

الفصل السابع عشر استخدام T-SQL داخل الاستعلامات

إذا كانت لك خبرة كافية بالتعامل مع Access وإنشاء استعلاماته وتعرف العلاقة الوطيدة بين الاستعلامات وكود SQL، فهذا لا يكفيك عند الانتقال للعمل مع مشروعات البيانات وقواعد بيانات SQL Server، حيث تختلف لهجة SQL الموجودة داخل SQL Server والمعروفة باسم Transact-SQL أو T-SQL عن لغة SQL الموجودة داخل Access في بعض السمات، وهو ما سنتعرف عليه معاً في هذا الفصل.

بانتهاء هذا الفصل سنتعرف على:

- استخدام SQL داخل Access 2007
- القواعد الأساسية للغة SQL
- كتابة استعلامات التحديد باستخدام SQL
- استخدام رموز وعلامات SQL
- ترجمة عبارات SQL داخل نافذة تصميم الاستعلام

استخدام SQL داخل Access 2007

يتناول هذا الفصل ما يعرف بـ **Structured Query Language** "لغة الاستعلامات التركيبية" وتختصر هكذا **SQL**، والقواعد المختلفة المستخدمة في كتابة هذه اللغة ثم نتعرف بعد ذلك على لغة **SQL** المستخدمة داخل قواعد بيانات **SQL Server** والمعروفة باسم **Transact-SQL** وتختصر **T-SQL**. وقد تعرفنا في الفصول السابقة من الكتاب على كيفية ترجمة **Access** للاستعلامات المختلفة وتحويلها إلى عبارات **SQL** من خلال نافذة تصميم الاستعلام. يمكن اعتبار كل من **SQL** التي تستخدمها **Access** في الاستعلامات والتقارير و**T-SQL** المستخدمة داخل قواعد بيانات **SQL Server** لهجات للغة أم هي **SQL**. وهناك بالطبع تشابهاً كبيراً بين لهجة **SQL** المستخدمة داخل **Access** ولهجة **T-SQL** المستخدمة داخل **SQL Server**، إلا أن الأولى لا يمكنها تدعيم العروض **Views** والإجراءات المخزنة **Stored Procedures** والحوادِم المرتبطة، وذلك لعدم وجود هذه العناصر داخل **Access** ووجودها كعناصر أساسية داخل **SQL Server**. وعلى الجانب الآخر، فإن **SQL** تدعم التعامل مع دوال لغة **VBA** الموجودة داخل **Access** كالدالة **CCur()** والدالة **DatePart()** على سبيل المثال. وتحتوي لغة **T-SQL** حقيقةً على ما يكافئ العديد من دوال **VBA** إلا أن الصيغة المستخدمة تكون مختلفة بالطبع، وهو ما سنحاول التعرف عليه في هذا الفصل.

ولغة **SQL** عبارة عن لغة مشتركة لقواعد بيانات الخادم والعميل، والتي تتميز بأنها لغة قياسية يمكن استخدامها عباراتها المختلفة مع جميع أنظمة إدارة قواعد البيانات المتوافقة مع **SQL**، فهي لغة تطبيقية لقواعد البيانات العلائقية، ولا تعد لغة برمجة، فهي لا تحتوي على عبارات التفرع التي تتحكم في اتجاه تنفيذ البرنامج، كما أنك لا تستطيع من خلالها إنشاء النماذج التي تستقبل البيانات من المستخدمين مثلاً.

وعلى عكس التوصيات القياسية الخاصة بلغة **HTML** ولغة **XML** مثلاً والتي تتغير من آن لآخر من خلال منظمة **W3C**، فإن إصدارات **SQL** قليلة وتتغير على فترات متباعدة.

فكانت البداية ب SQL-86 ثم SQL-92 ثم SQL-1999 وأخيراً SQL-2003. ويبنى Access SQL على الإصدار SQL-92 كما يدعم بعض السمات الموجودة في SQL1999.

وتفيد خلفيتك عن SQL في فهمك لعملية الاستعلام عن البيانات بشكل كبير، كما يمكنك استخدامها في تصميم استعلامات عالية الجودة سواء باستخدام SQL أو باستخدام T-SQL. فبقليل من المعرفة بلغة SQL يمكنك كتابة الاستعلامات الفرعية واستعلامات التوحيد وإنشاء مجموعات السجلات المستخدمة في تعبئة القوائم ومربعات السرد من خلال لغة VBA. وقد تعرفنا في الفصول السابقة على أمثلة كثيرة لاستخدام SQL توضح ما يحدث من وراء الستار عند إنشاء استعلام من خلال نافذة تصميم الاستعلام QBE وما بها من أدوات مختلفة، حيث يمكنك استخدام نافذة تصميم الاستعلامات في الحصول على كود SQL الخاص بأكثر من 90% من الاستعلامات التي تحتاج إليها داخل تطبيقات Access (حيث يجب كتابة استعلامات التوحيد Union Queries والاستعلامات المتداخلة Pass-through Queries من خلال نافذة SQL نفسها). أما في حالة مشروعات البيانات، فيمكنك الحصول على كود T-SQL الخاص بأكثر من 75% من الاستعلامات التي تحتاج إليها داخل مشروعات البيانات ADP من خلال مصمم المشروعات الرسومي Graphical Project Designer، بينما يجب عليك كتابة بقية الاستعلامات بنفسك. وعلى عكس SQL، تحتوي لغة T-SQL على العديد من العبارات المستخدمة في التحكم في تنفيذ الكود كعبارة IF ... ELSE وعبارة WHILE على سبيل المثال لا الحصر. فإذا ما أردت الحصول على معظم السمات الموجودة باستعلامات SQL Server 2005، يجب في هذه الحالة تعديل عبارات T-SQL من خلال مصمم المشروعات.

يمكنك تعلم عبارات SQL بسهولة شديدة من خلال أمثلة عملية. كلما قمت بإنشاء استعلام من خلال نافذة تصميم الاستعلام أو نافذة مصمم المشروعات، انتقل إلى عرض SQL وشاهد عبارات SQL أو T-SQL المكافئة للاستعلام الذي قمت بإنشائه.



القواعد الأساسية للغة SQL

عند البدء في تعلم القواعد الأساسية لإحدى اللغات الجديدة، من الأفضل تقسيم هذه القواعد إلى مجموعات مختلفة تبعاً للاستخدام ثم مجموعات أخرى تبعاً للتحدث. ويمكن تقسيم أوامر لغة SQL تبعاً للاستخدام بدايةً إلى ست مجموعات أساسية وهي:

- أوامر الاستعلام عن البيانات (Data Query Language (DQL) وتحتوى على الأوامر المستخدمة في الحصول على البيانات من الجداول وتحديد كيفية عرض هذه البيانات، ويأتى على رأس هذه المجموعة الأمر SELECT.
- أوامر معالجة البيانات (Data Manipulating Language (DML) وتحتوى على الأوامر المستخدمة في معالجة البيانات Data Manipulation Language (DML) كالأمر INSERT المستخدم في إضافة صفوف جديدة إلى الجداول والأمر DELETE المستخدم في حذف صفوف من الجداول والأمر UPDATE المستخدم في تعديل قيم أعمدة معينة بالصفوف الموجودة بالجداول.
- أوامر معالجة الإجراءات (Transaction Processing Language (TPL) ومن أمثلتها الأوامر BEGIN TRAN و COMMIT و ROLLBACK المستخدمة في تنفيذ عدد من أوامر المجموعة السابقة DML في خطوة واحدة. فإذا فشل تنفيذ أحد الأوامر الموجودة بالمجموعة، يتم على الفور التراجع عن بقية الأوامر، أى يتم تنفيذ مجموعة الأوامر كحزمة واحدة.
- أوامر تعريف البيانات (Data Definition Language (DDL) ومن أمثلتها الأوامر ADD | ALTER COLUMN و CREATE | ALTER TABLE

و **CREATE VIEW** المستخدمة في إنشاء الجداول والعروض. كما تستخدم أوامر **DDL** أيضاً لتعديل الجداول وإنشاء/حذف الفهارس.

- أوامر التحكم في مكان المؤشر **Cursor Control Language (CCL)** وتشمل الأوامر المستخدمة في اختيار صف معين داخل مجموعة السجلات الناتجة من أحد الاستعلامات مثل الأمر **UPDATE WHERE CURRENT**.
- أوامر التحكم في البيانات **Data Control Language (DCL)** أو أوامر تأمين البيانات **Data Security Language** وتقوم هذه المجموعة بأداء الوظائف الإدارية التي تحدد أولويات وأذونات استخدام قاعدة البيانات كالأمر **GRANT** والأمر **REVOKE**، أو استخدام جداول معينة بقاعدة البيانات أو أوامر **SQL** معينة.

هذا من حيث الاستخدام، أما من حيث المصطلحات، فيمكن تقسيم مصطلحات **SQL** إلى المجموعات التالية:

- الأوامر **Commands** مثل الأوامر **SELECT** و **EXECUTE** و **CREATE** و **ALTER** وتمثل الأفعال المستخدمة في تنفيذ الأحداث المختلفة.
- المحددات **Qualifiers** مثل كلمة **WHERE** المستخدمة في تحديد أو تقييد البيانات الناتجة من الاستعلام.
- العبارات **Clauses** مثل عبارة **ORDER BY** وتستخدم في تعديل ترتيب البيانات الناتجة.
- المسندات **Predicates** مثل الكلمات **IN** و **ALL** و **ANY** و **SOME** و **LIKE** و **UNIQUE** المستخدمة في اختبار حقائق معينة بقيم البيانات الناتجة من الاستعلام وإرجاع القيمة **TRUE** "صح" أو **FALSE** "خطأ" (وأحياناً القيمة **NULL**).
- المعاملات **Operators** ويطلق عليها أيضاً مسندات المقارنة مثل علامات **=** و **<** و **>** المستخدمة في مقارنة القيم وتستخدم مع العبارات التي تحتوى على **WHERE** و **JOIN**.

- دوال التجميع **Group aggregate functions** مثل الدوال **AVG()** و **COUNT()** و **MAX()** و **MIN()** و **SUM()** والتي تقوم بإرجاع قيمة واحدة من مجموعة من القيم.
- دوال التحويل بين أنواع البيانات **Data type conversions functions** والتي تقوم بتحويل القيم من نوع بيانات معين إلى نوع بيانات آخر، كالدالة **CAST()** والدالة **CONVERT()** على سبيل المثال.
- الدوال المساعدة **Utility functions** وتقوم بإرجاع القيم التي يتم تحديدها من خلال تعبيرات. ويمكنك على سبيل المثال استخدام الدالة **NULLIF()** لإرجاع القيمة **NULL** إذا كانت القيمة الناتجة من الدالة **TRUE**. ومن أمثلة دوال هذه المجموعة دوال معالجة النصوص ودوال التاريخ والوقت.
- الكلمات المحجوزة **Reserved words** ويتم من خلالها تعديل أحداث العبارات أو معالجة المؤشرات المستخدمة في التعامل مع الصفوف الناتجة من الاستعلامات، ومن أمثلتها كلمة **XML FOR** المستخدمة داخل **T-SQL** والتي تقوم بإرجاع مستند **XML** بدلاً من مجموعة السجلات الناتجة من الاستعلام.

يتم عادةً كتابة الكلمات المحجوزة داخل **SQL** بحروف كبيرة، إلا أنها غير حساسة لحالة الأحرف، أي يتم تنفيذها بشكل سليم حتى لو قمت بكتابتها بحروف صغيرة.



كتابة استعلامات التحديد باستخدام SQL

تعتبر عبارة **SELECT** أهم عبارات **SQL** على الإطلاق، حيث تستخدم هذه العبارة في الحصول على مجموعة معينة من السجلات من جدول أو أكثر وتحتوي على الصيغة العامة التالية:

```
SELECT [ALL|DISTINCT|DISTINCTROW] [TOP (n) [PERCENT]]
[WITH TIES] select_list
FROM table_names
[WHERE search_criteria]
[ORDER BY column_criteria [ASC|DESC]]
```

حيث:

- تمثل كلمة **SELECT** الأمر الأساسي المستخدم في الاستعلام.
- يتم استبدال **select_list** بقائمة الأعمدة (الحقول) التي ترغب في استرجاع بياناتها من خلال الاستعلام. وعند استخدام نافذة تصميم الاستعلام في تعيين الاستعلام، يتم تحديد هذه الأعمدة عن طريق إدراجها من الجداول الموجودة بالجزء العلوي إلى الصف **Fields** بشبكة **QBE** الموجودة بالجزء السفلي بشرط تنشيط مربع الاختيار الموجود بالصف **Show** مع الأعمدة (الحقول) التي ترغب في تضمينها بالقائمة **select_list**، حيث يتم فصل هذه الحقول باستخدام علامة الفاصلة (,).
- يدل القوسان [] في الجزء **[ALL|DISTINCT|DISTINCTROW]** على أن ما بينهما اختياري، كما يدل الحرف | على أنك تستطيع استخدام عنصر واحد فقط من العناصر الثلاثة. في حالة اختيار **ALL** يتم تضمين جميع الصفوف (السجلات) الناتجة من الاستعلام. وفي حالة اختيار **DISTINCT**، يتم حذف الصفوف التي تحتوي على بيانات مكررة. أما في حالة اختيار **DISTINCTROW**، فيتم حذف الصفوف التي تحتوي على بيانات مكررة كما في حالة اختيار **DISTINCT** مع القدرة على تعديل القيم الناتجة من الاستعلام، وهذا الخيار موجود في **Access SQL** فقط ولا يوجد في **T-SQL**.
- يستخدم الجزء الاختياري **[TOP (n) [PERCENT]]** في حالة الرغبة في استرجاع عدد معين أو نسبة معينة من السجلات المفترض الحصول عليها من الاستعلام، حيث يتم استبدال **n** بالعدد المطلوب أو النسبة المطلوبة مع اختيار **PERCENT** في الحالة الأخيرة. فعلى سبيل المثال، لاختيار أول ١٠٠ صف،

استخدم التعبير (100) TOP، ولاختيار نصف السجلات الناتجة، استخدم التعبير TOP (50) PERCENT. ويمكنك داخل SQL Server 2005 استخدام متغير بدلاً من n.

- يستخدم الجزء الاختياري [WITH TIES] داخل T-SQL فقط مع العبارة ORDER BY ويتسبب في إرجاع المزيد من الصفوف الناتجة عن القيم المتساوية داخل الحقل المستخدم بعبارة ORDER BY.
- يستخدم الجزء FROM table_name في تحديد الجدول أو الجداول مصدر السجلات الناتجة. وعند استخدام نافذة تصميم الاستعلام في تعيين الاستعلام، يتم تحديد هذه الجداول من خلال المربع الحوارى Add Table. وفي حالة تضمين حقول من أكثر من جدول بالجزء select_list، يجب في هذه الحالة تضمين هذه الجداول بالجزء table_name، حيث يتم فصل هذه الجداول باستخدام علامة الفاصلة (,). فإذا كان هناك أكثر من حقل بنفس الاسم في أكثر من جدول، يجب أن يكون كل حقل داخل الجزء select_list مصحوباً باسم الجدول المناسب.
- يحدد الجزء الاختياري WHERE search_criteria السجلات التي سيتم عرضها من القائمة الناتجة عن طريق استبدال search_criteria بإحدى علامات المقارنة مثل LIKE أو = في حالة الحقول النصية، < و > و = في حالة الحقول أو المعاملات الرقمية. فإذا لم تقم بتضمين هذا الجزء داخل عبارة SELECT، يتم إرجاع جميع السجلات الموجودة بالجدول المحدد بالجزء FROM table_name.
- يستخدم الجزء الاختياري [ORDER BY column_criteria] في تحديد ترتيب (فرز) السجلات الناتجة تبعاً للقيم الموجودة داخل عمود (حقل) أو أكثر من الأعمدة الموجودة بالاستعلام، كما يمكنك استخدام كلمة ASC للحصول على الفرز التصاعدي أو كلمة DESC للحصول على الفرز التنازلي، والأولى هي القيمة الافتراضية.



عند إضافة الحقول من جدولين أو أكثر ولم تقم بربط الجداول من خلال العبارة **WHERE Table1.Field1 = Table2.Field2** أو من خلال العبارة **JOIN**، يتم إرجاع جميع السجلات الموجودة بكل جدول من هذه الجداول وهو ما يعمل على تكديس الشبكة واختناقها إذا تم تنفيذ الاستعلام من خلال الشبكة، لذا يجب مراعاة ذلك عند تصميم الاستعلامات.

استخدام رموز وعلامات SQL

بالإضافة إلى معاملات المقارنة المستخدمة داخل التعبيرات المختلفة، تحتوي SQL على عدد من الرموز والعلامات الأخرى كالفاصلة والفاصلة المنقوطة والأقواس والنقطة، حيث لكل عنصر من هذه العناصر مدلوله الخاص داخل SQL وذلك كما يلي:

- الفاصلة (,) وتستخدم في فصل العناصر المختلفة الموجودة بقائمة واحدة كما في أسماء الحقول. ففي حالة وجود ثلاثة حقول على سبيل المثال، يتم كتابة هذه الحقول داخل عبارة SQL هكذا **Name, Address, Age**.
- الأقواس المربعة [] وتستخدم للإحاطة بأسماء الحقول التي تحتوي على مسافات أو كلمات محجوزة أو أي رموز أخرى كما في الحقل **[Order Details]**، فلا يصح كتابة هذا الحقل داخل عبارة SQL بدون هذه الأقواس. كما يجب استخدام هذه الأقواس مع معاملات الإدخال **Input Parameters** التي يتم استخدامها داخل استعلامات **Access** من خلال نافذة التصميم.
- النقطة [.] وتستخدم في فصل أسماء الكائنات ذات التصنيفات المتداخلة. فعلى سبيل المثال، في حالة تضمين حقول من أكثر من جدول داخل الاستعلام، يتم فصل كل حقل عن اسم الجدول المصاحب باستخدام النقطة هكذا مثلاً:
[Order Details]. Order_no
- معرفات النصوص وهي علامة التنصيص (") وعلامة التنصيص المزدوجة ("") المستخدمة في كتابة النصوص داخل عبارات SQL، ومن الأفضل استخدام العلامة " بدلاً من "" نظراً لاستخدام الأولى في كتابة عبارات SQL داخل كود VBA.

- الرموز الشاملة **Wildcards** وهى الرمز ؟ المستخدم فى الدلالة على غياب حرف واحد والرمز * المستخدم فى الدلالة على غياب أكثر من حرف.
- معرف التاريخ والوقت، وهو الرمز #، حيث يجب وضع قيم التاريخ والوقت داخل عبارات SQL بين الرمزين # #.
- معرفات المتغيرات (: و@)، حيث يتم استخدام المعرف : للإشارة إلى المتغيرات فى ANSI SQL بينما يتم استخدام المعرف @ للإشارة إلى المتغيرات فى T-SQL، من المهم أن تعرف أن SQL لا تدعم أي من المعرفين.
- الرمز ! ويستخدم كبديل للمعامل NOT. يتم استخدام التركيب != داخل ANSI SQL للإشارة إلى عدم التساوى، بينما يتم استخدام < داخل SQL و T-SQL للإشارة إلى ذلك، كما تدعم T-SQL أيضاً استخدام !=.

ترجمة عبارات SQL داخل نافذة تصميم الاستعلام

حينما تقوم بإنشاء استعلام تحديد باستخدام العبارة SELECT من خلال نافذة تصميم الاستعلام، يقوم Access بترجمة التصميم الذى قمت به إلى كود SQL المصاحب. وكذلك الحال فى مشروعات البيانات، يقوم مصمم المشروعات بترجمة الاستعلام إلى كود T-SQL المناسب. فإذا قمت بتغيير الكود الناتج من خلال نافذة SQL أو T-SQL، تنعكس التغييرات التى قمت بها على الاستعلام فى نافذة تصميم الاستعلام أو مصمم المشروعات على الترتيب.

إنشاء كود SQL

لإنشاء كود SQL المكافئ لاستعلام Access داخل قاعدة البيانات Sales.accdb، تابع معنا الخطوات الآتية:

١. تأكد من فتح قاعدة البيانات Sales.accdb وإلا قم بفتحها ثم انقر التبويب

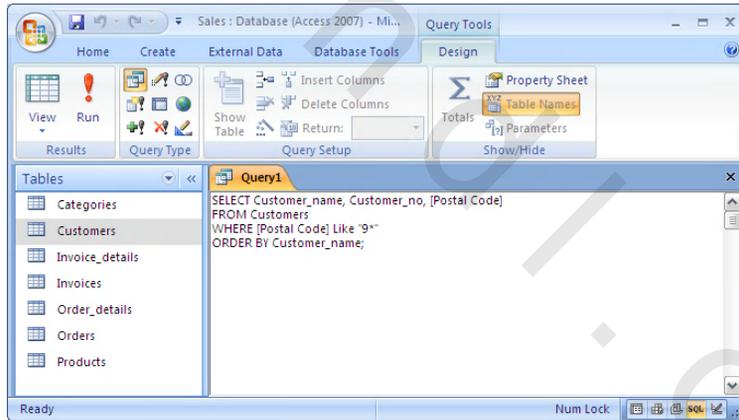
Create بالجزء العلوى من الشاشة وانقر زر تصميم الاستعلام  Query Design، وحينئذٍ يتم فتح استعلام جديد بنافذة تصميم الاستعلام.

٢. قم بإغلاق المربع الحوارى **Show Table** ثم انقر الزر  بالجانب الأيسر من شريط الأدوات، وحينئذٍ تظهر نافذة استعلام بعنوان **Query1** محتويةً على كلمة **.SELECT**

٣. قم بتعديل عبارة **SQL** كما يلي (انظر شكل ١-١٧):

```
SELECT Customer_name, Customer_no, [Postal Code]
FROM Customers
WHERE [Postal Code] Like "9*"
ORDER BY Customer_name;
```

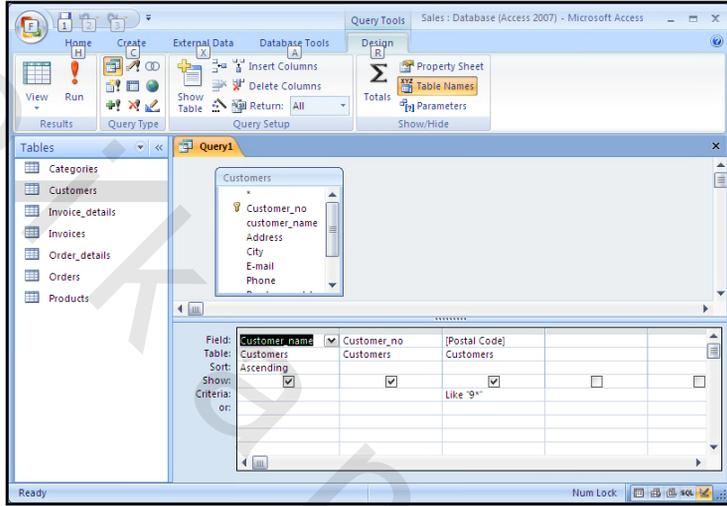
حيث تقوم هذه العبارة بعرض الأعمدة **Customer_name** و **Customer_no** و **Postal Code** من الجدول **Customers** للعملاء الذين يبدأ العمود **Postal Code** فيها بالرقم 9 مع ترتيب السجلات الناتجة تبعاً لاسم الشركة ممثلاً في العمود **Customer_name**.



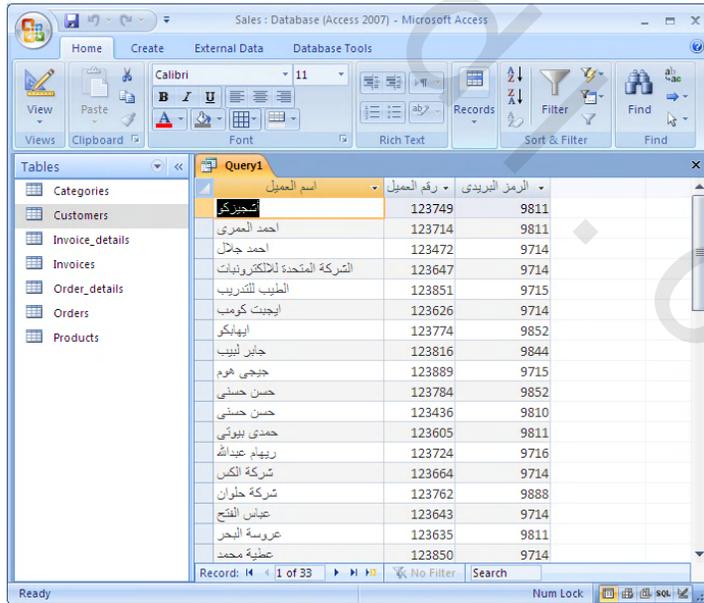
شكل ١-١٧ كتابة عبارة **SQL** من خلال نافذة عرض **SQL**.

٤. انقر الزر  بأقصى يسار شريط الأدوات ثم اختر **Design View** من القائمة الناتجة لعرض شبكة **QBE** وبها الاستعلام المكافئ لعبارة **SQL** التي قمت بتعيينها (انظر شكل ٢-١٧).

٥. انقر الزر **Run** بجانب الأيسر من شريط الأدوات لتشغيل عبارة **SQL** التي قمت بإدخالها وبالتالي عرض السجلات الناتجة من الاستعلام (انظر شكل ٣-١٧).



شكل ٢-١٧ الاستعلام المكافئ لعبارة **SQL** داخل نافذة تصميم الاستعلام.



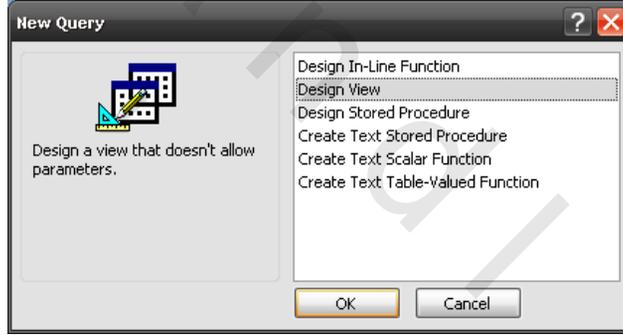
شكل ٣-١٧ نتيجة تشغيل عبارة **SQL** داخل نافذة عرض الاستعلام.

العمل مع العروض (SQL Server Views)

تشبه العروض Views داخل SQL Server الاستعلامات SELECT داخل Access، إلا أنك تستطيع داخل SQL Server الحصول على نتائج الاستعلامات باستخدام الإجراءات المخزنة Stored Procedures والدوال السطرية Inline Functions. العرض View عبارة عن جملة SELECT تستخدم كبديل لاستعلام التحديد QueryDef داخل SQL Server.

لإنشاء أحد العروض من عبارة SQL باستخدام مصمم المشروعات داخل مشروع بيانات (ADP)، تابع معنا الخطوات الآتية:

1. افتح المشروع SalesProjectCS.adp ثم انقر التبويب Create ومنه انقر الزر Query Wizard. يظهر المربع الحوارى New Query (انظر شكل ٤-١٧).



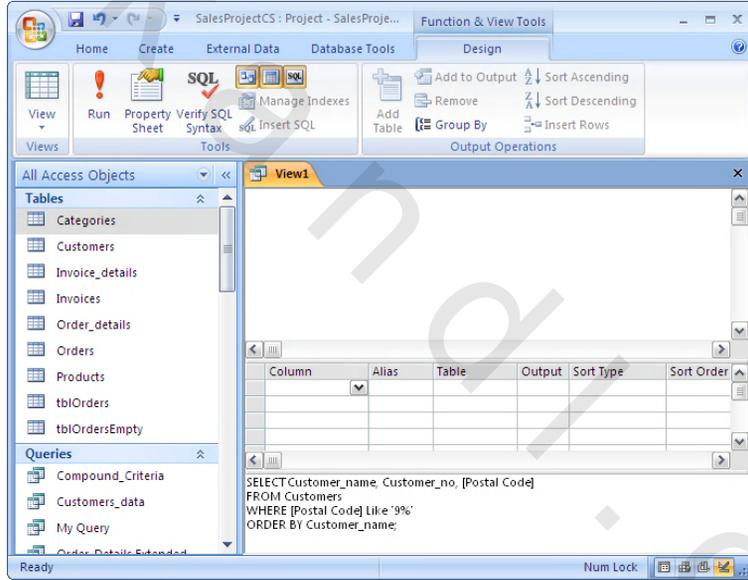
شكل ٤-١٧ المربع الحوارى New Query.

2. اختر Design View من مربع السرد ثم انقر زر OK لإغلاق المربع الحوارى.
3. أغلق المربع الحوارى Add Table ثم نشط التبويب Design ثم انقر الزر  بالجزء Tools لإظهار منطقة نص SQL التى تحتوى على عبارة SELECT ... FROM.

٤. قم بكتابة عبارة SQL بنافذة SQL بالجزء السفلي من الشاشة، وعلى فرض أننا نرغب في استخدام نفس عبارة SQL السابقة، سنقوم بإدخال بعض التعديلات على العبارة كي تتناسب مع T-SQL هكذا (انظر شكل ٥-١٧):

```
SELECT Customer_name, Customer_no, [Postal Code]
FROM Customers
WHERE [Postal Code] Like '9%'
ORDER BY Customer_name;
```

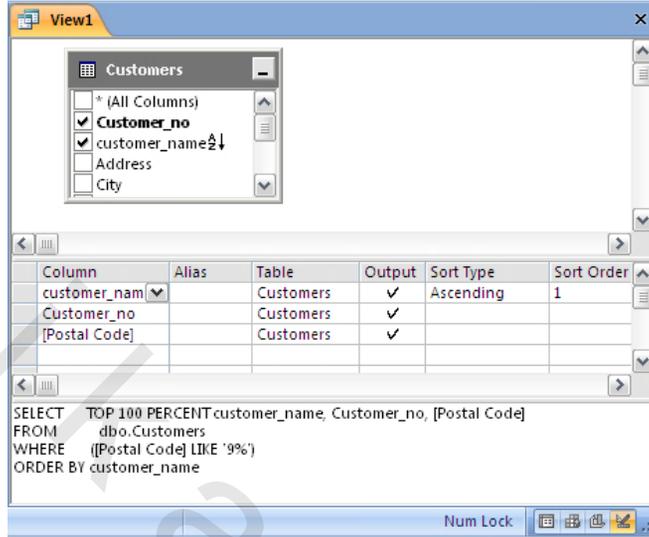
حيث قمنا باستبدال المعيار "9*" بالجزء "LIKE" داخل SQL بالجزء "9%" ، أى استبدال الأقواس المزدوجة بالأقواس المفردة واستبدال الرمز * بالرمز %.



شكل ٥-١٧ كتابة عبارة SQL.

٥. انقر الزر  ثم انقر زر OK رداً على مربع الرسالة الناتج. يظهر الاستعلام الناتج من عبارة SQL بشبكة QBE بالجزئين العلوي والأوسط (انظر شكل ٦-١٧).

الفصل السابع عشر : استخدام T-SQL داخل الاستعلامات



شكل ٦-١٧ عرض الاستعلام بشبكة QBE.

٦. احفظ العرض باسم مناسب ثم انقر زر التشغيل ! لعرض نتيجة الاستعلام بنافذة عرض صفحة البيانات (انظر شكل ٧-١٧)، حيث يجب داخل قواعد بيانات SQL Server حفظ العروض أولاً قبل تشغيلها.

الرمز البريدي	اسم العميل	رقم العميل
9810	حسن حسنى	123436
9811	محمد يوسف	123461
9714	احمد جلال	123472
9855	وحد	123519
9811	مصطفى حسان	123528
9714	محمد سامي	123539
9714	محمود ثابت	123552
9714	مؤسسة الأمل	123560
9858	محمد القومى	123562
9714	مراون ابراهيم	123589
9811	ليلان	123598
9811	حمدى بيوتى	123605
9714	هالة	123607
9714	على عبدالله	123623
9714	على درويش	123624
9714	ايحيى كرمب	123626
9811	عروسة البحر	123635

شكل ٧-١٧ عرض الاستعلام داخل صفحة البيانات.

استخدام الدوال التجميعية والدوال السطرية

يمكنك استخدام الدوال التجميعية (Aggregate Functions) في الحصول على الإجماليات والمتوسطات الحسابية والبيانات الإحصائية لمجموعة من السجلات تحتوى على قيمة مشتركة، وفي هذه الحالة يجب استخدام عبارة GROUP BY داخل عبارة SQL وذلك لتحديد السجلات (الصفوف) التي يتم تطبيق هذه الدوال عليها. كما يمكنك تصفية نتيجة عبارة GROUP BY باستخدام كلمة HAVING. والصيغة العامة لهذه العبارة كما يلي:

```
SELECT [ALL|DISTINCT] [TOP (n) [PERCENT]]
      Aggregate_function(field_name) AS alias_name
      [, select_list]
FROM table_names
[WHERE search_criteria]
GROUP BY group_criteria
[HAVING aggregate_criteria]
[ORDER BY column_criteria]
```

حيث:

- تتضمن select_list الدالة التجميعية aggregate_function التي تحتوى على معامل عبارة عن field_name حيث يجب أن يحتوى هذا الحقل على قيمة رقمية.
- يتم استبدال alias_name بعنوان للعمود الجديد الذى يحتوى على القيمة الناتجة من الدالة التجميعية، وهو يكافئ:

alias : aggregate_function (field name) بشبكة الاستعلام داخل

.Access

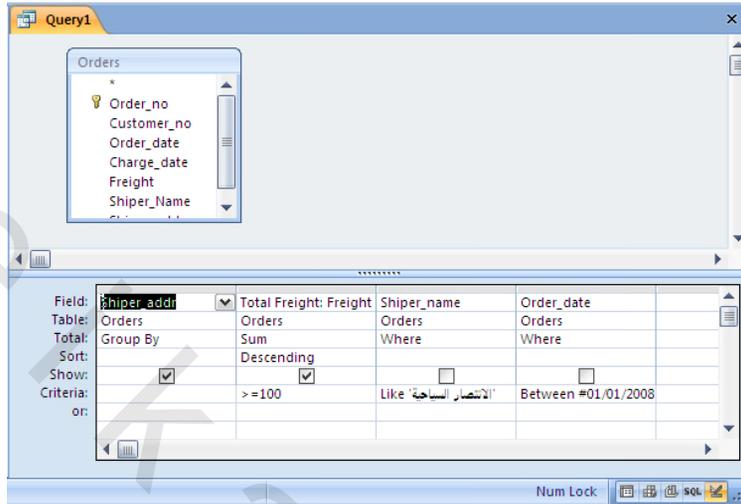
- يستخدم GROUP BY group_criteria في إنشاء العمود أو الأعمدة التي يتم تجميعها (أى تطبيق الدالة عليها)، حيث تظهر عبارة GROUP BY في هذا العمود بالصف Totals بشبكة الاستعلام. وهذه العبارة أساسية في حالة الاستعلامات التجميعية.

- يستخدم الجزء **HAVING aggregate_criteria** في تطبيق معيار أو أكثر على العمود الذي يحتوي على الدالة التجميعية، حيث يتم تطبيق الشرط **aggregate_criteria** بعد الانتهاء من عملية التجميع نفسها. وهذا الجزء اختياري.
- يعمل الجزء **WHERE search_criteria** قبل إجراء عملية التجميع، أي قبل تنفيذ الدالة. وهذا الجزء اختياري، إلا أنه نادراً ما يخلو منه أى استعلام تجميعي. للتعرف على كيفية استخدام الدوال التجميعية، قم بتنفيذ الكود التالي داخل قاعدة البيانات **Sales.accdb** من خلال استعلام بالطريقة التي ذكرناها منذ قليل:

```
SELECT      Orders.Shiper_addr,
            Sum(Orders.Freight) AS [Total Freight]
FROM Orders
WHERE (((Orders.[Shiper_name]) Like 'الانتصار السياحية')
AND ((Orders.[Order_date]) Between #1/1/2008# And
#12/31/2008#))
GROUP BY Orders.Shiper_addr
HAVING (((Sum(Orders.[Freight]))>=100))
ORDER BY Sum(Orders.Freight) DESC;
```

تقوم هذه العبارة باسترجاع عمودين، الأول عنوان شركة الشحن (العمود **Shiper_addr**) بينما يحتوي الثاني على إجماليات مصروفات الشحن لكل منطقة في سنة ٢٠٠٨ على ألا تقل هذه المصروفات عن ١٠٠.

يوضح شكل ٨-١٧ طريقة عرض تصميم الاستعلام المكافئ للكود السابق.



شكل ٨-١٧ كود SQL السابق بطريقة عرض تصميم الاستعلام.

إذا قمت الآن بتشغيل هذا الاستعلام، تحصل على ثلاثة صفوف لمناطق (عناوين) شركات الشحن المختلفة. فإذا قمت بحذف الجزء **HAVING** من العبارة (أى عدم اشتراط ألا تقل المصروفات عن ١٠٠)، تحصل على عدد أكبر من الصفوف في هذه الحالة. أما داخل **SQL Server**، فيمكنك استخدام العروض التي تدعم المعاملات من خلال نوع من الدوال يسمى **Table-valued functions** أو **TVFs** أو الدوال السطرية. للتعرف على كيفية استخدام عبارة **SQL** السابقة داخل **SQL Server**، تابع معنا الخطوات التالية:

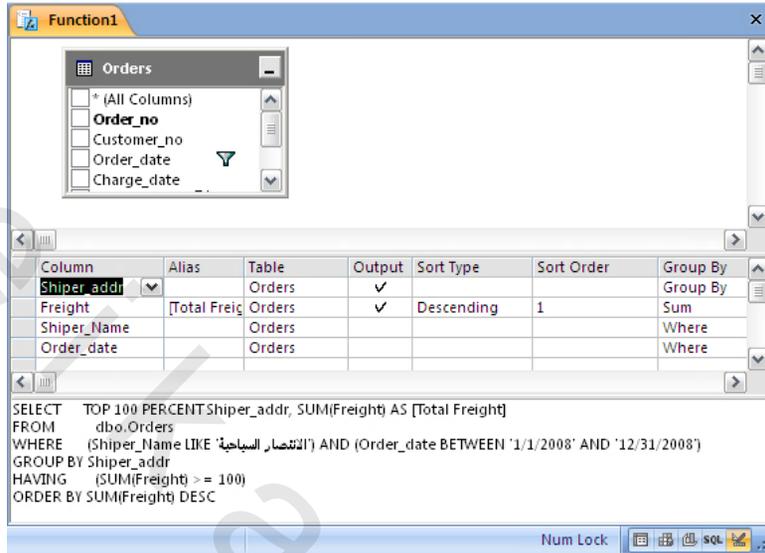
١. افتح مشروع البيانات **SalesProjectCS.adp** إذا لم يكن مفتوحاً بالفعل ثم نشط التبويب **Create** بالجزء العلوى من الشاشة وانقر الزر **Query Wizard**، يظهر المربع الحوارى **New Query** (راجع شكل ٤-١٧).
٢. اختر **Design In-Line Function** من مربع السرد ثم انقر زر **Ok** لإغلاق المربع الحوارى **New Query**. تظهر النافذة **Function1** وبها المربع الحوارى **Add Table**.

٣. انقر زر **Close** لإغلاق المربع الحوارى **Add Table** ثم قم بتنشيط التبويب **Design** ثم انقر الزر  بالجزء **Tools** لإظهار منطقة نص **SQL** التى تحتوى على عبارة **.SELECT ... FROM**.

٤. قم بكتابة عبارة **SQL** بنافاذة **SQL** بالجزء السفلى من الشاشة، وعلى فرض أننا نرغب فى استخدام نفس عبارة **SQL** السابقة، سنقوم بإدخال بعض التعديلات على العبارة كى تتناسب مع **T-SQL** هكذا (لاحظ استبدال # فى التواريخ بعلامة التنصيص ':'):

```
SELECT      Orders.Shiper_addr,
Sum(Orders.Freight) AS [Total Freight]
FROM Orders
WHERE (((Orders.[Shiper_name]) Like 'الانتصار السياحية')
AND ((Orders.[Order_date]) Between '1/1/2008' And '12/31/2008'))
GROUP BY Orders.Shiper_addr
HAVING (((Sum(Orders.[Freight]))>=100))
ORDER BY Sum(Orders.Freight) DESC;
```

٥. انقر الزر  للتأكد من صحة الكود الذى قمت بإدخاله ثم انقر زر **Ok** رداً على مربع الرسالة الناتج. يظهر الاستعلام الناتج من عبارة **SQL** بشبكة **QBE** بالجزئين العلوى والأوسط (انظر شكل ٩-١٧).



شكل ٩-١٧ عرض الاستعلام بشبكة QBE.

٦. قم بحفظ الدالة باسم مناسب ثم انقر زر التشغيل ! لعرض نتيجة الاستعلام بنافذة عرض صفحة البيانات، حيث نحصل في هذه الحالة على نفس النتيجة السابقة التي حصلنا عليها في حالة استعلام Access.

تستخدم الدوال السطرية بدلاً من الإجراءات المخزنة في الحصول على العرض المصاحب، وذلك لأن العروض لا تدعم استخدام المعاملات، كما أن مجموعات البيانات الناتجة من هذه الدوال قابلة للتحديث بعرض ورقة البيانات على عكس مجموعات البيانات الناتجة من الإجراءات المخزنة.



ربط الجداول من خلال كود SQL

يتم ربط جدولين أو أكثر داخل نافذة تصميم استعلامات Access أو داخل نافذة مصمم المشروعات باستخدام التركيب ... ON JOIN والذي يتم من خلاله تحديد الجداول المرتبطة والحقول المستخدمة في عملية الربط، حيث يحتوي التركيب على الصيغة العامة التالية:

SELECT [ALL|DISTINCT] select_list

```
FROM
table_name [INNER|LEFT [OUTER]] |FULL [OUTER] ]
JOIN join_table ON join_criteria
[table_name [INNER|LEFT [OUTER]|RIGHT [OUTER] ] |FULL
[OUTER] ]
JOIN join_table ON join_criteria]
[WHERE search_criteria]
[ORDER BY column_criteria]
```

حيث:

• يستخدم الجزء

```
table_name [INNER|LEFT [OUTER]] |FULL [OUTER] ] JOIN
join_table
```

في تحديد اسم الجدول المرتبط مع بقية الجداول المذكورة في `table_name`. فإذا
قمت بإنشاء ربط ذاتي لنفس الجدول عن طريق تضمين نسختين من قائمة حقول
نفس الجدول، يتم تمييز الجدول الثاني عن الأول بإضافة شرطة تحتية (_) ورقم ما.

يجب أن يسبق عبارة `JOIN` داخل استعلامات `Access` إحدى الكلمات
الخاصة بأنواع الربط المختلفة `INNER` أو `FULL` أو `LEFT` أو `RIGHT`، بينما
استخدام `INNER` اختياري في كود `T-SQL`.



• يستخدم الجزء `ON join_criteria` في تحديد الحقلين المستخدمين في عملية الربط
وكذلك العلاقة بين هذه الحقول، حيث يحتوي التعبير `join_criteria` عادةً على
المعامل = ويقوم بإرجاع القيمة `TRUE` أو القيمة `FALSE`.

ويتوقف عدد عبارات `JOIN` التي يتم استخدامها داخل عبارة `SQL` على عدد الجداول
المستخدمة في عملية الربط، حيث تقل هذه العبارات بمقدار واحد عن عدد الجداول. أي
لربط جدولين، يتم استخدام عبارة `JOIN` واحدة، بينما يتم استخدام عبارتين لربط ثلاثة
جداول، وهكذا.

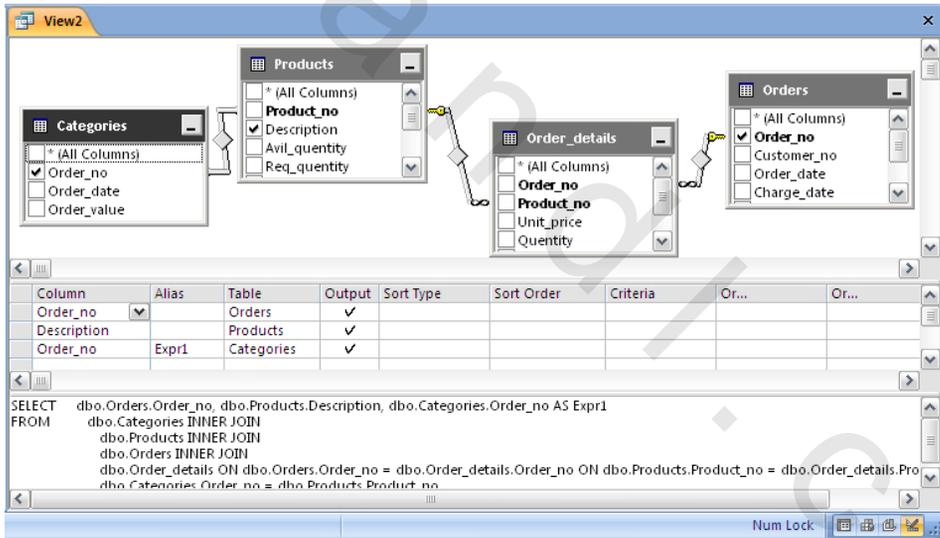
للتعرف على كيفية ربط الجداول باستخدام كود `SQL`، دعنا نرى العبارة التالية والتي يتم
فيها إنشاء عملية ربط داخلي بين أربعة جداول وهي الجداول `Orders` و `Order`

`Products` و `Categories`:

```

SELECT      Orders.Order_no,      Products.Description,
Categories.Order_no
FROM Categories INNER JOIN (Products INNER JOIN (Orders
INNER JOIN Order_details ON Orders.Order_no =
Order_details.Order_no) ON Products.Product_no =
Order_details.Product_no) ON Categories.Order_no =
Products.Product_no;
    
```

تقوم هذه العبارة بإرجاع قائمة المنتجات من الجدول Products والمجموعة الخاصة بها من الجدول Categories وذلك مع كل طلب صرف داخل الجدول Orders. ويمكنك نسخ هذه العبارة من نافذة عرض SQL داخل Access إلى مصمم المشروعات الخاص بعرض أو إجراء مخزن أو دالة سطرية للحصول على نفس النتيجة. يوضح شكل ١٠-١٧ كود T-SQL المكافئ للكود السابق داخل نافذة عرض داخل SQL Server.



شكل ١٠-١٧ ربط الجداول من خلال كود T-SQL

وبدلاً من استخدام JOIN في ربط الجداول، يمكنك أيضاً استخدام WHERE للوصول إلى نفس النتيجة، إلا أن هذه الطريقة غير قابلة للتحديث. فعلى سبيل المثال، للحصول على نفس النتيجة السابقة باستخدام WHERE، يمكنك استخدام عبارة SQL التالية:

```

SELECT      Orders.Order_no,      Products.Description,
Categories.Order_no
    
```

```
FROM Orders, Order_details, Products, Categories
WHERE Orders.Order_no = Order_details.Order_no
AND Products.Product_no = Order_details.Product_no
AND Categories.Order_no = Products.Product_no;
```

ولكن إذا قمت بنسخ هذا الكود إلى مصمم المشروعات داخل SQL Server ثم قمت بنقر زر  ، فسيقوم المصمم بتحويل عبارة WHERE إلى عبارات JOIN المصاحبة ونحصل على الكود التالي:

```
ELECT      dbo.Orders.Order_no, dbo.Products.Description,
dbo.Categories.Order_no
FROM      dbo.Orders INNER JOIN
          dbo.Order_details ON dbo.Orders.Order_no =
          dbo.Order_details.Order_no INNER JOIN
          dbo.Products ON dbo.Order_details.Product_no =
          dbo.Products.Product_no INNER JOIN
          dbo.Categories ON dbo.Products.Product_no =
          dbo.Categories.Order_no
```

إذا أردت الاحتفاظ بعبارة WHERE داخل الكود كما هي دون تحويل، يجب كتابة الاستعلام باستخدام إحدى أدوات تنفيذ الاستعلام مثل نافذة استعلام SQL Server Management Studio على سبيل المثال.

كتابة استعلامات التوحيد *Union Queries*

يمكنك من خلال استعلامات التوحيد تجميع البيانات الناتجة من أكثر من استعلام SELECT في مجموعة واحدة، وهذا ما يحدث عادةً مع الجداول متشابهة التركيب كالجداول Customers والجداول Suppliers على سبيل المثال، حيث يتم إنشاء هذا النوع من الاستعلامات سواءً داخل Access أو داخل SQL Server من خلال كود SQL على عكس الاستعلامات الأخرى التي يتم إنشاؤها من خلال نوافذ التصميم. وتحتوي استعلامات التوحيد على الصيغة العامة التالية:

```
SELECT field_list
UNION SELECT field_list
[GROUP BY group_criteria]
[HAVING aggregate_criteria]
[UNION SELECT field_list
```

```
[GROUP BY group_criteria]
[HAVING aggregate_criteria] ]
[UNION ...]
[ORDER BY column_criteria]
```

حيث:

- يجب أن يتساوى عدد الحقول الموجودة بالقائمة **field_list** داخل الجزء **SELECT** مع الحقول الموجودة بكل **field_list** داخل كل جزء **UNION** وإلا ستحصل على رسالة خطأ بمجرد تنفيذ هذا الكود.
- يجب ترتيب الحقول الموجودة داخل القائمة **field_list** ترتيباً واحداً في جميع عبارات **SELECT** وإلا ستحصل على نتائج غير سليمة، حيث لا تظهر رسائل خطأ في هذه الحالة. وفي حالة استعلامات توحيد **Access**، يتم ترقية نوع البيانات في حالة اختلاف نوع بيانات الحقل المصاحب في عبارة **SELECT** أخرى. أما **SQL Server** فلا يدعم عملية التحويل التلقائي لأنواع البيانات.
- يمكنك استخدام **ORDER BY** بعد عبارة **UNION SELECT**، كما يمكنك استخدام **GROUP BY** و **HAVING** داخل كل عبارة **SELECT** وعبارة **UNION SELECT**.

يوضح كود **SQL** التالي استعلام توحيد يقوم بتجميع الصفوف من جدولين، الأول **Orders** والآخر **Invoices**:

```
SELECT Customer_no, Order_date, 'Order' AS Relationship
FROM Orders
UNION
SELECT Customer_no, Invoice_date, 'Invoice'
FROM Invoices
ORDER BY Order_date
```

أما في كود **SQL** التالي:

```
SELECT Customer_no, Charge_date AS Code, Order_date,
'Order' AS Relationship
FROM Orders
UNION
```

```
SELECT Customer_no, Invoice_total AS Code, Invoice_date,
'Invoice'
FROM Invoices
ORDER BY Order_date
```

فقد قمنا بتضمين حقلين بنوعى بيانات مختلفين، الأول هو الحقل Charge_date ويحتوى على بيانات من النوع datetime، والثاني هو الحقل Invoice_total ويحتوى على بيانات من النوع float، وهذا ما يدعمه Access حيث يقوم بإجراء تحويلات أنواع البيانات متى أمكن ذلك. أما داخل SQL Server، فيجب أن تقوم بإجراء عملية التحويل بنفسك من خلال الدالة CAST() أو الدالة CONVERT() وإلا يتم استخدام نوع بيانات الحقل الأول (أى datetime) كنوع بيانات للحقل الثاني المصاحب فنحصل على النوع datetime بدلاً من القيمة float. فعلى سبيل المثال، للحصول على نفس نتيجة استعلام التوحيد السابق، يتم استخدام كود T-SQL التالى:

```
SELECT Customer_no, Charge_date AS Code, Order_date,
'Order' AS Relationship
FROM Orders
UNION
SELECT Customer_no, CAST(Invoice_total AS float) AS Code,
Invoice_date, 'Invoice'
FROM Invoices
```

كما لا يمكنك فى استعلامات التوحيد بكود T-SQL استخدام العبارة ORDER BY، وإنما يمكنك تعيين حقول الفرز من خلال الخاصية Order By بصفحة الخصائص.

للتعرف على نتيجة تنفيذ الكود السابق، انقر الزر  ثم قم بالانتقال إلى طريقة عرض الاستعلام (انظر شكل ١١-١٧)، حيث تظهر نفس النتيجة سواء مع كود SQL أو كود

.T-SQL

Customer_r	Code	Order_date	Relationship
12342	29/03/2007	24/03/2007	Order
123410	1100.24	07/05/2008	Invoice
123410	12/10/2006	06/10/2006	Order
123410	25/08/2006	22/08/2006	Order
123410	25/10/2006	19/10/2006	Order
123410	30/09/2006	21/09/2006	Order
123411	14/10/2006	07/10/2006	Order
123411	23/11/2006	20/10/2006	Order
123411	28/09/2006	22/09/2006	Order
123411	31/08/2006	23/08/2006	Order
123411	604.91	07/05/2008	Invoice
123412	04/10/2006	23/09/2006	Order
123412	19/10/2006	10/10/2006	Order
123412	23/09/2006	24/08/2006	Order
123412	2615.58	07/05/2008	Invoice
123412	28/10/2006	21/10/2006	Order
123413	12/09/2006	25/08/2006	Order
123413	18/10/2006	10/10/2006	Order

شكل ١١-١٧ استعمال التوحيد بطريقة عرض البيانات.

يمكنك استخدام استعلامات التوحيد في إضافة الخيار (All) أو أي خيارات أخرى عند تعبئة مربعات السرد ومربعات السرد والتحرير باستخدام استعلام تحديد. ففي عبارة SQL التالية على سبيل المثال يتم إضافة (All) إلى نتيجة الاستعلام المستخدم في تعبئة مربع سرد وتحرير مستخدم في اختيار أوامر الصرف من دولة معينة أو جميع الدول (الخيار All في هذه الحالة):

```
SELECT City FROM Customers
UNION SELECT '(All)'
FROM Customers
ORDER BY City
```

والسبب في استخدام الأقواس حول كلمة All هو جعل هذا الخيار في بداية العناصر الناتجة من الاستعلام.

تنفيذ الاستعلامات الفرعية

يمكنك داخل SQL و T-SQL كتابة استعلام تحديد يحتوي بداخله على استعلام تحديد آخر داخل الجزء WHERE وبالتالي نحصل على ما كان يعرف في الإصدارات السابقة من

Access بالاستعلامات المتداخلة. وتحتوى الاستعلامات الفرعية على الصيغة العامة التالية:

```
SELECT field_list
FROM table_list
WHERE [table_name.]field_name
IN (SELECT select_statment
    [GROUP BY group_criteria]
    [HAVING aggregate_criteria]
    [ORDER BY sort_criteria] )
[ORDER BY sort_criteria]
```

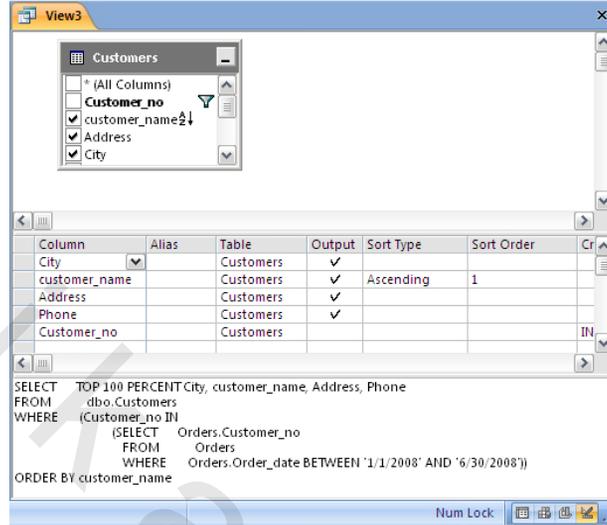
فعلى سبيل المثال، يوضح كود T-SQL التالى استعلاماً فرعياً بسيطاً يقوم بإرجاع أسماء وعناوين العملاء الذين لهم طلبات صرف في الفترة من ٢٠٠٨/١/١ إلى ٢٠٠٨/٦/٣٠:

```
SELECT Customers.City, Customers.Customer_name,
    Customers.Address, Customers.Phone
FROM Customers
WHERE Customers.Customer_no
    IN (SELECT Orders.Customer_no
        FROM Orders
        WHERE Orders.Order_date
            BETWEEN '1/1/2008' AND '6/30/2008')
ORDER BY Customers.Customer_name
```

في هذا الكود، يقوم الاستعلام الفرعي (الذي يبدأ بعد كلمة IN) بإرجاع قيم العمود Customer_no من الجدول Orders ومقارنتها بقيم العمود Customer_no بالجدول Customers.

وعلى عكس استعلامات التوحيد، يدعم عرض تصميم الاستعلامات داخل Access ومصمم المشروعات داخل SQL Server التصميم الرسومي للاستعلامات الفرعية، حيث يتم كتابة IN متبوعاً بعبارة SELECT الفرعية بالعمود Criteria داخل قوسين (انظر شكل ١٢-١٧).

البرمجة المتقدمة باستخدام قاعدة البيانات Access 2007



شكل ١٢-١٧ تمثيل الاستعلامات الفرعية من خلال مصمم المشروعات.

يوضح شكل ١٣-١٧ نتيجة تنفيذ عبارة T-SQL السابقة.

اسم العميل	العنوان	المدينة	رقم الهاتف
مركز مرسيديس	5 ميدان لاطوعلی	الغاهرة	225612540
رباب الشامي	43 شارع الميرغني - كلية البنات	أسوان	978856311
منير وحسين	65 شارع لبنان	أسوان	978856333
نجوى ابراهيم	شارع علي عبدالرازق	أسوان	978852145
علي الصمدی	الطريق الدائري	بسيون	508987890
مها	543 شارع المنصورة	سنهور	508741523
محمد مرسی	23 شارع وليد قاسم	فايد	504875210
ناسيم	شارع ابن المقفع	بسيون	508743269
خاطر	2 شارع أكتوبر	الغاهرة	225351400
نادية	ميدان عبدالمنعم رياض	الشين	402556987
فاطمة نبيل محمد	33 طريق العروبة	فايد	508974561
ابراهيم خليل	64 شارع النزهة	أسوان	978863625
مكتبة كريم علي	10 شارع البحر	أسوان	978865214
حسام هيثم	3 ميدان الحلمية	أسوان	978863255
ليان	21 شارع اللواء محمود حمدي	الإسكندرية	34569821
لامبو	35 شارع رمسيس	ديروط	932569874
جيمي	12 شارع الأهر	قليوب	509632556
عبدالفتاح نبيل محمد	21 شارع اللواء محمود حمدي	السويس	625594174
	67 شارع حسين الراوي	القصاصين	523144581

شكل ١٣-١٧ نتيجة تنفيذ عبارة T-SQL السابقة.

