Динамикалық SQL

ТОБЫ:ИС:17-11

Динамикалық SQL дегеніміз не?

- Динамикалық SQL - стандартты (немесе статикалық) SQL-тан айырмашылығы, бағдарлама мәлімдемелерінің автоматты түрде жасалуын және орындалуын жеңілдететін құрылымдық сұрау тілі (SQL) жетілдірілген нысаны. Бұл әр түрлі дерекқорларға, шарттарға немесе серверлерге реттеуге болатын кодты жазу қажет болған кезде пайдалы болуы мүмкін. Ол бірнеше рет қайталанатын тапсырмаларды автоматтандыруды жеңілдетеді.

Динамикалық SQL артықшылықтары

- Динамикалық SQL бағдарлама іске қосылған кезде енгізілетін таңбалар жолдары түрінде сақталады. Олар программист немесе Бағдарламаның өзі автоматты түрде енгізілуі мүмкін, бірақ SQL статикалық операторларына қарағанда, олар бастапқы бағдарламаға енгізілмеген. Сонымен қатар, SQL статикалық операторларына қарағанда, SQL динамикалық операторлары бір орындалудан екіншісіне өзгеруі мүмкін.

Динамикалық SQL әлсіз жақтары

- Динамикалық SQL өте қиын, оқуға қиын, қолдау және реттеу қиын болуы мүмкін.
- Рұқсат стандартты SQL-ден ерекшеленеді.
- Күтпеген енгізуден күтпеген нәтижелер.
- Динамикалық SQL (тырнақшаларда) әрқашан сәтті компиляцияланады, бірақ орындау кезінде қате тудыруы мүмкін.
- Динамикалық SQL функциясына пайдалануға болмайды.

Операторлар

- Динамикалық SQL операторлары салыстырмалы түрде аз бағдарламалау тәжірибесі бар адамдармен жазылуы мүмкін, себебі бағдарлама кодтың нақты генерациясының басым бөлігін жасайды. Ықтимал мәселе кез-келген уақытта жұмыс істейтін динамикалық SQL-тым көп болса, өнімділікті төмендетеді (өңдеу уақытын ұлғайту).

Динамикалық SQL

- Біздің жобаларымызда бізге түрлі міндеттерді шешуге тура келеді. Олардың кейбірін шешу үшін біз dynamic T-Sql пайдаланамыз.Dynamic sql не үшін қажет? Әркім өзі үшін шешеді. Dynamic sql көмегімен жобалардың бірінде біз динамикалық есептерді құру, басқаларында деректерді тасымалдау тапсырмаларын шештік. Сондай-ақ, dynamic sql деректер немесе нысандар жасау/өзгерту/алу қажет болған жағдайда, бірақ мәндер/атаулар параметрлер ретінде келеді.

Одан әрі біз dynamic sql арқылы жүзеге асыруға болатын бірнеше мысалдарды көрсетеміз.

- Динамикалық команданы бірнеше жолмен орындауға болады:
- EXEC/EXECUTE кілт сөзін пайдалану;
- Сақталатын sp_executesql процедурасын пайдалану
- Бұл тәсілдер өзара түбегейлі ерекшеленеді. Шағын мысалда біз олардың айырмашылығы туралы түсіндіруге тырысамыз.

EXEC/EXECUTE кілтік сөзі

```
DECLARE @sql varchar(1000)
DECLARE @columnList varchar(75)
DECLARE @city varchar(75)
SET @columnList = 'CustomerID, ContactName, City'
SET @city = 'London'
SELECT @sql = ' SELECT CustomerID, ContactName, City ' +
              ' FROM dbo.customers WHERE 1 = 1 '
   SELECT @sql = @sql + ' AND City LIKE ''' + @city + ''''
EXEC (@sql)
```

- Сұраудан көрініп тұрғандай, біз динамикалық команданы қалыптастырамыз. Егер select @sql орындалса, нәтиже келесі болады:

```
SELECT CustomerID, ContactName, City FROM customers WHERE City = 'London'
```

sp_executesql процедурасы

```
DECLARE @sqlCommand varchar (1000)
DECLARE @columnList varchar (75)
DECLARE @city varchar (75)
SET @city = 'London'
SET @sqlCommand = 'SELECT CustomerID, ContactName, City FROM customers WHERE City =
 @city'
EXECUTE sp_executesql @sqlCommand, N'@city nvarchar(75)', @city = @city
```

EXECUTE-дан айырмашылығы, sp_executesql пайдаланған кезде sp_executesql терілген параметрлерді пайдалансаңыз, ешқандай типті қою қажет емес.

Сондай-ақ, sp_executesql пайдаланудың артықшылықтарының бірі - OUT параметрі арқылы мәнді қайтару мүмкіндігі.

Курсорлар арқылы динамикалық SQL операторларымен жұмыс істеу

- Мұндай операторларды пайдалану үшін SQL стандартының курсорлар механизмін кеңейту қолданылады. Біріншіден, курсорды анықтау кезінде курсордың литералды ерекшелігін ғана емес, PREPARE операторының көмегімен енгізілетін оператордың атын да көрсетуге болады(бұл жағдайда PREPARE операторы мәтіндік түрде DECLARE операторынан жоғары болуы тиіс). Осылайша DECLARE операторының толық синтаксисі келесідей болады:
- <declare cursor> ::=
- DECLARE <cursor name> CURSOR
- FOR { <cursor specification> | <statement-name> }

- Сонымен қатар, статистикаға мұндай курсор үшін инклюзивті бағдарламаның кіріс және шығыс айнымалылары туралы ақпарат белгісіз, OPEN және FETCH операторларының басқа формалары пайдаланылады.Осы мәлімдемелердің толық синтаксисі келесідей:

```
<open statement> ::=
   OPEN <cursor name>
  [USING { <host-vars-list> | DESCRIPTOR <descr-name> }]
<fetch statement> ::=
   FETCH <cursor name>
 { INTO <fetch target list>
 ( USING <host-vars-list>
 ( USING DESCRIPTOR <descr-name> }
```

- Көріп отырғаныңыздай, нақты кіріс және шығыс параметрлерін орнатудың екі жолы бар: тікелей OPEN және / немесе FETCH бағдарламаларында айнымалылардың атауларын көрсету арқылы тікелей және жанама түрде параметрлер мен олардың мекен-жайлары қосымша дескриптор құрылымы арқылы байланысқан кезде.
- Бірінші әдісті формальды кіріс және шығыс параметрлерінің жиынтығы бекітілген таңдау операторларымен жұмыс істеу үшін пайдалану ұсынылады. Дәлірек айтқанда, Шығыс параметрлеріне келетін болсақ, таңдау тізімі элементтерінің саны мен түрлері белгіленуі тиіс.
- Екінші тәсіл динамикалық құрастырылған операторлармен жұмыс істеу, курсорларды пайдалануды талап етеді, динамикалық қалыптасатын параметрлер тізімдерінің дескрипторларын пайдаланудан тұрады. Бұл жағдайда нақты және формальды параметрлер үлгілерінің сәйкестігі үшін барлық жауапкершілік программистке жүктеледі. Мұндай тізімді қалыптастырудағы қателік нәтижесінде, атап айтқанда, Си-бағдарламаның жады бүлінуі мүмкін.