Vysoká škola báňská – Technická univerzita Ostrava
Fakulta strojní, Katedra automatizační techniky a řízení

9

Informační systémy

2008/2009

Radim Farana

Obsah

- MS SQL Server 2005,
  - Systémové procedury a funkce
  - Spouště (Trigger), aktualizace agregovaných dat, DELETE, INSERT, UPDATE .
- Doporučená literatura:
  Andrew J. Brust, Stephen Forte. *Mistrovství v programování SQL Serveru 2005*. Brno: Computer Press, a.s., 2007. ISBN 978-80-251-1607-4.

Bezpečnost uložených procedur

- Uložená procedura umožňuje manipulaci se záznamy.
- Uložená procedura je schopna zajistit kontrolu pravidel referenční integrity apod.
- Uložená procedura může vracet záznamy i z tabulky, ke které nemá uživatel přístup.

Výkon uložených procedur

- Optimalizace a kompilace při prvním použití.
- Při použití dynamických příkazů nemusí být zpracování optimální.
- Novou rekompilaci je možno vynutit klauzulí WITH RECOMPILE buď při její deklaraci nebo při jejím volání.
  - EXEC ProcName [Param] WITH RECOMPILE

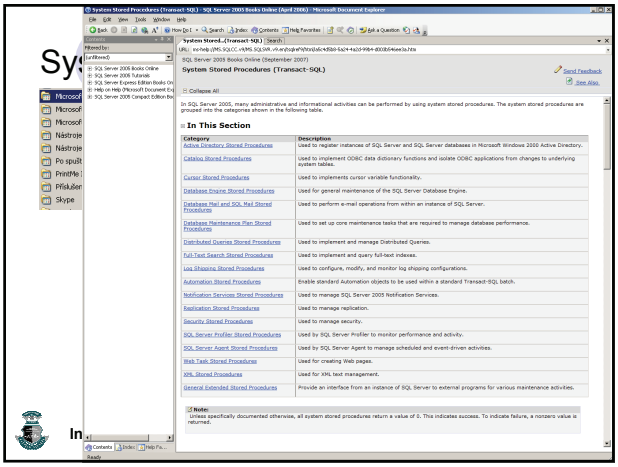
Systémové uložené procedury

- Jsou uloženy v databázi Master.
- Je možné je vyvolat názvem bez ohledu na databázi (pro využití v jiné databázi).
- Např.:
  - USE Prace
  - EXEC sp_columns 'prace'

Results | Messages

	TABLE_QUALIFIER	TABLE_OWNER	TABLE_NAME	COLUMN_NAME	DATA_TYPE	TYPE_NAME	PRECISION	LENGTH	SCALE	RADIX	NULLABLE	REMARKS
1	Prace	dbo	prace	p_id	4	int identity	10	4	0	10	0	NULL
2	Prace	dbo	prace	p_pracovnik	4	int	10	4	0	10	0	NULL
3	Prace	dbo	prace	p_typ	12	varchar	50	50	NULL	NULL	1	NULL
4	Prace	dbo	prace	p_kusu	6	float	15	8	NULL	10	1	NULL
5	Prace	dbo	prace	p_datum	11	datetime	23	16	3	NULL	1	NULL

Sy

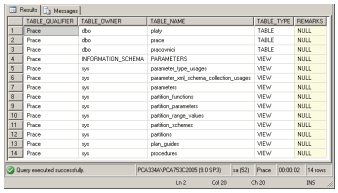


In

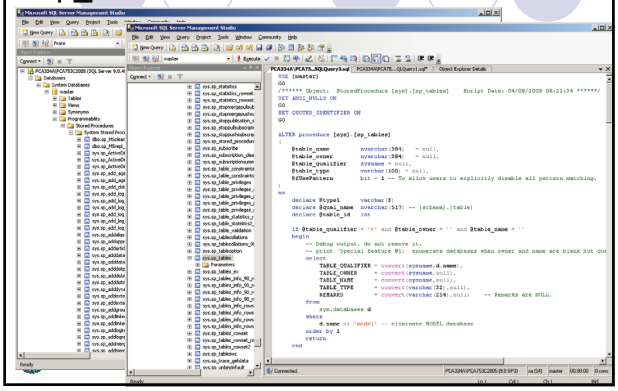
Catalog Stored Procedures

- sp_column_privileges
- sp_special_columns
- sp_columns
- sp_sproc_columns
- sp_databases
- sp_statistics
- sp_fkeys
- sp_stored_procedures
- sp_pkeys
- sp_table_privileges
- sp_server_info
- sp_tables

Např.:
USE Prace
EXEC sp_tables 'p%'



sp_tables



Rozšířené uložené procedury

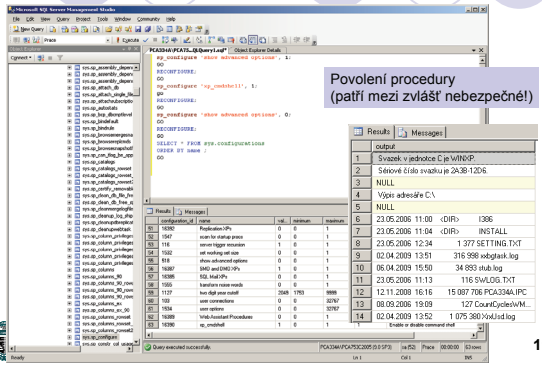
- **(Extended Stored Procedures)**
- **Ukládají se do databáze master**
- **Musí být volány master..xp_name**
- **Vytvářejí se vně SQL serveru pomocí jiného jazyka (C, C++).**
- **Běží jako interní (in-process) a mohou být velmi rychlé.**

Procedura xp_cmdshell

- Spuštění příkazu operačního systému
- xp_cmdshell 'řetězec příkazu' [, no_output]
- Příklad
  - EXEC master..xp_cmdshell 'dir "C:/*.*"'
  - Výstupem je soubor záznamů pro jednotlivé řádky výstupu

```
Msg 15281, Level 16, State 1, Procedure xp_cmdshell, Line 1
SQL Server blocked access to procedure 'sys.xp_cmdshell' of component
'xp_cmdshell' because this component is turned off as part of the security
configuration for this server. A system administrator can enable the use
of 'xp_cmdshell' by using sp_configure. For more information about
 enabling 'xp_cmdshell', see "Surface Area Configuration" in
SQL Server Books Online.
```

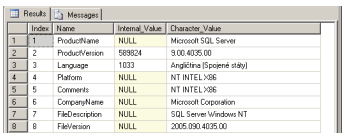
Procedura xp_cmdshell



Povolení procedury
(patří mezi zvlášť nebezpečné!)

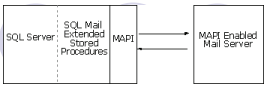
Procedura xp_msver

- Vrací informace o verzi systému.
- Podrobnější než funkce @@VERSION.
- Strukturovaný výstup.
  - EXEC master..xp_msver



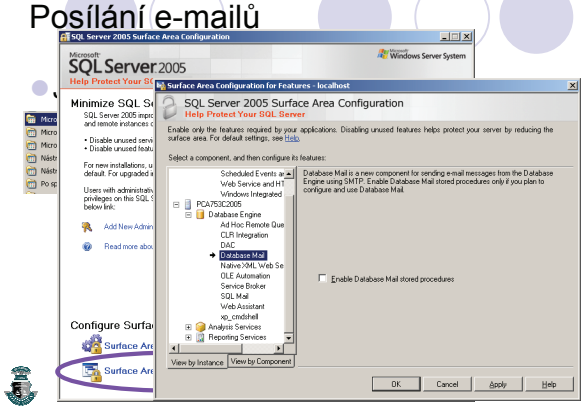
	Index	Name	Internal_Value	Character_Value
1	1	ProductName	NULL	Microsoft SQL Server
2	2	ProductVersion	589824	9.00.4035.00
3	3	Language	1033	Angličtina (Spojené státy)
4	4	Platform	NULL	NT INTEL X86
5	5	Comments	NULL	NT INTEL X86
6	6	CompanyName	NULL	Microsoft Corporation
7	7	FileDescription	NULL	SQL Server Windows NT
8	8	FileVersion	NULL	2005.090.4035.00

Posílání e-mailů



- **Odeslání e-mailu**
  - **xp_startmail [[@user =] *name*']**
    [,[@password =] '*password*']
  - **xp_sendmail …**
  - **xp_stopmail**
- **Přijetí e-mailu**
  - **sp_processmail**
- **Nebo**
  - **xp_findnextmessage**, **xp_readmail**, **xp_deletemail**

Posílání e-mailů



Rekurze

- Procedura volá sama sebe.
- Při vytvoření procedury je hlášena chyba (odkazuje se ne neexistující proceduru), ale procedura bude vytvořena.
- Počet vnoření je omezen na 32!
  Je možno zjistit pomocí @@NESTLEVEL.

Příklad - Faktoriál

```
CREATE  PROC Factorial
  @InputValue Int,
  @OutputValue Int OUTPUT
AS
  DECLARE @InVal Int, @OuVal Int
  IF @InputValue != 1
    BEGIN
      SELECT @InVal = @InputValue - 1
      EXEC Factorial @InVal, @OuVal OUTPUT
      SELECT @OutputValue = @OuVal * @InputValue
    END
  ELSE
    SELECT @OutputValue = 1
RETURN
GO
```

```
USE PRACE
DECLARE @in int, @out int
SET @in = 5
EXEC Factorial @in, @out OUTPUT
PRINT CAST(@in AS VARCHAR)+
  ' - ' + CAST(@out AS VARCHAR)
GO
```

5 – 120
funguje až do x = 12

pro x = 13:
Server: Msg 8115, Level 16, State 2, Procedure Factorial, Line 10
Arithmetic overflow error converting expression to data type int.

Cannot add rows to sysdepends for the current stored procedure because it depends on the missing object 'Factorial'. The stored procedure will still be created.

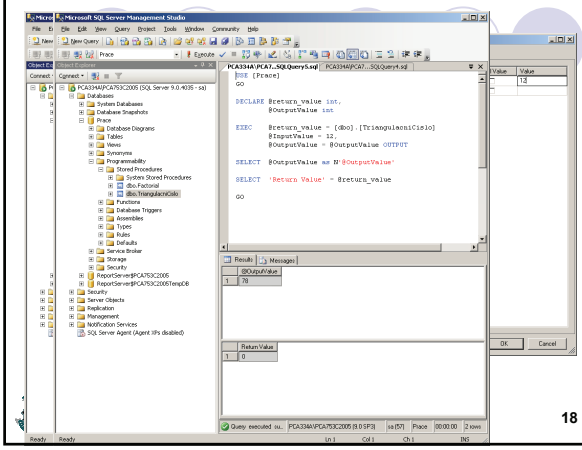
Příklad – Triangulanční číslo

```
CREATE PROC TriangulacniCislo
  @InputValue Int,
  @OutputValue Int OUTPUT
AS
  DECLARE @InVal Int, @OuVal Int
  IF @InputValue != 1
    BEGIN
      SELECT @InVal = @InputValue - 1
      EXEC TriangulacniCislo @InVal, @OuVal OUTPUT
      SELECT @OutputValue = @OuVal + @InputValue
    END
  ELSE
    SELECT @OutputValue = 1
RETURN
GO
```

```
USE PRACE
DECLARE @in int, @out int
SET @in = 13
EXEC TriangulacniCislo @in, @out OUTPUT
PRINT CAST(@in AS VARCHAR)+ ' - ' +
  CAST(@out AS VARCHAR)
GO
```

13 – 91
Funguje až do x = 32

Pro x = 33
Server: Msg 217, Level 16, State 1, Procedure TriangulacniCislo, Line 9
Maximum stored procedure, function, trigger, or view nesting level exceeded (limit 32).



Uživatelsky definované funkce

- Funkce mohou vracet
  - Skalární hodnotu,
  - Tabulku.
- Deklarace
- DECLARE FUNCTION fname
  (@ParName AS DataType)
  RETURNS DataType| TABLE
  AS
  BEGIN
  …
  END

Funkce vracející skalární hodnotu

```
CREATE FUNCTION DateOnly(@InputDate DATETIME)
 RETURNS DATETIME
AS
BEGIN
 RETURN CONVERT(datetime, CONVERT(varchar, @InputDate, 112))
END
GO
```

Funkce musí být volána se jménem vlastníka.
Neznámo proč.

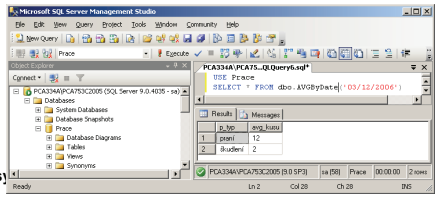
```
Použití
USE PRACE
PRINT dbo.DateOnly(getdate())
GO
Apr 17 2006 12:00AM
```

Datum bez času má čas 00:00:00,0

Funkce vracející tabulku

```
CREATE FUNCTION AVGByDate(@InputDate DATETIME)
  RETURNS TABLE
AS
  RETURN (SELECT p_typ, AVG(p_kusu) AS avg_kusu
  FROM prace WHERE p_datum=@InputDate GROUP BY p_typ)
GO
```

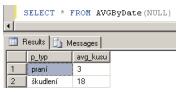
Funkce se používá stejně jako každý jiný datový zdroj,
(až na nutnost uvedení vlastníka funkce)



Úprava pro prázdný parametr

Kontrola existence objektu

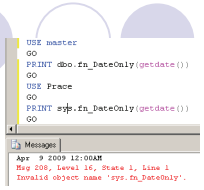
```
USE Prace
GO
IF EXISTS (SELECT * FROM dbo.sysobjects WHERE name='AVGByDate')
  DROP FUNCTION AVGByDate
Go
CREATE FUNCTION AVGByDate(@InputDate DATETIME)
  RETURNS @avg_table TABLE(p_typ varchar(50), avg_kusu float)
AS
BEGIN
  DECLARE @tmp_table TABLE(p_typ varchar(50), avg_kusu float)
  IF (@InputDate IS NULL)
    SELECT @InputDate = Max(p_datum) FROM prace
  INSERT @tmp_table SELECT p_typ, AVG(p_kusu) AS avg_kusu FROM prace
      WHERE p_datum=@InputDate GROUP BY p_typ
  INSERT @avg_table SELECT * FROM @tmp_table
  RETURN
END
GO
```



Systémové funkce

- Vytvářejí se v databázi Master.
- Začínají předponou fn_.
- Pro SQL Server 2000:
  Vlastníka je nutno změnit na system_function_schema pomocí uložené procedury sp_changeobjectowner

Systémové funkce




```
USE Prace
DROP FUNCTION DateOnly
GO
USE master
GO
CREATE FUNCTION fn_DateOnly(@InputDate DATETIME)
  RETURNS DATETIME
AS
BEGIN
  RETURN CONVERT(datetime, CONVERT(varchar, @InputDate, 112))
END
GO
```

Správné volání včetně databáze:
PRINT master.dbo.fn_DateOnly(getdate())

Odstranění systémové funkce

- Je nutno povolit systémové změny.
- Velmi nebezpečné!

```
USE master
GO
EXEC sp_configure 'allow updates', 1
GO
RECONFIGURE WITH OVERRIDE
GO
DROP FUNCTION dbo.fn_DateOnly
GO
EXEC sp_configure 'allow updates', 0
GO
RECONFIGURE WITH OVERRIDE
GO
```

Nekontroluje hodnotu parametru (možno zadat nedoporučovanou hodnotu)

Spouště (Triggers)

- Uložené procedury spouštěné automaticky systémem při manipulaci s tabulkami:
  - INSERT
  - DELETE
  - UPDATE
- Využití spouští
  - Zajištění referenční integrity nad REFERENCE
  - Kontrola omezení nad CHECK
  - Vytváření záznamů o sledování systému

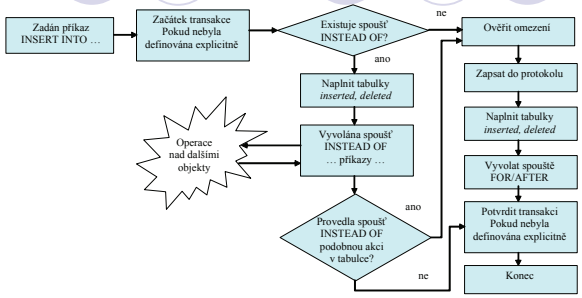
Vytvoření spouště

- CREATE TRIGGER TriggerName
  ON TableName
  {{{ FOR | AFTER} [INSERT] [,] [DELETE] [,] [UPDATE] } | INSTEAD OF}
  [WITH APPEND]
  [NOT FOR REPLICATION]
  AS
  …

Určení dotčených záznamů

- K určení záznamů dotčených změnou slouží dvě pracovní tabulky stejného schématu jako je samotná tabulka
  - inserted - seznam vkládaných záznamů (jejich nových hodnot),
  - deleted – seznam odstraňovaných záznamů (jejich původních hodnot).
  - Při aktualizaci jsou naplněny obě tabulky příslušnými záznamy.

Postup činnosti při vkládání



Spouště INSTEAD OF

- Pracují s daty před provedením všech kontrol (CHECK a REFERENCE).
  - Zatímco spouště FOR a AFTER probíhají až po kontrole omezení.
- Jsou povoleny také u pohledů.
- Mohou nahradit provedení příslušné operace jinou činností.

Příklad

Po vložení nové práce je třeba aktualizovat odměnu

```
CREATE  TRIGGER pracemeniplaty ON [dbo].[prace]
FOR INSERT, UPDATE, DELETE
AS
-- zpracujeme vložení nové práce
IF EXISTS (SELECT * FROM INSERTED)
  BEGIN
    DECLARE @pid INT, @pprac INT, @ptyp VARCHAR(50), @pkusu FLOAT,
     @pdatum DATETIME, @pcena money
    -- zjistíme kolik záznamů je přidáno
    DECLARE cpom CURSOR
    FOR
      SELECT p_id, p_pracovnik, p_typ, p_kusu, p_datum FROM inserted
    OPEN cpom
    -- vezmeme první nový záznam
    FETCH NEXT FROM cpom
      INTO @pid, @pprac, @ptyp, @pkusu, @pdatum
    WHILE @@FETCH_STATUS = 0
      BEGIN
        -- Určíme cenu vložené práce
        SELECT @pcena = Max(tp_cena) FROM typyprace WHERE tp_nazev=@ptyp
        -- Pokud již tento pracovník v daném dni pracoval, budeme záznam upravovat, jinak přidat nový
        IF EXISTS (SELECT * FROM platy WHERE pl_pracovnik=@pprac AND pl_datum=@pdatum)
          UPDATE platy SET pl_castka=pl_castka+@pkusu*CAST(@pcena AS FLOAT)
            WHERE pl_pracovnik=@pprac AND pl_datum=@pdatum
        ELSE
          INSERT INTO platy(pl_pracovnik, pl_datum, pl_castka) VALUES (@pprac, @pdatum,
            @pkusu*CAST(@pcena AS FLOAT))
        FETCH NEXT FROM cpom
          INTO @pid, @pprac, @ptyp, @pkusu, @pdatum
      END
    CLOSE cpom
    DEALLOCATE cpom
  END
GO
```

Možnosti spouští

- Spouště mohou být vnořené
  - Pokud je to povoleno (implicitně ano)
  - Spoušť svojí činností vyvolá jinou spoušť
  - Maximálně 32 úrovní vnoření
  - Stejná spoušť nebude vyvolána podruhé
  - Celý řetězec spouští se provádí v jedné transakci (ROLLBACK stornuje celý řetězec)
- Spouště mohou být rekurzivní

Určení pořadí provádění spouští

- Jedna spoušť může být určena jako první,
- jedna spoušť jako poslední,
- pomocí systémové uložené procedury:
- sp_settriggerorder
  [@triggername =] 'TriggerName',
  [@order =] '{ FIRST | LAST | NONE }',
  [@stmttype =] ' { INSERT | UPDATE | DELETE
  } '
- Využití: např. je možno definovat více spouští pro stejnou činnost.

Určení dotčených sloupců

- Funkce UPDATE(ColumnName) vrací logickou informaci o změně daného sloupce.
  - Má smysl jen uvnitř spouště.
- Funkce COLUMNS _UPDATED() vrací součet binárních vah odpovídajících jednotlivým změněným sloupcům
  - počínaje prvním 2^0, 2^1, 2^2, 2^3 , 2^4 , …
  - Je možno je testovat pomocí binárních operací
    - | = bitové OR, & = bitové AND, ^ = bitové XOR,
    - COLUMNS _UPDATED() > 0 – aktualizováno cokoliv,
    - COLUMNS _UPDATED() ^ 19 = 0 – aktualizovány právě sloupce 1, 2, 5 a žádný jiný,
    - COLUMNS _UPDATED() & 19 = 19 – aktualizovány sloupce 1, 2, 5, ostatní mohly být aktualizovány také,
    - COLUMNS _UPDATED() | 19 != 19 – aktualizovány sloupce 1, 2, 5 a ještě nějaký jiný.

Odstranění spouště

- Odstranění spouště
- DROP TRIGGER TriggerName
- Spoušť není nutno odstraňovat, dá se vypnout!
- ALTER TABLE TableName
  < ENABLE | DISABLE > TRIGGER
  < ALL | TriggerName >

Ladění spouští

- Přímo není možné.
- Lze obejít vytvořením uložené procedury, která vynutí spuštění spouště provedením příslušné operace.
- Uloženou proceduru krokujeme příkazem "Step Into" – dojde ke vstupu do spouště a její krokování bude probíhat standardně.