الفريق العربي للبرمجة

www.ArabTeam2000.com
الأحد، 01 نيسان، 2001

بناء تطبيقات قواعد البيانات العملاقة

باستخدام تكنولوجيا مايكروسوفت

Microsoft SQL Server 2000
Microsoft Access XP (2002)
Microsoft SQL Server Analysis Processing
Transact Structured Query Language "TSQL"
Multidimensional Expressions "MDX"
Visual Basic for Applications 6.3
Distributed Component Object Model "DCOM+"

تأليف / خضر يوسف ترزي
خبير معتمد من مايكروسوفت، مهندس نظم معتمد من مايكروسوفت، مختبر نظم لشركة مايكروسوفت

KhaderTarazi@Hotmail.com
http://www.ArabicOnline.net/Tarazi

المرجع العربي المجاني الوحيد عبر انترنت

فهرس المحتويات

قبل أن تبدأ بقراءة هذا الكتاب

- الكتاب لا يزال في قيد الإعداد ويحدث كل أسبوع تقريبا، للحصول على آخر نسخة من هذا الكتاب قم بزيارة موقع الفريق العربي للبرمجة http://www.ArabTeam2000.com.
- نظرا لان الكتاب لا يزال في مرحلة الإنشاء، فالأخطاء الإملائية والنحوية التي فيه كثيرة للغاية، كما إن هناك الكثير من الأجزاء الناقصة والتي اعمل عليها الآن.
- نظرا لان الكتاب لا يزال في مرحلة الإنشاء، قد يكون هناك بعض الأخطاء في المعلومات نفسها، في المصطلحات أو في الشرح، أنا بالطبع اعمل جاهدا لعدم حدوث ذلك.
- يمكنك طباعة هذا الكتاب للاستخدام الشخصي فقط، و يمنع استغلاله في أية أمور تجارية بدون الإذن الخطي من المؤلف.
- في حالة بان وجدت جزء من هذا الكتاب غير واضحة أو بحاجة إلى شرح مفصل أكثر فبرجاء إخباري بذلك فورا على بريدي الإلكتروني، حتى أتمكن من إضافة الأجزاء الناقصة أو توضيح الأجزاء الغير واضحة فورا.
- مواضيع الكتاب كثيرة للغاية، وأنا ابحث عن مساعدة في تكملته بأسرع ما يمكن، فان وجدت في نفسك القدرة على كتابة بعض المواضيع، فلا تبخل بمساعدتي بها، وانضم إلى فريق المؤلفين لهذا الكتاب، يمكن مساعدتي في أية جزء من الأجزاء التي يجب إضافتها إلى الكتاب.

ما رأيك في هذا الكتاب

آرائك بهذا الكتاب تهمنا للغاية، فهي السبيل الوحيد لتطويره و جعله أسهل للفهم والقراءة، فلو وجدت مثلا بان جزء منه غير واضح، أو لاحظت بان بعض المواضيع بحاجة إلى تفصيل أكثر، أو وجدت نفسك قادر على الكتابة في موضوع من مواضيع الكتاب، أو إن رغبت بإضافة مواضيع جديدة، أو أية شيء آخر يخطر على بالك لحظة قرائه هذا الكتاب

فبرجاء إخباري عن ذلك على بريدي الإلكتروني
KhaderTarazi@Hotmail.com

فريق المؤلفين

خضر يوسف ترزي

خبير معتمد من مايكروسوفت

مهندس نظم معتمد من مايكروسوفت

حاصل على شهادة البكالوريوس في علوم الكومبيوتر - نظم المعلومات

مدير نظم المعلومات في فرع من فروع شركة Vivendi Universal، بيت لحم، http://www.vivendi.com

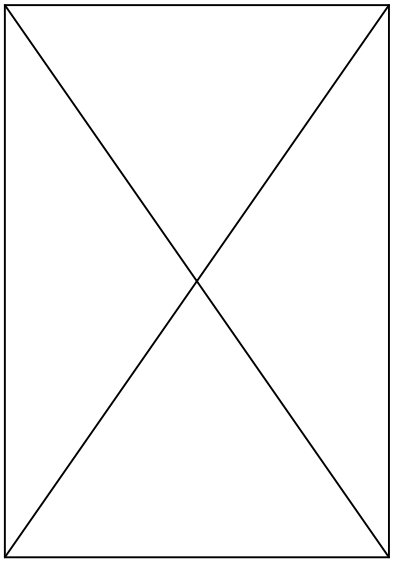
مستشار و مختبر نظم مايكروسوفت

- Microsoft Internet Explorer 5.0, 5.01, 5.5
- Microsoft Office XP
- Microsoft Visual Studio .NET

يعمل في مجال برمجة الكومبيوتر وقواعد البيانات منذ عام 1995 وحتى اليوم، واشترك في بناء العديد من البرامج، والأبحاث.

KhaderTarazi@Hotmail.com
http://www.ArabicOnline.net/Tarazi

انضم إلى فريق المؤلفين



تفضل بالانضمام إلى فريق المؤلفين، كل ما عليك هو اختيار موضوع من المواضيع الغير مكتملة في هذا الكتاب و الكتابة فيه.

المواضيع غير المكتملة موجودة في ملحق أسفل هذا الكتاب.

راسلني على بريدي الالكتروني

كلمة المؤلف

كثيرا ما تمنيت من أن اشغل جهاز الكومبيوتر الخاص بي في صباح يوماً باكر، افتح لغة البرمجة المثالية وابني بها برامجي في شتى المجالات، حيث تعمل تلك البرامج بالضبط كما أريد وبدون مشاكل و أتمكن توزيعها بحرية أينما شئت.

أحلام كبيرة أليس كذلك ؟ ولكنها بعيدة عن الواقع كل البعد، فعالم الكومبيوتر اليوم مليء بالتكنولوجيا المتنوعة في شتى المجالات، تكنولوجيا تتصارع ضد بعضها البعض محاولة لفت انتباهك إليها، لتجعلك تستخدمها لبناء برامجك.

كل عام تقريبا أقوم بتثبيت جميع لغات البرمجة، ونظم قواعد البيانات، على جهاز وابدأ بعمليات اختبار مكثفة عليها، و تهدف اختباراتي هذه إلى محاولة إيجاد أفضل طريقة لبناء نظم قواعد البيانات من حيث الوقت و السهولة والتكلفة ولإنجاز البرامج من جهة ، وجودة البرامج المنتجة من جهة أخرى.

وكل عام تنتج قائمة طويلة عريضة بأسماء النظم التي يجب أن استخدمها هذا العام لبناء البرامج العادية، قواعد البيانات الصغيرة، المتوسطة، الكبيرة، و شتى مجالات البرمجة الأخرى، ولذلك لن أتمكن حتى اليوم من تحقيق حلمي وهو لغة برمجة قادرة على القيام بكل شيء بأفضل الطرق.

لذلك سأناقش في كتابي هذا الكثير من التكنولوجيا معاً، وكيف نستخدمها جنباً إلى جنب لبناء برامج قواعد البيانات العملاقة.

فأنا بطبيعتي لا أحب الحلول الوسط، فأما البرنامج يعمل، أو إن البرنامج لا يعمل، حيث لا أتستطيع تقبل فكرة يعمل تقريبا ولكن هناك مشكلة في كذا و كذا، فهذا بالنسبة لي يعني انه لا يعمل، وهذا الرأي دائما يسبب لي الكثير من المشاكل، فأنا انتقد كل شيء تقريباً، حتى برامجي لا تسلم من ذلك.

كما إنني لا أحب أن اشرح شيء بخطوات فقط، فلا تتوقع أن تجد العبارة التالية لوحدها فقط "قم بالخطوات التالية لتحصل على النتيجة"، بالطبع لا يخلوا الكتاب من خطوات، ولكنها مرفقة مع الشرح أساسا، فأنا أحب أن أوضح لماذا قمنا بتلك الخطوات، ما هي الخطوات البديلة و الاحتمالات الممكن استخدامها، ومن ثم إن توفر لدي الوقت قد أضع بعض الخطوات لتنفيذ العملية.

لذلك ستجد الكثير من الشرح النظري في الكتاب، ولكن صدقني فكل هذا الشرح النظري سيساعدك في مرحلة بناء برامجك المستقبلية بصورة صحيحة وسليمة.

توفر شركة مايكروسوفت مجموعة متكاملة من الأدوات لبناء تطبيقات قواعد البيانات العملاقة، فبناء نظم عملاقة لا يقتصر فقط على لغة برمجة واحدة، بل إن عملية بنائها تتطلب مزيج من الأدوات المختلفة لتصميم تلك البرامج، والبرامج الناتجة في معظم الأحيان ليست قطعة واحدة، بل هي مكونة من أجزاء مترابطة مع بعضها البعض.

أجزاء من تلك البرامج يجب أن تستخدم من داخل الشركة، و أجزاء أخرى من خلال إنترنت، أجزاء من البرامج يجب أن تخصص للمستخدمين المختصين باستخدامها، و أجزاء أخرى يجب أن تخصص للإدارة العليا والتي لا ترغب بالتفاصيل بل هي بحاجة إلى النتائج فقط.

منذ فترة طويلة وأنا مرتاح مع تكنولوجيا مايكروسوفت، وذلك في بناء تطبيقات قواعد البيانات العملاقة باللغة العربية والإنجليزية، أنا اعلم بأنها ليست التكنولوجيا المثلى، فمثلها مثل النظم الأخرى فهي ملية بالنواقص، محكومة من مجموعة من القوانين التي لا تروق لي في بعض الأحيان، بها بعض المشاكل عندما نتحدث عن اللغة العربية، ولكنها في نهاية المطاف أفضل شركة توفر أدوات البرمجة اللازمة لبناء التطبيقات السابق ذكرها، من وجهة نظري طبعاً.

لذلك منذ فترة طويلة وأنا ابني برامجي بواسطتها، لقد تعلمت تخطي مشاكلها والالتفاف حولهم، وبناء برامج مستقرة باللغة العربية.

كثيرا ما سالت فريق تطوير مايكروسوفت Access و Office السؤال التالي "هل تعتقدون إن برامجكم حقاً تعمل بالعربية ؟" ، وفي الكثير من الأحيان كان ذلك يؤدي إلى مناوشات بيني وبينهم ، ولكن في نهاية المطاف فدعم اللغة العربية لا يأتي بليلة وضحاها ، و هو مرتبط باعتبارات كثيرة ومنها حجم السوق وأمور أخرى ، تخرج عن مجال كتابنا.

أنا استخدم تكنولوجيا قواعد بيانات Access منذ أول إصدار لها تحت بيئة Windows، لقد تغلبت كثيرا في تعلمها في بادئ الأمر، حيث انه ليس من السهل على الإطلاق الانتقال من بيئة DOS ل ++C إلى بيئة Access تحت Windows مرةً واحدة.

لقد تعلمت نظام Access بالقوة في بادئ الأمر، فهو كان شرط من شروط التوظيف لأول وظيفة لي كمبرمج، ومرت أعوام قبل أن اعتاد على بناء البرامج بصورة سليمة بواسطته ذلك النظام، حيث أن قواعد البيانات ترتكز على أسس علمية كنت أجهلها في بادئ الأمر، كما إن نظام Access نفسه يعتمد على مواصفات و معايير قياسية خاصة بمايكروسوفت والتي اكتشفتها لاحقاً.

طوال السنوات الأولى كنت اشكك في مقدور Access من أن تبني برامج قواعد بيانات بصورة سليمة، مع أنني بنيت بعض البرامج بواسطتها، وساهمت كثيرا في تخريب سمعتها، حتى أتى اليوم الذي قررت فيه أن اتركها واتجه إلى لغات البرمجة الأخرى، ولكنني كنت أعود إليها دائما، فهي الأفضل في قواعد البيانات بالنسبة إلي.

قد يستغرب البعض من كلمة "تطبيقات عملاقة"، فكما يشاع في العالم تكنولوجيا SQL Server مخصصة لقواعد البيانات الصغيرة والمتوسطة الحجم، بالطبع أنا لا اتفق مع ذلك على الإطلاق، فالشائعات في النهاية ناتجة عن المنافسة والصراع بين الشركات، و مع ذلك فأنا اترك الأمر إليكم لتكتشفوا قوة ذلك النظام وإمكانياته في صفحات الكتاب اللاحقة، وكم عملاقة هي التطبيقات التي يمكننا بنائها من خلاله.

أنا اشكر السيد باسم المصري (أول مدير لي في أول عمل كمبرمج)، والذي وفر لي هذه التكنولوجيا منذ صدور الأول، وساعدني في إتقانها والحصول على الكثير من الإصدارات اللاحقة منها ، كما انه ساعدني من توسيع وجهة نظري لأتمكن من رؤيتها من وجه النظر الإدارية أيضاً ، إضافةً إلى وجه النظر البرمجية.

كما أشكر السيد الدكتور أنور عكاشة، والذي علمني علوم الكومبيوتر في الجامعة بصورة شيقة ومختلفة عن الطريقة الطبيعية، فمنه تعلمت بان لا احفظ كيف تعمل تكنولوجيا الكومبيوتر فقط، ولكن الأساس هو أن ابحث و أفهم كيف تعمل تكنولوجيا الكومبيوتر، لان ذلك هو الطريق نحو التعلم و الإبداع.

و بالطبع أشكر أبي السيد يوسف ترزي، والذين بدون دعمه لم أكن احلم بان اصل لما وصلت إليه الآن، فرؤيته الثاقبة للمستقبل، وتحليله العميق للأحداث، كان دائماً الضوء الساطع الذي لا يزال يرشدني حتى اليوم في طريقي إلى المستقبل.

مقدمة الكتاب

تتطور نظم قواعد البيانات بسرعة كبيرة للغاية، ففي الأعوام الأخيرة ظهرت العديد من التكنولوجيا الجديدة، كما اختفت الكثير من التكنولوجيا السابقة في مجال قواعد البيانات.

الهدف من هذا الكتاب هو توفير مرجع باللغة العربية لجميع المبرمجين العرب الذين يرغبون بالتوجه نحو عملية بناء التطبيقات العملاقة بواسطة تكنولوجيا مايكروسوفت، و توفير ذلك المرجع مجاناً عبر إنترنت، بهدف إفادة المواطن العربي قدر المستطاع.

كما تفتقر سوقنا العربية اليوم من كتب في هذا المجال، وان وجدت بعض الكتب فهي في اغلب الأحيان تكون ترجمة من الكتب الإنجليزية، و مع أن الكتب المترجمة كتب ممتازة، إلا أنها تفتقر في الكثير من الأحيان لتوضيح الكثير من الأمور التي تهم المبرمج العربي، فبناء البرامج باللغة العربية يختلف عن بناء البرامج باللغة الإنجليزية.

يشرح الكتاب في فصوله القادمة قواعد البيانات، تاريخ تطورها، و نوعين أساسيين منها، قواعد البيانات ذات العلاقات، وقواعد البيانات المتعددة الأبعاد (و هي آخر ما توصل إليه العلم في مجال قواعد البيانات).

كما يشرح بأدق التفاصيل لغات البرمجة التالية:

- لغة الاستعلام T-SQL اختصار ل Transact Structured Query Language المستخدمة للاستعلام من قواعد البيانات ذات العلاقات عموماً و Microsoft SQL Server خصوصاً.
- لغة الاستعلام MDX اختصار ل Multidimensional Expression المستخدمة للاستعلام من قواعد البيانات متعددة الأبعاد عموما، ونظام Microsoft Analysis Services خصوصاً.
- لغة VBA اختصار ل Visual Basic للتطبيقات، والمستخدمة من كلاً من نظام Microsoft SQL Server و نظام Microsoft Access XP
- لغة Microsoft Visual Basic Script وهي لغة مصغرة من لغة Visual Basic و مستخدمة بكثرة في بناء صفحات الإنترنت المعتمدة على قواعد البيانات.

و يشرح بالتفاصيل أيضا البرامج التالية:

- Microsoft SQL Server 2000
- Microsoft SQL Server Analysis Services
- Microsoft Access XP (2002)
- Distributed Component Object Model "DCOM+"
- وبعض التكنولوجيا الأخرى المرتبطة بموضوع الكتاب

و بالطبع سوف تجدوا الكثير من الأمور النظرية والتي ستساعدكم في بناء الفكرة العامة لعملية بناء التطبيقات العملاقة بواسطة تلك التكنولوجيا.

أتمنى أن تستمتعوا و تستفيدوا بقراءة هذا الكتاب

مقدمة إلى قواعد البيانات

البرامج ما قبل قواعد البيانات

في السنوات الأولى من عملية بناء البرامج كانت معظم برامج الكومبيوتر تستخدم طريقة خاصة بها لتوصيف بياناتها ، حيث كانت تحفظ تلك البيانات على شكل ملفات في جهاز الكومبيوتر ، وكانت كل شركة من شركات صناعة البرمجيات لها طرقها الخاصة في بناء ملفات برامجها ، بل أكثر من ذلك ، حيث كان كل برنامج من البرامج له طريقته الخاصة في حفظ بياناته في ملفات بجهاز الكومبيوتر.

قد لا يرى البعض مشكلة كبيرة في بادئ الأمر بان يحفظ البرنامج بياناته في ملفات خاصة به ، إلا انه مع بناء أول أجزاء البرامج تبدأ المشاكل بالظهور ، وتزداد كلما ازداد حجم البرنامج و كمية البيانات المخزن بواسطته.

بالطبع قد لا يتخيل الكثير من المبرمجين الجدد مشكلة استخدام الملفات العادية ، حيث أن معظمهم اعتاد على ملفات قواعد البيانات ، وبات يستخدمها في حفظ جميع بيانات برامجه ، لذا سأبدأ كتابي هذا بمثال عن شركة ، وسأستخدم هذه الشركة في كل فصول هذا الكتاب تقريباً ، ولتكون الأمور أكثر و واقعية فسأستخدم الشركة التي اعمل معها ، حول ذلك حتى أتمكن من كتابة حقائق حدثت بالفعل في هذا الكتاب.

أنا أعمل في شركة تقوم بتطوير نظام خدمات المياه في الضفة الغربية بفلسطين، الشركة فرنسية والتمويل للبنك الدولي، وأهدافنا كثيرة ، ومنها أحد أهدافي و هو تطوير نظام الكومبيوتر ، واستبدال برامج إدارة المشتركين القديمة و التي تعتمد على ملفات عادية في بعض الأحيان ، وقواعد بيانات بدائية في أحيان أخرى ، كل هذه النظم موزعة في منطقة جغرافية واسعة مكونة من أكثر من أربعون بلدية تقريباً لكل منها نظامها الخاص.

برامج الكومبيوتر التي نطور جزء منها و نشتري الجزء الآخر منها لاستبدال النظم الموجودة حالياً، تلك البرامج تختص بإدارة مشتركين المياه والصرف الصحي، فهي التي تطبع الفواتير، وتحدد المشترك على الخرائط الجغرافية، وهي التي تقوم بالعمليات المحاسبية لاحتساب الإيرادات والمصروفات، وهي التي تحتسب فاقد المياه في الشبكة، وتصدر أوامر العمل إلى الأقسام المختلفة لإصلاح الأعطال في الشبكة، وهي تقوم أيضا بالكثير من الأمور الأخرى، أتقول هذا فقط كعملية توضيح للأسماء التي سأستخدمها لاحقا ، فعندما ترى كلمة مشترك ، فاعلم بأنني اقصد مشترك مياه ☺

عند بناء برنامج يقوم بتخزين بياناته في ملفات من صنعه فقط ، فهو يمر في مراحل دراسة موسعة للغاية ، حيث انه من الصعب إجراء أية تعديل على نظام الملفات بعد إنشاء الملفات لأول مرة ، حيث يضطر المبرمجون إلى بناء خوارزميات للوصول إلى البيانات ، تلك الخوارزميات دقيقة و محسوبة إلى أدق التفاصيل ، حيث أن تغير بسيط فيها قد يؤدي إلى تغير في كامل هيكلية البرنامج ، فعلى سبيل المثال ، لو خصصنا مساحة خمسون حرفا للاسم الرباعي للمشترك مثلا ، وفي منتصف مرحلة بناء البرنامج وجدنا بان قد يكون هناك مشتركين قد تتجاوز أسمائهم الخمسون حرفاً ، فسيكون من الصعب إعادة تعديل البرنامج ليستوعب الأحرف الإضافية ، فهذا يعني إعادة تكوين خوارزميات الحفظ والاسترجاع التي أنشئاها وبنينا برنامجنا عليها.

بالطبع عملية إضافة معلومة أخرى لنفس المشترك السابق عملية اعقد ، فلو أن نسينا بان نضع في البرنامج ، تاريخ ميلاد المشترك مثلا ، والذي قد يساعدنا في عملية ما في وقت لاحق ، فإضافتها إلى البرنامج ستكون اصعب ، وفي معظم الأحيان ستكلف مالاً إضافياً ، فالشركة الصانعة للبرنامج ستحتاج إلى جهد وموظفين للقيام بذلك.

كما إن عملية تطوير البرنامج مع الوقت تبدوا صعبة أيضاً ، فسرعان ما نكتشف بعد شهر أو شهرين من تشغيل البرنامج بعدم وجود بعض التقارير ، أو بوجود إستثنائات يجب على النظام تقبلها ، ولكن مسألة التعديل هنا أصبحت اعقد بكثير ، فعوضا عن صعوبة إعادة بناء الخوارزميات ، فهناك مشكلة أخرى ، وهي بان الملفات تحتوي على لبيانات ، و لذلك سنكون بحاجة إلى بناء برامج خاصة تقوم بنقل البيانات من الملفات القديمة إلى الملفات الجديدة و التي ستنتج بعد عملية التعديل.

في الكثير من الأحيان آيا تطلب الإدارة الكثير من التقارير الإضافية والتي لم يحسب حسابها لحظة إنشاء البرنامج ، وبما أن البرنامج يستخدم ملفات خاصة به ، فالطريقة الوحيدة لبناء التقارير هو طلب ذلك من الشركة المصنعة للبرنامج ، والتي بدورها تفرض الأسعار التي تريدها ، فهي بنت البرنامج وهي الوحيدة القادرة على بناء التقرير لذلك البرنامج.

وحتى لو وافقنا على عملية بناء التقارير الجديدة ، فمعظمها سيحتاج لبنائه الكثير من الوقت ، حيث انه قد لا تفيدنا بعد مدة ، وهذا يحصل مع الكثير من الشركات التي بحاجة إلى بيانات في اللحظة الحالية.

أما عملية الحفاظ على الملفات من التلف فكانت مشكلة أخرى، ولذلك سوف تجدوا بان معظم أجهزة الكومبيوتر القديمة وضعت في غرف خاصة بها و تمت عملية تزويدها بالطاقة عن طريق بطاريات خاصة UPS إضافة إلى التيار الكهربائي العادي ، حيث تتدخل تلك البطاريات للعمل في غضون أجزاء بسيطة من الثانية بعد انقطاع التيار الكهربائي ، حتى لا يشعر الجهاز بذلك.

فانقطاع الكهرباء عن الجهاز بسبب خلل كهربائي ، أو بسبب خطاء أحد الموظفين ، قد يكلف الشركة غالياً ، فلو قطعت في منتصف عملية الحفظ ، قد يؤدي بالبرنامج بان لا يتعرف على ملفاته عند إعادة تشغيل الجهاز ، بل اخطر من ذلك ، فقد يدمرها بدون أن يعرف ذلك ، فهو يعتمد على خوارزميات للتعامل مع البيانات و تلك الخوارزميات قد تتصرف بطريقة أخرى في الملف المعطوب ، فهي لا تفهم معني وجود مشكلة في الملف.

و كخلاصة عملية بناء برامج تعتمد على الملفات الخاصة بها فقط عملية معقدة ، فقبل بناء البرنامج الشركة بحاجة إلى إجراء دراسات موسعة لتحاول تغطية معظم الأمور الخاصة بالبرنامج وذلك قبل البدء بعملية بنائه ، لان بعد ذلك فالتعديل صعب للغاية ، ومع ذلك تم بناء مجموعة هائلة من البرامج التي تستخدم ملفات خاصة بها طوال السنوات السابقة ، وذلك لان الملفات كانت الطريقة الوحيدة لتخزين البيانات لسنوات طويلة.

قواعد البيانات ، ما هي بالضبط ؟

كما ذكرنا سابقاً ، من المستحيل على شركة برمجة بان تبني برنامج قادر على تغطية كافة الاحتمالات والأسئلة التي قد نطلبها منه ، و لو حاولنا بناء البرنامج ليتمكن من ذلك ، لكلفنا الكثير من المال وسنوات عديدة من التطوير.

لذلك اتجهت أنظار شركات البرمجة إلى بناء نظام تخزين موحد للبيانات ، بحيث يكون ذلك النظام قادر على التالي:

الفصل بين الهيكلية المنطقية و الفيزيائية للملف

بكلمات أخرى ، يجب أن لا اهتم كمبرمج بتفاصيل حفظ الملف فيزيائيا على القرص ، حيث على نظام إدارة قواعد البيانات الاهتمام بذلك ، و في نفس الوقت لا يجب أن تتأثر البيانات التي يحتوي عليها الملف عندما أغير خريطة الملف ، بكلمات أخرى ، لو خزنت في الملف اسم المشترك و رقم هويته و تاريخ ميلاده ، وخزنت مثلاً بيانات ألف مشترك ، ورغبت في لحظة من اللحظات إضافة عمود جديد إلى الملف وبه رقم هاتف المشترك ، يجب أن أتمكن من إضافة هذا العمود إلى الملف بدون أن يتأثر الملف وبياناته و البرامج المعتمدة عليه.

طريقة موحدة لحفظ واسترجاع البيانات

يجب على النظام الجديد أن يتبع طريقة موحدة عالمية لحفظ البيانات واسترجاعها ، ومن هنا ظهر مفهوم حفظ البيانات على شكل جداول مكونة من أعمدة وصفوف ، وظهرت أيضاً لغة SQL و هي اختصار ل Structured Query Language كلغة موحدة للوصول إلى البيانات و معالجتها.

حماية البيانات من التلف

يجب على النظام الجديد أن يحفظ البيانات من التلف ، فلو قطعت الكهرباء مثلا في منتصف عملية التسجيل ، تبق مشكلة إصلاح الملفات مشكلة ذلك النظام ، فعند إعادة التيار الكهربائي يجب أن أتمكن من استخدام ملفاتي بدون أن تحتوي على أية مشاكل.

حماية البيانات من سوء الاستخدام

يجب على النظام الجديد أن يحافظ على البيانات من سوء استخدام بعض الموظفين لها ، وحجبها من الموظفين الذين لا صلاحيات لهم بالإطلاع عليها ، ويشفرها ، ويقوم بكافة عمليات الحماية اللازمة عليها.

المحافظة على تكامل البيانات منطقياً ككتلة واحدة

يجب على النظام الجديد أن يحافظ على البيانات ككتلة واحدة ، ففي البرامج الكبيرة على سبيل المثال تقسم البيانات في الكثير من الملفات (الجداول) ، فهناك ملف للمشترك مثلا ، وملف آخر لكافة الحركات المالية الخاصة بالمشترك مثلا ، فعلى النظام الجديد أن ينتبه بأنه لا يجوز أن أضيف حركة ما على مشترك ، بدون أن يكون هناك مشترك في الأساس ، كما لا يجوز أن احذف مشترك وان اترك حركاته المالية كما هي في الجداول الأخرى.

الوصول المباشر إلى البيانات عند الحاجة

وهو أحد أهم الأمور ، حيث عندما يتعذر على البرنامج تزويدي بتقرير ما ، فيجب أن تتوفر لدي الفرصة لان اصل إلى تلك البيانات بدون البرنامج نفسه ، واخذ البيانات التي أريدها من الملفات.

و بناءاً على النقاط السابقة والكثير من النقاط الأخرى بدئت بعض مؤسسات الأبحاث والشركات الكبيرة وضع اللمسات الأولى لنظم إدارة قواعد البيانات ، وبدئت بعد فترة أوائل نظم قواعد البيانات بالظهور ، والتي حملت عصر جديد لصناعة البرمجيات.

يمكن بان نجمل بان قاعدة البيانات هي طريقة تخزين موحدة للبيانات ، بحيث يمكن للبرامج العادية استخدامها لتخزين بياناتها ، كما يمكن في نفس الوقت الوصول إلى البيانات مباشرة بدون الحاجة إلى البرنامج نفسه ، ولا يجب علينا أن نهتم بتفاصيل طريقة الحفظ الفيزيائية لتلك البيانات.

تاريخ تطور قواعد البيانات

بدء يظهر مفهوم قواعد البيانات منذ بداية السبعينات ، لا تتوفر لدي الآن معلومات دقيقة عن أول من كون ذلك المفهوم ولكننا يمكن أن نعتبر بأنه ظهر وتطور مع تطور برمجيات الكومبيوتر.

بالطبع لم تنشئ قواعد البيانات كما نعرفها اليوم ، فلقد كانت هناك الكثير من المحاولات و البرامج التي سمعنا عن بعضها ولم نسمع عن الآخر نتيجة لاندثاره.

في البداية انتشرت في الأجهزة العملاقة ، ومن ثم انتقلت إلى الأجهزة المتوسطة ، وكانت صناعتها حكراً على الشركات الكبيرة ، و مع الوقت و مع انتشار الأجهزة المكتبية ، بدئت تشق قواعد البيانات طريقها إلى الشركات الصغيرة ، والمكاتب العادية ، حتى أنها أصبحت متوفرة الآن كجزء من نظام التشغيل نفسه ، فنظم Microsoft Windows تأتى محملة مسبقا بتكنولوجيا ADO و ODBC للتعامل مع قواعد البيانات.

و هناك اليوم مؤسسات مسئولة عن وضع المواصفات القياسية لقواعد البيانات و للغة SQL لغة الاستعلام الأساسية ، ومع ذلك فهناك بعض الاختلافات ، حيث حاولت بعض الشركات بان تسهل التعامل مع لغة الاستعلام و من هنا أنشئت لغات استعلام خاصة بها معتمدة على لغة الاستعلام الأساسية ، ولذلك اليوم نجد لغات استعلام مثل SQL و SQL 92 و Transact SQL و PL/SQL وغيرها.

و مع اختلاف لغات الاستعلام ، فهي تبقى محافظة على الأوامر الأساسية للتعامل مع البيانات و التي حددتها الهيئة المسئولة عن SQL لذلك ستجد عندما تتعلم أية نوع من هذه اللغة بأنه عليك تعلم اللغة الأم في البداية.

اليوم هناك الكثير من نظم إدارة قواعد البيانات ومن تلك النظم SQL Server, Oracle, Informix, Sybase, Access, DBASE, FoxPro ، والعديد غيرها ، ولكلا منها مميزاته و مساوئه ، ومجالات استخدامه.

الأجزاء الرئيسية لقواعد البيانات

تتكون جميع نظم قواعد البيانات عموماً وقواعد بيانات مايكروسوفت خصوصاً من أجزاء رئيسية ، وسأذكر تلك الأجزاء بصورة سريعة ، وسأشرحها نظريا فقط ، حيث يلي في صفحات الكتاب اللاحقة الشرح العملي لكل جزء بالتفصيل.

لنتمكن من تخزين واسترجاع البيانات بشكل سليم علينا اتباع أسس وقوانين ومواصفات قياسية للقيام بذلك ، حيث بدونها تصبح المسالة بدون جدوى ، أي انه هناك يجب أن تكون طريقة موحدة لتخزين تلك البيانات وعلى الجميع الالتزام بها ، كما أن هناك يجب أن تكون طريقة موحدة لاسترجاع ومعالجة البيانات وعلى الجميع الالتزام بها أيضاً.

تخزين البيانات في جداول

الجداول هي الطريقة التي تخزن بها قواعد البيانات بياناتها ، فالجدول عبارة عن الملف الذي يستخدمه البرنامج لتخزين بياناته ، حيث يمكننا بناء جدول للمشتركين و جدول آخر لحركات المشتركين المالية وجداول عديدة أخرى.

الجداول تخزن بياناتها على شكل صفوف و أعمدة ، كما يمكنك إضافة أعمدة جديدة للجداول متى تشاء ، بدون أن يؤثر ذلك على البرامج المستخدمة لتلك الجداول أو على البيانات الموجودة في تلك الجداول ، فخادم قواعد البيانات هو المسؤول عن ذلك أولاً و أخيراً.

لكل عمود نوع معين من البيانات التي يمكن أن نخزنها فيه ، كما يجب أن نحدد عدد الأعمدة و أسمائها ونوع البيانات التي ستخزن في كل منها ، بالطبع ذلك ضروري للبرنامج الذي سيستخدم البيانات ، فأنواع البيانات مختلفة فقد تكون نص وتشمل الأرقام أيضا ، أو أرقام فقط أو تواريخ أو ما إلى غير ذلك مثل صور و وسائط متعددة وغيرها

إذن من الضروري كل عمود أن يحتوي على نوع معين من البيانات ، وقد يسأل البعض لماذا أنواع مختلفة من البيانات لماذا ليس بيانات من نوع نص والتي يمكن أن نخزن بها كل شيء ، والجواب سهل فأنواع البيانات تساعدنا على تخفيف الذاكرة المستخدمة لتخزين تلك البيانات ، فقد يكون من غير المهم الاهتمام بنوع البيانات إن قررنا تخزين 100 مشترك في برنامج إدارة المشتركين مثلا ، فلن تأخذ الكثير من مساحة القرص الصلب ، ولكن الأمر يختلف إن سجلنا خمسة آلاف مشترك مثلا مع وحركات حساباتهم لمدة خمسة أعوام على سبيل المثال.

قد لا يقتنع البعض ويقول "الأقراص الصلبة اليوم تتسع لاستيعاب عشرات آلاف المشتركين و حركات حساباتهم لقرون" وهذا صحيح ، ولكن في حال إن خزنا المشتركين وكل بياناتهم على شكل نصوص فقط فلن نتمكن من إجراء الحسابات على حركاتهم المالية ، وهذا صحيح مائة بالمائة ، حيث إن لغة SQL غير قادرة على جمع رقمين خزنا في مكان مخصص للنص ، وان تمكنت في بعض أنظمة قواعد البيانات فإننا سنحصل على الرقم 11 عند جمع الرقمين 1 و 1 ذلك في حال تمثيلهما على شكل نصوص ، وسنحصل على الرقم 2 في حال تمثيلهما على شكل أرقام.

بناء علاقات بين الجداول و الحفاظ على تكامل البيانات

بالطبع عملية تخزين البيانات في جداول عملية رائعة ، ولكن حتى ابسط برامج قواعد البيانات بحاجة إلى اكثر من جدول لتخزين البيانات ، وقد يطرح البعض سؤال لماذا نجزئ البيانات إلى جداول ، فخزن كل ما له علاقة بالمشتركين و حركات المشتركين المالية معا.

حتى أتمكن من الإجابة على ذلك تخيلوا بأننا بنينا جدول لتخزين بيانات المشتركين ، وقمنا بإنشاء أعمدة لاسم المشترك ، و تاريخ اشتراكه ، و المبلغ المطلوب أن يدفعه لنا (مقابل خدمات توصيل المياه) ، و بعض الأمور الأخرى ، ولان أتى دور دفعات المشترك ، حيث يمكننا وضع أعمدة للدفعة في شهر يناير من العام أتفين و واحد مثلاً ، و عمود آخر للدفعة في شهر فبراير ، وهكذا.
ولكن هل يمكننا أن نغطي كل الاحتمالات ، جميع الأعوام و الأشهر ، بالطبع لا ، هذا بالمناسبة يعتبر هدرا للذاكرة ولعملية تصميم قاعدة البيانات.

لذلك علينا ببناء جدول آخر للحركات المالية للمشترك ، وربط هذين الجدولين معاً عن طريق مفاتيح ربط خاصة سنشرحها لاحقاً.

تذكروا دائما ، لو تكررت البيانات في قواعد البيانات التي تصنعوها فهناك خطاء ما ، فلا يعقل في أية حال من الأحوال أن يكون لدي جدولين أو اكثر يحتويان على نفس المشترك وبياناته ، و إن حدث ذلك فهذا يعتبر خطاء في تصميم قاعدة البيانات

العلاقات تضمن لنا بتكامل البيانات ، حيث لا يمكن أن نضيف حركة مالية في جدول الحركات المالية للمشترك ، بدون أن نقوم بإضافة المشترك أساسا إلى قاعدة البيانات ، كما لا يمكننا حذف مشترك من المشتركين و ذلك بدون حذف حركاته ، وإلا اختلت البيانات في قاعدة البيانات.

يمكن أن نعرف العلاقات بأنها عبارة عن أوامر لخادم قاعدة البيانات ، حيث تعطى تلك الأوامر لحظة إنشاء قاعدة البيانات وتخبره مثلا بان يرفض إدخال أية حركة مالية إن لم تتبع لمشترك ما ، كما قد تخبره تلك الأوامر بأنه بمجرد حذف مشترك من قاعدة البيانات اذهب إلى جميع البيانات المرتبطة بذلك المشترك واحذفها من جميع جداول قاعدة البيانات ، كما يمكن القول بان قاعدة بيانات بدون علاقات هي ليست قاعدة بيانات.

قد يقول البعض "يا أخي عن ماذا تتكلم ، حذف مشترك و حركاته المالية قد يخل بنظام المحاسبة الداخلي كليا ، فكيف نحذف المبالغ التي دفعها ؟" ، وهذا صحيح حيث انه لا يجب أن نجعل البرنامج يقوم بذلك ، وهناك علاقات مخصصة لذلك ، كما إن هناك ما يسمى بال Triggers والتي قد يسميها البعض في عالمنا العربي "الزنادات" أنا لا احب هذا الاسم فهو لا يعبر لي عن أية شيء ، ولكن حتى اليوم لا يوجد موقع إنترنت واحد يوحد الأسماء العربية ليلتزم الجميع به ، ولكن هذه مشكلة أخرى.

إذن أل Triggers يمكن برمجتها لتقوم بحذف مشترك (إن أصررنا على حذفه ☺) مع القيام بعمليات التعديل اللازمة على الجداول المحاسبية لضمان تكاملها.

إذن العلاقات هي بدرجة أهمية بالغة ، فهي المسيطر على تكامل قاعدة البيانات ، يمكن القول قاعدة بيانات بدون علاقات هي ليست قاعدة بيانات ، لذلك كن حريق للغية وأنت تصم قاعدة بياناتك.

الاستعلام و معالجة البيانات بواسطة لغة SQL

الاستعلامات عبارة عن أوامر بلغة SQL مخصصة للقيام بعملية ما على البيانات ، وبالمناسبة الاستعلامات لا تقوم بعملية الاستعلام فقط كما قد يوحي اسمها بذلك ، ولكنها أيضا قادرة على إجراء التعديلات على البيانات في قاعدة البيانات نفسها ، بل إنها قادرة أيضا على بناء جداول أخرى في قاعدة البيانات.

معظم الاستعلامات تعامل معاملة الجداول ، حيث إن نتيجتها دائما تأتى على شكل جدول ، فلو طلبنا من قاعدة البيانات أسماء جميع المشتركين لدينا والذين لم يدفعوا ثمن المياه المستهلكة منذ سنة مثلاً ، فستصلنا البيانات على شكل جدول ، وهذا طبيعي حيث إن قاعدة البيانات في النهاية هي عبارة عن جداول مترابطة مع بعضها البعض.

بالطبع معظم قواعد البيانات تسمح لتخزين الاستعلامات بداخلها لتستخدم اكثر من مرة ، ولا تخزن نتيجة الاستعلامات طبعا بل يخزن الكود الخاص بالاستعلام ، حيث أن بمجرد طلب الاستعلام يقوم النظام بتنفيذه و إعطاء النتيجة إلى المستخدم النهائي.

على العموم ستجدوا في SQL Server 2000 استعلامات غريبة للغاية ، ففي حالة قواعد البيانات العملاقة قد يطلب استعلام ألف مرة في اليوم ، وقد تعدل البيانات مرة أسبوعيا ، لذلك ستجدوا بان النظام قد يحافظ على بيانات الاستعلام في حال عدم تعديل البيانات في الجداول ، بل اكثر من ذلك فقد يفهرس أعمدة الاستعلام وأمور أخرى من هذا النوع موجودة فقط في SQL Server والتي سنتحدث عنها لاحقاً بالتفاصيل.

يسألني البعض أحياناً "هل يمكننا توجيه أية سؤال لنظام إدارة قواعد البيانات مهما كان ذلك السؤال؟" ، الجواب هو نعم ، يمكن توجيه أسئلة بسيطة مثل "ما هو عدد المشتركين لدينا" ، كما يمكنك توجيه أسئلة معقدة بطلب احتساب فواتير المشتركين لو زدنا سعر المياه عند 1 و خمسة و عشر سنتات مثلا ، ويمكنك توجيه أسئلة بالغة التعقيد تربط الكثير من الجداول أو تقوم بالكثير الكثير من الحسابات معا.

حماية قواعد البيانات (المستخدمين و صلاحيات الوصول إلى البيانات)

جزء مهم آخر من قواعد البيانات هو الصلاحيات والمستخدمين ، فتخيلوا لو كان الجميع يستطيع الوصول إلى البيانات وتعديلها ، لكانت كارثة حقا ، حيث أن خادم قواعد البيانات يعطينا إمكانية إغلاق البيانات بواسطة أسماء مرور وكلمات سر.

وبهذا قد نعطي الإمكانية لقسم إدارة المشتركين مثلا من الاطلاع على بيانات المشتركين بدو الاطلاع على بعض المعلومات المالية الحساسة الخاصة بهم ، ونعطي صلاحية للمحاسبين في دائرة الحسابات تلك الصلاحية فقط ، ونمنع قسم الصيانة من الوصول إلى جدول المشتركين تماماً ،حيث انه لا علاقة لهم بذلك الجدول على الإطلاق.

إذن خادم قاعدة البيانات هو المسئول عن الصلاحيات لا البرامج نفسها ، فلو لم تتوفر لديك الصلاحيات فلن تتمكن من الوصول إلى البيانات مع البرنامج أو بدونه ، وهذه أحد أهم مميزات قواعد البيانات.

ويمكن تقسيم الصلاحيات إلى صلاحيات للقراءة ، للكتابة ، للإضافة فقط بدون إمكانية الاطلاع على البيانات الأخرى في الجدول ، أو دمج صلاحيات مع بعضها مثل للقراءة والكتابة معاً ، وقد نقسم الصلاحيات على الجدول نفسه من الداخل ، حيث نمنع إمكانية تغير اسم المشترك مع إعطاء إمكانية تغير عنوانه.

بالطبع إدارة الصلاحيات تختلف من محرك قواعد بيانات إلى آخر ، أما مع SQL Server 2000 فيمكنكم الحصول على نتائج مذهلة.

حماية البيانات من التلف

خادم قاعدة البيانات لا يجب أن يكون مسؤولا عن عملية تكامل البيانات بداخل الجداول فقط ، بل اكثر من ذلك ، فتخيلوا إن قطعت الكهرباء في منتصف عملية تخزين البيانات ، ماذا سيحدث ، فقد نكون حفظنا بيانات المشترك و لن نتمكن من تكملة عملية حفظ الحركات المالية ، أو افظع من ذلك ، قد تكون عملية الحفظ غير المكتملة دمرت ملف قاعدة البيانات.

لذلك على محركات قواعد البيانات أن تتبع طريقة تمنع إمكانية إحداث تلف في تلك الملفات مهما كانت الظروف ، فالبيانات كلفتنا الكثير من المال ، فهي عمل الشركة ، فضياع حركات آخر نصف ساعة من حركات البيانات في بنك قد تكون كارثة ، وضياع حركة واحدة قد تكون مشكلة كبيرة للبنك نفسه.

وهنا يكمن الفرق الكبير بين البرامج العادية التي تستخدم ملفاتها الخاصة وبين البرامج التي تستخدم قواعد البيانات فقواعد البيانات تصلح ملفاتها بنفسها فوراً ، أما في حال استخدام الملفات يجب أن ننادي الشركة المصممة للبرنامج والتي قد تأخذ مبالغ طائلة إصلاح العطل ، لذلك إن لم يدمر البرنامج نفسه الملفات بصورة اكبر عندما تعودت الكهرباء من جديد

الاستعلام باللغة الإنجليزية English Query Language

الاستعلامات باللغة الإنجليزية هو مشروع طموح لمايكروسوفت بهدف استبدال لغتي SQL و MDX ، حيث يمكننا باستخدام لغة الاستعلام هذه طلب البيانات من قاعدة البيانات مباشرة باللغة الإنجليزية.

يتمتع ذلك النظام بنوع من الذكاء الاصطناعي ، ولكنه بحاجة إلى توصيف خاص لهيكلية الجداول في قاعدة البيانات ليتمكن من العمل ، وبمجرد توصيف الجداول و الأعمدة التي تحتوي على البيانات يتعرف النظام عليها و يبدأ بتنفيذ الطلبات القادمة باللغة الإنجليزية.

الاستعلام باللغة الإنجليزية هو أحد افضل أنواع الاستعلامات بالنسبة للإدارة العليا ، حيث كل ما يجب أن تضعه في غرفة المدير هي شاشة و مربع حوار ليتمكن من الكتابة إلى الكومبيوتر ، و يقوم الكومبيوتر بدوره بالإجابة على الأسلة بلغة صحيحة أيضاً.

كما يرسل ذلك النظام جميع الأسئلة التي لم يتعرف عليها النظام ، أو جميع الكلمات الجديدة أيضا إلى مدير قواعد البيانات ، أو الشخص المسئول عنها ، حتى يتمكن من أضافتها إلى النظام وتعليمه إياها.

سنشرح طريقة استخدام تلك اللغة و كيفية أضافتها إلى برامجنا (الجزء الإنجليزي من برامجنا) ، ونأمل بان تعرب مايكروسوفت مستقبلاً تلك اللغة.

قواعد البيانات المركزية

يسألني الكثير من الإخوة دائما سؤال مهم وهو "لماذا نحن بحاجة إلى قاعدة بيانات مركزية مثل SQL Server أو غيريهما ، لماذا لا نستخدم Access بصورة مبشرة ، وماذا ستقدم لنا قاعدة البيانات المركزية أكثر مما تقدمه Microsoft Access ؟".
وهذا سؤال مهم للغاية ، حيث أن الكثير من الإخوة الذين لم يخوضوا تجربة بناء التطبيقات الكبيرة ، يتحيرون من فائدة قاعدة البيانات المركزية.

تخزن قواعد البيانات المحلية مثل Microsoft Access وغيرها ، تخزن قواعد البيانات تلك بياناتها في ملف مشترك ، حيث تخزن الجداول والبيانات وكل ما له علاقة بها في ذلك الملف.

تخزين البيانات في ملف واحد لا يعتبر مشكلة على الإطلاق إن رغبا ببناء برنامج صغير أو متوسط يستخدم من جهاز واحد ، كما لن تكون أيضا مشكلة إن تم استخدام ذلك البرنامج من عدة أجهزة على شبكة محلية صغيرة لا تعتبر بها مسألة أمن البيانات ضرورة قصوى.

ولكنها تعتبر مشكلة كبيرة في شبكات الكومبيوتر الأكبر حجما ، وتعتبر مشكلة مع التطبيقات الأكبر حجما ، وتحتوي على العديد من النواقص.

تخيلوا إن حفظنا مائة ميجابايت من البيانات في ملف Access موزعة في جداول عدة ، و وضعنا قاعدة البيانات في الجهاز المركزي ، و قام أحد الموظفين بفتح قاعدة البيانات و قام بالبحث عن مجموعة سجلات ، ولكن ليتمكن Access من البحث في الجداول العملاقة فهو بحاجة إلى نسخ كافة بيانات الجداول لديه في كل مرة تقوم بها بعملية البحث ، فلو هناك مائة ألف مشترك في جدول المشتركين وبحثنا عن المشترك باسم "محسن" فعلى برنامج أل Access طلب مائة ألف سجل من قاعدة البيانات ليتمكن من القيام بالبحث فيهما.

كما قد تكن عملية البحث اعقد ، فقد نبحث عن كل المشتركين بين أعمار معينة و حركات مالية معينة ، مما قد يؤدي بنا إلى فتح جداول الحركات الخاصة بالمشتركين ، فلو كان لدينا مائة ألف مشترك ، فجدول الحركات قد يحتوي عن ما لا يقل عن عشرة مليون سجل على الأقل ، وطلب عشرة مليون سجل عبر الشبكة لإجراء عملية بحث و فلتره ليست بالعملية السهلة.

فقد يتمكن مستخدم واحد لقاعدة البيانات العملاقة السابقة بالبحث وذلك بعد أن يكون اغرق الشبكة بالبيانات ، وبالطبع ذلك سيستغرق مدة قد تكون كبيرة نظرا لان حركة البيانات عبر الشبكة بكميات هائلة ليست بهذه السرعة ، حتى مع الشبكات الفائقة السرعة في أيامنا هذه.

وقد يتمكن أيضا مستخدمان أو ثلاثة مستخدمين على الأكثر القيام بالتعامل مع قاعدة البيانات تلك ، مع العلم بان قاعدة بيانات بذلك الحجم قد يتطلب أن تستخدم من مائة أو مائتين مستخدم معا في نفس الوقت ، ولكن لا اعتقد بان اكثر من ثلاثة مستخدمين سيتمكنوا من التعامل مع قاعدة البيانات تلك بصورة متزامنة ، نظرا لحجمها الكبير ، ونظرا لان الشبكة ستكون قد أغرقت بالبيانات وستصبح بطيئة إلى أقصى الحدود.

لذلك كانت هناك الحاجة إلى بناء خادم لقاعدة البيانات المركزية ، خادم يستقبل طلبات البرامج عبر الشبكة ويقوم بتنفيذها على الجهاز المركزي ومن ثم يرسل النتيجة إلى المستخدم ، ومن هنا ظهر نظام SQL Server

فبمجرد دمج نظام Microsoft Access مع SQL Server يقوم الأول بتخزين بياناته في قاعدة لبيانات الجهاز المركزي مباشرةً ، وتنتقل جميع الطلبات إلى الجهاز المركزي ، فلو قمت بالبحث عن مشترك على سبيل المثال ، فسيطلب نظام Access من SQL Server أن يبحث له عن المشترك ، وسترسل النتيجة فقط إلى Access وبذلك تكون قد حلت مسالة إغراق الشبكة بالمعلومات.

حتى اكثر من ذلك ، حيث سيتمكن المستخدمون العاديين من الاتصال بالجهاز المركزي حتى عن طريق المودم ، أي أبطأ صور الاتصال بالشبكة ، ومع ذلك سيمكنهم إنجاز عملهم عبره بصورة طبيعية وبدون مشاكل

ولكن لا يقتصر عمل محرك قواعد البيانات المركزية على تقليل حركة المعلومات عبر الشبكة ، بل أيضا حمايتها ، حمايتها من المستخدمين أنفسهم ، وحمايتها من التلف أيضا ، والحفاظ على تكامل البيانات.

فلو رجعنا إلى مثالنا السابق حيث تخزن كل البيانات في ملف واحد على الشبكة ، فعملية توزيع الصلاحيات على المستخدمين ستكون محدودة للغاية ، حيث أن نظريا يمكننا وضع أسماء وكلمات مرور على قاعدة بيانات أل Access العادية ، ولكن تلك الحماية هي حماية بسيطة عندما نتحدث على شبكات كبيرة ، حيث انه لن تمنع تلك الحماية المستخدم من نسخ ملف قاعدة البيانات إلى جهازه ومن ثم محاولته فك الحماية والحصول على البيانات.

بل اكثر من ذلك ، حيث يمكننا من داخل Access منعه من مسح بيانات في بعض الجداول والسماح له بحذفها من جداول أخرى ، ولكن بمجرد بان سمحنا له بتعديل أو حذف بيانات فيجب علينا بان نعطيه صلاحية قراءة وتعديل لملف قاعدة البيانات على الشبكة ، وبمجرد حصوله على صلاحية تعديل فبإمكانه مسح ملف قاعدة البيانات كلها ، حتى لو انه لن تكن له الصلاحيات اللازمة بداخل Access.

بل اكثر من ذلك ، حيث لا يمكننا من تتبع العمليات التي تحدث على قاعدة البيانات ، فقد يسيئ البعض بصلاحياته و يعدل البيانات بدون علم الإدارة بذلك.

أما مع استخدام محرك قاعدة بيانات مركزية فالأمر يختلف للغاية ، حيث لا يوجد وصول مباشر إلى ملف قاعدة البيانات ، حيث يتم الوصول إلى البيانات عن طريق خادم قاعدة البيانات فقط ، وبذلك لا يمكن لأحد الوصول إلى بيانات لا صلاحية له لوصولها.

وتحدد الصلاحيات من الجهاز المركزي ، ويمكننا تحديد صلاحيات معقد أيضا ، حيث قد نسمح بالوصول إلى ملف المشتركين ولكن قد نمنع الوصول إلى عمود الرصيد مثلاً في نفس جدول المشتركين ، حيث أن القليل فقط يجب أن يتمكنوا من ذلك ، وقد نمنع مثلا إمكانية تعديل اسم المشترك مثلا ، مع إمكانية تعديل عنوانه ، والكثير من الأمور الأخرى ، فخادم قاعدة البيانات هو المسئول عن ذلك مباشرة.

بل اكثر من ذلك ، حيث يمكننا أن نطلب من خادم قاعدة البيانات المركزية بان يتتبع كل التغيرات التي تجري على قاعدة البيانات ، ويحفظها ، لنتمكن بدورنا من اكتشاف أية استخدامات خاطئة لقاعدة البيانات ، ولتدقيق قاعدة البيانات أيضاً.

يقوم أيضا خادم قاعدة البيانات بحماية البيانات من التلف ، فهو يقوم بإصلاحات مستمرة لقاعدة البيانات في حال حصول أية تلف ، و مع أن Access يقوم بإصلاح ملف قاعدة البيانات الخاصة به أيضا ، إلا أن احتمال تعرض ملفات

قاعدة البيانات للتلف نتيجة لانقطاع التيار الكهربائي مثلا ، احتمال يزداد كلما ازداد عدد المستخدمين لقاعدة البيانات تلك معاً.

هناك الكثير ما يميز قواعد البيانات المركزية عن قواعد البيانات العادية ، واعتقد بأنكم ستكتشفوا ذلك بأنفسكم في فصول الكتاب القادمة.

مقدمة إلى قواعد البيانات المتعددة الأبعاد

تاريخ تطور قواعد البيانات متعددة الأبعاد

ظهرت فكرة تخزين البيانات في مكعبات، عوضاً عن تخزينها في جداول منذ مطلع الستينات، وذلك في الجامعات والمؤسسات التعليمية، و قامت بعض الشركات في الفترات اللاحقة ببناء نظم متكاملة خاصة بها لتخزين بياناتها على هذا النحو.

معظم تلك النظم كان يخصص لشركة ما فقط، فعلى سبيل المثال طورت شركات الطيران نظم خاصة بها، و المحلات العملاقة نظم أخرى خاصة بها، و ما إلى ذلك.

جميع النظم المطورة اعتمد على ملفات خاصة بها وليس على قواعد البيانات، مع انه في مراحل متقدمة تم بناء المكعبات اعتماداً على قواعد البيانات، ولكن في المراحل الأولى اعتمدت الشركات الكبيرة نظم خاصة بها.

في منتصف التسعينات أجريت دراسات عن تكلفة مشاريع قواعد البيانات المتعددة الأبعاد في الشركات العملاقة، حيث أتضح بأنها تنفق ما يقارب مليوني دولار للمشروع الواحد، ومنها ما يقارب المأتي ألف دولار لكل مجموعة من المكعبات المخصصة للقسم الواحد.

قد تستغرب من التكلفة العالية، ولكن بناء برنامج لشركة ما يقوم بتوصيف بياناتها على شكل مكعبات قد يستغرق الكثير من الجهد والوقت، حتى أن الكثير من المشاريع التي بدئت في منتصف التسعينات لم تنتهي أساسا، حيث أن الموضوع كان اعقد مما تصور المبرمجون لحظة بداية المشروع.

نظراً لتزايد الطلبات على قواعد البيانات المتعددة الأبعاد، بدئت العديد من الشركات تطوير نظما قياسية يمكننا استخدامها بالضبط كما نستخدم قواعد البيانات العادية، و من هنا ظهرت العديد من المنتجات و منها Oracle Data Mart و SQL Server OLAP والذي تغير اسمه لاحقا في النسخ القادمة وأصبح Analysis Services و بعض المنتجات الأخرى.

و مع ذلك بقيت تكلفة شراء نظام كهذا عالية نسبيا، فقد تصل إلى خمسة وعشرون ألف دولار للنسخة الواحدة، حتى أتت شركة Microsoft و بدئت توزع نظامها الجديد ضمن SQL Server، كأحد أجزاء النظام الأساسية، ولذلك فهو يأتي مجانا ضمن المبلغ الكلي لتكلفة أل SQL Server.

قواعد البيانات المتعددة الأبعاد، ما هي بالضبط ؟

و هي عبارة عن قاعدة بيانات تحتوي على مجموعة من مكعبات البيانات المترابطة معا و المخصصة للعمليات الإحصائية، و تتكون المكعبات بدورها من مجموعة متعددة من الجداول الثنائية الأبعاد المترابطة معا.

تستخدم جميع تلك المكعبات لعمليات إحصائية معقدة لا نستطيع أن نقوم بها عن طريق الجداول ثنائية الأبعاد، كما يمكنها أن تعطينا نظرة اشمل إلى البيانات، فعلى سبيل المثال عوضا عن حفظ البيانات الخاصة بالبضاعة في جدول ثنائي الأبعاد يحتوي معلومات عن البضاعة لوحدها فقط، يمكننا دمج عدة جداول لتكوين قاعدة بيانات متعددة الأبعاد بحيث تعطينا البضاعة مع التعديلات الزمنية، أي أن الزمن أصبح بعدا إضافيا يمكننا الاستفادة منه في إحصائياتنا.

نظرا لان قواعد البيانات المجسمة تقوم بعشرات آلاف الحسابات على المكعبات قبل إعدادها، فان معظم قواعد البيانات المتعددة الأبعاد تستمد بياناتها من قواعد البيانات العادية في فترات المساء و ذلك بعد إقفال الشركة و توقف استخدم النظام، وبهذا فهي دائما تحتوي على آخر تحديث تم على البيانات في اليوم السابق، بالطبع

هناك شركات بحاجة إلى قواعد بيانات متعددة الأبعاد فورية التحديث، وهذا أمر يمكن حدوثه أيضا، ولكن بالطبع يجب أن تحسب جيدا حجم الأجهزة المطلوبة لإنجاز ذلك العمل.

نستطيع بواسطة صفحة حسابية بسيطة في برنامج Microsoft excel مثلا أن نقوم بإيجاد النقطة التي تتقاطع بها الأرباح مع الخسائر لشركة ما، و أفضل سعر لسلعة ما، وذلك بعد تزويد النظام ببيانات الإنتاج و التكلفة الأولية، وبعض الأمور البسيطة الأخرى، و هذا مثال يدرس لكل شخص يجلس أمام نظام Microsoft Excel لأول مرة.

و لكن مع قوة هذا النظام فهو يبقى أسيرا للجداول ثنائية الأبعاد و يعجز عن القيام بالكثير من الأمور الحسابية الأخرى و التي تتطلب تدخل عوامل إحصائية أخرى، ومن هنا يأتي دور قواعد البيانات المتعددة الأبعاد.

يمكننا بواسطة نظم قواعد البيانات المتعددة الأبعاد على سبيل المثال القيام بكل الحسابات الخاصة بحملة دعائية لمجموعة من المنتجات في سوبر ماركت عملاق مثلا، تخيلوا سوبر ماركت كبير للغاية لديه كل أنواع البضائع الالكترونية مثلا، و يعني في نفس الوقت من قلة الزيارات، و من قلة عمليات الشراء، بالطبع الطريقة الوحيدة لجلب الزبائن هي الحملات الإعلامية.

ولكن الحملات الإعلامية نفسها لا تكفي، أو ليست بهذا المشجع الكبير والذي سيجلب الزبائن إلى هذا السوق الضخم، و كلنا نعلم بان المفتاح يكمن في تخفيض الأسعار، ولكن تخفيض الأسعار أمر ليس سهل على الإطلاق، فقد تأتي الزبائن و تشتري جميع السلع التي خفضت سعرها و من ثم تذهب و ينتهي الأمر.

لذلك نحن بحاجة إلى سياسة تخفيض ما نقوم بها بتخفيض أسعار بعض المنتجات الالكترونية، و في نفس الوقت وضع تلك المنتجات بجانب منتجات أخرى أفضل منها و لم تخفض أسعارها، بهدف إغراء المشتري بشرائها.

هنا يأتي دور قواعد البيانات المتعددة الأبعاد، حيث يمكننا استخدامها للقيام بعملية حسابية معقدة للغاية تعطينا كل الاحتمالات الممكنة لعملية تخفيض أسعار المنتجات، بصورة تأتي لنا بأكبر ربح و فائدة ممكنة، و من هنا كل ما علينا أن نعبئ النظام بما لدينا من أصناف و أسعارها و كمياتها و حتى طرق ترتيبها، و نعبئه بمقاييس ذوق المستخدم، وبعض العوامل الأخرى، ومن ثم نبدأ بعملية الحساب العملاقة لنأتي بأفضل خطة إعلامية ممكنة، وهذا بالمناسبة ما يحدث في جميع المحلات في الولايات المتحدة، ومن اشترى صحيفة هناك و قرأ الإعلانات سيفهم قصدي..

بالطبع استخدامات قواعد البيانات المتعددة الأبعاد لا تقتصر على المحلات العملاقة، بل إنها تمتد إلى جميع المجالات الأخرى، حيث تستخدمها بعض شركات الطيران لتحدد أفضل أوقات مواعيد رحلات الطياران و أفضل المواسم و أفضل الأسعار في تلك المواسم لتأتي بأفضل عائد ممكن، فكما نعلم طيران الطائرة نصف فارغة أو بمقاعد فارغة يعتبر خسارة لشركة الطيران، كما إن بقائها في مطار لمدة طويلة يتتبع رسوم مكوث وصيانة وأمور أخرى، هذا عوضا على إنها لا تعمل و هذه خسارة أخرى.

بالطبع البنوك تستخدم قواعد البيانات متعددة الأبعاد أيضا، تخيلوا مثلا لو قام بنك من البنوك بتصنيف عملائه حسب درجة الأهمية، طبعا درجة الأهمية تحدد حسب العائد للبنك من عملائه المختلفون، التصنيف السابق قد تقوم به أية قاعدة بيانات عادية، ولكن هذه قصة قد تحدث نتيجة لهذا التصنيف
تقوم طالبة جامعية بالاتصال بالبنك للاستفسار عن رصيدها، بالطبع ماذا تتوقع استفادة البنك من طالبة جامعية، فهي تقوم بتحرير بعض الشيكات أحيانا عندما تشتي بعض الكتب مثلا، عند الاتصال بالبنك سيجيب عليها احد الموظفين، تكتب اسمها في الكومبيوتر مقابله، سيجد بأنها مصنفه بين قائمة الزبائن القليلين الأهمية، سيساعدها موظف البنك بصورة عامة، لن يتعمق في مشاكلها، وسينهي المكالمة بسرعة، مما بالطبع قد يجعلها تشعر بأنها لم تحصل على المساعدة المطلوبة.

تخيلوا في صباح اليوم التالي يتصل احد أهم عملاء البنك بمدير البنك ليحتج له عن طريقة معاملة بنته و عدم مساعدتها بالصورة المطلوبة، هذا بالطبع شيء لا يجب أن يحدث على الإطلاق في مؤسسة كهذه تسعى إلى الحفاظ على زبائنها ولكن كيف لموظف الاستفسارات على الهاتف ليعرف من المتصلة به.

هنا يأتي دور قواعد البيانات المجسمة، والتي لا تحفظ البيانات عن عميل واحد فقط، بل تستطيع إيجاد علاقة بين العميل و أسرته التي لديها حسابات مختلفة في البنك نفسه، و من هنا يساعد ذلك البنك بتحسين طرق تصنيف زبائنه ، أو عملائه و ذلك بناءا على الأسرة نفسها.

هذه إحدى الطرق التي تستخدم بها قواعد البيانات المتعددة الأبعاد، بالطبع هي اشمل من ذلك بكثير و لكنني لا أريد أن أكمل الكتاب بالكتابة عن فوائد هذا النظام فقط، حيث انتم سوف تكتشفوا مميزاته لوحدكم عندما تقوموا بأول محاولة لتعلمه.

مكعبات البيانات

المكعبات هي مفتاح قواعد البيانات المتعددة الأبعاد و هي عبارة عن مجموعة من الجداول الثنائية الأبعاد المرتبطة ببعضها البعض بواسطة طرق خاصة لتكون مكعب متعدد الأبعاد، و تحفظ المكعبات البيانات كما هي، كما تقوم بتلخيص تلك البيانات إحصائيا أيضا وذلك لتسريع العمليات الإحصائية.

الباقي قيد الإعداد

الاستعلامات على المكعبات بواسطة شبكات تحليل البيانات

هناك طرق كثيرة للاستعلام على مكعبات البيانات، و هناك الكثير من البرامج المخصصة لذلك، تقدم مايكروسوفت عدة طرق و منها شبكات تحليل البيانات، أو كما يسميها البعض شبكات الحسابات، أو كما تسميها Microsoft ب Pivot Tables أو Pivot Charts، أما الترجمة للعربية فهي الترجمة الخاصة بي، و برجاء إن كانت هناك ترجمة أخرى إخباري بها.

شبكات تحليل البيانات هي شبكات تقوم بعرض تفاصيل مكعبات البيانات على شكل ثنائي الأبعاد، مع إعطائنا كامل الإمكانيات لتبديل طرق عرض البيانات لنراها من الجهة الناسبة و بالطريقة المناسبة.

لقد تم إدخالها أول مرة مع نظام Microsoft Excel في النسخة أل 97 على ما اعتقد و من ثم تطورت و تم أخيرا إعادة تصميمها للعمل مع Microsoft SQL Server و Microsoft Analysis Services في النسخة XP من Microsoft Office و تم دمجها مع Excel و Access و صفحات الويب Web Pages.

و تعتبر إحدى الطرق الأسهل استخداما لتحليل مكعبات البيانات، و بالطبع هناك الكثير من البرمجيات الأخرى الأكثر تعقيدا، ولكن جميعها تقريبا تستخدم نفس الطريقة لعرض و تحليل البيانات.

الشرح غير كافي، سيتم إضافة المزيد مع أمثلة في النسخ القادمة من الكتاب

لغة الاستعلام MDX (Multidimensional Expressions)

و هي لغة تستخدم لعمليات الاستعلام على مكعبات البيانات، و على غرار لغة SQL فلغة MDX قادرة على القيام بكل احتمالات الاستعلامات الممكنة على مكعبات البيانات.

تشبه لغة MDX لغة SQL كثيرا، ولقد طورت كذلك لتقليل الوقت الذي يحتاجه المبرمجون لتعلم اللغة الجديدة، بالطبع هناك أوامر و معاملات إضافية كثيرة إضافتها MDX على SQL، ومن أول تلك المعاملات أو الأوامر هي ON و ROWS و ON COLUMNS و التي يجب إضافتها على عبارة أل SQLECT عند اختيار البيانات، و هذا مثال على الصيغة العامة ل MDX

```
SELECT axis specification ON COLUMNS,
axis specification ON ROWS
```

```
FROM cube_name
WHERE slicer_specification
```

و يلي مثال على الصيغة العامة

```
SELECT {[Store Type].[Store Type].MEMBERS} ON COLUMNS,
{[Store].[Store State].MEMBERS} ON ROWS
FROM [Sales]
WHERE (Measures.[Sales Average])
```

سيجد كل من استخدم لغة SQL من قبل إن العبارة السابقة مألوفة قليلا، وهذا صحيح فهي تشبهها كثيرا، مع إنها تختلف بعض الشيء عنها.

و لأهمية لغة MDX خصصنا جزء من هذا الكتاب يقدم دورة سريعة عن هذه اللغة.

مستقبل قواعد البيانات

تتطور قواعد البيانات باستمرار ، فهناك فكرة جديدة تظهر في الأسواق كل ثلاثة شهور تقريبا ، وآخر ما ظهر حتى الآن هو قواعد البيانات المتعددة الأبعاد ، والتي تسمح لك بتخزين البيانات على شكل مكعبات.

وهناك الآن لغة الاستعلام الطبيعية باللغة الإنجليزية العادية ، فهي الآن تتطور لاستبدال لغة SQL فلماذا لا نسأل الكومبيوتر بلغة طبيعية عن شيء ما عوضا عن توجيه الاستفسار إليه بلغة SQL ، لماذا لا اطلب منه مثلا "ما هو عدد الموظفين لدينا" عوضا عن الاستعلام باستخدام عبارات SQL قد لا يفهمها إلا خبراء الكومبيوتر

كما هناك اتجاهات لتطوير نظام الملفات ، حيث تعتقد مايكروسوفت بأنها مع عام 2003 ستتمكن من دمج قواعد البيانات مع نظام الملفات في الجهاز العادي ، و بذلك سيمكنك بناء جداول جنبا إلى جنب مع الملفات العادية ، واستخدام لغة أل SQL ولغات أخرى للاستفسار من نظام الملفات عن الكثير من الأمور مباشرة.

على العموم اعتقد شخصيا بان العالم يتجه أكثر إلى الذكاء الاصطناعي ، وهذا يحتاج الكثير من العمل ، فكما تلاحظوا بدئت قواعد البيانات تفهم أكثر لغة الإنسان عن طريق لغات الاستعلام الجديدة والتي ستمكنها مستقبلا من التوحد من خادمات أل Web لتساعدك في إيجاد ما تبحث عنه بسهولة أكبر.

اعتقد بان عام 2010 سيمكنك من الاستفسار في Yahoo بالعبارة التالي "ابحث عن مقال في صحيفة صدرت عام 1990 يتحدث عن مشكلة قانونية حدثت بين فلان وفلان ، أعطني جميع المقالات لو سمحت" ، طلب كهذا لن يمكن القيام به أن لم يكون الكومبيوتر قد أدرك معنى البيانات ، وإدراك معني البيانات هو أحد أهم أجزاء الذكاء الاصطناعي ، وهو مستقبل قواعد البيانات.

أتذكر فريق ياباني في منتصف الثمانينات قال "أعطونا الخبراء و الكثير من المال لنجعل الكومبيوتر يفكر في غضون عامين" وبعد خمسة أعوام من التجارب والمحاولات فشلوا في إنجاز ذلك ، اعتقد من الصعب جعل الكومبيوتر يفكر ، ولكنه من الممكن جعله إدراك بعض معنى البيانات في قواعد البيانات في المستقبل القريب.

تكنولوجيا قواعد بيانات مايكروسوفت

تقدم شركة مايكروسوفت مجموعة كبيرة و متكاملة من نظم قواعد البيانات ، وذلك بدئاً من الأجهزة الكفية الصغيرة ، مرورا بالأجهزة المكتبية ، و نهايةً بالأجهزة الكبيرة ، و في هذا الكتاب لن أتطرق كثيراً إلى معظم التكنولوجيا ، ولكنني سأتطرق على ما له علاقة بكتابنا الآن.

تاريخ تطور منتجات قواعد بيانات مايكروسوفت

بدئت شركة مايكروسوفت تطور نظم قواعد البيانات منذ بداية التسعينات تقريباً ، حيث أصدرت النسخة الأولى من برنامجها الشهير Microsoft Access لنظام التشغيل DOS ، وبعد مدة قليلة اشترت الشركة المصنعة لنظام FoxPro ، و بدئت تطوره أيضا كقاعدة بيانات للمبرمجين المحترفين.

لقد كانت النسخة الأولى من Microsoft Access تعمل تحت بيئة DOS و كانت أشبه بقاعدة بيانات بسيطة مع نظام إدارة الاتصالات والمواعيد ، و بعض الأمور المكتبية الأخرى.

في عام 1992 شهر نوفمبر ، قامت مايكروسوفت بإصدار النسخة الأولى من Microsoft Access لبيئة Windows ، وذلك بعد إعادة تطوير البرنامج بالكامل ، لقد لقي النظام الجديد رواجاً كبيراً في الأسواق.

توقعت مايكروسوفت عند إصدار النسخة الأولى من Access بان تقوم ببيع ما لا يزيد عن مائة ألف نسخة من هذا البرنامج في غضون عام واحد ، ولكن التوقعات لم تكن دقيقة أبداً ، حيث تمكنت الشركة من بيع هذا العدد في أول أسبوعين فقط ، وحتى أن انتهي العام كانت قد تمكنت من بيع مليون نسخة.

لقد صممت Access بالأساس لتستخدم من مبرمجي قواعد البيانات ، ولكن كانت الدهشة كبيرة عندما علمت مايكروسوفت بان فقط أربعون بالمائة من إجمالي المستخدمين هما مبرمجين بالأصل ، أما الباقي فهم مستخدمي كومبيوتر عاديين فقط.

و في عام 1994 أصدرت الشركة ثاني نسخة من هذا النظام ، والذي لقي رواجاً كبيرا أيضا ، وفرت النسخة الثانية الكثير من المميزات ، فأصبحت اسهل في الاستخدام ، و قامت مايكروسوفت بتوفير تلك النسخة أيضا في الكثير من الدول ، وهي أول نسخة تم تعريبها من هذا النظام ، حيث مكنت تلك النسخة من صناعة برامج قادرة على العمل على شبكات الكومبيوتر الصغيرة ، والعمل مع قواعد بيانات كبيرة الحجم نسبياً ، و تعد الصلاحيات والمستخدمين ، والاتصال بقواعد البيانات الأخرى عن طريق ODBC و الكثير من المميزات الأخرى.

و يمكن اعتبارها أول برنامج وفر للمستخدمين واجهة سلسة للتعامل مع كم هائل من البيانات بدون الحاجة إلى البرمجة ، وبدون الحاجة إلى إتقان لغة SQL حيث أنها كانت تبنى استعلامات SQL الخاصة بها بصورة أتوماتيكية من الاستعلامات ، والذي يحدث حتى اليوم.

عام 1995 شهر سبتمبر ، وفرت الشركة النسخة Access 95 من نظامها الشهير والتي أصبحت قاعة بيانات من عيار 32 بت متكاملة ، واصبح بمقدورنا بناء ملفات قواعد بيانات بحجم يصل إلى اثنين جيجابايت ، وهو رقم هائل بالنسبة إلى قاعدة بيانات الشخصية ، أو الصغيرة الحجم.

و تتالت الإصدارات من ذلك النظام ففي شهر نوفمبر عام 1996 ظهرت النسخة Access 97 ومن ثم في صيف عام 1999 ظهرت النسخة Access 2000 و أخيرا النسخة Access 2002 أو كما تسمى أيضا Access XP في منتصف عام 2002.

بالطبع طوال الفترات السابقة طوت مايكروسوفت في منتجات قواعد البيانات الأخرى مثل FoxPro ، حيث أصدرت العديد من النسخ منه و من ثم نقلته إلى بيئة Windows و ضمته كجزء رئيسي من تكنولوجيا Visual Studio ، لتظهر آخر نسخة منه و هي Visual FoxPro 6.

ولكن مع ذلك فن تهتم به الشركة كما اهتمت بنظم قواعد بياناتها الأخرى مثل Access ، وهو لم يلاقي ذلك الرواج الكبير عند المبرمجين ، بالطبع هناك العديد من البرامج التي تمت بنائها بواسطته ، و قريباً ستنتهي مايكروسوفت من تطويره ضمن مجموعة Visual Studio حيث إن لغات البرمجة هناك أصبحت متطورة اكثر منه تقريباً.

لقد أضافت الشركة تكنولوجيا قواعد البيانات إلى لغات البرمجة الخاصة بها ، أصبحت لغات البرمجة قادرة عل القيام بالكثير من العمليات التي يمكن أن تقوم بها قاعدة البيانات ، إن لم يكن جميع العمليات ، وهكذا تم ضم نظام قواعد البيانات إلى Visual Basic و من ثم إلى Visual C++ وبعد أن أتت Java إلى السوق ، أصدرت الشركة نسخة خاصة منها تشمل قواعد بيانات Microsoft.

وانتقل دعم قواعد البيانات إلى الإنترنت ، حيث دعمت مايكروسوفت مزودات إنترنت الخاصة بها بإمكانية الوصول إلى قاعدة البيانات ، وكذلك ضمتها إلى برامجها المختلفة مثل FrontPage و InterDev ، و غيرهما من التكنولوجيا الصغيرة.

و من جهة أخرى قامت مايكروسوفت بالتركيز على بناء خادمات قواعد بيانات مركزية ، و أصدرت عام 1990 النسخة الأولى من نظامها الشهير Microsoft SQL Server لنظام التشغيل IBM OS/2 ، حيث كانت Microsoft و IBM تعملان جنباً إلى جنب لاستبدال نظام التشغيل Windows 3.x بنظام OS/2 في ذلك الوقت، ولكنه سرعان ما اختلفت الشركتان و انسحبت مايكروسوفت من مشروع OS/2 و أنتجت Windows NT .

لقد طورت Microsoft عدة نسخ مختلفة من خادم قواعد البيانات SQL Server للنظام OS/2 و لكنها سرعان ما نقلت النظام إلى Windows NT مباشرةً بعد إنتاجه ، و أصدرت النسخة الرابعة من SQL Server في ذلك الوقت.

و من ثم على فترات قصيرة أصدرت الشركة عدة نسخ من ذلك النظام و منها النسخ 4.1 و 6.0 و 6.5 ، حيث اعتمدت تلك انسخ في برمجتها بواسطة تكنولوجيا ODBC ، وهي تكنولوجيا الوصول إلى البيانات التي صنعتها مايكروسوفت.

ولكن تكنولوجيا ODBC احتاجت للكثير من الكود لاستخدامها بالصورة الصحيحة ، و مع أن SQL Server لقي رواجاً كبيراً في ذلك الوقت ، فهو كان غريباً بعض الشيء لمبرمجي Access والذين لم يعتادوا على كتابة الكود للوصول إلى البيانات.

و في نهاية عام 1997 قامت مايكروسوفت باستقطاب اثنين من مصممي نظم قواعد البيانات من كل شركة مشهورة في تصميم تلك النظم ، و سألتهم "لو أتاحت لكم الفرصة في إعادة بناء خادم قواعد بيانات مرة أخرى ، فماذا كنتم ستنتجون" ، لقد كانت تلك خطوة ذكية جداً ، حيث قام الفريق الجديد بإدارة فريق المبرمجين ، وتمت عملية إعادة بناء تكنولوجيا SQL Server جذرياً ، و ظهرت النسخة السابعة من ذلك النظام والتي امتازت بسرعتها العالية و تكاملها مع Microsoft Access 2000 و الكثير من المميزات الأخرى التي سنتحدث عنها لاحقا في هذا الكتاب.

و في عام 2000 ظهرت النسخة ألفين من SQL Server والتي قدمت تطوير إضافي للنسخة السابعة و أضافت الكثير من المميزات إليها ، حتى انه اصبح التعامل مع SQL Server 2000 من خلال Access XP اسهل من التعامل مع قواعد بيانات Access العادية.

طوال الفترات السابقة كانت مايكروسوفت أيضا تقوم بتطوير تكنولوجيا الوصول إلى البيانات ، فكما يعلم البعض لغات البرمجة مثلا لا تعتبر قواعد بيانات ، ولكنها تصل إلى قواعد البيانات عبر تكنولوجيا الوصول إلى ملفات قواعد البيانات.

وفي الكثير من الأحيان كانت مايكروسوفت تستبدل التكنولوجيا بصورة كاملة و مرة واحدة ، مما كان و لا يزال يثير غضب الكثير من المبرمجين ، حيث أن الكود الذي تكتبه في Access 1.0 لن يعمل في Access 2.0 بصورة طبيعة ، وهكذا مع لغات البرمجة الأخرى ، فهناك فرق واضح بين Visual Basic و ذلك بين الإصدارة الخامسة والإصدارة السادسة ، حيث تستخدم الأخيرة مفهوم مختلف جذريا للوصول إلى قاعدة البيانات.

تهدف الشركة من عملية التطوير هذه ، الوصول إلى افضل واسهل و أسرع طريقة للتعامل مع قواعد البيانات ، فقد بدئت بتوفير تكنولوجيا ODBC في أول مرة ، و من ثم طورتها و ظهرت تكنولوجيا DAO والتي سرعان ما طورتها مرة أخرى ، حتى إن الإصدارة الجديدة من تلك التكنولوجيا اختلفت طريقة التعامل معها بعض الشيء عن الإصدارة القديمة.

و لكن لم تدم تلك التكنولوجيا كثيراً ، مع أنها استخدمت من Access 97 و Visual Basic 5 ، و في اقل من عام ونصف انتهت عملية تطويرها ، و ظهرت تكنولوجيا ActiveX Data Object باختصار ADO والتي استقرت حتى اليوم.

و اليوم تقوم مايكروسوفت بتطوير نسخة محسنة من ADO بالاسم ADO+ والتي ستستخدم من Visual Studio .NET و من النسخ القادمة من Microsoft Access.

هدفت مايكروسوفت طوال فترة تطوير تكنولوجيا الوصول إلى قواعد البيانات ، هدفت إلى توفير وصول إلى جميع أنواع قواعد البيانات المشهورة بنفس الطريقة ، حتى تتمكن من خلال Visual Basic مثلا الاتصال مع Oracle أو SQL Server أو Access Database بنفس الطريقة ، ولتتمكن من تبادل البيانات بينهما بسهولة.

هناك أيضا الكثير من التكنولوجيا الأخرى والتي تأتى كجزء من التكنولوجيا المذكورة سابقاً ومنها تكنولوجيا JET و OLE DB وغيرها من التكنولوجيا.

يسألني الكثير من المبرمجين ، "هل ساهم كل ذلك التطوير في زيادة سرعة الوصول إلى البيانات؟" ، و جوابي هو نعم ، حيث أن SQL Server 2000 وصل إلى أقصى سرعة يمكن أن يصل إليها محرك قواعد بيانات في بيئة وندوس ، فلقد تمت عملية إعادة بناء خوارزميات حفظ و استرجاع البيانات بصورة متتالية لتصل إلى أقصى ما يمكن أن تصل إليه تلك الخوارزميات.

مقدمة إلى Microsoft SQL Server 2000

كما ذكرنا سابقاً SQL Server 2000 هو عبارة عن خادم قواعد بيانات مركزية ، وظيفته الأساسية إدارة البيانات و كل ما يتعلق بها فقط.

يعتقد الكثيرون بأنه لغة برمجة أيضا تمكنك من بناء النماذج والتقارير ، ولكنه ليس كذلك ، فهو عن صحيح يحتوي على لغة برمجة ، ولكنه لا يحتوي على أية أنظمة لإنشاء التقارير أو الشاشات ، مثله مثل كل قواعد البيانات المركزية الأخرى.

حيث تتوفر مجموعة من الأدوات لتكوين البرامج ضمن SQL Server ومنها Microsoft Access و Microsoft Visual Basic و Microsoft Visual J++ و Microsoft Visual C++ و Microsoft InterDev و Microsoft C#.

كل اللغات السابقة قادرة على الاتصال بخادم قاعدة البيانات و بناء البرامج لتعمل ضمنه ، ولكن أفضلهم على الإطلاق هو نظام Microsoft Access XP والذي يقدم العديد من المميزات التي لا تتوفر ضمنيا في لغات البرمجة الأخرى ، حيث انك بحاجة إلى كتابة الكود لإنشائها.

يتوفر نظام SQL Server 2000 في عدة نسخ وهي

- للأجهزة العادية SQL Server 2000 Personal
- للأجهزة المركزية SQL Server 2000
- للأجهزة المركزية المتوسطة والكبيرة SQL Server 2000 Enterprise
- للأجهزة الكفية SQL Server CE (Windows CE)
- للمبرمجين SQL Server 2000 Developer
- نسخة مصغرة للأجهزة العادية (المحرك فقط بدون شاشات الإدارة) MSDE

جميع النسخ السابقة من هذا النظام تقوم بالعمل نفسه تقريبا ، باثناء النسخة CE والتي هي مصغرة بدرجة كبيرة ، و مخصصة للأجهزة المحمولة بالكف ، تلك الأجهزة للذين لم يتعرفوا عليها سابقا عبارة على شاشة بشكل ورقة A5 أي نصف الورقة العادية وتعمل من خلال قلم ، و بها برامج Microsoft Office المصغرة مع Windows CE المصغر.

الاختلاف الوحيد بينها هو أنها خصصت لتركب على أجهزة مختلفة و تستهلك موارد مختلفة من موارد النظام ، ولكنها جميعا تحتوي على كل المميزات الأساسية مع اختلافات بسيطة للغاية في بعض الأحيان.

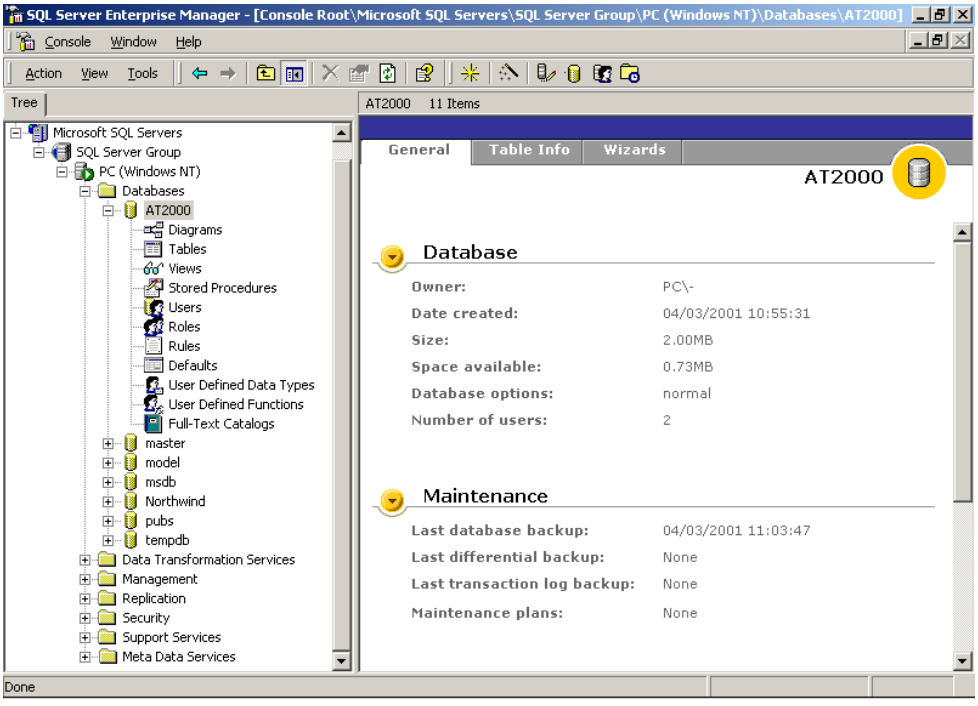
تتم عملية إدارة SQL Server 2000 عن طريق برنامج رئيسي قادر على القيام بكل عمليات الإدارة ، مع وجود بعض البرامج الأخرى ولكنها مرتبطة ضمن البرنامج الرئيسي.

و هذا ما يميز SQL Server عن بقية نظم قواعد البيانات المركزية الأخرى ، حتى أن الكثير من النظم الأخرى بدئت تصنع برامج خاصة لإدارة قواعد البيانات الخاصة بها ، و ذلك بعد أن لقيت فكرة الإدارة المركزية رواجاً كبيراً ، حتى انه في العروض عندما كانت تقدم شركة مايكروسوفت نظام SQL Server كانت في الكثير من الأحيان تأتي بأطفال و طلاب مدارس ليقوموا بإدارة الجهاز المركزي والقيام ببعض العمليات عليه مثل عملية نسخ قاعدة البيانات مثلاً ، وذلك كدعاية لسهولة الاستخدام لهذا النظام.

لوحة الإدارة الرئيسية تسمى SQL Server Enterprise Manager وهي مصممة على شكل مستكشف أل Windows (مدير الملفات كما يسميه البعض)، على شكل شجري بحيث تفصل لك كل تفاصيل الجهاز المركزي بصورة متكاملة، كما يمكننا من خلالها ربط مجموعة اكبر من الأجهزة المركزية و إدارتها معا، و هي مخصصة للعمل مع الشبكات البطيئة و الإنترنت، حيث يمكنك بواسطتها إدارة أجهزة في أماكن مختلفة من العالم.

تحتوي لوحة الإدارة على مجموعة كبيرة من المعالجات، و المخصصة للقيام بكل العمليات التي يمكننا أن نطلبها من الجهاز المركزي، وتشمل تلك العمليات، عمليات مثل النسخ الاحتياطي، استرجاع البيانات، إصلاح قواعد البيانات، بناء قواعد بيانات جديدة، التعامل مع الصلاحيات والمستخدمين، تصدير البيانات و نقلها إلى قواعد بيانات أخرى، و الكثير من الأمور الأخرى.

يعبر الشكل التالي صورة للوحة التحكم بقاعدة البيانات



الصورة رقم 1: (لوحة التحكم بقاعدة البيانات)

حيث كما ترون في الصورة السابقة، تبين الأقسام الرئيسية للجهاز المركزي، و في قسم قواعد البيانات، تظهر جميع قواعد البيانات بشكل مرتب و منطقي، وتظهر في كل قاعدة بيانات محتوياتها، وهكذا.

نظام SQL Server قادر على التعامل مع قواعد بيانات يصل حجمها أكثر بقليل من مليون تيرابايت، والتيرابايت عبارة عن ألف جيجابايت، و هو رقم يفوق معظم مساحات التخزين المتوفرة اليوم.
و هو قادر على تخزين حتى اثنين جيجابايت في كل سطر من اسطر الجداول وذلك على شكل صور أو ملفات ، أو بيانات ضخمة.
و يمكن للنسخة أل Enterprise الخاصة منه أن تعمل مع اثنين وثلاثين معالج مرتبطين مع بعضهم البعض، لتحصل على سرعة سبعون بالمائة من حاصل مجموع سرعة جميع المعالجات ، أي إن وضعت 32 معالج Pentium 4 في جهاز مركزي فقد تحصل على سرعة تقترب من الثلاثين ألف ميجاهيرتز في التعامل مع البيانات.
كما يمكن تركيب اكثر بقليل من اثنين وثلاثين ألف نسخة من البرنامج على نفس الجهاز المركزي معاً و ذلك إن تطلب الأمر ، ولا توجد هناك حدود قصوى لعدد السجلات في الجداول ، فهي محدودة بمساحة التخزين المتوفرة ، أما بالنسبة للذاكرة فيمكن للنسخة أل Enterprise أن تتعامل مع 64 جيجابايت من الذاكرة RAM في الجهاز.

ويمكننا ربط عدد كبير من الأجهزة الخادمة معاً و التي تحتوي بدورها على العديد من المعالجات وذلك لتكوين Super Computer يعمل عل تكنولوجيا Windows و SQL Server 2000.

أنا صراحة في بعض الأحيان استغرب من التعبير الشائع وهو "SQL Server هو نظام مخصص لقواعد البيانات الصغيرة" ، إن كانت الأرقام السابقة تعتبر صغيرة ، فماذا يقصدوا بقواعد البيانات الكبيرة ☺ (مزح فقط).

بالمناسبة ، إن كنت تقرا هذا الكتاب بعد خمسة أعوام ، أو صدف وان اطلعت عليه في المستقبل فلا تضحك علي ، أنا اعلم بأنه بعد أعوام الأرقام السابقة ستكون صغيرة للغاية ، حتى انه قد لا ترضى بها في جهازك العادي ☺.

هذه هي مقدمة بسيطة عن SQL Server 2000 ، وسيلي شرحه الآن بالتفصيل في الصفحات اللاحقة للكتاب.

مقدمة إلى Microsoft Access 2002 (XP)

لقد ظل برنامج Microsoft Access وطوال السنوات السابقة ومنذ إصداره الأول ، رائدا في مجال قواعد البيانات الشخصية ، حيث انه قدم طريقة جديدة ومثالية في بناء قواعد البيانات و إدارتها ، ففي الوقت الذي كان يعتمد فيه المبرمجون على كتابة جمل ألSQL يدويا للقيام بأبسط العمليات، أتى ذلك البرنامج الرائع وقدم الكثير من الحلول التي اختصرت وقت بناء التطبيقات إلى الحد الأقصى ، وقدم واجه سهلة حتى للمستخدم العادي ليتمكن بنفسه من بناء برامجه الخاصة.

ومنذ تلك اللحظة وحتى اليوم وهو ينتشر في معظم الشركات الصغيرة والكبيرة ، حتى انه أصبح أحد الأشياء المهمة والتي يجب معرفة التعامل معها قبل الحصول على وظيفة ، ففي المؤسسة التي اعمل بها من الصعب أن توظف سكرتيرة مثلاً وهي لا تعرف كيفية استخدام Access.

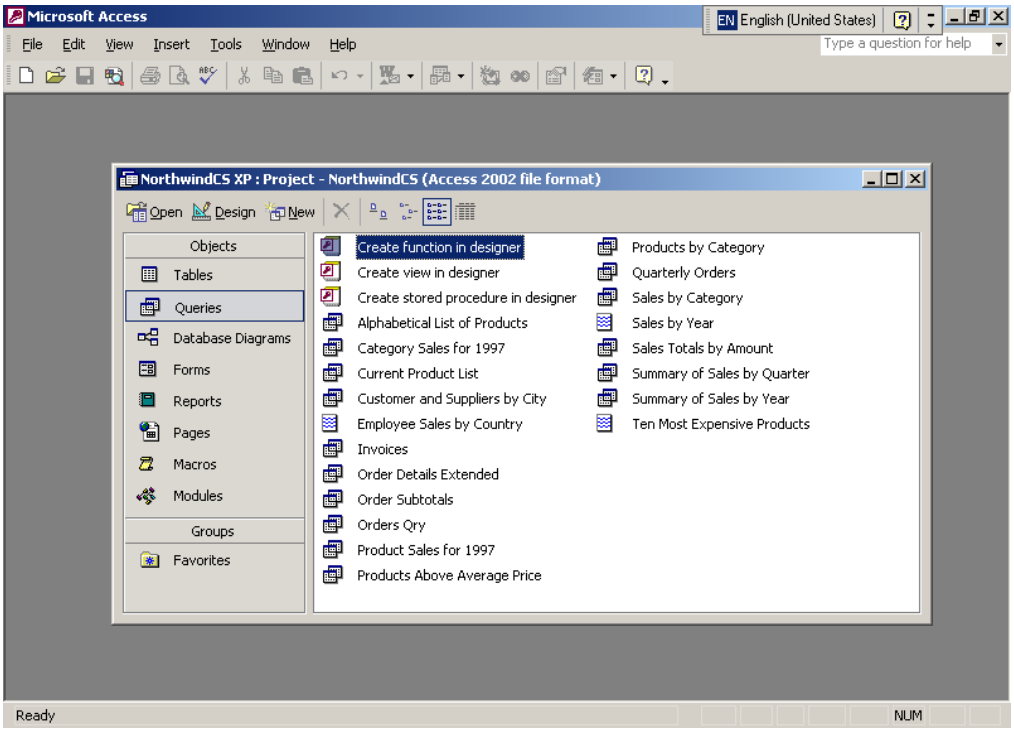
انتشر ذلك النظام طوال السنوات السابقة بسرعة فائقة، ونظرا لسهولة استخدامه لم يهتم به الكثير من المبرمجون بصورة جدية في بادئ الأمر ، وخصوصا لصعوبة تأقلمهما مع وندوس الذي كان في بدايته في تلك المرحلة.

يمكن أن أجمل بعد سبعة سنوات من استخدام هذا النظام بأنه عبارة عن بيئة تصميم للبرامج، ولغة برمجة، وبيئة تشغيل للبرامج مندمجين معاً ككتلة واحدة، وهذا الاندماج هو سر نجاح Access.

يختلف Access عن لغات البرمجة الأخرى، حيث يستصعب المبرمجون عملية الانتقال إليه، وذلك لأنهم اعتادوا على بناء برامجهم ضمن مواصفات و مقاييس خاصة بهم، والتي تختلف في الكثير من الأحيان بالمواصفات والمقاييس الأساسية التي بنيت على أساسها قواعد البيانات.

يوفر نظام Access جميع الأدوات التي تلزم إنشاء برامج قواعد البيانات، فالنماذج الخاصة به قادرة على التعامل مع البيانات بصورة مباشرة، قادرة على البحث بداخل البيانات، و فرزها، ترتيبها تصاعديا وتنازليا ، عرضها على شكل جداول ، شاشات ، شبكات اتخاذ القرار ، مخططات بيانية ذكية (المميزتين الجديدتين توفرتا ضمن النسخة الأخيرة من Access).

يعبر الشكل التالي صورة لقاعدة بيانات في Access XP مرتبطة مع SQL Server 2000



الصورة رقم 2: (قاعدة بيانات في Access XP مرتبطة ب SQL Server 2000)

و هكذا بمجرد بدئك ببناء مشروعك ضمن Access فأنت لا تهتم بالبرمجة تقريبا، كل ما يهمك هو بناء البرنامج و الدخول في تفاصيله الدقيقة.

يوفر نظم Access كل الأدوات التي تلزمك لبناء تطبيقات قواعد البيانات، أدوات مثل إنشاء الجداول، إنشاء الاستعلامات و نص أل SQL بناء الشاشات (النماذج)، والتقارير، كود للتحكم في البرنامج، و شاشة مخصصة لبناء العلاقات بين الجداول.

و يمكن لذلك النظام بان يخزن البيانات بداخل ملفات قواعد البيانات الخاصة به ، أو الاندماج مع SQL Server و تخزين البيانات بداخله والاستفادة من جميع مميزاته الأخرى ، وهو الواجهة الأفضل لبرمجة ذلك النظام.

بمجرد بنائك برنامج بواسطة Access فيمكنك وضع اسم خاص لبرنامجك حيث يستبدل ذلك الاسم اسم Access ويمكنك وضع شعاراتك الخاصة لتظهر بدل شعار Access في بداية التشغيل ، كما يمكنك وضع قوائم و أشرطة أدوات خاصة بك تستبدل بها التي تأتى مع النسخة الأصلية ، كما يمكنك إخفاء أية جزء لا يعجبك في النظام.

و هكذا فأنت تحصل على برنامجك الخاص و الذي يمكنك حمايته و من ثم توزيعه مجانا (مايكروسوفت لن تتقاضى اجر مقابل عملية توزيعه مثلما تفعل بعض الشركات الأخرى) ، حيث أن هناك نسخة مخصصة من Access لتشغيل البرامج فقط وهي مجانية.

عملية حماية برامج Access عملية لا يفهمها الكثيرون في معظم الأحيان، حيث يعتقد البعض بان الحماية هي منع المستخدم من الوصول إلى أية جزء للبرنامج، و منعه من الوصول المباشر إلى البيانات و وضع كلمة سر على قاعدة البيانات، ولكن الحماية مختلفة قليلاً.

كما ذكرنا سابقاً في أول الكتاب هناك أسس لقواعد البيانات ، وأحد الأسس هو إمكانية الوصول إلى البيانات والتعامل معها بصورة مباشرة ، فان لم يتوفر ذلك الشرط أصبحت Access قاعدة بيانات منقوصة ، أو حتى إننا لا يمكننا تصنيفها كقاعدة بيانات.

لذلك حتى وبعد عملية تحويل برنامجنا إلى هيئة MDE و هي الهيئة المخصصة لتوزيع البرامج كما سنرى لاحقاً ، حتى وإن تم ذلك فسيكون بإمكاننا دائما الوصول إلى جداول قاعدة البيانات وتعديلها و بناء الاستعلامات ، و وحدات الماكرو ، و تعديل القوائم و أشرطة الأدوات.

ولكننا لا يمكن أن نعدل على الكود الخاص بالبرنامج و بشاشاته و تقاريره المختلفة ، بالطبع يمكننا باستخدام قاعدة بيانات أخرى نربطها مع قاعدة البيانات الأساسية ، يمكننا بذلك بناء شاشاتنا و تقاريرنا الإضافية.

هذه هي مقدمة بسيطة ل Access XP ، وسيلي شرحه الآن بالتفصيل في الصفحات اللاحقة للكتاب.

إعداد نظام Microsoft SQL Server 2000 لأول مرة

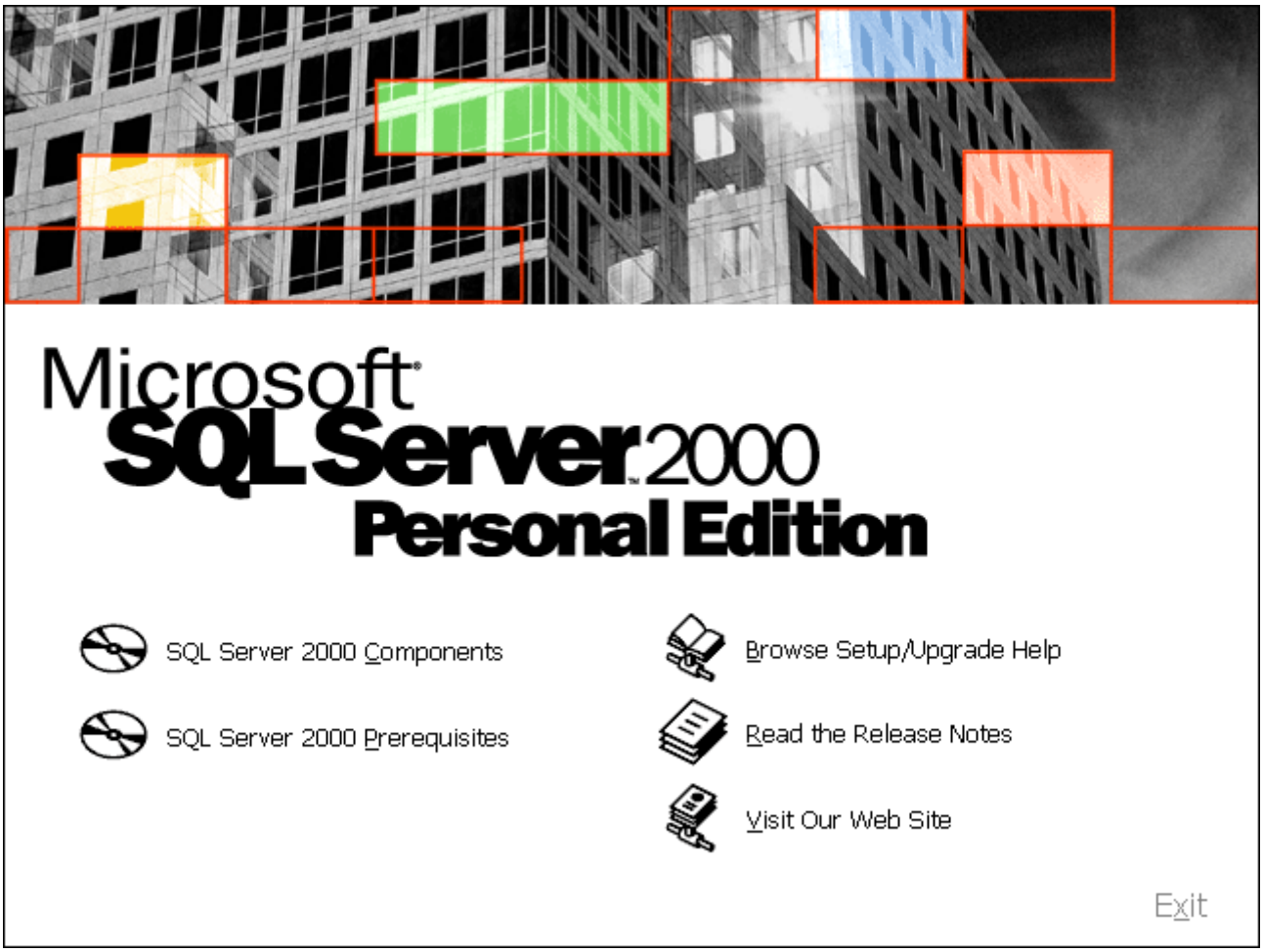
عملية إعداد نظام SQL server بسيطة للغاية ، يكفيك أن تشغل برنامج الإعداد وتغمض عينيك و تضغط على زر أل enter لتجاوب بنعم على كل الأسئلة التي سيطرحها لك النظام.

مع أن النظام سيعمل بصورة طبيعية ، إلا أنني لا أحب هذه الطريقة كثيرا ، لذلك سأحاول أن اشرح قدر الإمكان عن عملية تركيب النظام ، و احتمالات التركيب البديلة الممكن اتباعها.

كما أن عملية التركيب هي نفسها لكل النسخ المتوفرة من هذا النظام تقريبا ، باستثناء النسخة المصغرة جدا ل Windows CE.

قم بوضع القرص الليزر في الجهاز الذي ترغب بان تركب النظام به (استخدم النسخة المخصصة لنظام التشغيل الخاص بها ، حيث مثلا لا يمكنك تركيب النسخة Enterprise على Windows 98 مثلا ، فهي مخصصة للأجهزة العملاقة)

ستظهر لك شاشة التركيب الأولى والتي ستفتح بصورة أتوماتيكية



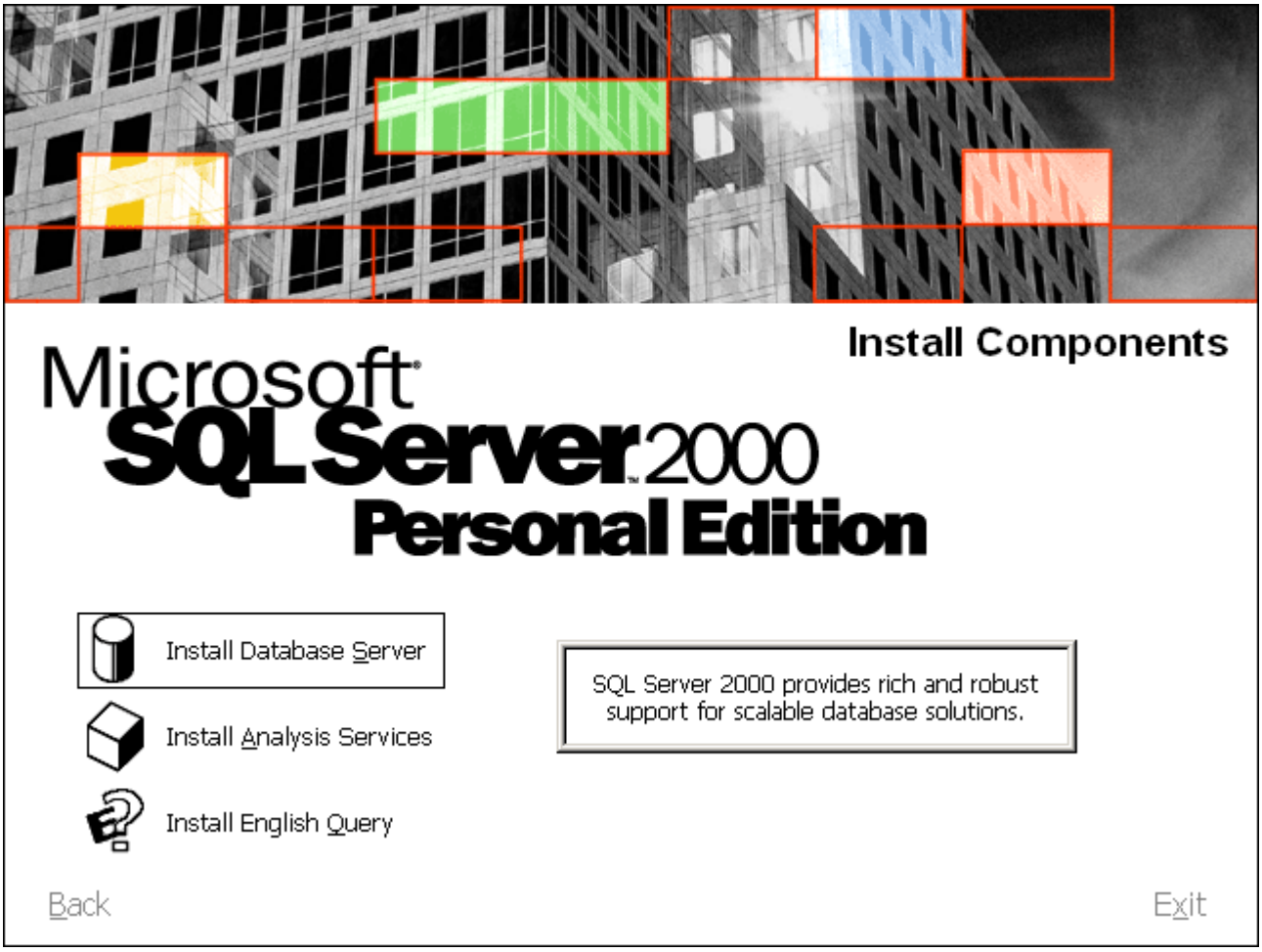
شاشة التركيب رقم (1)

الشاشة رقم واحد مخصصة لتزويدك بالمعلومات المهمة قبل بدء عملية التركيب ، وهي مخصصة ايضاً لتجهيز جهازك لعملية التركيب ، ولتتأكد من أن جهازك جاهز لاستقبال SQL Server يمكنك النقر على Prerequisites حيث يفترض أن تظهر قائمة بالأشياء المطلوبة.

لن أتطرق إلى تلك الشاشة ، حيث أننا لسنا بحاجة إليها في Windows 2000 و ما بعده ، ولكنك بحاجة إليها إن استخدمت Windows 98 مثلاً.

قم بالضغط على أل SQL Server 2000 Components لنبدأ بعملية تركيب النظام

حيث ستظهر الشاشة رقم (2)



شاشة التركيب رقم (2)

تعطيك تلك الشاشة الإمكانية لتركب أجزاء SQL Server المختلفة ، وينقسم هذا النظام إلى ثلاثة أقسام ، الأول هو خادم قاعدة البيانات ، و هو أساس قاعدة البيانات ، والثاني هو قواعد البيانات المتعددة الأبعاد ، ولا يمكن أن يعمل بدون الجزء الأول ، وكذلك أيضا بالنسبة للجزء الثالث و هو الاستعلام باللغة الإنجليزية.

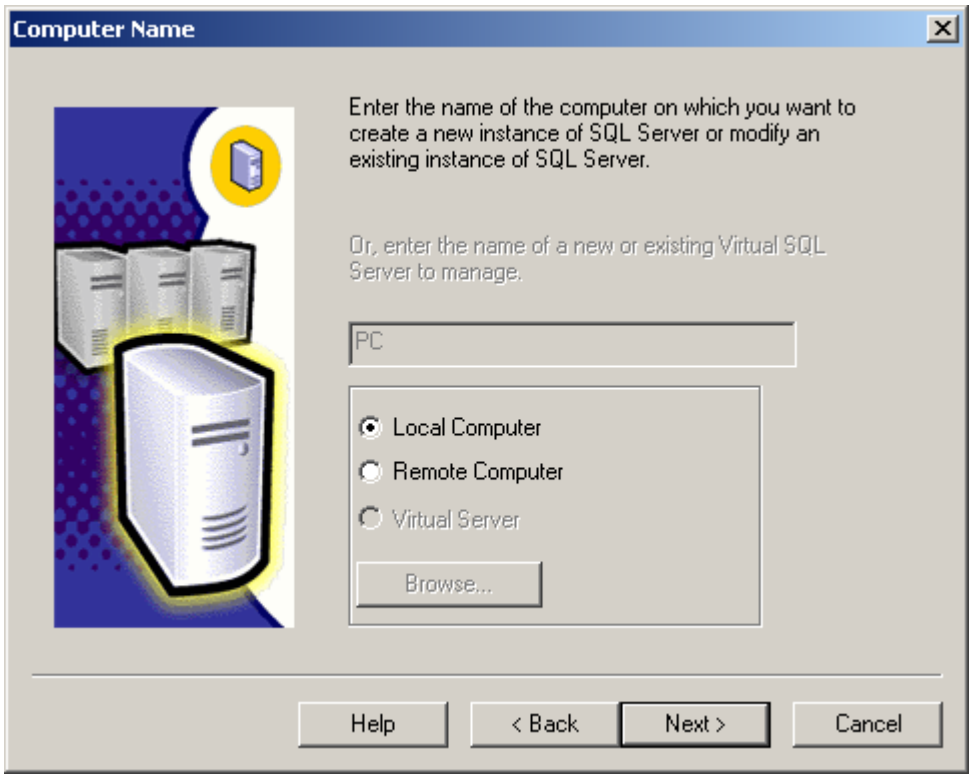
يجب دائماً البدء بتركيب محرك قاعدة البيانات ، ومن ثم يمكننا أن نركب الأجزاء الأخرى أن رغبنا بذلك.

ستظهر الشاشة رقم (3) عند الضغط على Install SQL Server لتبدأ عملية التركيب.

شاشة التركيب رقم (3)

و تحتوي بمعلومات بسيطة هدفها إعلامك ببدء عملية التركيب ، يتم الانتقال بين الشاشات المختلفة بالزرين Next و Back ، كما يمكنك إيقاف عملية التركيب في أية لحظة بالضغط على الزر Cancel.

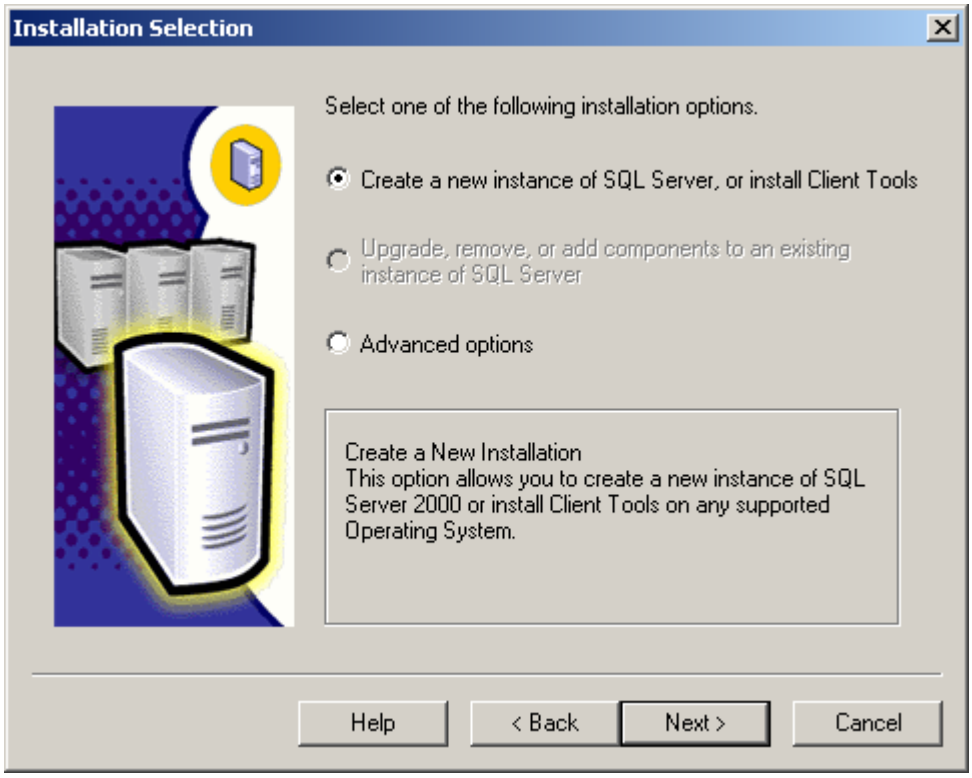
لننتقل إلى الشاشة التالية بالضغط على الزر Next لتظهر شاشة التركيب رقم (4)



شاشة التركيب رقم (4)

أحد المميزات المهمة في SQL Server هو انه بإمكاننا أن نركبه عن طريق جهازنا إلى جهاز آخر ، أو إلى مجموعة من الأجهزة المركزية معاً ، وهذه إحدى طرق التركيب الجديدة التي توفرها مايكروسوفت لتسريع عملية بناء الشبكات.

في حالتنا هذه سنختار التركيب على الجهاز المحلي Local ومن ثم ننتقل إلى الشاشة رقم (5) بالزر Next.



شاشة التركيب رقم (5)

شاشة التركيب رقم (5) تعطينا خيارين مهمين للتركيب ، الخيار الأول هو تركيب النظام على جهاز الكومبيوتر لدينا ، و إنشاء SQL Server Instance و الخيار الثاني يشمل إمكانيات متقدمة ومنها بناء ملفات تركيب لتستخدم في عملية التركيب المرة القادمة.

الخيار الثاني يقوم بطريقتين للتركيب ، الطريقة الأولى هي مثل عملية التركيب على جهاز آخر التي رأيناها في الشاشة السابقة ، ولكن الاختلاف هنا بأننا ننشئ نظام تركيب جاهز ، حيث نقوم بالإجابة على جميع أسئلة التركيب أول مرة ومن ثم ينشئ البرنامج برنامج تركيب جديد لا يسألنا أية أسئلة ، ويمكننا وضعه على قرص ليزر مثلا أو استخدامه من خلال الشبكة لنقوم بإعداد جميع الأجهزة المركزية لدينا.

و الطريقة الثانية في الخيار الثاني هو تركيب عنقود SQL Server وهي طريقة متقدمة جداً لبناء أجهزة أل Super Computers حيث يمكننا أن نركب عدد ضخم من العناقيد على أجهزة مختلفة ، لتعمل كلها و كأنها جهاز واحد تقريباً.

قد أقوم بشرح الطريقة الثانية مستقبلاً ، حيث إنها بحاجة إلى طرق برمجة خاصة لتنجح ، كما أنها بحاجة إلى بناء هيكلية قاعدة البيانات بطرق خاصة أيضا ، وذلك لضمان الاستفادة القصوى من العناقيد المترابطة.

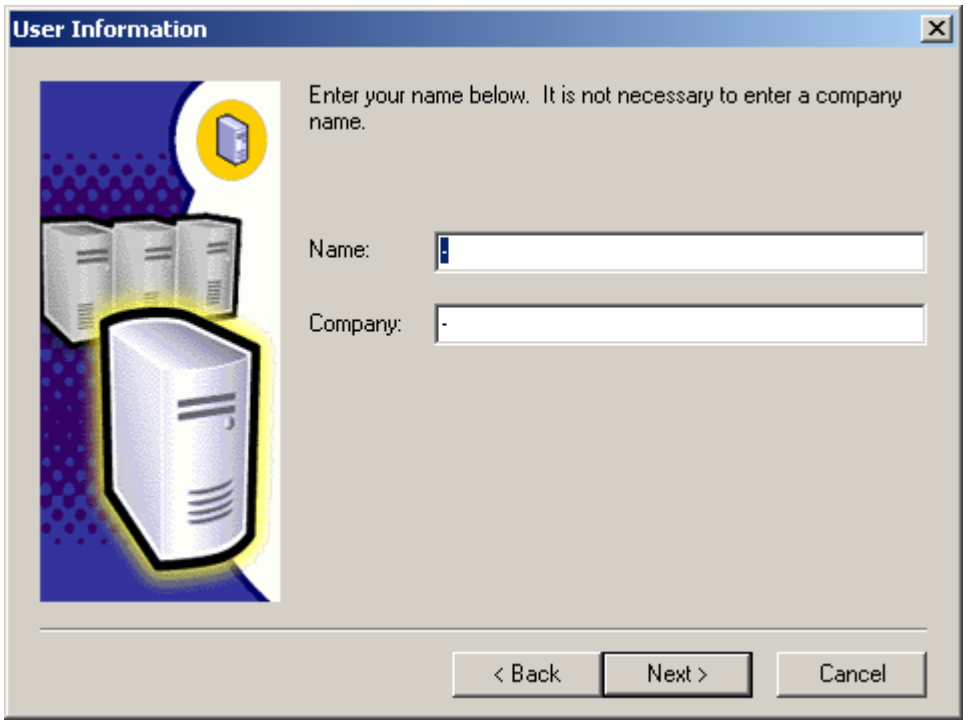
الطريقة الأولي في الخيار الثاني مهمة جداً للشركات التي تقوم بإعداد الكثير من خادمات قواعد البيانات ، فقد تحتاج أحيانا أن تركب البرنامج على ما يقارب مائة جهاز معاً ، وعملية أن تجلس أمام الكومبيوتر وان تضيع الوقت مجيباً على جميع تلك الأسئلة ليست بالسهلة.

أل SQL Server Instance هو عبارة عن تركيب نسخة من نسخ أل SQL Server على الجهاز ، حيث يمكن للجهاز استيعاب اكثر من نسخة من هذا النظام ، وبذلك ستتصرف كل نسخة و كأنها جهاز مركزي لوحدها.

هذه العملية جيدة إن كنت ترغب بتأجير خادم أل SQL server عبر إنترنت ، حيث يمكنك على جهاز مركزي واحد تركيب عشرة نسخ من SQL Server و تأجير كل منها على أنها نسخة متكاملة من هذا النظام.

كما يمكن الاستفادة من الطريقة السابقة في الكثير من الحالات الأخرى ، الحد الأقصى للنسخ الممكن أن نركبها من SQL Server على جهاز واحد يزيد قليلا عن اثنين و ثلاثين ألف نسخة ، حيث أن جميعها قادرة على العمل معاً ، وهذا رقم كبير للغاية.

لنختار تركيب نسخة من SQL Server ومن ثم ننتقل إلى الشاشة رقم (6) بالضغط على الزر Next



شاشة التركيب رقم (6)

و هي شاشة مختصة لتسجل اسم صاحب المنتج والشركة صاحبة المنتج ، وهي شاشة قانونية ، وذلك أن كنت تستخدم النسخ الأصلية من البرامج ، وكنت تخاف أيضا من إمكانية أن تفحص الشرطة الأجهزة في مؤسستك.

و بما أننا دخلنا في موضوع التراخيص ، فهناك شاشة لن تظهر لك إلا في حال تركيب SQL Server على الجهاز المركزي ، و هي الحد الأقصى من المستخدمين المرخصين لاستخدام البرنامج.

وهناك نوعين من التراخيص ، التراخيص على حد أقصى من المستخدمين لكل جهاز مركزي ، و التراخيص للمستخدم نفسه بدون ترخيص الجهاز المركزي.

الحالة الأولى تستخدم في الشركات الصغيرة والتي ستستخدم جهاز مركزي واحد فقط ، حيث يمكننا على سبيل المثال شراء ترخيص استخدام للجهاز المركزي من عشرون مستخدم مثلا ، وبذلك قد نكون قد وفرنا الكثير من المال على المؤسسة.

الحالة الثانية شراء ترخيص لكل مستخدم ، و تستخدم تلك الحالة إن كان لدينا مائة موظف مثلا وعشرة أجهزة مركزية تحمل SQL Server ، حث لا يعقل بان اشتري مائة ترخيص استخدام لكل جهاز مركزي ، لذلك يمكنني اشتري ترخيص استخدام SQL Server عبر الشبكة للمستخدم نفسه ، وبذلك يمكن للمستخدم استخدام أية جهاز SQL Server في العالم يريد استخدامه.

بالنسبة للطرق السابقة فتذكر بأنه لا يمكننا مزج طريقتي الترخيص معاً على نفس الجهاز ، كما انه في الحالة الأولى أن كان لدينا عشرون مستخدم ، و علمنا بأنه لن يستخدم الجهاز المركزي اكثر من عشرة مستخدمين

في نفس الوقت فيمكننا شراء عشرة تراخيص استخدام مثلا ، أما في الحالة الثانية فنحن بحاجة إلى ترخيص للمستخدم نفسه ولا يمكننا نقل الترخيص من جهاز إلى آخر.

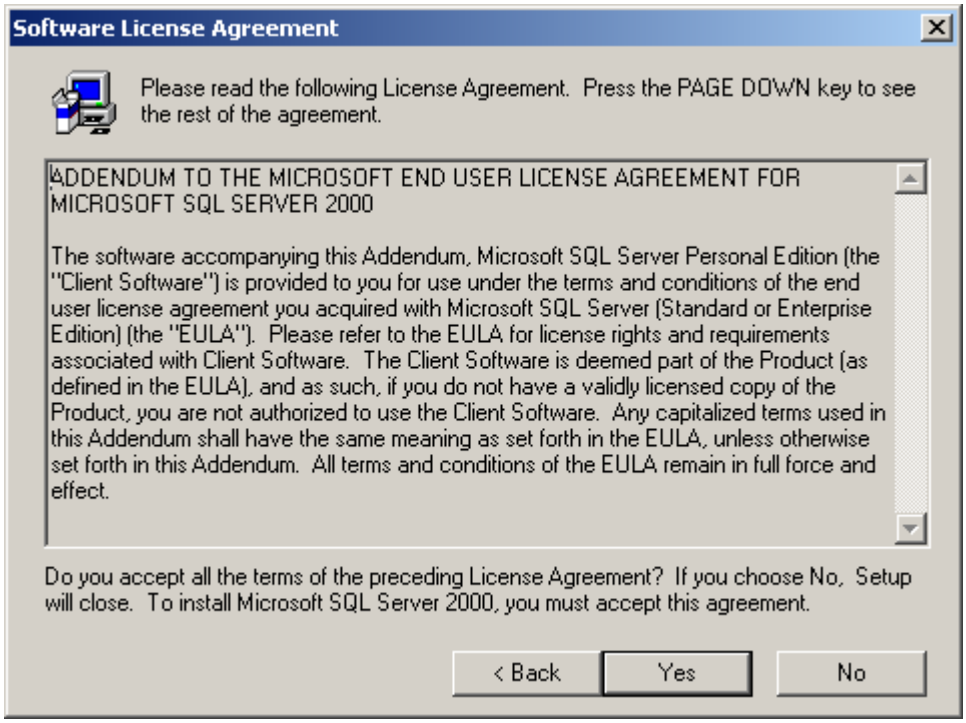
تذكر أيضاً بأنه إن حددت عدد المستخدمين الأقصى بعشرة مستخدمين مثلا فسيرفض الجهاز المركزي اتصال من المستخدم الحادي عشر ، وذلك حتى ينفصل أحدهم من الجهاز المركزي.

كما عن عملية الترخيص هي عملية قانونية ، بالطبع قد تضع ألف مستخدم لحظة التركيب ولكن في حال إن أتت الشرطة على شركتك وفحصت الجهاز ووجدت بأنك ركبته ليستخدم من ألف مستخدم ، فتوقع مخالفة ضخمة للغاية ، حتى ولو إن عدد المستخدمين لن يتجاوز العشرة مثلا.

هناك أيضا اتفاقيات ترخيص للاستخدام عبر إنترنت و هي اتفاقية ترخيص عدد غير محدود من المستخدمين ، وتقاس حسب سرعة المعالج و عدد المعالجات في الجهاز الواحد.

يمكنك دائماً زيادة رخص الاستخدام لديك من لوحة التحكم في الجهاز المركزي

لننتقل إلى شاشة التركيب رقم (7) بالضغط على الزر Next



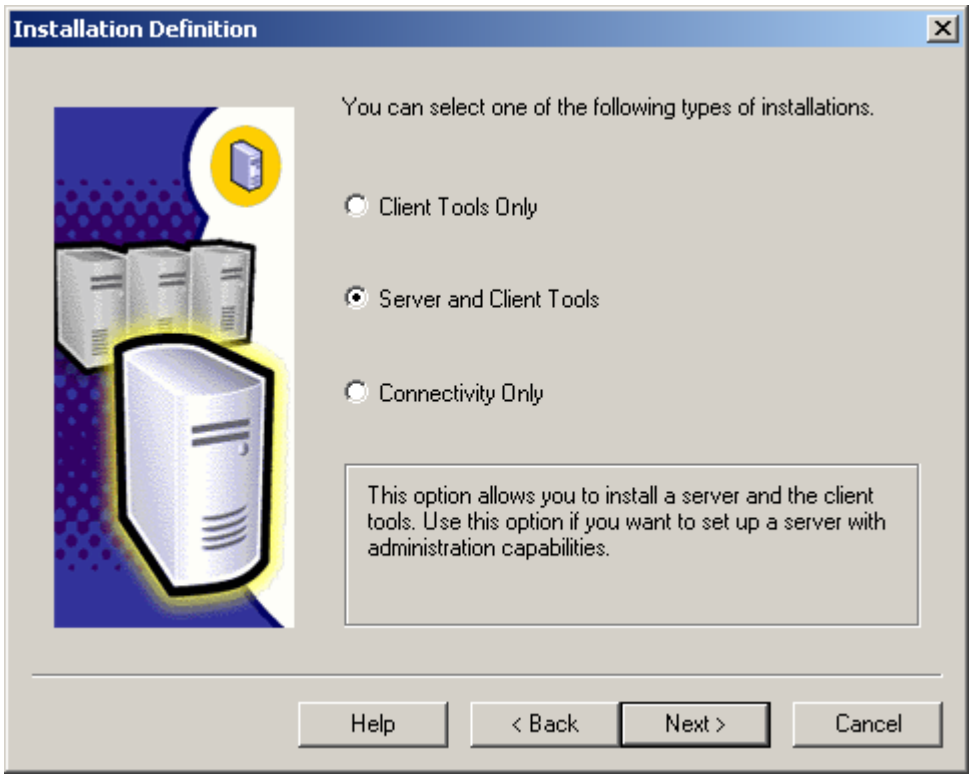
شاشة التركيب رقم (7)

و هي اتفاقية الترخيص ، وهي ملزمة قانونياً ، اقرأها بعناية قبل الضغط على الزر YES والذي يعتبرك موافق لاتفاقية الترخيص وبنودها.

في الوطن العربي قليلا ما يتهم الأفراد في اتفاقيات الترخيص وهذه مشكلة ، حيث انه في بضع نسخ برنامج أل ICQ وهو برنامج حوار شهير ، تقول اتفاقية الترخيص في ICQ بان المحاكم الإسرائيلية هي المخولة بالفصل في أية نزاع قد يحدث بسبب البرنامج.

لا تخافوا ذلك ليس موجود في منتجات مايكروسوفت ، ولكن تأكدوا دائما من قراءة اتفاقيات الترخيص.

إن كنت موافق على الاتفاقية فاختر الزر YES ولننتقل إلى الشاشة رقم (8)



شاشة التركيب رقم (8)

تعطيك الشاشة رقم (8) ثلاثة إمكانيات للتركيب ، وهي أحد شاشات المهمة ، وخيارات التركيب هي:

- Client Tools Only

و تعني تركيب الأدوات الخاصة بالتحكم بخادم قواعد البيانات فقط ، حيث يتم تركيب البرمجيات الصغيرة المختصة للاتصال مثل تكنولوجيا ADO و غيرها ، ويتم تركيب لوحة إدارة الجهاز المركزي ، أمثلة برمجية ، وبعض الأمور الأخرى ، ولكنه لا يركب محرك قواعد البيانات ، يتم تركيب فقط برمجيات التحكم و ما له علاقة بها.
كما تعلم فانه من الطبيعي في الحالات العادية أن يتم تركيب SQL Server على جهاز مركزي ، ومن الطبيعي أن لا يستخدم المبرمجين الجهاز المركزي بل أجهزتهم الشخصية ، و باختيار هذا الخيار سيحصل كل مبرمج على إمكانية تحكم و إدارة الجهاز المركزي من جهازه الشخصي ، وذلك كأنه يستخدم الجهاز المركزي بالضبط (بالطبع إن كانت له الصلاحيات لذلك).

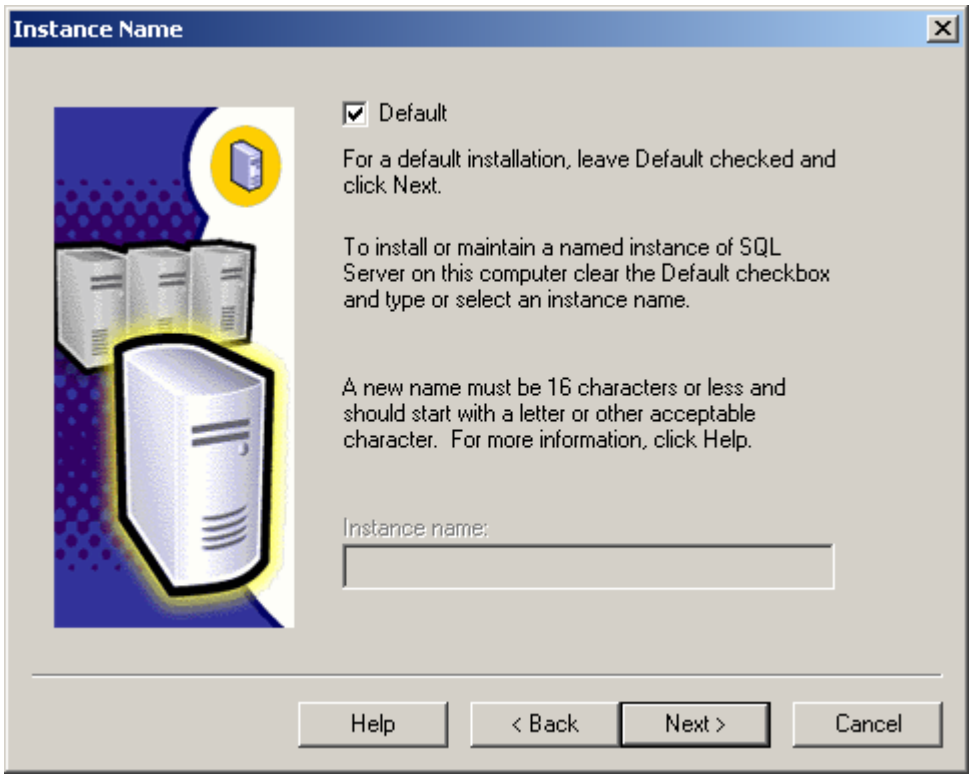
- Server and Client Tools

يركب هذا الخيار جميع محتويات الخيار السابق من برمجيات تحكم ، كما يركب أيضاً محرك قاعدة البيانات ، و بهذه فهو يركب أل SQL Server ليعمل على الجهاز المركزي ، أو على جهازنا في هذه الحالة ، حيث إننا سنقوم باختيار هذا الخيار.

- Connectivity Only

و هو تركيب البرمجيات المتخصصة لإنشاء اتصال مع محرك قواعد البيانات فقط ، مثل تكنولوجيا ADO و ODBC و غيرها.
حيث انه عندما تقوم بإنشاء برامجك الخاصة و قمت بتركيبها على الأجهزة يدويا (بدون عملية الإعداد الآلي Setup) فسيلزمك بان تركب برمجيات الاتصال مع الجهاز المركزي.
مع العلم بان تلك البرمجيات تأتى ضمنيا مع Access XP ولذلك لن نحتاجها ، ولكن قد تستخدم لغة برمجة تصنع برامج لا تأتى ضمنياً معها تلك البرمجيات.
لن نحتاج هذا الخيار تقريبا لأنه هناك ملف تنفيذي متوفر منه يمكن دمجه بداخل برامج الإعداد للبرامج التي لا تحتوي على برمجيات الاتصال ، لينفذ أتوماتيكيا لحظة تركيب تلك البرمجيات.

اختار الخيار الثاني و لننتقل إلى الشاشة رقم (9) بالضغط على الزر Next.



شاشة التركيب رقم (8)

و هي مخصصة للفصل بين نسخ أل SQL Server المختلفة التي ترغب بتركيبها في الجهاز ، حيث انه يمكننا كما ذكرنا سابقاً تركيب مجموعة كبيرة من النسخ من ذلك النظام لتعمل بصورة مستقلة.

يميز كل نسخة من نسخ النظام التي قد نركبها ، يميزها اسم النسخة و رقم البوابة الفرعية في بروتوكول TCP/IP وبعض الأمور الأخرى حسب نوع البروتوكول.

لن أتطرق لتفصيل عمليات التركيب المتعددة فهي مخصصة لخبراء الشبكات و أجهزة التشغيل المركزية ، لكنني سأحاول أن أشرح بعض مميزاتها لمن يراها غريبة قليلاً.

هناك العديد من الشركات الأمريكية الصغيرة ترغب باستخدام تكنولوجيا قواعد البيانات المركزية ، ولكن معظمها لا يملك المال الكافي لتوظيف مبرمج و خبير شبكات ليدير له الجهاز المركزي ، و من هنا نشئت فكرة تأجير SQL Server عبر إنترنت ، خصوصا بعد انخفاض أسعار الاتصال بإنترنت بصورة رهيبة.

فبالأسعار الجديدة يمكن للشركات العادية الحصول على اشتراكات إنترنت تصل إلى 2 ميجابت في الثانية بمبلغ لا يزيد عن 40 – 60 دولار شهريا ، بالطبع هناك خطوط أسرع تصل سرعتها إلى 100 ميجابت في الثانية ولكنها أغلى ، في فلسطين لحظة كتابة هذا الكتاب خط بسرعة اثنين ميجابت في الثانية مرتبط بشبكة إنترنت بصورة مباشرة (بخطوط البنية التحتية للإنترنت) قد لا تقل تكلفته عن عشرون ألف دولار شهرياً ☺.

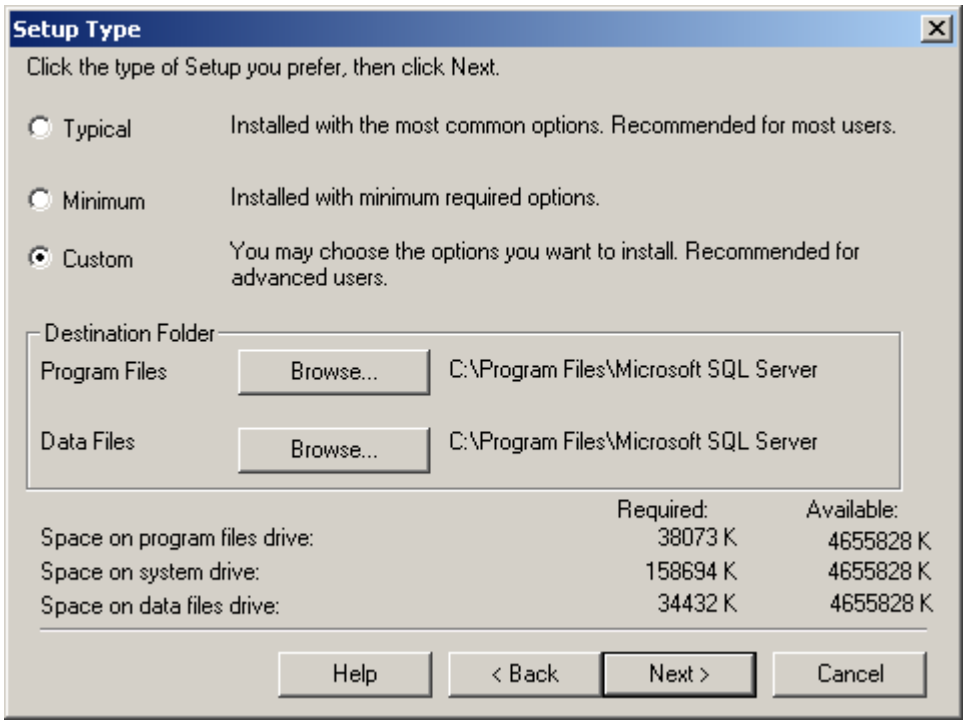
الاتصال ب SQL Server بسرعة اثنين ميجابت في الثانية تعتبر سرعة اكثر من كافية للشركات الصغيرة ، وهي سرعة عملاقة بالنسبة لمقدمي خدمة الإنترنت في العالم العربي ، حيث اعرف بان مزود الإنترنت الذي استخدمه يزود اكثر من ألف وخمسمائة مشترك عن طريق و سرعة خطه المربوط بإنترنت هو اقل قليلا من 2 ميجابت في الثانية.

لنرجع لموضوع تأجير أل SQL Server ، حيث تركب الشركات المتخصصة بالتأجير مجموعة من النسخ من SQL Server على الجهاز المركزي لديها ، وتقوم بتأجير النسخة ، وبذلك تكون قد وفرت المال لاستخدام جهاز مركزي واحد بدلاً من عشرة أو عشرون جهاز مركزي ، و من جهة أخرى تكون الشركة الصغيرة استفادت نظرا لان الخدمة تعتبر ارخص من توظيف موظفين إضافيين.

و يمكن استخدام عملية التركيب المزدوج لبعض الأمور الأخرى مثل تركيب نسختين من SQL Server على الجهاز المركزي ، واستخدام أحد النسخ للتجارب ، والثانية للعمل ، ولكنه يفضل في تلك الحالة تركيب نسخة واحدة فقط و بناء قاعدتي بيانات واستخدام أحدها للتجارب.

ذلك مرتبط بسياسة الشركة ، كما بالمناسبة يمكن تأجير قواعد البيانات بصورة منفصلة ، عوضاً عن تركيب نسخ متعددة من SQL Server على نفس الجهاز ، وذلك أيضا منوط بسياسة الشركة ، ولكن من الناحية التقنية ، فجميع الإمكانيات متوفرة.

لنختار الخيار Default ليقوم النظام أتوماتيكيا بإعداد أول نسخة و لننتقل إلى الشاشة رقم (9) بالضغط على الزر Next

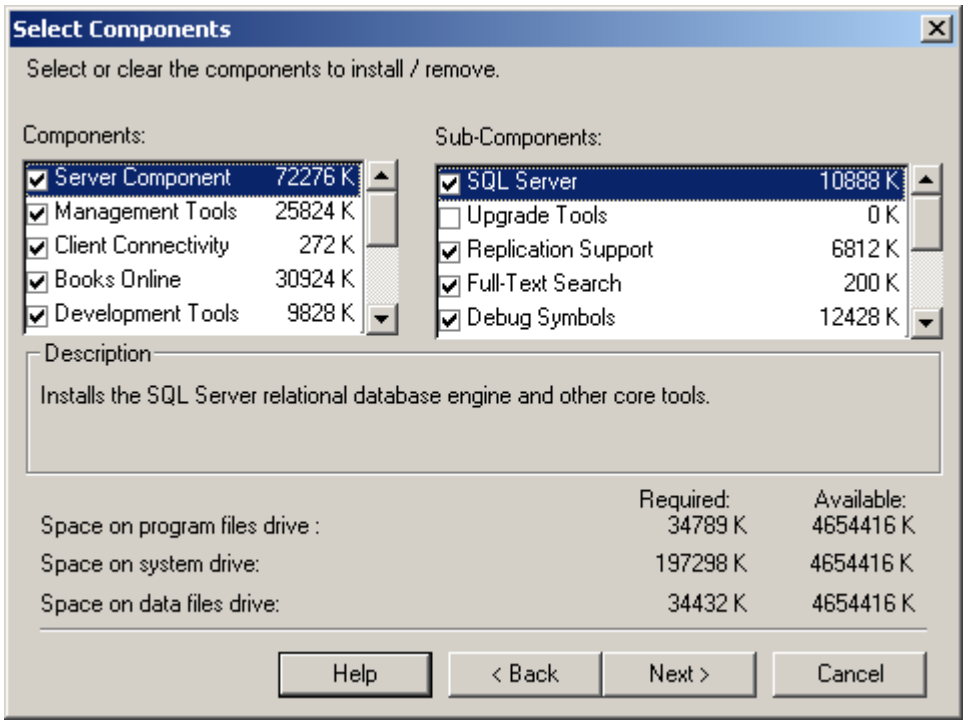


شاشة التركيب رقم (9)

و هي شاشة مخصصة لاختيار الأجزاء التي نرغب بتركيبها ، حيث يمكننا اختيار التركيب الافتراضي Typical ليركب معظم البرمجيات اللازمة ، مع محرك قواعد البيانات ، ويمكننا اختيار التركيب Minimum ليركب محرك البيانات فقط بدون برامج الإدارة ، والاختيار Custom لعرض قائمة بكل ما نرغب بتركيبه.

أنا كثيرا ما أفضل Custom حيث ارغب دائماً بالاطلاع على ما سأحصل بعد عملية التركيب ، و لأتمكن من تحديد ماذا أريد بالضبط.

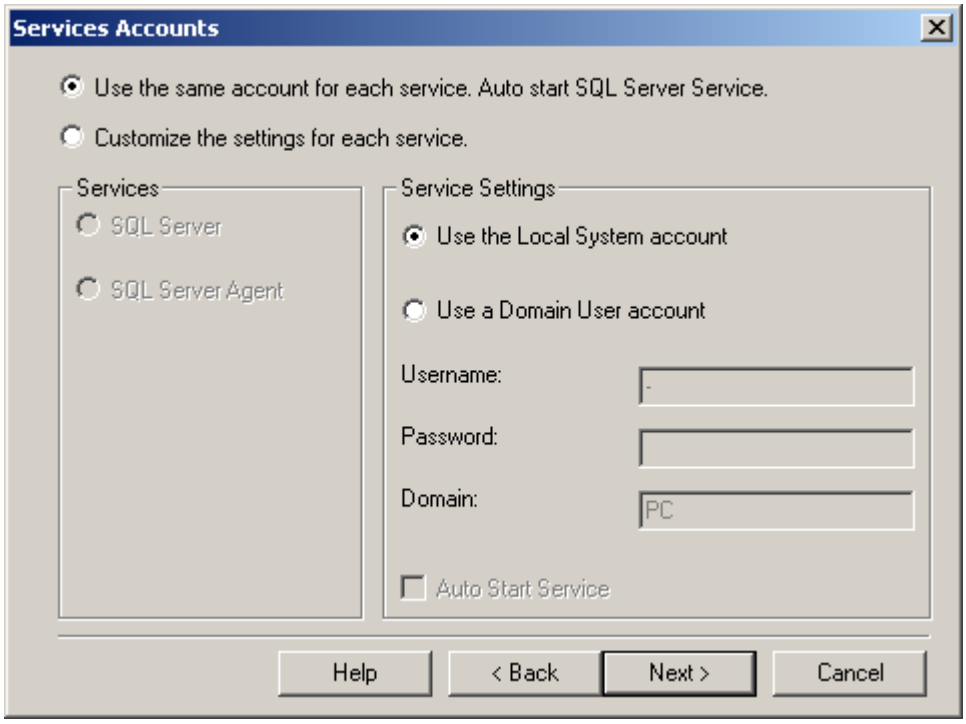
لنختار الخيار Custom ولننتقل إلى الشاشة رقم (10) بالضغط على الزر Next



شاشة التركيب رقم (10)

تبين شاشة التركيب رقم (10) قوائم بكافة الخيارات الممكن تركيبها ، لا وقت الآن لدي لشرح كل تلك الخيارات ، ولكنني سأترك ملاحظة حتى أتتمكن من النسخ القادمة من هذا الكتاب تفصيل جميع البرمجيات و استخداماتها ، على العموم يمكنكم اختيار كل القوائم ما عدا Upgrade Tools حيث أنها مخصصة لنقل نظم أل SQL Server القديمة إلى النظام الجديد ، مثل النظام SQL Server 6.5 ، ونحن لسنا بحاجة إليها.

بعد الانتهاء من تعليم جميع المربعات ، لننتقل إلى الشاشة رقم (11) بالضغط على الزر Next.

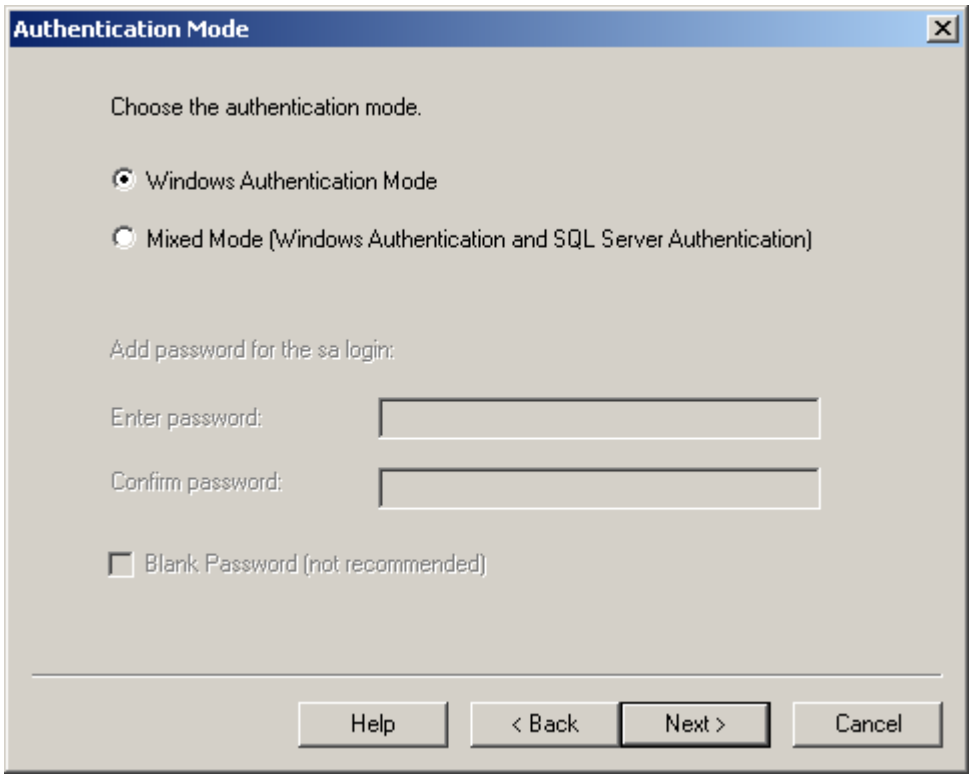


شاشة التركيب رقم (11)

و هي شاشة مخصصة لصلاحيات النظام SQL Server على الجهاز المركزي ، قد يستغرب البعض ولكن برنامج أل SQL Server مثلثة مثل البرامج الأخرى يحتاج إلى اسم دخول وكلمة مرور ليتمكن من العمل ضمن الجهاز المركزي ، في الأحيان العادية نستخدم أل System ، وهو اسم داخلي مسجل للنظام تستخدمه كل البرامج ، وسنستخدمه نحن أيضا ، لذلك قم بتعديل شاشة التركيب لديك لتشبه الشاشة السابقة رقم (11).

عملية الصلاحيات ، وتحديد اشتراك مستخدم أل SQL Server لوحده عملية ضرورية للغاية في الشبكات الكبيرة ، حيث نحاول في تلك الشبكات أن نفصل بين صلاحيات المبرمجين ، والصلاحيات عل الجهاز المركزي ، و كذلك صلاحيات الطواقم الفنية المتخصصة بإصلاح قواعد البيانات والنسخ الاحتياطي و ما إلى ذلك.

بعد الانتهاء من تعديل الشاشة السابقة ، لننتقل إلى الشاشة رقم (12) بالضغط على الزر Next.



شاشة التركيب رقم (12)

و هي تحدد طريقة توزيع الصلاحيات ، قد تبدوا غريبة في بادئ الأمر ولكن لها أسبابها

حيث يقدم SQL Server طريقتين لتحديد الصلاحيات ، الطريقة الأولى و هي أن يستخدم أل SQL Server أسماء المستخدمين و كلمات السر الخاصة بالجهاز المركزي ، حيث إن الجهاز المركزي هو المسؤول الوحيد عن تحديد الصلاحيات.

الطريقة الثانية هي ترك SQL Server ليقوم ببناء أسماء مستخدمين وكلمات سر خاصة بهما ، وذلك بمعزل عن الجهاز المركزي ، و الاستفادة من الأسماء الموجودة في الجهاز المركزي مسبقاً ، تلك الطريقة استخدمت سابقاً في الشبكات الكبيرة نظراً لمحدودية نظام الصلاحيات في Windows NT 4 ولكنه بعد توفر شجرة الصلاحيات Active Directory في ويندوس ألفين فلم نعد بالحاجة إلى هذا الخيار.

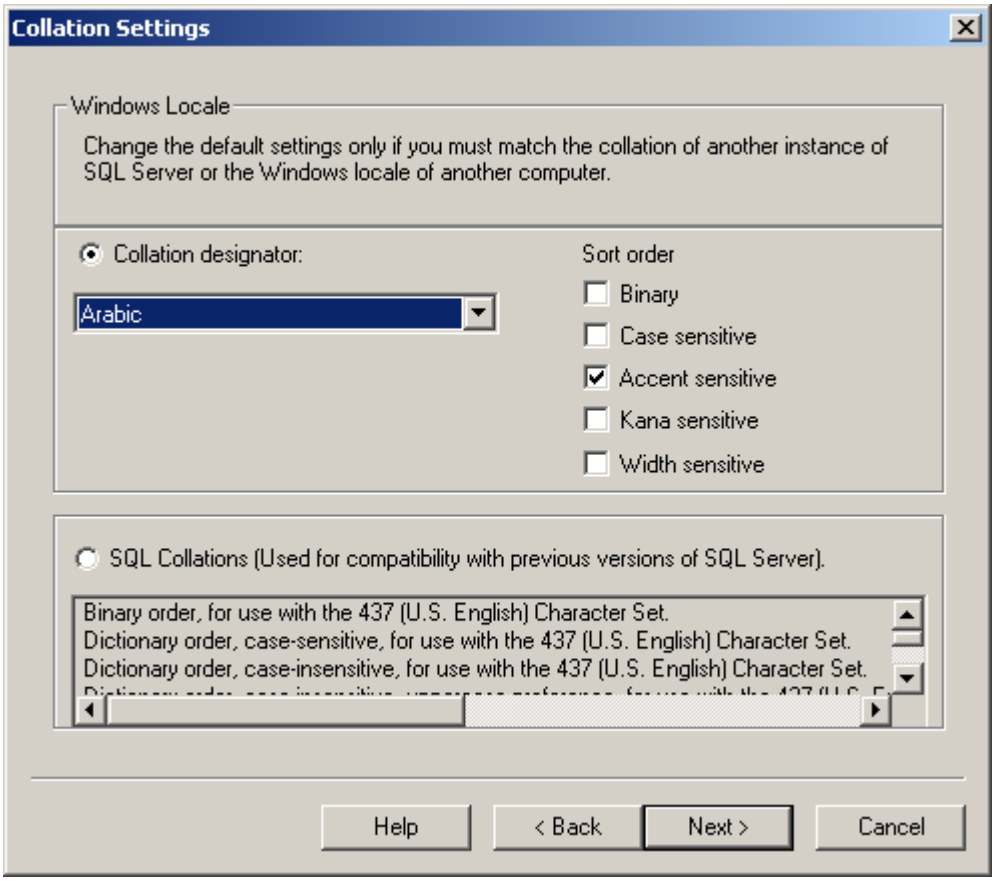
في حالة أن اخترنا الخيار الثاني فيجب أن نحدد كلمة سر لمدير قاعدة البيانات في SQL Server حيث أن تركنا كلمة السر فارغة ستمكن خبراء أل SQL Server من اقتحام جهازنا فوراً ، حيث إن مدير الجهاز بدون كلمة سر!. اسم الدخول لمدير الجهاز هو sa و كلمة السر هي الكلمة التي ستضعها في المربعين السابقين.

أنا افضل اختيار الخيار الأول لأنه ائمن ، ولكن حتى ولو اخترت الخيار الثاني أو الأول فيمكنك تغييره بعد ذلك من لوحة إدارة خادم قاعدة البيانات بسهولة ، كما انه لو اخترت أية من الخيارين السابقين فلن تتغير الشاشات الخاصة بإدارة المستخدمين من داخل SQL Server حيث سيمكنك من إضافة وتعديل بيانات مستخدمين لا ينتمون إلى قائمة المستخدمين الخاصة بالجهاز المركزي.

هذا يربك البعض في البداية ، حيث تسائلون و ما الفائدة من الشاشة السابقة ، و ارغب بان الفت انتباهكم بان الشاشة السابقة تفعل أو تبطل مفعول الصلاحيات المذكورة سابقاً ، فلو على سبيل المثال اخترت الخيار الأول ،

سيمكنك إضافة مستخدمين بداخل أل SQL Server و ذلك بمعزل عن الجهاز المركزي ، ولكن الجهاز المركزي سيرفض جميع الاتصالات من هؤلاء المستخدمين ، والذين لا يتواجدون في سجلاته أيضا.

أنا افضل اختيار الخيار الأول ، و من ثم الانتقال إلى الشاشة رقم (13) بالضغط على الزر Next.



شاشة التركيب رقم (13)

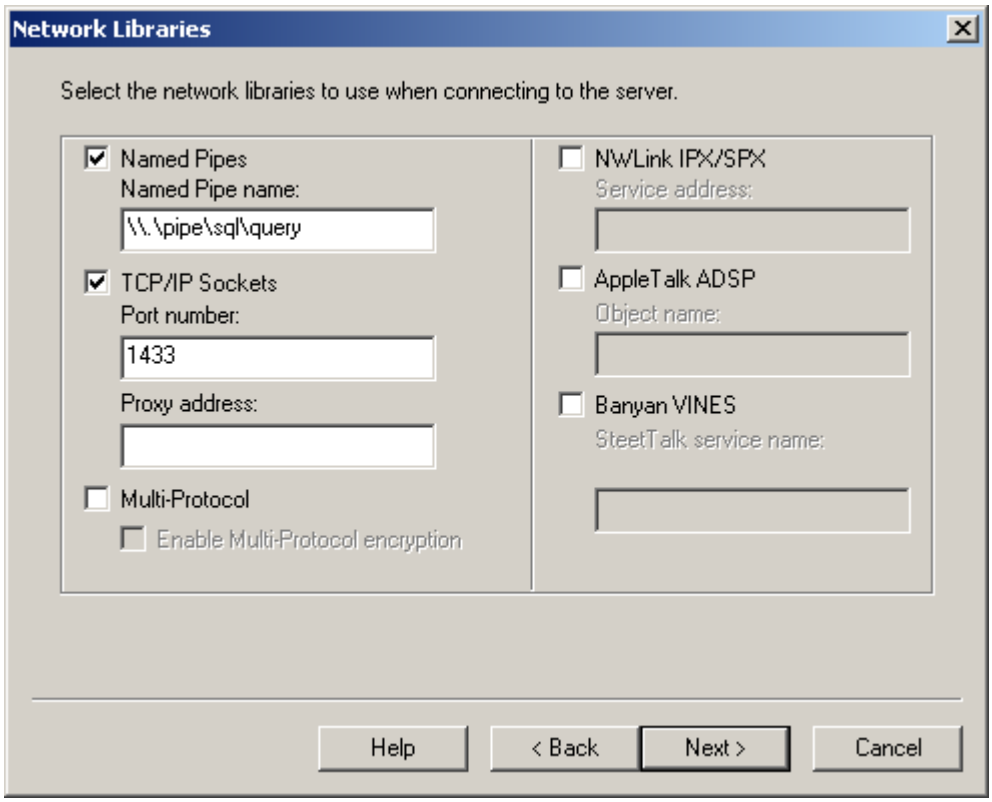
في الشاشة رقم (13) تظهر لنا خيارات إعداد لغة محرك واعد البيانات المركزية ، سوف تلاحظوا في المستقبل بأنه هناك طرق كثيرة للتحكم باللغات و بناء برامج متعددة اللغات.

لقد عدلت مايكروسوفت أيضا مسألة دعم لغات متعددة أيضا في جميع نسخ أل SQL Server ، حيث كل مرة كانت تعتقد بأنها وصلت إلى افضل دعم للغات ، و كلما انتشر SQL Server اكثر في العالم كلما اكتشفت مايكروسوفت عيوب جديدة وكلما قامت بتعديل مبدأ دعم اللغات المتعددة.

لذلك ستجد فرق واضح بين النسخ 6.5 و 7.0 و 2000 في طريقة دعمها للغات.

سأقوم بشرح تلك الشاشة في النسخ القادمة من الكتاب ، أما الآن فحاول أن تظهر تقوم بعملية التركيب كما في الشاشة السابقة.

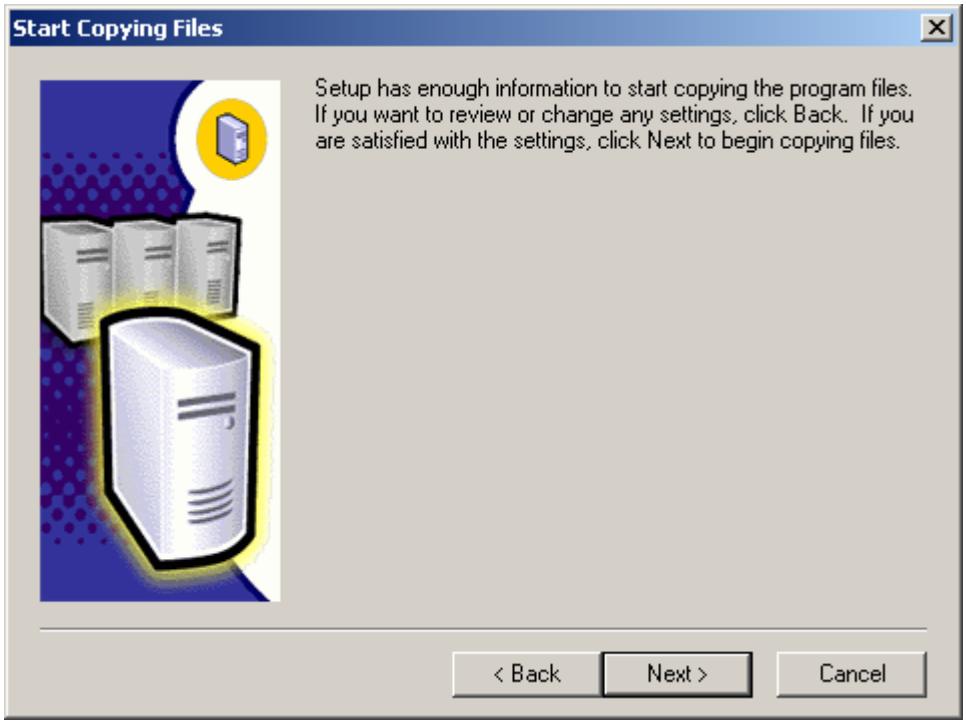
لننتقل إلى الشاشة رقم (14) بالضغط على الزر Next.



شاشة التركيب رقم (14)

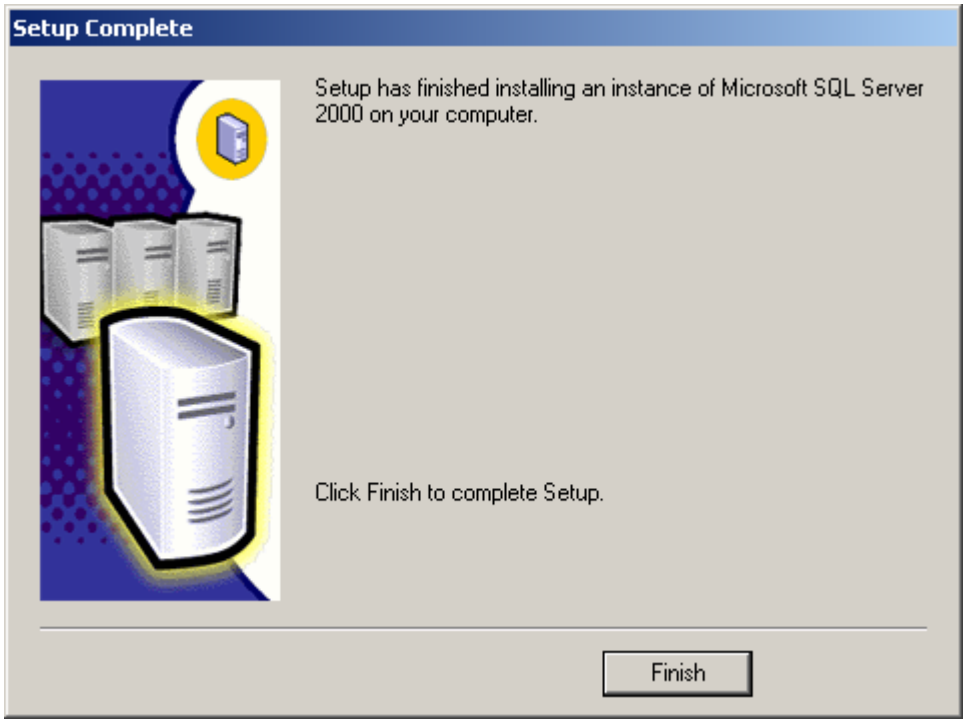
وهي مخصصة لتحديد البروتوكولات التي نرغب باستخدامها للاتصال ب SQL server ، برجاء عدم تغير أية شيء في تلك الشاشة ، حيث سأشرحها لاحقا لضيق الوقت الآن.

لننتقل إلى الشاشة رقم (15) بالضغط على الزر Next.



شاشة التركيب رقم (15)

و هي تخبرك باستعداد النظام ببدء عملية التركيب ، حتى هذه اللحظة لا يزال بإمكانك التراجع ، أو البدء بتركيب الجهاز المركزي.
اضغط على الزر Next لتبدأ عملية التركيب ، و عند انتهائها ستظهر لك الشاشة رقم (16)



شاشة التركيب رقم (16)

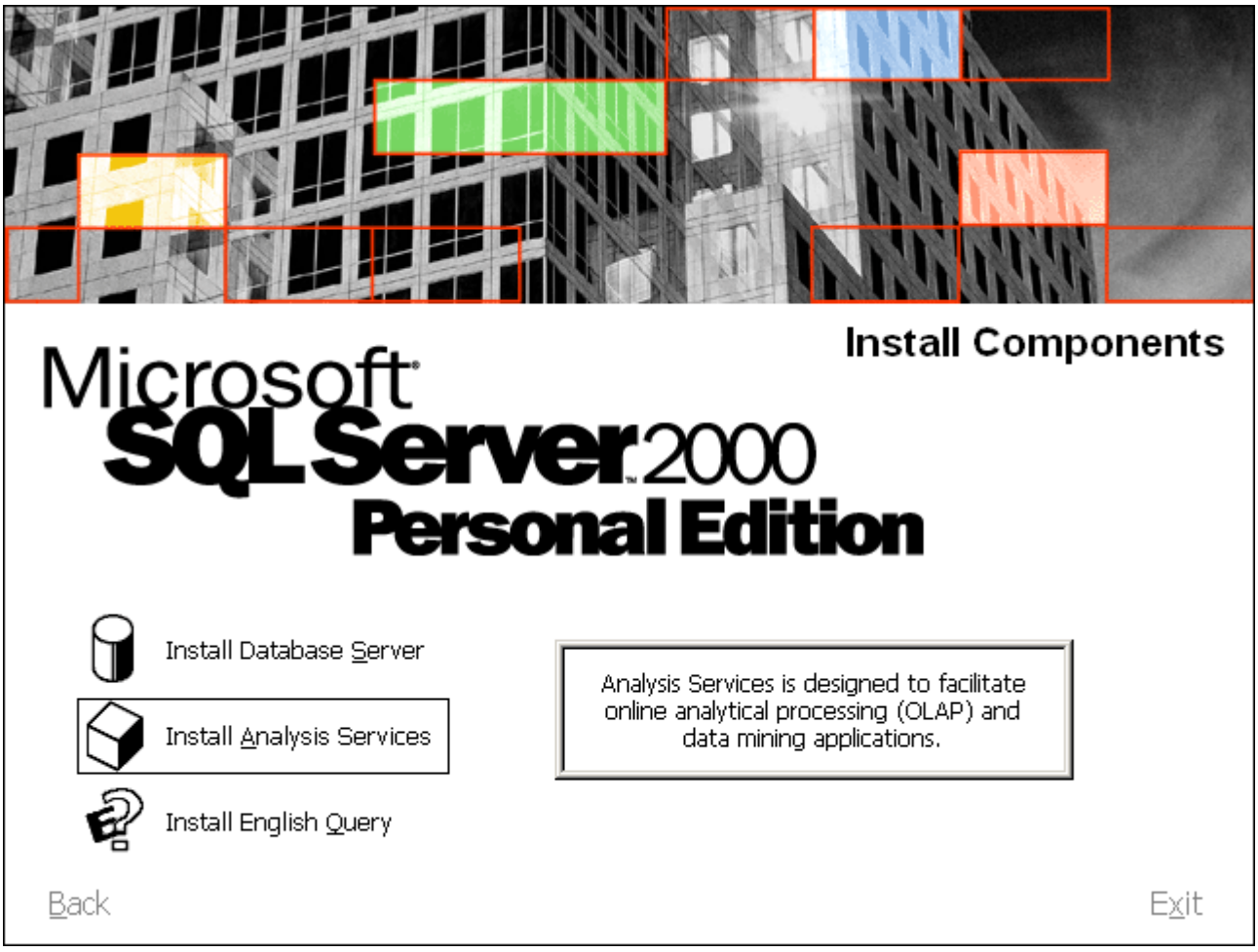
حيث تخبرك بانتهاء عملية التركيب بسلام ، وجهازك جاهز لبدء العمل مع SQL Server 2000 نصيحتي الشخصية هي بان تقوم بإعادة تشغيل جهازك ، مع انه ليس بالأمر الضروري ، ولكنني افضل ذلك.

أما إن كنت تخطط لتركيب الجزئيين الإضافيين من SQL server فيمكنك تركيبهما وبعد ذلك القيام بإعادة تشغيل الجهاز.

إعداد Microsoft SQL Server 2000 Analysis Services لأول مرة

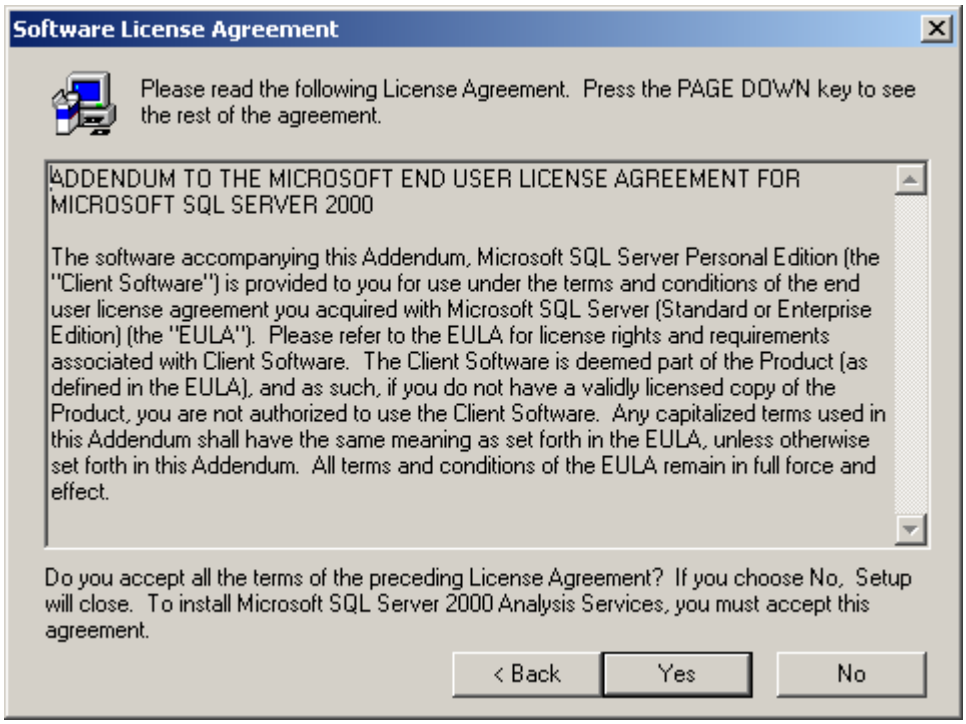
ستتوفر في النسخة القادمة من الكتاب أي بعد أسبوع تقريبا عملية التركيب مع الشرح ، أما الآن فاعذروني لوضع الشاشات فقط حيث ارغب بان اكمل الشرح في أجزاء أخرى من الكتاب ، لتركيب النظام اتبع الشاشات التالية وستركبه

شاشة التركيب رقم (1)

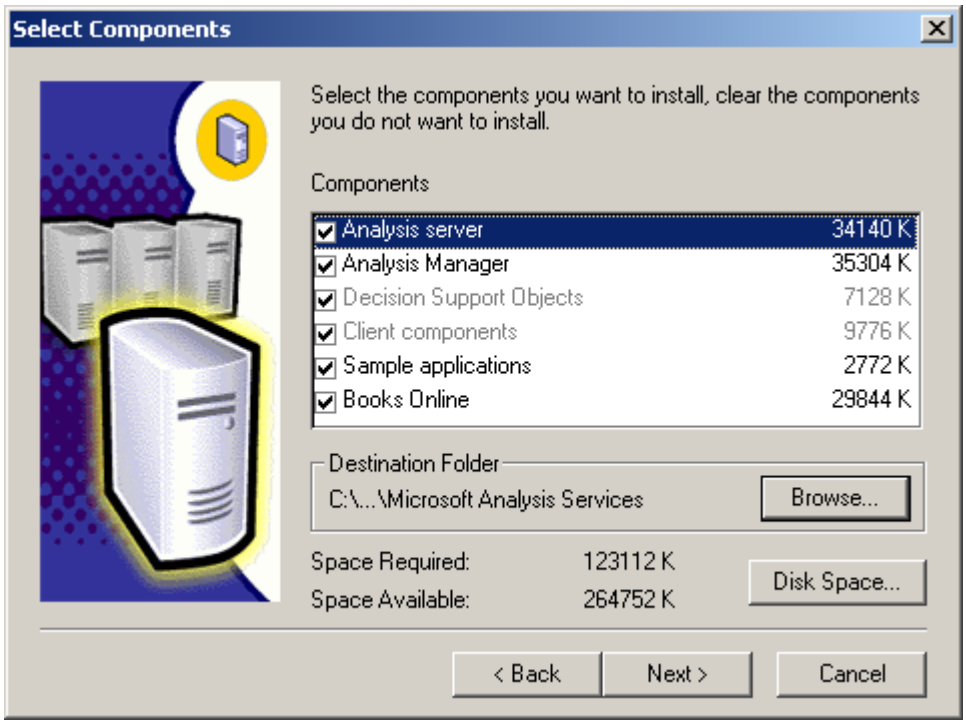


شاشة التركيب رقم (2)

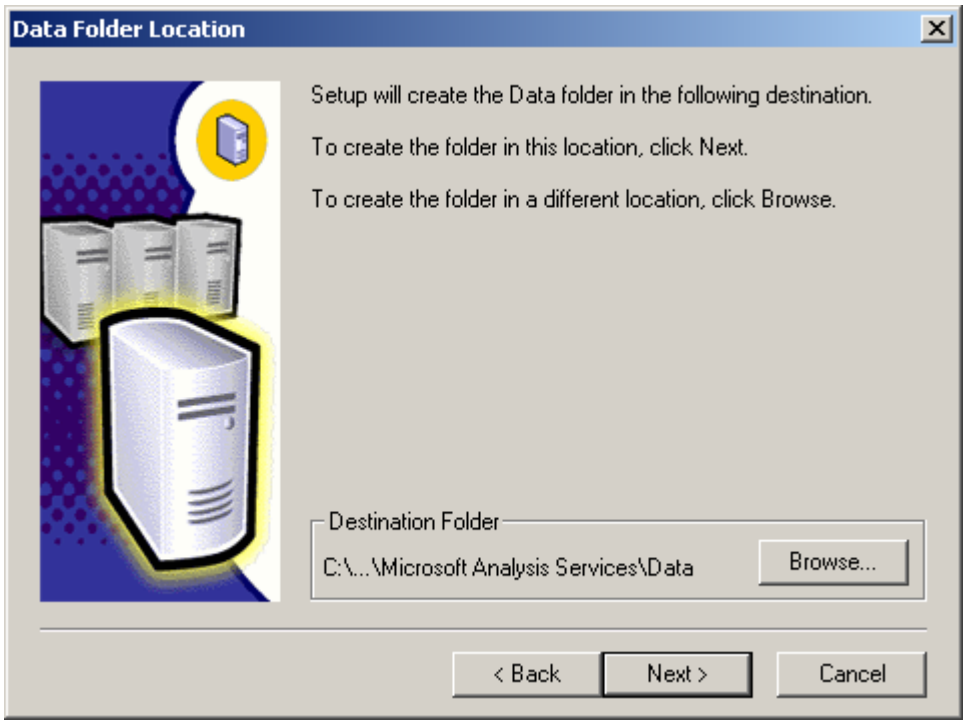
شاشة التركيب رقم (3)



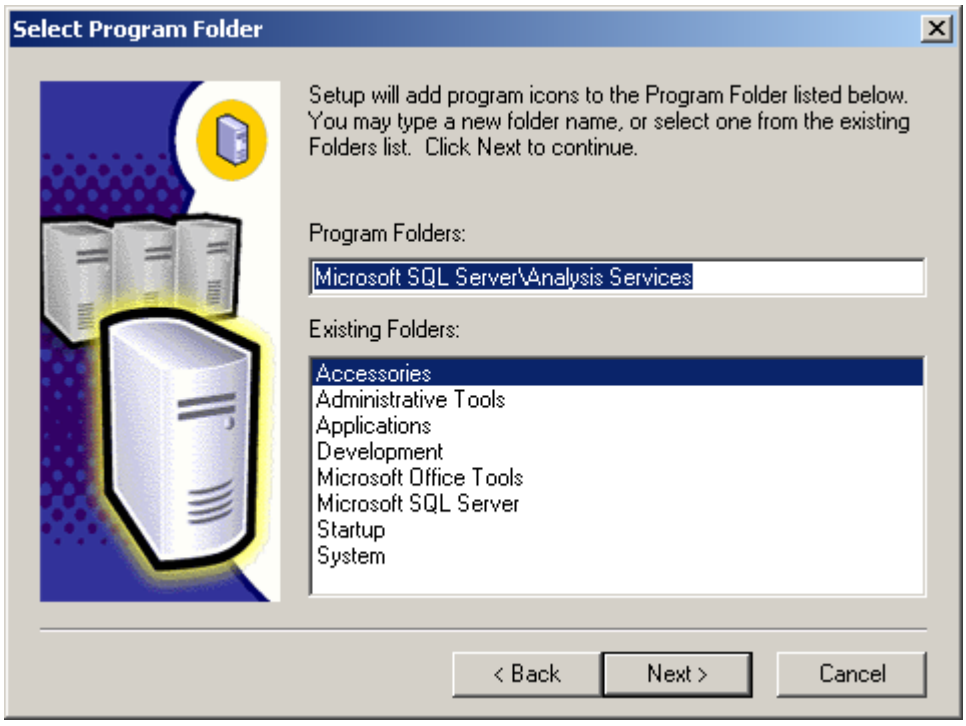
شاشة التركيب رقم (4)



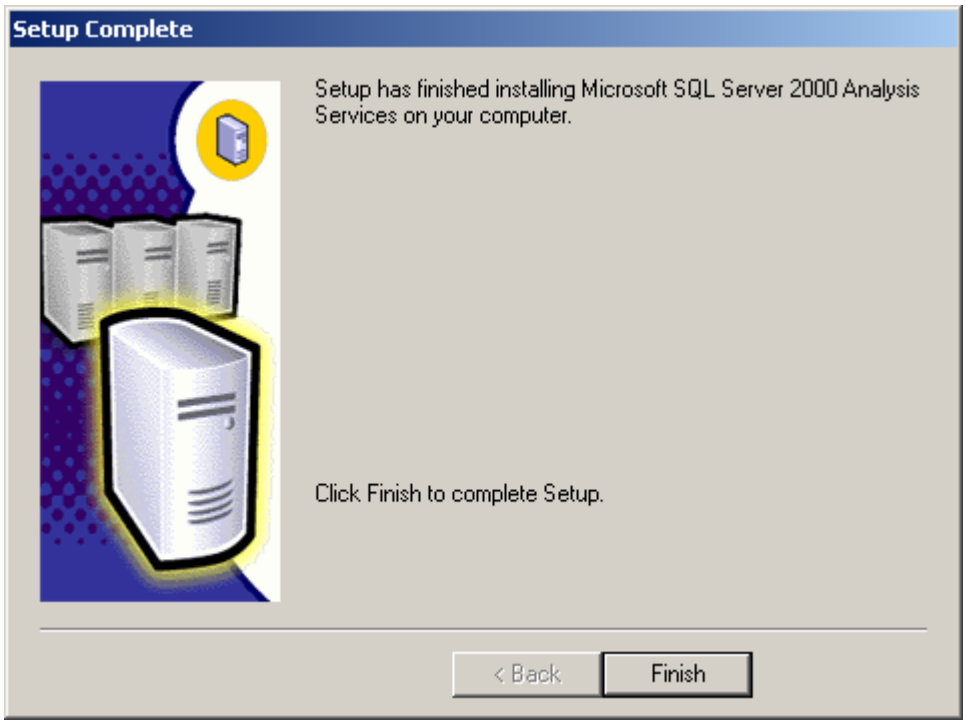
شاشة التركيب رقم (5)



شاشة التركيب رقم (6)



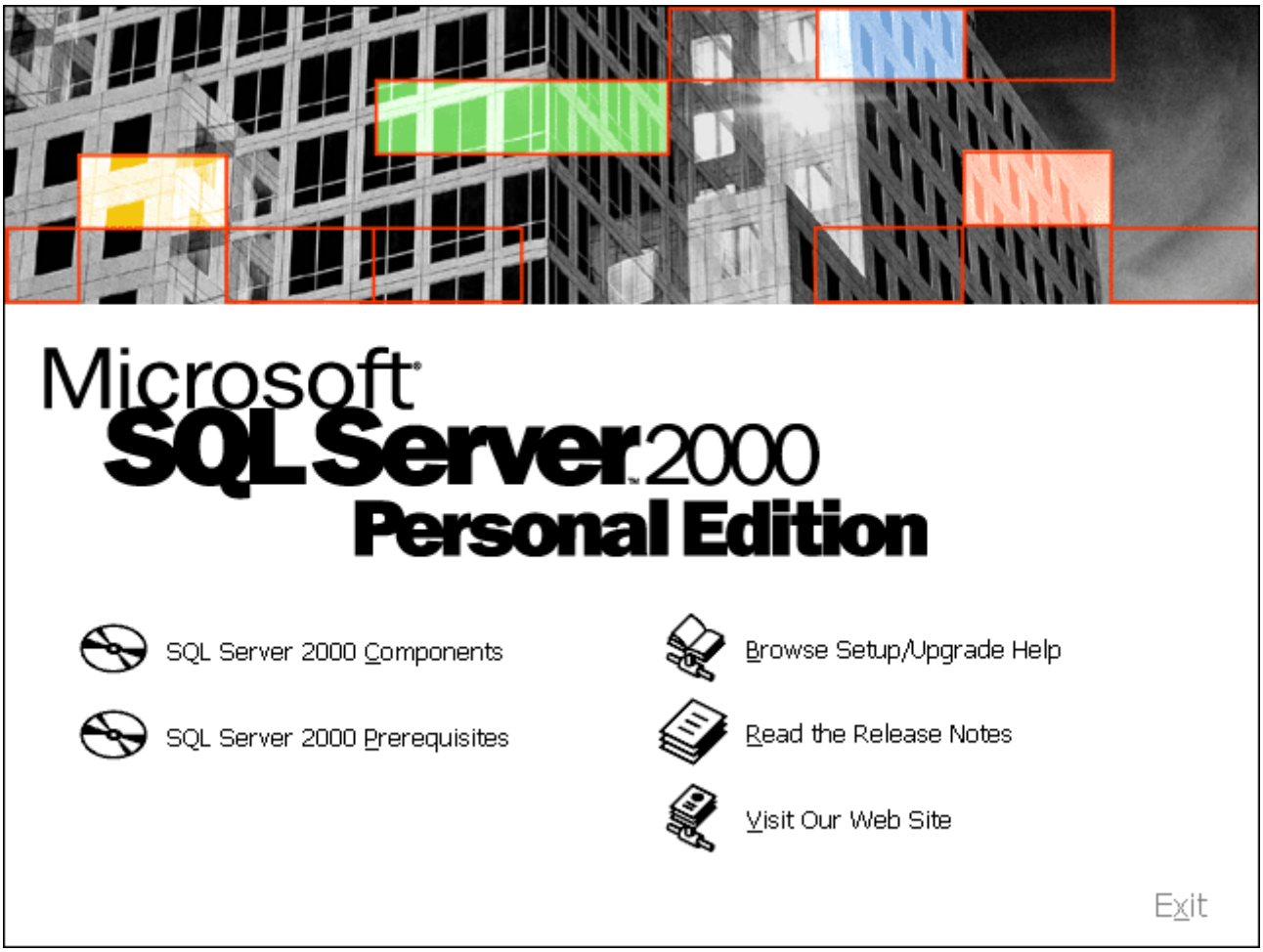
شاشة التركيب رقم (7)



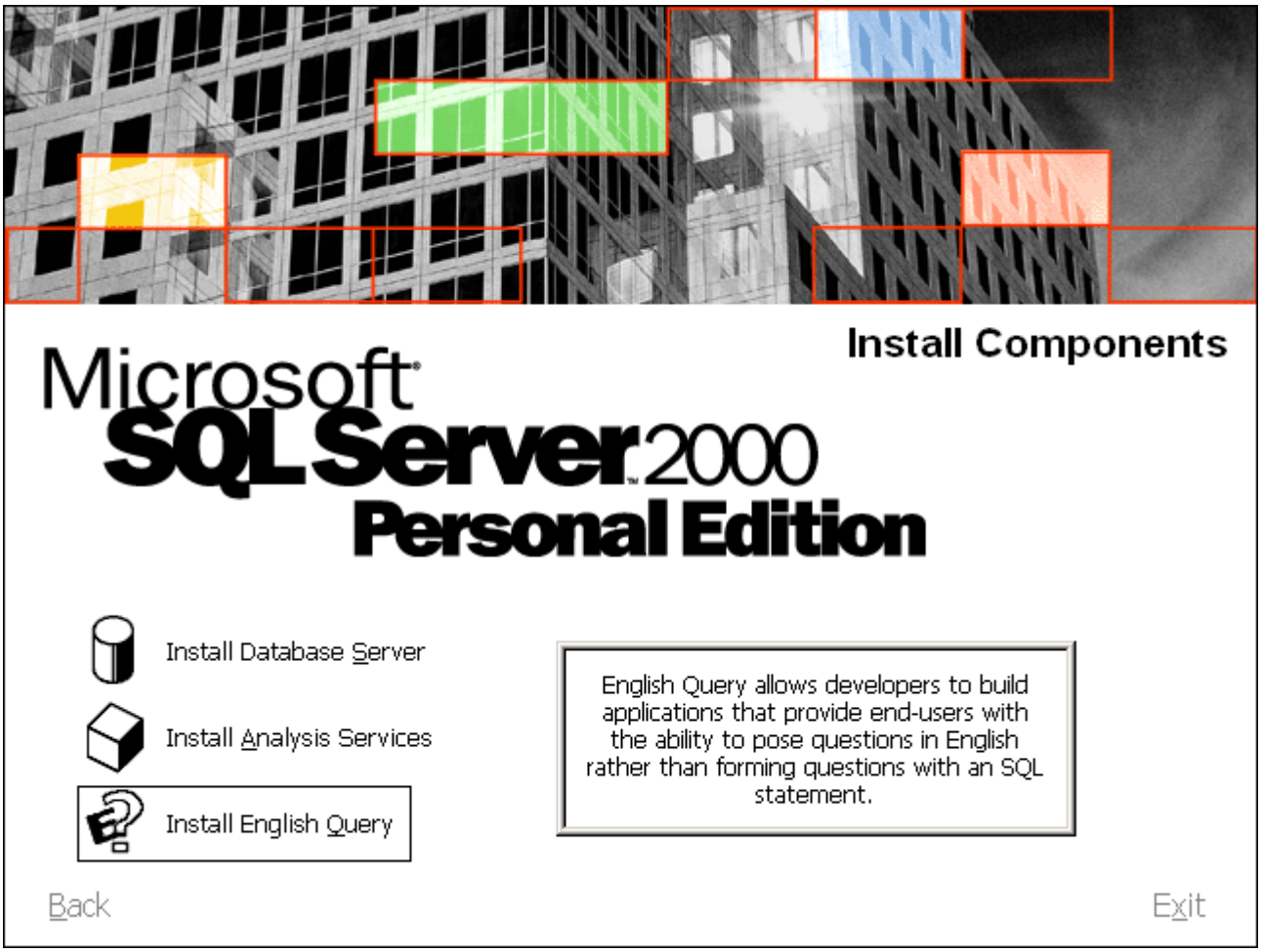
شاشة التركيب رقم (8)

إعداد Microsoft SQL Server 2000 English Query لأول مرة

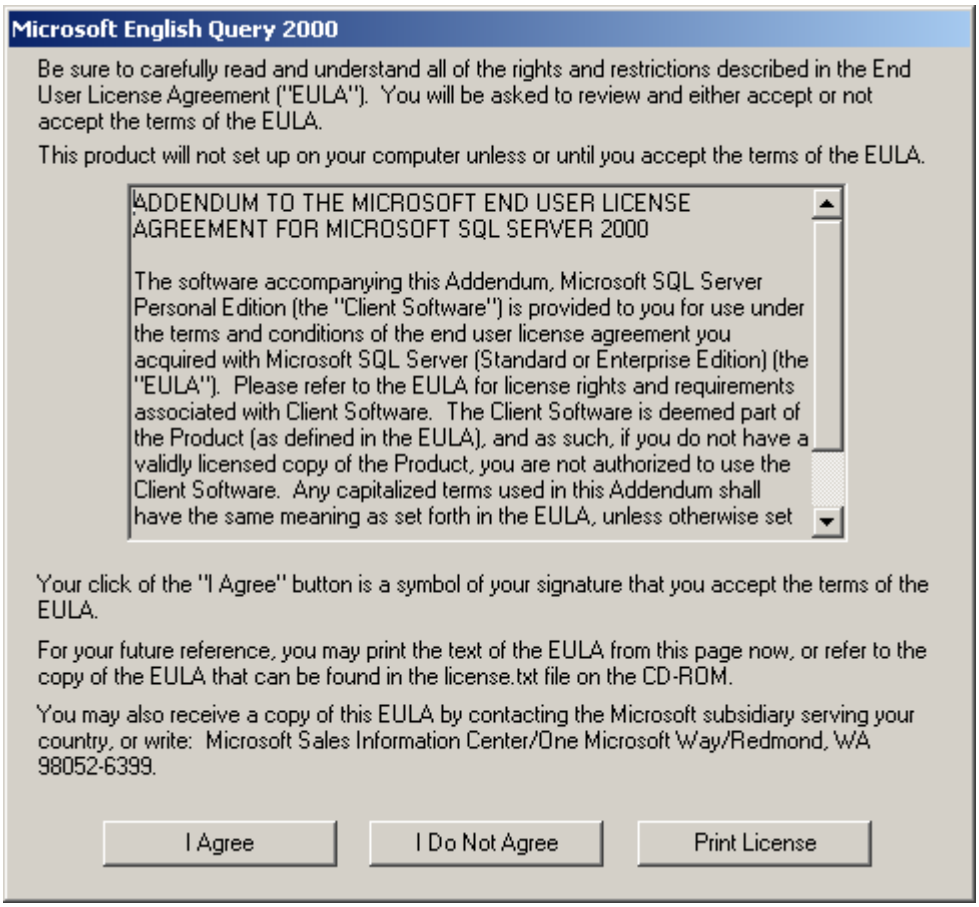
ستتوفر في النسخة القادمة من الكتاب أي بعد أسبوع تقريبا عملية التركيب مع الشرح ، أما الآن فاعذروني لوضع الشاشات فقط حيث ارغب بان اكمل الشرح في أجزاء أخرى من الكتاب ، لتركيب النظام اتبع الشاشات التالية وستركبه.



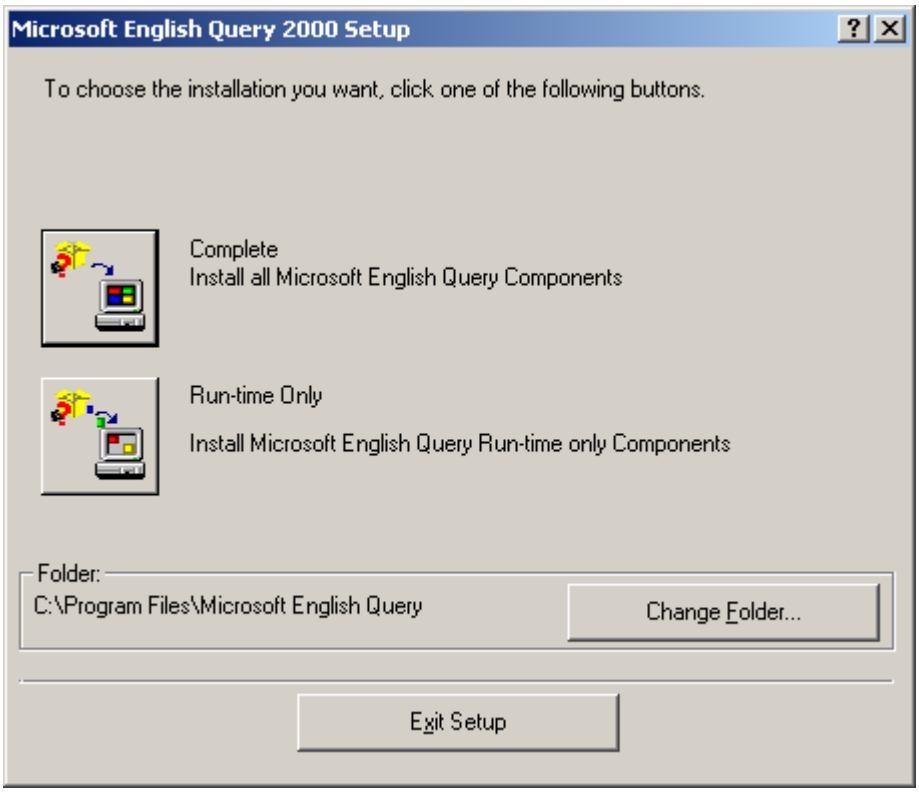
شاشة التركيب رقم (1)



شاشة التركيب رقم (2)



شاشة التركيب رقم (3)



شاشة التركيب رقم (4)

شاشة التركيب رقم (5)

دورة سريعة في لغة Structured Query Language

قد يتساءل البعض، يعني هل أن لم افهم لغة SQL لن أتمكن من بناء برنامج قواعد بيانات بواسطة Access و SQL Server ؟

جوابي بالطبع لا، فقد تصنع برامج كبيرة بدون أن تتعلم تلك اللغة، أو مع معرفة بسيطة فيها، ولكنك لن تتمكن من صناعة برامج قواعد بيانات بصورة صحيحة، فالمعالجات قد تساعدك وتبني لك الجزء المهم من الكود، ولكن اللمسات اليدوية على ذلك الكود مهمة للغاية.

لذلك يفشل الكثير في بناء البرامج بواسطة Microsoft Access فهم لا يعرفون SQL ولا ينتبهون إلى قوتها والى المميزات التي يمكن أن تقدمها لك في عملية بناء برنامجك.

لذلك يلجا الكثيرون إلى Oracle، حيث يتم تعليمهم لغة PL/SQL رغما عنهم، فبدون أن تعرف PL/SQL فلن تتمكن من القيام بشيء هناك، وهكذا يعتادون عليها ويكتشفون أهميتها، نفس الشيء سأقوم به هنا أيضا، يمكنك بالطبع تخطي ذلك الجزء أو الإطلاع عليه بصورة سريعة، ولكنك حتما ستحتاجه في النهاية.

مقدمة إلى لغة SQL.

تأتي كلمة SQL اختصارا لكلمة Structured Query Language حيث يوحي اسمها للوهلة الأولى بأنها لغة مخصصة لاسترجاع المعلومات من قاعدة البيانات فقط، ولكنها في الحقيقة أكثر من ذلك، فهي قادرة على تعديل وإضافة وحذف البيانات من قاعدة البيانات ، كما إنها قادرة أيضا على القيام بالكثير من الأمور الأخرى.

لا يجب أن تكون عبقري في كتابة أوامر SQL الرئيسية، فقد تجعل المعالجات Wizards تكتب لك جزء كبير من الكود، حيث يمكنك بعد ذلك البناء على هذا الكود وتطويره، لذلك يجب عليك فهم هذه اللغة على الأقل.

بالنسبة لمن اعتاد بالبرمجة بالصورة العادية، البرمجة الكائنيه أو الإجرائية، و هما الطرق المستخدمة في لغات البرمجة اليوم، فمن اعتاد عليها سيتغلب قليلا في فهم منطق لغة SQL وذلك في البداية فقط، حيث ستشعر بان كل شيء مقلوب بصورة كبيرة، وأحيانا غير منطقي، ولكن SQL تبقى SQL فهي موجودة منذ عشرات السنوات و مع أنها بحاجة إلى تغير جذري في بنيتها إلا أن ذلك أصبح من شبه المستحيل الآن، فلقد تم بناء الكثير من التكنولوجيا عليها في غضون الفترة السابقة، ولذلك يجب أن نتغلب قليلا لنتعلمها كما هي.

حسناً، قبل أن ابدأ أود أن أوضح شيء بدء يدور في ذهنكم، وهو بان Microsoft SQL Server يستخدم Transact SQL وليست SQL العادية، كما إن Microsoft Access يستخدم لغة SQL لا يستطيع SQL Server فهمها، أي إنها تختلف قليلا في تركيبتها.

وهذا صحيح، فهناك اختلاف بسيط للغاية بين أل SQL في Access وذلك عندما يعمل لوحده و SQL Server اختلاف بسيط ولكنه موجود في كل عبارة، لذا من تعود بكتابة أوامر أل SQL في Access قد يواجه بعض المصاعب البسيطة في البداية عندما يحاول كتابة نفس الأوامر في SQL Server.

ولكن تلك الاختلافات البسيطة آخذة في الزوال، حيث إننا سنتعلم هنا SQL بالصورة الصحيحة و حسب المواصفات القياسية العالمية وهي SQL- 92.

بالنسبة للاختلاف بين SQL و Transact SQL فاللغة الثانية هي تطوير للغة الأولى، أي أنها اشمل منها بكثير واقرب إلى لغة برمجة متكاملة، وهي تضم اللغة الأولى بصورة كاملة.

أحد أكثر الأوامر شيوعاً والذي ستجده في معظم عبارات تلك اللغة هو الأمر SELECT وهذا طبيعي نظراً لان تلك اللغة صممت أساسا لاسترجاع البيانات من قواعد البيانات، ومن ثم تم تطويرها لمعالجة البيانات أيضا.

الأمر SELECT أحد أوامر اللغة الرئيسية

وظيفته الأساسية هي اختيار مجموعة من السجلات من جدول في قاعدة البيانات.

الاستعلام البسيط

كما ذكرنا سابقا فمفهوم قاعدة البيانات هو تخزين البيانات في جداول، مقسمة إلى أعمدة وأسطر، ولنتمكن من الاستفسار عن مجموعة من الأسطر (التي نسميها سجلات) نستخدم الأمر SELECT

مثال على ذلك

لنفترض لدينا جدول مخزن في قاعدة البيانات و يحتوي على بيانات الموظفين كالجدول التالي

Table Name: EmpTABLE

KEY	Name	Address	City	Salary	Benefits
1	Employee 1	Address 1	City 1	1000 $	150$
2	Employee 2	Address 2	City 1	1500 $	200$
3	Employee 3	Address 3	City 2	1700 $	50$
4	Employee 4	Address 4	City 1	1200 $	0$
5	Employee 5	Address 5	City 2	1000 $	300$
6	Employee 6	Address 6	City 2	2000 $	20$
7	Employee 7	Address 7	City 1	1350 $	45$

ورغبنا بان نطلب الجدول من قاعدة البيانات لنراه على الشاشة، مع عرض لكل البيانات على الشاشة، فببساطة سنستخدم الأمر SELECT بالصورة التالية:

```
SELECT    KEY, NAME, ADDRESS, CITY, SALARY
FROM    EmpTABLE
```

والنتيجة ستكون بأننا سنحصل على الجدول السابقة على شاشتنا ، ومن المثال السابق يمكننا أن نستنتج بأن الأمر Select يتصرف بالصورة التالية:

```
SELECT  اسم العموم  ,  اسم العمود  ,  اسم العمود  ,  ...
FROM    اسم الجدول
```

إذن لنجرب الأمر SELCET مرة أخرى و لنقم باختيار اسم الموظف وراتبه فقط

```
SELECT   NAME, SALARY
FROM   EmpTABLE
```

والنتيجة ستكون الجدول التالي

Name	Salary
Employee 1	1000 $
Employee 2	1500 $
Employee 3	1700 $
Employee 4	1200 $
Employee 5	1000 $

Employee 6	2000 $
Employee 7	1350 $

كما سنلاحظ لقد حصلنا على كل الصفوف من الجدول و لكن الأعمدة التي قمنا باختيارها فقط.

الآن ، ماذا إن افترضنا بان الجدول يحتوي على الكثير من الأعمدة ورغبا في الحصول على كل الأعمدة معاً ، هل يجب أن نعيد كتابة كل أسماء الأعمدة ؟ ، لا اعتقد ذلك فببساطة يمكننا استخدام رمز النجمة مع عبارة أل SELECT لتعطينا كل الأعمدة.

العبارة التالية مثال على ذلك

```
SELECT   * FROM   EmpTABLE
```

ونتيجتها هي عرض جميع أعمدة جدول الموظفين.

ملاحظة مهمة: يمكننا أن نجزئ عبارة أل SQL إلى عدة سطور، أو يمكن كتابة العبارة في سطر واحد، و في كل من الحالتين ستعمل العبارة بدون أية مشاكل ، وهذا شيء جيد للغاية في عملية تنظيم الكود.

الاستعلام المشروط

عملية استرجاع البيانات في اغلب الأحيان لا تقتصر على طلب كل البيانات ، ففي الكثير من الأحيان قد نطلب من النظام بان يرجع لنا قائمة بأسماء الموظفين من المدينة الأولى مثلا ، أو أسماء الموظفين الذين يتقاضون راتب أعلى من 1500 دولار ، أو غيرها من الأمور.

جميع تلك الطلبات تعالج أيضا من خلال الأمر SELECT حيث بإمكانه فلتره السجلات أيضا ، وإحضار السجلات التي نحن بحاجة إليها فقط.

إذن الأمر SELECT قادر على القيام بالاختيار المشروط للبيانات ، وليتمكن من القيام بذلك فهو يستخدم الرموز التالية

الرمز	الوصف
=	يساوي
<	اصغر من
>	اكبر من
<=	اصغر من أو يساوي
>=	اكبر من أو يساوي
<>	لا يساوي
=!	لا يساوي
>!	ليس أكبر من
<!	ليس أصغر من

ملاحظة: آخر ثلاثة رموز لا تتفق مع المواصفات العالمية ل SQL-92 ولكنها تتفق مع Transact SQL

تتم عملية الاختيار المشروط عن طريق الأمر المساعد WHERE والذي يندمج مع الأمر SELECT بالطريقة التالية

```
SELECT   العمود الأول, العمود الثاني , العمود الثالث , ...
FROM   الجدول
WHERE   الشرط
```

و في المثال التالي نقوم باختيار أسماء جميع الموظفين من المدينة رقم واحد

```
SELECT NAME, CITY FROM EmpTABLE WHERE CITY = 'City 1'
```

والنتيجة ستكون بعرض اسم الموظف و اسم المدينة ، حيث فرزنا الموظفين أظهرنا الموظفين من المدينة رقم واحد فقط ، وبذلك قد نكون قد أجرينا فرز للأعمدة و للصفوف في الجدول.

Name	City
Employee 1	City 1
Employee 2	City 1
Employee 4	City 1
Employee 7	City 1

يمكننا أيضا استخدام رموز أخرى مع الأمر where ففي المثال السابقة استخدمنا رمز أل "يساوي"، ولكننا يمكن أن نطلب مثلا قائمة بجميع الموظفين الذين تزيد رواتبهم عن ألف وخمسمائة دولار مثلا، و هنا يجب استخدام الرمز "أكبر من"، كالمثال التالي

```
SELECT * FROM EmpTABLE
WHERE SALARY > 1500
```

نتيجة ذلك الاستعلام ستكون جميع الموظفين الذين تزيد رواتبهم عن 1500 دولار، ولكن ماذا سيحدث بالنسبة للموظفين الذين تساوي رواتبهم الألف وخمسمائة دولار ؟

الجواب هو بأنه لن يتم إحضارهم، لأننا طلبنا اكبر من و ليس أكبر من و يساوي، والجهاز نفذ ما طلبناه بالضبط، على العموم انتبه لتلك التفاصيل الصغيرة، فهي ضمن الأخطاء الشائعة التي يقع بها جميع المبتدئون في لغة SQL، حتى المتقدمون أيضا، أنا شخصيا كثيراً ما أجد مشاكل في برامجي، وعند الفحص الدقيق ألاحظ بأنني وقعت في الخطاء السابق أيضا.

قبل أن نكمل ارغب بالإجابة على سؤال يطرحه بعضكم الآن ، وهو لماذا استخدمنا الإشارتين التاليتين ' ' في الكود السابق الذي يفرز الموظفين من المدينة رقم واحد ، حيث أننا وضعنا سام المدينة بين الإشارتين ، و في نفس الوقت لم نستخدمهم في الكود الذي يفرز الموظفين الذين تزيد رواتبهم عن 1500 دولار!

و هذا سؤال مهم للغاية، حيث تعامل لغة أل SQL القيم النصية بصورة مختلفة عن القيم الرقمية ، ولذلك يجب وضع كل القيم النصية دائماً بين إشارتين ' ' وإلا ستنتج عبارة خطاء أثناء تنفيذ البرنامج ، والإشارتين السابقتين هما شرط أساسي لتخبر لغة البرمجة بان القيمة عبارة عن نصوص.

بالطبع الأمر ليس غريب على المبرمجين، ولكنه غريب على من يستخدم SQL لأول مرة ولم يبرمج من قبل، حيث يتساءل الآن "ما هي القيم بالضبط ؟"، والجواب سهل، كما سترى في كود أل SQL فهو عبارة عن مجموعة أوامر مثل الأمر Select و From وغيرهما، و هي أوامر يتعرف عليها النظام على الفور، وتلي تلك الأوامر في معظم الأحيان أسماء الأعمدة أو أسماء الجداول، وهي أيضا أسماء يتعرف عليها النظام.

ولكنه من الصعب عليه التعرف على القيم ، كما أن القيمة قد تحتوي على بيانات تحمل أسماء شبيه بأسماء أوامر لغة أل SQL ولذلك توجب علينا وضعها بين الرمزين ' ' . (إلى الاخوة الكرام ، أنا لا اعرف الترجمة العربية للرمزين السابقين ، لذلك ليتكرم من يعرف بإرسال الأسماء العربية الممكن استخدامها لهم لأتمكن من إضافته إلى النسخة القادمة من الكتاب).

و القيم هي مثل أن نطلب منه إيجاد مدينة معينة ، فاسم المدينة هي قيمة ، كما أن راتب موظف معين هي قيمة أيضا ، حيث ببساطة القيم هي محتويات الخلايا في الجداول ☺.

شيء آخر ، هناك قيم نصية كما ذكرنا و قيم رقمية ، لو وضعنا القيم الرقمية بين الرمزين ' ' فسيعتقد النظام على انهم نصوص ، وبذلك سيعاملهم معاملة أخرى ، حيث أن الرموز مثل اكبر من أو أصغر من لن تعمل معهم بصورة صحيحة ، ذلك إن رضيت أن تعمل معهم من الأصل.

الاستعلام المشروط المركب (الأوامر المنطقية)

كثيراً ما قد نطلب أمور أكثر تفصيلا من طلباتنا السابقة ، فعلى سبيل المثال قد نرغب بان نعرف جميع الموظفين الذين تزيد رواتبهم عن ألف وخمسمائة دولار ، وعنوانهم في المدينة رقم واحد.

للقيام بذلك علينا استخدام استعلام مركب ، و الاستعلام المركب هو استعلام يحتوي على المعاملات المنطقية ، وحتى لا اعقد الموضوع فيمكنك ببساطة استخدام المعاملات AND و OR مع استعلامك ، بالطبع هناك مجموع كبيرة من المعاملات ستجدها في ملحقات الكتاب في المستقبل ، ولكنها جميعها تعمل بنفس الطريقة.

حيث ببساطة يمكننا القول SELECT لجميع الأعمدة في الجدول EmpTABLE عندما يكون راتب الموظف اكبر أو يساوي 1500 دولار و عنوانه هي المدينة رقم واحد ، الطريقة بسيطة للغاية أليس كذلك ؟ ☺ ، إليك الكود بلغة SQL كالتالي.

```
SELECT *
FROM EmpTABLE
WHERE SALARY >= 1500 AND CITY = 'City 1'
```

الآن بالنسبة لعمل المعاملات المنطقية السابقة AND و OR فقد يتغلب البعض في استخداماتها في الاستعلامات المعقدة ، وذلك في حال انه لم يفهم طريقة عملها بالضبط.

طريقة عمل AND بسيطة للغاية ، فهو سيرجع بيانات شرط أن تتحقق جميع العبارات المنطقية ، أي انه إذا صدف بان هناك عشرة موظفين مثلا يعيشون في المدينة الأولي و لكن رواتبهم لا تزيد أو تساوي ألف وخمسمائة دولار ، فلن ترجع أية سجلات ، نظراً لعدم توفر طلبنا.

طريقة عمل OR بسيطة للغاية أيضاً ، فهو سيرجع بيانات شرط أن تتحقق أحد العبارات المنطقية ، أي انه إذا صدف بان هناك عشرة موظفين مثلا يعيشون في المدينة الأولي و لكن رواتبهم لا تزيد أو تساوي ألف وخمسمائة دولار (وذلك إن استخدمنا OR بدلاً من AND في المثال السابق)
فسيأتي بأسماء العشرة موظفين من المدينة الأولى و جميع الموظفين الذين تساوي أو تزيد رواتبهم عن ألف وخمسمائة دولار من كل المدن ! ، وهذا منطقي لأننا طلبنا جميع الموظفين الذين يعيشون في المدينة رقم واحد أو رواتبهم تزيد عن ألف وخمسمائة دولار (حيث لم نحدد مدينة معينة للفئة الثانية).

كما يمكننا دمج أوامر AND و OR معاً في استعلام واحد ، ولكنه يفضل استخدام الأقواس في هذه الحالة ، حيث سينفذ النظام ما بين الأقواس الداخلية أولاً و ينتقل بصورة تدريجية إلى الأقواس الخارجية و هكذا.

واليكم المثال التالي حيث سنطلب بالأمر SELECT من جدول أل EmpTABLE لكل الموظفين من المدينة رقم واحد والذين يتقاضون راتب أكبر من ألف وخمسمائة دولار ، أو يتقاضون علاوات أكبر من 100 دولار)

```
SELECT *
FROM EmpTABLE
WHERE CITY = 'City 1' AND (SALARY > 1500 OR BENEFITS > 100)
```

و النتيجة بأنه سيأتي بكل الموظفين في المدينة رقم واحد فقط ، الذين يتقاضون راتب أكبر من 1500 دولار مهما كانت علاواتهم ، و الموظفين من نفس المدينة والذين يتقاضون علاوات اكبر من مائة دولار مهما كان راتبهم ، ظريف أليس كذلك ☺ ، حسناً أن لم يكن ظريف فيما الكفاية فيجب أن تتمرن قليلاً.

هناك أيضا المعامل NOT و المتخصص بالنفي ، و يمكن استخدامه مع AND و OR وغيرهما ، مثل NOT AND و NOT OR الذي يتفهم طريقة عملة في الصفحات اللاحقة.

الحسابات المنطقية في الاستعلامات

عندما نستخدم المعاملات المختلفة مثل AND و OR فنحن نميزها منطقيا ، ولكن الكومبيوتر يميزها حسابيا ، و تتم الحسابات لديه على شكل True و False أي صح و خطاء ، فان كانت نتيجة الشروط بعد Where صحيحة فسيأتي الكومبيوتر ببيانات ، وان كانت النتيجة خاطئة فلن يأتي بأية بيانات.

لا أتكلم هنا عن خطئ في كتابة الجمل ، أتكلم في الحسابات المنطقية التي يجريها الكومبيوتر لحظة تنفيذ الأمر المساعد Where في الأمر Select ، وذلك حتى لا يختلط الأمر على الكثيرين.

و تقوم الحسابات بالصورة التالية

True AND True = True
True AND False = False
False AND True = False
False AND False = False

True OR True = True
True OR False = True
False OR True = True
False OR False = False

NOT True = False
NOT False = True

فلو رجعنا مثلا إلى أحد الشروط التي استخدمنها سابقا مثل

```
WHERE CITY = 'City 1' AND (SALARY > 1500 OR BENEFITS > 100)
```

فسيحللها الكومبيوتر كالتالي

```
CITY = 'City 1' AND (SALARY > 1500 OR BENEFITS > 100)
1) TRUE AND (TRUE OR TRUE)
2) TRUE AND TRUE
3) TRUE
```

حيث أن نتيجة CITY تساوي المدينة رقم واحد تعتبر صحيحة ، لأنه وجد عدة سجلات تحتوي على المدينة رقم واحد ، وهكذا بالنسبة للسجلات الأخرى ، وبعد أن استبدل جميعها بالعبارات True و False قام بعملية الجمع والطرح المنطقي و من ثم أتت النتيجة.

و تذكر دائماً بأنه لو صدفت النتيجة النهائية للاستعلام False فهذا يعني انه لن يتم إحضار أية سجلات ، وهذا منطقي أيضا حيث إن طلبنا غير موجود.

استخدام المعاملات In و Between

استخدام المعاملات المنطقية بداخل الاستعلام مثل اكبر من أو اصغر من أو يساوي أمر جيد للغاية، ولكننا أحيانا بحاجة إلى مقارنة عمود مع عدة قيم، فعلى سبيل المثال أريد جميع الموظفين الموجدين في المدينة رقم 1 و رقم 2 و رقم 3، بالطبع يمكنني استخدام OR للقيام بذلك، ولكن هناك طريقة أفضل و هي IN.

و يقوم المعمل المساعد IN بمقارنة المدينة مثلا، مع قائمة بمجموعة مدن بها الموظف، ويرجع لنا جميع الموظفين في الذين ينتمون إلى تلك المدن، ويمكن اعتباره ك OR متعددة أيضا.

إليكم المثال التالي

```
SELECT *
FROM EmpTABLE
WHERE CITY IN (City 1’, City 2’, City 3’)
```

فهو ببساطة يطلب بان يأتي لنا النظام ببيانات جميع الموظفين الموجودين في المدينة رقم 1، و المدينة رقم 2، والمدينة رقم 3، وان دققنا قليلا في الأمر، فهذا يغنينا عن ثلاثة معاملاتOR إن أردنا باستخدام رمز المساواة المشروح سابقاً.

و هناك المعامل المساعد الآخر Between و هو رائع مع الأرقام، فبدلا بان نستخدم معاملات الأكبر من والأصغر من عندما نطلب جميع الموظفين الذين تتراوح رواتبهم بين ألف دولار و ألفين دولار على سبيل المثال، يمكننا ببساطة استخدام المعامل المساعد Between لتسهيل الأمر و القيام بالمهمة.

إليكم المثال التالي

```
SELECT *
FROM EmpTABLE
WHERE SALARY BETWEEN 1000 AND 2000
```

و نتيجته هي إحضار بيانات جميع الموظفين الذين تتراوح رواتبهم بين الألف و الألفي دولار

بالطبع يمكننا استخدام المعاملات المنطقية السابقة مثل AND و OR و NOT مع IN و BETWEEN، فعلى سبيل المثال ماذا لو رغبنا بان يحضر لنا النظام بيانات جميع الموظفين الغير ساكنين في المدينة رقم واحد و المدينة رقم اثنين مثلا، أو الموظفين الذين لا تتراوح رواتبهم بين ألف و خمسمائة و الألفين دولار.

بالطبع يمكننا استخدام معامل النفي المنطقي بالصورة التالية، إليكم المثالين التاليين

المثال الأول

```
SELECT *
FROM EmpTABLE
WHERE CITY NOT IN (City 1’, City 2’)
```

و نتيجته هي إحضار جميع الموظفين الساكنين في كل المدن ما عد المدينتين رقم واحد واثنين

المثال الثاني

```
SELECT *
FROM EmpTABLE
WHERE SALARY NOT BETWEEN 1500 AND 2000
```

و نتيجته إحضار بيانات جميع الموظفين الذين لا تتراوح رواتبهم بين الألف وخمسمائة و الألفي دولار.

استخدام المعامل LIKE

أحيانا لا تكفينا عملية البحث في حقول النص على مطابقة الحقل بالكامل بالنص الذي نبحث عنه، فعلى سبيل المثال عندما نطلب من النظام أن يأتي لنا ببيانات الموظفين في المدينة رقم واحد مثلا باستخدام معامل المساواة، مثلا

```
WHERE CITY = 'City 1'
```

فهو سيقوم بمطابقة اسم المدينة الذي أعطيناه إياه باسم المدينة الموجود في الجدول بصورة كاملة، و في حال وجود أية فراغ في اسم المدينة في الجدول، أو أن تم كتابة اسم المدينة في الجدول بصورة مختلفة قليلا، فلن يقم النظام بالتعرف عليها.

بالطبع مع أسماء المدن يجب أن نحذر قليلا عن تصميم قاعدة البيانات، ولكن الموضوع اعقد مع أسماء الأشخاص، فقد يحتوي الحقل الخاص بالاسم في الجدول على الاسم رباعي و مكتوب بطريقة ما، و إن رغبنا بان يبحث لنا النظام عن هذا الموظف بالطريقة السابقة، فإما أن نكتب اسمه كما سجل في قاعدة البيانات بالضبط، أو انه لن يتمكن من إيجاده.

و من هنا يأتي دور المعامل LIKE و الذي يعطينا الإمكانية بالبحث في جزء من الخلية التي تحتوي على نصوص في العامود المخصص للاسم مثلا.

فقد لا نتذكر الأسماء الرباعية للموظفين، بل نتذكر اسم العائلة فقط، أو الاسم الأول و اسم العائلة، أو أجزاء من الاسم الأول، و ما إلى ذلك.

إليكم المثال التالي

```
SELECT *
FROM EmpTABLE
WHERE NAME LIKE 'M%'
```

نتيجة الكود السابق هو إعطائنا جميع الموظفين الذين يبدأ اسمهم بالحرف N وكما سوف تلاحظوا فلقد استخدمنا معامل جديد مع LIKE و هو رمز %، ويعني "أية أحرف"

فعند وضعه بعد الحرف أو مجموعة الأحرف التي نرغب بالبحث عنها فهو يبحث عن جميع الكلمات التي تبدأ بالأحرف السابق ذكرها و تنتهي بأية أحرف أخرى، وعند وضعه قبل الكلمة فهو يأتي بجميع الكلمات التي تبدأ بأية أحرف وتنتهي بالأحرف المحددة.

و هذه أمثلة على ذلك

```
WHERE NAME LIKE 'Mo%'
WHERE NAME LIKE '%KR'
WHERE NAME LIKE 'M%N'
```

السطر الأول سيأتي بجميع الأسماء التي تبدأ بالحرفين Mo و تنتهي أية أحرف، السطر الثاني يأتي بجميع الكلمات التي تنتهي بالأحرف KR مهما كانت بدايتها، والسطر الثالث سيأتي بجميع الكلمات التي تبدأ بالحرف M و تنتهي بالحرف N.

بالطبع يمكننا استخدام المعاملات المنطقية مثل NOT لنكون NOT LIKE حتى يأتي لنا النظام بجميع الأسماء ما اعد الأسماء المذكورة بعد LIKE أي انه يقلب العملية.

بالطبع هناك معاملات و رموز أخرى كثيرة يمكنها أن تستخدم مع LIKE و لمعرفتها يمكنكم الرجوع إلى كتاب أل TSQL الذي يركب مباشرة مع تركيب Microsoft SQL Server في جهازكم.

عملية ربط الجداول معاً JOIN

بالطبع البحث بداخل جدول وحد عملية ممتعة و مفيدة للغاية ولكنها في معظم الأحيان غير كافية على الإطلاق، ففي اصغر برمجيات قواعد البيانات نضطر إلى البحث بداخل مجموعة من الجداول المترابطة بعلاقات، بحيث نقوم بتوحيدها معا عن طريق أوامر أل SQL.

فعلى سبيل المثال قد نقوم بحفظ بيانات الموظف في جدول، وحركة الرواتب الخاصة به في جدول آخر، و لكننا عندما نرغب بطباعة تقرير عن موظف ما نكون بحاجة إلى استرجاع بياناته و بيانات كل الرواتب التي حصل عليها في مدة زمنية ما، و هنا بالضبط يأتي الأمر JOIN الذي يسمح لنا بدمج الجدولين معا و الحصول على النتيجة في جدول واحد.

و قبل أن أكمل الشرح لنأخذ الجداول التالية لاستخدامهم في أمثلتنا القادمة

جدول الموظفين

Employee KEY	Name	Address	City
1	Employee 1	Address 1	City 1
2	Employee 2	Address 2	City 1
3	Employee 3	Address 3	City 2
4	Employee 4	Address 4	City 1
5	Employee 5	Address 5	City 2
6	Employee 6	Address 6	City 2
7	Employee 7	Address 7	City 1

و يحتوي على رقم الموظف الداخلي (و هو مفاتح للربط بين الجداول)، اسم الموظف، و العنوان، و المدينة.

جدول الرواتب

Salary KEY	Employee KEY	Date	Salary
1	1	1/1/2001	1000 $
2	1	1/2/2001	1000 $
3	3	1/1/2001	1700 $
4	3	1/2/2001	1700 $
5	3	1/3/2001	1700 $
6	7	1/1/2001	2000 $
7	7	1/2/2001	2000 $

و يحتوي على رقم الحركة SALARY KEY و رقم الموظف الذي يتبع له الراتب، تاريخ صرف الراتب، والمبلغ الذي تم صرفه.

المفاتيح الرئيسية و المفاتيح التابعة (الربط بين الجداول)

مفاتيح الربط بين الجداول هي الأساس لأية عملية استعلام على قاعدة البيانات، فبدون وجود مفاتيح تربط بين تلك الجداول معدة مسبقا عن طريق العلاقات، فلن يمكننا القيام بأية عملية استعلام.

و تحدد العلاقات بين الجداول عند تصميم قاعدة البيانات، ولكنني رغبت بتذكيركم قليلا عنها في دورتنا السريعة عن SQL لأنها الأساس في عمليات الاستعلام عن طريق الأمر JOIN.

عندما نقوم ببناء قاعدة بيانات جديدة لرواتب الموظفين مثلا، فنحن نقوم بتجميع كل البيانات التي لها علاقة مع بعضها في جدول لوحدها، مثلا نحن نضع الموظفين في جدول، رواتب الموظفين في جدول آخر (ربما نضعهم في الجدول الخاص بالموظفين أيضا)، الحركات على الراتب في جدول آخر (حيث أن الحركات تجدد شهريا ولا يمكننا دمجها مع جدول الموظفين)، ساعات حضور وغياب الموظفين والساعات الإضافية قد نضعها في جدول لوحدها، و ما إلى ذلك.

ولكن، ما هي الطريقة التي ستمكننا من الربط بين الجداول المختلفة، فجدول الموظفين مثلا واضح للغاية، فبه اسمن الموظف طبعا، ولكن جدول الحركات على الراتب به معلومات الراتب فقط، نظريا يمكننا إعادة كتابة اسم الموظف مرة أخرى في كل حركة من حركات الراتب، ولكن ذلك سيؤدي إلى مشاكل في برنامجنا المستقبلي، و سينتهي مفهوم قواعد البيانات ذات العلاقات، كما انه سيكرر اسم الموظف مئات المرات، و كما نعلم فلو تكرر اسم الموظف مرة واحدة على الأقل في أية جدول فهذا يعني بأننا في مشكلة عويصة وهناك خطاء في برنامجنا.

لذلك تم ابتكار طريقة الربط بالأرقام، و كما نسميها المفاتيح الرئيسية و المفاتيح التابعة، حيث إننا نعطي لكل موظف رقم لا يمكن تكراره في جدول الموظفين، في SQL Server يمكننا جعل الرقم ألياً، ومن ثم عندما نقوم بوضع حركة في جدول حركات الرواتب فنحن نضع رقم الموظف لندل على إنها تابعة له، ونسميها مفتاح تابع.

أي إن رقم الموظف في الجدول الرئيسي (جدول الموظفين) يسمى المفتاح الرئيسي، وفي جدول حركات الرواتب يسمى مفتاح تابع، بالطبع جدول الحركات له مفتاح رئيسي آخر لعمليات الربط بجداول أخرى تابعه له ربما، أو لاستخدامات أخرى.

و العلاقة السابقة هي واحد إلى مالا نهاية، حيث إن موظف واحد قد يكون له عدد لا نهائي من الحركات على الرواتب.

الربط البسيط بين الجداول (بدون الأمر JOIN)

و هو عمليات استدعاء البيانات من جدولين معا في جدول واحد مع تكرار البيانات من كل الجدولين، شيء شبيه بالضرب المنطقي بين الجداول

سوف أشرحه لاحقاً الآن سأركز قليا على SQL Server في صفحات الكتاب الأخرى

قيد الإعداد

أوامر SQL متنوعة

قيد الكتابة

بناء الجداول بواسطة SQL

قيد الكتابة

إضافة البيانات إلى الجداول بواسطة SQL

قيد الكتابة

حذف البيانات من الجداول بواسطة SQL

قيد الكتابة

تعديل البيانات في الجداول بواسطة SQL

قيد الكتابة

تلخيص البيانات بأوامر Group By و Having

قيد الكتابة

الاستعلامات المتداخلة

قيد الكتابة

دورة سريعة في لغة Transact SQL

قيد الإعداد

دورة سريعة في لغة MDX (Multidimensional Expressions)

قيد الإعداد

دورة سريعة في لغة Visual Basic للتطبيقات

قيد الإعداد

مواصفات ومقاييس عملية تصميم البرامج العملاقة

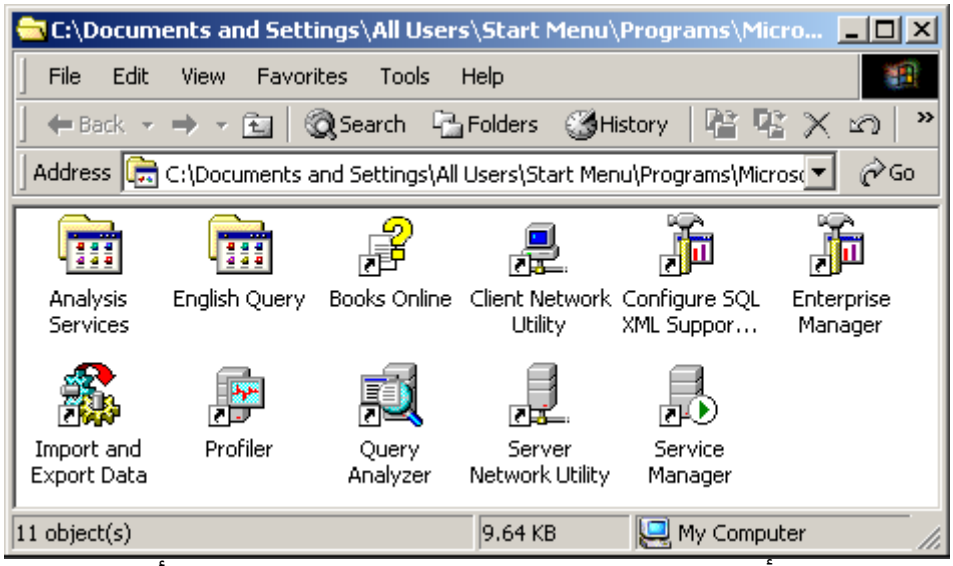
قيد الإعداد

تشغيل Microsoft SQL Server 2000 لأول مرة.

لقد تحدثت كثيراً عن مميزات خامد قواعد البيانات المركزية من مايكروسوفت، وصراحة كل ما ذكرت سابقاً هو جزء بسيط لإمكانيات هذا النظام، حيث يمكنني الجلوس والتحدث لأيام.

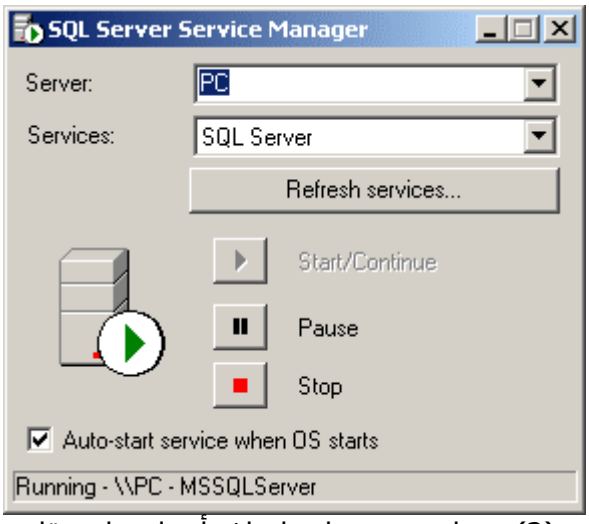
ولكن الحديث فقط لا يكفي لذلك أفضل بان نبدأ باستخدام هذا البرنامج و سأذكر المميزات التي يقدمها في كل جزء من أجزائه، كما سأتحدث لماذا تم بناء الأجزاء بالشكل التي هي عليه الآن.

بعد الانتهاء من عملية التركيب، ستظهر مجموعة جديدة من الأيقونات في الزر ابدأ، حيث تعبر هذه الأيقونات عن البرامج المختلفة التي تقوم بإدارة خادم قاعدة البيانات المركزية وذلك كما في الصورة رقم (1)



الصورة رقم (1): أيقونات برنامج SQL Server 2000 والبرامج الأخرى التابعة له

كما ستظهر أيقونة صغيرة على شريط المهام بجانب الساعة في الكومبيوتر لدي (بعد إعادة تشغيل الجهاز أول مرة) ، حيث تخبرك بحالة خادم قواعد البيانات ، كما تسمح لك بتشغيل أو إطفاء أو الإيقاف المؤقت لخادم قاعدة البيانات، بمجرد النقر المزدوج على هذه الأيقونة سيظهر برنامج بسيط يمكن استخدامه لعملية التشغيل و الإيقاف لأجزاء أل SQL Server ، كما في الصورة رقم (2)



الصورة رقم (2): برنامج تشغيل وإيقاف أجزاء خادم قاعدة البيانات

لوحة الإدارة المركزية

كما ذكرنا سابقاً فيمكنك إدارة الجهاز المركزي من داخل لوحة الإدارة الرئيسية والتي بدورها يمكنها استدعاء معظم الأيقونات الأخرى في هذه الشاشة ، كما يمكننا تنفيذ بعض العمليات لوحدها بتشغيل البرامج المخصصة لها من هنا مباشرة ، فالأمر عائد لنا.

لوحة الإدارة الرئيسية هي أل Enterprise Manager و هي عبارة عن كائن MMC و باعتبارها كذلك فهذا يعني إنها اتحدت مع البرنامج Computer Management المخصص لإدارة الأجهزة المركزية ، حيث تهدف مايكروسوفت بذلك بان توفر لك إمكانية إدارة كل أجزاء الجهاز المركزي من نقطة واحدة.

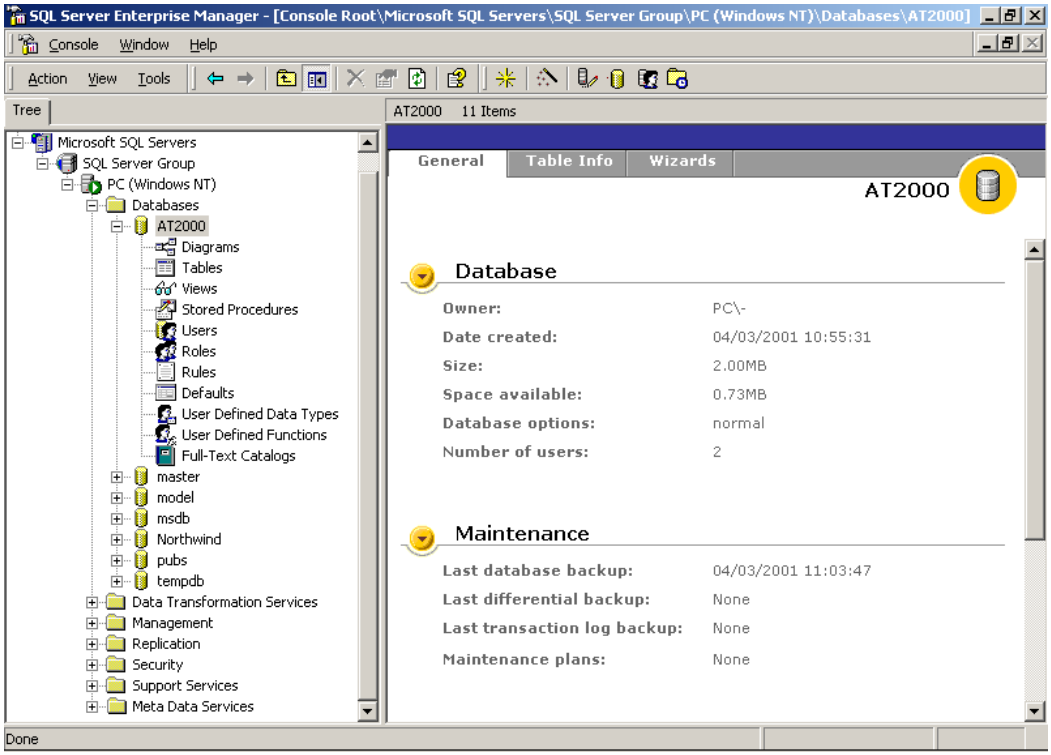
كما يمكن للإخوة المعتادين على استخدام لغة أل SQL لإدارة خادم قواعد البيانات ، مثل المعتادين على استخدام نظم Oracle يمكنهم القيام بذلك من Query Analyzer حيث بواسطته يمكنهم أرساغ كافة أوامر SQL المطلوبة لإدارة جهاز مركزي.

كما يمكن لمن اعتاد أيضا باستخدام الشاشات الحرفية ، مثل شاشات DOS يمكنه الذهاب إلى الفهرس الفرعي Bin بداخل الفهرس الخاص بال SQL Server ليستخدم بعض برامج الإدارة من خلال الشاشات النصية ، ملاحظة مهمة ، شاشات DOS كما يسميها البعض تحت وندوس ألفين أو ما بعده فقط ، هي ليست شاشات DOS حيث انه لم يعد يدير النظام ، فهي ببساطة شاشات محاكاة لأنظمة DOS ولكنها تتمتع بكامل مميزات إدارة الذاكرة والمعالجة الخاصة ب وندوس ألفين ، وهذا لا يشمل النظم السابقة مثل Windows 95, 98, ME حيث أن نظام DOS هو لا يزال المسيطر في الكثير من الحالات.

و بالنسبة لمن يعتقد بان برنامج الإدارة معقد فيمكنه أن يصمم برنامج إدارة خاص به عن طريق لغة أل Visual Basic أو أية لغة أخرى لمايكروسوفت ، حيث إنها توفر كافة الكائنات اللازمة للقيام بذلك بدون كتابة الكثير من الكود ، كما إنها توفر ما لا يقل على عشرة آلاف صفحة مساعدة إلكترونية لمن يرغب بذلك.

و بخصوص الكتب الإلكترونية، لا توفر مايكروسوفت الكثير من الكتب المطبوعة لحظة شراء المنتج، حيث انك تحصل على كتاب يحتوي على خمسمائة صفحة من الشرح السريع، وباقي الكتب تركب إلكترونيا مع النظام، و ستجدها بين الأيقونات بالاسم Books Online.

لنشغل لوحة الإدارة المركزية Enterprise Manager لتظهر كما في الصورة رقم (3)



الصورة رقم (3) : لوحة الإدارة المركزية

نظراً لان اللوحة كائن MMC اختصار ل Microsoft Management Console فهي تحتوي على قائمتين رئيسيتين ، تربك البعض في بادئ الأمر ، القائمة العليا و مخصصة للتحكم بكائنات MMC و لا علاقة لها بعملية إدارة أل SQL Server نفسه ، حيث يمكن استخدامها لربط مجموعة برامج معاً و أمور MMC الخاصة ، والتي هي خارجة عن مجالنا.

أما اللوحة نفسها فتنقسم إلى ثلاثة أجزاء رئيسية ، من الأعلى القائمة الرئيسية (ثاني قائمة وهي المخصصة ل SQL Server و سأسميها في هذا الكتاب بالقائمة الرئيسية) ، من الجهة اليسار شجرة الكائنات و التي تحتوي على كل الكائنات الخاصة بخادم قاعدة البيانات ، ومن الجهة اليمين تعرض تفاصيل و محتويات كل كائن من الكائنات السابقة لحظة الضغط عليه.

كما إن هناك قائمة تظهر لحظة الضغط على أية كائن بالزر الثاني للفأرة ، و تتغير محتويات تلك القائمة بحسب الكائن نفسه ، حيث إنها تعرض أهم العمليات التي يمكن إجرائها على هذا الكائن ، وبهذا تغنينا عن البحث بداخل القوائم المختلفة.

كما إن القائمة الرئيسية تتغير أيضاً حسب الكائن الذي تم اختياره ، فهي تعرض بالطبع أشياء ثابتة لا تتغير ، وفي معظم الأحيان تغطى الأشياء الثابتة كافة الكائنات في الشجرة ، كل ذلك يساعد الكثيرون في استخدام البرنامج و ذلك حتى لا يختلط عليهم الأمر أن تم وضع جميع الأمور في قائمة واحدة معاً.

تنقسم الشجرة إلى مستويات متعددة ، فالمستويات العليا مخصصة لتصنيف الأجهزة المركزية التي نرغب بالتحكم بها في مجموعات مختلفة ، وفي حالتنا هذه هناك جهاز مركزي واحد أو خادم قواعد بيانات واحد و هو جهازنا.

مباشرة تحت ذلك التقسيم يأتي خادم قاعدة البيانات والذي في معظم الأحيان يحمل اسم الجهاز نفسه ، ويحتوي على الأجزاء الرئيسية التالية:

- Databases
- Data Transformation Services
- Management
- Replication
- Security
- Support Services
- Meta Data Services

سنهتم في الأجزاء اللاحقة من كتابنا بالقسم الأول بصورة رئيسية ، وبالطبع سنهتم في المراحل المتقدمة بالأقسام الأخرى عندما يأتي دورها ، ومنها كيف ننقل البيانات بين الأجهزة المختلفة على الشبكة ، و إدارة الصالحات والمستخدمين ، وإرسال بريد إلى الجهاز المركزي ، و أمور أخرى.

يلي شرح للأجزاء الرئيسية:

قواعد البيانات (Databases)

و هو القسم الذي يحتوي على جميع قواعد البيانات الخاصة بالجهاز المركزي ، وهي منظمة بصورة سهلة و مرئية ، مما يجعل التعامل معها في غاية البساطة ، و هذا أيضا ما يجعلها تختلف عن قواعد البيانات الأخرى المنافسة ، لا أريد أن اذكر أسماء ولكن أكثر الأشياء الغير واضحة في محركات قواعد البيانات الأخرى هو طريقة تقسيم قواعد البيانات و أماكن حفظ الملفات.

يعتمد SQL Server عن فصل قواعد البيانات عن بعضها البعض و تخزين بيانات كل منها في ملفات منفصلة، ويمكننا أن نعرف مكان الملفت بالنقر المزدوج على ملف قاعدة البيانات ، أو بالذهاب إلى خصائصها.

هناك أيضا أربعة قواعد بيانات رئيسية في النظام وهي مهمة للغاية، و هي master المسئولة عن كل إعدادات النظام، و tempdb التي تقدم مساحة مؤقتة لتنفيذ الاستعلامات ، و model حيث تعتبر القالب الذي يكون منه النظام قواعد البيانات الجديدة ، و msdb حيث يخزن بها أل SQL Server بيانات مواعيد النسخ الاحتياطي ، مواعيد إصلاح قواعد البيانات و أمور أخرى.

قواعد البيانات السابقة مهمة للغاية ، حيث إن أية تعديل غير مسئول في قاعدة البيانات master مثلاً قد يدمر جميع قواعد البيانات الأخرى ، ولذلك علينا بالنسخ الاحتياطي لقاعدتي البيانات msdb و master عندما نقوم بعمليات النسخ الاحتياطي لقواعد البيانات الأخرى.

أنا صراحة لم استوعب مبدأ وجود قاعدة بيانات ليخزن بها النظام إعداداته ، حيث إنها يجب أن تعتبر مكانا لتخزين البيانات ، ولكنه مع المدة لاحظت بان لغة SQL يجب أن تكون قادرة على إدارة الجهاز المركزي بصورة مطلقة ، و بذلك فالطريقة الوحيدة لإنجاز ذلك هو بناء قاعدتي البياناتmaster و msdb وهي طريقة ذكية للغاية و مبدئها بسيط للغاية.

تحتوي قواعد البيانات السابقة على جداول متنوعة حيث تحتوي تلك الجداول على كل إعدادات قواعد البيانات الأخرى ، و إعدادات الجهاز المركزي ، و أسماء المستخدمين ، و أسماء الجداول في كل قاعدة بيانات ، و أسماء كل الكائنات و ما إلى ذلك.

ويراقب الجهاز المركزي قاعدتي البيانات السابقة و محتويات جداولها ، حيث إن قمت مثلا بإنشاء جدول جديد في قاعدة بيانات ما ، فسيقوم النظام بإضافة اسم الجدول الجديد كسجل في الجدول المسئول ف قاعدة بيانات master الذي يحتوي على أسماء الجداول ، وان حذفت الجدول فسيقوم النظام بحذف اسمه من قاعدة بيانات master.

الفكرة رائعة حيث بعد ذلك بعبارة SELECT يمكنك معرفة لأسماء كل الجداول و أسماء كل قواعد البيانات و هكذا ، و أكثر من ذلك ، فبحذف اسم الجدول من الجدول المسئول فسيحذف فعلياً من قاعدة البيانات ، وهكذا.

و بتلك الطريقة يمكنك إدارة SQL Server بصورة كاملة فقط بتغير البيانات في قاعدة البياناتmaster و قاعدة البيانات msdb و ذلك من برامجكم مباشرةً، فكرة رائعة أليس كذلك.

تبادل البيانات عبر النظم المختلفة (Data Transformation Services)

تبادل البيانات المدمج في نظام Microsoft SQL Server 7.0 and 2000 ، حيث انه احد أفضل الإضافات التي تميز هذا النظام عن جميع النظم الأخرى في السوق، صراحة فاجئنني مايكروسوفت كثيرا عندما رأيته ضمن حزمة SQL Server 7.0 لأول مرة، وهو حقا من الأشياء الرائعة.

إمكانياته كبيرة للغاية، وهو يختلف كثيرا عن نظام تناسخ البيانات المدمج في SQL Server أيضا، مع أن البعض قد لا يجد فرقا بين النظامين لأول مرة، ولكن الفرق يصبح واضح للغاية بعد الاستخدام الأول ل DTS.

استخداماته كبيرة للغاية، فهو يستخدم لتحضير البيانات لقواعد البيانات المجسمة (كممر بين النظامين)، كما يستخدم أيضا لنقل البيانات و نسخها بين النظم المختلفة، كما يستخدم للعديد من الأشياء الأخرى، أي ببساطة يمكننا استخدامه مع أية عملية تحتاج إلى نقل بيانات من نقطة إلى أخرى.

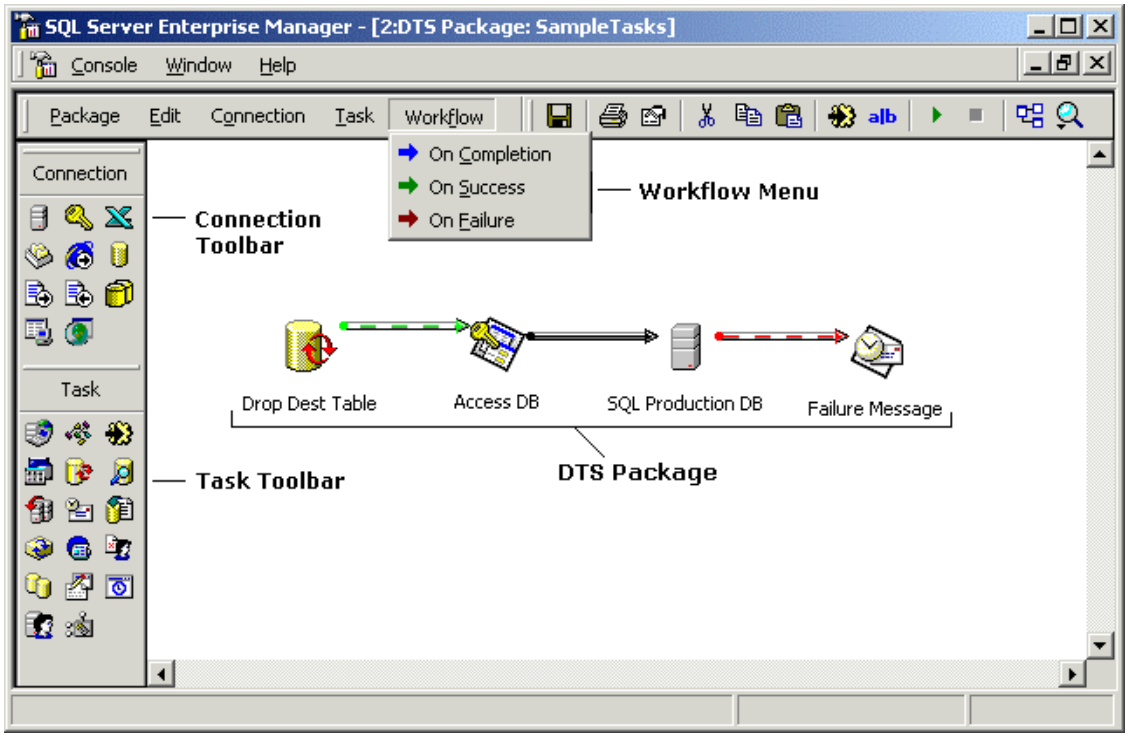
تخيل مثلا بأنك تقوم باستبدال نظام قواعد البيانات الحالي في شركة ما، أو انك تقوم باستبدال برنامج ما في شركة، أو عملية من هذا النوع.

استبدال البرامج و تحديثها من نظام إلى آخر (إلى SQL Server) قد لا تكون بالعملية اليسيرة، خصوصا إن كانت قاعدة البيانات القديمة التي يستخدمها البرنامج السابق كبيرة للغاية، بالطبع الاستبدال ليس إيقاف البرنامج القديم و نسخ البيانات و تشغيل البرنامج الجديد، ذلك لا يستطيع أن يحلم به حتى أفضل خبراء البرمجة، فعملية استبدال برنامج كبير عملية بطيئة وتحتاج اشهر، حيث أننا نضطر إلى تشغيل النظامين معا، ونسخ البيانات من النظام القديم إلى النظام الجديد عدة مرات خلال عملية النقل، وما إلى ذلك.

و هنا يأتي دور DTS و الذي يوفر لنا واجهة رسومية جذابة وسهلة للغاية للقيام بكل ذلك، بل حتى أكثر من ذلك، فهو يوفر لنا إمكانية معالجة البيانات عند نقلها، والنقل يتم في اتجاهين (تصدير و استيراد للبيانات)، كما يسمح لنا بمعالجة البيانات أثناء مرورها من النظام القديم إلى النظام الجديد أيضاً.

حيث انه من الصعب جدا بان تكون قاعدتي البيانات متطابقة بين النظامين، فالنظم الجديدة غالبا ما تقدم توصيف محسن لقواعد البيانات، ولذلك عملية معالجة البيانات ضرورية للغاية أثناء النقل.

الصورة التالية تبين مخطط نقل بسيط للبيانات



الصورة رقم 4: مخطط نقل بيانات DTS

تحتوي مخططات أل DTS على مجموعة كبيرة من الكائنات المعدة مسبقا للقيام بعمل ما، فكل ما علينا هو أن نضع الكائن و نذهب إلى خصائصه و نعده للقيام بعملية ما، بالطبع هناك لغات برمجة بالداخل، فيمكننا استخدام Visual Basic للقيام بعملية معالجة ما على البيانات، كما إن عملية الوصول إلى البيانات قد تكون بواسطة الأقراص الصلبة أو حتى بواسطة بروتوكول FTP عن طريق انترنت ☺، نعم هذا صحيح، فهو يعطينا الإمكانية لمعالجة البيانات بين النظم المختلفة حتى لو كانت على بعد آلاف الكيلومترات و ذلك عن طريق FTP و انترنت.

بالطبع أل DTS لا يقتصر عملها على تحديث النظم القديمة ولكنها تستخدم لأية عملية نقل للبيانات بين نظم مختلفة، حتى إنها قادرة على نقل البيانات بين أجهزة SQL Server المختلفة.

و تتقسم DTS بطبيعتها في SQL Server 2000 إلى ثلاثة أجزاء رئيسية، و سنقوم بشرح تلك الأجزاء في الكتاب لاحقا.

الإدارة (Management)

ثالث أجزاء لوحة إدارة نظام SQL Server 2000 هي الجزء المسمى بالإدارة Management، و يحتوي بدوره على الكائنات المخصصة بإدارة و مراقبة و صيانة خادم قاعدة البيانات المركزية.

ففي هذا الجزء نستطيع تتبع اتصالات المستخدمين لخادم قاعدة البيانات، الطلبات التي تنفذ والتي تنتظر دورها للتنفيذ، عمليات النسخ الاحتياطي لقاعدة البيانات، العمليات المجدولة لصيانة قاعدة البيانات، وغيرها من الامور.

و تنقسم الإدارة إلى الأقسام التالية

- SQL Server Agent
- Backup
- Current Activity
- Database Maintenance Plan
- SQL Server Logs

أل SQL Server Agent وهو عبارة عن Service يعمل ضمن أل Services في لوحة التحكم و وظيفته تسهيل إدارة خادم قواعد البيانات، أنا اسميه أيضا على سبيل أمزاح فقط ب"الطيار الآلي"، فهو حقا يقوم بعمليات تستطيع أن تجعله ب "مدير قواعد البيانات الآلي"، وهو بدوره يميز بين SQL Server و بعض النظم الأخرى.

وظيفته الأساسية متابعة قائمة المهام التي نبرمجه بها، فيمكننا برمجته بفحص قاعدة البيانات كل ليلة، وضغطها، وإعادة تنظيم البيانات و ترتيبها مثل أل Defrag في نظام الملفات، و إخبارنا عن أية مشاكل فورا عن طريق رسائل التحذير، فهو قادر على إرسال رسالة تحذير على شاشة أية جهاز في الشبكة لحظة حدوث المشكلة.

كما يمكننا برمجته بالقيام بعمليات النسخ الاحتياطي كل ليلة و إخبارنا عن حالتها و من إلى ذلك، فمع وجود SQL Server Agent تستطيع أن تترك جهازك المركزي لمدة طويلة بدون مراقبة، وأنتا مرتاح بان هناك من يسهر عليه، عكس نظم كثيرة أخرى في السوق و التي يجب أن تفحصها عدة مرات يوميا لتتأكد من أنها تعمل بدون مشاكل.

أل Backup هو المكان التي تخزن به البيانات عن النسخ الاحتياطية، و روابط إلى ملفات النسخ الاحتياطية، وللتذكير فإننا نستطيع القيام بالنسخ الاحتياطي إلى قرص صلب آخر، أو شريط نسخ احتياطي، أو قرص ليزر، أو أقراص ZIP أو أية أقراص أخرى يمكنها أن تعمل ضمن Windows.

و يمكننا القيام بعملية النسخ الاحتياطي من ضمن SQL Server مباشرة على أية وسيط تخزيني ذكر سابقا و ذلك بدون الحاجة إلى برامج نسخ احتياطي أخرى، أما عملية النسخ الاحتياطي نفسها فهي متقدمة لدرجة لم يحلم بها الكثير من خبراء الكومبيوتر، حيث إننا نستطيع القيام بعملية نسخ كامل لقاعدة البيانات، أو نسخ التغيرات التي حدثت على قاعدة البيانات من آخر عملية نسخ احتياطي كاملة، أو من آخر عملية نسخ التغيرات السابقة، أو بطرق أخرى.

و نسخ التغيرات فقط مهم للغاية مع قواعد بيانات قد يصل حجمها إلى عشرات الجيجابايت، والتي يصبح نسخها الاحتياطي كابوس كبير في حال عدم توفر إمكانية نسخ التغيرات التي حدثت على قواعد البيانات فقط.

ألCurrent Activity مخصصة لمراقبة العمليات التي تحدث الآن على خادم قاعدة البيانات، حيث يمكنك أن ترى ماذا يحدث عليها في الوقت الحقيقي، ما هي الطلبات التي تنفذ و التي تنتظر تنفيذها و ما إلى ذلك.

أل Database Maintenance Plan هو المكان التي تخزن به مخططات صيانة قواعد البيانات وهي مخططات ذكرت بان المسئول علن تنفيذها هو SQL Server Agent.

أل SQL Server Logs هي المكان الذي يخزن به البيانات عن حالة خادم قاعدة البيانات طوال الفترات السابقة، مثل ساعات تشغيل محرك قاعدة البيانات، و وقت إيقافه، والمشاكل التي واجهها إذا واجه مشاكل، ورسائل الخطاء و ما إلى ذلك.

تناسخ البيانات (Replication)

رابع أجزاء لوحة إدارة خادم قواعد البيانات هي الجزء المسئول عن تناسخ البيانات بين قواعد بيانات نظم SQL Server المختلفة أو حتى مع قواعد بيانات Oracle.

تناسخ البيانات عملية ضرورية جدا في نظم قواعد البيانات الموزعة على مساحات جغرافية كبيرة، وهي تختلف قليلا عن عملية نقل البيانات التي شرحناها سابقا ب DTS في عدة أمور، حيث إن تناسخ البيانات عملية ابسط ومخصصة لتتم بين قواعد بيانات SQL Server فقط مع استثناء Oracle حيث تم إضافته لتلك العملية ولكن مع ضياع بعض المميزات نتيجة لاختلاف النظامين.

بالمناسبة، تعبير تناسخ البيانات هو من ترجمتي الخاصة، فلو وجدت لديكم ترجمة أفضل فلا تترددوا على إبلاغي على بريدي الالكتروني، كما لو وجدتم أية موقع مخصص لترجمة المصطلحات الأجنبية إلى العربية فبرجاء إخباري عنه أيضا.

هناك عدة كرق لعملية تناسخ البيانات و التي سوف اشرحها لاحقا في هذا الكتاب، ولكنني أحب أو أوضح مميزات هذا النظام.

عملية تناسخ البيانات هي عملية تبادل البيانات بين جهازين مركزيين أو مجموعة من الأجهزة المركزية، بحيث يحصل الجهازين على بيانات الآخر ويدمجها بقاعدة بياناته.

و تلك العملية ليست بسيطة إلى الإطلاق و تجبرنا على استخدام قوانين و مقاييس خاصة عند بناء برامجنا، وذلك إن رغبنا باستخدام تلك الخاصية، ولكن المميزات التي نحصل عليها من تلك الخاصية اكبر مما يمكن أن تتصوروا.

لنفترض بأننا شركة كبيرة لدينا ثلاثة بنايات في ثلاثة مدن رئيسية، ولدينا قاعدة بيانات نرغب باستخدامها في جميع أجزاء الشركة في الدن السابقة.

للقيام بذلك علينا بان نقوم بربط فروعنا الثلاثة بشبكة كومبيوتر قوية والتي في معظم الأحيان ستكلفنا باهظا، أو لشراء ثلاثة أجهزة مركزية لكل مدينة من المدن السابقة، الثلاثة أجهزة المركزية ستقوم بإنجاز جميع الطلبات التي تحدث على قاعدة البيانات في كل مدينة من المدن، فكما نعلم الاستخدامات الأساسية لقواعد البيانات في الشركات هي أساسا للاستعلام و الدراسات و التقارير، والاستعلامات المتتالية من الأجهزة المركزية بحاجة إلى شبكة قوية.

أما عملية تحديث قاعدة البيانات فهي في معظم الأحيان لا تحتاج إلى هذه الشبكات العملاقة و يمكنها أن تحدث عبر شبكات صغيرة أو حتى عبر خطوط الهاتف، وعملية تناسخ البيانات هذا ما تقوم به بالضبط، فيمكننا ربط الثلاثة أجهزة المركزية السابقة بخطوط هاتف للتبادل البيانات بينها كل ليلة ولتحدث نفسها عن طريق عملية التناسخ، و قد نربطها بخطوط مؤجرة بطيئة و رخيصة الثمن لتقوم بذلك على الفور لحظة تحديث قاعدة البيانات.

للتناسخ طرق كثيرة للغاية، واستخدامات كثيرة أخرى، فيمكننا بواسطة النسخة المصغرة من SQL Server للأجهزة العادية بان نبرمجها لان تتناسخ مع النسخة الخاصة بالجهاز المركزي، بحيث إن حدث انقطاع بين الجهاز العادي و الجهاز المركزي، تبقى نسخة من البيانات المهمة للغاية في الجهاز العادي، كما نكمل عمليات إدخال البيانات مثلا إلى الجهاز العادي بدون أن نتأثر بعملية انقطاع الشبكة، وعند رجوع الاتصال ترجع الأجهزة و تتناسخ فيما بينها وبين الجهاز المركزي.

النموذج السابق نستخدمه في شركتنا (على الأقل تجريبيا الآن) و هو بالمناسبة لا يحتاج إلى البرمجة على الإطلاق و يحدث أليا بواسطة SQL Server، مما يعطي برامجك قوة لم يحلم بها الكثيرون من قبل.

ففي البنوك مثلا، يمكنك إيداع النقود حتى ولو توقفتا الشبكة نفسها، أو دفع الفواتير المختلفة، أو من هذه الأمور، بالطبع عملية سحب النقود معقدة قليلا لان البيانات موجودة ي الجهاز المركزي و الذي بدوره توقف، ولكن لو إن الزبون معروف للبنك يمكن أن يسمح له مدير البنك بسحب المبلغ بناءا على الثقة بنهم، وهذا بالضبط ما يحدث في الوطن العربي و في الكثير من الأماكن الأخرى في العالم.

و هنا يأتي دور عملية التناسخ، حيث يحفظ جهاز الموظف في البنك كل العمليات و بمجرد أن ترجع الشبكة للعمل مرة أخرى تتناسخ تلك العمليات مع الجهاز المركزي، وهكذا لا يتعطل احد.

الحماية، صلاحيات و مستخدمين (Security)

خامس أجزاء لوحة إدارة Microsoft SQL Server 2000 هي الأمن و الحماية، وهي مسئولة عن حماية خادم قاعدة البيانات ككل، فكما ذكرت سابقا كل قاعدة بيانات لديها نظام الصلاحيات و المستخدمين الخاص بها، ولكني ذكرت أيضا بان المستخدمين بجب أن يتم تعريفهم أولاً بداخل خادم قواعد البيانات لأول مرة، وهنا هو المكان المناسب لذلك.

و يمكننا إضافة نوعين من المستخدمين، استيرادهم من نظام Windows، حيث يقوم Windows بدوره بعملية التحقق منهم ومن كلمات السر الخاصة بهم، وإمكانية إضافتهم في SQL Server مباشرة، حيث يقوم أل SQL Server بدوره من إدارتهم.

بالطبع لكل من الطريقتين السابقتين مميزاته و لكنني أفضل أن تتم عمليات الحماية من Windows و بالمناسبة هذا هو الخيار الافتراضي عندما نقوم بتثبيت النظام لأول مرة.

هنا أيضا يمكننا أن نجمع المستخدمين في مجموعات مختلفة لتسهيل عملية إدارتهم، كما يمكن أن نجمع المستخدمين من أجهزة SQL Server أخرى مرتبطة مع جهازنا عبر الشبكة.

خدمات إضافية تدعم النظام (Support Services)

هناك مجموعة من الخدمات الإضافية التي يقدمها نظام SQL Server 2000 و هي بدرها تقدم مميزات إضافية للنظام، و من اشهرها Full Text Search و الذي يفهرس حقول النصوص الكبيرة والتي طبيعيا لا يمكننا أن بحث بداخلها في قواعد البيانات الأخرى.

كما يقدم أيضا SQL Mail و هي خدمة رائعة و مبتكرة للاستفسار من خادم قواعد البيانات، حيث يمكننا إذا سافرنا إلى أية مكان في العالم الاتصال بخادم قواعد البيانات الخاص بنا عن طريق البريد الالكتروني و معاملته مثلما نعامل الأفراد العاديين، بالطبع بتحفظ فهو آلة في نهاية الأمر.

حيث يمكننا أن نرسل له رسالة برد الكتروني نطلب منه بان يزودنا بتقرير ما، أو نستفسر منه عن شيء معين، أو نخبره بتشغيل برنامج داخلي بلغة SQL أو TSQL و ما إلى ذلك، حيث يمكننا القيام بواسطته بالية عملية نرغب القيام بها ، ف Transact SQL يغطي كل طرق الإدارة الخاصة ب SQL Server.

هناك أيضا خدمة أل DTC اختصار ل Distributed Transaction Coordinator و الذي يعطينا إمكانية بناء برامج جزء منها يعمل كجزء من خادم قواعد البيانات نفسه، حتى أكثر من ذلك فباستخدام أل Transaction Server لن يشعر البرنامج بان جزءا منه تعمل بداخل أل SQL Server بل سيشعر بان خادم قواعد البيانات نفسه جزء لا يتجزءا من إضافتنا الجديدة له أو من برنامجنا.

اعرف بان ذلك يبدوا معقدا بعض الشيء ولكننا سنوضحه في صفحات كتابنا لاحقاً.

Meta Data Services

و هي الجزء الأخير من الشجرة الرئيسية في لوحة إدارة خادم قواعد بيانات مايكروسوفت وهي مخصصة لعملية نقل البيانات بين Microsoft SQL Server Analysis Services و خادم قواعد البيانات نفسه، حيث إنها توفر ممر لنقل البيانات من النظام الثنائي الأبعاد إلى النظام المتعدد الأبعاد.

إنشاء قاعدة بيانات في SQL Server لأول مرة

إنشاء قواعد البيانات في SQL Server عملية بسيطة للغاية ولا تستغرق أكثر من دقيقة في اغلب الأحيان، بالطبع ذلك يتعلق بقاعدة البيانات وحجمها، فكلما ازداد حجمها كلما ازدادت المدة اللازمة لإنشائها.

قبل أن ابدأ بالكتابة عن طرق إنشاء قواعد البيانات، ارغب بان أوضح قليلا المفهوم المستخدم في SQL Server لإنشاء قواعد البيانات، و بماذا تتميز مقابل النظم الأخرى.

التركيبة الداخلية لقاعدة البيانات

كما ذكرنا من قبل في هذا الكتاب، أحد الوظائف الأساسية لخادم قواعد البيانات هو حماية البيانات من التلف وذلك بكل وسيلة ممكنة، و تقليل المشاكل التي يمكن أن تحدث معها إلى أقصى حد ممكن، و هذا ما يقوم به نظام SQL Server 2000 بالضبط.

في SQL Server 2000 تخزن كل قاعدة بيانات في نوعين من الملفات، ملفات البيانات و ملفات الحركات، حيث أن كل قاعدة بيانات بحاجة إلى ملفين على الأقل لتعمل، (ملف واحد من كل نوع).

النوع الأول من الملفات هي الملفات الأساسية MDF و تخزن بها البيانات بصورة منظمة للغاية عن طريق خوارزميات خاصة، حتى يتمكن خادم قواعد البيانات من حفظ واسترجاع البيانات بأسرع ما يمكن.

النوع الثاني من الملفات هي ملفات الحركات LDF و هي ملفات تتابعيه، تشبه ملفات النص، و تخزن بها جميع عمليات الحفظ والتعديل التي تتم على قاعدة البيانات، ولا تستخدم تلك الملفات في عملية استرجاع البيانات.

الهدف من حفظ قاعدة البيانات في نوعين من الملفات هو حمايتها القصوى من التلف، و يقوم النظام بعملية الحماية بالطريقة التالية:

- في حالة طلب أية استعلامات من النظام يقوم النظام بالتعامل مع الملف الأساسي، حيث لا تخزن تلك الاستعلامات كحركات لأنها في الأصل لم تقم بأية عملية تعديل للبيانات.
- في حال حدثت أية عملية إضافة أو تعديل للبيانات يقو النظام بحفظ التعديل أو الإضافة كحركة في ملف الحركات و تحفظ على إنها حركة مفتوحة.
- بمجرد الانتهاء من حفظ الحركة يتم تعديل أو حفظ البيانات في ملف قاعدة البيانات الأساسي.
- بمجرد نجاح العملية السابقة يتم إغلاق الحركة في ملف الحركات.

في حال فشل العملية السابقة في أية جزء من أجزائها بسبب انهيار مفاجئ للجهاز، انقطاع التيار الكهربائي أو أية ظرف طارئ يقوم النظام عند إعادة تشغيل الجهاز مقارنة ملف الحركات بملف قاعدة البيانات الأساسي و إصلاح المشاكل التي قد تكون قد نجمت عن انهيار النظام سابقا.

في حال حدوث تلف كبير في قاعدة البيانات الأساسية يمكننا إعادة بنائها بواسطة ملف الحركات، لترجع بالضبط كما كانت في السابق.

بالطبع ملفات الحركات تكبر بصورة سريعة للغاية في قواعد البيانات العملاقة ولذلك يخصص لها أقراص صلبة خاصة بها، كما يمكننا حفظ تلك الملفات في أماكن أخرى ليست بالضرورة نفس الأماكن التي حفظنا بها ملفات قواعد البيانات الأساسية.

يمكننا إعداد النظام ليتصرف بطرق مختلفة مع ملفات الحركات حسب طبيعة نظامنا ، ولكن في معظم الأحيان يقوم النظام بإفراغ ملفات الحركات بعد عملية النسخ الاحتياطي لقاعدة البيانات، و بهذا إن حدثت أية مشكلة في

ملف قاعدة البيانات الأساسي يمكننا استخدام النسخة الاحتياطية الأخيرة و ملف الحركات لجميع الحركات الجديدة التي حدثت بعد النسخة الاحتياطية للقيام بعملية إصلاح ناجحة لقاعدة البيانات.

تعتبر تلك الطريقة من أفضل طرق حماية قاعدة البيانات على الإطلاق لأنها تخفف الأضرار إلى أقصى حد ممكن، وهي جيدة للاستخدام في المؤسسات الحساسة و التي تحسب ألف حساب لكل سجل في كل جدول، مثل البنوك، والمؤسسات المالية، وشركات الطيران، و غيرها.

لملف الحركات أيضا استخدامات كثيرة أخرى، فتخيل بأنك وظفت مجموعة مدخلي بيانات و بعد أن قاموا بإدخال الكثير من البيانات، لنفترض لمدة ساعتين من الساعة العاشرة حتى الثانية عشرة مثلاً، وتخيل بان مدخلي البيانات أتلفوا جزء كبير من البيانات نظرا لضعف خبرتهم في قاعدة البيانات الحالية، و هذا ما حدث معي مرة بالضبط، حيث كنت قد قمت بعملية النسخ الاحتياطي قبل يومين سابقين.

الآن لدي حل من اثنين، أن أقوم باسترجاع قاعدة البيانات من النسخة الاحتياطية من قبل يومين أو ثلاثة أيام، أو ربما أكثر ، أو أجد طريقة لإصلاح ما قد يكون مدخلي البيانات قد قاموا بإتلافه ، وهذا ما يمكن القيام به عن طريق ملف الحركات ، حيث يمكننا أن نطلب من خادم قواعد البيانات بان يلغي آخر ساعتين من التعديل الذي تم إجرائه على قاعدة البيانات ، وعند إذن يستخدم النظام ملف الحركات لإرجاع قاعدة البيانات كما كانت قبل ساعتين.

كما ذكرنا سابقا فكل قاعدة بيانات بحاجة إلى ملفين على الأقل أحدهم للحركات و الآخر للبيانات، ولكن في قواعد البيانات الكبيرة نحن بحاجة إلى أكثر من ملفين.

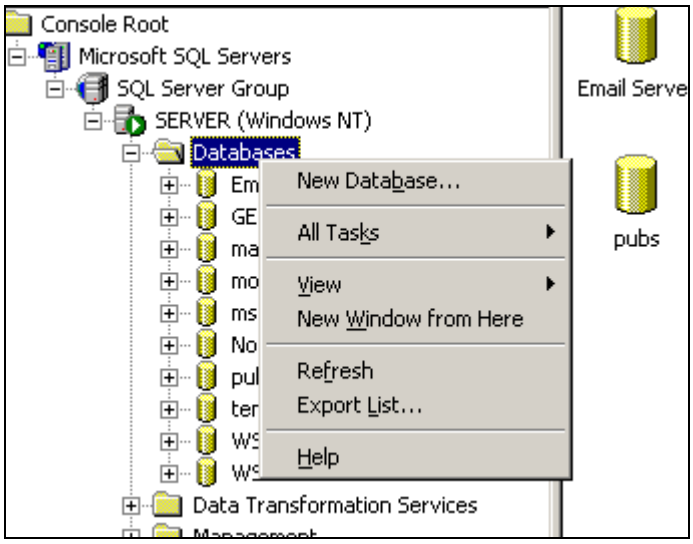
حيث إن حجم الملف الأقصى في نظام Windows يصل إلى أربعة جيجابايت، أي انه لا يمكن إنشاء ملف اكبر من ذلك (بالطبع ذلك متعلق أيضا بنظام الملفات المستخدم)، كما إن حجم الأقراص الصلبة محدود أيضاً، و كما نعلم فخادم قواعد البيانات من مايكروسوفت يمكنه العمل مع قواعد بيانات عملاقة، و لإنجاز ذلك يمكننا أن نحفظ قاعدة البيانات في أكثر من ملف أساسي واحد وذلك في نفس القرص الصلب أو في عشرات الأقراص الصلبة، ونفس الشيء بالنسبة لملفات الحركات.

ولا يشترط أن تحدد جميع الملفات منذ البداية، فيمكنك مثلا حجز ملف واحد للحركات و ملف واحد للملف الرئيسي، و من ثم عندما يزداد حجم قاعدة البيانات و تقترب من الوصول إلى الحد الأقصى من مساحة القرص الصلب، يمكنك عند إذن إضافة قرص صلب إضافي للجهاز المركزي، وإضافة ملف رئيسي آخر أو ملف حركات آخر لقاعدة البيانات على القرص الصلب الجديد حسب الحاجة، وذلك بكل بساطة وبدون تعب على الإطلاق.

إنشاء قاعدة بيانات جديدة

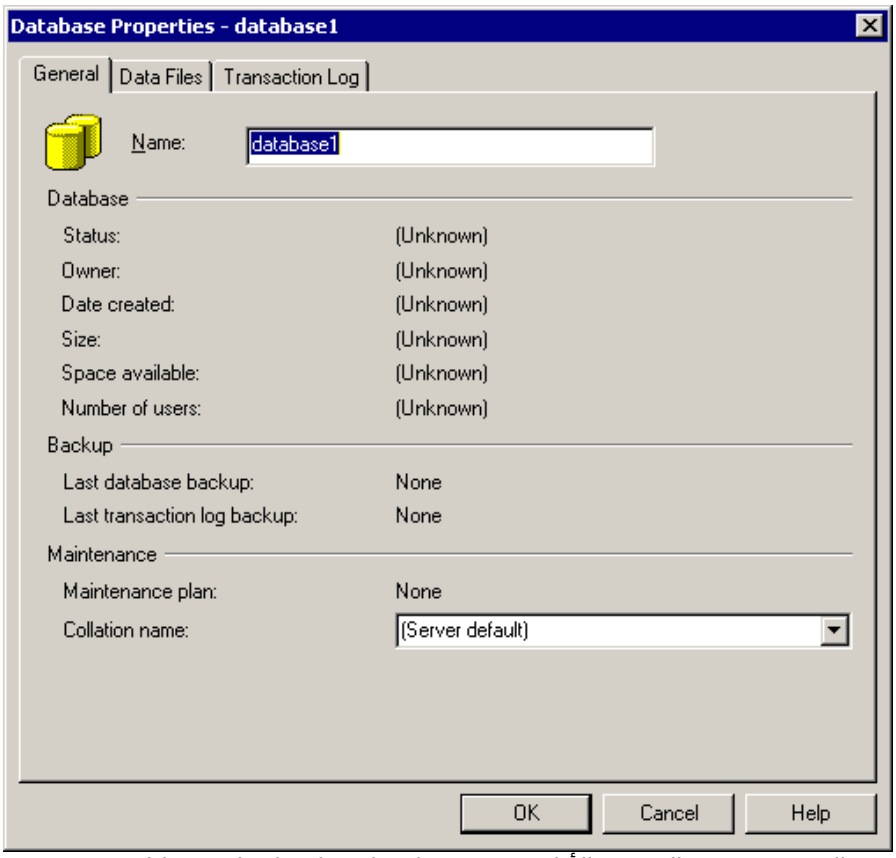
إنشاء قاعدة بيانات جديدة عملية في غاية البساطة، اتبع الخطوات التالية:

1. استخدم شجرة الكائنات في الجهة اليسار من الشاشة و اذهب إلى الفرع الرئيسي الذي يحتوي على كل قواعد البيانات و يسمى Database.
2. اضغط على Database بالزر الثاني للفائرة، حيث ستظهر قائمة بها كل العمليات التي يمكن القيام بها على هذا الكائن.
3. اختار إنشاء قاعدة بيانات جديدة New Database، كما في الصورة رقم 1.



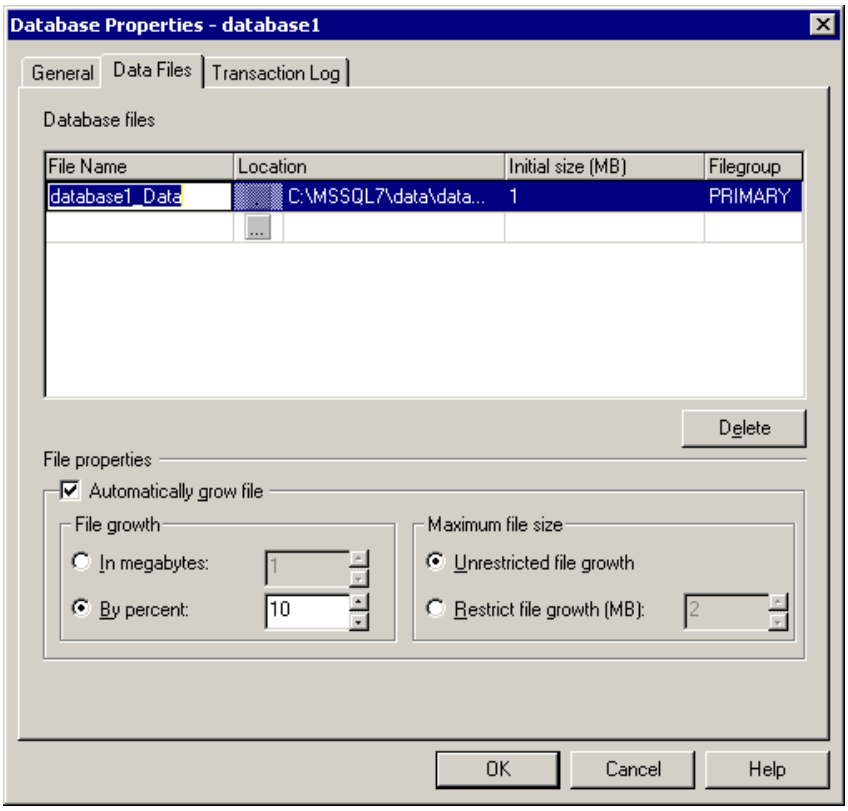
الصورة رقم 1 : إنشاء قادة بيانات جديدة

4. لتظهر لك نافذة تطلب منك البيانات الأولية المطلوبة لعملية إنشاء قاعدة البيانات.
5. ستفتح لك الشاشة رقم 2 التالية و التي تتكون من ثلاثة صفحات رئيسية، وتطلب منك الصفحة الأولى تحديد اسم قاعدة البيانات، أكتب أية اسم ترغب به، في مثالي هذا لقد استخدمت الاسم Database1.



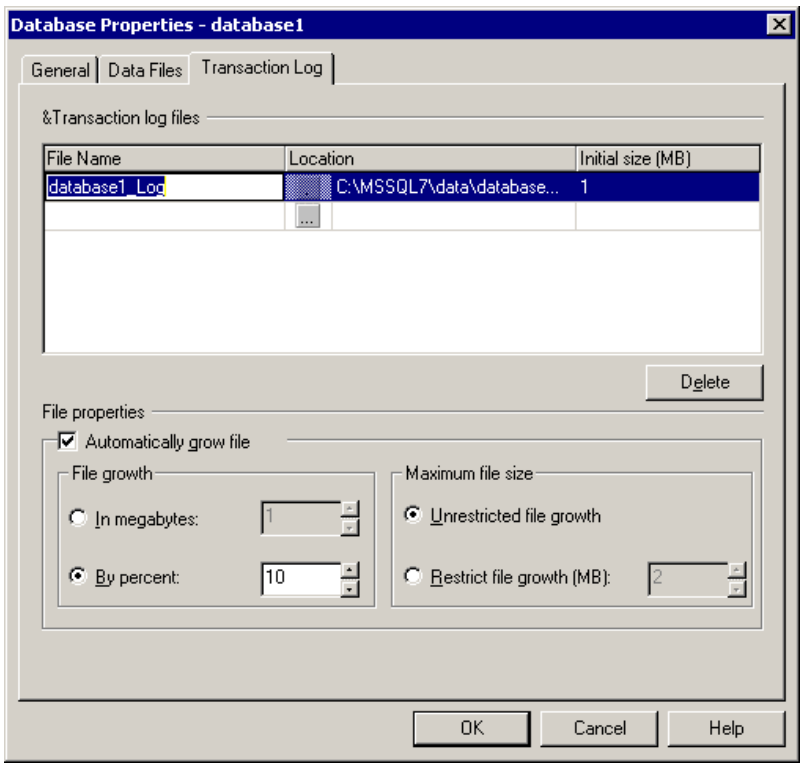
الصورة رقم 2 : الصفحة الأولى من صفحات نافذة إنشاء قاعدة بيانات جديدة

6. اترك لغة قاعدة البيانات كما هي في أسفل الشاشة حيث سنتحدث عن اللغات لاحقا في هذا الكتاب
7. انتقل إلى الصفحة الثانية و التي تظهر في الصورة رقم 3، حيث يطلب منك النظام تحديد الاسم و المكان المخصص في القرص الصلب ليخزن به ملف قاعدة البيانات الأساسي، بالطبع يمكنك اختيار ملف أو عدة ملفات لذلك.



الصورة رقم 3 : الصفحة الثانية من صفحات نافذة إنشاء قاعدة بيانات جديدة

8. يجب تحديد حجم ملف قاعدة البيانات لحظة الإنشاء ، في معظم الأحيان يحدد ب ميجابايت واحد مع تركه يزداد حسب الحاجة ولكن هناك البعض يفضل تحديد المساحة الأولية لحظة إنشاء قاعدة البيانات.
9. الجزء الأسفل من تلك النافذة مخصص لتحديد الزيادة الطبيعية للملف، حيث يمكننا إعداده ليزداد بنسبة مئوية من الحجم الذي وصل إليه أو بزيادة بمساحة ثابتة ، كما يمكننا منع زيادة حجمه عن مساحة ما نحددها كمساحة قصوى.
10. انتقل إلى الصفحة الثالثة و التي تظهر في الصورة رقم 4 ، حيث يطلب منك النظام تحديد الاسم و المكان المخصص في القرص الصلب ليخزن به ملف الحركات لقاعدة البيانات ، بالطبع يمكنك اختيار ملف أو عدة ملفات لذلك ، والباقي في هذه الشاشة كالسابق ولكنه بخصوص ملف الحركات وليس الملف الرئيسي.



الصورة رقم 4 : الصفحة الثالثة من صفحات نافذة إنشاء قاعدة بيانات جديدة

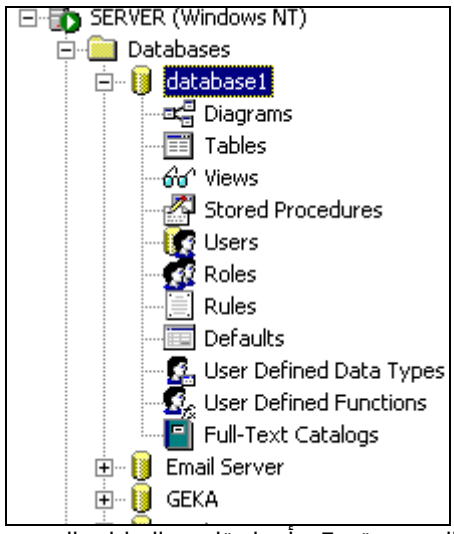
اضغط على الزر موافق، ليقوم النظام بإنشاء قاعدة البيانات الجديدة حيث ستجدها الآن بين قواعد البيانات الأخرى في شجرة الكائنات.

إن حسبت الوقت ستجد بان كل العملية لم تستغرق أكثر من دقيقة واحدة و هذا رقم خيالي عندما نقارن SQL Server بنظم قواعد البيانات المركزية الأخرى والتي يستغرق بعضها الكثير من الوقت لإنشاء قاعدة البيانات أول مرة.

الآن يمكنك التحرك بداخل الكائنات الخاصة بقاعدة البيانات الجديدة ، إنشاء الجداول و الاستعلامات و القيام بمختلف أنواع التجارب بحرية ، وان قمت بإتلافها في أحد التجارب ، فيمكنك حذفها بالنقر عليها بالزر الثاني للفائرة واختيار Delete و من ثم إنشائها من جديد ، حيث كل ذلك لن يؤثر على الجهاز أو على خادم قواعد البيانات على الإطلاق.

استكشاف قاعدة البيانات

بمجرد إنشائك لقاعدة البيانات الجديدة ستجد بأنها تتكون من إحدى عشر جزء كل منهم يحتوي على الكائنات الخاصة به ، كما في الصورة رقم 5



الصورة رقم 5 : أجزاء قاعدة البيانات الجديدة

أحب أن الفت انتباهك بأنه لو صدف ولم تجد كل الأجزاء السابقة فلا تخاف ، حيث انه قد ينقصك جزء أو جزأين في بعض الأحيان حسب النسخة المستخدمة من SQL Server الأمثلة الحالية تمت باستخدام النسخة Enterprise ولكنها ستعمل أيضا مع النسخ العادية.

الأجزاء الرئيسية لقاعدة البيانات هي كالتالي:

- مخططات العلاقات Diagrams
- الجداول Tables
- الاستعلامات البسيطة Views
- الاستعلامات المعقدة Stored Procedures
- المستخدمين Users
- مجموعات المستخدمين Roles
- قوانين التعامل مع البيانات Rules
- القيم الافتراضية Defaults
- أنواع جديدة من البيانات من صناعة المستخدم User Defined Data Types
- إجراءات جديدة مصنوعة من المستخدم User Defined Functions

- فهرس النصوص (بيانات النص الكبيرة في الجداول) Full-Text Catalog

كل جزء من الأجزاء السابقة له استخداماته و أهميته في مراحل إنشاء قاعدة البيانات، حيث أن قواعد البيانات الممتاز هي تلك التي تستغل الأجزاء السابقة بأفضل ما يمكن، و استغلالها جيدا و بصورة سليمة يجعل برامجنا أكثر سرعة وقليلة المشاكل والأعطال.

و يلي شرح لكل جزء من أجزاء قاعدة البيانات بصورة سريعة، حيث سنفصلهم لحقا في هذا الكتاب.

مخططات العلاقات Diagrams

مخططات العلاقات هو المكان الذي تخزن به الخرائط المرئية والتي تبين العلاقات بين الجداول، كما تتذكرون في بداية هذا الكتاب ذكرنا بأن العلاقات جزء أساسي من قاعدة البيانات، حيث أنها تساعدنا في وضع مختلف القواعد بين الجداول.

فعلى سبيل المثال يمكننا علاقة بين جدول الموظفين و جدول الرواتب، بحث نمنع من إضافة أية سجل في جدول الرواتب إن لم يتبع لموظف في جدول الموظفين، حيث العلاقة السابقة ستجبر النظام قبل إضافة الراتب للموظف من التحقق من صحة البيانات، وبهذا سنحافظ على البيانات مترابطة داخليا بصورة منطقية.

بالطبع هناك الكثير من المميزات الأخرى للعلاقات و التي ستشرح لاحقا، ولكن أحب أن الفت انتباهكم بان SQL Server هي قاعدة البيانات الوحيدة تقريبا التي تظهر العلاقات بين الجداول بصورة مرئية رائعة تبين لنا العلاقات بين الجداول بصورة رائعة.

الجداول Tables

الجداول هي المكان الذي تخزن به البيانات، و هي نوعين جداول النظام و جداول المستخدم، إن قمت بالضغط على قاعدة البيانات الجديدة، ستجد بأنها تحتوي على عدة جداول جاهزة، و هي جداول النظام، و يفضل أن لا تحاول استخدامها في المرحلة الأولي حيث إننا سنتطرق إلى طريق الاستفادة منها لاحقا.

كما إن لكل جدول اسم خاص به واسم الشخص الذي صنعه، حتى انه يمكننا بناء جدولين بنفس الاسم شرط أن يكون صنعوا من شخصين مختلفين، و ذلك سنشرحه في الأجزاء المتقدمة من الكتاب.

الاستعلامات البسيطة Views

الاستعلامات البسيطة هي عبارة عن أوامر SQL تقوم باسترجاع البيانات من الجداول ، وتستخدم الاستعلامات البسيطة لربط عدة جداول أو لتصفية البيانات في بعض الجداول ، أو لأمور أخرى ، كما قد لا تكون بسيطة فبإمكانها أن تكون بالغة التعقيد ، ولكننا نسميها بسيطة لأنها لا تستقبل متغيرات ، فهي دائما تأتى بالبيانات بنفس الطريقة ، أي انه يمكننا الطب من استعلام بسيط على سبيل المثال فرز الموظفين من مدينة رقم 1 لوحدهم و حفظ الاستعلام (لا تحفظ البيانات مع الاستعلام).

فكلما قمنا بتشغيل الاستعلام سيأتي بجميع الموظفين من المدينة رقم 1 فقط، ولكنه لا يمكن بناء استعلام بسيط يسألنا عن اسم المدينة قبل تنفيذه (أي يستقبل الاسم في متغير) فهذا من تخصص الاستعلامات المعقدة.

الهدف من بناء نوعين من الاستعلامات هو المساعدة في تخفيف العبء على الجهاز المركزي في البرامج الكبيرة ، فالاستعلامات البسيطة يمكن التنبؤ بنتائجها ، ففي البرامج الكبيرة قد يطلب البرنامج الموظفين في المدينة رقم 1 ألف مرة في اليوم ، فبدلا من تنفيذ العملية ألف مرة يمكن للنظام تنفيذها أول مرة فقط ومن ثم حفظ

النتيجة في الذاكرة المؤقتة و حتى فهرستها ، وان طلب جهاز آخر الاستعلام نفسه لا يجب على الجهاز المركزي عادة تنفيذ الاستعلام في حال إن بيانات الموظفين لم تتغير من وقت تنفيذ الاستعلام السابق ، بل إعطائه النتيجة المخزنة مسبقاً.

الاستعلامات المعقدة Stored Procedures

و هي عبارة عن مجموعة من أوامر SQL و Transact SQL مخصصة للقيام بعمليات ما ، و تستخدم في الكثير من الحالات ، حيث يمكننا استخدامها لفرز بيانات بناءا على متغيرات مرسلة إليها ، أو القيام بعمليات احتساب في طرف الجهاز المركزي ، أو أية أمور أخرى يتطلبها برنامجنا ، حيث أن Transact SQL هي لغة برمجة متكاملة تقريبا ، تمكننا من القيام بأية عمليات على البيانات.

المستخدمين Users

و هما المستخدمين الذين لهم الصلاحيات للوصول إلى البيانات في قاعدة البيانات، هناك مكانين لتحديد الصلاحيات في قاعدة البيانات، المكان الرئيسي وهو الذي يحدد أسماء المستخدمين الذين يمكنهم الاتصال بخادم قاعدة البيانات، و بداخل كل قاعدة بيانات حيث يحدد المستخدمين الذين يمكنهم استخدام قاعدة البيانات.

لذلك في جزء المستخدمين الخاص بقاعدة البيانات لا يمكن إنشاء مستخدمين جدد، بل إحضارهم من قائمة المستخدمين الرئيسية، وعملية إنشاء المستخدمين تتم في قائمة المستخدمين الرئيسية.

الصلاحيات يمكن تحديدها أيضا على أدق أجزاء قاعدة البيانات ، بدءاً من الجداول مرورا بالأعمدة و السجلات و وصولا إلى مستوى الخلية نفسها

مجموعات المستخدمين Roles

المجموعات هي المكان الذي يحتوي على المستخدمين مقسمين إلى مجموعات ، وذلك بهدف تسهيل عملية توزيع الصلاحيات ، فتخيل مثلا بوجود مائة موظف في قسم المبيعات و رغبت بإعطائهم صلاحيات وصول إلى بعض البيانات في قسم المبيعات ، فعملية وضع الصلاحيات لكل منهم على حدا عملية طويلة ، و لتسريع لك يمكنك إنشاء مجموعة وضمهم إليها ليحصلوا على الصلاحيات نفسها

قوانين التعامل مع بيانات Rules

و تحتوي على جميع القوانين المخصصة للتعامل مع البيانات ، هي بالمناسبة قوانين ليست الكلمة الأفضل و سأحاول إيجاد بديل لها لاحقا و لكن بصورة عامة القوانين تفيدنا لتقييد طرق إدخال و تعديل البيانات ، على سبيل المثال يمكننا إنشاء قانون لتحديد عمر الشخص ، و يمكن ربطه بجداول الموظفين و جميع الجداول التي تحتوي على أعمار للأفراد.

حيث يقوم القانون بإظهار رسالة خطاء إن قمنا بإدخال تاريخ ميلاد شخص لمائتي عام إلى الوراء مثلا ☺ ، أو انه ولد بعد عشر سنوات ☺ ، حيث سيخبرنا بان تلك البيانات خاطئة ولا يمكنه قبولها.

لأذلك قوانين إدخال البيانات مهمة للغاية في البرامج الكبيرة ، فهي تحافظ على منطقية البيانات من عمليات الإدخال أو التعديل الخاطئة التي قد يقوم بها المستخدم.

القيم الافتراضية Defaults

القيم الافتراضية هي قيم تنتج بصورة آلية بناءاً على معادلة ما ، و هي مخصصة لتعبئة أجزاء من الخلايا في الجداول والتي لم يقم مستخدم قاعدة البيانات بتعبئتها ، فمثلا يمكننا بناء قيمة افتراضية لتاريخ اليوم وربطها مع جدول ما ، فمباشرة عند إضافة سجل جديد من شاشة المستخدم سيكتب النظام التاريخ آلياً في الجزء المخصص له و بذلك سيوفر بعض الجهد لمدخل البيانات ، خصوصا إن كانت البيانات المطلوب إدخالها كثيرة.

بالطبع هناك غير التاريخ يمكن إنشاء قيم افتراضية كثيرة للغاية و يمكن ربط كل منها مع جداول مختلفة ، فالقيم الافتراضية تسهل من عملية إضافة البيانات إلى الجداول.

أنواع جديدة من البيانات من صناعة المستخدم User Defined Data Types

كما نعلم فلكل عمود من أعمدة الجدول نوع معين من البيانات ، بيانات رقمية ، نصوص ، تواريخ ، صور ، عملة ، و ما غير ذلك ، ولكن في الكثير من الأحيان يرغب المبرمجين ببناء أنواع خاصة من البيانات وذلك بناءا على الأنواع الموجودة مسبقا في النظام ، و هنا يمكنهم بناء أنواع البيانات تلك و من ثم استخدامها في الجداول.

إجراءات جديدة مصنوعة من المستخدم User Defined Functions

قيد الاعداد

فهرس النصوص (بيانات النص الكبيرة في الجداول) Full-Text Catalog

فهرس النصوص كما اسميه هو عبارة عن عملية فرز وترتيب لكل البيانات النصية في بعض الجداول بهدف البحث بها ، فكما نعلم حتى وقت طويل كان من الغير ممكن البحث في النصوص الكبيرة في قواعد البيانات ، فهناك نوعين من النصوص ، نصوص محددة الحجم مسبقا حيث تخزن بها مثل الأسماء ، العناوين ، أمور أخرى ذات طول قصير ، من غير المرغوب أن يزيد عن 100 إلى 200 حرف ، ونظرا لان طول الحقل محدد مسبقا بحد أقصى فيمكننا البحث بها و فرزها بسهولة.
و نوع آخر من النصوص ذات حجم مفتوح، أي يمكننا تخزين النصوص كما نشاء و لكننا لا يمكننا البحث بها ، ومع التطوير الجديد الذي أضيف إلى SQL Server أصبح بالإمكان البحث و فرز تلك النصوص.

عملية الفرز تتم بالضبط كما تتم عمليات الفرز التي تقوم بها محركات البحث الشهيرة مثل Yahoo وغيرها ، حيث يسجل النظام مكان كل كلمة في كتالوج الفرز لنتمكن من الوصول إليها لاحقا بصورة سريعة.

تشغيل Microsoft Access XP لأول مرة.

كما نعلم فان Microsoft Access XP هو جزء من حزمة برمجيات Microsoft Office XP و هو يتوفر أيضا لوحده كبرنامج مستقل.

بعد الانتهاء من عملية تثبيته على الجهاز ستظهر لك أيقونة هذا البرنامج في الزر Start في قسم البرامج، كما يبين ذلك الشكل رقم 1 أدناه

الشكل رقم (1): أيقونة برنامج Microsoft Access XP

بمجرد النقر عليها سيفتح البرنامج و سيكون مستعدا للعمل.

برنامج Microsoft Access XP هو عبارة عن برنامجين متداخلين معا، فهو قادر للعمل كقاعدة بيانات متكاملة و مستقلة، ذلك لعملية بناء البرامج البسيطة و التي تعمل بصورة مستقلة عن Microsoft SQL Server 2000، كما انه قادر على العمل كواجهة رسومية لنظام Microsoft SQL Server 2000، حيث يقوم ببناء الشاشات و التقارير و بعض الكود الخاص بالبرامج المعتمدة على خادم قواعد البيانات الخاص بمايكروسوفت.

الكثير من مستخدميه القدامى لا ينتبه إلى هذه النقطة في اغلب الأحيان، نظرا لان في كل من الحالتين يستخدم Access XP نفس شكل قاعدة البيانات، مع بعض الاختلافات طبعا، ونفس مفاهيم قواعد البيانات المستخدمة في الإصدارات السابقة تقريبا، وهذا يحير المبرمج الغير معتاد على ذلك بعض الشيء.

ذلك لا يعجب بعض المبرمجين أيضاً، فهناك الكثير يسألني لماذا لم تقم مايكروسوفت ببناء نظام آخر ليقوم كواجهة رسومية ل SQL Server و ذلك نتيجة للسمعة الغير جيدة ل Access بصورة عامة بين المبرمجين العرب، اعتقد بان الجواب على ذلك عندكم، وجوابي شخصيا هو (هل ستجد مايكروسوفت أقوى من Access للقيام بتلك العملية).

المشكلة لدينا في الوطن العربي بأننا ننتقد أحيانا بدون الاستناد على وثائق و دراسات رسمية و بدون معرفة كاملة في الموضوع، هذا ينطبق علي أيضا (مع إنني أحاول التخلص من ذلك).

بالطبع Microsoft Access XP ليس الواجهة الوحيدة الممكن استخدامها مع SQL Server 2000 حيث بإمكانك استخدام كل لغات البرمجة التي تنتجها مايكروسوفت مع هذا النظام، حتى صفحات الويب العادية، و إن لم ترغب باستخدام تكنولوجيا مايكروسوفت فيمكنك استخدام تكنولوجيا الشركات الأخرى، فلغات البرمجة الخاصة بشركة Borland قادرة على التعامل مع SQL Server بصورة رائعة، حتى Oracle Developer 2000 قادر على ذلك، ولكن يبقى Access XP الأكثر تجهيزا لذلك.

عندما تشغل Microsoft Access XP لأول مرة ستظهر لك النافذة الرئيسية للبرنامج و التي تنقسم إلى قسمين، مساحة العمل، ولوحة المساعدة في الجهة اليمين من الشاشة، لوحة المساعدة هي لوحة مؤقتة تظهر فقط في حالات بسيطة عندما تقوم بعملية مثل فتح للملفات، الترجمة، أو بعض الأمور الأخرى.

لن أتطرق إليها الآن، حيث أفضل أن نقوم بفتح قاعدة بيانات موجودة في الجهاز لأشرح النظام عن طريقها، لذلك عن طريق القائمة الرئيسية افتح قاعدة البيانات التالية:

C:\Program Files\Microsoft Office\Office10\Samples\NorthwindCS.adp

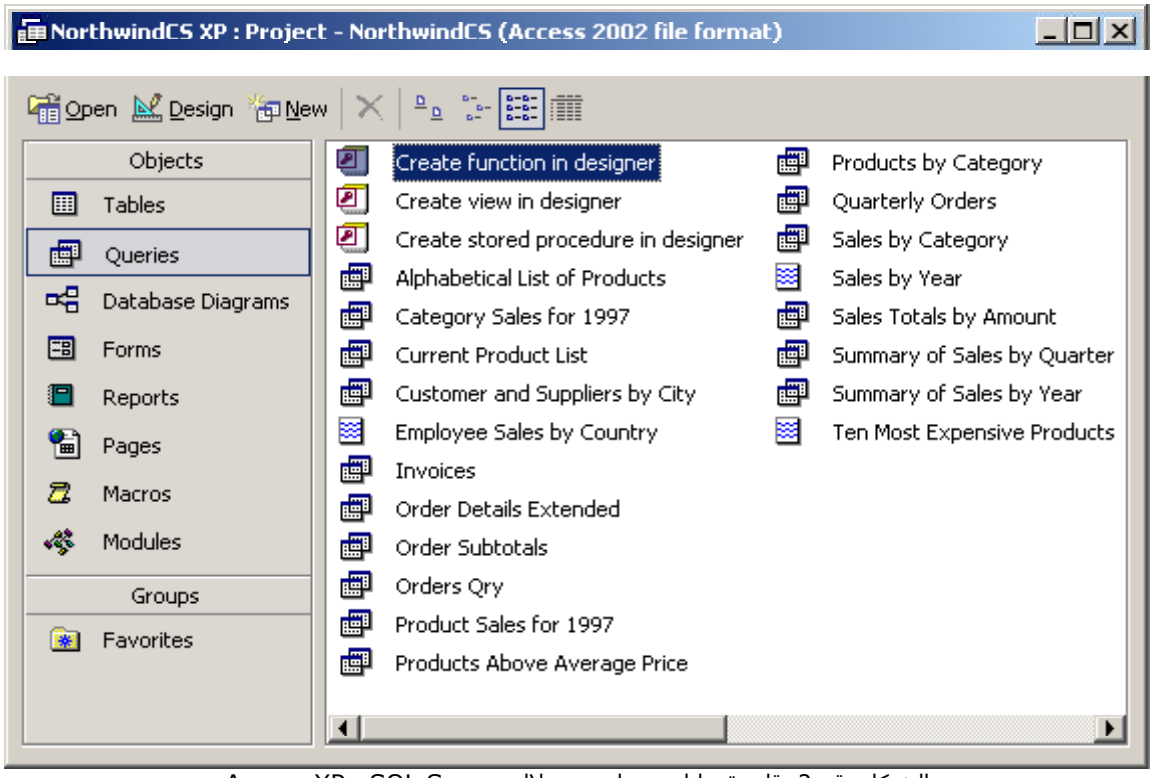
وهي مثال لقاعدة بيانات تعمل من خلال SQL Server و Microsoft Access XP، وبمجرد فتحها ستقوم ببناء الجزء الخاص ب SQL Server بصورة آلية و بدون تدخلك، كل ما في الأمر هو أن تسألك إن كنت ترغب بذلك.

و هذه العملية بالمناسبة احد أوجه القوى بين البرامج المعتمدة على SQL Server و Access XP و هي العمليات الآلية، فهي تعطيك مميزات ضخمة تنفرد بها شركة مايكروسوفت، و حتى كتابة هذا الكتاب، البرمجيات العملاقة المبنية على النظامين السابقين هي البرمجيات العملاقة الوحيدة القادرة على تركيب و إعداد نفسها بصورة آلية بمجرد النقر على أيقونة التركيب فقط.

بكلمات أخرى يمكننا بناء برنامج عملاق و وضعه على قرص ليزر، قرص مخصص للأجهزة المركزية و قرص للأجهزة العادية، أو قرص للاثنين معا، حيث يقوم هذا القرص بفحص إعدادات الجهاز لديك و تركيب SQL Server إن لزم الأمر و تركيب Microsoft Access XP Runtime إن لزم الأمر، و تركيب برنامجك طبعا، وإعداد الأنظمة لتعمل معا، وبناء قاعدة البيانات لأول مرة في طرف الجهاز المركزي، ووضع الصلاحيات و المستخدمين الافتراضيين و الكثير غير ذلك بصورة آلية.

ذلك يعطيك الإمكانية ببيع برامجك العملاقة و كأنها أية برمجيات بسيطة أخرى يستطيع أية شخص تثبيتها مثلما تركب البرامج الأخرى، فكما ذكرت سابقا مايكروسوفت هي الشركة الوحيدة التي تنفرد بذلك حتى لحظات كتابة هذا الكتاب.

لنلقي نظرة على نافذة قاعدة البيانات الجديدة، كما يبين الشكل رقم 2



الشكل رقم 2: قاعدة بيانات تعمل من خلال SQL Server و Access XP

ما سيفاجئ البعض هو احتواء شريط العنوان الخاص بقاعدة البيانات على نوع قاعدة البيانات و هو قاعدة بيانات متوافقة مع Access 2000، في الشكل السابق أنا قمت بتحويلها إلى Access 2002 XP، وتستطيعون انتم أيضا القيام بذلك.

الإعدادات الافتراضية في Microsoft Access XP هي بناء قواعد بيانات متوافقة مع Access 2000 و ذلك لان الشركات العملاقة تواجه مشاكل كبيرة مع عمليات الترقية بين النسخ المختلفة، فكما نعلم Access 95 لا يمكنه العمل مع قواعد البيانات المصممة على Access 2.0 مثلا، حيث عليك ترقيتها إلى الإصدار الأحدث وهكذا مع باقي النسخ من هذا البرنامج.

فقد أدت مشكلة عدم عمل النسخ الجديدة من Microsoft Access من ملفات قواعد البيانات المصنعة بإصدارات أقدم، أدى ذلك إلى امتناع الكثير من الشركات من الترقية من Access 97 إلى Access 2000 خوفا من دفع مبالغ طائلة لترقية البرامج المعدة للعمل مسبقا مع Access 97 ومنها فرع المحاسبة في شركة HP في وادي السليكون، حيث عندما زرته في أوائل عام 2001 كانوا يدرسوا إمكانية استبدال خمسة الآلف نسخة من Access 97 إلى Access 2000 و ما سيترتب على ذلك من تطوير لقواعد البيانات لديهم.

أما مع Access XP فلقد انتهى الأمر حيث يمكنك التعامل مع قواعد البيانات السابقة المبنية على Access 2000 بدون أية مشاكل و بدون الحاجة إلى ترقيتها، بالطبع إن لم تقم بترقيتها قد لا تستفيد من الكثير من المميزات الموجودة في النظام الجديد، ولكن الإمكانية متاحة و موجودة.

إن كنت ترغب بتغيير الإعدادات الافتراضية ل Access XP ليستخدم ملفات قواعد البيانات الجديدة، يمكنك القيام بذلك من القائمة الرئيسية

Tools -> Options -> Advance -> Default File Format -> Access 2002

لنرجع إلى نافذة قاعدة البيانات السابقة، وهي عبارة عن نافذة مقسمة إلى ثمانية أقسام رئيسية، يحتوي كلاً منها على كائنات ذات صفات مشتركة أو متشابه، جزء من هذه الكائنات موجودة في طرف أل SQL Server و الجزء الآخر موجودة في طرف Microsoft Access XP، ولكنها جميعها تظهر في قاعدة البيانات و كأنها موجودة في نفس المكان معا.

إن لم تكن متأكد من ذلك قم بإيقاف أل SQL Server 2000 و اذهب إلى الأقسام المختلفة لقاعدة البيانات و قارنها مع قاعدة البيانات قبل إيقاف خادم قواعد البيانات، حيث يمكنك عند إذن استنتاج مكان تخزين الكائنات المختلفة.

على العموم، تتكون نافذة قاعدة البيانات كما ذكرنا من ثمانية أجزاء رئيسية و هي

- الجداول Tables
- الاستعلامات Queries
- خرائط العلاقات Database Diagrams
- النماذج (الشاشات) Forms
- التقارير Reports
- صفحات الويب Pages
- الماكرو Macros
- الوحدات النمطية Modules

و يلي شرح ملخص لكلا منها

الجداول Tables

الجداول هي المنطقة التي تخزن به قواعد البيانات جداولها، وفي قاعدة بياناتنا السابقة هي نفس الجداول الخاصة بخادم قواعد البيانات SQL Server 2000 و هي لا تخزن في ملف قاعدة البيانات، بل تخزن في طرف أل SQL Server.

بمجرد بنائك لأية جدول من داخل Microsoft Access فستجده مباشرة في أل SQL Server، لان العملية بالأصل تتم في كرف أل SQL Server، حتى أن شاشات إنشاء الجداول تختلف عن شاشات صناعتها العادية قيد الكتابة

الاستعلامات Queries

و هي تربط بين الاستعلامات البسيطة Views و الاستعلامات المعقدة Stored Procedures و الإجراءات المبنية من المستخدم User Defined Functions الخاصة ب SQL Server 2000.
الثلاثة أجزاء السابقة المخزنة في طرف أل SQL Server تظهر تحت هذا البند في Microsoft Access مع اختلاف أيقونات كلا منها.
قيد الكتابة

خرائط العلاقات Database Diagrams

قيد الكتابة

النماذج (الشاشات) Forms

قيد الكتابة

التقارير Reports

قيد الكتابة

صفحات الويب Pages

قيد الكتابة

الماكرو Macros

قيد الكتابة

الوحدات النمطية Modules

قيد الكتابة

مواضيع ناقصة في هذه الوحدة

- التحدث عن أجزاء Access المنطقية
  - بيئة العمل و التحكم بها و الاستفادة منها
  - نافذة قواعد البيانات و طريقة ترابطها مع بيئة العمل
- مفهوم بناء التطبيقات بواسطة Access
- الفرق بين ملفات MDB و ملفات ADP

إنشاء واجهة لقاعدة البيانات في Access XP لأول مرة

قيد الاعداد

مواضيع لم تكتمل بعد

- كتابة برامج أل Transact SQL بواسطة Microsoft SQL Server 2000 Query Analyzer.
- إعداد Microsoft Office XP لأول مرة.
- إعداد Internet Information Service 5 لأول مرة.
- مفاهيم تصميم جداول قواعد البيانات.
- بناء العلاقات بين الجداول (مخططات العلاقات وطرق استخدامها).
- تصميم الاستعلامات البسيطة Views وطرق استخدامها.
- تصميم الاستعلامات المعقدة Stored Procedures وطرق استخدامها.
- تصميم النماذج وطرق استخدامها.
- تصميم التقارير وطرق استخدامها.
- تصميم صفحات أل WEB واستخداماتها.
- بناء وحدات أل Macro وطرق الاستفادة منها في البرامج المختلفة.
- الصلاحيات وطرق إدارتها (المستخدمين والمجموعات).
- تقنيات متقدمة في SQL Server (إنشاء فهارس لحقول النص، بناء أنواع جديدة من البيانات، Triggers، ومواضيع أخرى).
- التعريب و طرق التعريب المتقدمة.
- مراقبة الطلبات على قاعدة البيانات بواسطة Microsoft SQL Server 2000 Profiler، وطرق تسريعها و حل مشاكلها المختلفة.
- تناسخ البيانات بين قواعد البيانات Database Replication.
- تبادل البيانات بين الأجهزة المركزية Data Transformation Services.
- إدارة قاعدة البيانات المركزية (النسخ الاحتياطي، مراقبة النشاطات، إصلاح قاعدة البيانات)
- إعداد صفحات أل WEB لتعمل من خلال إنترنت (IIS 5).
- إنشاء برامج إعداد تجهيزاً لعملية توزيع البرامج أو بيعها.
- حيل متقدمة في Microsoft Access XP
- تشغيل Microsoft Analysis Services لأول مرة (قواعد البيانات المتعددة الأبعاد)
- بناء قاعدة بيانات متعددة الأبعاد لأول مرة
- بناء المكعبات و طرق التعامل بها
- شبكات اتخاذ القرار و طرق ربطها مع مكعبات البيانات
- استخدام طاقم Microsoft Office XP لزيادة فعالية برامجك
- مقدمة إلى الاستعلام باللغة الإنجليزية و مبادئ الذكاء الاصطناعي
- مبادئ بناء مشاريع استعلام باللغة الإنجليزية الطبيعية و ربطها مع Access XP أو لتعمل من خلال Web مباشرةً.
- استخدام Microsoft Office Developer XP و طريقة تحسين البرمجيات من خلاله
- مقدمة إلى DCOM+ و تفتيت البرامج على مجموعة واسعة من الأجهزة
-