وفي كل مرة تود إضافة عنصر إلى الطلب، فالإجراءOrders.AddItemToOrder تحت الخدمة:

كود:

```
ALTER PROCEDURE [Orders.AddItemToOrder]
(
    @itmOrderID uniqueidentifier,
    @itmProductID uniqueidentifier,
    @itmQuantity smallint
)
AS
    SET NOCOUNT ON
    INSERT INTO Items
    (
        itmOrderID,
        itmProductID,
        itmQuantity
    )
    VALUES
    (
        @itmOrderID,
        @itmProductID,
        @itmQuantity
    )
    RETURN
```

الإجراء Orders.RemoveItemFromOrder يحذف احد العناصر من الطلب، ويمكنك معرفة جميع العناصر التابعة للطلب من خلال الإجراء Orders.GetAllItemInOrder. أخيرا، بالنسبة للإجراء Orders.DeleteOrder سيقوم بحذف الطلب وجميع العناصر التابعة له في جدول Items :

كود:

```
ALTER PROCEDURE [Orders.DeleteOrder]
(
    @ordID uniqueidentifier
)
AS
    SET NOCOUNT ON
    DELETE FROM
        Orders
    WHERE
        ordID = @ordID

    DELETE FROM
        Items
    WHERE
        itmOrderID = @ordID
RETURN
```

قد تتساءل عن سبب قيامي بحذف العناصر في جدول Items يدويا، فالمتوقع إن يقوم محرك قاعدة البيانات بحذفها تلقائيا بسبب العلاقة، ولكن في الحقيقة لا توجد علاقات بين الجداول فلا أحبذ استخدامها كثيرا بسبب أنها تضعف كفاءة التنفيذ، وبالنسبة للعلاقات الواضحة بالصورة فهي للتوضيح فقط (كما يقول المثل: لا تصدق كل ما تراه).

خاتمة

التطبيقات متعددة الطبقات N-Tier Applications ليست سوى أسلوب لتصميم معماري Architecture Design يتبعه مصممو التطبيقات، الغرض منه تقسيم التطبيق إلى طبقات مستقلة أكثر تنظيما وقابلية للصيانة. في المقال القادم سنكمل ورشة العمل لبناء تطبيق ثلاثي الطبقات، والى ذلك الحين يمكنك إنزال شفرة قاعدة البيانات Database Script (اضغط على الجملة المسطرة) شاملة للجداول وإجراءات Stored Proceduresالتي صممناها للتو.