



التحسينات على T-SQL

Transact-SQL (T-SQL).

SQL Server 2005

.

.2005

T-SQL

التحسينات على T-SQL

SQL Server

Transact-SQL

:

.

:(SNAPSHOT)

❖

.

:

❖

(service broker)

DML DDL

:

❖

IMAGE TEXT

:

❖

DDL

:DDL

❖

❖ (Common Table Expressions) :

.

❖ :

❖ PIVOT :

❖ APPLY JOIN .XML

❖ TOP :

❖ TRY/CATCH :

عزل اللقطة

SQL Server

ACID :

- -

.

(atomicity)

(consistency)

:

(Isolation)

(durability)

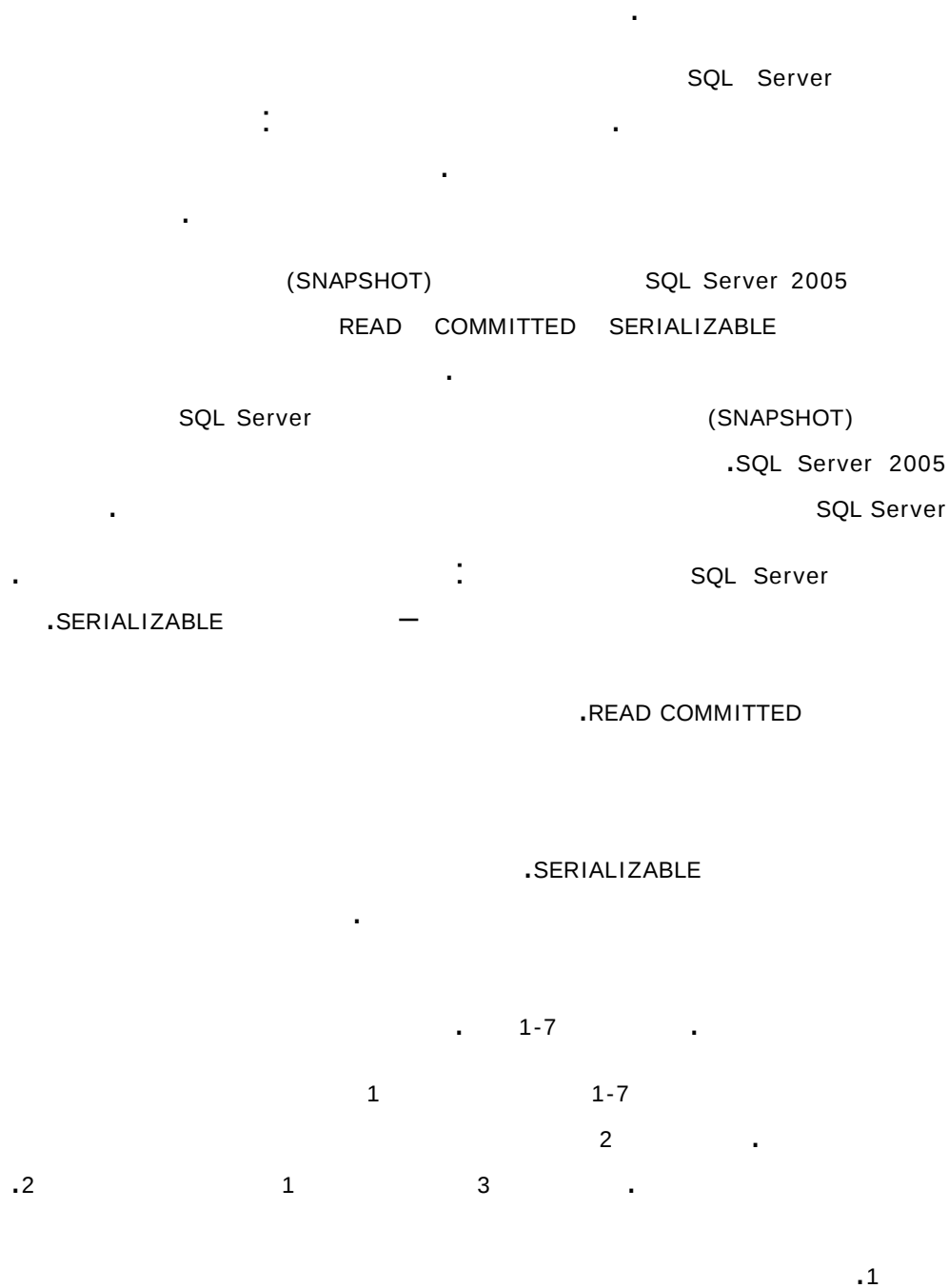
READ REPEATABLE :

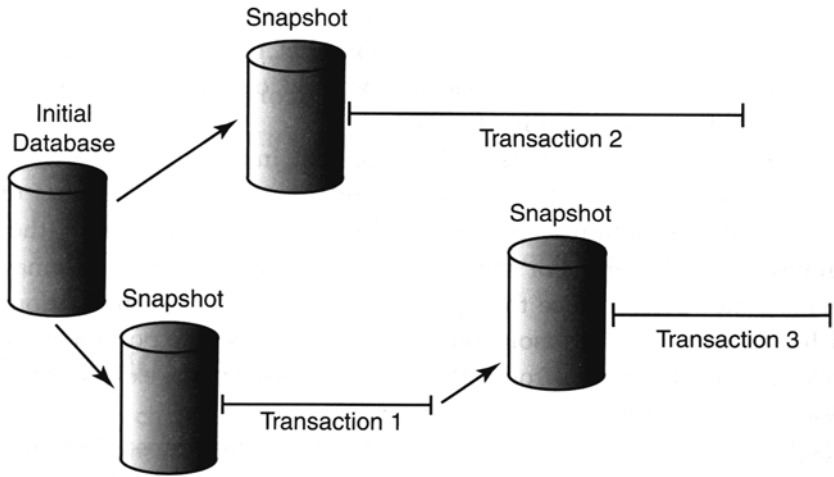
SQL Server 2005

.SERIALIZABLE READ COMMITED READ UNCOMMITTED

(SERIALIZABLE)

:-



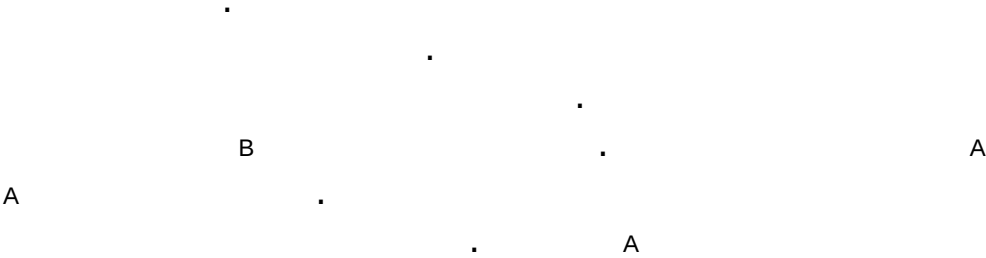


الشكل 1-7: إنشاء لقطة لقاعدة المعطيات

SERIALIZABLE

SQL Server

row versioning : (:)



READ COMMITTED

1-7 .name ID : (tab .READ COMMITTED

الجدول 7-1: تعدد إصدارات قاعدة المعطيات في عزل READ COMMITTED

الخطوة	المستخدم الأول	المستخدم الثاني
1	BEGIN TRAN SELECT name FROM tab WHERE id = 1 **value is 'Name'	
2		BEGIN TRAN UPDATE tab SET name = 'Newname' WHERE id = 1
3	SELECT name FROM tab WHERE id = 1 **value is 'Name'	
4		COMMIT
5	SELECT name FROM tab WHERE id = 1 **value is 'NewName'	
6	COMMIT	
7	SELECT name FROM tab WHERE id = 1 **value is 'NewName'	

— SERIALIZABLE —

1-7.

2-7

SERIALIZABLE.

الجدول 7-2: تعدد إصدارات قاعدة المعطيات في العزل SERIALIZABLE

الخطوة	المستخدم الأول	المستخدم الثاني
1	BEGIN TRAN SELECT name FROM tab WHERE id = 1 **value is 'Name'	
2		BEGIN TRAN UPDATE tab SET name = 'Newname' WHERE id = 1

الخطوة	المستخدم الأول	المستخدم الثاني
3	SELECT name FROM tab WHERE id = 1 **value is 'Name'	
4	COMMIT	
5	SELECT name FROM tab WHERE id = 1 **value is 'Name'	
6	COMMIT	
7	SELECT name FROM tab WHERE id = 1 **value is 'NewName'	

2 1-7
 — SERIALIZABLE — 1
 . "transactrion-level SNAPSHOT isolation " " SQL Server

SQL .ALTER DATABASE SNAPSHOT
 .pubs

ALTER DATABASE pubs
 SET ALLOW_SNAPSHOT_ISOLATION ON

SNAPSHOT SQL .
 ALTER DATABASE pubs
 SET ALLOW_SNAPSHOT_ISOLATION ON
 GO
 USE pubs
 GO
 SET TRANSACTION ISOLATION LEVEL SNAPSHOT
 BEGIN TRAN
 — SQL Expressions
 COMMIT TRAN

```

SQL
.BEGIN TRANSACTION

.READ_COMMITTED_SNAPSHOT
SNAPSHOT_ISOLATION
READ UNCOMMITTED
READ COMMITTED
READ COMMITTED
SQL
.
.

-- alter the database
ALTER DATABASE pubs
SET ALLOW_SNAPSHOT_ISOLATION ON
SET READ_COMMITTED_SNAPSHOT ON
GO
USE pubs
GO
SET TRANSACTION ISOLATION LEVEL READ UNCOMMITTED
BEGIN TRAN
-- SQL expression will be executed as READ COMMITTED using versioning
END TRAN
SET TRANSACTION ISOLATION LEVEL READ COMMITTED
BEGIN TRAN
-- SQL expression will be executed as READ COMMITTED using versioning
END TRAN

(ON) ALLOW_SNAPSHOT_ISOLATION
.DATABASEPROPERTYEX (OFF)
.
: SnapshotIsolationFramework

SELECT DATABASEPROPERTYEX ('pubs', 'SnapshotIsolationFramework')

.SNAPSHOT
( ) TEMPDB
(timestamp)
TEMPDB
SQL Server
SQL Server
.
```

SQL Server

.TEMPDB

3-7

- snaptest

- it is necessary to run
 - SET ALLOW_SNAPSHOT_ISOLATION ON
 - if that's not done already
 CREATE TABLE snapTest ([id] INT IDENTITY,
 col1 VARCHAR(15))

 - insert some data
 INSERT INTO snapTest VALUES(1, 'Niels')

الجدول 3-7: مثال عن عزل اللقطة

الخطوة	المستخدم الأول	المستخدم الثاني
1	SET TRANSACTION ISOLATION LEVEL SNAPSHOT BEGIN TRAN UPDATE snapTest SET col1 = 'NewNiels' WHERE id = 1	
2	SET TRANSACTION ISOLATION LEVEL SNAPSHOT BEGIN TRAN SELECT col1 FROM snapTest WHERE id = 1 ** receives value 'Niels'	
3	COMMIT TRAN	
4	SELECT col1 FROM snapTest WHERE id = 1 ** receives value 'Niels'	
5	COMMIT TRAN	
6	SELECT col1 FROM snapTest WHERE id = 1 ** receives value 'NewNiels'	

3-7 :

SQL Server . ❖

.TEMPDB

sp-lock . ❖

(READ COMMITTED) SQL Server

TEMPDB SQL Server

."Niels" ❖

() SELECT 2 ❖

."Niels"

2 ❖

() SELECT 2 ❖

."NewNiels"

.SQL Server

SQL Server

SQL Server 2005 .

2005 .

: — —

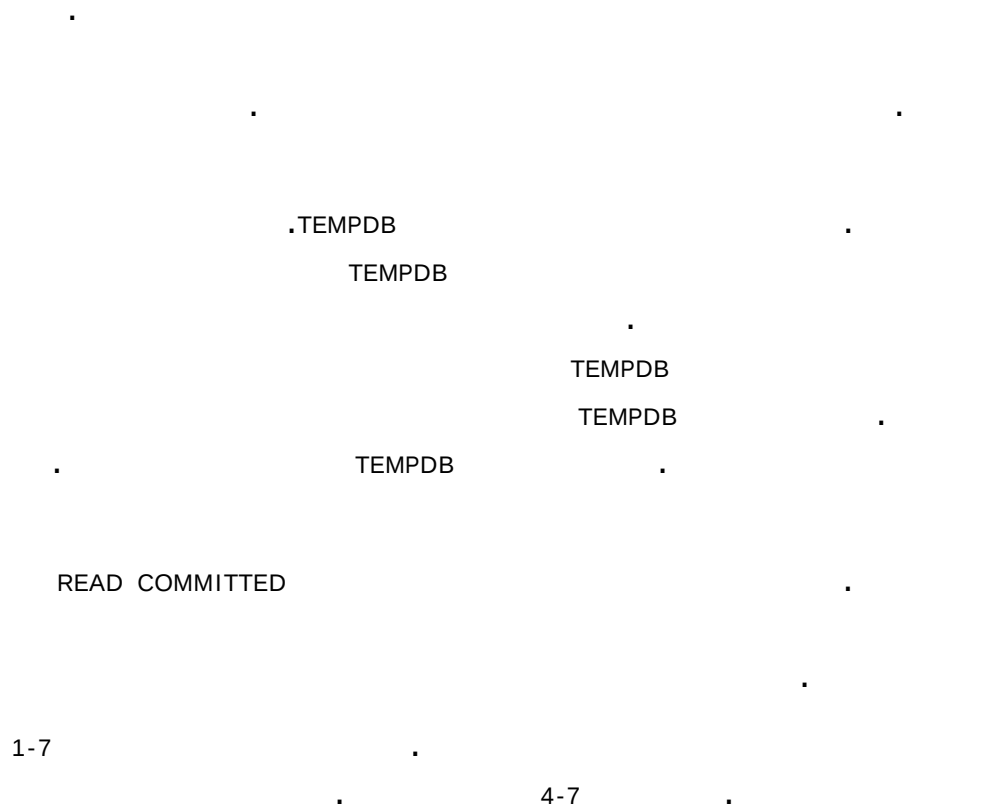
.READ_COMMITTED_SNAPSHOT SNAPSHOT

SQL Serve .

MASTER

.MSDB

عقابيل تعدد الإصدارات



الجدول 4-7: تعدد إصدارات قاعدة المعطيات ضمن عزل SERIALIZABLE – التعديلات المتزامنة

الخطوة	المستخدم الأول	المستخدم الثاني
1	BEGIN TRAN SELECT name FROM tab WHERE id = 1 **value is 'Name'	
2	BEGIN TRAN UPDATE tab SET name = 'Newname' WHERE id = 1	
3	COMMIT	

SQL Server
 READ_COMMITTED_SNAPSHOT
 SQL Server REPEATABLE READ
 SELECT SQL Server ANSI SELECT FOR UPDATE
 .REPEATABLE READ
 .UPDLOCK
 .SQL Sever 2005

مراقبة تعدد الإصدارات

TEMPDB
 T-SQL
 . SNAPSHOT :DATABASEPROPERTYEX ❖
 . :sys.fn_top_version_generators() ❖
 SNAPSHOT :sys.fn_transaction_snapshot() ❖
 :sys.fn_transaction_generators() ❖
 :
 .() ❖
 .() ❖
 .() TEMPDB ❖
 .() ❖

❖ () SNAPSHOT .

SNAPSHOT .

SQL Profiler .SNAPSHOT .

إعادة الترجمة على مستوى التعليمة

T-SQL :

SQL Server 2000 .(statement recompilation)

:(query plan architecture)

.(executable) (compiled)

❖ :

() .

-

.

❖ :

-

-

.

-

-

(batches)

.

:

❖ .

❖ .

❖ SET .

SQL Server 2000

SET

1-7

.CONCAT_NULL_YIELDS_NULL

1-7

SQL Server 2000

السرد 1-7: إجرائية تسبب إعادة ترجمة

```
CREATE PROCEDURE test2
AS
Select 'before set option'
-// change a set option
SET CONCAT_NULL_YIELDS_NULL OF

SELECT 'after set option'
```

: SQL Server 2000

.1-7 .1

.Trace New File SQL Server Profiler .2

.Events Trace Properties .3

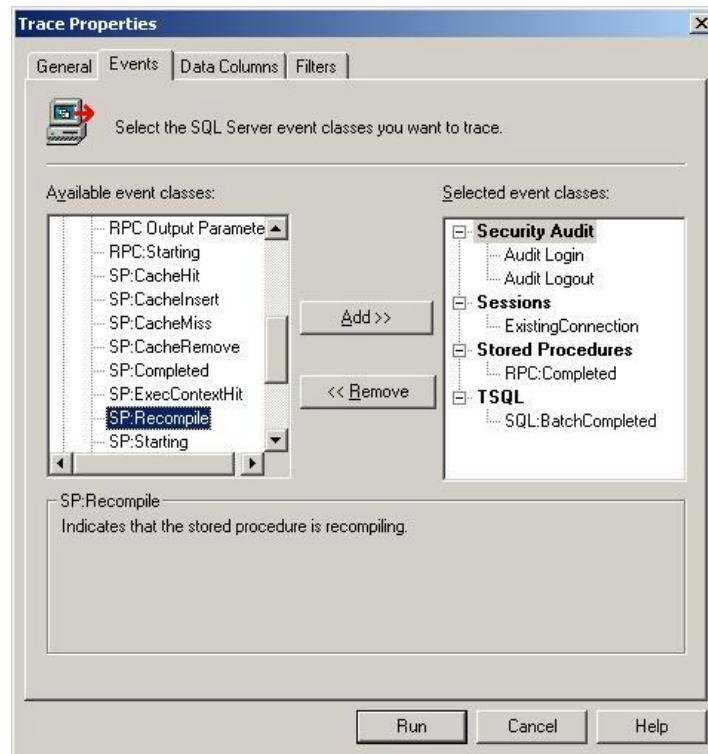
SP: Recowpile Stored Procedures .4

.Run 2-7 Add

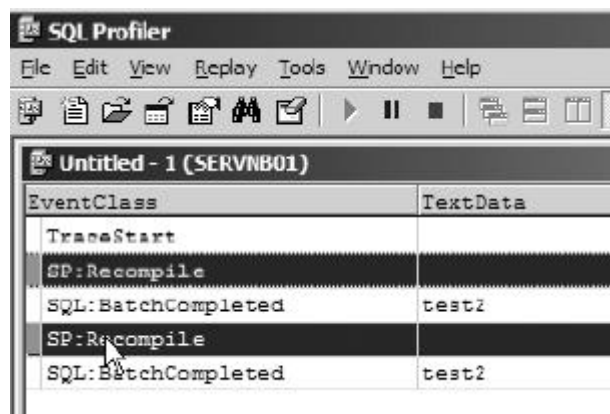
Query Anelyzer

sp:Recompile EventClass

.3-7



الشكل 7-2: مربع الحوار Trace Properties في SQL Server Profiler.



الشكل 7-3: خرج تعقب الأثر.

SQL Server 2005

SQL Server 2005

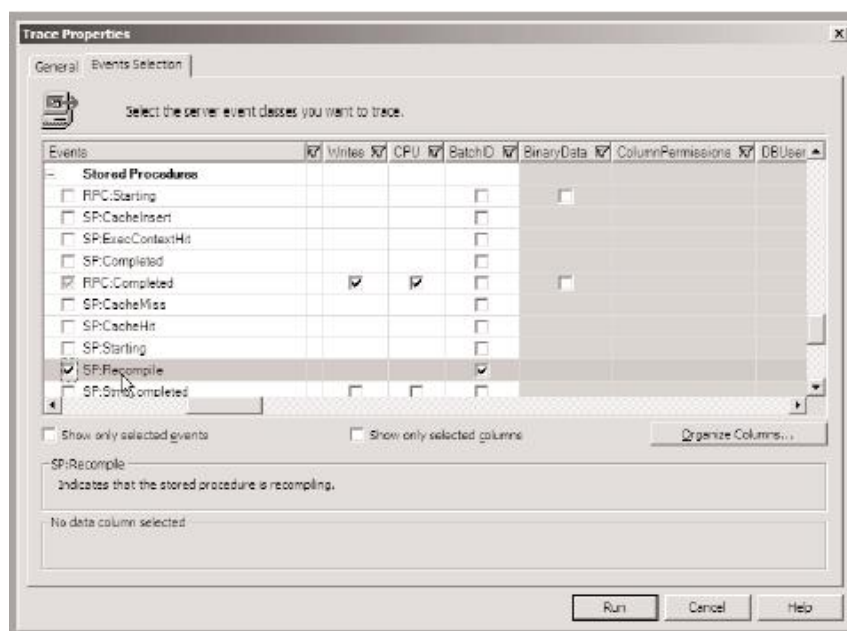
2005 2000

SQL Server 2005

1-7

SQL Server 2005

```
SELECT * FROM syscacheobjects
WHERE dbid = db_id('pubs')
AND objid = object_id('test2')
```



الشكل 4-7: مربع الحوار Trace Properties في SQL Server 2005

.dbcc freeproccache
 .SQL Server 2000 SQL Profiler
 .Trace Properties Event Selection 4-7

DBA

قواعد DDL

() SQL
 SQL Server .
 — 2005 T-SQL
 DML .NET
 (DDL)
 SQL Server 2005 .SQL Server 2005 —
 .DML DDL
 DDL 2-7 DDL
 .DML — .NET

السرد 2-7: صياغة قاعد DDL

```
CREATE TRIGGER trigger_name
ON { ALL SERVER | DATABASE }
[ WITH ENCRYPTION ]
{ FOR | AFTER } { event_type [ , ...n ] | DDL_DATABASE_LEVEL_EVENTS }
[ WITH APPEND ]
[ NOT FOR REPLICATION ]
{ AS
```

```
{ sql_statement [ ...n ] | EXTERNAL NAME < method specifier > }
}
< method_specifier > ::=
    assembly_name:class_name[::method_name]
```

```

        .                                DML                                DDL
        (ALL SEVRER)                                DDL        ON        ❖
                                (DATABASE)
                                INSTEAD OF        DDL                                ❖
                                event_type                                ❖
                                .
                                .DDL_DATABASE_LEVEL_EVENTS
                                SQL Server Books Online        DDL
.DDL_DATABASE_LEVEL_EVENTS                                event_type
                                .                                DDL
                                .(log)
-first create a table to log to
CREATE TABLE ddlLog (id INT PRIMARY KEY IDENTITY,
    logTxt VARCHAR(MAX))
GO

-create our test table
CREATE TABLE triTest (id INT PRIMARY KEY)
GO

- create the trigger
CREATE TRIGGER ddlTri
ON DATABASE
AFTER DROP_TABLE
AS
INSERT INTO ddlLog VALUES('table dropped')

-                                VARCHAR (MAX)
                                .
```

(ON DATABASE)
 .(ON DROP TABLE)
 .
 DROP TABLE tritest
 SELECT * FROM ddllog
 ddllog
 DROP TABLE
 . SELECT
 .
 DDL
 DML .
 deleted inserted
 .
 .eventdata .DDL
التابع Eventdata
 . DDL eventdata()
 .(XSD) XML XML
 : XSD
 . ❖
 . SPID ❖
 . ❖
 . ❖
 SQL Server Books online eventdata
 .
 : DROP TABLE
 ❖ Database
 ❖ Schema
 ❖ Object

- ❖ ObjectType
- ❖ TSQLCommand

.ddllog eventdata 3-7

.DDL

السرد 3-7: تعديل القادح لكي يستخدم التابع eventdata

```

- alter the trigger
ALTER TRIGGER ddlTri
ON DATABASE
AFTER DDL_DATABASE_LEVEL_EVENTS
AS
INSERT INTO ddlLog VALUES CONVERT(VARCHAR(max)eventdata())

```

4-7

```

-delete all entries in ddlLog
DELETE ddlLog

-create a new table
CREATE TABLE evtTest (id INT PRIMARY KEY)

-select the logTxt column with the XML
SELECT logTxt
FROM ddlLog

```

السرد 4-7: خرج التابع eventdata

```

<EVENT_INSTANCE>
  <PostTime>2004-01-30T11:58:47.217</PostTime>
  <SPID>57</SPID>
  <EventType>CREATE_TABLE</EventType>
  <ServerName>ZMV44</ServerName>
  <LoginName>ZMV44\Administrator</LoginName>
  <UserName>ZMV44\Administrator</UserName>
  <DatabaseName>pubs</DatabaseName>
  <SchemaName>dbo</SchemaName>
  <ObjectName>foo</ObjectName>
  <ObjectType>TABLE</ObjectType>

```

```

<TSQLCommand>
  <SetOptions ANSI_NULLS="ON" ANSI_NULL_DEFAULT="ON"
    ANSI_PADDING="ON" QUOTED_IDENTIFIER="ON"
    ENCRYPTED="FALSE" />
  <CommandText>
    CREATE TABLE evtTest (id int primary key)
  </CommandText>
</TSQLCommand>
</EVENT_INSTANCE>

```

XQuery

XML

.

EventType

.

.ddllog

eventdata

CommandText Object

.

Xquery

XML

```

DECLARE @data XML

```

```

SELECT @data = logTxt FROM ddlLog

```

```

WHERE id = 11

```

```

SELECT

```

```

  CONVERT(NVARCHAR(100),

```

```

    @data.query('data(//EventType)')) EventType,

```

```

  CONVERT(NVARCHAR(100),

```

```

    @data.query('data(//Object)')) Object,

```

```

  CONVERT(NVARCHAR(100),

```

```

    @data.query('data(//TSQLCommand/CommandText)')) Command

```

```

  XQuery XML

```

.

```

    XQuery XML

```

DDL

DML

DDL

DML

.

.

.

SQL Server 2005

.

```

  .(event notifications)

```

:

.

إشعارات الأحداث

SQL Server (SSB) SQL Server
1 (Server service Broker)

DDL DML

:

```
CREATE EVENT NOTIFICATION event_notification_name
ON { SERVER | DATABASE |
[ ENABLED | DISABLED ]
{ FOR { event_type |
DDL_DATABASE_LEVEL_EVENTS } [ ,...n ]
TO broker_service
```

: DDL

. :event_notification_name ❖

:SERVER ❖

:DATABASE ❖

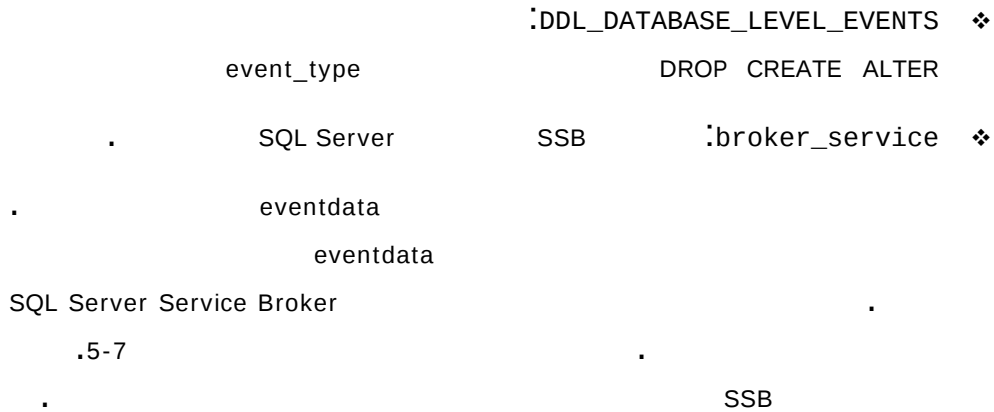
.CREATE :ENABLED ❖

ALTER :DISABLED ❖

EVENT NOTIFICATION

- - :event_type ❖

.SQL Server Books Online event-type



السرد 5-7: خطوات إنشاء مثل من وسيط الخدمات

```

--first we need a queue
CREATE QUEUE queue evtDdlNotif
WITH STATUS = ON

```

```

--then we can create the service
CREATE SERVICE evtDdlService
ON QUEUE evtDdlNotif

```

```

--this is a MS supplied contract

```

```

--which uses an existing message type

```

```

--{http://schemas.microsoft.com/SQL/Notifications}EventNotification
(http://schemas.microsoft.com/SQL/Notifications/PostEventNotification)

```



```
CREATE EVENT NOTIFICATION ddlEvents
ON DATABASE
FOR DDL_DATABASE_LEVEL_EVENTS
TO SERVICE evtDdlService
```

SQL Server Management Studio

```
( ) RECEIVE WAITFOR
:

WAITFOR (
RECEIVE * FROM evtDdlNotif)

WAITFOR DDL
WAITFOR CREATE TABLE evtNotifTbl(id int) .
.20 message_type_id
{http://schemas.microsoft.com/SQL/Notification}EventNotification :
.message_body eventdata
. WAITFOR

DECLARE @msgtypeid INT
DECLARE @msg VARBINARY(MAX)

WAITFOR(
RECEIVE TOP(1)
@msgtypeid = message_type_id,
@msg = message_body
FROM evtDdlNotif
)
--check if this is the correct message type
IF @msgtypeid = 20
BEGIN
--do something useful WITH the message
--here we just select it as a result
SELECT CONVERT(NVARCHAR(MAX), @msg)
END
```

CREATE TABLE

.4-7

.DDL

.SQL Server

```
CREATE EVENT NOTIFICATION loginEvents ON SERVER
FOR audit_login TO SERVICE evtLoginService
```

ON SERVER

eventdata 6-7

السرد 6-7:

```
<EVENT_INSTANCE>
  <PostTime>2003-06-29T09:46:23.623</PostTime>
  <SPID>51</SPID>
  <EventType>AUDIT_LOGIN</EventType>
  <ServerName>ZMV44</ServerName>
  <LoginName>ZMV44\Administrator</LoginName>
  <UserName>ZMV44\Administrator</UserName>
  <Database>eventstest</Database>
  <!-- additional information elided -->
</EVENT_INSTANCE>
```

VARCHAR (MAX)

أنماط المعطيات الضخمة

SQL Server 2000 (7) VARCHAR

8000 VARBINARY NVARBINARY NVARCHAR 4000

TEXT (Large Objects LOBs) IMAGE NTEXT

231 .MAX SQL Server 2005

. 230 Unicode

VARCHAR (MAX) NVARCHAR (MAX)

. VARBINARY (MAX)

- (Large Value data types)

.

```
CREATE TABLE largeValues (
  lVarchar VARCHAR(MAX),
  lnVarchar NVARCHAR(MAX),
  lVarbinary VARBINARY(MAX)
)
```

. LOBs

— .

. (concatenation)

```
CREATE FUNCTION dovmax(@in VARCHAR(MAX))
RETURNS VARCHAR(MAX)
AS
BEGIN
  — supports concatenation
  RETURN @in + '12345'
END
```

VARCHAR (MAX) SQL Server

.NVARCHAR (MAX)

() .SUBSTRING

VARBINARY NVARCHAR VARCHAR

.

. LOBs LOBs

(8000 256) SQL Server 7 VARCHAR

8000 VARCHAR

7 2000 SQL Server .TEXT

2005 . (8060)

.

التحسينات على لغة T-SQL

SQL Server CLR

. - T-SQL

.

الكلمة المفتاحية TOP

2005 .SQL Server 7 TOP
 .SELECT ()
 DELETE INSERT UPDATE TOP SQL Server 2005
 .TOP (expression) [percent] :
 .DELETE UPDATE INSERT TOP
 .TOP

```
--create a table and insert some data
CREATE TABLE toptest (col1 VARCHAR(150))
INSERT INTO toptest VALUES('Niels1')
INSERT INTO toptest VALUES('Niels2')
INSERT INTO toptest VALUES('Niels3')
INSERT INTO toptest VALUES('Niels4')
INSERT INTO toptest VALUES('Niels5')
```

```
--this returns 'Niels1' and 'Niels2'
SELECT TOP(2) * FROM toptest
```

```
--this sets 'Niels1' and 'Niels2' to 'hi'
UPDATE TOP(2) toptest SET col1 = 'hi'
SELECT * FROM toptest
```

```
--the two rows with 'hi' are deleted
DELETE TOP(2) toptest
SELECT * FROM toptest
```

```
--create a new table and insert some data
CREATE TABLE toptest2 (col1 VARCHAR(150))
INSERT INTO toptest2 VALUES('Niels1')
INSERT INTO toptest2 VALUES('Niels2')
INSERT INTO toptest2 VALUES('Niels3')
INSERT INTO toptest2 VALUES('Niels4')
INSERT INTO toptest2 VALUES('Niels5')
```

```
--'Niels1' and 'Niels2' are inserted
INSERT top(2) toptest
SELECT * FROM toptest2
```

```
SELECT * FROM toptest
```

```
SQL Server 2005 TOP
) .
.(
```

```
--declare 3 variables
DECLARE @a INT
DECLARE @b INT
DECLARE @c INT
```

```
--set values
SET @a = 10
SET @b = 5
SELECT @c = @a/@b
```

```
--use the calculated expression
SELECT TOP(@c)* FROM toptest
```

```
--insert some more data in toptest
INSERT INTO toptest VALUES('Niels6')
INSERT INTO toptest VALUES('Niels7')
INSERT INTO toptest VALUES('Niels8')
```

```
--use a SELECT statement as expression
--this should return 5 rows
SELECT TOP(SELECT COUNT(*) FROM toptest2) *
FROM toptest
```

.SQL Server

T-SQL

.OUTPUT


```

DELETED    INSERTED                UPDATE    OUTPUT
          INSERTED                DELETED    .
          .                OUTPUT    .

```

```

--update records, this populates
--both the inserted and deleted tables
DECLARE @changes TABLE
(id INT, oldValue VARCHAR(15), newValue VARCHAR(15))
UPDATE outputtbl
SET col1 = 'updated'
OUTPUT inserted.id, deleted.col1, inserted.col1
INTO @changes
WHERE id < 5
SELECT * FROM @changes
GO
id            oldValue            newValue
-----
3            row5                updated
4            row6                updated

(2 row(s) affected)

```

تعايير الجداول المشتركة والاستعلامات العودية.

(Common Table Expression: CTE)

```

:                CTE    .

```

```

[WITH <common_table_expression> [,...n] ]
<common_table_expression>::=
    expression_name
    [(column_name [,...n])]
AS
    (<CTE_query_expression>)

```

```

.                CTE                SQL

```

```

WITH MathConst(PI, Avogadro)
AS
(SELECT 3.14159, 6.022e23)
SELECT * FROM MathConst
GO

```

```

PI
-----
3.14159
(1 row(s) affected)

MathConst      .      WITH
.AS            SELECT  .Avogadro PI
                WITH    .MathConst      SELECT
                        CTE              .
                MathConst      .CTE
                        .CTE

                SQL      .      CTE
                .
                CTE

WITH MathConst(PI, Avogadro)
AS
(SELECT 3.14159, 6.022e23),
-- second table
Package(Length, Width)
AS (SELECT 2, 5)
SELECT * FROM MathConst, Package
PI      Avogadro      Length      Width
-----
3.14159      6.022E+23      2      5
(1 row(s) affected)

                .      CTE

                CTE

                .

                SQL      CTE

                CTEs      :      -

                .      SQL Server      SQL-92

                .      SQL Server      CTEs

```

CTEs

— AdventureWorks

```

SalesHeader SalesPerson : .SQL Server
SalesPerson .SalesDetail
SalesDetail SaleHeader .AdventureWorks
(ID) SalesHeader .
SalesDetail .
. 90
"MakeCall" (ID)
.90
. "Relax"

```

CTE

.90

```

SELECT DISTINCT SH.SalesPersonId FROM SalesOrderHeader SH JOIN
SalesOrderDetail SD ON SH.SalesOrderId = SD.SalesOrderId
AND SD.ProductID = 90
SalesPersonId
GO
SalesPersonId
- - - - -
14
21
22
. . .
more rows
(14 row(s) affected)

```

:

Action	SalesPersonID	SalesQuota	Value
MakeCall	22	250000.0000	2332.7784
... more lines			
Relax	35	250000.0000	0

90 .

"Relax" : . "MakeCall" :

.0

CTEs

: SQL — 90

```

SELECT 'MakeCall' AS Action, S.SalesPersonID, S.SalesQuota,
(SELECT SUM(SD.UnitPrice * SD.OrderQty) FROM SalesOrderHeader SH
JOIN SalesOrderDetail SD ON
SH.SalesOrderId = SD.SalesOrderId
AND SD.ProductID=90 AND SH.SalesPersonID=S.SalesPersonID
)
FROM SalesPerson S
WHERE EXISTS
(
SELECT * FROM SalesOrderHeader SH JOIN SalesOrderDetail SD ON
SH.SalesOrderID = SD.SalesOrderID AND SD.ProductID = 90
AND SH.SalesPersonID = S.SalesPersonID
)
UNION
SELECT 'Relax' AS Action, S.SalesPersonID, S.SalesQuota, 0
FROM SalesPerson S
WHERE NOT EXISTS
(
SELECT * FROM SalesOrderHeader SH JOIN SalesOrderDetail SD ON
SH.SalesOrderID = SD.SalesOrderID AND SD.ProductID = 90
AND SH.SalesPersonID = S.SalesPersonID
)

```

—

.

:CTE

```

WITH Missing(SP, AMT)
AS(
SELECT SH.SalesPersonID, SUM(SD.UnitPrice * SD.OrderQty) FROM
SalesOrderHeader SH
JOIN SalesOrderDetail SD ON SH.SalesOrderId = SD.SalesOrderId
AND SD.ProductID=90 GROUP BY SH.SalesPersonID

```

```
)
SELECT 'MakeCall' AS Action, S.SalesPersonID, S.SalesQuota,
Missing.AMT
FROM Missing JOIN SalesPerson S ON Missing.SP = S.SalesPersonID
UNION
SELECT 'Relax' AS Action, S.SalesPersonID, S.SalesQuota, 0
FROM SalesPerson S WHERE S.SalesPersonID NOT IN (SELECT SP FROM
Missing)
```

CTE
Missing .

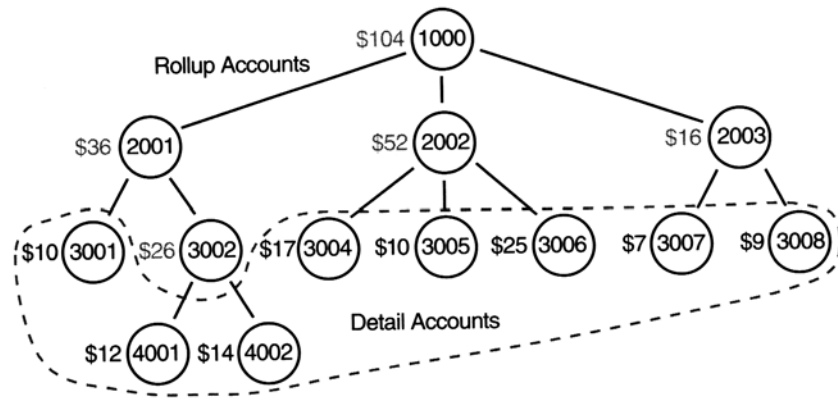
.
:
— — CTE .90
. Missing

.SQL:1999 SQL Server 2005 CTE
.(recuesive) :

.
.
:
5-7
.
4001 5-7 .
.\$12

. - - .
3002 5-7 .
.4002 4001 :

:
.
.()
.5-7 SQL



الشكل 5-7: مخطط حسابات

```

CREATE TABLE DetailAccount(id INT PRIMARY KEY,
parent INT, balance FLOAT)
CREATE TABLE RollupAccount(id INT PRIMARY KEY,
parent INT)
INSERT INTO DetailAccount VALUES (3001, 2001, 10)
INSERT INTO DetailAccount VALUES(4001, 3002, 12)
INSERT INTO DetailAccount VALUES(4002, 3002, 14)
INSERT INTO DetailAccount VALUES(3004, 2002, 17)
INSERT INTO DetailAccount VALUES(3005, 2002, 10)
INSERT INTO DetailAccount VALUES(3006, 2002, 25)
INSERT INTO DetailAccount VALUES(3007, 2003, 7)
INSERT INTO DetailAccount VALUES(3008, 2003, 9)
INSERT INTO RollupAccount VALUES(3002, 2001)
INSERT INTO RollupAccount VALUES(2001, 1000)
INSERT INTO RollupAccount VALUES(2002, 1000)
INSERT INTO RollupAccount VALUES(2003, 1000)
INSERT INTO RollupAccount VALUES(1000, 0)

```

5-7

: SQL

```

SELECT id, balance FROM Rollup - a handy view
id      balance
-----
1000    104
2001    36

```

2002	52
2003	16
3001	10
3002	26
3004	17
3005	10
3006	25
3007	7
3008	9
4001	12
4002	14

(13 row(s) affected)

```
SELECT id, balance FROM Rollup WHERE id = 2001
id      balance
- - - - - - - - - -
2001      36
```

(1 row(s) affected)

```

                                Rollup
(recursive)                    .
                                .
                                .
                                .(1000)
```

```
With Rollup(id, parent)
AS
(
  -- anchor
  SELECT id, parent FROM RollupAccount WHERE id=1000
  UNION ALL
  --recursive call
  SELECT R1.id, R1.parent FROM
  (
    SELECT id,parent FROM DetailAccount
    UNION ALL
    SELECT id,parent FROM RollupAccount
  )R1
  JOIN Rollup R2 ON R2.id = r1.parent
)
--selecting results
SELECT id,parent FROM Rollup
GO
id parent
```



```

        UNION ALL
        .Rollup :CTE
        .
        6-7 . Rollup .
        .
        . 1000
        .1000

        UNION ALL
        .RollupAccount DetailAccount
        : - Rollup SELECT CTE
        .CTE (UNION)
        .
        : -
        .( ) 1000
        )
        . SELECT .(1000
        .

WITH Rollup(id, parent, balance)
AS
(
    -- anchor
    SELECT id, parent, balance FROM DetailAccount
    UNION ALL
    -- recursive call
    SELECT R1.id, R1.parent, R2.balance
    FROM RollupAccount R1
    JOIN Rollup R2 ON R1.id = R2.parent
)
SELECT id, SUM(balance) balance FROM Rollup GROUP BY id
GO

```

id	balance
1000	104
2001	36
2002	52
2003	16
3001	10
3002	26
3004	17
3005	10
3006	25
3007	7
3008	9
4001	12
4002	14

(13 row(s) affected)

.

.

.

2001

.

:

:

id	balance
2001	14
2001	12
2001	10

4002 4001 3001

10 12 14

CTE

.2001

.SUM

.CTEs

.(cursor)

CTE

SQL

.

—

.2001

```
CREATE VIEW Rollup
AS
WITH Rollup(id, parent, balance)
AS
(
SELECT id, parent, balance FROM DetailAccount
UNION ALL
SELECT R1.id, R1.parent, R2.balance
FROM RollupAccount R1
JOIN Rollup R2 ON R1.id = R2.parent
)
SELECT id, SUM(balance) balance FROM Rollup GROUP BY id
GO

-- get the balance for account 2001
SELECT balance FROM rollup WHERE id = 2001
GO
```

```
balance
- - - - -
36
```

```
(1 row(s) affected)
```

CTE

SQL Server

```
.                .               100
-OPTION(MAXRECURSION 10) :      -
```

```
WITH Rollup(id, parent, balance)
AS
(
  -- body of CTE removed for clarity
)
SELECT id, SUM(balance) balance FROM Rollup GROUP BY id
OPTION (MAXRECURSION 10)
GO
```

العامل APPLY

OUTER APPLY CROSS APPLY :

T-SQL

JOIN

CROSS APPLY

SQL

.

ON

.

```

CREATE TABLE T1 (ID int)
CREATE TABLE T2 (ID int)
GO
INSERT INTO T1 VALUES (1)
INSERT INTO T1 VALUES (2)
INSERT INTO T2 VALUES (3)
INSERT INTO T2 VALUES (4)
GO

SELECT COUNT(*) FROM T1 CROSS APPLY T2
-----
4

```

APPLY

APPLY

.

CROSS JOIN

.

.

SQL

```

CREATE TABLE Belt
(
    model VARCHAR(20),
    length FLOAT
)
GO
-- fill table with some data
DECLARE @index INT
SET @index = 5
WHILE(@index > 0)
BEGIN
    INSERT INTO BELT
        VALUES ('B' + CONVERT(VARCHAR, @index), 10 * @index)
    SET @index = @index - 1
END
GO

```

```
-- make a table-valued function
CREATE FUNCTION Stretch (@length FLOAT)
RETURN @T TABLE
(
    MinLength FLOAT,
    MaxLength FLOAT
)
AS
BEGIN
    IF (@length > 20)
        INSERT @T VALUES (@length - 4, @length + 5)
    RETURN
END
GO
SELECT B.*, S.MinLength, S.MaxLength FROM Belt AS B
CROSS APPLY Stretch(B.Length) AS S
```

GO

```
-- - - - - -
B30    26    35
B40    36    45
B50    46    55
```

```

                                .Stretch      Belt
                                20             @length
                                .
                                .                CROSS APPLY
                                .                CROSS JOIN
                                .
                                .                OUTER JOIN      OUTER APPLY
                                Stretch      Belt                OUTER APPLY      SQL
                                .
```

```
SELECT B.*, S.MinLength, S.MaxLength FROM Belt AS B
CROSS APPLY Stretch(B.Length) AS S
GO
```

```
-- - - - - -
B10    6     15
B20    16    25
B30    26    35
B40    36    45
B50    46    55
```

CROSS APPLY .OUTER JOIN CROSS JOIN
XML XML

الأمر PIVOT

T-SQL PIVOT SQL Server 2005
OLAP PIVOT .
.PIVOT .SQL:1999

PIVOT

()

5-7

الجدول 5-7: المبيعات الفردية، وهي تتضمن رقم الفصل

المبلغ	الربع (الفصل)	العام
100	Q1	2001
120	Q2	2001
70	Q2	2001
55	Q3	2001
110	Q3	2001
90	Q4	2001
200	Q1	2002
150	Q2	2002

المبلغ	الربع (الفصل)	العام
40	Q2	2002
60	Q2	2002
120	Q3	2002
110	Q3	2002
180	Q4	2002

- PIVOT
-
- 6-7.

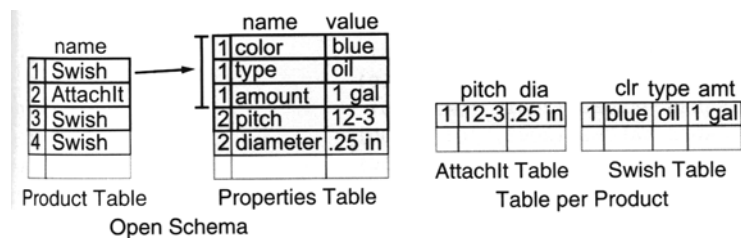
الجدول 6-7: المبيعات السنوية موزعة إلى أرباع.

Q4	Q3	Q2	Q1	العام
90	165	190	100	2001
180	230	250	200	2002

PIVOT
: (entity)
:
"AttachIt" : "Swish"
:
"Swish, 1 gt, green, latex"
:
"Swish, 1 gt, blue, oil"
Swish

.
 "Drying Time"
 Swish
 (Product)
 .(Properties)
 .
 .
 7-7

PIVOT
 PIVOT
 PIVOT
 :



الشكل 7-7: جداول مخزن قطع الحواسيب

UNPIVOT

PIVOT

استخدام PIVOT للتحليل

PIVOT

SQL

SALES

```

CREATE TABLE SALES
(
  [Year] INT,
  Quarter CHAR(2),

```

```

Amount FLOAT
)
GO
INSERT INTO SALES VALUES (2001, 'Q2', 70)
INSERT INTO SALES VALUES (2001, 'Q3', 55)
INSERT INTO SALES VALUES (2001, 'Q3', 110)
INSERT INTO SALES VALUES (2001, 'Q4', 90)
INSERT INTO SALES VALUES (2002, 'Q1', 200)
INSERT INTO SALES VALUES (2002, 'Q2', 150)
INSERT INTO SALES VALUES (2002, 'Q2', 40)
INSERT INTO SALES VALUES (2002, 'Q2', 60)
INSERT INTO SALES VALUES (2002, 'Q3', 120)
INSERT INTO SALES VALUES (2002, 'Q3', 110)
INSERT INTO SALES VALUES (2002, 'Q4', 180)
GO

```

Quarter

: SQL .

```

SELECT * FROM SALES
PIVOT
(SUM (Amount) – Aggregate the Amount column using SUM
FOR [Quarter] – Pivot the Quarter column into column headings
IN (Q1, Q2, Q3, Q4)) – use these quarters
AS P
GO

```

Year	Q1	Q2	Q3	Q4
2001	100	190	165	90
2002	200	250	230	180

```

PIVOT
PIVOT
.SALES
SELECT
PIVOT
.FOR
.PIVOT

```

SQL — Q2

```
SELECT * FROM SALES
PIVOT
(SUM (Amount)
FOR [Quarter]
IN (Q2))
AS P
GO
Year      Q2
-----
2001      190
2002      250
```

```
SELECT          PIVOT
GROUP BY        PIVOT          .GROUP BY [year]
—
SALES
.Other          (SALES2)
```

```
CREATE TABLE SALES2
(
    [Year] INT,
    Quarter CHAR(2),
    Amount FLOAT,
    Other INT
)
INSERT INTO SALES2 VALUES (2001, 'Q2', 70, 1)
INSERT INTO SALES2 VALUES (2001, 'Q3', 55, 1)
INSERT INTO SALES2 VALUES (2001, 'Q3', 110, 2)
INSERT INTO SALES2 VALUES (2001, 'Q4', 90, 1)
INSERT INTO SALES2 VALUES (2002, 'Q1', 200, 1)
INSERT INTO SALES2 VALUES (2002, 'Q2', 150, 1)
INSERT INTO SALES2 VALUES (2002, 'Q2', 40, 1)
INSERT INTO SALES2 VALUES (2002, 'Q2', 60, 1)
INSERT INTO SALES2 VALUES (2002, 'Q3', 120, 1)
INSERT INTO SALES2 VALUES (2002, 'Q3', 110, 1)
INSERT INTO SALES2 VALUES (2002, 'Q4', 180, 1)
```

```
SELECT * FROM Sales2
PIVOT
(SUM (Amount)
FOR Quarter
```

IN (Q3))

AS P

GO

Year	Other	Q3
2001	1	55
2002	1	115
2001	2	110

.Other

2001

PIVOT

SELECT

.PIVOT

. SQL -

SELECT * FROM

(Select Amount, Quarter, Year from Sales2) AS A

PIVOT

(SUM (Amount)

FOR Quarter

IN (Q3))

AS P

GO

Year	Q3
2001	165
2002	230

. PIVOT FOR
:
- NULL

SELECT * FROM SALES

PIVOT

(SUM (Amount)

FOR [Quarter]

IN (Q2, LastQ))

As P

GO

Year	Q2	LastQ
2001	190	NULL
2002	250	NULL

"LastQ"

SALES

Quarter

.NULL

LastQ

استخدام عبارة PIVOT للمخططات المفتوحة

Product

PIVOT

PIVOT

.7-7

Properties

Properties

.8-7

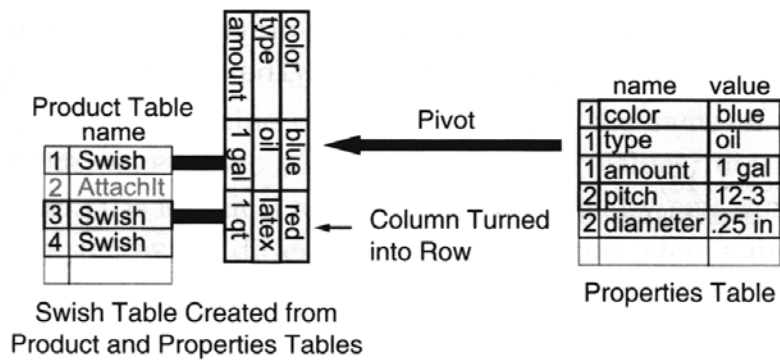
Product

Swish

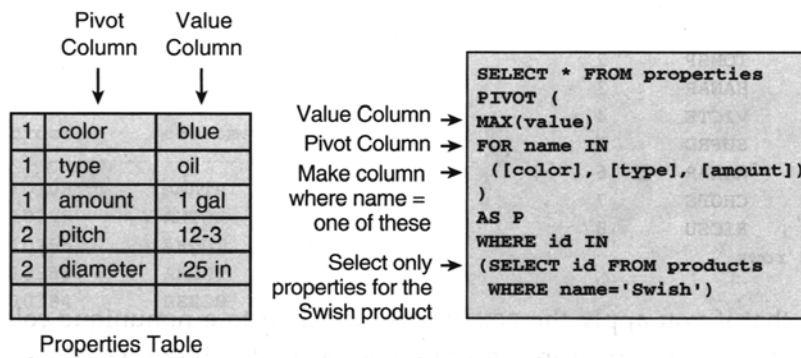
PIVOT

9-7

.Properties



الشكل 8-7: تدوير الخصائص



الشكل 9-7: PIVOT الأساسي

"type"	"Color"	Properties			
.PIVOT		MAX	value	value	"amount"
IN		"amount"	"type"	"Color"	
—			IN		.FOR
				.	
		.10-7	9-7		PIVOT

id not mentioned
in pivot expression

↓

id	color	type	amount
1	blue	oil	1 gal
3	red	latex	1 qt
4	white	oil	1 pt

Pivoted properties of Swish product

الشكل 7-10: نتائج المخطط المفتوح

Properties

ترتيب والتأثير

SQL Server 2005

SQL Server 2005

ROW_NUMBER()

ORDER BY OVER

توابع الترتيب والتأطير

.(ranking) : SQL Server 2005

. SQL Server 2005 .

. ROW_NUMBER()

. ORDER BY OVER

```
WHERE orderid < 10400
AND customerid <= 'BN'
```

Produces

orderid	customerid	num
10248	VINET	1
10249	TOMSP	2
10250	HANAR	3
10251	VICTE	4
10252	SUPRD	5
10253	HANAR	6
10254	CHOPS	7
10255	RICSU	8
... more rows		

```

      —                                ROW_NUMBER( )
      .                                —                                customerid
      :                                ROW_NUMBER( )
      .

```

```
SELECT orderid, customerid,
       ROW_NUMBER() OVER(ORDER BY customerid) AS num
FROM orders
WHERE orderid < 10400
AND customerid <= 'BN'
```

Produces

orderid	customerid	num
10308	ANATR	1
10365	ANTON	2
10355	AROUT	3
10383	AROUT	4
10384	BERGS	5
10278	BERGS	6
10280	BERGS	7
10265	BLONP	8
10297	BLONP	9
10360	BLONP	10

RANK()
(OVER ORDER BY)
ROW_NUMBER() RANK()
.RANK()

SELECT orderid, customerid,
RANK() OVER(ORDER BY customerid) AS [rank]
FROM orders
WHERE orderid < 10400
AND customerid <= 'BN'
Produces

orderid	customerid	rank
10308	ANATR	1
10365	ANTON	2
10355	AROUT	3
10383	AROUT	3
10384	BERGS	5
10278	BERGS	5
10280	BERGS	5
10265	BLONP	8
10297	BLONP	8
10360	BLONP	8
... more rows		

rank
RANK() DENSE_RANK()
"n" NTILE(n)
NTILE

SELECT orderid, customerid,
ROW_NUMBER() OVER(ORDER BY customerid) AS num,
RANK() OVER(ORDER BY customerid) AS [rank],
DENSE_RANK() OVER(ORDER BY customerid) AS [denserank],
NTILE(5) OVER(ORDER BY customerid) AS ntile5

```
FROM orders
WHERE orderid < 10400
AND customerid <= 'BN'
```

Produces

orderid	customerid	num	rank	denserank	ntile5
10308	ANATR	1	1	1	1
10365	ANTON	2	2	2	1
10355	AROUT	3	3	3	2
10383	AROUT	4	3	3	2
10278	BERGS	5	5	4	3
10280	BERGS	6	5	4	3
10384	BERGS	7	5	4	4
10265	BLONP	8	8	5	4
10297	BLONP	9	8	5	5
10360	BLONP	10	8	5	5

.(windowing)

PARTITION BY

.OVER

```
SELECT *,
       RANK() OVER(PARTITION BY COUNTRY ORDER BY age) AS [rank]
FROM
(
  SELECT lastname, country,
         DATEDIFF(yy,birthdate,getdate())AS age
  FROM employees
) AS a
```

produces

lastname	country	age	rank
Dodsworth	UK	37	1
Suyama	UK	40	2
King	UK	43	3
Buchanan	UK	48	4
Leverling	USA	40	1
Callahan	USA	45	2
Fuller	USA	51	3
Davolio	USA	55	4
Peacock	USA	66	5

```

select
    ORDER BY PARTITION BY
    FROM
    :

SELECT lastname, country,
    DATEDIFF(yy,birthdate,getdate())AS age,
    RANK() OVER(PARTITION BY COUNTRY ORDER BY age) AS [rank]
FROM employees

'age' : "Invalid column name 'age'" :
WHERE
    .

-- 10 rows to a page, we want page 40

-- this won't work
SELECT
    ROW_NUMBER() OVER (ORDER BY customerid, requireddate) AS num,
    customerid, requireddate, orderid
FROM orders
WHERE num BETWEEN 400 AND 410

-- this will
SELECT * FROM
(
    SELECT
        ROW_NUMBER() OVER (ORDER BY customerid, requireddate) AS num,
        customerid, requireddate, ordered
    FROM orders
) AS a
WHERE num BETWEEN 400 AND 410

SELECT num

.ORDER BY SELECT
    .

```

```

SELECT *,
    RANK() OVER(PARTITION BY COUNTRY ORDER BY age)) AS [rank]
FROM
(
    SELECT lastname, country,
        DATEDIFF(yy,birthdate,getdate())AS age
    FROM employees
) AS a
ORDER BY RANK()

```

```

    OVER(PARTITION BY COUNTRY ORDER BY age), COUNTRY
lastname      country      age      rank
- - - - -
Dodsworth     UK          37       1
Leverling     USA         40       1
Suyama        UK          40       2
Callahan      USA         45       2
King          UK          43       3
Fuller        USA         51       3
Buchanan      UK          48       4
Davolio       USA         55       4
Peacock       USA         66       5

```

```

)
OVER (
-
.
.

```

```

-- there is one oldest employee age for each country
SELECT *,
    RANK() OVER(PARTITION BY COUNTRY ORDER BY age) AS [rank],
    MAX(age) OVER(PARTITION BY COUNTRY) AS [oldest age in country]
FROM
(
    SELECT lastname, country,
        DATEDIFF(yy,birthdate,getdate())AS age
    FROM employees
) AS a
GO

```

```

lastname      country      age      rank      oldest age in country
- - - - -
Dodsworth     UK          37       1          48
Suyama        UK          40       2          48
King          UK          43       3          48

```

Buchanan	UK	48	4	48
Leverling	USA	40	1	66
Callahan	USA	45	2	66
Fuller	USA	51	3	66
Davolio	USA	55	4	66
Peacock	USA	66	5	66

معالجة إبطال المناقلة

SQL Server

SQL Server 2005 .GOTO

. TRY/CATCH

```
BEGIN TRY
    { sql_statement | statement_block }
END TRY
BEGIN CATCH TRAN_ABORT
    { sql_statement | statement_block }
END CATCH
```

.TRY

CATCH

CATCH

ON XACT_ABORT

21

SQL Server

21

TRY

CATCH

TRY/CATCH

. TRY/CATCH

CATCH

.TRY/CATCH

:

--make sure we catch all errors

SET XACT_ABORT ON

BEGIN TRY

--start the tran

BEGIN TRAN

--do something here

```

        COMMIT TRAN
    END TRY
    BEGIN CATCH TRAN_ABORT
        ROLLBACK
        --cleanup code
    END CATCH

```

```

.(ROLLBACK)                                CATCH

```

```

        .
        : "doomed" "failed" :
        .
        :3930

```

Transaction is doomed and cannot make forward progress Rollback transaction.

```

        .
        :
        -
        . ROLLBACK
        .
        : — RAISERROR
        RAISERROR SQL Server 2005
        TRAN_ABORT
        .CATCH

```

أين نحن الآن ؟

```

.NET SQL Server 2005 CLR
T-SQL .T-SQL SQL Server
( )
SQL Server T-SQL
T-SQL

```