحظ رمز 1 الذي يشير إلى الجانب الفردي من العلاقة ورمز ∞ الذي يشير إلى الجانب المتعدد.

في مثال علاقة جزء بمجموعة أجزاء، يعتبر حقل CustomerID هو الذي يربط بين الجدولين. في جدول العملاء، يعمل هذا الحقل كمفتاح أساسي

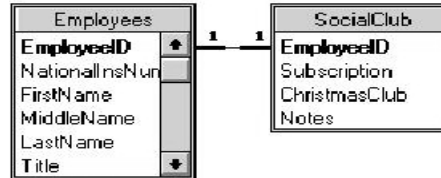
(primary key). إذ يقوم المفتاح الأساسي بتحديد كل سجل في الجدول، وفي بعض الحالات، قد يكون المفتاح الأساسي عبارة عن حقل واحد (مثل CustomerID) أو قد يكون مجموعة من الحقول (مثل OrderID + LineNo الذي سيحدّد سجلاً في جدول طلب الشراء). وبمعرفتك للبيانات الموجودة في الجدول، يمكنك تحديد الحقل أو مجموعة الحقول التي ستحدّد السجلات الموجودة في هذا الجدول.

في جدول طلبات الشراء، يعمل CustomerID كمفتاح خارجي (foreign key) يرتبط بالجدول الأساسي في العلاقة وهو هنا جدول العميل.

علاقة جزء بمجموعة أجزاء



علاقة جزء بجزء



علاقة مجموعة أجزاء ببعضها



لماذا نستخدم قواعد البيانات الإلكترونية بتنفيذ المهام نفسها التي تتم في قواعد البيانات التقليدية ولكن مع المميزات التالية:

- السرعة الفائقة

الإلكترونية؟

- سهولة الاستخدام
 - تخزين قدر هائل من البيانات
 - السماح بإدخال وتحرير البيانات بسهولة شديدة
 - التحديث التلقائي وإعادة حساب البيانات
 - السماح بفرز البيانات في سرعة وسهولة
 - السماح بالبحث عن البيانات وتحديدها
 - تنسيق وتنظيم وتقديم البيانات بالطريقة التي تفضلها
 - مشاركة المعلومات مع برامج / تطبيقات أخرى
- وعلى شبكات الاتصال، تسمح لك قواعد البيانات بمشاركة مجموعة واحدة من المعلومات مع العديد من المستخدمين وهو ما يقلل من تكرار البيانات. الإقلال من عملية تكرار البيانات وبالتالي معالجة مشكلة الاحتفاظ بعدد من النسخ المحدثة لقاعدة البيانات نفسها.

لماذا

نستخدم

قاعدة

بيانات

أكسيس؟

تعتبر قاعدة بيانات أكسيس بمثابة نظام لإدارة قواعد البيانات العلانية وتشكل مع باقي تطبيقات مايكروسوفت مثل : وورد وإكسيل وباوربوينت والبريد الإلكتروني ومجموعة برامج مايكروسوفت أوفيس بروفيشنل (Microsoft Office Professional).

تقدم قاعدة بيانات أكسيس طريقة فعالة لتخزين وفرز ومعالجة واستعادة البيانات. تم تطوير أكسيس بواسطة مايكروسوفت، الشركة نفسها التي طورت نظام التشغيل ويندوز.

تجدر الإشارة إلى أن أهم سمات أكسيس تكمن في سهولة الاستخدام بالنسبة للمبتدئين حيث يمكنك باستخدام عمليات بسيطة تطوير قواعد البيانات في سهولة كبيرة. كما تُستخدم قواعد بيانات أكسيس لتطوير نظم التطبيق المتقدمة.

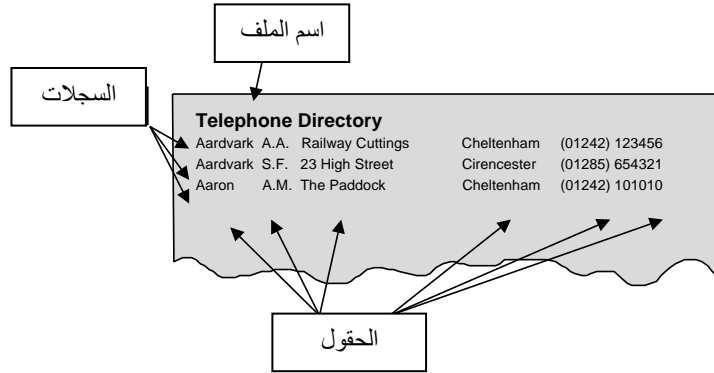
يسمح برنامج أكسيس أيضًا بقدر كبير من إمكانية تعديل الحقول والجداول بعد إدخال البيانات.

تستخدم قاعدة بيانات أكسيس إلى حد بعيد المصطلحات نفسها التي سبق لنا توضيحها من قبل. وذلك باستثناء استخدام مصطلح الجداول بدلاً من الملفات، وهذا الأمر مناسب حيث إن معظم المعلومات التي تستخدمها تكون بطبيعتها بصورة جداول.

مقدمة لقاعدة

بيانات

أكسيس



Customer ID	Company Name	Contact Name
ALFKI	Alfreds Futterkiste	Maria Anders
ANATR	Ana Trujillo Emparedados y helados	Ana Trujillo
ANTON	Antonio Moreno Taquería	Antonio Moreno
AROUT	Around the Horn	Thomas Hardy
BERGS	Berglunds snabbkp	Christina Berglund
BLAUS	Blauer See Delikatessen	Hanna Moos
BLONP	Blondel père et fils	Frédérique Citeaux
BOLID	Blido Comidas preparadas	Martin Sommer
BONAP	Bon app'	Laurence Lebihan
BOTTM	Bottom-Dollar Markets	Elizabeth Lincoln
BSBEV	B's Beverages	Victoria Ashworth
CACTU	Cactus Comidas para llevar	Patricio Simpson
CENTC	Centro comercial Moctezuma	Francisco Chang
CHOPS	Chop suey Chinese	Yana Wong

يطلق على طريقة عرض هذا الجدول اسم ورقة البيانات، وهي عادةً الطريقة التي يعرض من خلالها برنامج أكسيس البيانات. بطبيعة الحال، قد تكون قاعدة البيانات صغيرة الحجم جدًا مثل دفتر عناوين لأحد الأشخاص، وقد تكون ضخمة تحتوي على كميات هائلة من البيانات التي تختص بعمليات وأنشطة تجارية كبيرة جدًا.

<p>عند العمل في قاعدة بيانات، سيطلب منك القيام بأكثر من مجرد عرض البيانات. ولكي تقوم بمعالجة البيانات وتقديمها على شكل معلومات، تحتاج لمجموعة من الأدوات أو الوسائل.</p> <p>يمكنك أن تستخدم النموذج في إدخال أو تحرير أو عرض المعلومات في سجلات قاعدة البيانات.</p>	<p>تقديم أدوات أكسيس ما هي النماذج؟</p>
<p>يسمح لك كذلك بعرض ما ترغب فيه بالطريقة المفضلة لديك.</p> <p>وبذلك، يمكنك إنشاء النماذج لكي تؤدي عمل المستندات الورقية.</p> <p>الاستعلام عبارة عن سؤال خاص بالبيانات، مثل:</p> <p>ما عدد الحسابات المتأخرة؟</p> <p>ما عدد عملاء الشركة في مملكة البحرين؟</p> <p>عندما تقوم بالاستعلام من قاعدة البيانات، ستحصل على أحدث المعلومات المتوفرة.</p>	<p>ما هو الاستعلام؟</p>
<p>يتم استخدام التقرير في طباعة وعرض المعلومات الموجودة في قاعدة البيانات.</p> <p>تسمح لك التقارير بتقديم معلوماتك بالطريقة التي ترغب فيها وتمكنك من:</p> <p>تجميع المعلومات</p> <p>حساب الإجماليات والمتوسطات</p>	<p>ما هي التقارير؟</p>

الجزء الثاني:

برنامج قواعد البيانات

Microsoft Access

الفصل الثالث: مدخل إلى قواعد البيانات Microsoft Access

قواعد البيانات Database : هي عبارة عن تجميع لكمية كبيرة من المعلومات أو البيانات وعرضها بطريقة أو أكثر من طريقة لتسهيل الاستفادة منها .

وتشترك معظم نظم إدارة قواعد البيانات في مجموعة من الوظائف منها :

- أ. إضافة معلومة أو بيان إلى الملف .
- ب. حذف البيانات القديمة .
- ج. تغيير البيانات الموجودة .
- د. ترتيب وتنظيم البيانات داخل الملفات .
- هـ. عرض البيانات على شكل تقرير أو نموذج .

هذا ويعتبر برنامج Microsoft Access واحد من أشهر قواعد البيانات والتي تستخدم في

ترتيب قواعد البيانات واستخراج النتائج منها وعمل الإستعلامات اللازمة .

وهو عبارة عن برنامج رسومي يعمل تحت بيئة Windows الرسومية . ويحتوي هذا البرنامج

على مجموعة متنوعة من الكائنات التي يمكن استخدامها لعرض المعلومات وإدارتها مثل

الجدول والنماذج والتقارير والاستعلامات ووحدات لماكرو ووحدات نمطية وصفحات وصول

للبيانات .

من مميزات هذه القواعد :

1- جمع جميع كائنات القاعدة في ملف واحد يأخذ الامتداد MDB ، وهذا ولاشك أسهل في التعامل مع القاعدة وإن كان قد يمثل خطورة على القاعدة من جهة أن تلف هذا الملف يتلف معه كل كائنات القاعدة .

2- استيراد وتصدير أنواع مختلفة من البيانات إلى برامج مجموعة الأوفس أو إلى قواعد وبرامج أخرى .

3- تعدد درجات الأمان في القاعدة وتعدد المستخدمين .

4- إمكانية وضع القاعدة على شبكة اتصالات داخلية وتشغيلها من عدة مستخدمين في آن واحد .

5- وجود خصائص وطرق تمكن المستخدم من التحكم الكامل في القاعدة وبياناتها ومنع تغيير تصميمها .

يطلق على قواعد بيانات ميكروسوفت أكسس اسم قواعد البيانات العلائقية ويقصد بها قواعد البيانات التي تكون الجداول فيها مترابطة بينها بعلاقات في حقل واحد أو أكثر. والهدف الأساسي من ربط الجداول هو منع تكرار البيانات والحد من مساحات التخزين الضائعة والرفع من كفاءة قاعدة البيانات . وسيتم تفصيل أنواع العلاقات وكيفية الربط بين الجداول في قسم العلاقات .

وقد وضعت ميكروسوفت في هذا البرنامج كائنات تساعد المستخدم لإدخال البيانات واستخراجها من القاعدة وطباعتها ، وهذه الكائنات هي :

(1) الجداول : وهي مكان تخزين البيانات في القاعدة ، وتتكون الجداول من حقول (أعمدة) وسجلات (صفوف) .

(2) استعلامات : وهي كما يتضح من اسمها استعلام عن بيانات معينة في القاعدة تنطبق عليها معايير محددة ، أو كائنات لتنفيذ عمليات على البيانات في الجداول كحذف سجلات أو تحديثها أو إنشاء الجداول أو إلحاق سجلات بها .

(3) النماذج : وهي مكان تسجيل البيانات التي ترغب في حفظها في الجدول ، وتحريرها .

(4) التقارير : وهي كائنات عرض وطباعة البيانات بأشكال وطرق وتنسيقات متنوعة .

(5) الصفحات : وهي صفحات تعرض البيانات في ملفات من نوع HTML منفصلة عن ملف القاعدة الأساسي وذلك لعرضها على شبكة الانترنت .

(6) الماكرو : أبسط تعريف له هو كائن يمكن وضع أمر أو عدة أوامر أو إجراءات فيه ليتم تنفيذها .

(7) الوحدات النمطية : هي مكان تخزين أوامر وإجراءات ليتم تنفيذها أو استدعاؤها بأكثر من طريقة وتختلف عن الماكرو بإمكانية التحكم في هذه الأوامر بشكل أكبر وأنها ذات إمكانيات أوسع وأكبر وأدق وتحكم أكثر فيها .

ميكروسوفت أكسس Microsoft Access هو برنامج لإنشاء وتصميم قواعد بيانات يمكنك هذه القواعد من :

- 1- تسجيل أسماء أشخاص أو وجهات وعناوينهم وأرقام هواتفهم .
- 2- تسجيل مبيعات ومشتريات واستخراج فواتير متنوعة .
- 3- تسجيل بيانات ودرجات طلاب واستخراج نتائجهم .

- 4- مرضى وبياناتهم الشخصية وإحصاءات متنوعة لهم .
- 5- فهارس كتب ومكتبات وإعارات .
- 6- عاملين في المؤسسة وتقارير بالمستحقات والإجازات .
- 7- اتصالات إدارية (صادر ووارد) .
- 8- مكاتب سفريات وحجوزات .
- 9- تسجيل تبرعات ومصروفات وأنشطة خيرية .
- 10- فهارس مكتبات صوتية (أشرطة صوتية) .

الفصل الرابع: تشغيل وإنهاء Access 2000

- تشغيل Access 2000: من قائمة البرامج :-

1. من سطح المكتب انقر على زر Start الموجود في أسفل الشاشة على اليسار في شريط المهام .

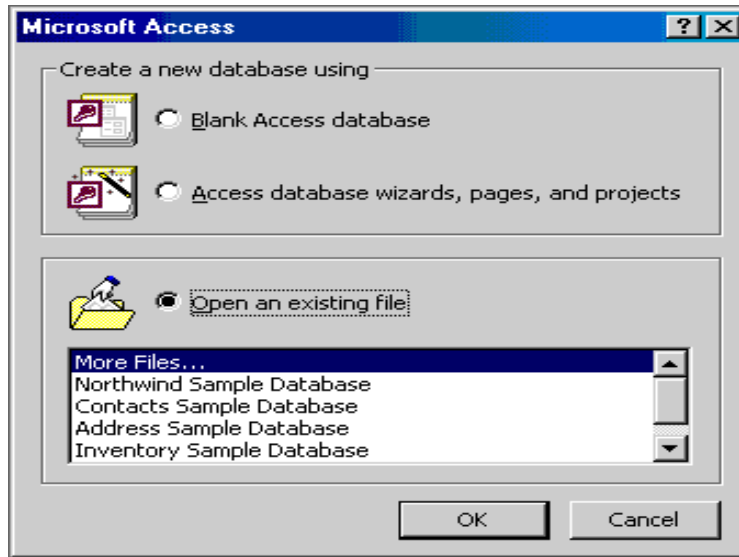
2. تظهر قائمة Start ، نختار منها Programs فتظهر قائمة أخرى نختار منها البرنامج . Access Microsoft

- تشغيل Access 2000 من نافذة My Computer :-

1. من سطح المكتب انقر نقرًا مزدوجاً على رمز My Computer .
2. من نافذة My Computer انقر نقرًا مزدوجاً على رمز مشغل القرص C .
3. انقر نقرًا مزدوجاً على مجلد Access 2000 لفتحه .

وبمجرد تشغيل البرنامج يتم فتح مربع حوار يطلب منها إنشاء قاعدة بيانات جديدة باستخدام

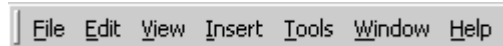
قاعدة بيانات فارغة أو باستخدام معالجات ، أو فتح قاعدة بيانات موجودة .



- الشاشة الافتتاحية لبرنامج Access 2000 :-

تتكون شاشة البرنامج من :-

أ. شريط القوائم Menu bar : حيث يحتوي على 7 قوائم وكل قائمة تحتوي على مجموعة من الأوامر .



ب. شريط الأدوات القياسي Tool bar : يوجد أسفل الشريط السابق ويحتوي على أزار ورموز تستخدم بديلاً للأوامر .



ج. شريط المعلومات bar Status : يوجد في أسفل الشاشة ويوجد عليه بعض المعلومات عن



الملف الفتح مثل اسمه وعدد سجلاته وموقع المؤشر داخل الملف .

د. مربع قائمة التحكم Menu Box Control : ويوجد في أقصى اليسار العلوي من الشاشة على شكل مفتاح حيث النقر المزدوج على هذا المفتاح إلى إغلاق البرنامج والنقر الفردي يؤدي إلى إظهار قائمة التحكم .

هـ. زر الإغلاق والتقليص والتكبير : وتوجد في أقصى يمين الشاشة العلوي وتستخدم هذه الأزرار في إلاق النافذة وتصغيرها وتكبيرها .

و. منطقة العمل : وهي المنطقة الكبيرة والتي تظهر بها كائنات قاعدة البيانات التي نتعامل معها مثل الجداول والنماذج والاستعلامات ... الخ

- إنهاء Access 2000 :-

بعد حفظ العمل نقوم بإنهاء البرنامج وذلك بإحدى الطرق التالية :

أ. فتح قائمة ملف ومن القائمة التي تظهر نختار إنهاء .

ب. اضغط مفتاح F4 + Alt .

ج. انقر نقراً مزدوجاً على مربع قائمة التحكم .

د. انقر زر الإغلاق × .

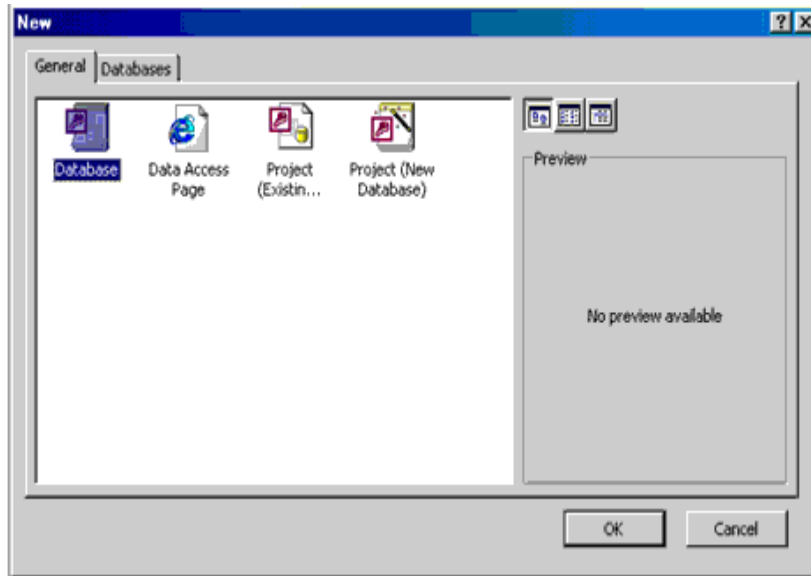
إنشاء قاعدة بيانات جديدة

- إنشاء قاعدة بيانات فارغة:

1. بعد تشغيل البرنامج يتم فتح مربع حوار نختار منه إنشاء قاعدة بيانات فارغة . ثم ننقر

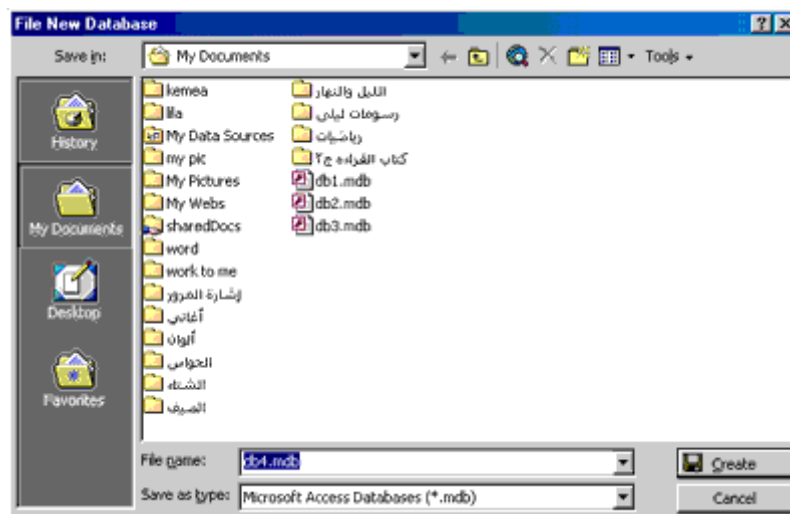
على زر موافق.

أما في حالة عدم ظهور مربع الحوار السابق نقوم بفتح قائمة ملف ثم نختار جديد أو ننقر على زر قاعدة بيانات جديدة الموجود على شريط الأدوات ، وفي كلا الحالتين يظهر نافذة (جديد) .



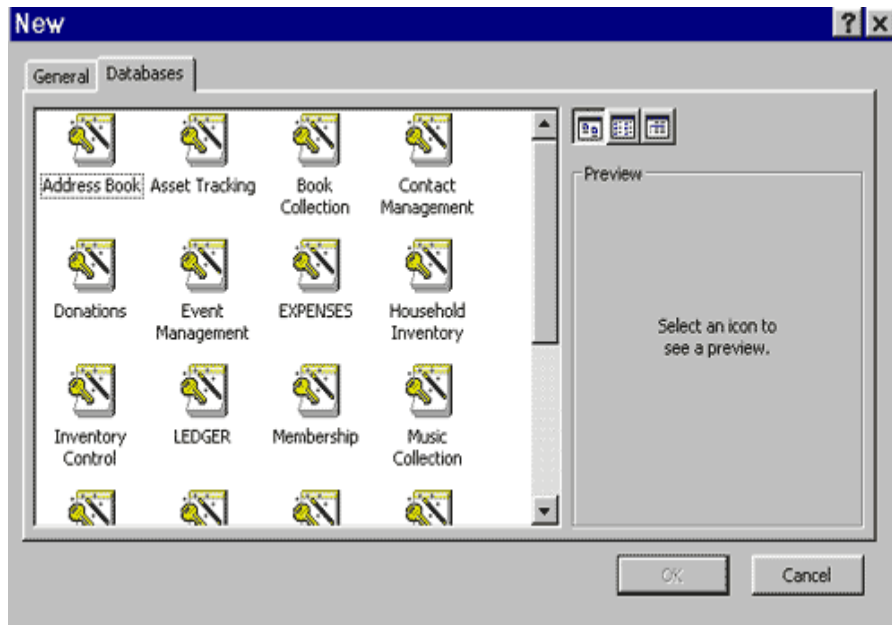
2. من خلال التبويب General ننقر على قاعدة بيانات ثم نختار موافق فيتم فتح نافذة مربع ملف قاعدة بيانات جديدة .

3. أمام خانة اسم الملف نكتب اسم قاعدة البيانات ثم ننقر على الزر إنشاء ، فيقوم البرنامج بإنشاء قاعدة بيانات جديدة فارغة وبالاسم الذي اخترناه .



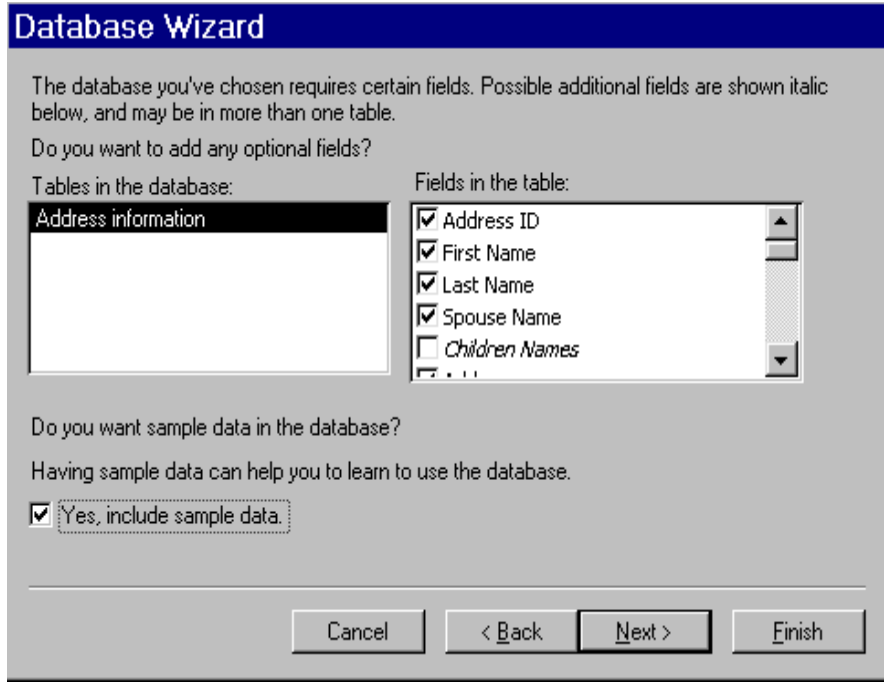
- إنشاء قاعدة بيانات باستخدام المعالج :-

1. بعد تشغيل البرنامج يتم فتح مربع حوار نختار منه انشاء قاعدة بيانات باستخدام المعالج ثم نقر على زر موافق ، أما إذا لم يظهر المعالج السابق فنختار ملف ثم جديد وفي كلا الحالتين يتم اظهار نافذة مربع "جديد" .
2. من خلال التبويب Data base انقر أحد المعالجات التي تريدها ثم انقر زر موافق يتم فتح مربع ملف قاعدة بيانات جديدة .



3. أمام خانة اسم الملف اكتب اسم قاعدة البيانات ثم انقر زر "إنشاء" ، يبدأ المعالج بالعمل وتظهر بعد قليل معلومات توضح ماذا سيفعل المعالج .

4. انقر زر التالي للاستمرار ، فتظهر بعد قليل قائمة بأسماء الجداول التي أنشأها المعالج لقاعدة البيانات (انظر الشكل) حيث تظهر أسماء الجداول على اليمين بينما تظهر أسماء الحقول الموجودة في الجدول المختار على اليسار .
نقوم باختيار الحقول التي نريدها وذلك بنقر المربع الموجود على يسار الحقل .



4. انقر زر التالي للاستمرار فيظهر مربع آخر يطلب منك اختيار النمط الذي ترغب باستخدامه في عروض الشاشة ، قم باختيار النمط الذي تريد ثم اضغط على زر التالي ، فيظهر مربع حوار آخر يطلب منك تحديد النمط الذي ترغب باستخدامه في التقارير المطبوعة ، اختر النمط الذي تريده ثم انقر الزر التالي . فيظهر مربع حوار آخر يطلب منا وضع عنوان لقاعدة البيانات وهل نريد تضمين صورة أم لا .

5. انقر زر التالي فيظهر آخر شكل من مربعات الحوار والذي يسأل هل نريد بدء قاعدة البيانات . نقوم بتنشيط الخيار نعم ثم ننقر على زر إنهاء .

6. يبدأ المعالج بإنشاء قاعدة بيانات تحتوي على جداول ونماذج وتقارير ... الخ . وبعد الانتهاء من إنشاء قاعدة البيانات تظهر شاشة آخر تحتنا على إدخال البيانات المطلوبة . نقوم بإدخال البيانات وبعد ذلك نغلق النافذة فيظهر إطار آخر اسمه

لوحة التبديل : حيث لا تعتبر هذه اللوحة ذات قيمة كبيرة وإنما هي نموذج جميل يسمح لك بأداء الأعمال التي تطلبها من قاعدة البيانات بمجرد النقر على الزر المناسب .

وتظهر هذه اللوحة في كل مرة يتم فتح قاعدة البيانات حيث نقوم بإغلاقها وبمجرد إغلاقها يظهر إطار قاعدة البيانات حيث يحتوي على جميع الجداول والنماذج والاستعلامات ... الخ ، الذي قام المعالج بإنشائها .

- حفظ قاعدة البيانات :-

يمتاز برنامج Access 2000 بميزة حفظ القاعدة بمجرد تسميتها ويتم أيضاً حفظ أي سجل بمجرد إدخاله .

ولكن إذا قمت بعمل أي تغييرات في تصميم جدول أو نموذج أو تقرير وقمت بإغلاقه يظهر مربع حوار يطلب منك هل تريد حفظ التغييرات أم لا .

- إغلاق قاعدة البيانات :-

يتم إغلاق قاعدة البيانات بإحدى الطرق التالية :

1. انقر نقرًا مزدوجاً على مربع قائمة التحكم .

2. انقر زر الإغلاق T .

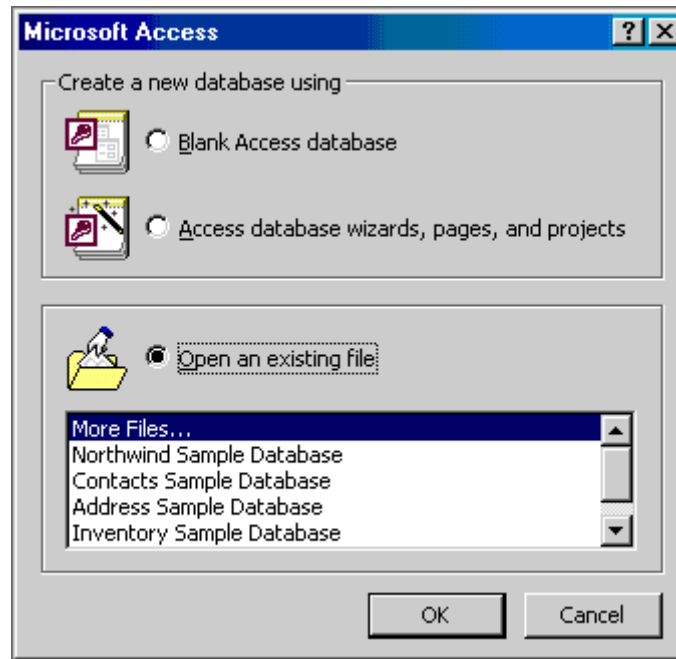
3. من قائمة ملف نختار إغلاق .

4. اضغط مفتاح Ctrl + F .

- فتح قاعدة البيانات :-

يتم فتح قاعدة البيانات بإحدى الطرق التالية :

1. من قائمة ملف نختار فتح ومن مربع الحوار الذي يظهر نحدد مكان الملف المطلوب ونقوم بفتحه . أو ننقر على زر فتح الموجود على شريط الأدوات .
2. عند فتح برنامج Access 2000 من مربع الحوار الذي يظهر ، نختار فتح ملف موجود ونقوم بتحديد الملف الذي نريد فتحه.



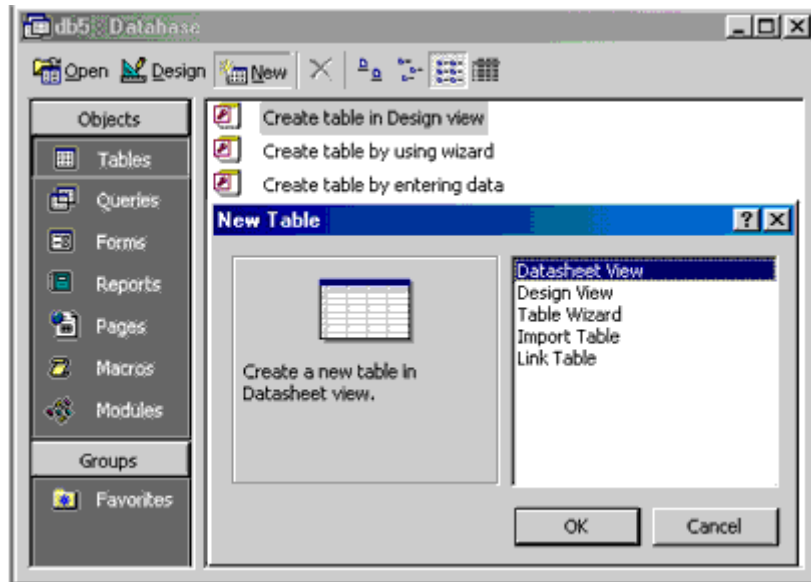
الفصل الخامس: إنشاء الجداول

إنشاء الجدول باستخدام المعالج :-

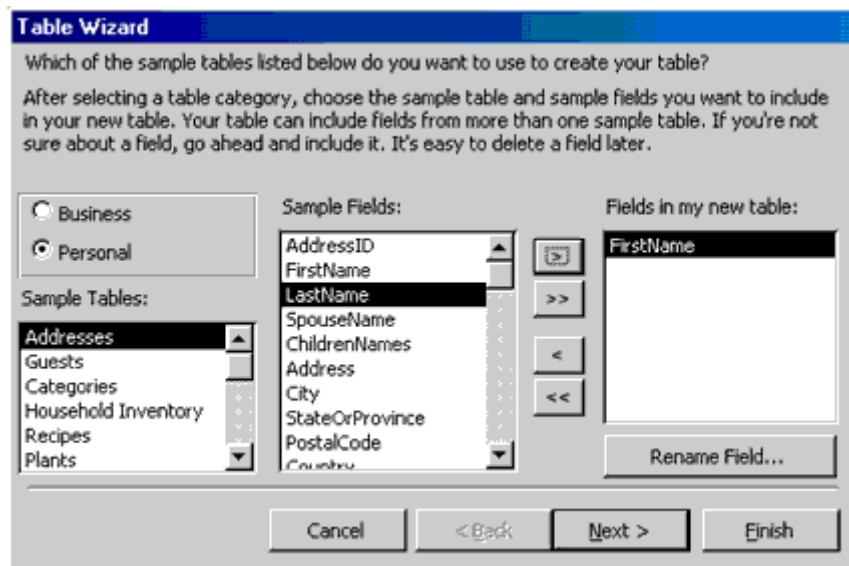
الجدول هو الأساس في أي قاعدة بيانات حيث أن الكائنات الأخرى مثل النماذج والتقارير والاستعلامات تستخرج عادة من بيانات الجداول وليس من أي كائن آخر .

1. نقوم باختيار التبويب جداول من إطار قاعدة بيانات ثم ننقر على زر "جديد" الموجود في أعلى إطار قاعدة البيانات ، حيث يظهر مربع "جدول جديد" (انظر الشكل) ، ويمكن الحصول على نفس المربع من قائمة إدراج أو بالنقر على زر كائن جديد من شريط الأدوات .

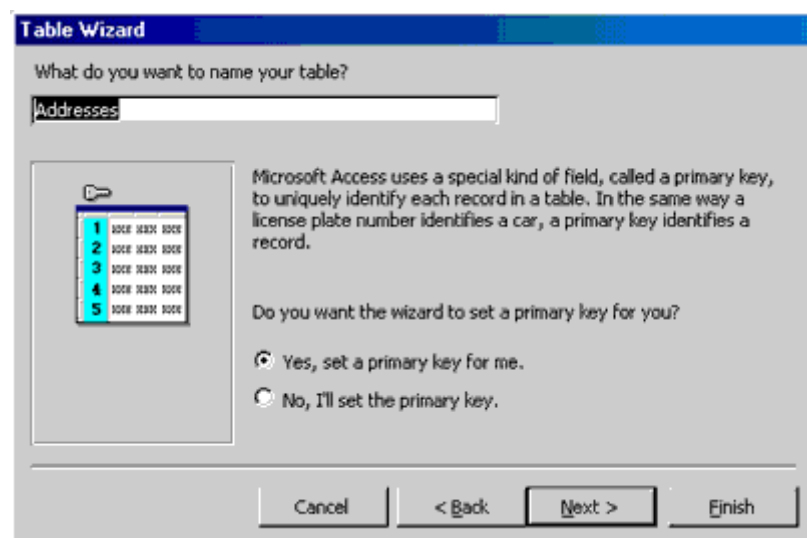
ويتم إنشاء الجدول باستخدام المعالج كما يلي



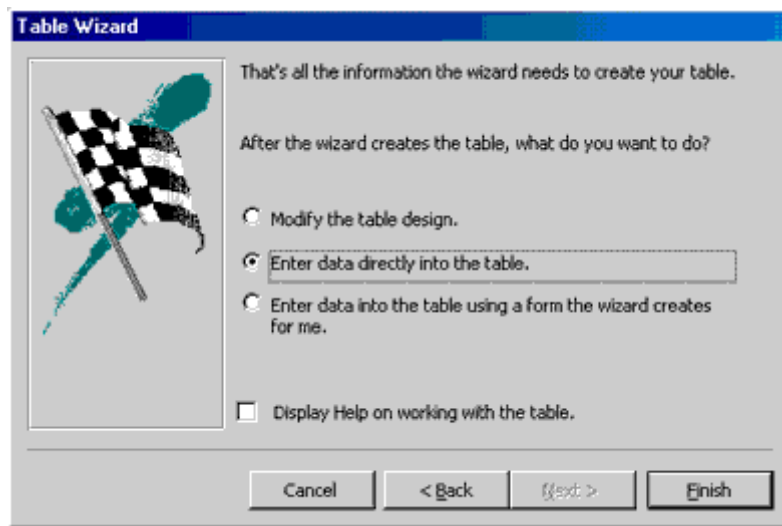
2. نختار من المربع معالج الجدول ثم ننقر زر موافق فيظهر معالج الجداول (انظر الشكل) ، تظهر تلقائياً قائمة الجداول التي تخص العمل وإذا رغبت في إظهار قائمة الجداول الشخصية نشط خانة الاختيار "شخصي" .



3. من خانة نماذج الجداول ننقر على الجدول الذي نريده ، حيث تظهر حقول هذا الجدول في خانة نماذج الحقول ، نقوم بعد ذلك باختيار الحقل الذي نريده ثم النقر على زر > حيث يؤدي إلى نقل إلى الجدول الجديد . ومن الممكن أيضاً اختيار حقول أخرى من خانة نماذج الجداول وإضافتها إلى الجدول الجديد ، بعد اختيار الحقول التي نريدها وتم نقلها إلى الجدول الجديد ننقر على زر التالي فيظهر مربع معالج الجداول حيث نقوم بكتابة اسماً للجدول .



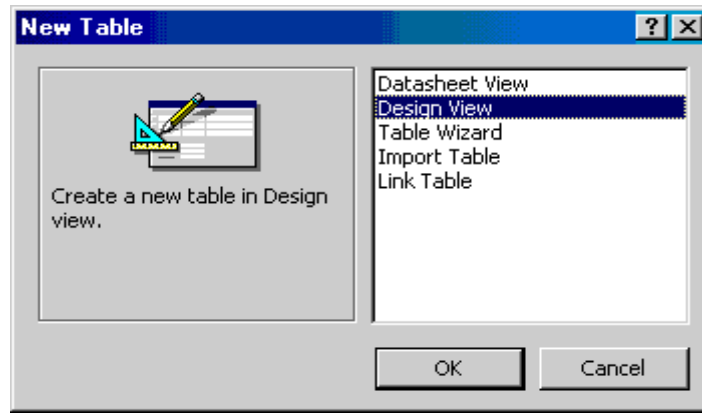
4. من مربع معالج الجداول نقوم بتنشيط الخيار (نعم) قم بتعيين مفتاح أساسي نيابة عنه ، حيث يقوم المعالج بتعيين حقل معين ليكون مفتاح أساسي ، (المفتاح الأساسي عبارة عن علامة مميزة تكون موجود بجانب كل سجل نقوم بتمييزه عن غيره وذلك لمنع دخول نفس البيانات في نفس الحقل المستخدم كمفتاح أساسي . وعادة يتم اختيار السجل الذي يحتوي رقم وليس على اسم بحيث لا يتكرر). ثم انقر زر التالي فتظهر آخر شاشة من شاشات معالج الجداول ، حيث يحتوي على خيارات كثيرة .



5. انقر على الخيار الذي تريده ثم انقر زر إنهاء فيتم إنشاء الجدول حسب الاختيار الذي تم تحديده بالسابق وبعد عمل المطلوب قم بإغلاق الجدول .

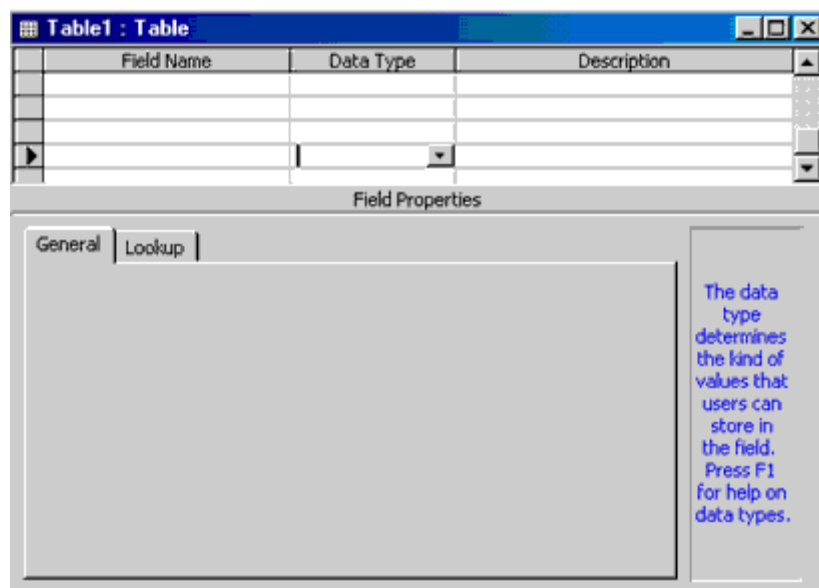
- إنشاء جدول بدون المعالج :-

1. من إطار قاعدة البيانات نشط التبويب جداول ثم انقر الزر جديد فيظهر مربع (جدول جديد) ومن هذا المربع انقر "طريقة عرض التصميم" ثم انقر زر موافق فتظهر نافذة عرض تصميم الجدول .



2. في أول سطر وتحت عمود اسم الحقل اكتب اسم أول حقل بعد ذلك انقل المؤشر إلى العمود الثاني (عمود نوع البيانات) حيث يوجد بجانبه سهم صغير وعند الضغط عليه تظهر قائمة مسند له تحتوي على مجموعة من أنواع الحقول مثل (نص ، رقم ، تاريخ، نعم / لا ، عمله ، مذكر ، ترقيم تلقائي ، كائن ، معالج البحث) قم باختيار نوع الحقل حسب العمود الذي قبله اسم الحقل .

3. انتقل إلى العمود الثالث (الوصف) ثم اكتب وصف للحقل الذي تعمل عليه ، وهو أمر اختياري .



4. نلاحظ أيضاً في النصف السفلي من نافذة عرض التصميم تظهر خصائص الحقل الحالي

المختار قم تحديد خصائص الحقل بما يناسبك . مثل (الحجم ، التنسيق ، الأماكن العشري ، قناع الإدخال ، تعليق ... الخ) .

5. قم بتعبئة أسماء جميع الحقول التي ترغب بها مع تحديد نوع البيانات والخصائص .

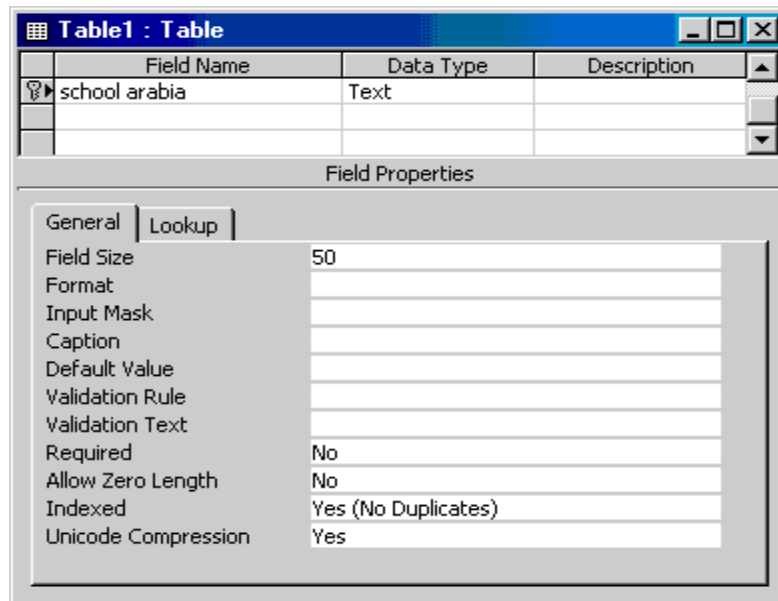
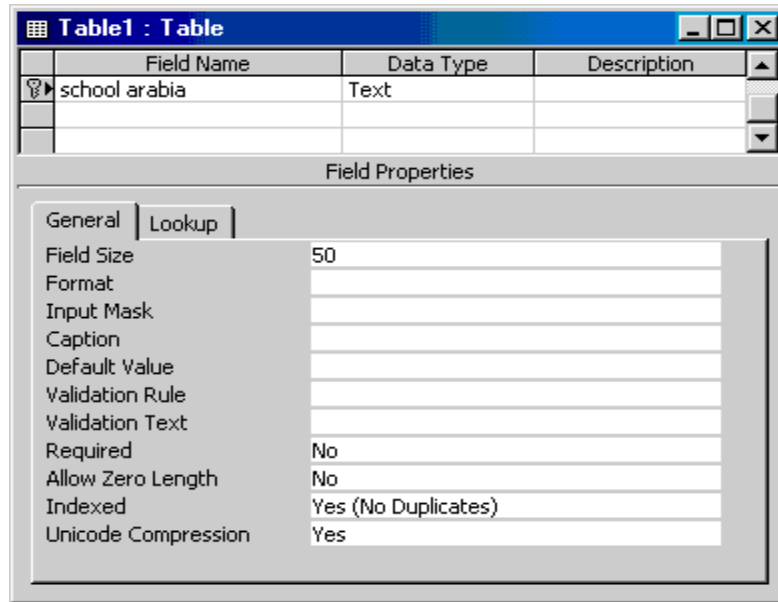
6. انقر زر الإغلاق فتظهر رسال تخبرك بحفظ التغييرات ، اختر نعم ، فيظهر مربع حفظ باسم اكتب اسماً للجدول ثم انقر موافق .

- ضبط المفتاح الأساسي بدون معالج :-

يجب تخصيص حقل أو أكثر من حقول الجدول وجعله مفتاح أساسي Primary Key ويراعى عند اختيار الحقل أن يشتمل على بيانات لا يسمح بتكرارها داخل الجدول مثل رقم حساب العميل أو الرقم الوطني ... الخ .

ولعمل ذلك :

1. ضع المؤشر أمام الحقل حتى يتحول إلى سهم أسود صغير ثم انقر زر الفأرة حيث يظهر رأس سهم صغير على يمين اسم الحقل ويتم إضاءة السجل كله .
2. وجه المؤشر إلى شريط الأدوات واضغط على زر المفتاح ، فيظهر رمز المفتاح على يمين اسم الحقل دلالة على أن هذا الحقل أصبح مخصصاً كمفتاح أساسي .



- التبديل بين عرض التصميم وعرض صفحة البيانات :-

يسمح برنامج Access 2000 بعرض جداول البيانات بطريقتين :

1. طريقة عرض التصميم : حيث يظهر فيها أسماء الحقول وخصائصها .
2. طريقة عرض صفحة البيانات : حيث يظهر فيها البيانات المسجلة بالجدول .

ويتم التبديل بين طريقتي العرض عن طريق الضغط على زر (عرض الجدول) الموجود في

أقصى اليسار من شريط الأدوات .

واختيار طريقة العرض. أو عن طريق فتح قائمة عرض ثم اختر طريقة العرض.

- تعديل الجداول

قبل القيام بعملية تعديل الجدول يجب أن يتم فتح الجدول بطريقة عرض التصميم .

- تعديل الحقول وخصائصها :-

1. بعد عرض الجدول بطريقة عرض التصميم من عمود اسم انقر اسم أي حقل لاختيار ، وقم بتعديل ما تريد .

2. من عمود نوع البيانات قم بتعديل نوع بيانات الحقل .

3. في مربع خصائص الحقل الذي يظهر بالأسف قم بتغيير الخاصية التي تريدها سواءً بنقر مربع الكتابة الذي يظهر أمام الخاصية أو من خلال القوائم المنسدلة .

4. كرر الخطوات السابقة لكل حقل ترغب في تعديل اسمه أو نوع بياناته أو خصائصه .

- إضافة حقول جديدة :-

1. اختر الحقل الذي ترغب في إضافة حقل جديد قبله .

2. من شريط الأدوات انقر زر إدراج صفوف حيث يظهر صفًا خاليًا من البيانات .

3. اكتب اسم الحقل ونوع البيانات .

- حذف حقول من الجدول :-

1. اختر الحقل الذي ترغب في حذفه .

2. من شريط الأدوات انقر زر حذف صفوف ، أو اضغط مفتاح Del ولحذف الحقل في طريقة عرض صفحة البيانات :

- أ) اختر العمود الخاص بالحقل الذي ترغب في حذفه .
- ب) افتح قائمة تحرير ثم اختر الأمر حذف عمود .
- ج) تظهر رسالة تحذيرية اختر نعم لتأكيد الحذف .

إدخال سجل إلى جدول :-

1. افتح الجدول في طريقة عرض صفحة البيانات .
2. بمجرد كتابة آخر سجل في الجدول يتم فتح سجل جديداً تحته انتظاراً لكتابة سجل آخر ، ويتم حفظ السجل بمجرد الانتقال إلى سجل جديد . وأثناء إضافة السجلات تظهر رموز على يمين السجل وهذه الرموز هي :-

▶ يعني هذا الرمز أن هذا السجل هو الحالي .

✎ يعني هذا الرمز أن هذا هو المكان الذي سيدخل فيه سجلاً جديداً

* يعني هذا الرمز أن تغيير حدث على السجل ولكنه لم يحفظ بعد

اختيار السجلات :-

1. لاختيار سجل بالكامل وجّه المؤشر إلى يمين السجل وعندما يتحول المؤشر إلى سهم ، انقر زر الفأرة الأيسر .

2. لاختيار سجلات متجاوزة اختر أول سجل ثم اضغط مفتاح Shift واستمر ضاغطاً أثناء اختيار باقي السجلات ، أو استخدم الفأرة باختيار السجل الأول ثم السحب .
3. لاختيار كل السجلات افتح قائمة تحرير ثم اختر تحديد كافة السجلات .

حذف السجلات :-

1. اختر السجل أو السجلات المطلوبة .
2. اضغط مفتاح Del .

نقل ونسخ البيانات :-

1. اختر البيانات التي تريد نسخها سواء كانت خلية أو سجل أو مجموعة سجلات .
2. ثم اختر الامر نسخ من شريط الأدوات .
3. حدد المكان الذي سوف تنسخ إليه البيانات .
4. اختر الأمر لصق من شريط الأدوات .

الانتقال داخل الجدول :-

1. يمكن استخدام الفأرة لاختيار أي حقل أو سجل .
2. يمكن استخدام لوحة المفاتيح للتنقل داخل الجدول .

تنسيق الجداول

تغيير عرض الأعمدة :-

1. وجه المؤشر إلى الخط الرأس الذي يفصل بين أسماء الحقول وعندما يتحول المؤشر إلى شكل سهم برأسين 1 اسحب الخط الرأسي لجهة اليسار أو اليمين لزيادة عرض العمود .
 2. أو اختر العمود أو الأعمدة التي نريد تغيير عرضها ثم افتح قائمة تنسيق ثم اختر أمر "عرض العمود" فيظهر مربع (عرض العمود) .
- قم بضبط عرض العمود وذلك بكتابة رقم عرض العمود داخل خانة عرض العمود . أو قم باختيار الاحتواء الأفضل وذلك لضبط حجم العمود ليتناسب تماماً مع البيانات الموجودة به .
- بعد ذلك انقر زر موافق .

تغيير ارتفاع الصفوف :-

وجه المؤشر إلى عمود اختيار السجل ثم ثبته على أي خط من الخطوط الشبكية التي تظهر تحت السطور ، فيتغير شكل المؤشر إلى سهم برأسين 2 ، اسحب السهم لأسفل لتزيد من ارتفاع السطور .

تغيير خط الكتابة :-

إن اختيار الخط الذي نريد سوف يؤثر على كل بيانات الجدول ولن يؤثر على الخط الموجود في النماذج والتقارير .

لاختيار خط اتبع ما يلي :-

1. افتح قائمة تنسيق ثم اختر أمر خط فينظهر مربع حوار خط .



2. من خانة الخط حدد نوع الخط المطلوب .

3. من خانة النمط حدد النمط الذي تريده .

4. من خانة الحجم حدد الحجم الذي تريده .

5. من خانة اللون حدد اللون الذي تريده .

6. انقر زر موافق .

تجميد الأعمدة وإعادة تحريرها :-

تستخدم فكرة تجميد الأعمدة لتثبيت حقل معين أثناء طي الشاشة لرؤية الحقول الأخيرة من

الشاشة مع الحقل الأول مثلاً . ولعمل ذلك :-

1. اختر العمود الذي تريد تجميده .

2. افتح قائمة تنسيق ثم اختر تجميد أعمدة من القائمة المنسدلة .

3. انقر شريط التمرير الأفقي إلى الجهة المعاكسة للعمود الذي اخترت تجميده . سوف ترى بأن

العمود الذي اخترت تجميده سوف يبقى ثابتاً وبقية الأعمدة سوف تتحرك .

4. لإزالة التجميد ، افتح قائمة تنسيق ثم اختر تحرير كافة الأعمدة من القائمة المنسدلة

البحث عن البيانات وترتيبها

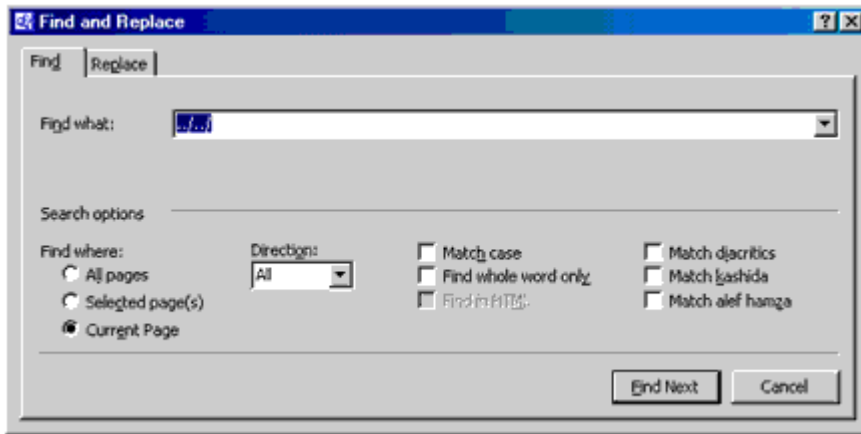
البحث عن المعلومات يعني توجيه سؤال والحصول على الإجابة . وتوجد عدة طرق للبحث

منها :

البحث باستخدام خاصية البحث :-

للبحث عن سجل معين اتباع الخطوات التالية :

1. من صفحة البيانات اختار العمود الذي يحتوي على المعلومة التي تبحث عنها .
2. من شريط الأدوات انقر زر بحث فيظهر مربع حوارى بحث واستبدال .



3. اختر التبويب بحث ثم اكتب في خانة البحث عن النص الذي تبحث عنه .
- وفي خانة البحث في حدد اسم الحقل الذي تريد البحث فيه ثم انقر على زر بحث عن التالي .
- فيتم في هذه الحالة البحث عن السجل المطلوب وعندما يجده يضعه تحت الشريط المضاء .
4. انقر زر الإغلاق لتعود إلى جدول البيانات وقراءة البيانات التي تبحث عنها .

البحث بجزء من المعلومة :-

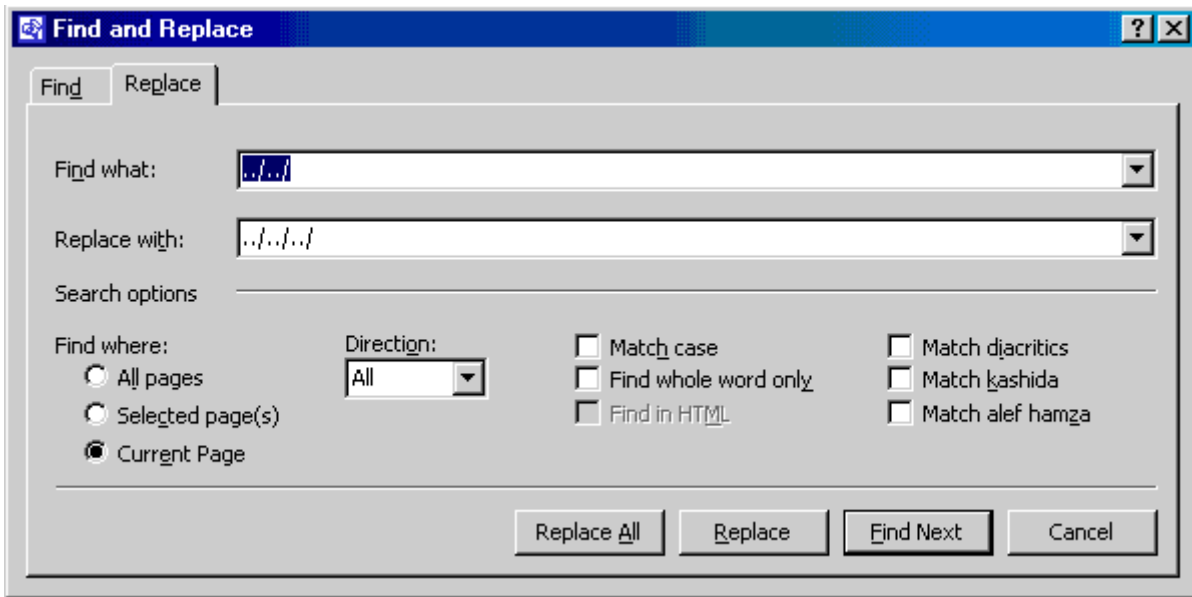
نستخدم للبحث عن سجل لا نعرف إلا جزء من النص .

1. من مربع الحوار السابق انقر الزر، الموجود في خانة مطابقة فتظهر قائمة منسدلة بالاختيارات التي يمكن البحث فيها. (انظر الشكل)

2. اختر (أي جزء من الحقل) ثم انقر زر بحث عن التالي فيقوم البرنامج بالبحث عن أول سجل توجد به المعلومة المتوفرة وتضعه تحت الشريط المضاء .

البحث باستخدام خاصية الاستبدال :-

- نستخدم هذه الخاصية للبحث عن معلومة معينة واستبدالها بواحدة أخرى .
1. اختر العمود الذي يحتوي على المعلومة التي تريد استبدالها، ثم اختر أمر استبدال من قائمة تحرير ، فيظهر مربع الحوار السابق ، قم بتنشيط التبويب استبدال .




2. اكتب المعلومة التي تبحث عنها في خانة البحث عن .
3. اكتب المعلومة التي تريد استبدال القديمة بها في خانة استبدال ب .
4. حدد الحقل الذي تريد البحث فيه .
5. انقر زر بحث عن التالي فيقوم البرنامج بالبحث عن المعلومة وعندما يجدها يضعها تحت الشريط المضاء .
6. انقر زر استبدال وذلك لاستبدال المعلومة القديمة بالجديدة .


استخدام خاصية التصفية Filtering للبحث عن مجموعة سجلات :-

يستخدم عامل التصفية لعزل سجلات تحتوي على معلومة معينة، وتوجد عدة طرق لفرز السجلات منها :-

1. التصفية حسب التحديد :-


(أ) افتح الجدول الذي تريد ، ثم حدد الحقل الذي تريد تصفية السجلات تبعاً لمحتويات ثم حدد القيمة التي تريد التصفية على أساسها .

(ب) انقر زر تصفية حسب التحديد  من شريط الأدوات فينتج فرز البيانات حسب التحديد الذي حددته .

(ج) بعد الاطلاع على السجلات انقر زر إزالة عامل التصفية .

2. التصفية حسب النموذج :-

يتم في هذه التصفية استخدام أكثر من معيار لتصفية السجلات .


(أ) افتح الجدول الذي تريد عمل تصفية له ثم انقر زر "تصفية حسب النموذج"  من شريط الأدوات . ويظهر نموذج خال بعنوان "تصفية حسب النموذج" يحتوي على سجل واحد بدون بيانات .

(ب) وجه المؤشر إلى الحقل الذي تريد ثم انقر السهم المتجه إلى أسفل في داخل الخلية واختر المعيار الذي تريده من القائمة المنسدلة .

(ج) وجه المؤشر إلى حقل آخر وانقر السهم المتجه إلى أسفل داخل الخلية واختر المعيار الثاني الذي تريده وهكذا .

(د) انقر التبويب (أو) الذي يظهر في أسفل المربع الحوار فيظهر سطر خال من البيانات

لنكتب الشرط الثاني وهكذا يمكن إضافة العديد من الشروط لمعايير التصفية .

هـ) بعد الانتهاء من كتابة كل معيار التصفية انقر زر (تطبيق عامل التصفية)  من شريط

الأدوات فتظهر السجلات التي ينطبق عليها معايير التصفية الذي حددتها .

و. بعد الانتهاء انقر على زر إزالة عامل التصفية / الفرز .

فرز السجلات تصاعدياً :-

لترتيب سجلات الجدول ترتيباً تصاعدياً أي بحسب الحروف الأبجدية من الألف إلى الياء

أوحسب الأرقام من صفر إلى 9 نقوم بما يلي :-

1. انقر أي سجل من سجلات الجدول في الحقل الذي سيتم الفرز طبقاً لمحتوياته .

2. انقر زر (فرز تصاعدي)  من شريط الأدوات .

فرز السجلات تنازلياً :-

أي الفرز حسب الحروف الأبجدية من الياء إلى الألف أو حسب الأرقام من 9 إلى الصفر :-

1. انقر أي سجل من سجلات الجدول في الحقل الذي سيتم الفرز طبقاً لمحتوياته .

2. انقر زر (فرز تنازلي)  من شريط الأدوات .

الفصل السادس: إنشاء الاستعلامات واستخدامها

Queries

الاستعلام هو تطبيق معايير بحث على بيانات الجدول ثم استعراض سجلات البيانات التي تتطابق مع الشروط المحددة .

طرق إنشاء الاستعلام : توجد طريقتين لإنشاء الاستعلام هي باستخدام المعالج أو بدون استخدام المعالج (بنفسك) (طريقة عرض التصميم)

1. إنشاء استعلام باستخدام المعالج :

يحتوي البرنامج على أربعة أنواع من المعالجات المستخدمة في الاستعلام هي :

أ) معالج الاستعلامات البسيط : وهو من أسهل أنواع الاستعلامات ، وهو أكثر أنواع الاستعلامات استخداماً ، حيث لا يتضمن أي معايير أو شروط يمكن تطبيقها على سجلات الجدول / الجداول .

ب) معالج الاستعلامات الجدولية : حيث يظهر ملخصات مثل المجموع والعدد والمتوسط الحسابي لبيانات حقل معين ، ويضعهم في مجموعة واحدة .

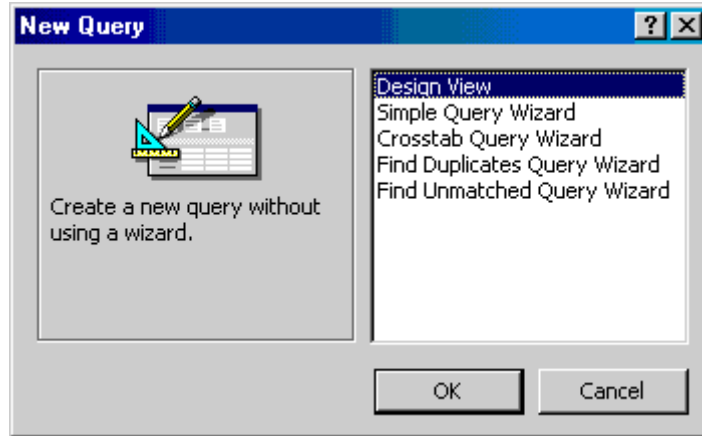
ج) معالج استعلام البحث من التكرار : حيث يقارن بين جدولين ويبحث عن السجلات المتطابقة في كليهما .

د) معالج استعلام البحث من غير المتطابقات : حيث يقارن بين جدولين ويبحث عن السجلات غير المتطابقة في كليهما .

إنشاء استعلام باستخدام معالج الاستعلامات البسيط :

أ) افتح ملف قاعدة بيانات ، ثم نشط التبويب استعلامات .

ب) انقر الزر جديد فيظهر مربع استعلام جديد .



ج) انقر (معالج الاستعلامات البسيط) ثم انقر زر موافق فيظهر أول مربع من مربعات معالج الاستعلامات البسيطة . (انظر الشكل) ، ممكن أداء الخطوتين السابقتين بخطوة واحدة وذلك

عن طريق نقر إنشاء استعلام باستخدام المعالج من إطار قاعدة البيانات نقرأ مزدوجاً

د) اختر الجدول الذي ستختار منه حقول الاستعلامات وذلك من خانة (جداول / استعلامات)

هـ) من خانة (الحقول المتاحة) حدد الحقول التي تريدها ثم قم بنقلها إلى خانة (الحقول المحددة)

عن طريق نقر الزر > بعد ذلك انقر زر التالي . فتظهر نافذة أخرى تطلب منك تحديد اسم

للاستعلام اكتب الاسم الذي تريده ثم انقر على زر إنهاء .تظهر بعد ذلك نتيجة الاستعلام في

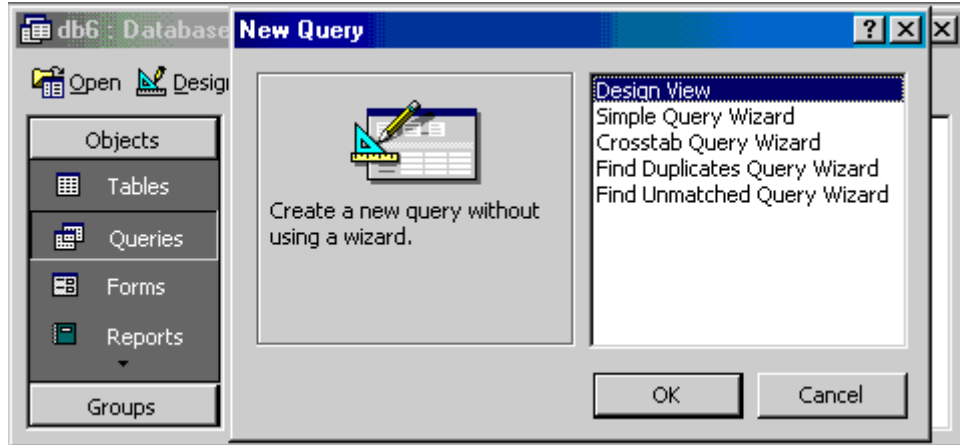
طريقة عرض صفحة بيانات .

*وممكن ايضاً عرضه بطريقة عرض التصميم وذلك للتعديل عليه .

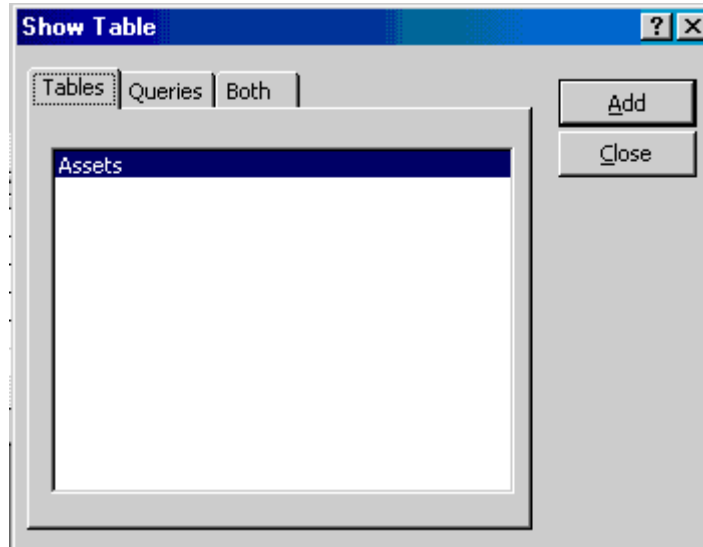
2. انشاء استعلام بطريقة عرض التصميم (بنفسك) :

أ) افتح ملف قاعدة بيانات ثم نشط التبويب استعلامات .

ب) انقر زر جديد فيظهر مربع استعلام جديد



ج) انقر (طريقة عرض التصميم) ثم انقر زر موافق ، فيظهر مربع (إظهار جدول)



يشتمل هذا المربع على 3 تبويبات وهي :

1. جداول : يظهر قائمة بأسماء الجداول الموجودة .

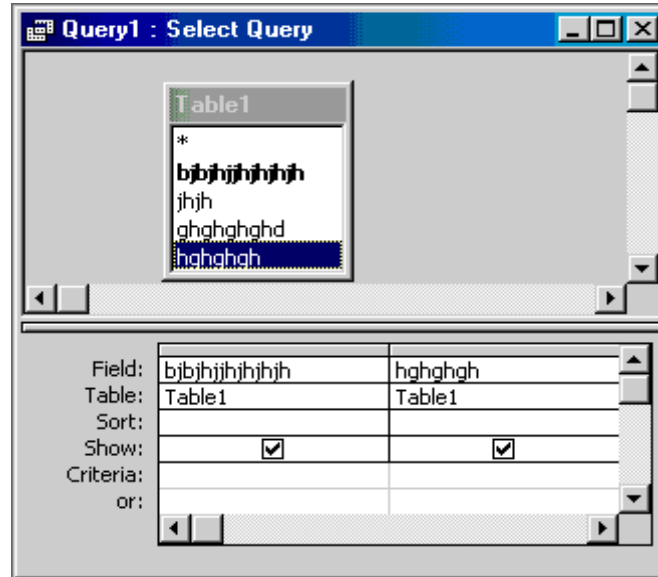
2. استعلامات : يظهر قائمة بأسماء الاستعلامات الموجودة .

3. كلاهما : يظهر قائمة بأسماء الجداول والاستعلامات الموجودة .

د) نشط التبويب جداول ثم انقر نقراً مزدوجاً على الجداول المطلوب أو حدد الجدول ثم انقر زر

إضافة (إذا قمت بإضافة أكثر من جدول يجب أن تنشأ علاقة ارتباط بينهما) .

هـ) قم بإغلاق مربع (إظهار جدول) فيظهر مربع (استعلام تحديد)



حيث يحتوي هذا المربع على قسمين علوي وسفلي ، القسم العلوي يحتوي على الجدول / الجداول

الذي تم اختياره بالسابق والقسم الآخر يحتوي على منطقة معايير الاستعلام (شبكة QBE) .

و) قم بسحب الحقول التي تريدها من الجدول إلى منطقة معايير الاستعلام .

ي) انقر زر (عرض صفحة البيانات) الموجود على شريط الأدوات فتظهر شاشة تحتوي على

البيانات المختارة .

ملاحظة : تتم عمل التعديلات المطلوبة سواءً حذف أو إضافة أو ترتيب أو فرز بطريقة عرض

التصميم . كما يمكن إضافة معايير معينة إلى الاستعلام وذلك في طريقة عرض التصميم أمام

سطر معايير من شبكة QBE . وذلك لتحديد شرط أو أكثر من شرط.

علاقة لغة SQL ببرنامج الاكسيس:

لا بد أنك قد لاحظت الآن أن هنالك الكثير من العمليات التي تستطيع قاعدة البيانات أدائها، فيمكنها أن تنشأ سجلاً أو تحذف سجلات أو تعدل سجلاً أو تغير جدولاً أو تحذف جدولاً وغيرها الكثير، هذه العمليات تجرى بواسطة لغة SQL وهي اختصار لعبارة Structured Query Language أي لغة الاستعلامات البنوية، تسمى كل عملية تنفذ عن طريق لغة SQL بالاستعلام Query، وعلى حسب لغة البرمجة التي تستخدمها وبرنامج قواعد البيانات الذي تستخدمه فإن الاستعلامات ترسل وتنفذ بطرق مختلفة وكيفية الحصول على نتائج الاستعلام تختلف أيضاً، في هذه الدورة لن يهتما كيف تقوم بإرسال الإستعلام إلى قاعدة البيانات وتنفذه وتحصل على نتائجه، ولكن ما سنتحدث عنه هو كيفية كتابة الاستعلام نفسه، لغة SQL شبه متفق عليها بين جميع نظم قواعد البيانات، على سبيل المثال في موقعنا نستخدم لغة البرمجة PHP للوصول إلى مزود قاعدة البيانات MySQL، وهي من التوليفات الشهيرة، ومنها أيضاً استخدام برنامج Visual Basic للوصول إلى مزود قاعدة بيانات Microsoft SQL Server لعمل البرامج التي تتطلب وصولاً إلى بيانات مركزية كما في المحلات والأسواق والمخازن والمستشفيات وغيرها، ويمكن استخدام لغة SQL أيضاً عن طريق البرامج المكتوبة بلغة C و ++C و Perl و Delphi للوصول إلى بيانات مخزنة في قواعد بيانات Oracle و Sybase و Informix و IBM/DB2 و Access وغيرها.

مزودات قواعد البيانات

من الطرق الشهيرة لتخزين قواعد البيانات هي وضعها في صورة ملفات نصية بسيطة Plain Text، بحيث يوضع كل سجل في سطر من أسطر الملف ويفصل بين الحقول المختلفة في كل

سجل بفاصلة comma (,)، وتكون أحيانا في صورة ملفات أكثر تعقيدا بحيث يحتوي الملف على الكثير من الجداول والفهارس التي تسرع عمليات البحث في قواعد البيانات والاستعلامات الجاهزة والنماذج كما في قواعد بيانات Microsoft Access.

هذه الطريقة قد تعتبر جيدة في البرامج البسيطة ولكن في بيئات الشبكات المعقدة والبرامج التي تحتاج وصولا مشتركا إلى البيانات تصبح هذه الفكرة صعبة التطبيق حيث أن ملف البيانات يكون في جهاز المزود، ويتم الوصول إليه عبر الشبكة من قبل الزبون، وفي حالة الملفات الثابتة يجب أن يتم نقل ملف البيانات بأكمله عبر الشبكة حتى يصل إلى الزبون الذي يأخذ المعلومات منه وإذا قام الزبون بتغيير أي من المعلومات فإن المزود يجب أن ينتظر حتى يقوم الزبون بإعادة الملف بعد عمل التغييرات عليه، وهذا أمر يستغرق الكثير من الوقت خاصة إذا كان حجم الملف كبيرا وكانت الشبكة بطيئة، وكذلك إذا أراد أكثر من شخص الوصول إلى البيانات في نفس الوقت وتغييرها في نفس الوقت سيؤدي ذلك إلى عطب البيانات وحدث خلل فيها، لهذا فإن المزود يقوم بإعطاء الملف لمن يطلبه أولا ويقوم بقفل الملف بحيث يجب على كل من يطلب الملف بعد ذلك أن ينتظر حتى ينتهي هذا الشخص من الملف، وبعدها الذي يليه وهكذا، وهو أمر غير معقول أبدا!

الحل لهذه المشكلة كان بعمل ما يسمى بمزود SQL، يقوم مزود SQL باستقبال جميع الأوامر والطلبات في صورة استعلامات SQL ثم يقوم بتنفيذ هذه العمليات على المزود ويرسل نتيجة التنفيذ فقط إلى الزبون دون الحاجة إلى إرسال قاعدة البيانات بأكملها، فإذا أردت أن تحصل على حقل واحد فقط من أحد السجلات في أحد الجداول في قاعدة البيانات فإن الجزء الذي سيتم إرساله عبر الشبكة هو الطلب في صورة SQL والجزء الذي سيتم إعادته عبر الشبكة هو هذا الحقل فقط

والذي قد لا يصل في حجمه إلى 10 بايتات مثلا إذا كان مكونه من 10 أحرف، مقارنة مع عملية إرسال قاعدة البيانات بأكملها والتي قد يصل حجمها إلى العديد من الميجابايتات وربما الجيجابايتات في بعض الأحيان، إضافة إلى ذلك فإن مزود SQL يقوم بتوزيع الأعباء على الطلبات بشكل رائع عندما يكون هنالك أكثر من طلب في نفس الوقت.

العبارة SELECT

يمكنك الحصول على البيانات في صورة سجلات باستخدام لغة SQL وذلك عن طريق العبارة SELECT والتي تأخذ الشكل العام التالي :

SELECT fields FROM tables;

حيث أن fields هي أسماء الحقول و tables هي أسماء الجداول التي نريد أن نحصل على الحقول منها، فإذا كان لدينا الجدول التالي في قاعدة البيانات على سبيل المثال :

Directory Table	
Telephone	Name
1291233	John
1682340	Tim
2462466	Jim
3636778	Dill

إذا أردنا أن نحصل على الحقلين Name و Telephone في الجدول السابق فإن العبارة التي

سنستخدمها هي : `SELECT name,telephone FROM directory;`

والعبارة السابقة تعيد الجدول كما هو في الأعلى بالضبط، ولكن ماذا لو كتبنا كلمة telephone

قبل كلمة name في عبارة SQL السابقة فتصبح كالتالي : `SELECT telephone,name`

`FROM directory;`

عند تنفيذ العبارة السابقة سنحصل على الجدول التالي :

telephone,name	
Name	Telephone
John	1291233
Tim	1682340
Jim	2462466
Dill	3636778

أما إذا كتبنا العبارة كالتالي : `SELECT name,telephone,name FROM directory;`

فسنحصل على النتيجة التالية :

name,telephone,name		
Name	Telephone	Name
John	1291233	John
Tim	1682340	Tim
Jim	2462466	Jim
Dill	3636778	Dill

وماذا لو قمنا بتنفيذ العبارة التالية : `SELECT name FROM directory;`

فالناتج سيكون كالتالي :

name
Name
John
Tim
Jim
Dill

كما تلاحظ ليست هنالك أية قواعد ثابتة، فلا يمكنك أن تقول بأن الحقل name هو الحقل الأول في الجدول، أنت من يحدد الآن ما هو الحقل الأول وما هو الحقل الثاني وهكذا.

قد تحتاج في بعض الأحيان أن تعرض جميع الحقول في الجدول، فيمكنك استخدام علامة النجمة (*) في مكان الحقول للحصول على جميع الحقول التي في الجدول، فاستخدام العبارة التالية

سيعيد الجدول بأكمله كما كتبناه أول مرة : `SELECT * FROM directory;`

قواعد اللغة

عند كتابتك لاستعلامات SQL يجب أن تتذكر الأمور التالية دائما :

- لغة SQL لا تفرق بين الحروف الكبيرة والصغيرة فلا فرق بين كتابة الكلمة `SELECT` والكلمة `select` والكلمة `SeLeCt` وكلها تعامل بنفس الطريقة، قد تكون هنالك بعض الاستثناءات في أسماء الجداول أو الحقول، يجب أن تراجع دليل الاستخدام المرفق مع برنامج قاعدة البيانات الذي تستخدمه للتأكد من ذلك.

- المسافات البيضاء ليس لها اعتبار في لغة SQL، فيمكنك وضع أي قدر تريد من المسافات البيضاء لتنسيق استعلاماتك، فيمكنك مثلا أن تكتب الاستعلام في الصورة التالية، ولا توجد أية مشكلة في ذلك :

- `SELECT *`
- `FROM directory;`

- تنتهي جميع الاستعلامات بالفاصلة المنقوطة (;).
- العبارات النصية التي لا تعتبر جزءا من عبارات الاستعلامات توضع بين قوسي اقتباس مفردين ويتيح بعض برامج قواعد البيانات استخدام أقواس الاقتباس المزدوجة أيضا (" .. ").

- إذا كنت تريد استخدام علامات الاقتباس كجزء من النص الذي تريد إدخاله إلى قاعدة البيانات فإنك تضع قبلها علامة الشرطة الخلفية لتصبح هكذا (\) أو في بعض برامج قواعد البيانات فيتم ذلك بمضاعفة علامة الاقتباس (")، فمثلا إذا أردت وضع العبارة

I'm me : قاعدة البيانات

فإنك تكتبها في أحد الصور التالية حسب برنامج قاعدة البيانات الذي تستخدمه :

'I'm me'

I"m me'

الشروط

يمكنك أثناء جلب السجلات أن تضع شروطا معينة للسجلات التي تريد الحصول عليها بواسطة العبارة SELECT وذلك باستخدام المقطع WHERE وبعده تضع الشروط التي تريدها، أنظر

مثلا : `SELECT telephone FROM directory WHERE name='Tim';`

فإن ناتج تنفيذ الاستعلام السابق سيكون كالتالي :

name='Tim'
telephone
1682340

حيث أن هنالك حقلا واحد فقط يطابق الشرط، والشرط هو أن يكون الاسم name يساوي Tim ولأن كلمة Tim جزء من البيانات المخزنة في قاعدة البيانات فإننا نحيطها بعلامات الاقتباس المفردة، لاحظ أيضا أن البرنامج لن يعيد الاسم أيضا ضمن النتائج وذلك لأننا لم نطلب الاسم في عبارتنا، ويمكننا الحصول على الاسم أيضا باستخدام هذه العبارة :
 SELECT telephone, name FROM directory WHERE name='Tim';

كما تلاحظ فإن عبارة SELECT من بدايتها وحتى ما قبل كلمة WHERE تعمل كما شرحنا في السابق بالضبط.

الجزء الذي يهمنا الآن في العبارة هو الجزء الذي يأتي بعد الكلمة WHERE أو ما يسمى بالشرط condition، تتكون عبارة الشرط الواحدة من ثلاثة أجزاء، الجزء الأول هو الطرف الأيسر من العبارة والجزء الثاني هو الطرف الأيمن من العبارة والجزء الثالث هو المعامل الذي يقع بين الطرفين، والمعامل في مثالنا السابق هو علامة المساواة (=) حيث أن شرطنا هو أن يكون الطرف الأيمن يساوي الطرف الأيسر حتى يتحقق الشرط :

rightside operator leftside
 'Tim' = name

والعوامل المستخدمة في الشروط مختلفة، أهما ما يلي :

معاملات الاختبار في SQL	
المعامل	اسمه
=	يساوي
>	أكبر من
<	أصغر من
<=	أكبر من أو يساوي
>=	أصغر من أو يساوي
<>	لا يساوي
LIKE	يشبه

العامل الأول هو عامل المساواة وهو يتحقق عندما يكون الطرف الأيمن والأيسر متساويان، كما رأينا في المثال السابق، العوامل التالية تبدو واضحة وهي < ويتحقق عندما يكون الطرف الأيسر أكبر من الطرف الأيمن، > ويتحقق عندما يكون الطرف الأيمن أصغر من الطرف الأيسر، وبعدها أكبر من أو يساوي ثم أصغر من أو يساوي.

أما المعامل السادس <> فيعني لا يساوي ويتحقق الشرط فيه عندما يكون الطرف الأيمن لا يساوي الطرف الأيسر.

قد تبدو مقارنة النصوص باستخدام العوامل < و > غريبة نوعا ما، فكيف نقول مثلا بأن 'Tim' > 'Jim' ؟ الأمر في غاية البساطة، كل ما عليك فعله هو أن تتخيل بأنك تريد أن ترتب هذا

الجدول تنازليا فهذا يعني بأن القيم العليا تكون فوق والقيم الدنيا تكون في أسفل الترتيب، ولهذا فإن 'z' < 'a' تعتبر عبارة صحيحة.

العامل الأخير من عوامل المقارنه هو العامل LIKE (يشبه) وهو يستخدم لمقارنة النصوص عادة، ويتحقق الشرط فيه عندما يكون الطرف الأيمن يشبه الطرف الأيسر، ويكتب طرفها الأيمن في صورة نص يحتوي على علامات النسبة المؤوية (%) وهي تعني (أي شيء) بمعنى أنك إذا قلت :
 SELECT name,telephone FROM directory WHERE name LIKE '%m';

فهذا يعني بأن الاسم يجب أن يكون (أي شيء) ثم الحرف 'm'، أو بمعنى آخر سيكون الشرط متحققا في السجلات التي ينتهي الاسم فيها بالحرف 'm'، وإذا كتبنا :
 SELECT name,telephone FROM directory WHERE name LIKE 'm%';

فهذا يعني m ثم (أي شيء) أي أنها تطابق حقول name التي تبدأ بالحرف m، حسنا ماذا لو قلنا :
 SELECT name,telephone FROM directory WHERE name LIKE '%m%';

أما هذه فتعني (أي شيء) ثم الحرف m ثم (أي شيء) أي أنها ستطابق جميع السجلات التي يحتوي الحقل name فيها على الحرف m.

يمكنك أن تقيس على ذلك الكثير من الأمور، فيمكنك أن تبحث في قاعدة البيانات عن حقل يبدأ بكلمة 'this' وبعدها بعدة أحرف أو كلمات أو (أي شيء) تأتي كلمة 'one' فتكتب هكذا :
 text LIKE 'this%one%'

دعنا نجرب تطبيق هذا الاستعلام على قاعدة البيانات التي نعمل عليها :
 SELECT name,telephone FROM directory WHERE name LIKE '%m';

فإن النتائج ستكون كالتالي :

name LIKE '%m'	
Telephone	Name
1682340	Tim
2462466	Jim

استخدام أكثر من شرط

يمكنك استخدام أكثر من شرط واحد عن جلب سجلات بياناتك، فيمكنك مثلا أن تبحث عن جميع الحقول التي يبدأ اسم صاحبها بالحرف J والحرف D، أو ربما تريد البحث عن جميع الأشخاص الذين تاريخ ميلادهم أكبر من 1 يناير 2000 وأصغر من 5 فبراير 2000 وهكذا أمور، يمكنك أن تربط بين الشروط باستخدام أداتين مختلفتين للربط هما AND و OR، وهاتان الأداةان تساويان && و || على التوالي في بعض لغات البرمجة.

الأداة الأولى ومن اسمها AND (و) تجعل السجل محققا للشرط عندما يتحقق الشرط الذي على يمينها والشرط الذي على يسارها معا، فمثلا عندما نقول :

```
SELECT name,telephone FROM directory WHERE
name LIKE '%m' AND telephone > 20000000;
```

ويعني ذلك بأن السجلات الوحيدة التي سيتم عرضها هي التي يتحقق فيها كل من الشرطان معاً، فتكون name تنتهي بالحرف m ورقم الهاتف أكبر من 20000000، فهي تعيد السجل الوحيد الذي يحقق الشرطان كالتالي :

```
name LIKE '%m' AND telephone >
20000000
```

Telephone	Name
2462466	Jim

أما OR (أو) فيكفي لتحقيقها أن يتحقق أحد الشرطان فقط، فإذا قلنا :

```
SELECT name,telephone FROM directory WHERE
name LIKE '%m' OR telephone > 20000000;
```

فإن جميع الحقول التي ينتهي فيها الحقل name بالحرف m بالإضافة إلى جميع الحقول التي يكون فيها الحقل telephone أكبر من 20000000، أي أنها تعيد جميع الحقول التي تحقق الشرط الأول، وجميع الحقول التي تحقق الشرط الثاني وجميع الحقول التي تحقق الشرطان معاً، ولذلك فإن ناتج تنفيذ العبارة هو ثلاثة سجلات كالتالي :

```
name LIKE '%m' OR telephone >
20000000
```

Telephone	Name
2462466	Jim
1682340	Tim
3636778	Dill

كما تلاحظ، السجل الأول Tim كان من ضمن السجلات التي حققت الشرطان معا فكان من ضمن جدول النتائج، أما السجل الثاني Jim فلم يحقق الشرط الثاني وكان الحقل telephone فيه أقل من 20000000 إلا أنه كان من ضمن جدول النتائج لأنه حقق الشرط الأول وهذا كاف، والسجل الثالث Dill حقق الشرط الثاني ولم يحقق الأول حيث أنه لا ينتهي بالحرف m.

استخدام OR و AND معا

يمكنك ربط أكثر من شرط باستخدام أكثر من أداة ربط ولأكثر من مرة، فيمكنك أن تقول مثلا :

```
SELECT name,telephone FROM directory WHERE
telephone > 20000000 OR telephone = 30000000 AND name LIKE '%m';
```

العبارة السابقة تنطوي على حيلة ما، قد يبدو لك في الوهلة الأولى أن العبارة تقوم بمطابقة جميع السجلات التي يكون الحقل name فيها ينتهي بالحرف m وفي نفس الوقت يكون رقم الهاتف فيها أكبر من 20000000 أو يكون يساوي 30000000، ولكن الواقع يختلف عن ذلك، عندما يكون هنالك أكثر من أداة ربط، ويكون هنالك أكثر من شرطين، فإن برنامج قاعدة البيانات سيقوم بتنفيذ أداة الربط AND أولا حسب ترتيبها بالجملة، وبعد ذلك يقوم بربط الجمل التي تستخدم الأداة OR.

يعني ذلك بأنه في العبارة السابقة الأداة AND تربط شرطان هما telephone = 30000000 و 'name LIKE '%m' أما الأداة OR فتربط شرطان هما telephone < 20000000 و 'name LIKE '%m' AND name LIKE '%m'، ونقول هنا بأن AND لها أولوية التنفيذ حيث قامت بربط الشروط التي على جانبيها، وأصبح شرطا واحدا وبعد ذلك يبدأ عمل

الأداة OR في ربط الشروط التي على جانبيها واللذان كانا الشرط الكبير الناتج من ربط الأداة AND للشرطان الصغيران، بالإضافة إلى الشرط الآخر الذي على شمالها.

ماذا لو أردنا الآن أن نجبر برنامج قاعدة البيانات على تنفيذ الشرط OR أولا؟ لعمل ذلك نقوم بإحاطة الشرطان اللذان على جانبا الأداة OR بالأقواس، فتصبح العبارة هكذا :

```
SELECT name,telephone FROM directory WHERE
(telephone > 20000000 OR telephone = 30000000) AND name LIKE
'%m';
```

لأن الأقواس لها أولوية أعلى في التنفيذ من العبارة OR (الأقواس لها أعلى أولوية على الإطلاق)، فإن البرنامج يقوم بتنفيذ ما بداخل الأقواس أولا، وفي داخل الأقواس سيجد شرطان مربوطان بالأداة OR فيقوم بربطهما وينتج شرط واحد كبير، وبعد ذلك يأتي دور العبارة AND فترتبط الشرط الكبير الذي بداخل الأقواس مع الشرط الذي يأتي على يمينها وهو name LIKE '%m'.

لاحظ أن الجزء الذي أتى بعد العبارة OR داخل الأقواس غير ضروري أبدا لأن الشرط سيتحقق دائما إذا كان الرقم يساوي 30000000 حتى لو لم نطلب منه ذلك تحديدا، حيث أن الشرط الذي على يسار العبارة يكفي لذلك (تذكر بأن 30000000 أكبر دائما من 20000000)، ولكننا أوردناها هنا كمثال فقط.

النفى بالعبارة NOT

تقوم العبارة NOT بنفي الشرط الذي يأتي بعدها، فتصبح العبارة صحيحة إذا كانت خاطئة

وتصبح خاطئة إذا كانت صحيحة، مثلا العبارة التالية : `SELECT name,telephone`

`FROM directory WHERE NOT name = 'Tim';`

ستعيد جميع السجلات التي لا يكون فيها الحقل name يساوي Tim وهكذا، ويمكن استخدامها

في تراكيب أكثر تعقيدا مع الأدوات AND و OR والأقواس.

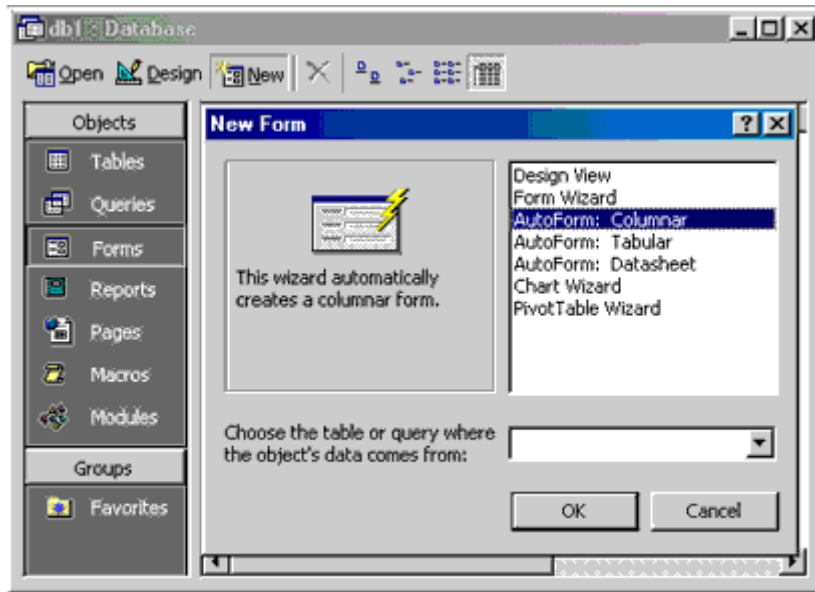
الفصل السابع: تصميم النماذج واستخدامها (Forms)

النموذج عبارة عن مستند يشتمل على بيانات سجل واحد بحيث تظهر بشكل جذاب مع إمكانية التحكم في كل عنصر من عناصره .

ويتم إنشاء النماذج بثلاث طرق هي :-

- إنشاء نموذج تلقائي : ويتم انشاءه بالطرق التالية :-

أ) من إطار قاعدة البيانات ، نشط التبويب "نماذج" ثم انقر زر "جديد" يظهر مربع نموذج جديد.

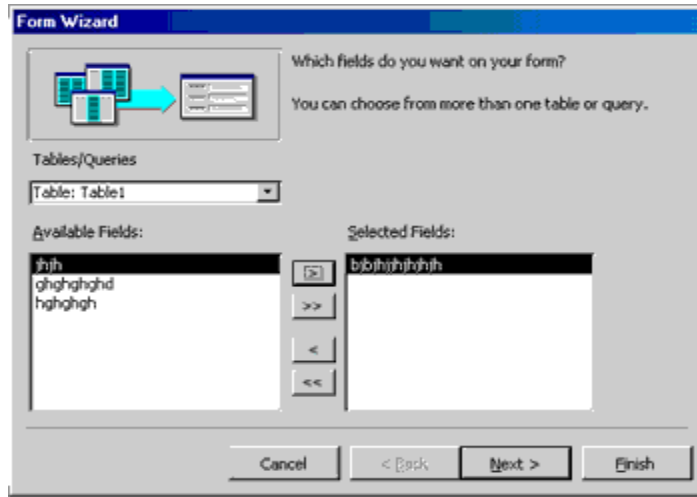


ب) يحتوي هذا المربع على 3 أشكال من النماذج التلقائية وهي :

1. نموذج تلقائي عامودي : حيث تظهر الحقول في عمود واحد .
 2. نموذج تلقائي جدولي : حيث تظهر الحقول على شكل جدول .
 3. نموذج تلقائي صفحة بيانات : حيث تظهر على شكل صفحة البيانات .
- ج) اختر النموذج التلقائي الذي تريد ثم حدد الجدول الذي سوف تأخذ منه الحقول .
- د) انقر الزر موافق .

انشاء نموذج باستخدام معالج النماذج :

1. من إطار قاعدة البيانات ، نشط التبويب "نماذج" ثم انقر زر جديد فيظهر مربع "نموذج جديد" .
2. انقر معالج النماذج وحدد الجدول الذي سوف تأخذ منه الحقول .
3. انقر الزر موافق يظهر مربع معالج النماذج .



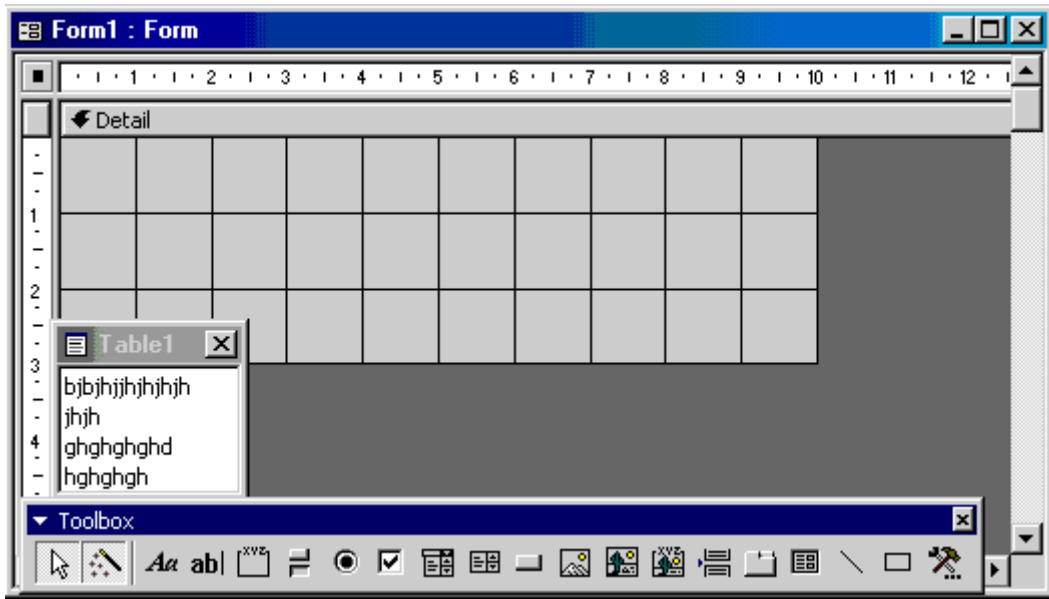
4. اختر الحقول التي تريدها من قائمة الحقول المتاحة وقم بإضافتها إلى قائمة الحقول المحددة وذلك عن طريق الزر ***. بعد ذلك انقر زر التالي .
5. يظهر مربع آخر يطلب منك تحديد نوع التخطيط الذي تريده ، اختر النوع الذي تريده ثم انقر زر التالي . يظهر مربع حوار آخر يطلب منك كتابة عنوان للنموذج ، اكتب العنوان ثم انقر زر انتهاء .

انشاء نموذج بطريقة عرض التصميم (يدوي) :

1. من إطار قاعدة البيانات نشط التبويب "نماذج" ثم انقر زر جديد فيظهر مربع نموذج جديد .
(انظر الشكل)

2. انقر طريقة عرض التصميم ، وحدد الجدول الذي سوف تأخذ منه الحقول .

3. انقر زر موافق ، فتظهر شاشة تصميم النموذج



ويظهر أيضاً مع الشاشة مربعين بحجم صغير هما مربع الحقول ومربع الأدوات ، وإذا لم يظهروا قم باظهارها عن طريق الضغط على زر قائمة الحقول وزر مربع الأدوات على شريط الأدوات .

4. قم بسحب الحقول التي تريدها من قائمة الحقول إلى شاشة تصميم النموذج في قسم تفصيل.

5. قم بحفظ النموذج .

تستطيع أن تتحكم في معظم العناصر الموجودة داخل شاشة تصميم النموذج مثل إضافة حقول جديدة أو نقل حقل إلى مكان آخر أو ترتيب الحقول أو إضافة نص إلى الحقل أو إضافة رأس وتذييل إلى النموذج أو تغيير حجم الأقسام وحجم الحقول وإضافة عناصر تحكم داخل النموذج (أزرار أوامر) خانات اختيار ، مربعات كتابة ، مربعات سرد ، مربعات كتابة سرد .

إضافة عناصر التحكم :

يتعامل برنامج Access من عناصر التحكم بصفة مستقلة أي أن كل عنصر يعامل ككائن مستقل ولذلك يمكن اختياره وسحبه ونقله متى شئت .

ويتم إضافة عناصر التحكم من شريط مربع الأدوات الذي يظهر في نافذة تصميم النموذج .



1. انشاء خانة اختيار (Check Box) □ : تستخدم هذه الخانه لاختيار قيمة من اثنين (نعم)

أو (لا) ولانشائها :


أ) انقر زر خانة الاختيار □ الموجود في مربع الأدوات حيث يتحول المؤشر إلى مربع مصحوب بعلامة زائد حدد المكان الذي سوف تضع به خانة الاختيار فيتم ربط هذه الخانة بالحقل المنطقي لها .


كما يمكن سحب حقل منطقي من نافذة الحقول ووضعه داخل النموذج فيقوم البرنامج بربطه بالخانة بحيث يظهر عنوان الحقل على يمين خانة الاختيار .

ب) إضافة مربع تحرير وسرد Box Combo : يشتمل هذا المربع على جزأين بجرء تكتب فيه القيمة التي تريدها وجزء يحتوي على قائمة تختار منها ما تريد . ويتم انشاءه كما يلي :

1. قم بإظهار زر مربع الأدوات  وزر قائمة الحقول  من شريط الأدوات إذا لم يكونوا

ظاهرين .

2. تأكد أن زر معالجات عناصر التحكم  مختاراً في مربع الأدوات .

3. انقر زر مربع التحرير والسرد  الموجود في مربع الأدوات حيث يتحول المؤشر إلى

مربع مصحوباً بعلامة + .

4. ضع المؤشر أمام الحقل الذي تريده في شاشة تصميم النموذج ، أو قم بسحب حقل


من قائمة الحقول إلى نافذة


شاشة التصميم فيظهر مربع حوار معالج التحرير والسرد .

5. حدد الخيار الذي تريده ثم تابع مربعات الحوار التي تظهر حتى النهاية .

بعد ضغط زر إنهاء يغلق مربع الحوار وترجع إلى نافذة تصميم النموذج قم بسحب مربع

التحرير والسرد إلى المكان المناسب .

(ج) إنشاء أزرار تبديل أو مجموعة خيار 

1. اختر طريقة تصميم النموذج ، ثم انقر على زر مجموعة الخيار  يتحول المؤشر إلى

مربع مصحوب بعلامة + .

2. ضع المؤشر أمام الحقل الذي تريده في شاشة التصميم فيظهر مربع معالج مجموعة

الخيار .

3. قم بكتابة الخيارات التي تريدها ثم انقر زر التالي ثم تابع مربعات الحوار حتى النهاية .

انقر على زر إنهاء يغلق مربع الحوار وترجع إلى نافذة التصميم (انقل العنصر إذا لم يعجبك

مكانه) بدل إلى طريقة عرض النموذج لكي تشاهد النتيجة .

العمليات الحسابية في النماذج

لإجراء أي عملية حسابية في النماذج كالتالي

1. يتم فتح النموذج على التصميم

2. يتم الضغط على الحقل المراد إيجاد الناتج به بيمين الماوس ثم نختار خصائص ثم من

مصدر عنصر التحكم في باب بيانات نضغط على الزر منشأ التعبير

3. يتم حذف الكلمة القديمة ثم من يتم الضغط مرتين متتاليتين على دالات ثم على وظائف مضمنة يتم الضغط مرة واحدة ثم نختار العملية الحسابية المطلوبة مثل sum او average المعدل (avg) نضغط مرتين متتاليتين .

4. ثم نضغط على expr مرة واحدة.

5. ثم نختار الحقل المراد إيجاد الناتج له مرتين متتاليتين ثم موافق ثم يتم التنفيذ .

ولعمل العمليات الحسابية بطريقة المعادلة

6. يتم فتح النموذج على التصميم

7. ثم في الحقل المراد إيجاد الناتج به يتم الضغط عليه بيمين الماوس ثم خصائص ثم من

مصدر عنصر التحكم نختار منشأ التعبير

8. يتم كتابة =

9. ثم التأشير على الحقل المطلوب مرتين متتاليتين ثم كتابة العلاقة الرياضيه المطلوبة مثل

+ / * - الخ والتأشير على الحقل الآخر مرتين أيضا لادراجة في المعادلة

وتكرار العملية حسب الحقول ثم يتم التنفيذ.

مثال: لإيجاد ناتج الراتب - الخصم

1. يتم فتح النموذج على التصميم .

2. يتم الضغط بيمين الماوس على حقل الناتج ثم نختار خصائص ثم الضغط على منشأ التعبير.

3. يتم كتابة = ثم الضغط على حقل الراتب مرتين وكتابة - والضغط مرتين على حقل الخصم ثم يتم التنفيذ.

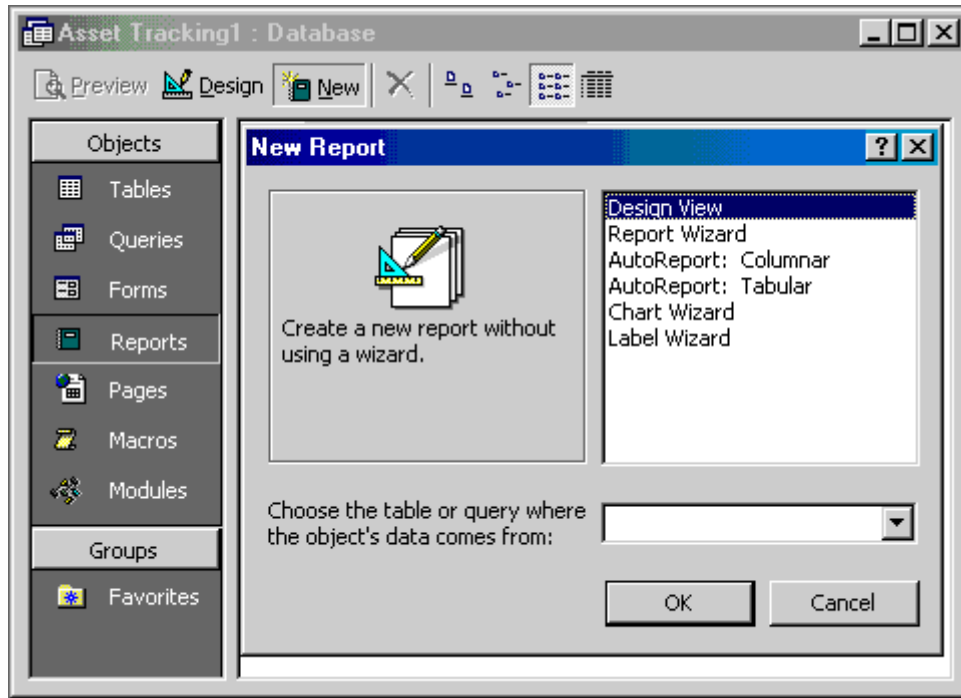
الفصل الثامن: التقارير وبطاقات التسمية (Reports)

التقارير عبارة عن مستند يمكن طباعته أو عرضه على شاشة الكمبيوتر أو حفظه في ملف :

توجد 3 طرق لإنشاء التقارير وهي :-

- إنشاء تقرير تلقائي :

1. من إطار قاعدة البيانات ، نشط التبويب (تقارير) ثم انقر زر (جديد) فيظهر مربع (تقرير جديد).

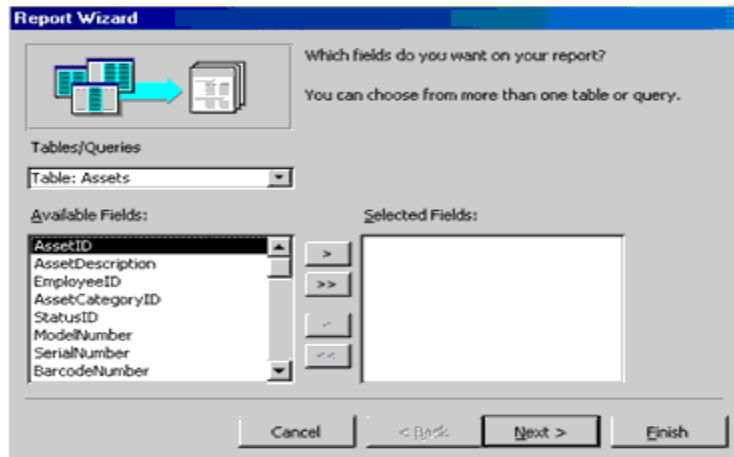


2. اختار تقرير تلقائي عمودي أو جداولي . ثم حدد الجدول الذي سوف نأخذ منه الحقول .

3. انقر الزر موافق .

انشاء تقرير باستخدام معالج التقارير :-

1. من إطار قاعدة البيانات اختر التبويب تقرير ثم انقر زر (جديد) فيظهر مربع (تقرير جديد).
2. اختار معالج التقارير وعدد الجدول الذي سوف تأخذ منه الحقول .
3. انقر الزر موافق ، فيظهر مربع معالج التقارير .



4. اختار الجدول الذي تريده من خانة جداول / استعلامات .
5. انقل الحقول التي تريدها من خانة الحقول المتاحة إلى خانة الحقول المحددة بواسطة الزر *** ثم انقر الزر التالي . فيظهر مربع حوار آخر يسألك عن رغبتك في إضافة مستويات تجميع للتقرير إذا رغبت بذلك قم بتحديد الحقول وإضافتها بواسطة الزر *** ، ومن ثم انقر زر خيارات تجميع . أما إذا لم ترغب فاضغط على زر التالي ، فيظهر مربع حوار آخر فيسألك ما هو الترتيب الذي تريده للسجلات ، ضع الترتيب الذي تريده ثم انقر زر التالي . فيظهر مربع حوار آخر يطلب منك تحديد نمط التقرير واتجاه الطباعة .
6. حدد الذي تريده ثم انقر زر التالي ، فيظهر مربع حوار آخر يطلب منك إدخال عنوان للتقرير ، اسحب العنوان ثم انقر زر إنهاء .

إنشاء تقرير بطريقة عرض التصميم :

1. من إطار قاعدة البيانات نشط التبويب (تقارير) ثم انقر زر جديد فيظهر مربع (تقرير جديد).
2. اختار طريقة عرض التصميم ثم حدد الجدول الذي سوف تأخذ منه المعلومات .
3. انقر زر موافق .
4. تظهر شاشة تصميم النموذج حيث تحتوي على قسم تفصيل وتحتوي أيضاً على مربع قائمة حقول .

اسحب الحقول التي تريدها إلى قسم تفصيل وقم بعد ذلك بعمل ما تريده كما تم طرحه في إنشاء نموذج بطريقة عرض التصميم . كما تم طرحه في إنشاء نموذج بطريقة عرض التصميم .

إنشاء بطاقات التسمية (الملصقات) :

وهي عبارة عن بطاقات تستخدم غالباً في طباعة عنوان وذلك لاستخدامه كلاصق على مظروف معين . ويتم انشائه كما يلي :

1. من نافذة قاعدة البيانات نشط التبويب "تقارير" ثم انقر زر جديد فيظهر مربع تقرير "جديد" .
2. اختار معالج التسمية ثم حدد الجدول الذي سوف تأخذ منه المعلومات .
3. انقر زر موافق ، فيظهر مربع معالج التسمية .
4. اختر حجم التسمية ونوعها ثم انقر زر التالي فتظهر نافذة أخرى حدد منها نوع الخط وحجم الخط ... الخ ثم انقر زر التالي.
5. يظهر مربع حوار آخر يطلب منك إضافة الحقول الموجودة في خانة الحقول المتاحة ووضعها في خانة النموذج الأولي للتسمية وذلك عن طريق الزر > .

6. انقر زر التالي ، فيظهر مربع حوار آخر يطلب منك الحقول التي ترغب في إجراء
***** . حدد الحقول ثم انقر زر التالي فيظهر مربع حوار آخر يطلب منك عن كتابة اسم
للتقرير .

7. اكتب الاسم ثم انقر زر انهاء .

8. اعلق نافذة المعاينة ، فيتم حفظ تقرير التسمية .

الفصل التاسع: الماكرو

تعريف الماكرو : هو سلسلة من العمليات التي تنفذ كامر واحد الهدف منه السرعة والسهولة للوصول الى غرض ما. وهنا سوف استعرض بالشرح المبسط جدا لعملية إنشاء الماكرو. يتم عمل الماكرو بالضغط على زر وحدات الماكرو في قاعدة البيانات ثم جديد ونختار الاجراء المناسب من القائمة المنسدلة للاجراءات يمكنك اختيار الاجراء (وهذه بعض الاجراءات) :-

إصدار صوت	Beep
إغلاق	Close
للتنقل بين السجلات التالي /السابق /الاولالخ.	Go to record
لوضع رسالة	msgbox
تكبير	Maximize
تصغير	Minimize
للخروج من البرنامج	quit
لفتح نموذج	open form
لفتح استعلام	open query
لفتح جدول	open table
طباعة	Print out
تشغيل برنامج مثل pbrush على سبيل المثال	runapp

مثلا :- عند اختيار الإجراء go to record يجب اختيار الكائن المراد ربط الماكرو به مثل النموذج او استعلام او جدولالخ...ثم اختيار اسم الكائن والتسجيل المناسب مثال التالي السابق.....الخ .

1. عند عمل الماكرو يتم اغلاقه وحفظه باسم .
2. لإدراج الماكرو في النموذج مثلاسوف اشرح طريقة بسيطة جدا بدون تعقيدافتح نموذجك على التصميم وضع حجمة مصغر جزئيا لكي تظهر قاعدة البيانات معه على نفس الشاشة ثم اضغط ضغط مستمر واسحب الماكرو من قاعدة البيانات للنموذج وقم بتنسيق الازرار .

لعمل مجموعة ماكرو

مجموعة الماكرو هي عدة اجراءات يتم حفظها باسم واحد أي كملف واحد ومن ثم وضعها في الكائن كالنموذج كزر واحد .

مثال: لعمل ماكرو لفتح نموذج وتكبير الشاشة ثم ظهور رسالة ترحيبية يتم اختيار الإجراءات التالية open form ثم maximize ثم msgbox ثم حفظهم جميعا كملف واحد ووضعه كزر واحد في النموذج.

ملاحظة: للتعديل بالماكرو من زر تصميم و لعمل قائمة تحتوي على الماكرو

كما سبق وشرحنا إدراج قائمة يمكن من قائمة عرض ثم اشرطة الادوات ثم من تخصيص جهة الاوامر نختار قائمة جديدة ثم نضعها بالضغط والسحب جهة القوائم ومن تعديل التحديد يمكن ان نغير اسم القائمة .

ومن جهة وحدات الماكرو يتم بالضغط والسحب نقل ألما كروبات للقائمة الجديدة حيث يمكن تغيير الاسم أو شكل الزر من زر تعديل التحديد .

الفصل العاشر: ربط الجداول

ربط الجاؤل يعني انشاء علاقة ارتباط بين جدولين أو أكثر. وتستخدم الحقول المشتركة بين الجداول في عملية الربط ، ويجب أن تكون البيانات الموجودة بين الحقول المشتركة متشابهة . وتوجد 3 أنواع من العلاقات هي :

1- علاقة ارتباط رأس برأس (واحد مقابل واحد)


2- علاقة ارتباط رأس بأطراف (واحد مقابل مجموعة)

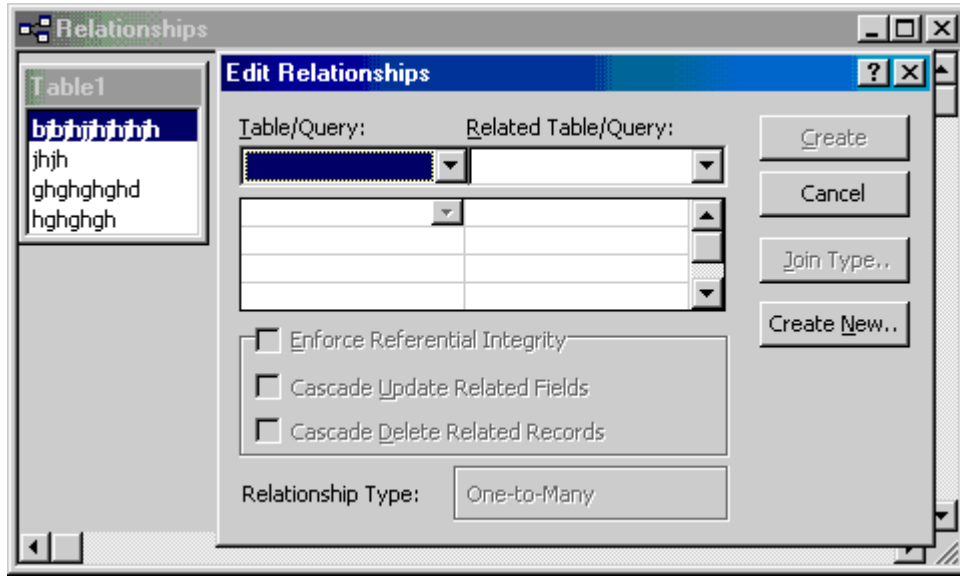
3- علاقة ارتباط أطراف بأطراف (مجموعة مقابل مجموعة)

ومن شروط إنشاء العلاقة بين جدولين :

1. تأكد أن كلا الجدولين المراد إنشاء علاقة بينهما يشتملا على حقل أو حقول متشابهة في كل شيء .
2. يجب أن تعرف من سيكون الجدول ارييس (Primary) ومن سيكون الجدول التابع (Related) . بحيث أنه إذا لم يكن الجدول التابع يشتمل على حقل يتطابق مع حقل المفتاح الأساسي في الجدول الرئيسي قم بإضافة حقلاً جديداً في الجدول التابع وبنفس مواصفات حقل المفتاح الأساسي في الجدول الرئيسي .
3. يقوم البرنامج بتحديد نوع العلاقة وفقاً لخصائص الحقول المستخدمة .

انشاء العلاقات بين الجداول :

1. افتح نافذة قاعدة البيانات .
2. اختر الأمر علاقات من قائمة أدوات أو انقر زر علاقات  من شريط الأدوات فتظهر نافذة علاقات كما في الشكل .



3. إذا لم يظهر مربع (إظهار جدول) تلقائياً اختر أمر إظهار جدول من قائمة علاقات أو انقر زر **** من شريط الأدوات .
4. من مربع إظهار جدول قم باختيار الجداول التي تريدها وقم بإضافتها بواسطة زر إضافة ، ثم انقر زر الإغلاق فتظهر نافذة العلاقات محتوية على الجدول التي تم اختيارها .
5. من نافذة علاقات اسحب الحقل من الجدول الذي تريد ربطه مع الجدول الآخر حيث يعتبر هذا الجدول هو الجدول الأساسي ، ضع الحقل الذي سحبتة فوق حقل مشابه له في الجدول الآخر (الجدول المرتبط) .
6. يظهر مربع حوار بعنوان (تحرير علاقات).

- تأكد أن الحقل المشترك في الجدولين ظاهراً في كلا الجدولين ، وإلا انقر السهم الموجود على يسار اسم الحقل ثم اختر الحقل الصحيح من القائمة المنسدلة .
7. انقر زر (إنشاء) من المربع الحواري (تحرير علاقات) فيتم غلق المربع الحواري ويظهر خط يصل الحقلين المتشابهين في كلا الجدولين ، ليوضح أن علاقة ارتباط قد أنشئت بين الجدولين.
8. قم بحفظ وإغلاق النافذة .
9. عند ربط أكثر من جدول تظهر رموز على الخطوط الواصلة بين جدولين دلالة على ارتباطهما ، رمز الدائرتين معناه أن أكثر من سجل في هذا الجدول مرتبطة بسجل واحد في الجدول الآخر ، ومعنى الرقم 1 أن كل سجل من هذا الجدول يقابله سجل في الجدول الآخر .
10. ولإلغاء العلاقة بين جدولين ، انقر الخط الواصل بين الجدولين ثم اضغط مفتاح Del .

نسخ جدول مثلا من برنامج الاكسس لبرنامج الورد

1. يتم فتح الجدول
2. ثم من قائمة أدوات نختار ارتباطات office ثم نختار النشر باستخدام ms word حيث ينسخ الملف لبرنامج الورد ويت حفظه بنفس الاسم الاصلي ولكن بامتداد rtf . rich text format

جدول مثلا لبرنامج الأكسل

1. يتم فتح الجدول
2. ثم من قائمة أدوات نختار ارتباطات office ثم نختار التحليل باستخدام ms excel حيث يتم نسخ الجدول لبرنامج excel وحفظه بنفس الاسم وببنفس امتداد برنامج الاكس xls .

ضغط قاعدة البيانات لنقلها لـدسك مرن

تأكد أولاً من حجم القاعدة من قائمة ملف ثم خصائص قاعدة البيانات ثم من باب عام .
إذا كانت القاعدة أكبر من حجم الدسك فيجب ضغطها أولاً ثم انسخها للدسك المرن كالتالي :

1- يتم إغلاق القاعدة

2- ثم من قائمة أدوات اختر ادوات مساعدة لقواعد البيانات ثم اختر ضغط قاعدة البيانات

ونختار القاعدة المراد ضغطها ثم الضغط على زر ضغط .

3- ثم نختار الجهة المراد النسخ لها مثل الدسك المرن a ثم اضغط زر حفظ .

لاخفاء قاعدة بيانات

من قائمة إطار نختار اخفاء

ولإظهارها من قائمة إطار ثم اظهار .

ملحق 1

مواصفات Microsoft Access

▪ قاعدة بيانات Access

الحد الأقصى	السمة
2 غيغا بايت مطروحاً منه المساحة اللازمة لكائنات النظام.	حجم ملف قاعدة بيانات Microsoft Access - MDB
32,768	عدد الكائنات في قاعدة بيانات
1,000	الوحدات النمطية (بما في ذلك النماذج والتقارير المعين فيها الخاصية ذات_وحدة نمطية إلى حقيقي).
64	عدد الحروف في اسم كائن
14	عدد الحروف في كلمة مرور
20	عدد الحروف في اسم مستخدم أو اسم مجموعة
255	عدد المستخدمين المتزامنين

▪ جدول

الحد الأقصى	السمة
64	عدد الحروف في اسم جدول
64	عدد الحروف في اسم حقل
255	عدد الحقول في جدول
2048 قد يكون العدد الفعلي أقل نظراً لوجود جداول مفتوحة داخلياً.	عدد الجداول المفتوحة
2 غيغا بايت مطروحاً منه المساحة اللازمة لكائنات النظام.	حجم الجدول

255	عدد الحروف في حقل نص
65,535 عند إدخال البيانات خلال واجهة تطبيق المستخدم، 1 غيغا بايت لمقدار تخزين الأحرف عند إدخال البيانات برمجياً.	عدد الحروف في حقل مذكرة
1 غيغا بايت	حجم حقل كائن OLE
32	عدد الفهارس في جدول
10	عدد الحقول في فهرس
255	عدد الحروف في رسالة تحقق من الصحة
2,048	عدد الحروف في قاعدة تحقق من الصحة
255	عدد الحروف في وصف جدول أو حقل
2,000	عدد الحروف في سجل (باستثناء حقل مذكرة وكائن OLE)
255	عدد الحروف في إعداد خاصية حقل

▪ استعلام

الحد الأقصى	السمة
32 لكل جدول مطروحاً منه عدد الفهارس الموجودة في الجدول لحقول أو تركيبات الحقول غير المتضمنة في العلاقات.	عدد العلاقات المفروضة
32	عدد الجداول في استعلام
255	عدد الحقول في مجموعة سجلات
1 غيغا بايت	حجم مجموعة سجلات
255 حرف في حقل واحد أو أكثر	حد الفرز

50	عدد مستويات الاستعلامات المتداخلة
1,024	عدد الحروف في خلية في شبكة تصميم الاستعلام
255	عدد حروف معلمة في استعلام معلمات
99	عدد كلمات AND في جملة WHERE أو HAVING
حوالي 64,000	عدد الحروف في عبارة SQL

■ النماذج والتقارير

الحد الأقصى	السمة
2,048	عدد الأحرف في التسمية
65,535	عدد الأحرف في مربع نص
55.87سم	عرض النموذج أو التقرير
55.87سم	ارتفاع المقطع
508سم	ارتفاع كافة المقاطع إضافة إلى رؤوس المقاطع (في طريقة عرض التصميم)
7	عدد مستويات النماذج والتقارير المتداخلة
10	عدد الحقول أو التعبيرات التي يمكنك تخزينها أو تجميعها في تقرير
رأس/تذييل تقرير واحد، رأس/تذييل صفحة، عشرة رؤوس/تذييلات	عدد الرؤوس والتذييلات في تقرير

مجموعات	
65,536	عدد الصفحات المطبوعة في تقرير
754	عدد عناصر التحكم والمقاطع التي يمكنك إضافتها فوق عمر النموذج أو التقرير
32,750	عدد الأحرف في عبارة SQL التي تستخدم كخاصية مصدر سجلات أو مصدر صفوف للنموذج، أو التقرير، أو عنصر التحكم (كل من .mdb و .adp).

▪ ماكرو

الحد الأقصى	السمة
999	عدد الإجراءات في ماكرو
255	عدد الحروف في شرط
255	عدد الحروف في تعليق
255	عدد الحروف في وسيطة إجراء

▪ مواصفات مشروع Microsoft Access

▪ مشروع Access

الحد الأقصى	السمة
32,768	عدد الكائنات في مشروع Microsoft Access (adp.)Access

1,000	الوحدات النمطية
64	عدد الحروف في اسم كائن
250 (Microsoft SQL Server 6.5) 1024 (Microsoft SQL Server 7.0) و2000	عدد الأعمدة في جدول

▪ قاعدة بيانات Microsoft SQL Server

تم وصف الحد الأقصى لمواصفات قدرة Microsoft SQL Server ضمن مجموعة وثائق SQL Server. لمزيد من المعلومات حول كتب SQL Server الفورية، انظر موقع Microsoft Developer's Network على ويب. ملاحظة تنقلك هذه الارتباطات التشعبية في هذا الموضوع إلى ويب. يمكنك الرجوع إلى التعليمات في أي وقت.

▪ النماذج والتقارير

الحد الأقصى	السمة
2,048	عدد الأحرف في التسمية
65,535	عدد الأحرف في مربع نص
55.87سم	عرض النموذج أو التقرير
55.87سم	ارتفاع المقطع
508سم	ارتفاع كافة المقاطع إضافة إلى رؤوس المقاطع (في طريقة عرض التصميم)
7	عدد مستويات النماذج والتقارير المتداخلة
10	عدد الحقول أو التعبيرات التي يمكنك تخزينها أو تجميعها

	في تقرير
رأس/تذييل تقرير واحد، رأس/تذييل صفحة، عشرة رؤوس/تذييلات مجموعات	عدد الرؤوس والتذييلات في تقرير
65,536	عدد الصفحات المطبوعة في تقرير
754	عدد عناصر التحكم والمقاطع التي يمكنك إضافتها فوق عمر النموذج أو التقرير
32,750	عدد الأحرف في عبارة SQL التي تستخدم كخاصية مصدر سجلات أو مصدر صفوف للنموذج، أو التقرير، أو عنصر التحكم (كل من .mdb و .adp).

▪ ماكرو

الحد الأقصى	السمة
999	عدد الإجراءات في ماكرو
255	عدد الحروف في شرط
255	عدد الحروف في تعليق
255	عدد الحروف في وسيطة إجراء

الجزء الثالث:

لغة الإستعلام الهيكلية

SQL

الفصل الحادي عشر: الاستعلام

منذ ابتكار لغة الإستعلام الهيكلية في مطلع السبعينات، خضعت للعديد من التعديلات والتطورات، كما قامت العديد من الشركات والمؤسسات البحثية بتصميم نسخها الخاصة من لغة الإستعلام الهيكلية، للتعامل مع هذا الوضع قام المعهد الوطني الامريكي American National Standards Institute ANSI بإصدار أول نسخة معيارية من لغة الإستعلام الهيكلية عام 1987 وهي التي عرفت باسم SQL1987 ، ثم توالى التعديلات والطبعات المعيارية، وحالياً فإن اللغة المعيارية المعتمدة هي SQL2003 .

يمكن تقسيم أوامر اللغة إلى عدة مجموعات:

- DDL Data Definition Language لغة تعريف البيانات:

هذا الجزء يختص ببناء و حذف و تعديل بنية وتعريف الجداول والمناظر. بالإضافة الى تحديد وتعديل محددات التكامل في اي وقت. يحوي هذا الجزء على اوامر لبناء و حذف الفهارس على الجداول.

- DML Data Manipulation Language لغة تغيير البيانات

يمكن هذا الجزء من اللغة من الإستعلام عن المعلومات بالإضافة الى تمكين المستخدم من تعديل وحذف واطافة البيانات على الجداول.

مع انه يمكن فهم لغة SQL بسهولة دون ان نتعلم الجبر العلائقي الا ان الجبر العلائقي باعتباره الاساس النظري للغة SQL يسهل عملية فهم و بناء الإستعلامات المعقدة بالإضافة الى انه يسهل فهم كيف يقوم نظام قواعد البيانات بتنفيذ الإستعلامات داخلياً والذي من شأنه ان يعمق فهم اللغة و يعزز من مهارات و قدرات مستخدميها.

الشكل الاساسي للاستعلام

```
SELECT [DISTINCT] select-list102
```

```
FROM from-list
```

```
WHERE qualification
```

كل جملة استعلام يجب ان تبدأ عبارة **SELECT** والتي تحدد أسماء الحقول او الاعمدة التي

سوف تظهر في النتيجة. وعبارة **FROM** وهي بمثابة عبارة ضرب كارتيزي تحدد اسماء

الجدول التي سوف يتم استخراج المعلومات منها اما عبارة **WHERE** فهي إختيارية وتحدد

معايير الاختيار من الجداول المذكورة في جملة **FROM** .

لنبدأ بمثال بسيط:

نفرض ان لدينا الجدول التالي و المسمى student

sid	sname	age	avg-marks	level
21	Ahmad	20	66	3
27	Mohammed	23	78	4
29	Tawfiq	23	72	4
34	Ammar	19	78	1
36	Mahmoud	20	76	2
46	Ruba	22	79	3
48	Tareq	22	67	3
54	Ahmad	21	81	2

مثال: اوجد اسماء واعمار الطلاب

```
SELECT DISTINCT sname, age
FROM student
```

النتيجة:

sname	age
Ahmad	20
Tawfiq	23
Ammar	19
Mahmoud	20
Ruba	22
Tareq	22

النتيجة هي مجموعة من الاسطر في كل منها اسم وعمر طالب بدون تكرار، وهذا يكافئ عملية الاسقاط في الجبر العلائقي.

اما اذا حذفنا كلمة DISTINCT فان النتيجة ستعتمد على عدد السجلات الموجودة في الجدول، اي كل سجل في الجدول يعطي سطر في النتيجة رغم ظهور تكرار، فتكون النتيجة ليست مجموعة بمعناها الرياضي لانها تسمح بالتكرار.

```
SELECT `sname` , `age`
FROM `student`
```

sname	age
Ahmad	20
Tawfiq	23
Ammar	19

sname	age
Mahmoud	20
Ruba	22
Tareq	22
Ahmad	20

نلاحظ تكرار السجل Ahmad,20

المثال التالي يوضح كيف يمكن استخدام عبارة WHERE

مثال: اوجد كل الطلاب اللذين تتجاوز اعمارهم 21 عام

```
SELECT `sid` , `sname` , `age` , `avg-marks`
FROM `student`
WHERE `age` > 21
```

sid	sname	age	avg-marks
29	Tawfiq	23	72
46	Ruba	22	79
48	Tareq	22	67

يمكن استخدام صيغة * SELECT لكي تظهر جميع الحقول، استخدام هذه الصيغة مفيد في البيئة التفاعلية لكنها ضعيفة عندما يكون الإستعلام معد للاستخدام المتكرر كجزء من نظام ما. من المثاليين السابقين نرى ان جملة SELECT تستخدم للاسقاط اما جملة WHERE فتعبر عن الاختيار. ولمزيد من فهم جملة الإستعلام يمكن ان نتصور ان تنفيذ الجملة داخل الحاسب سيتم كما يلي:

1. يتم تنفيذ الضرب الكارتيزي بين جميع الجداول المذكورة في جملة FROM

2. يتم الغاء جميع الاسطر التي لا تحقق الشروط المذكورة في جملة WHERE

3. يتم الغاء جميع الاعمدة التي لا تظهر في جملة SELECT

4. في حال وجود كلمة DISTINCT يتم الغاء الاسطر المكررة

في الواقع لا يتم تنفيذ جملة الإستعلام بالصورة الموصوف هنا بسبب عدم كفاءة العملية للتنفيذ

في الحاسب الا انها تبقى طريقة جيدة لفهم نتيجة تنفيذ جملة الإستعلام .

لنر كيف يمكن تطبيق هذا المفهوم على المثال التالي:

مثال: اوجد اسماء جميع المستخدمين اللذين استعاروا الكتاب رقم 10

جملة SQL لهذا الإستعلام هي

```
SELECT uname
FROM user u, borrow b
WHERE u.uid=b.uid
AND b.bid = 10
```

على فرض ان الجداول في قاعدة البيانات هي كمايلي:

user

UID UNAME RATING AGE

22	Haitham	7	45
31	Khaled	8	55.5
58	Zaid	10	35

borrow

UID BID borrow_date

22	10	2008-05-20
31	20	2008-02-12

إذا أردنا تطبيق الخطوات المذكورة أعلاه نبدأ بعملية الضرب الكارتيزي والتي تنتج الجدول:

UID	BID	DATE	UID	UNAME	RATING	AGE
22	10	2008-05-20	22	Haitham	7	45
31	20	2008-02-12	22	Haitham	7	45
22	10	2008-05-20	31	Khaled	8	55.5
31	20	2008-02-12	31	Khaled	8	55.5
22	10	2008-05-20	58	Zaid	10	35
31	20	2008-02-12	58	Zaid	10	35

وتم حذف جميع الاسطر التي لا تحقق الشروط في جملة WHERE فتصبح النتيجة كمايلي

UID	BID	DATE	UID	UNAME	RATING	AGE
22	10	2008-05-20	22	Haitham	7	45

و من ثم يتم حذف جميع الاعمدة عدا العمود المذكور في جملة SELECT لتصبح النتيجة:

UNAME
Haitham

الاسماء المستعارة

في بعض الاحيان نحتاج ان نعطي اسماء مستعارة للجداول، اي اسماء يمكن ان تستخدم مكان الاسم الاصلي للجداول، هذه الاسماء تكون صالحة فقط داخل جملة استفسار واحدة، من الاسباب التي تستدم لاجلها الاسماء المستعارة تقليل كمية الطباعة.

لايجاد اسم مستار لجداول نقوم بكتابة الاسم المستعار بعد اسم الجدول في جملة FROM مع ترك فراغ واحد على الاقل بين اسم الجدول و الاسم المستعار. أو بإلحاق إسم الجدول بكلمة AS ثم الإسم المستعار.

مثال: جملة المثال السابق

```
SELECT uname
FROM user u, borrow b
WHERE u.uid=b.uid
AND b.bid = 10
```

هنا قمنا بتعريف اسمان مستعاران الاول u و هو اسم مستعار لجدول user و الثاني b كاسم مستعار لجدول borrow. هذا الإستعلام يمكن أن يكتب بالصيغة التالية:

```
SELECT uname
FROM user , borrow
WHERE user.uid=borrow.uid
AND borrow.bid = 10
```

يمكن اسخدام الاسماء المستعارة قبل تعريفها اي بجملة SELECT كما يوضح المثال التالي:

```
SELECT u.uname
FROM user u, borrow b
WHERE u.uid=b.uid
AND b.bid = 10
```

امثلة على جمل استعلام بسيطة

لننظر مرة اخرى الى الإستعلام السابق استخدمنا في الإستعلام اسماء مستعارة كبديل عن استخدام اسماء الجداول كما يلي:

```
SELECT uname
FROM user , borrow
WHERE user.uid=borrow.uid
AND borrow.bid = 10
```

في المثال هنا لم نستخدم اسم مستعار للجدول و لنزيل الالتباس بين اسماء الحقول المتشابهة في الجداول المختلفة يجب استخدام اسم الجدول قبل اسم الحقل.

مثال: اوجد ارقام المستخدمين اللذين استعاروا كتاب تاريخ

```
SELECT borrow.uid
FROM book , borrow
WHERE borrow.bid = book.bid
AND type = 'History'
```

هذا الإستعلام يتطلب ربط بين جدولين الاستعارة والكتب، إستخدم كمفتاح ربط رقم الكتاب الموجود في كلا الجدولين. اذا افترضنا ان قاعدة البيانات تحتوي التالي:

book

bid	btitle	type
10	Learn PHP and MySQL Programming	
20	Al-Maurid	Dictionary
30	Ibn Khuldoun	History
40	Kalila wa Dimna	Letreture

borrow

UID	BID	borrow_date
22	20	2008-05-20
31	10	2008-05-30
58	30	2008-05-31
58	40	2008-05-31

نتيجة الإستعلام ستكون الرقم 58

اذا اردنا ان نعرف اسماء المستعيرين لكتب في التاريخ نحتاج الى ربط جدول المستخدمين.

```

SELECT user.username
FROM book, borrow, user
WHERE borrow.bid = book.bid
AND user.uid = borrow.uid
AND TYPE = 'History'

```

تحتوي هذه الجملة الربط بين ثلاث جداول بالاضافة الى الاختيار، لذا نرى ان هنالك شرطا ربط بالاضافة الى شرط الاختيار. مثال اخر

مثال: اوجد عناوين الكتب التي استعارها "زيد"

```

SELECT k . btitle
FROM book k , borrow b , user u
WHERE b . bid = k . bid
AND u . uid = b . uid
AND u . username = 'Zaid'

```

النتيجة

```

btitle
-----
Ibn Khuldoun
Kalila wa Dimna

```

هذا الإستعلام شبيه بالإستعلام السابق.

مثال: اوجد اسماء المستخدمين اللذين استعارو كتاب على الاقل

```

SELECT u.username
FROM borrow b, user u
WHERE u.uid = b.uid

```

الربط بين جدول المستخدمين و جدول الاستعارة يضمن عدم وجود سطر يحتوي اسم المستخدم اذا لم يكن له سطر يحمل رقمه في جدول الاعارة. اذا طبقنا نموذج تنفيذ جمل الإستعلام المشروح اعلاه فاننا في الخطوة الثانية نحذف جميع الاسطر التي لا تحقق شرط الدمج. يمكن ان نربط الجدول بنفسه، فمثلا اذا اردنا ان نرى اسماء المستخدمين و اعمارهم مع اسماء و اعمار المستخدمين اللذين تقل اعمارهم عنهم يمكن ان نستخدم الجملة التالية:

```
SELECT u1.username , u1.age , u2.username , u2.age
FROM user u1, user u2
WHERE u1.age > u2.age
```

النتيجة:

UNAME	AGE	UNAME	AGE
Haitham	45	Zaid	35
Husni	49	Zaid	35
Husni	49	Haitham	45
Khaled	55.5	Haitham	45
Khaled	55.5	Zaid	35
Khaled	55.5	Husni	49

هنا تفسر النتيجة بان زياد على سبيل المثال عمره اقل من هيثم.

التعابير الحسابية والدوال في عبارة SELECT

في بعض الاحيان لا تكون المعلومات المخزنة في قاعدة البيانات هي بالضبط ما نريد ان نراه. على سبيل المثال: "اراد مدرس ان يرى كيف ستكون علامات الطلاب فيما اذا رفع 5 علامات لكل طالب" او ربما نريد ان نرى العلامات بعد رفع 10% من العلامة الاصلية لكل طالب،

تطبيق المثال:

قاعدة البيانات تحوي جدول كما يلي:

Student(sid:integer, sname:varchar(40), age:number, avg-marks:number(3,1), level:number)

جملة الإستعلام التي تحقق ما يريد المدرس

```
SELECT sid , sname, avg-marks+5 AS average
FROM student
```

لاحظ استخدام كلمة AS لتغيير عنوان العمود الناتج عن عملية الجمع وذلك لان ناتج العملية الحسابية ليس عمود اصلي موجود في قاعدة البيانات بل هو اشتقاق من عمود (او اكثر) من قاعدة البيانات.

ستكون النتيجة كما في الجدول التالي:

sid	sname	average
21	Ahmad	71
29	Tawfiq	77
34	Ammar	83
36	Mahmoud	81
46	Ruba	84
48	Tareq	72
54	Ahmad	86

يمكن ان نضع اي تعبير حسابي نريد بنفس الطريقة. مع الملاحظة ان الجملة لا تقوم بتعديل قاعدة البيانات بل تعرض المعلومات بعد معالجتها حسابياً وحسب. تستخدم هذه التقنية لعرض تقارير و فحص بعض السيناريوهات او اخراج المعلومات باشكال مختلفة.

هناك مجموعة من الدوال المعرفة في لغة SQL التي تاخذ قيم و تعطي قيمة ما و يتم تنفيذها مع كل سطر مسترجع في جملة الإستعلام مرة واحدة. في كثير من الاحيان تعتمد اسماء الدوال على نظام قاعدة البيانات.

مثال: جميع الجمل التالية تعطينا عدد الاحرف فى نص معين.

```

/* On MySQL and PostgreSQL */
SELECT CHAR_LENGTH('hello');
SELECT OCTET_LENGTH(book_title) FROM titles;

/* On Microsoft SQL Server */
SELECT DATALENGTH(title)
FROM titles
WHERE type = 'popular_comp'
GO

/* On Oracle */
SELECT LENGTH('HORATIO') "Length of characters"
FROM dual;

```

مثال: لتشبيك مجموعة من النصوص (الاحرف) نستخدم CONCAT او

CONCATINATE

```

SELECT CONCAT('My ', 'bologna ', 'has ', 'a ', 'first ', 'name...');
-> 'My bologna has a first name...'

```

المزيد عن جملة WHERE

استخدام BETWEEN

يمكننا تحديد مدى محدد من القيم التي نريد استرجاعها باستخدام $=$ و $<=$ و $>=$ و AND. تتكرر هذه الإستعلامات كثيرا لذا تعطينا لغة SQL اختصار للتعبير عن هذا النوع من الإستعلامات. يمكننا بسهولة تحديد مدى من القيم باستخدام BETWEEN و NOT BETWEEN. الصيغة العامة للجملة هي:

```
SELECT column_name
FROM table_name
WHERE column_name BETWEEN value1 AND value2
```

و نفيها

```
SELECT column_name
FROM table_name
WHERE column_name NOT BETWEEN value1 AND value2
```

في الصيغة الاولى يتم استرجاع السجلات اكبر من او تساوي value1 وأقل من أو تساوي value2 فقط، أما في الحالة الثانية فقط السجلات التي تقع خارج المدى المذكور يتم استرجاعها.

مثال: اوجد اسماء و معدلات الطلاب اللذين معدلاتهم بين 70 و 79

```
SELECT `sname` , `avg-marks`
FROM `student`
WHERE `avg-marks`
BETWEEN 70
AND 79
```

النتيجة:

sname	avg-marks
Mohammed	78
Tawfiq	72
Ammar	78
Mahmoud	76
Ruba	79

التعامل مع النصوص في جمل SELECT

لمقارنة النصوص يمكن ان نستخدم الاشارات (< و > و = ...الخ) حيث ان ترتيب النصوص يحدد حسب ترتيب الاحرف في الابتنية. اما اذا إحتجنا تغيير ترتيب الاحرف عن الترتيب المعروف يمكننا استخدام مفهوم يسمى Collation الذي توفره اللغة المعيارية SQL-92. والذي من خلاله يمكن تحديد ترتيب الاحرف حسب الحاجة، فمثلا يمكننا ترتيب اسماء ايام الاسبوع حسب ما هو معمول به في بلد ما.

بالاضافة للعمليات العادية توفر لغة SQL طريقة لمقارنة الانماط عن طريق كلمة LIKE والتي تستخدم مع الاشارات % و _ . LIKE مفيدة جدا لاسترجاع المعلومات عن طريق جزء من الحقل. فالاشارة % تعني اي عدد من الاحرف والتي يمكن استخدامها في اي مكان في النمط، و يمكنك ان تضع اي عدد من الاشارات في النمط الواحد. اما الاشارة _ فهي تعني حرف واحد فقط.

مثال: اوجد ارقام و اسماء الطلاب الذين تبدأ أسمائهم بحرف M و تنتهي بحرف d

```
SELECT *
FROM `student`
WHERE sname LIKE 'M%d'
```

. النتيجة ستكون الاسماء مثل **Mohammed, Mahmoud**

ترتيب نتائج الإستعلام

ترتيب السجلات بالنسبة للغة SQL لا يعني شيء، على اعتبار اننا نتعامل مع مجموعات. لذا فان المعلومات المسترجعة تعرض بترتيب عشوائي. في بعض الاحيان يهمننا ان نرى المعلومات مرتبة حسب ترتيب محدد. يمكن اضافة الجملة **ORDER BY** الى جملة الإستعلام لتحديد الترتيب المرغوب فيه للبيانات المسترجعة. الصيغة العامة لجملة الإستعلام تصبح كالآتي:

```
SELECT [DISTINCT] <select-list>
FROM <from-list>
WHERE <qualification>
ORDER BY <column names>
```

اسماء الحقول المذكورة بعد جملة **ORDER BY** تحدد الحقول المراد ترتيب النتائج على أساسها (تسمى في بعض الاحيان مفاتيح الترتيب).

مثال: اوجد اسماء و اعمار و مستويات الطلاب مرتبة حسب اسم الطالب

```
SELECT `sname` , `age` , `level`
FROM `student`
ORDER BY `sname`
```

النتيجة:

sname	age	level
Ahmad	20	3
Ahmad	21	2
Ammar	19	1
Mahmoud	20	2
Mohammed	23	4
Ruba	22	3
Tareq	22	3
Tawfiq	23	4

يتم ترتيب الأسطر حسب الترتيب الأبثني للاسماء. يمكن ترتيب الأسطر (السجلات) حسب أكثر من حقل أي يتم ترتيب القائمة (النتيجة) حسب الحقل الأول وفي حالة تطابق القيم في الحقل الأول بين سجلين أو أكثر يتم الترتيب حسب الحقل الثاني.

مثال: اوجد اسماء ومستويات وأعمار الطلبة مرتبة حسب المستوى الدراسي وكل مستوى مرتبة السجلات به حسب الترتيب الابتنى للاسم.

```
SELECT `level` , `sname` , `age`
FROM `student`
ORDER BY `level` , `sname`
```

النتيجة:

level	sname	age
1	Ammar	19
2	Ahmad	21
2	Mahmoud	20
3	Ahmad	20
3	Ruba	22
3	Tareq	22
4	Mohammed	23
4	Tawfiq	23

إمكانية أخرى هي أن نحدد الحقول التي نريد ترتيب البيانات على اساسها حسب موقعها و ليس حسب اسمها، الموقع يحدد حسب ظهورها في النتيجة وليس حسب موقعها في الجدول في قاعدة البيانات ويبدأ الترتيب من الرقم 1 من اليسار الى اليمين.

مثال: اوجد أسماء وأعمار ومستويات الطلاب مرتبة حسب اسم الطالب

```
SELECT `sname` , `level` , `age`
FROM `student`
ORDER BY 1
```

النتيجة:

sname	level	age
Ahmad	3	20
Ahmad	2	21
Ammar	1	19
Mahmoud	2	20
Mohammed	4	23
Ruba	3	22
Tareq	3	22
Tawfiq	4	23

إلا أن هذه الطريقة غير محبذ ونستخدم كلمة DESC بعد اسم الحقل لعكس الترتيب (تنازلي).

مثال: اوجد اسماء واعمار الطلاب مرتبة من الاكبر الى الأصغر عمراً

```
SELECT `age` , `sname`
FROM `student`
ORDER BY `age` DESC
```

النتيجة:

age	sname
23	Mohammed
23	Tawfiq
22	Ruba
22	Tareq
21	Ahmad
20	Ahmad
20	Mahmoud
19	Ammar

الفصل الثاني عشر: العمليات علي المجموعات

العمليات على المجموعات (الاتحاد التقاطع و الفرق)

بما ان الجداول يمكن ان تعتبر نوع من انواع المجموعات ونواتج الإستعلامات هي عبارة عن مجموعات (مع السماح بالتكرار) فانه من الطبيعي ان تعطينا لغة SQL الادوات المستخدمة في عالم المجموعات وهي الاتحاد UNION التقاطع INTERSECTION والفرق EXCEPT ، بالاضافة الى ادوات اخرى ناتي الى ذكرها لاحقا.

نبر المثال التالي: اوجد اسماء الطلاب اللذين درسوا مساق "History" او

"Programming" على فرض ان لدينا قاعدة البيانات التالي:

الطلاب

Student(sid:integer,sname:varchar(40), age:number, avg-marks:number(3,1), level:number)

sid	sname	age	avg-marks	level
21	Ahmad	20	66	3
27	Mohammed	23	78	4
29	Tawfiq	23	72	4
34	Ammar	19	78	1
36	Mahmoud	20	76	2
46	Ruba	22	79	3
48	Tareq	22	67	3
54	Ahmad	21	81	2

المقررات

Course(cid: integer, cdesc:varcvhar(20), credit:integer, prerequisite:integer)

course

cid	cdesc	credit	prerequisit
10	english 1	3	0
11	English 2	2	10
101	calculus 1	3	0
102	calculus 2	3	101
201	programming 1	4	0
202	programming 2	4	201
301	data structure	3	202

الطالب درس مقرر

Enroll(sid:integer,cid:integer, mark:integer)

enroll

cid	sid	mark
10	21	70
11	21	69
10	27	78
11	27	75
101	27	79
102	27	60
101	29	70
201	29	77

```

SELECT sname
FROM student S, course C, enroll E
WHERE S.sid = E.sid
AND E.cid = C.cid
AND ( C.cdsc = 'calculus 1' OR C.cdsc = 'calculus 2')

```

sname

Mohammed

Mohammed

Tawfiq

هذا الإستعلام بسيط باستخدام OR في جملة WHERE لكن اذا غيرنا كلمة "او" في المثال الى كلمة "و" فيصبح المثال اصعب بكثير كما هو موضح في الاتي:

مثال: اوجد اسماء الطلاب اللذين درسوا مساق " calculus 1 " و " calculus 2 "

اذا قمنا ببساطة بتبديل كلمة OR في جملة الإستعلام بكلمة AND فان الإستعلام سيكون معناه اننا نريد اسماء الطلاب اللذين قد درسوا مساق اسمه 'calculus 1' و 'calculus 2' وهذا غير ممكن لان لكل مساق اسم واحد فقط، اي ان الإستعلام سيعطينا صفر اسطر (مجموعة خالية من الاسطر) كنتيجة. ويمكن ان نحقق الإستعلام على الشكل الاتي:

```

SELECT sname
FROM student S, course C1, enroll E1, course C2, enroll E2
WHERE S.sid = E1.sid AND E1.cid = C1.cid
AND S.sid = E2.sid AND E2.cid = C2.cid
AND C1.cdsc = 'calculus 1' AND C2.cdsc = 'calculus 2'

```

sname

Mohammed

هنا نتعامل مع جدول المساقات كانه جدولين متطابقين و قمنا بدمج الجدول مع نفسه هذه الجملة صعبة الفهم و كفاءتها منخفضة عند التنفيذ، لذا يمكن استخدام عملية الاتحاد كالاتي:

```
SELECT sname
FROM student S, course C, enroll E
WHERE S.sid = E.sid
AND E.cid = C.cid
AND C.cdsc = 'calculus 1'
UNION
SELECT sname
FROM student S, course C, enroll E
WHERE S.sid = E.sid
AND E.cid = C.cid
AND C.cdsc = 'calculus 2'
```

sname

Mohammed

Tawfiq

جملة الإستعلام الأولى تعطينا المجموعة {محمد ، توفيق} اما جملة الإستعلام الثانية تعطينا {محمد} الاتحاد بين المجموعتين يعطينا {محمد ، توفيق} كما هو مبين في الجدول أعلاه، نلاحظ هنا انه عند استخدام العمليات على المجموعات فان النتيجة تكون بدون تكرار.

يمكننا ان نعيد صياغه المثال السابق باستخدام التقاطع لاستخراج أسماء الطلاب اللذين أنهموا دراسة مساق 'calculus 1' و 'calculus 2' مساق كما يلي:

```
SELECT sname
FROM student S, course C, enroll E
WHERE S.sid = E.sid
AND E.cid = C.cid
AND C.cdsc = 'calculus 1'
INTERSECT
SELECT sname
FROM student S, course C, enroll E
WHERE S.sid = E.sid
AND E.cid = C.cid
AND C.cdsc = 'calculus 2'
```

التقاطع بين المجموعتين {محمد ، توفيق} ∩ {محمد} = {محمد}

مثال : اوجد اسماء الطلاب اللذين درسوا مساق 'calculus 1' ولم يدرسوا

'calculus 2'

```
SELECT sname
FROM student S, course C, enroll E
WHERE S.sid = E.sid
AND E.cid = C.cid
AND C.cdsc = 'calculus 1'
EXCEPT
SELECT sname
FROM student S, course C, enroll E
WHERE S.sid = E.sid
AND E.cid = C.cid
AND C.cdsc = 'calculus 2'
```

الإستعلام الأول يعطينا المجموعة {محمد ، توفيق} أما الثاني {محمد} وبالتالي المجموعة الاولى
عدى المجموع الثانية هي المجموعة {توفيق}

يمكن استخدام الاتحاد والنقاط والفرق بين اي جملي استفسار بشرط ان تكون متوافقة اي ان
النتيجة لكل منهما بها نفس عدد الاعمدة و ان الاعمدة لها نفس النوع و بالترتيب نفسه.
اي ان شروط الاتحاد هي:

1- النتيجة لكل منهما بها نفس عدد الاعمدة

2- ان الاعمدة لها نفس النوع و بالترتيب نفسه

مثال: اوجد ارقام الطلاب اللذين سجلوا مساق رقم 101 و اعمارهم اقل من 20

```
SELECT sid
FROM student
WHERE age < 20
UNION
SELECT sid
FROM enroll
WHERE cid = 101
```

```
sid
-----
34
27
29
```

الفصل الثالث عشر: جمل الاستعلام المتداخلة

جمل الإستعلام المتداخلة

من اكثر الميزات قوة في لغة SQL هي تضمين جملة استفسار داخل جملة استفسار اخرى و هو ما يسمى بجمل الإستعلام المتداخلة او المعششة Nested queries. عادتاً ما يكون الإستعلام الداخلي هو عبارة عن جدول غير موجود في قاعدة البيانات ولكن يمكن بنائه باستخدام جملة استفسار. اي اننا يمكن ان نعتبر جمل الإستعلام الداخلية وكأنها جدول مؤقت.

مثال: اوجد اسماء الطلاب اللذين سجلوا مساق رقم 101

```
SELECT sname
FROM student
WHERE sid IN (SELECT sid
              FROM enroll
              WHERE cid =101)
```

sname

Mohammed

Tawfiq

جملة الإستعلام الداخلية تعطينا ارقام الطلبة اللذين درسوا مساق رقم 101 و حسب المثال المعطى اعلاه لقاعدة البيانات ستكون النتيجة هي 27 و 29 و الإستعلام الخارجي يعطينا اسماء الطلبة اللذين يحملون الارقام الموجودة داخل المجموعة التي يعطيها الإستعلام الداخلي. كلمة IN تفحص اذا كانت قيمة معينة داخل مجموعة من القيم. و بكل بساطة يمكننا ان نعدل على الإستعلام كي نسترجع اسماء الطلبة اللذين لم يسجلوا مساق رقم 101 باضافة NOT قبل IN.

يمكن ان نتخيل ان جملة الإستعلام تنفذ بالطريقة المشروحة سابقا لكن مع بعض التعديل البسيط كالتالي: اولاً يتم تنفيذ الضرب الكارتيزي لجميع الجداول المشاركة في جملة الستفسار الرئيسية وال خارجية ثم لكل سطر من اسطر نواتج الضرب يتم تنفيذ جملة الإستعلام الداخلية. بالطبع يمكن لجملة الإستعلام الداخلية ان تحوي جملة استفسار اخرى داخلها و بالتالي تطبق نفس الإجراء عليها. في المثال المعطى هنا يتم تقييم جملة الإستعلام الداخلية لكل سجل من السجلات الموجودة في جدول الطالب ويتم تقييم الشرط.

كمثال على جمل الإستعلام المتداخلة مايلي:

اوجد اسماء الطلاب اللذين درسوا مساق ساعاته المعتمدة تساوي اربع ساعات.

```
SELECT sname
FROM student
WHERE sid IN ( SELECT sid
                FROM enroll
                WHERE cid IN ( SELECT cid
                              FROM cours
                              WHERE credit = 4 ) )
```

جملة الإستعلام الداخلية تعطينا ارقام المساقات ذات 4 ساعات معتمدة (في المثال مساق رقم 201 و 202) جملة الإستعلام الوسطية تعطينا ارقام الطلبة المسجلين في هذه المساقات (في المثال الطالب رقم 29) اما جملة الإستعلام الخارجية فتعطينا اسماء الطلبة في المجموعة {29} و هو الطالب المدعو توفيق.

لنجد اسماء الطلبة اللذين لم يدرسوا مساقات ذات اربع ساعات معتمدة، بكل بساطة نضيف كلمة NOT قبل كلمة IN في جملة الإستعلام الخارجية.

```
SELECT sname
FROM student
WHERE sid NOT IN ( SELECT sid
                    FROM enroll
                    WHERE cid IN ( SELECT cid
                                   FROM course
                                   WHERE credit = 4))
```

اذا عدلنا جملة لاستفسار اعلاه ووضعنا كلمة النفي في الجملة الوسطية، قد يحدث تغيير على النتيجة. الجملة الداخلية تعطينا ارقام المساقات ذات اربع ساعات معتمدة و بالتالي الجملة الوسطية ستعطينا ارقام الطلاب اللذين درسوا مساقات غير المساقات ذات الاربع ساعات معتمدة. اي ان الطلاب اللذين لم يدرسوا اي مساق بعد لن يظهروا في النتيجة. اما اذا اضفنا النفي على الجملتين الخارجية و الوسطية فان النتيجة تكون اسماء الطلاب اللذي لم يدرسوا مساقات غير المساقات ذات الاربع ساعات اي اللذين درسوا فقط مساقات ذات اربع ساعات معتمدة او لم يدرسوا اي مساق بعد.

جمل الإستعلام المتداخلة المترابطة

في الامثلة السابقة كانت الجمل الداخلية مستقلة وغير مترابطة مع الجمل الخارجية، و هذا ليس شرطا لصحة الجملة بل يمكن ان تعتمد الجملة الداخلية على السطر او السجل الجاري فحصه في الجملة الخارجية لكي نقيم الجملة الداخلية.

مثال: اوجد اسماء الطلاب اللذين درسوا المساق رقم 101

```

SELECT S.sname
FROM student S
WHERE EXISTS ( SELECT *
                FROM enroll E
                WHERE E.cid =101
                AND E.sid = S.sid )

```

كلمة EXISTS من العمليات التي تطبق على المجموعات وتفحص فيما اذا كانت المجموعة خالية ام لا. في هذا المثال نفحص فيما اذا وجد سطر على الاقل يحقق الشرط أن رقم المادة هو 101 و $E.sid = S.sid$ ، اذا كان هذا الشرط صحيح فان الطالب قد درس المادة التي تحمل الرقم 101. جملة الإستعلام الداخلية تعتمد بشكل واضح على السطر الحالي المسمى S و يجب ان يعاد احتسابه لكل سطر في الجدول student. وجود S (على شكل S.sid) في الجملة الداخلية (وهو ينتمي الى الجملة الخارجية) هو الارتباط بين الجملتين لذا نسمي الجملة بالمتراطة.

نلاحظ في المثال احد استخدامات *. هنا لسنا معنيين بحقل ما، فقط نريد ان نعلم اذا كان السطر موجود ام لا بغض النظر عن الحقول الموجودة به.

يمكن استخدام النفي NOT EXISTS، في حال استخدامها هنا تعطينا اسماء الطلاب اللذين لم يدرسوا مساق رقم 101.

ادوات مقارنة المجموعات

بالإضافة الى الادوات المذكورة اعلاه (EXISTS , IN) تعطينا لغة SQL الادوات ANY و ALL التي تستخدم مع ادوات المقارنة (> , < , = , <= , >= , <>) هذه الادوات تستخدم في مقارنة مجموعة من القيم بدلاً من مقارنة قيمة مفردة.

مثال : اوجد اسماء الطلاب اللذين اعمارهم اكبر من عمر طالب يدعى احمد

```
select sname
from student
where age > any(select age
                 from student
                 where sname='Ahmad' )
```

إذا وجد أكثر من طالب اسمائهم احمد فان الإستعلام يعطينا جميع الطلاب اللذين اعمارهم اكبر من احد الطلاب اللذي اسمها احمد، اي ان اذا كان الطالب اكبر من اي طالب اخر اسمه احمد موجد في الجدول سيكون جزء من النتيجة. في المثال اعلاه النتيجة ستكون:

```
sname
-----
Mohammed
Tawfiq
Ruba
Tareq
Ahmad
```

إذا أعطتنا الجملة الداخلية مجموعة خالية فان النتيجة النهائية تكون خالية لان المقارنة مع المجموعة الخالية تعطي "خاطيء". لنفهم مقارنة المجموعات يمكن ان نفكر بها و كأنها مقارنة

متكررة اي نقارن مع كل عنصر من عناصر المجموعة اذا أعطتنا احد المقارنات النتيجة صائبة
فان نتيجة المقارنة صائبة.

مثال: اوجد اسماء الطلاب اللذين اعمارهم اكبر من عمر كل الطلاب المدعوون احمد

يمكن تنفيذ الإستعلام باستخدام نفس الصيغة في المثال السابق مع تغيير كلمة ALL مكان

ALL

```
SELECT sname
FROM student
WHERE age > ALL ( SELECT age
                  FROM student
                  WHERE sname = 'Ahmad')
```

نتيجة الإستعلام على الجداول اعلاه هو الجدول:

sname

Mohammed

Tawfiq

Ruba

Tareq

مثال اخر يوضح المفهوم نفسه:

مثال: اوجد اكبر طالب عمراً

```
SELECT sname
FROM student
WHERE age >= ALL ( SELECT age
                  FROM student)
```

الإستعلام الفرعي (الداخلي) يعطينا اعمار الطلاب، وشرط الإستعلام الخارجي يحدد اسماء الطلاب بشرط ان اعمارهم اكبر من جميع اعمار الطلاب او يساويها، اي انه اذا كان اصغر من عمر طالب من الطلاب لا يعد من النتيجة. في المثال امعطى النتيجة هي الاتي:

sname

Mohammed

Tawfiq

امثلة اخرى حول جمل الإستعلام المتداخلة

مثال: اوجد اسماء الطلاب اللذين درسوا مساق " calculus 1 " و " calculus 2 "

راينا حل لهذا الإستعلام عن طريق INTERSECT يمكننا الان ان نجد طريقة اخرى:

```
SELECT sname
FROM student s, enroll e, course c
WHERE s.sid = e.sid
AND c.cid = e.cid
AND cdesc = 'calculus 1'
AND s.sid IN ( SELECT s2.sid
                FROM student s2, enroll e2, course c2
                WHERE s2.sid = e2.sid
                AND c2.cid = e2.cid
                AND cdesc = 'calculus 2')
```

يمكن تفسير جملة الإستعلام باسترجاع اسماء الطلبة المسجلين في مساق calculus 1 ارقامهم موجودة في مجموعة ارقام الطلبة المسجلين في مساق calculus 2. بهذه الطريقة يمكن ان نستبدل INTERSECT بـ IN. و بنفس الطريقة يمكن استبدال EXCEPT بـ NOT IN حيث ان بعض الانظمة لا توفر الدعم الكامل للغة SQL.

مثال: اوجد اسماء الطلبة المسجلين في جميع المساقات

```
SELECT sname
FROM student s
WHERE NOT EXISTS (
    SELECT c.cid
    FROM course c
    WHERE c.cid NOT IN (
        SELECT e.cid
        FROM enroll e
        WHERE e.sid = s.sid))
```

لا حظ ان الإستعلام مترابط لكل طالب نفحص فيما اذا كانت المساقات المدروسة هي جميع المساقات الموجودة في جدول المساقات.

الفصل الرابع عشر: دوال التجميع

دوال التجميع Aggregate Operators

بالإضافة الى استرجاع البيانات تدعم لغة SQL العمليات الحسابية على البيانات لاستخراج ملخصات مثل المجاميع. بالإضافة الى العمليات الحسابية التي تتم على مستوى السطر او السجل لغة SQL تعطينا امكانية حساب يدخل بها مجموعة من الاسطر لاجراء نتائج تلخيصية. هناك خمسة عمليات تجميعية اساسية تستخدم مجموعة من الاسطر في الجداول لاعطاء نتيجة ما هي:

1-COUNT: تعطينا عدد القيم في العمود او الاعمدة و في حال استخدام DISTINCT تصبح عدد القيم المختلفة.

2-SUM: مجموع القيم في العمود و في حالة استخدام DISTINCT تصبح مجموع القيم المختلفة.

3-AVG: معدل القيم في العمود و في حالة استخدام DISTINCT تصبح معدل القيم المختلفة.

4-MAX: القيمة العظمى في العمود.

5-MIN: القيمة الصغرى في العمود.

مثال: اوجد معدل اعمار الطلاب

```
SELECT avg( age )
FROM student
```

النتيجة:

avg(age)

21.2500

لاحظ ان جملة الإستعلام تستخدم جميع الاسطر في الاوجدول لاعطاء قيمة واحدة.

يمكن استخدام جملة WHERE لادخال شرط على القيم المراد تجميعها

مثال: اوجد معدل معدلات الطلاب اللذين اعمارهم تساوي 20

```
SELECT avg(`avg-marks` )
FROM student
WHERE age =20
```

باستخدام MIN و MAX يكن ايجاد عمر اكبر طالب سنا و عمر اصغرهم سنا، او اعلى معدل

او اقل معدل.

مثال : اوجد اسم الطالب صاحب اعلى معدل

انظر لهذه المحاولة الخاطئة لحل المثال

```
SELECT sname, max(`avg-marks` )
FROM student
```

نحاول هنا معرفة اسم الطالب صاحب اعلى معدل، لكن هذه الجملة خاطئة في لغة SQL.

حيث تعطي خطأ من مثل الخطأ التالي:

```
ERROR XXX : Mixing of GROUP columns
(MIN(),MAX(),COUNT(),...) with no GROUP columns is illegal if there
is no GROUP BY clause
```

عند استخدام عملية تجميع يجب ان تكون جميع جملة select تحوي ادوات تجميع واي حقل سيظهر يجب ان يكون جزء من جملة GROUP BY التي سناتي على ذكرها لاحقا. اي انه لا يمكن ان نستخدم sname مع MAX(age). لايجاد هذا الإستعلام علينا ان نستخدم الجملة المتداخلة كالآتي:

```
SELECT sname
FROM student
WHERE `avg-marks` = (
    SELECT max(`avg-marks`)
    FROM student )
```

لاحظ اننا استخدمنا = مع جملة select دون استخدام ادوات المقارنة للمجموعات اي اننا اعتبرنا نتيجة الإستعلام الفرعي هي قيمة واحدة وذلك لاننا نستخدم دالة تجميع اللذي يضمن لنا حصولنا على قيمة واحدة فقط.

مثال: اوجد عدد الطلاب

```
SELECT count( * )
FROM student
```

في هذا المثال نرى استخدام اخر ل * مما يعطينا امكانية عد جميع الاسطر بسهولة، كما اسلفنا فان * تعني جميع الحقول.

مثال: اوجد عدد الاسماء المختلفة للطلاب

```
SELECT count( DISTINCT sname )
FROM student
```

المثال الحالي لا يعطينا نفس العدد الذي يعطينا المثال السابق اذا كان هنالك تشابه (او تطابق) اسماء.

جمل GROUP BY و HAVINGمثال: اوجد اعلى معدل لكل عمر من اعمار الطلاب

اذا كنا نعلم ان اعمار الطلاب هي ارقام صحيحة ضمن مدى محدد يمكننا ان نقوم بتكرار جملة الإستعلام التالية بعدد الاعمار المختلفة لكي ننفذ الإستعلام المطلوب:

```
SELECT max(avg-marks)
FROM student
WHERE age = i
```

حيث ان i هي الاعمار المختلفة (مثلا 18,19,25...)، كتابة ثمانية جمل هو ليس بالعمل المريح، و الاهم من ذلك اننا لا نعلم الاعمار المختلفة الموجودة في قاعدة البيانات سلفا بالاضافة الى ان قاعدة البيانات تتغير باستمرار اي اننا يجب ان نكتب برامج جديدة عند تعديل البيانات في قاعدة البيانات. ولكتابة مثل هذا الإستعلام نحتاج الى GROUP BY وهو اضافة مهمة على لغة SQL. الصيغة العامة لجملة الإستعلام مع هذه الاضافة هي:

```
SELECT [DISTINCT] select-list
FROM from-list
WHERE qualification
GROUP BY grouping-list
HAVING group-qualification
```

لحل المثال باستخدام GROUP BY يمكن كتابة الإستعلام كلاتي:

```
SELECT age, max(avg-marks)
FROM studet
GROUP BY age
```

```
age max(`avg-marks` )
```

```
19 78
```

```
20 76
```

```
21 81
```

```
22 79
```

```
23 78
```

يجب اخذ الامور التالية بعين الاعتبار:

- تحوي Select-list في العبارة SELECT على اسماء حقول والعمليات التجميعية وعادتا ما تلحقها AS لتعطي النتيجة اسم جديد. (مثال AS max(avg-marks) Average). كل الحقول المدرجة في عبارة SELECT يجب ان تظهر في عبارة GROUP BY، وذلك لان كل سطر في النتيجة يعبر عن مجموعة، والتي هي تجميع لاسطر تحمل حقولها المدرجة في عبارة GROUP BY نفس القيمة. الحقل الظاهر في SELECT والغير ظاهر في GROUP BY لا يمكن ان نعرف ما هي القيمة التي يجب ان يحملها اذ يمكن ان يحمل قيم مختلفة للاسطر الممثلة في المجموعة لذا يعطينا خطأ.

- التعابير الظاهرة في عبارة HAVING يجب ان تحمل قيمة واحدة لكل سطر من الاسطر في النتيجة (اي لكل مجموعة). عبارة HAVING تحدد الاسطر التي ستظهر

في النتيجة (اي مجموعات الاسطر). اي ان الحقول التي تظهر في عبارة HAVING

يجب ان تظهر في عبارة GROUP BY.

- عند حذف عبارة GROUP BY كل الجدول يعتبر كمجموعة واحدة.

مثال: اوجد معدل علامات الطلاب من كل مستوى (level) بحيث تكون علامة

الطالب اكبر من 65% و عدد الطلاب التي تنطبق عليهم الشروط اكثر من احد.

```
SELECT level, avg(`avg-marks`) AS average
FROM student
WHERE `avg-marks` >70
GROUP BY level HAVING count(*) >1
```

لنحاول تطبيق هذه الجملة على الجدول اعلاه، الخطوة الاولى هي الضرب الكارتيزي و بما انه لا

يوجد سوى جدول واحد فان النتيجة تصبح كالآتي:

sid	sname	age	avg-marks	level
21	Ahmad	20	66	3
27	Mohammed	23	78	4
29	Tawfiq	23	72	4
34	Ammar	19	78	1
36	Mahmoud	20	76	2
46	Ruba	22	79	3
48	Tareq	22	67	3
54	Ahmad	21	81	2

الخطوة الثانية هي ان نطبق الشروط الموجدة في جملة `WHERE `avg-marks` >70` لتصبح النتيجة كالاتي:

sid	sname	age	avg-marks	level
27	Mohammed	23	78	4
29	Tawfiq	23	72	4
34	Ammar	19	78	1
36	Mahmoud	20	76	2
46	Ruba	22	79	3
54	Ahmad	21	81	2

الخطوة الثالثة تحذف الاعمدة الغير مرغوب بها فقط الاعمدة التي تظهر في عبارة `SELECT` و عبارة `group by` و عبارة `HAVING` اي يمكن ان نحذف `sid,sname, age` لتصبح النتيجة كالاتي:

level	avg-marks
4	78
4	72
1	78
2	76
3	79
2	81

الخطوة الرابعة نرتب النتيجة حسب level لتصبح النتيجة كالتالي:

level avg-marks

1 78

2 76

2 81

3 79

4 78

4 72

الخطوة الخامسة نطبق عمليات التجميع الموجودة في عبارة $\text{HAVING count} (*) > 1$ اي

يتم عد الاسطر في كل مجموعة لها نفس قيمة الحقل المذكور في عبارة GROUP BY اي

level. يتم حذف المجموعة level=1 و level=3 لان عدد الاسطر في كل منهما هو واحد

فقط. لا حظ اهمية ترتيب تنفيذ WHERE و HAVING ، في حالة تغيير الترتيب فان

النتيجة قد تختلف، في المثال هنا فان level=3 يظهر في النتيجة لانه يحقق الشرط الموجود في

.HAVING

الخطوة السادسة هي ايجاد سطر واحد لكل مجموعة (level في هذا المثال) يطبق فيه نتيجة

العملية التجميعية الموجودة في SELECT. يكون في سطر النتيجة الحقول المذكورة في عبارة

SELECT بالاضافة لنتيجة العمليات التجميعية. في المثال النتيجة تحتوي على عمودين هما

level و average والذي تم احتسابه بتطبيق avg على حقل avg-marks، النتيجة النهائية:

level average

2 78.5

4 75.0

مثال: اوجد عدد الطلاب الدارسين لكل مساق عدد ساعاته المعتمدة هي 3

```
SELECT e.cid, count( e.sid ) as number-of-students
FROM `enroll` e, course c
WHERE e.cid = c.cid and c.credit = 3
GROUP BY e.cid
```

لاحظ ان الجملة التالية خاطئة

```
SELECT e.cid, count( e.sid ) as number-of-students
FROM `enroll` e, course c
WHERE e.cid = c.cid
GROUP BY e.cid
HAVING c.credit = 3
```

فقط الحقول التي تظهر في GROUP BY يمكن ان تظهر في HAVING الا اذا كانت

باراميتر لدالة تجمعية.

مثال: اوجد المستويات التي لها اقل معدل من بين باقى معدلات العلامات فى

المستويات كافة

```
SELECT temp1.level, temp1.average
FROM (SELECT s.level , avg( s.`avg-marks` ) AS average
      FROM student s
      GROUP BY level ) AS temp1
WHERE temp1.average = ( SELECT min( temp2.average )
                       FROM (SELECT level , avg( `avg-marks` ) AS average
                             FROM student
                             GROUP BY LEVEL) AS temp2 )
```

القيم الخالية NULL values

عملياً لا يمكننا دائماً تحديد قيم جميع الحقول، فبعض الحقول لا تكون قيمها معروفة لدينا في وقت من الاوقات، فمثلا لا يمكننا تحديد علامة طالب جديد لم يدرس مساقات بعد. وفي احيان اخرى لا تكون قيمة الحقل ذات معنى في مواقع معينة، فمثلا بعض المساقات ليس لها متطلب سابق فلا يمكن ان نضع اي قيمة مكان المتطلب السابق.

تعطينا لغة SQL قيمة خاصة وهي غير الصفر او الفراغ و هي القيمة الخالية او ما يسمى NULL لاستخدامها في مثل تلك المواقع. وجود القيمة الخالية تعقد بعض الامور، التي سندرسها هنا.

على سبيل المثال: عند مقارنة حقل علامة الطالب مع قيمة محددة ماذا يجب ان تكون النتيجة عندما يكون الحقل فارغ (avg-marks=60) او avg-marks>60 هل القيمة الفارغة اكبر من ام اصغر من 60 ربما يقترح البعض ان تكون النتيجة غير معرفة اي غير صائبة وغير خاطئة.

هنا اصبح لدينا منطق رياضي بثلاث قيم (صائبة، خاطئة و غير معرفة) لذا عند استخدام OR, AND, NOT يتحتم علينا تغيير طريقة التقييم لتصبح كالآتي:

- NOT مع الغير معرف تعطينا غير معرف
- اذا كانت القيم بينها OR اذا كان احد الحدود صائب تصبح النتيجة صائبة انما اذا كانت جميع القيم خاطئة و غير معرفة فان النتيجة غير معرفة
- اذا كان لدينا AND فان وجود قيمة واحدة بين الحدود غير معرفة تعطينا النتيجة غير معرفة.

توفر لنا لغة SQL اداة لفحص القيم الخالية وهي IS NULL حيث تعطينا قيمة صائبة اذا كانت القيمة خالية والعكس بالعكس بالاضافة الى اننا يمكن ان ننفىها IS NOT NULL.

كما اشرنا سابقا جملة WHERE تلغي جميع الاسطر التي لا تعطينا القيمة "صائب" (في الجبر البولي) عند تطبيق الشروط عليها اي انه مع وجود القيمة الخالية فان جميع الاسطر التي تعطينا خطأ او غير معرف سوف يتم الغائها من النتيجة. هذه النتيجة لها تاثير كبير على نتائج جمل الإستعلام خصوصا عند استخدام الجمل المتداخلة و التي تحتوي على EXISTS.

جميع العمليات الحسابية تعطي القيمة الخالية اذا كان احد حدودها قيمة خالية. اما القيم الخالية مع الدوال التجميعية قد تعطي نتائج غير متوقعة احيانا. على سبيل المثال count(*) تتعامل مع القيمة الخالية كاي قيمة اخرى اي يتم عدها. اما العمليات الاخرى (COUNT, SUM,) (AVG,MIN,MAX) فانها تتجاهل وجود القيمة الخالية، اي لا تدخلها في الحساب.

الربط الخارجي Outer Joins

الربط العادي حينما يكون هناك سطر ليس له اي مقابل حسب شرط الربط في الجدول الاخر لا يظهر السطر، لغة SQL تعطينا امكانية اخرى باستخدام القيم الخالية وهي ان يظهر كل سطر في الجدول الاول مرة على الاقل حتى ولو لم يكن له مقابل في الجدول الاخر مع وجود قيم خالية مكان الحقول القادمة من الجدول الثاني التي ستظهر في النتيجة. على سبيل المثال الطلاب اللذين لم يدرسوا اي مساق لا يوجد لهم في جدول enroll اي ذكر حينما يتم عمل ربط عادي بين student و enroll فانهم لن يظهروا في النتيجة. اما في حالة الربط الخارجي فان الطلاب اللذين لم يدرسوا اي مساق سيظهروا في النتيجة مع قيمة المساق المدروس تساوي القيمة الخالية.

الربط الخارجي له عدة اشكال هناك ربط من اليسار left outer join جميع الاسطر من الجدول الأول التي ليس لها مقابل في الجدول الثاني ستظهر في النتيجة وليس العكس والثاني الربط من اليمين right outer join، عكس الاول اي ان الاسطر في الجدول الثاني التي ليس لها مقابل في الجدول الاول تظهر في النتيجة مرة واحدة ومع وجود القيم الخالية مكان الحقول القادمة من الجدول الاول و ليس العكس. اما الاخير الربط الخارجي الكامل full outer join، اي سطر من كلا الجدولين ليس له مقابل في الجدول الاخر يظهر في النتيجة مع ما يقابله من قيم خالية. بالطبع كل الاسطر التي لها مقابل في الجدول الاخر تظهر في النتيجة لكل انواع الربط بما فيها الربط العادي والذي يسمى احيانا الربط الداخلي.

مثال: لدينا الجدولين التاليين (نفس الامثلة السابقة)

الطالب

sid	sname	age	avg-marks	level
21	Ahmad	20	66	3
27	Mohammed	23	78	4
29	Tawfiq	23	72	4
34	Ammar	19	78	1
36	Mahmoud	20	76	2
46	Ruba	22	79	3
48	Tareq	22	67	3
54	Ahmad	21	81	2

المقررات

cid	sid	mark
10	21	70
11	21	69
10	27	78
11	27	75
101	27	79
102	27	60
101	29	70
201	29	77

عمليات الربط المختلفة على الجدولين تكون كمايلي:

```
SELECT student.sid, enroll.cid
FROM student
NATURAL LEFT OUTER JOIN enroll
```

كلمة NATURAL تعني ان الربط هو تساوي بين جميع الحقول المشتركة في الجدولين. جملة

WHERE غير ضرورية الا اذا اردنا اضافة شروط غير شروط الربط.

sid cid

21 10

21 11

27 10

27 11

27 101

27 102

29 101

29 201

34 NULL

36 NULL

46 NULL

48 NULL

54 NULL

جملة JOIN تاخذ عدة صيغ بالاضافة الى الصيغة اعلاه فيمكن ان نستخدمها بدون كلمة

NATURAL وبالتالي يجيب ان نحدد حقول الربط و يمكن ان نبدل كلمة OUTER بكلمة

.INNER

مثال:

```
SELECT student.sid, enroll.cid  
FROM student INNER JOIN enroll  
ON student.sid= enroll.sid
```

تكافئ التالي باستخدام WHERE

```
SELECT student.sid, enroll.cid  
FROM student , enroll  
WHERE student.sid= enroll.sid
```

الفصل الخامس عشر: الجداول

انشاء وحذف الجداول

انواع البيانات

البيانات المخزنة في قاعدة البيانات تكون لها انواع. يمكن ان نعرف الانواع بمجموعة من القيم المحددة. مثال الارقام الصحيحة هي نوع من انواع البيانات المحددة فالقيم مثل 45 و -1 هي قيم من نوع الارقام الصحيحة.

لكل حقل في كل جدول يخزن قيم من نوع محدد. انظمة ادارة قواعد البيانات المختلفة تعرف انواع مختلفة من البيانات. الانواع الاساسية المعيارية المعرفة في معيار ISO/ANSI هي التالي:

الحروف CHARACTER : الحقل من هذا النوع يخزن سلاسل من الحروف يتم تحديد حجمها (عدد الاحرف) عند تعريف الجدول. يتم حجز حجم ثابت في قاعدة البيانات لكل حقل من هذا النوع بغض النظر عن عدد الاحرف المدخلة فعلا في الحقل. يسمى اختصارا CHAR

الحروف المتغيرة CHARACTER VARYING: مشابهه للنوع اعلاه لكن الحجم المحدد عند انشاء الجدول هو الحد الاعلى للاحرف التي يمكن ان تخزن في الحقل و ليس مساحة التخزين التي ستم حجزها للحقل. يقوم النظام بشكل ديناميكي بحجز مساحة التخزين حسب عدد الاحرف المخزنة في الحقل فعليا. يستخدم الاسم VARCHAR للدلالة على هذا النوع من الحقول.

NUMERIC: هذا النوع من الحقول يخزن الارقام. يمكن تحديد عدد خانات الرقم NUMERIC(5) يخزن حتى 99999 و يمكن تحديد الدقة NUMERIC(3,2) يخزن حتى الرقم 9.99 اي ثلاث خانات و منها خانتان بعد الفاصلة العشرية.

DECIMAL: شبيه جدا ب NUMERIC و غالبية الانظمة تعتبرهما نوع واحد.

INTEGER: شبيهه بالسابق لكن لا يوجد ارقام بعد الفاصلة العشرية. يمكن اختصاره ب INT

SMALINT: نفس السابق الا انه يكون اصغر في بعض الاحيان.

BIT: تحتوي على عدد محدد من البتات و لا يقوم نظام ادارة البيانات بتفسير معنى البيانات

المخزنة في الحقل. التطبيق يحدد معنى البيانات. (يسمى RAW في Oracle)

BIT VARYING: كما سبق لكن لا يقوم النظام بحجز مساحة التخزين مسبقا بل يقوم

بتحديدها عند ادخال البيانات بشكل ديناميكي.

FLOAT: تحتوي الحقول من هذا النوع ارقام حقيقية. يمكن تحديد الدقة و المدى كما هو في

NUMERIC

REAL: مشابه للسابق لكن بلا تحديد دقة او مدى.

DOUBLE PRECISION: كما السابق لكن في بعض الانظمة له دقة اكبر.

DATE: يخزن تاريخ يوم محدد.

TIME: يخزن وقت محدد. اي يخزن الساعة والدقيقة و الثانية. يمكن تخزين اجزاء من الثانية

ايضا بالاضافة لامكانية مصاحبة الوقت برمز منطقة التوقيت. (مثلا القدس في منطقة التوقيت

(2+

TIMESTAMP: يخزن هذه الحقول التاريخ و الوقت بعض التطبيقات تسميه DATETIME.

INTERVAL: يخزن فترات زمنية. على سبيل المثال 3 ايام او 10 اشهر.

بالطبع يمكن تخزين القيمة الخالية NULL في اي نوع من الانواع فهي ليس لها نوع محدد.

انشاء الجداول

يمكن انشاء جداول بالصيغة العامة التالية:

```
CREATE TABLE <table>
(<column description>)
```

الكلمتان الأولتان CREATE TABLE هي امر انشاء الجدول ويأتي بعدها إسم الجدول المراد إنشائه. ثم داخل الاقواس نضع أسماء و أنواع الأعمدة أو الحقول المكونة للجدول. نفصل بين حقل وآخر بفاصلة عادية.

مثال لإنشاء الجدول student في الأمثلة السابقة:

```
CREATE TABLE student
( sid number(4) ,
sname varchar(40),
age number (4),
avg-marks number(3,1)
level number(1) )
```

تنفيذ هذه الجملة ينشيء جدول فارغ به خمسة اعمدة لكن لا يوجد به بيانات (لا يوجد به اسطر)، يمكن تخزين ارقام في جميع الاعمدة عدى عمود name (الاسم) يمكن تخزين احرف به حتى 40 حرف كحد اقصى. اي لا يمكن ان يتجاوز طول الاسم 40 حرف (مع الفراغات). يمكن ان يكون إسم الحقل أي شيء يبدأ بحرف ولا يحتوي على فراغات. بشكل عام كتابة الحقول كل على سطر مستقل لا يؤثر على الجدول المنشأ وضعناه هنا فقط لكي يتم قراءة الجملة بشكل مريح. جميع الفراغات وضعت لنفس السبب و هو تسهيل قراءة الجملة عدى الفراغ اللذي يتبع اسم الحقل و ذلك لكي نفصل نهاية اسم الحقل عن بداية نوع الحقل.

مثال آخر جدول course يمكن إنشائه بتنفيذ الجملة التالية:

```
CREATE TABLE course
( cid` int(11),
  cdesc` varchar(30),
  credit` smallint(4) ,
  prerequisite` int(11) )
```

يمكن تغيير بنية الجدول بعد انشاءه، حيث يمكننا ان نضيف او نلغي اعمدة باستخدام جملة ALTER TABLE. على سبيل المثال يمكننا ان نضيف رقم الهاتف على جدول student

كما يلي:

```
ALTER TABLE student ADD phone CHAR (10);
```

و يمكننا ان نلغي العمود نفسه باستخدام الصيغة التالية:

```
ALTER TABLE student DROP phone;
```

لتغيير نوع الحقل

```
ALTER TABLE student
MODIFY phone VARCHAR(15);
```

لتغيير اسم الحقل (ORACLE 9)

```
ALTER TABLE student
RENAME COLUMN phone TO phone_num;
```

لتغيير اسم الجدول

```
ALTER TABLE student
RENAME TO students;
```


منع القيمة الخالية في الاعمدة

اثناء انشاء الجداول يمكننا ان نمنع بعض الاعمدة او الحقول من ان تاخذ قيم خالية. حينما نضع هذا المحدد يتم منع ادخال قيمة خالية في الحقل. فمثال اذا اردنا منع ادخال قيمة خالية مكان رقم الطالب تكون جملة انشاء الجدول كالاتالي:

```
CREATE TABLE student
( sid      number(4) NOT NULL,
  sname    varchar(40),
  age      number (4),
  avg-marks number(3,1)
  level    number(1) )
```

حيث اضفنا عبارة NOT NULL بعد نوع الحقل. النتيجة هي مطابقة لانشاء الجدول في المثال اعلاه مع فرق وحيد انه حينما نقوم بادخال سطر جديد على الجدول لا يمكن ان يكون حقل رقم الطالب قيمة خالية.

القيم الفريدة

عند انشاء الجدول يمكننا ان نحدد ان القيم في عمود ما لا يمكن ان تتكرر. فعلى سبيل المثال اذا اردنا ان نحدد لكل طالب رقم خاص به لا يمكن ان يشترك به مع اخر، يمكننا ان نستخدم الكلمة UNIQUE كما يلي:

```
CREATE TABLE student
( sid      number(4) NOT NULL,
  sname    varchar(40),
  age      number (4),
  avg-marks number(3,1)
  level    number(1) ,
  UNIQUE KEY sid (sid))
```

اي محاول لادخال نفس الرقم في حقل رقم الطالب مرتين يتم رفضها من النظام و اعطاء رسالة خطأ تفيد بذلك. كل حقل يتم اعلانه باستخدام UNIQUE يجب ان يتم اعلانه ايضا باستخدام .NOT NULL. يمكن ان نحدد اي عدد من الحقول في الجدول نفسه بالمحدد UNIQUE. فيمكننا ان نمنع تكرار اسم المساق و رقمه كما يلي:

```
CREATE TABLE course
( cid int(11) NOT NULL,
  cdesc varchar(30) NOT NULL,
  credit smallint(4) ,
  prerequisit int(11) ,
  UNIQUE KEY cid (cid),
  UNIQUE KEY cdesc (cdesc) )
```

كيفية تحديد المفاتيح بلغة SQL

تستخدم لغة SQL العبارة UNIQUE لتحديد ان مجموعة من الحقول تكون مفتاحا مرشحا. واحد من المفاتيح يمكن ان يعرف مفتاح رئيسي باستخدام PRIMARY KEY. لا تقبل اللغة تعريف اكثر من مفتاح رئيسي للعلاقة الواحدة. لنرجع للمثال السابق و نضيف له المفاتيح المطلوبة:

```
CREATE TABLE students (
sid CHAR( 20 ),
name CHAR( 30 ),
login CHAR( 20 ),
age INT,
avg FLOAT ,
UNIQUE (name, age),
CONSTRAINT Studentkey PRIMARY KEY (sid) )
);
```

جملة انشاء الجدول تعني ان sid هو المفتاح الرئيسي و الاسم مع العمر معا يشكلان مفتاحا مرشحا. نظهر في المثال ايضا استخدام اسم المحدد. يفيد هذا الاسم حينما نحاول ادخال معلومات تتعارض مع هذا المحدد. في هذه الحالة يقوم النظام بتحديد اسم المحدد الذي تتعارض معه البيانات.

مثال: لانشاء جدول المدرسين TEACHERS

```
CREATE TABLE teachers
( teacher_num SMALLINT NOT NULL,
teacher_name CHAR(18),
phone CHAR(10),
salary FLOAT)
UNIQUE (teacher_num)
```

كيفية تحديد المفاتيح الاجنبي بلغة SQL

لنعرف العلاقة enrolled(sid : string, cid : string, grade : string)

```
CREATE TABLE enrolled ( sid char(20),
cid char(20),
grade char(10),
PRIMARY KEY(sid, cid),
FOREIGN KEY (sid) REFETRENCES students)
```

محدد المفتاح الاجنبي FOREIGN KEY في هذا المثال يعني ان كل قيمة تظهر في الحقل sid في العلاقة enrolled يجب ان تظهر في sid في العلاقة student. اما محدد المفتاح الرئيسي PRIMARY KEY فيعني ان لكل طالب علامة واحدة في كل مساق. اذا اردنا ان

نضع اكثر من علامة للكتاب في نفس المساق (المساقات المعادة على سبيل المثال) يجب ان نغير المحدد المذكور.

محددات باستخدام CHECK

يمكننا تحديد محددات وشروط على الجدول باستخدام مايسمى بمحددات الجدول في لغة SQL باستخدام عبارة CHECK في تعريف الجدول CREATE TABLE.

على سبيل المثال لكي نضمن ان مستوى الطالب يتراوح بين 1(طالب جديد) و 4 (طالب خريج) يمكن ان نكتب الاتي:

```
CREATE TABLE students (
sid CHAR( 20 ),
name CHAR( 30 ),
login CHAR( 20 ),
age INT,
avg FLOAT ,
UNIQUE (name, age),
CONSTRAINT Studentkey PRIMARY KEY (sid) ,
CHECK (level >=1 and level <=4)
);
```

حذف الجداول

تعطينا لغة SQL امكانية حذف الجداول عن طريق جملة DROP ذات الصيغة العامة التالية:

```
DROP TABLE <table>;
```

طبعا يتم وضع اسم الجدول المراد حذفه مكان <table> على ان يكون موجودا في قاعدة البيانات. هذه الجملة لها مفعول عكسي لجملة انشاء الجدول (تسمى احيانا تدمير الجدول). حيث يتم حذف تعريف الجدول نهائيا من قاعدة البيانات و لا يمكن عكس فعله. لذا كن حذرا عند استخدام هذه الجملة.

مثال لحذف جدول student

```
DROP TABLE student;
```

انشاء جدول من جدول اخر

يمكن انشاء جدول من جدول اخر بنسخ الاعمدة من الجدول الاول. يمكن نسخ جميع الحقول من الجدول الاول كما يلي:

```
CREATE TABLE new_table
AS (SELECT * FROM old_table);
```

مثال:

```
CREATE TABLE graduate_student
AS (SELECT *
FROM student
WHERE level > 4);
```

هنا يتم انشاء جدول جديد اسمه graduate_student له نفس وصف الجدول القديم وبه نسخة جزئية من المعلومات الموجودة في الجدول القديم حسب ما تحدده جملة الإستعلام. أي أن الجملة هي جملة انشاء جدول بالاضافة الى ادخال بيانات الى الجدول.

يمكن ان نحدد بعض الحقول التي نريد نسخها و يمكن ان ننشئ جدول من عدة جداول كل هذا تحدده جملة الإستعلام. أي يتم إنشاء جدول دائم من نتائج جملة الإستعلام.

اضافة حذف وتعديل البيانات في الجداول

اضافة سجلات للجداول

بعد انشاء الجدول يمكننا ان نملأه قيم. جميع القيم المخزنة في الجداول تخزن على شكل سجلات لذا لاضافة قيمة على الجدول يتعين علينا ان نضيف سجل كامل. لاضافة سجل نستخدم جملة

.INSERT

الشكل العام لجملة INSERT هو:

INSERT INTO <table> (<column names>) VALUSE (<values>)

تضيف هذه الجملة سطر على الجدول المسمى <table>. يمكن ان يحتوي السجل الجديد على قيم لكل او بعض الحقول في السجل. يمكن تحديد اسماء الحقول المراد اضافة قيم عليها بعد اسم الجدول و يمكن الاستغناء عن تحديد اسماء الحقول اذا اردنا اضافة قيم على جميع الحقول. ثم تاتي كلمة VALUES متبوعة بالقيم التي نريد ان نضيفها الى السجل.

مثال: لاضافة سجل طالب جديد على جدول students المعرف في اعلاه.

```
INSERT INTO students
(sid, name, login, age, avg)
VALUES ( 303, 'Sulaiman', 'sul303', 19, 70.1 )
```

او

```
INSERT INTO students VALUES ( 303,'Sulaiman','sul303',19,70.1 )
```

كلا الجملتين تضيف سجل الى الجدول.

يجب ان تتلائم انواع القيم المدخلة مع انواع الحقول. على سبيل المثال لا يمكن ان نضع القيمة "غير معرف" الى حقل المعدل avg لان القيمة هي CHAR و الحقل معرف على ان يكون من نوع FLOAT.

السجلات لا تكون مرتبة داخل الجدول اي ان السجل اما ان يكون في الجدول او لا يكون ولا معنى للسؤال اين يوجد السجل داخل الجدول او ما هو ترتيبه. اضافة سجل الى الجدول لا تعني انه يضاف الى اخر الجدول او الى اول الجدول او بعد سجل معين، فقط يصبح السجل عنصر جديد في الجدول.

رأينا في المثال انه يمكن تحديد اسماء الحقول او عدم تحديدها. لكن يجب ان تكون القيم مرتبة حسب الترتيب الذي رتبته به الحقول وقت انشاء الجدول. اذا اردنا ادخال القيم بترتيب غير ترتيب الحقول يترتب علينا ان نعطي قائمة بالحقول مرتبة حسب الترتيب المراد اعطاء القيم به.

مثال:

```
INSERT INTO students
    (name, login, sid, age, avg)
VALUES ( 'Sulaiman','sul303', 303, 19, 70.1 )
```

هنا قمنا بادخال القيم بترتيب غير الترتيب الذي انشأت به الحقول. لاحظ ان الحقول مرتبة و السجلات غير مرتبة.

ماذا يحدث اذا اضفنا قيم في بعض الحقول لسجل جديد؟ مثال:

```
INSERT INTO students
(sid, name, login )
VALUES ( 303,'Sulaiman','sul303' )
```

ما هي القيمة التي سوف ياخذها الحقول age, avg ؟ كل حقل يجب ان ياخذ قيمة لذا فان القيمة التي ستخزن في الحقول التي لم يتم تحديد قيم لها هي NULL. طبعا اذا اردنا ان نضيف سجل يجب ان نعطي قيمة لجميع الحقول التي وضعنا عليها المحدد NOT NULL. لجملة INSERT شكل اخر حيث يمكن اضافة حقول باستخدام جملة استفسار SELECT. الصيغة العامة للجملة هي:

```
INSERT INTO <table>(<column names>)
SELECT <column names>
FROM <table>
WHERE < PREDICATE>
```

مكان كلمة VALUES و قائمة القيم نضع جملة استفسار، يمكن أن تكون جملة الإستعلام بسيطة او معقدة بالشكل الذي نريد لافرق. لا يقوم المستخدم بتحديد القيم المراد ادخالها. يتم اضافة نتيجة الإستعلام من جدول او اكثر الى الجدول المحدد في جملة INSERT. مثلا نفرض اننا نريد اضافة الطلاب الحاصلين على معدل 90 فما فوق الى جدول المدرسين كمساعدي تدريس. طبعا الحقول في جدول الطلاب يختلف عن الحقول في جدول المدرسين. يمكن ادخال اسم الطالب في حقل اسم المدرس و رقم الطالب في حقل رقم المدرس. اما باقي الحقول فتبقى فارغة.

```
INSERT INTO teachers (teacher_num, teacher_name)
SELECT sid, name
FROM students
```



```
WHERE avg >=90
```

عند تنفيذ هذه الجملة يتم اضافة عدة حقول الى جدل المدرسين في كل حقل جديد رقم الهاتف والراتب يكون بهم القيمة الخالية NULL.

تعديل السجلات في الجداول

بعد اضافة السجلات الى الجداول نكون في بعض الاحيان بحاجة الى تعديل القيم الموجودة في السجلات. جملة التعديل في لغة SQL لها الصيغة العامة التالية:

```
UPDATE <table>
SET <column name> = <value>
WHERE <predicate>
```

اسم الجدول المراد تعديل سجلاته يظهر بعد كلمة UPDATE مباشرة. و بعد كلمة SET نضع مجموعة من جمل التخصيص لتعديل القيم الموجودة في الحقول على صيغة (اسم الحقل = القيمة). يجب ان نضع اسم الحقل و بعد اشارة = نضع قيمة ثابتة او تعبير من اي نوع. يمكن في النهاية اضافة عبارة WHERE لتحديد السجلات المراد تعديل حقولها. اذا لم تظهر جملة WHERE يسري التعديل على جميع سجلات الجدول المحدد.

مثال : نريد تعديل معاش (راتب) المدرس رقم 121 الى 800.

```
UPDATE teachers
SET salary = 800
WHERE teacher_num = 121
```

يمكن استخدام التعبيرات الحسابية لتعديل القيم في الحقول

مثال: نريد اضافة غلاء معيشة بنسبة 5% الى راتب المدرس رقم 121

```
UPDATE teachers
SET salary = salary*1.05
WHERE teacher_num = 121
```

لاحظ استخدام اسم الحقل بعد اشارة = حيث تعني القيمة الموجودة في الحقل قبل التعديل.

يمكن تعديل حقلين في الان ذاته

مثال:

```
UPDATE teachers
SET teacher_name = 'Khalil' ,
phone = '12348743'
WHERE teacher_num = '124'
```

لاحظ الفاصلة التي تفصل بين حقل واخر.

يمكن حذف عبارة WHERE لتعديل جميع السجلات.

مثال: لاعطاء علاوة غلاء المعيشة لجميع المدرسين بنسبة 5%

```
UPDATE teachers
SET salary = salary*1.05
```

يمكن ان تكون عبارة WHERE معقدة، فمثلا اذا اردنا ان نرفع معاشات المدرسين اللذين تقل

معاشاتهم عن المعدل بنسبة 7.5% ننفذ الجملة التالية:

```
UPDATE teachers
SET salary = salary*1.075
WHERE salary < (SELECT AVG(salary)
FROM teachers)
```

لاحظ انه يتم تنفيذ الجملة الفرعية اولا لتحديد المعدل ثم تنفيذ الجملة الاساسية. عكس الترتيب يغير من النتيجة لان الجملة الخارجية تغير من قيمة المعدل.

حذف السجلات

الصيغة العامة لالغاء السجلات من الجداول هي:

```
DELETE FROM <table>
```

```
WHERE <predicate>
```

يتم حذف جميع السجلات التي ينطبق عليها شرط عبارة WHERE من الجدول المحدد بعد كلمة FROM. تشبه الى حد ما جملة الإستعلام لكن بدلا من ان نقوم باسترجاع السجلات يتم حذفها. (يمكنك تنفيذ جملة استفسار قبل الحذف للتأكد من السجلات المراد حذفها ثم استبدال عبارة SELECT بكلمة DELETE مع الابقاء على عبارة WHERE.

تقوم جملة DELETE بحذف سجل او اكثر اعتمادا على الشرط المحدد في جملة WHERE مثال اذا اردنا حذف سجل الطالب رقم 234 ننفذ الجملة التالية:

```
DELETE FROM student
```

```
WHERE sid =234
```

يمكن ان نزيد من تعقيد جملة WHERE كما نشاء بنفس الطرق التي طبقناها في جملة الإستعلام.

إذا حذفنا جملة WHERE يتم حذف جميع السجلات الموجودة في الجدول مع الإبقاء على تعريف الجدول أي يتم حذف السجلات دون الغاء وجود الجدول, و يمكننا بعد ذلك إضافة سجلات على الجدول.

مثال:

```
DELETE FROM student
```

يتم حذف جميع السجلات في الجدول student

مثال: احذف جميع السجلات في جدول enroll بحيث يكون حقل العلامة فارغ

```
DELETE FROM enroll
WHERE mark IS NULL
```

مثال: احذف جميع السجلات الموجودة في جدول enroll بحيث لا يكون له سجل

مقابل في جدول student

```
DELETE FROM enroll E
WHERE NOT EXISTS
( SELECT sid from student S
WHERE S.sid= E.sid)
```

مثال: احذف جميع السجلات التي تحتوي على علامات خاطئة في جدول الطلاب student

```
DELETE FROM student
WHERE mark >100 OR mark <35
```

الفصل السادس عشر: المناظر

VIEWS المناظر

الجدول المكونة لقاعدة البيانات لها وجود فعلي مادي، هنالك مساحة على القرص الصلب تشغلها البيانات الموجودة في الجدول. يمكن ان نعرف جداول اخرى بناء على تلك الجداول غير موجودة ماديا ولا تشغل حيز لكن يمكننا ان نتعامل معها وكأنها موجودة حقيقةً هذه الجداول تسمى مناظر .views

المنظر هو عبارة عن جدول مبني من كل او جزء من جدول او اكثر. لان المناظر غير موجودة بشكل مادي تسمى في بعض الاحيان جداول افتراضية virtual tables. و الجداول الاخرى المادية تسمى الجداول الحقيقية او الاساسية. المناظر مثل الجداول يتم انشائها باستخدام جملة CREATE و خلافا للجدول تكون السجلات المكونة لها موجودة مسبقا في قاعدة البيانات. الصيغة العامة لجملة انشاء المنظر هي:

```
CREATE VIEW <view> ( <column names>) AS
SELECT <column names>
FROM <table>
WHERE <predicate>
```

اسم المنظر يتبع كلمة VIEW ثم يليه اسماء الاعمدة داخل اقواس تفصلها فواصل دون تحديد انواعها، يمكن عدم تحديد اسماء الحقول حيث يتم تسميتها بناء على اسماء الحقول في نتيجة الإستعلام، و بعد كلمة AS تأتي جملة الإستعلام التي يمكن ان تكون اي جملة استفسار بدون أي تقييد لمدى تعقيدها سواء كانت من جدول او اكثر او حتى باستخدام جملة GROUP BY، او باستخدام عمليات حسابية و تعابير رياضية.

من استخدامات المناظر الشائعة بعض المعلومات عن بعض المستخدمين فمثلا اذا اردنا ان نعطي للطالب امكانية ان يرى اسماء المدرسين و بعض المعلومات عنهم و نريد ان نخفي عنه رواتبهم مثلا، يمكن ان نعطي الطالب امكانية الإستعلام من المنظر كما نرى في المثال التالي.

مثال: أنشئ منظر يحتوي على جميع حقول teachers عدا حقل الراتب.

```
CREATE VIEW faculty AS
SELECT teacher_num, teacher_name, phone
FROM teachers
```

في هذا المثال لم نعطي اسماء لحقول المنظر وبالتالي تكون أسماء الحقول في المنظر مطابقة لأسماء الحقول في الجدول الاصيل. يستخدم المنظر كما يستخدم الجدول العادي حيث يمكن استرجاع المعلومات منه بالطريقة التي نريد لكن هنا نضمن ان الراتب لن يظهر في اي استفسار على المنظر.

مثال: أنشئ منظر يحتوي على اسماء و ارقام الطلبة من المستوى الرابع

```
CREATE VIEW grad_students AS
SELECT sname, sid, level
FROM student
WHERE level = 4
```

يمكن انشاء مناظر تحتوي على حقول غير موجودة في اي من الجداول الاساسية لكنها مشتقة من الجداول الاساسية. تسمى الاعمدة المشتقة (derived columns).

مثال: انشاء منظر يحتوي على اسماء المدرسين و ارقامهم وراتبهم السنوي

```
CREATE VIEW year_sal (teacher_name, teacher_num, y_salary) AS
SELECT teacher_name, teacher_num, salary*12
FROM teachers
```

اضافة السجلات الى المنظر ليست بالبساطة التي يمكن ان تنفذ على الجداول الاساسية لان المنظر ليس له وجود مادي اي ان اضافة سطر على المنظر تعني اضافة على الجدول الاساسي. و يمكن ان تكون اضافة الى اكثر من جدول و في احيان اخرى قد يحتوي المنظر على حسابات مما يعقد الامور او قد يجعلها مستحيلة. كقاعدة عامة اذا كان المنظر يستخدم جملة استفسار تحتوي على GROUP BY او دالة تجميع او عمود مشتق يكون للاستفسار فقط. ماذا يحدث اذا اردنا ان نضيف سجل لا يحقق شرط المنظر كما في المثال التالي:

```
CREATE VIEW grad_students AS
SELECT sname, sid, level
FROM student
WHERE level = 4
```

اذا اردنا ان نضيف السجل

```
INSERT INTO grad_student VALUES ('Jamal', 100, 3)
```

هذا السطر لا يسبب مشكلة اذا اضيف الى الجدول الاساسي لكن لا يمكن ان يظهر في المنظر لانه لا يحقق الشرط. هنا يمكن اضافة السطر. و يمكننا ان نمنع اضافة السطر اذا اضعنا عبارة

WITH CHECK OPTION كما يلي:

```
CREATE VIEW grad_students AS
SELECT sname, sid, level
FROM student
```

```
WHERE level = 4
```

```
WITH CHECK OPTION CONSTRAINT grad_constraint
```

وجود هذه العبارة يمنع اضافة اي سجل لا يحقق الشرط المحدد في جملة الإستعلام المستخدمة

لإنشاء المنظر .

إذا لم نعد بحاجة الى المنظر يمكننا ان نحذفه دون التأثير على وجود الجداول الاساسية كما يلي:

```
DROP VIEW grad_student
```


الفصل السابع عشر: ربط لغة الاستعلام الهيكلية مع

لغات البرمجة

يمكن ربط لغة SQL مع لغات البرمجة الأخرى و التي توفر امكانات عامة لا توفرها لغة الإستعلام الهيكلية و هي اللغة المخصصة للاستفسار فقط بعدة طرق منها (Open ODBC (DataBase Connectivity). توفر ODBC للمبرمج واجهة الاتصال بين التطبيقات وقاعدة البيانات عن طريق مجموعة من الاوامر بطريقة معيارية و هو ما يسمى API (application programming interface). طريقة الاتصال هذه تعطي للمبرمج امكانية الاتصال مع عدة قواعد بيانات مهما اختلف الصانع وفي نفس الوقت.

يوفر ODBC امانية تنقل البرنامج من قاعدة بيانات الى اخرى بسهولة عن طريق وضع وسيط بين قاعدة البيانات و البرنامج. كل الاتصالات التي تتم بين البرنامج وقاعدة البيانات تتم عبر المشغل driver. يوفر المشغل امكانية الاتصال مع اي قاعدة بيانات حتى لو كانت لا تدعم لغة SQL عن طريق تحويل الاوامر الى اللغة التي تفهمها قاعدة البيانات كل ذلك دون ان يشعر المبرمج.

يبقى ان ننوه ان بإمكان المبرمج الاتصال مع قاعدة بيانات موجودة على جهاز غير الجهاز الذي يقوم بتنفيذ البرنامج و ذلك من خلال الاتصال عبر شبكة الكترونية.

التطبيق الذي يتفاعل مع قاعدة البيانات عن طريق ODBC يقوم باجراء الاتي: يبدأ باختيار مصدر البيانات، يتم تحديد و تشغيل المحرك المناسب و يتم فتح اتصال مع مصدر البيانات. يمكن فتح اي عدد من الاتصالات مع قواعد البيانات في نفس الوقت ومن نفس التطبيق. بينما

يكون الاتصال قائم يمكن ان نقوم بارسال البيانات و طلب الإستعلامات و تنفيذ اوامر SQL على قاعدة البيانات. عند انهاء العمليات المطلوبة يتم اغلاق الاتصال.

ربط ODBC من خلال PHP

لفتح الاتصال مع قاعدة البيانات نستخدم الدالة `odbc_connect()` و التي تاخذ اربع بارامترات :
مصدر البيانات، اسم المستخدم، كلمة السر، و نوع المؤشرة `cursor`.
`odbc_exec()` تستخدم لتنفيذ اوامر SQL. نتيجة الإستعلام توضع في الذاكرة داخل تركيبة بيانات تسمى المؤشرة `cursor`.

مثال: يتصل مع مصدر بيانات مسمى `university` بدون كلمة اسم مستخدم او كلمة سر ثم يقوم بتنفيذ امر الإستعلام.

```
$conn=odbc_connect('university',"");
$sql="SELECT * FROM student";
$rs=odbc_exec($conn,$sql);
```

يتم استرجاع السجلات من نتيجة تنفيذ امر SQL (من المؤشرة) عن طريق `odbc_fetch_row()` تاخذ الدالة على الاقل كمدخل اسم المتغير الذي يحتوي على نتيجة الإستعلام كما يلي

```
odbc_fetch_row($rs)
```

تعطي `true` اذا كان هناك اسطر و `false` اذا خلت النتيجة من الاسطر. و تقوم بتحضير سطر. كلما استدعينا الدالة تقوم بتحضير السطر التالي حتى تنهي جميع ما تم استرجاعه من قاعدة البيانات.

لاسترجاع الحقول من الاسطر نستخدم odbc_result() حيث نزود الدالة باسم المتغير الذي يحتوي على النتيجة واسم او رقم الحقل المراد استرجاعه.

مثال:

```
$comname=odbc_result($rs,1);
```

حيث يتم استرجاع الحقل الاول او

```
$comname=odbc_result($rs,"sname");
```

حيث يتم استرجاع الحقل المسمى sname

لاغلاق الاتصال نستخدم الدالة odbc_close() كما يلي

```
odbc_close($conn);
```

مثال متكامل

```
<html>
<body>
<?php
$conn=odbc_connect('northwind',"");
if (!$conn)
    {exit("Connection Failed: " . $conn);}
$sql="SELECT * FROM customers";
$rs=odbc_exec($conn,$sql);
if (!$rs)
    {exit("Error in SQL");}
echo "<table><tr>";
echo "<th>Companyname</th>";
```

```
echo "<th>Contactname</th></tr>";
while (odbc_fetch_row($rs))
{
    $comname=odbc_result($rs,"CompanyName");
    $conname=odbc_result($rs,"ContactName");
    echo "<tr><td>$comname</td>";
    echo "<td>$conname</td></tr>";
}
odbc_close($conn);
echo "</table>";
?>
</body>
</html>
```

الجزء الرابع: ملحق رقم (1)

SQL

لغة الإستعلام الهيكلية

الشيء الذي لن تستطيع تجاهله عند تعلم قواعد البيانات هو لغة الإستعلام و التي يعبر عنها بـ SQL

فما هي الـ SQL ؟

أولا : هي اختصار لكلمة Structured Query Language

ثانيا : هي لغة غير إجرائية 00 أي لا يوجد بها If , Select case , Loop , for Next

ثالثا : SQL لغة قياسية ANSI

ماذا يعني أن لغة SQL هي لغة قياسية ANSI؟

ANSI هي اختصار لـ (American National Standards Institute) ، اعتمد هذا المعهد لغة الـ SQL

لجعلها قياسية في التعامل مع جميع قواعد البيانات.

رابعا : نقوم عن طريق هذه اللغة بتحديد العمليات التي نريد أن ننفذها علي قواعد البيانات و تتولي DBMS تنفيذ هذه العمليات.

DBMS هي اختصار لـ (Database Management System) (نظم إدارة قواعد البيانات) و يقصد بها

البرامج التي تتعامل مع قواعد البيانات مثل MS Access

أين نطبق SQL ؟

تعمل مع جميع برامج قواعد البيانات مثل :

DB2, Informix, Oracle, Sybase, MySQL, PostgreSQL , MS Access, MS SQL Server

ما الذي سوف استفيده من تعلم SQL ؟

إدارة قواعد بياناتك بصورة أفضل , أقوى 00 و بشكل احترافي

قواعد اللغة : -

SQL لا تفرق بين الحروف الكبيرة والصغيرة

SQL لا تهتم بالمسافات البيضاء

تنتهي جميع جُمَل اللغة بالفاصلة المنقوطة(;) :

• و إن كان أطلق علي هذه التقنية (لغة) باستخدام اللفظ المفرد إلا إنها تتضمن داخلها لغتان لكل منها وظائف محددة تقوم بها تختلف عن الأخرى تماما.

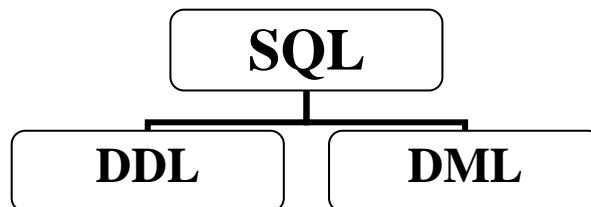
• إذا ذكرت قواعد البيانات 00 انصرف الذهن مباشرة إلي الجداول 00 إذ أن الجداول هي الكائن الأول و الرئيسي في قواعد البيانات 00 ثم تليه باقي الكائنات المكونة لهيكل قواعد البيانات.

• إذا نظرنا إلي قواعد البيانات 00 نجد إنها تتكون من جزئين : رئيسيين -

(1) هيكل قاعدة البيانات Structure : و هي مجموعة الكائنات المكون للقاعدة مثل الجداول و الإستعلامات

(2) البيانات Data التي يتعامل معها المستخدم بالاضافة و الحذف و التعديل

بهذه العجلة السابقة السريعة يمكن معرفة جزئي لغة الإستعلام و هما : -



لغة معالجة البيانات (DML) Data Manipulation Language

لغة تعريف البيانات (DDL) Data Definition Language

لغة توصيف البيانات

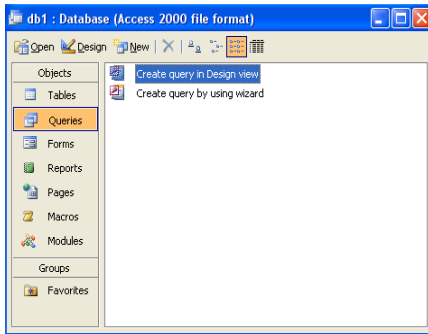
Data Definition Language (DDL)

يُقصد بـ " توصيف " البيانات تعريف هيكل قاعدة البيانات Structure التي سوف يتم تصميمها 00 أي أن هذه اللغة مسنولة عن إنشاء و حذف و تعديل قواعد البيانات و كائناتها مثل الجداول و الإستعلامات و العلاقات و الفهارس داخل الجدول و تحديد الصلاحيات لمستخدمي قواعد البيانات يستخدم لذلك مجموعة أوامر منها (CREATE , DROP , ALTER , GRANT)
و يمكن النظر إلي الجدول التالي لتوضيح بعض وظائف هذه اللغة :
أي تقوم بالوظائف الآتية :

إنشاء CREATE	تعديل ALTER	حذف DROP
Database	Database	Database
Table	Table	Table
Index	Index	Index

سؤال: قبل ذكر أوامر هذه اللغة و كيفية تعلمها 00 أين نكتبها ؟
جواب: سوف نستخدم برنامج Microsoft Office Access 2003 في تعلم أوامر هذه اللغة.

Database Window



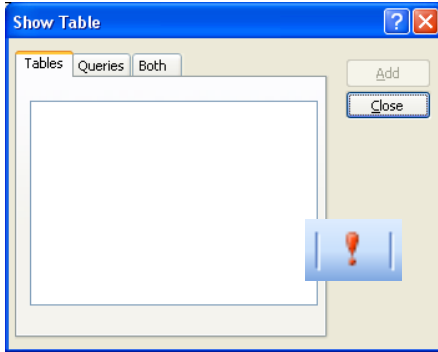
← طريقة كتابة أوامر SQL في Access

- (1) شغل برنامج Microsoft Office Access 2003
- (2) إنشئ قاعدة بيانات فارغة
- (3) في نافذة قاعدة البيانات Database Window من شريط الكائنات Objects حدد الكائن استعلامات Queries
- (4) اضغط مرتين علي الخيار " Create Query in Design View "

كما في الشكل المقابل



- (5) إغلق مربع حوار Show Table (لا يظهر به أي جداول)



- (6) في الركن الأيسر العلوي اضغط الزر



- (7) تظهر النافذة التالية " Query1 "
- في هذه النافذة يمكنك كتابة جُمَل SQL



- بهذا الزر من شريط أدوات Query Design يمكنك تشغيل Run جملة SQL التي أنشيتها

الإنتشاء CREATION

أولا إنشاء قاعده بيانات Create Database
Syntax الشيفرة:

CREATE DATABASE database_name

ملحوظة : لا يمكنك إنشاء قاعدة بيانات بهذا الأمر في MS Access لأنك أنشئت من قبل قاعدة بيانات فارغة

ثانياً: إنشاء جدول داخل قاعده البيانات Create Table
Syntax الشيفرة:

```
CREATE TABLE table_name
(
Field_1 data_type null or not null,
Field_2 data_type null or not null,
Field_3 data_type null or not null,
Primary key ( Field_1 )
);
```

شرح :

Field_1: اسم الحقل

data_type: نوع بيانات الحقل.

Null أو Not null: وهي تعني هل إدخال البيانات اجباري أم اختياري؟

مثلاً هل تريد أن تجبر المستخدم علي ادخل بيانات في الحقل لذلك تكون قيمة خاصية الحقل not null أم يترك الحقل فارغ لذلك تكون قيمة خاصية الحقل Null.

تلاحظ في الشيفرة Syntax السابقة أننا :

(1) نستخدم الأقواس لتعريف حقول الجدول بداخلها.

(2) نستخدم الفاصلة (,) للفصل بين الحقول.

(3) ونستخدم الفاصلة المنقوطة (;) بعد إغلاق أقواس تعريف الحقول 00 أي بعد الانتهاء من الأمر تماماً

(4) بعد الفاصلة المنقوطة نستطيع الشروع في إنشاء جدول جديد . وهكذا..

أنواع البيانات Data Type

(1) جدول الأرقام الصحيحة

النوع	نوعية البيانات التي يمكن تخزينها
INT()	أرقام صحيحة اعتيادية
INTEGER()	مرادف لـ INT()
BIGINT()	أرقام صحيحة كبيرة جداً
MEDIUMINT()	أرقام صحيحة متوسطة الحجم
SMALLINT()	أرقام صحيحة صغيرة
TINYINT()	أرقام صحيحة صغيرة جداً
DECIMAL(size,d)	لتعريف الأعداد ذات الكسور العشرية ، داخل القوسين نحدد في المتغير الأول أكبر عدد لخانات الرقم الصحيح ، وفي المتغير الثاني نحدد عدد الخانات العشرية بعد الفاصلة
NUMERIC(size,d)	

(2) جدول أنواع بيانات التاريخ و الوقت

النوع	نوع البيانات التي يمكن تخزينها
DATE	تاريخ 000 سيظهر YYYY-MM-DD
TIME	وقت 000 سيظهر HH:MM:SS
DATETIME	تاريخ ووقت 000 سيظهر YYYY-MM-DDHH:MM:SS
TIMESTAMP()	يحتوي على التاريخ والتوقيت ، نطاقه يبدأ من "00:00:00-01/01/1970" وهي تقابل (January 1,1970 . 00:00:00) وينتهي النطاق بـ "19:14:07-18/01/2038"
YEAR()	سنة

(3) أنواع TEXT

النوع	الوصف
TEXT	حقل TEXT حجم عادي 00 لتعريف حقل نص يستقبل 65535 حرف كحد أقصى
TINYTEXT	حقل TEXT صغير جداً
MEDIUMTEXT	حقل TEXT متوسط الحجم 00 لتعريف حقل نص يستقبل 16777215 حرف
LONGTEXT	حقل TEXT طويل

(4) أنواع BLOB

حقل BLOB حجم عادي	BLOB
حقل كائن BLOB صغير جدا	TINYBLOB
حقل متوسط الحجم BLOB	MEDIUMBLOB
حقل BLOB طويل	LOB

(5) أنواع السلاسل الاعتيادية

النوع	الوصف
CHAR(Size)	حقل يقوم بتخزين 1 إلى 255 حرف 00 نحدد الحجم داخل القوسين
VARCHAR(Size)	تخزين سلسلة حرفية متغيرة الحجم يتم تحديد أقصى قيمة لها بين قوسين

مثال على إنشاء جدول: إنشاء جدول خاص بالموظفين **Employees**
 جدول الموظفين : رقم الموظف - رقم القسم - اسم الموظف - اسم الأب - التليفون

```
CREATE TABLE employees (
  id INT NOT NULL,
  Dprt_Nmbr INT NOT NULL,
  Fname varchar(200) NOT NULL ,
  Lname char(90) NOT NULL ,
  phone varchar(150) NOT NULL ,
  Primary key (id)
);
```

بالنظر في الكود السابق يمكنك معرفة التالي :

اسم الجدول : employees
 حقول الجدول: (phone , Lname , Fname , id)

الجملة : id INT NOT NULL,

الكلمة الاولى : اسم الحقل
 الكلمة الثالثة : قد تكون Null أو not null و تم شرحها من قبل

الجملة [Primary Key (id)] : ضبط المفتاح الأساسي للجدول علي الحقل (id)

Required	Yes
Indexed	Yes (No Duplicates)

عند ضبط المفتاح الأساسي يقوم Access بضبط
 الخاصية (مطلوب = نعم) (Required = Yes)
 و الخاصية (مفهرس = نعم دون تكرار)
 (Indexed = Yes(No Duplicates)) كما يظهر بالشكل المقابل.

```
CREATE TABLE Person (
  LastName text(30),
  FirstName text(30),
  Address text(150),
  Age Number
);
```

ثالثا : إنشاء الفهارس **CREATE INDEX** :

(أ) الهدف منه تسريع عملية البحث

(ب) يوجد نوعان من الفهارس

1- فريد (لا يتكرر) Unique Index

الشيفرة للفهرس من النوع الفريد (لا يتكرر) Unique Index

```
CREATE UNIQUE INDEX index_name
ON table_name (column_name)
```

مثال :

```
CREATE UNIQUE INDEX index_aa
ON Person (Age)
```

2- بسيط (يتكرر) Simple Index

البناء للفهرس من النوع البسيط Simple Index

CREATE INDEX index_name
ON table_name (column_name [DESC])

➤ إنشاء فهرس بترتيب عكسي :

CREATE INDEX PersonIndex
ON Person (Age DESC)

➤ إنشاء فهرس لحقلين في نفس الجدول

CREATE INDEX PersonIndex
ON table_name (FieldName_1, FieldName_2)

الحذف DROPPING

أولا حذف قاعد بيانات :

Syntax الشيفرة:

DROP DATABASE database_name

ثانيا حذف جدول من قاعده البيانات

Syntax الشيفرة:

DROP TABLE table_name

وإذا كان هذا الجدول مرتبط بعلاقات مع جداول أخرى في قاعدة البيانات ، فلا بد من تحديث بيانات الجداول المتبقية ويكون ذلك بإضافة كلمة **CASCADE** بعد أمر الحذف مباشرة وقبل الفاصلة المنقوطة .

مثال:

حذف جدول الموظفين والذي يرتبط بعلاقة مع جدول آخر في نفس قاعدة البيانات بالأمر التالي:

DROP TABLE employees **CASCADE**;

ثالثا حذف فهرس :

Syntax الشيفرة:

DROP INDEX index_name ON table_name

ملحوظة : تختلف طريق حذف الفهرس حسب نوع قاعدة البيانات 00 الجملة السابقة تتعامل مع قواعد البيانات MS Access

التعديل ALTERING

الأمر **ALTER TABLE** يستخدم في التعديل على الجدول من إضافه وحذف حقول

إضافه حقل :

Syntax الشيفرة:

ALTER TABLE table_name **ADD** column_name **datatype**;

حذف حقل:

Syntax الشيفرة:

ALTER TABLE table_name **DROP COLUMN** column_name;

أو

ALTER TABLE table_name **DROP** column_name **CASCADE**;

مثال : على الجدول التالي

Person:

LastName	FirstName	Address
عبد الرحمن	محمد	شارع النصر

إضافه حقل جديد إلى جدول: إضافة حقل جديد باسم City في جدول Person نوع البيانات Text حجمه (30)

ALTER TABLE Person **ADD** City **varchar**(30)

LastName	FirstName	Address	City
عبد الرحمن	محمد	شارع النصر	

حذف حقل من جدول : حذف حقل Address من جدول Person

ALTER TABLE Person **DROP COLUMN** Address

جملة SELECT INTO

تستخدم عادة في حفظ نسخة احتياطية من الجدول
Syntax الشيفرة:

```
SELECT column_name(s) INTO newtable [IN externaldatabase]
FROM source
```

لعمل نسخة احتياطية من الجدول في نفس القاعده:

```
SELECT * INTO Persons_backup
FROM Persons
IN نستخدم : أخرى بيانات لقاعده الجدول لنسخ
SELECT Persons.* INTO Persons IN 'Backup.mdb'
FROM Persons
```

يمكن أيضا نسخ حقول محددة فقط:

```
SELECT LastName,FirstName INTO Persons_backup
FROM Persons
Where شرط إدخال مع ولكن السابق المثال نفس
SELECT LastName,Firstname INTO Persons_backup
FROM Persons
WHERE City='Aswan'
```

إذا أردنا نسخ الجدول لكن في وجود علاقة مع جدول آخر:

```
SELECT Employees.Name,Orders.Product
INTO Empl_Ord_backup
FROM Employees
INNER JOIN Orders
ON Employees.Employee_ID=Orders.Employee_ID
```

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

تنقسم هذه اللغة إلى نوعين من الإستعلام و هما :

1- استعلامات التحديد Selection Query

2- الإستعلامات الإجرائية Action Query

Selection Query : خاص باستخراج البيانات من الجدول حسب الشرط أو الشروط المطلوبة

Action Query : خاص بعمليات (إضافة – حذف – تعديل) البيانات بالجدول أو الجداول.

Selection Query: Select

Action Query: Insert – Update – Delete

Action Query

INSERT INTO Statement

جملة الإضافة

INSERT INTO

Syntax:

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

(ل) إضافة البيانات إلى الجدول :

بعد إنشائك للجدول تريد بالفعل أن تضيف إليه البيانات 00 إليك الشيفرة الخاصة بذلك

```
INSERT INTO table_Name VALUES ('value_1',' value_2', .....);
```

لاحظ :

- 1- يجب أن تكون عدد القيم مساوية لعدد الحقول بالجدول
- 2- كتابة القيم بالترتيب حسب ترتيب الحقول بالجدول
- 3- البيانات يتم إضافتها سجل سجل.

مثال (1) : إضافة سجل موظف إلى جدول employees الذي أثنائه 00 قيمة كتالي :

```
1 = id          Fname = 'عبدالرحمن'      Lname = 'محمد'      phone = '3120130'
```

```
INSERT INTO employees
VALUES ('1', '3120130', 'محمد', 'عبدالرحمن');
```

① قد ترغب في عدم إضافة بعض البيانات إلى أحد السجلات 00 في هذه الحالة يجب كتابة أسماء الحقول المراد ادخال قيم لها.

مثال (2) : عدم كتابة رقم تليفون الموظف (لاحظ : أن خاصية الحقل يجب أن تكون Null)

```
INSERT INTO employees ( id, Fname, Lname )
VALUES ('2', 'سيد', 'علي');
```

Update Statement**جملة التحديث****UPDATE****Syntax:**

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

(ل) تحديث البيانات

🔗 يقصد بتحديث البيانات تغيير قيمها داخل الجدول 00 نحتاج في هذا الأمر إلى **اسم الجدول** و**اسم الحقل** الذي نريد تعديله ، و**شرط** يحدد لنا السجل الذي نريد تحديثه.
في المثال التالي سنقوم بتغيير الاسم الأول عبد الرحمن إلي Abduo و ذلك باستخدام كلمة الشرط Where حتى يعرف DBMS أي الحقول سيتم تحديثها.

DBMS هي اختصار لـ (Database Management System) (نظم إدارة قواعد البيانات) و يقصد بها البرامج التي تتعامل مع قواعد البيانات مثل MS Access

مثال :

```
UPDATE employees SET Fname = 'Abduo'
WHERE id=1;
```

🔗 إذا أردنا تحديث أكثر من حقل في نفس السجل فإننا نكتب أسماء الحقول وقيمها الجديدة ونفصل بينها بالفاصلة (,).

```
UPDATE table_name
SET column1_name = new_value , column2_name = new_value
WHERE column_name = value ;
```

🔗 الحقل المستخدم في الشرط من الممكن أن يكون نفس الحقل الذي نريد تحديثه ومن الممكن أن يكون حقلاً آخر و المثالي التالي يوضح لنا ذلك :

```
UPDATE employees
SET Fname = "Omar" , Phone = 7120130
WHERE Phone =7130120;
```

OR

UPDATE employees**SET** Fname = "Omar" , Phone = 7120130**WHERE** id = 3;**Delete Statement****جملة الحذف****DELETE**

Syntax

DELETE FROM table_name;**TRUNCATE TABLE** table_name;**(ل) حذف البيانات**

حذف البيانات من دخل الجدول دون حذف الجدول 00 أي تفريغ الجدول من محتوياته دون المساس بهيكله
و يتم ذلك بطريقتين أو أمرين إن صح التعبير

DELETE FROM table_name;**TRUNCATE TABLE** table_name;

الأول :

الثاني :

Where هل يمكن حذف حقل واحد فقط من الجدول ؟ طبعاً 00 ذلك أمر سهل باستخدام كلمة **Where****DELETE FROM** table_name**WHERE** column_name = value ' لحذف حقل معين '**Selection Query****Select Statement****جملة الإصطفاء****الغاية منها اصطفاء (استخراج) المعلومات من الجدول:**

إذ أن استخراج المعلومات من الجداول بعد إدخال البيانات لهو الغاية المرادة من قواعد البيانات
و تسمى المعلومات التي نحصل عليها من الجدول أو مجموعة جداول **استعلام Query**
تكونية جملة الاصطفاء للحصول علي استعلام 00 وأبسط جمل الاستخراج أو الإستهلام

Syntax:

SELECT field(s)_name**FROM** table_name**[1] استخراج جميع المعلومات من جدول :**

مثال (1) : استخراج جميع المعلومات من جدول الموظفين employees

① نستخدم رمز النجمة * ليعبر عن أسماء جميع الحقول.

SELECT * FROM employees**[2] استخراج المعلومات من حقول معينة في الجدول :**

① تفصل بين الحقول المطلوبة عند استخدام هذا الأمر بالفاصلة(,)

مثال(2): استخراج حقول أسماء الموظفين من جدول الموظفين employees

SELECT Fname , Lname**FROM** employees;هل جملة **SELECT** بذلك تفي بالغاية المرجوة منها 00 بالطبع لا 00 نحن نريد منها الكثير و الكثير لمعالجة البيانات و استخراجالمعلومات 00 لكل هذه الأسباب و الأسباب التي لم تذكر 00 هناك مجموعة من الكلمات التي تستخدم مع جملة **SELECT**

لتعطيها مرونة و قوة علي تنفيذ ما يطلب منها .

ما هي هذه الكلمات ؟ 00 و كيف نستخدمها ؟ انطلق معنا و سوف تعرف.

DISTINCT : منع التكرار :

Syntax :

SELECT DISTINCT field_name**FROM** table_name;

إذا كانت البيانات في الحقل الواحد من الممكن أن تكون متكررة ؛ مثل حقل رقم القسم Dprt_Nmbr في جدول الموظفين employees 000 ما هو الحل ؟ الحل بسيط جدا 00 استخدم كلمة *Distinct* مثال(1) : منع التكرار في المعلومات (النتائج المستخرجة)

```
SELECT DISTINCT Dprt_Nmbr
FROM employee;
```

WHERE : شرط

Syntax :

```
SELECT * FROM table_name
WHERE field_name =Value
```

- إذا ذكرت Where لا بد أن تذكر شرط 00 أي استخراج معلومات من جدول أو جداول وفق شرط معين
- تعتبر كلمة Where مشابهة لجمل Do While , Select Case , Loop , If في اللغات الإجرائية
- استخدامها مع العوامل الرياضية **Operator Mathematic**

العملية	وصفها
=	يساوي
<>	لا يساوي
<	أكبر من
>	اصغر من
=<	أكبر من أو يساوي
=>	أصغر من أو يساوي
!=	لا يساوي مع SQL Server

في حالة التساوي "="

1- استخراج البيانات من جدول واحد

```
SELECT * FROM employees WHERE Fname='Abduo';
```

2- استخراج البيانات من جدولين

استخراج أسماء الموظفين ورقم القسم الذي يعملون فيه واسم المشروع الذي يعملون عليه يتم استخراج هذه البيانات من جدولين 00 جدول الموظفين employees و جدول المشاريع projects الحقل الرابط بين الجدولين هو Dprt_Nmbr 00 و هو مفتاح أساسي في جدول employees و أجنبي في جدول projects. ملحوظة : نظرا لأن اسم المفتاح الأساسي و الأجنبي واحد (نفس الاسم) يجب التفريق بينهم باسم الجدول كما في هذا المثال 00 أبصر جملة الشرط Where .

```
SELECT Fname , Dprt_Nmbr, Pname
FROM employees , projects
WHERE employees.Dprt_Nmbr = projects.Dprt_Nmbr;
```

في حالة عدم التساوي

```
SELECT * FROM employees WHERE Fname<>'Abduo';
```

- استخدام العوامل المنطقية **Operator Logic** و هي **Not , Or , And**

```
SELECT * FROM TableName
WHERE field_name =Value
AND field_name =Value
SELECT * FROM TableName
WHERE field_name =Value
Or field_name =Value
```

مثال :

```
SELECT * FROM employees WHERE Fname='Abduo' Or 'سيد';
```

مثال آخر :

```
SELECT * FROM employees WHERE Fname 'سيد' =and id=2;
```

مثال آخر :

```
SELECT * FROM employees WHERE Fname = 'عبد الرحمن' and Age >25;
```

مثال آخر علي تعدد الشروط

```
SELECT * FROM employees WHERE Fname = 'عبد الرحمن' or 'علي' or 'سيد'
```

Operator IN

Syntax:

```
SELECT column_name FROM table_name
WHERE column_name IN (value1,value2,..)
```

أولا : استخدام كلمة in مكان Or عند تعدد الشروط 00 يمكن صياغة التعبير السابق في تعدد الشرط كتالي :

```
SELECT * FROM employees
WHERE Fname IN('عبد الرحمن','علي','سيد')
```

ثانيا : ربط أكثر من استعمال

```
SELECT * FROM table1_name
WHERE FirstName IN (SELECT * FROM table2_name)
```

• استخدام Between And

```
SELECT *
FROM employees
WHERE Age Between 20 and 30;
```

• استخدام Like

مثال : استخراج كل الاسماء التي تبدأ بحرف D
تستخدم * في Access للتعبير عن مجموعة حروف
تستخدم % SQL Server للتعبير عن مجموعة حروف
تستخدم علامة الاستفهام (؟) مع Access للتعبير عن حرف
تستخدم علامة الكشيده (-) مع SQL Server للتعبير عن حرف

أمثلة :

```
SELECT Fname
FROM employees
WHERE Fname Like 'a*';
WHERE Fname Like 'a%';
WHERE Fname Like 'a???'; a الاسم مكون من 4 أحرف أولها a
WHERE Fname Like 'a---'; a الاسم مكون من 4 أحرف أولها a
```

مثال آخر : استخراج كل الأسماء التي تبدأ بحرف D و تنتهي بحرف L

```
SELECT *
FROM employees
WHERE Fname Like 'D*L ';
```

• استخراج البيانات مع ترتيبها تصاعديا أو تنازليا :

- الكلمة الحاكمة للترتيب هي Order By
- Asc ترتيب تصاعدي Ascending
- Desc ترتيب تنازلي Descending

مثال :

```
SELECT * FROM employees
ORDER BY id DESC ;
```

• استخدام Alias (الاسم المستعار)

```
SELECT field FROM table AS Table_Alias جدول
```

`SELECT field AS field_alias FROM table` حقل

يستخدم في تغيير اسم الحقل الذي سوف يظهر للمستخدم
مثال : حقل القسم في جدول الموظفين باسم Dprt_Nmpr و عند استخراج المعلومات تريد أن يظهر الحقل للمستخدم باسم
Department Number

`SELECT Dprt_Nmpr AS [Department Number] FROM employees`

ملحوظة : إذا كان اسم الحقل الذي تريد أن يظهر للمستخدم مكون من كلمتين منفصلتين يجب أن تضعهما بين قوس مربع []
أو تربطهم بكشيدة كتالي Department_Number
يمكن استخدام الاسم المستعار في اظهر الحقل للمستخدم باللغة العربية في حين أنه داخل قاعدة البيانات باللغة اللاتينية.
مثال :

`SELECT Dprt_Nmpr AS [رقم القسم] FROM employees`

عمل الربط التسلسلي [+] :

يمكن عن طريق عامل الربط التسلسلي ربط حقلين أو أكثر و أظهارهم للمستخدم كحقل واحد
مثال : اسم الموظف يتكون من حقلين 00 الحقل الأول FName و الحقل الثاني LName 00 نريد ربطهم
و إظهارهم للمستخدم كحقل واحد باسم (اسم الموظف)

`Select FName + ',' + LName AS [اسم الموظف]`

الحقول المحسوبة :

يمكنك اجراء بعض العمليات الحسابية علي الحقول أثناء استخراج المعلومات منها.

مثال : علي قاعدة بيانات Northwind

`Select Orders.OrderID,ProductID, [Order Details].UnitPrice * [Order Details].Quantity (1- * [Order Details].Discount) AS SubTotal`
`From Orders , [Order Details]`
`Where Order.OrderID = [Order Details].OrderID`

عند تنفيذ هذه الجملة ادخل في مربع الحوار الذي سوف يظهر OrderID و ليكن 10248 مثلاً

JOIN الربط

هو الربط بين الجداول :

الغرض من ربط الجداول ببعضها هو :

- 1- استخراج البيانات من أكثر من جدول في قاعدة البيانات 00 بشرط وجود علاقة بين الجداول.
- 2- يستخدم الربط كوسيلة لمنع تكرار القيم في أي حقل من حقول الجداول بصورة مزعجة أحياناً .
فمثلاً حينما يكون للطالب أكثر من مادة وتُدرج جميع المواد مع اسم الطالب في نفس الجدول ، ستكون النتيجة وجود عدد كبير من السجلات لنفس الطالب في نفس الجدول ، كل سجل خاص بمادة !!
وهذا غير عملي أبداً إذ أنه يؤدي إلى كبر الجدول وبالتالي زيادة المساحة التخزينية له !!
لذلك فإننا نقوم بعمل جدولين جدول للطالب بكامل بياناته (الاسم -الرقم - تاريخ الميلاد -التخصص - المعدل ... إلخ) و جدول آخر للمواد يحوي أسم المادة وأرقام الطلاب الذين يدرسونها ، ونربط بين الجدولين برقم الطالب بالتأكد حيث أننا اخترناه مفتاحاً أساسياً لاستحالة تكراره لأكثر من طالب !!

تنكر : الحقل الذي يكون مفتاح أساسى Primary Key لا يمكن تكرار البيانات بداخله **Unique**
لاستخراج بيانات من جدولين ، نستخدم المفتاح الذي يربط بينهما في الشرط ، كالتالي:

`SELECT table_1.any_field, table_2.any_field`

`FROM table_1, table_2`

`WHERE table_1.field_PrimaryKey = table_2.field_ForeignKey`

- في جدول Employees حقل (Employee_ID) هو حقل مفتاح أساسى لجدول الموظفين
- في جدول Orders حقل (Order_ID) هو حقل مفتاح أساسى لجدول الطلبيات
- في جدول Orders حقل (Employee_ID) و يسمى هنا مفتاح أجنبي.

مثال : نريد أن نعرف من طلب منتج وما هو المنتج

`SELECT Employees.Name, Orders.Product`

`FROM Employees, Orders`

`WHERE Employees.Employee_ID=Orders.Employee_ID`

مثال : نريد ان نعرف من طلب المنتج Printer

```
SELECT Employees.Name
FROM Employees, Orders
WHERE Employees.Employee_ID=Orders.Employee_ID
AND Orders.Product='Printer'
```

هذه هي الطريقة الاعتيادية ، بينما يوجد في لغة SQL عدة أشكال لعملية الربط JOIN

الشكل الأول INNER JOIN :

هذا النوع من الربط يستخرج النتائج من الجدولين 00 والتي يتحقق فيها الرابط المذكور:

INNER JOIN ON

Syntax:

```
SELECT field_1, field_2, field_3
FROM first_table INNER JOIN second_table
ON first_table.field_PrimaryKey = second_table.field_Foreignkey ' الرابط
```

مثال:

```
SELECT Employees.Name, Orders.Product
FROM Employees INNER JOIN Orders
ON Employees.Employee_ID=Orders.Employee_ID
```

تقوم الجملة INNER JOIN بعرض جميع البيانات المشتركة بين الجدولين

مثال آخر :

نريد عرض أسماء الموظفين الذن طلبوا المنتج (Printer)

```
SELECT Employees.Name
FROM Employees INNER JOIN Orders
ON Employees.Employee_ID=Orders.Employee_ID
WHERE Orders.Product = 'Printer'
```

الشكل الثاني: LEFT JOIN :

هذا النوع من الربط يعطينا جميع السجلات من الجدول الأول (وهو الجدول الذي يكتب على يسار كلمة LEFT JOIN في الأمر سواء المرتبطة بقيم في الجدول الثاني أو غير المرتبطة 00 أي أن جميع سجلات الجدول الأول ستظهر في النتائج :

LEFT JOIN ON

Syntax:

```
SELECT field_1, field_2, field_3
FROM first_table LEFT JOIN second_table
ON first_table.field_PrimaryKey = second_table.field_Foreignkey
```

مثال

```
SELECT Employees.Name, Orders.Product
FROM Employees LEFT JOIN Orders
ON Employees.Employee_ID=Orders.Employee_ID
```

تقوم الجملة LEFT JOIN بعرض جميع البيانات من الجدول الأول Employees حتى لو لم توجد في الجدول الثاني Orders 00 بمعنى عرض جميع أسماء الموظفين Employees.Name

الشكل الثالث: RIGHT JOIN :

هذا النوع من الربط هو معكوس النوع السابق تماماً ، فهو يعطينا جميع السجلات من الجدول الثاني (وهو الجدول الذي يأتي على يمين كلمة RIGHT JOIN عند كتابة الأمر) سواء كانت مرتبطة بقيم في الجدول الأول أو لا ، جميع سجلات الجدول الثاني ستظهر في النتائج

RIGHT JOIN ON

Syntax:

```
SELECT field_1, field_2, field_3
FROM first_table RIGHT JOIN second_table
ON first_table.field_PrimaryKey = second_table.field_Foreignkey
```

مثال:

SELECT Employees.Name, Orders.Product**FROM** Employees**RIGHT JOIN** Orders**ON** Employees.Employee_ID=Orders.Employee_IDتقوم الجملة **RIGHT JOIN** بعرض جميع البيانات من الجدول الثاني **Orders** حتى لو لم توجد في الجدول الأول

الدمج

UNION and UNION ALL

تستخدم لدمج حقلين من جدولين مختلفين ولكن يجب أن يكون نوع البيانات في الحقلين واحد.

Syntax:

SQL Statement 1

UNION

SQL Statement 2

مثال :

الجدول الأول Student_School_1 و الجدول الثاني Student_School_2

Student_School_1

Student_School_2

Std_No	Std_Name
1	محمد
2	ياسر
3	سيف
4	عبد الرحمن

Std_No	Std_Name
1	جمال
2	جابر
3	جميل
4	جبران

نريد دمج الحقل Std_Name من الجدولين

SELECT Std_Name **FROM** Student_School_1**UNION****SELECT** Std_Name **FROM** Student_School_2

النتيجة:

Std_Name
محمد
ياسر
سيف
عبد الرحمن
جمال
جابر
جميل
جبران

نلاحظ أن النتيجة ظهرت بدون تكرار للبيانات

استخدام **UNION ALL** مثل استخدام **UNION** الفرق بينهم عرض جميع البيانات حتى لو وجد تكرار.**Syntax:**

SQL Statement 1

UNION All

SQL Statement 2

الدوال

SQL Functions

لغة SQL بها الكثير من الدوال العددية والحسابية
البناء الأساسي لأي دالة:

SELECT function(column) FROM table و الدوال

ايجاد الوسط الحسابي للحقل المحدد	AVG(column)
معرفة عدد السجلات في الحقل بدون السجلات الفارغة	COUNT(column)
معرفة عدد الصفوف في الجدول	COUNT(*)
معرفة قيمة أول سجل في الجدول	First(column)
معرفة قيمة آخر سجل في الجدول	last(column)
معرفة أكبر قيمة في الحقل	Max(column)
معرفة أصغر قيمة في الحقل	Min(column)
معرفة أجمالي القيم في الحقل	SUM(column)
عدد السجلات في الحقل بدون تكرار تعمل فقط مع SQL SERVER	COUNT(DISTINCT column)

مجموعه من الأمثله:

SELECT AVG (Column) From Table
SELECT COUNT (column) From Table
SELECT COUNT (*) From Table
SELECT First (column) From Table
SELECT Last (column) From Table
SELECT Max (column) From Table
SELECT Min (column) From Table
SELECT SUM (column) From Table
SELECT COUNT (DISTINCT column) From Table

التجميع و الإحتواء

GROUP BY and HAVING

التجميع و دوال التجميع: Group & Aggregate Function

نقصد بدوال التجميع في SQL أي دالة من الدوال التي تعلمناها سابقاً إذا طبقناها على جدول بعد فرز سجلاته
دوال التجميع في السكيول دائماً تحتاج إلى تثبيت حقل يتم الفرز من خلاله يسمى
"حقل التجميع grouping column يتعين هذا الحقل عن طريق أمر Group by ، صيغته كالتالي:

Syntax:

SELECT column,SUM(column) **FROM** table_name **GROUP BY** grouping_column

مثال للتوضيح : على الجدول التالي

Company	Amount
W3Schools	5500
IBM	4500
W3Schools	7100

اسم الجدول
Sales

تنفذ الجملة التاليه:

SELECT Company, SUM(Amount) **FROM** Sales

النتيجة :

Company	SUM(Amount)
IBM	17100
W3Schools	17100
W3Schools	17100

النتيجة

نلاحظ ان جميع السجلات أخذت المجموع كله ولم نعرف مجموع كل سجل.
الآن نجرب جملة الاستعلام بعد إضافته GROUP BY

```
SELECT Company,SUM(Amount) FROM Sales
GROUP BY Company
```

Company	SUM(Amount)
IBM	4500
W3Schools	12600

النتيجة

نلاحظ انه تم جمع كل سجل وحده وأصبحت النتيجة أوضح

HAVING & Aggregate Function: التجميع والاحتواء

يستخدم أمر الاحتواء مع دوال SQL لأنه يستحيل استخدام أمر الشرط *Where* على الدوال وخصوصاً إذا طبقنا عليها التجميع *Group by* لذلك فأمر الاحتواء *Having* يمكننا بطريقة ما من الشرط على الدوال والتجميع في أمر واحد.

```
SELECT grouping_column,function(column) FROM table_name
GROUP BY grouping_column
```

```
HAVING function(column) condition value;
```

إذن، فأمر *HAVING* ببساطة هو أمر للشرط على الدوال *GROUP BY* يستخدم في الجملة إذا استخدمنا فيها دالة و أمر التجميع *GROUP BY*

الدالة *HAVING* تستخدم لفرز البيانات حسب شرط معين *GROUP BY* تطبيقاً على المثال السابق :

Company	Amount
W3Schools	5500
IBM	4500
W3Schools	7100

اسم الجدول
Sales

نقد جملة الاستعلام الآتية :

```
SELECT Company,SUM(Amount) FROM Sales
GROUP BY Company
HAVING SUM(Amount)>10000
```

Company	SUM(Amount)
W3Schools	12600

النتيجة

مثال آخر :

جدول المشاريع *projects* و جدول العمل *works_on*
المطلوب : معرفة متوسط ساعات العمل التي تقل عن 15 ساعة

```
SELECT Proj_name, AVG(hours) AS Hours_Avg
FROM projects, works_on
WHERE Proj_number=PNO
GROUP BY Proj_name
HAVING AVG(hours) < 15;
```

تمارين

التمرين الاول:

على اعتبار ان لديك الجدولين التاليين و اللذين يمثلان قاعدة بيانات لمتجر حواسيب. اجب عن

الاسئلة التالية

```
CREATE TABLE Manufacturers (
    Code INTEGER PRIMARY KEY NOT NULL,
    Name TEXT NOT NULL
);
```

```
CREATE TABLE Products (
    Code INTEGER PRIMARY KEY NOT NULL,
    Name TEXT NOT NULL ,
    Price REAL NOT NULL ,
    Manufacturer INTEGER NOT NULL
    CONSTRAINT fk_Manufacturers_Code REFERENCES
    MANUFACTURERS(Code)
);
```

- 1- اوجد اسماء جميع المنتجات في المتجر
- 2- اوجد اسماء و اسعار جميع المنتجات في المتجر
- 3- اوجد اسماء و اسعار المنتجات التي يقل سعرها عن 200 \$
- 4- اوجد اسماء و اسعار المنتجات التي تنحصر اسعارها بين 60 \$ و 120 \$
- 5- اوجد اسماء و اسعار المنتجات بالدينار الاردني (سعر الصرف 0.71)

- 6- اوجد معدل اسعار المنتجات جميعها
- 7- اوجد معدل اسعار المنتجات التي ينتجها المنتج صاحب الرمز 2
- 8- اوجد عدد المنتجات التي يساو سعرها او يزيد عن \$180
- 9- اوجد اسماء و اسعار المنتجات التي يزيد سعرها عن \$180 مرتبة اولا تنازليا حسب السعر ثم تصاعديا حسب الاسم.
- 10- اوجد اسماء المنتجات مع اظهار اسم المصنع لكل منتج.
- 11- اوجد معدل اسعار المنتجين لكل مصنع.
- 12- اوجد اسماء المصنعين الذين يصنعون منتجات معدل اسعارها اكبر من \$180
- 13- اوجد اسم وسعر اخص منتج في قاعدة البيانات
- 14- اوجد اسم كل مصنع و اسم اغلى سلعة يصنعها.
- 15- اضع الى قاعدة البيانات المنتج سماعات بسعر \$70 للمصنع رقم 2.
- 16- عدل ايم المنتج رقم 8 الى طباعة ليزر
- 17- عدل الاسعار لكل المنتجات بحيث تطبق خصم 10%
- 18- عدل اسعار المنتجات التي يزيد سعرها عن \$180 بحيث تطبق خصم مقداره 5% عليها.

التمرين الثاني:

على اعتبار ان لديك الجدولين التاليين و اللذين يمثلان قاعدة بيانات لمتجر حواسيب. اجب عن

الاسئلة التالية:

```
CREATE TABLE Movies (
  Code INTEGER PRIMARY KEY NOT NULL,
  Title TEXT NOT NULL,
  Rating TEXT
);
```

```
CREATE TABLE MovieTheaters (
  Code INTEGER PRIMARY KEY NOT NULL,
  Name TEXT NOT NULL,
  Movie INTEGER
  CONSTRAINT fk_Movies_Code REFERENCES Movies(Code)
);
```

- 1- اوجد اسماء جميع الافلام
- 2- اوجد كل التصنيفات Rating في الجدول.
- 3- اوجد اسماء الافلام غير المصنفة.
- 4- اوجد اسماء المسارح التي لا تقوم بعرض افلام حاليا.
- 5- اوجد معلومات جميع المسارح و معلومات الافلام التي تعرض في المسرح ان كان هناك فلم يعرض في المسرح.
- 6- اوجد اسماء الافلام التي لا تعرض في اي من المسارح.
- 7- ضع التصنيف "عام" لجميع الافلام غير المصنفة.

التمرين الثالث: اسئلة قصيرة

- 1 كيف يمكنك التنقل إلى كائنات مختلفة في قاعدة بيانات؟
- 2 كيف يمكنك بدء تشغيل Access؟
- 3 ما هو "مساعد Office"؟
- 4 لماذا تعد الجداول أساسًا لكافة كائنات قواعد البيانات الأخرى؟
- 5 لماذا تعتبر إمكانية ربط الجداول في قاعدة بيانات مرتبطة ميزة؟
- 6 ما الطرق الثلاث التي يمكنك استخدامها للتنقل في جدول في Access؟
- 7 كيف يمكنك عرض قائمة بكل أسماء النماذج الموجودة في قاعدة بيانات Access؟
- 8 ما الوضعان المتاحان لجدول بالطريقة "عرض ورقة البيانات" وما وجه الاختلاف بينهما؟
- 9 كيف يمكنك فتح جدول في "طريقة عرض التصميم"؟
- 10 كيف يمكن تغيير طريقة عرض جدول من "طريقة عرض التصميم" إلى الطريقة "عرض ورقة البيانات"؟

التمرين الرابع: التمارين الشاملة

- التمرين 1:** من الشريط "الكائنات"، انقر فوق "نماذج" وقم بفتح النموذج frmEmployees.
اطرح سؤالاً على "مساعد Office" مفاده ما الجديد في Access 2002؟ ثم أغلق "مساعد Office" ثم النموذج frmEmployees.

التمرين 2: من الشريط "الكائنات"، انقر فوق "جداول" وافتح الجدول tblEmployees. قم بمعاينة الجدول في "طريقة عرض التصميم" ثم في الطريقة "عرض ورقة البيانات". قم بالتنقل إلى الحقل الأخير من السجل الأخير باستخدام مفاتيح الاختصار. انتقل إلى السجل 18 باستخدام "مربع سجل محدد". انتقل إلى الحقل "تعريف_الموظف" في السجل 18 باستخدام مفاتيح الاختصار. باستخدام أزرار التنقل، انتقل إلى السجل الأول ثم أغلق الجدول.

التمرين 3: من الشريط "الكائنات"، انقر فوق "جداول" وافتح tblReservations. قم بمعاينة الجدول في "طريقة عرض التصميم" ثم في الطريقة "عرض ورقة البيانات". بعد فحص الحقول المحددة لهذا الجدول، هل يمكنك التفكير في أي حقول أخرى قد تكون مفيدة في هذا الجدول؟ هل تعتقد أنه يمكن حذف حقول من هذا الجدول دون التأثير على عمل المنتج؟

التمرين 4: من الشريط "الكائنات"، انقر فوق "جداول" وافتح tblReservations. قم بمعاينة الجدول في الطريقة "عرض ورقة البيانات". اضغط المفتاح F11 للوصول إلى الإطار "قاعدة بيانات" وانقر فوق الجدول tblSuites لفتحه. قم بعرض هذا الجدول في "عرض ورقة البيانات". بما أن برنامج Access يعتبر قاعدة بيانات مرتبطة، يمكنك بالتالي ربط أحد الجداول بجدول آخر. ما الحقل (الحقول) الذي يمكنك استخدامه لربط الجدول tblSuites بالجدول tblReservations لإنشاء علاقة بين هذين الجدولين؟

التمرين 5: لقد أوضح لك هذا الدرس كيفية تشغيل أشرطة الأدوات وإيقاف تشغيلها. هل يمكنك التفكير في الأسباب التي قد تدفعك دائماً إلى إيقاف تشغيل أحد أشرطة الأدوات المعروضة أمامك؟

استخدام لوحة المفاتيح للتنقل في الطريقة عرض ورقة البيانات

يمكنك أيضًا استخدام لوحة المفاتيح للتنقل داخل جدول بضغط مفاتيح تسمى مفاتيح الاختصار على لوحة المفاتيح. بعد أن تصبح محترفًا في استخدام مفاتيح الاختصار، قد تصبح هذه الطريقة أسرع في التنقل من استخدام الماوس.

يمكن استخدام مفاتيح الاختصار التالية للتنقل داخل جدول:

اضغط	لانتقال
مفتاح الجدولة (Tab)	إلى الحقل التالي في السجل الحالي.
مفتاح الإدخال (Enter)	إلى الحقل التالي.
مفتاح صفحة للأعلى (Page Up)	شاشة واحدة إلى أعلى مع تحديد سجل بعيد إلى الأعلى في الجدول.
مفتاح صفحة للأسفل (Page Down)	شاشة واحدة إلى أسفل وتحديد سجل بعيد إلى أسفل في الجدول.
مفتاح سهم لليمين	إلى الحقل التالي في السجل الحالي.
مفتاح سهم لليسار	إلى الحقل السابق.
مفتاح سهم للأسفل	إلى السجل التالي.
مفتاح سهم للأعلى	إلى السجل السابق.
مفتاح البداية (Home)	إلى الحقل الأول من السجل الحالي.
مفتاح النهاية (End)	إلى الحقل الأخير من السجل الحالي.
مفتاح التحكم (Ctrl)+مفتاح سهم للأسفل	إلى السجل الأخير في الحقل الحالي.
مفتاح التحكم (Ctrl)+مفتاح سهم للأعلى	إلى السجل الأول في الحقل الحالي.
مفتاح التحكم+مفتاح البداية (Ctrl+Home)	إلى الحقل الأول من السجل الأول.

إلى اليمين شاشة واحدة مع تحديد حقل بعيد إلى اليمين في الجدول.	مفتاح التحكم+مفتاح صفحة للأسفل (Ctrl+Page Down)
إلى اليسار شاشة واحدة مع تحديد سجل بعيد إلى اليسار في الجدول.	مفتاح التحكم+مفتاح صفحة للأعلى (Ctrl+Page Up)
إلى الحقل الأخير من السجل الأخير.	مفتاح التحكم+مفتاح النهاية (Ctrl+End)
إلى الحقل السابق.	مفتاح العالي+مفتاح الجدولة (Shift+Tab)

في هذا التمرين، تستخدم مفاتيح الاختصار للتنقل داخل الجدول tblGuests.

- 1 مع تحديد حقل "تعريف_النزيل" في السجل الأول، اضغط مفتاح الجدولة (Tab).
- 2 اضغط مفتاح الإدخال (Enter). يحدد Access حقل "الاسم_الأول" في السجل الأول.
- 3 اضغط مفتاح صفحة للأسفل (Page Down).
- 4 ينتقل الجدول شاشة واحدة إلى الأسفل، ويجعل Access يقوم بتحديد سجل بعيد إلى الأسفل في الجدول.
- 5 اضغط مفتاح صفحة للأعلى (Page Up).
- 6 ينتقل الجدول شاشة واحدة إلى الأعلى، وبذلك يحدد Access السجل الأول.
- 7 اضغط مفتاح سهم اليمين.
- 8 يحدد Access حقل "العنوان" في السجل الأول.
- 9 اضغط مفتاح سهم اليسار.
- 10 يحدد Access حقل "الاسم_الأخير" في السجل الأول.
- 11 اضغط مفتاح العالي+مفتاح الجدولة (Shift+Tab).
- 12 يحدد Access حقل "الاسم_الأول" في السجل الأول.
- 13 اضغط مفتاح سهم للأسفل.
- 14 يحدد Access حقل "الاسم_الأول" في السجل الثاني.
- 15 اضغط مفتاح سهم للأعلى.
- 16 يحدد Access حقل "الاسم_الأول" في السجل الأول.
- 17 اضغط مفتاح النهاية (End).
- 18 يحدد Access حقل "قائمة_المراسلات" في السجل الأول.
- 19 اضغط مفتاح البداية (Home).
- 20 يحدد Access حقل "تعريف_النزيل" في السجل الأول.
- 21 اضغط مفتاح التحكم+مفتاح النهاية (Ctrl+End).
- 22 يحدد Access حقل "قائمة_المراسلات" في السجل الأخير.
- 23 اضغط مفتاح التحكم+مفتاح البداية (Ctrl+Home).
- 24 يحدد Access حقل "تعريف_النزيل" في السجل الأول.
- 25 يمكنك أيضًا إغلاق جدول بالنقر فوق "إغلاق" من القائمة "ملف".
- 26 انقر فوق الزر "إغلاق" في الزاوية العلوية اليمنى من الجدول tblGuests.
- 27 يتم إغلاق الجدول.

قائمة المصطلحات

انجليزي	عربي
Union	اتحاد (مجموعات)
Query	استفسار
Alias	اسم المستعار
Ambiguity	التباس
order	ترتيب
Alphabetical order	ترتيب الابتثي
Ascending order	ترتيب تصاعدي
Descending	ترتيب تنازلي
Concatenation	تشبيك (سلاسل الاحرف، النصوص)
Expression	تعبير حسابي
Intersection	تقاطع (مجموعات)
Table	جدول
Query statement	جملة استفسار
Field	حقل
Row	سطر (السجل)
String	سلسلة احرف
Selection condition	شرط اختيار

Join condition	شرط ربط
Phrase	عبارة
Column title	عنوان /ترويسة العمود
Difference	فرق (مجموعات)
Index	فهرس
Null	قيمة الخالية
Efficiency	كفاءة التنفيذ
Structured Query Language	لغة الإستعلامات الهيكلية
Data Definition Language	لغة تعريف البيانات
Data Manipulation Language	لغة تغيير البيانات
Set	مجموعة
Range	مدى
Sort key	مفتاح الترتيب
Query results	نتائج الإستعلام
Text	نص
Pattern	نمط