

Microsoft®
SQL Server™ 2000

Microsoft® SQL Server

문제발생 시
이렇게하세요. ☆

Troubleshooting 가이드

Microsoft®

Microsoft
SQL Server™ Troubleshooting 가이드

저자의 글

비정상적인 상태를 정상적인 상태로 수정하는 작업을 트러블슈팅과 튜닝이라고 말할 수 있습니다. 이와 같은 작업을 위해서는 문제에 대한 올바른 접근과 문제 해결을 위한 충분한 사전 지식이 있어야 합니다. 지면 관계상 많은 사례를 포함시키지 못하였지만 본 가이드가 SQL Server 주변에서 발생하는 문제에 대해서 올바르게 접근하고 효율적으로 해결할 수 있는 기본적인 지식과 방법을 습득하는데 도움이 되었으면 합니다.

저자 약력

이종인 / 다우데이터시스템 SQL Server 컨설턴트 겸 전임강사

- 이론인포텍 DB팀장
- 산업은행 여신지원시스템 DBA
- 동서증권 대리
- 산업은행 / LG.Philips LCD 등 다수의 SQL Server 튜닝 및 재난대비 컨설팅
- MCSE+Internet, MCDBA, MCT, OCP, HP Master ASE

감수자의 글

그동안 간단하게 명령어 몇 개로 해결할 수 있는 문제인데 그 단순한 해결 방법을 몰라서 고생한 SQL Server 사용자들을 많이 보았습니다.

데이터베이스라는 것이 워낙 복잡한 제품이어서 이 책에서 언급한 몇 가지 가이드로 실제 업무 환경에서 접할 수 있는 모든 문제들을 해결할 수는 없을 것입니다.

그렇지만 이 포켓북에 보편적으로 자주 발생하는 문제들에 대한 해결책들이 제시되어 있으므로 좋은 참고가 될 거라 믿습니다.

감수자 약력

하성희 / 에이디컨설팅 대표컨설턴트 / Microsoft MVP

- 데이터베이스 컨설턴트
(CGV, 싸이월드, 인터파크지마켓, 아이템베이, 신세계닷컴, 에이스화재, 롯데카드, 삼성생명, 엔씨소프트, 부동산써브, SK케미칼 등 다수의 SQL Server 고객사들에 대한 DB 컨설팅 수행)
- 옥션 DBA 팀장
- 마이크로소프트 SQL Server 기술지원 엔지니어 / Data Access 팀장
- SYBASE DB 엔지니어 (다수의 기업체, 병원, 학교 DB 기술지원)
- POSCO MS SQL Server 및 SYBASE SQL Server DBA
- 프로그래머, UNIX 시스템 관리자

목차

Troubleshooting 방법론	1
---------------------------	---

이벤트 로그와 SQL Server 오류 로그 확인하기	3
-------------------------------------	---

1. 이벤트 로그	3
2. SQL Server 오류 로그	10

SQL Server 와 연결 불가의 경우	14
------------------------------	----

CASE 1. 다른 클라이언트들에서는 이상이 없는 상황에서 최초 구성한 클라이언트만 연결이 실패하는 경우	16
--	----

① 통신의 이상이 있는 경우	16
② 서버의 자원이 과도하게 사용되는 경우	19

CASE 2. 특정 클라이언트에서만 연결이 잘 되다가 갑자기 실패하는 경우	23
---	----

CASE 3. 일부 클라이언트들에서 연결이 실패하는 경우	24
---------------------------------------	----

CASE 4. 모든 클라이언트들에서 연결이 실패하는 경우	24
---------------------------------------	----

① 통신의 이상이 있는 경우	24
② 서버의 자원이 과도하게 사용되는 경우	25
③ 메모리 단편화가 발생된 경우	25
④ Max worker thread의 임계값에 도달한 경우	27
⑤ 컴퓨터의 이름을 변경한 경우	34
⑥ 윈도우즈 인증모드인 경우	36

쿼리의 응답속도가 갑자기 느려지는 경우	42
-----------------------------	----

CASE1. 통계정보 점검	43
----------------------	----

CASE2. 인덱스 단편화 점검	46
-------------------------	----

CASE3. 잠금에 의한 대기 점검	49
---------------------------	----

CASE4. 로그 파일 설정의 문제 점검	51
------------------------------	----

CASE5. 데이터 파일 설정의 문제 점검	53
-------------------------------	----

CASE6, tempdb의 설정 문제 점검	53
CASE7, 디스크 사용률 점검	54
CASE8, 병렬 처리 버그 문제 점검	55
CASE9, 바이러스 등의 문제 점검	57

노련한 Troubleshooter가 되려면	60
--------------------------------------	----

손상된 사용자 데이터베이스 복구	63
--------------------------------	----

1. 데이터 파일 손상 시 복구 방법	66
2. 데이터베이스 파일이 저장된 디스크와 운영체제의 연결이 비정상 적인 경우 ...79	
3. 로그 파일 손상 시 복구	81
① 로그 파일 손상 시 진행중인 트랜잭션이 없는 단일 로그 파일인 경우	83
② 다중 로그 파일인 경우	84
③ 손상된 트랜잭션이 존재하는 경우	85
④ 데이터베이스 분리 중 오류가 발생한 경우	91

손상된 시스템 데이터베이스 복구	93
--------------------------------	----

1. master 데이터베이스 복구 및 이동	93
2. model 데이터베이스 복구 및 이동	102
3. msdb 복구 및 이동	109
4. tempdb 복구 및 이동	113

분산 트랜잭션을 시작할 수 없는 경우	118
-----------------------------------	-----

1. SET XACT_ABORT 옵션을 활성화하지 않은 경우	118
2. 윈도우즈 2003 서버에서 DTC 사용 시 오류	119
3. 분산 쿼리 사용시 주의 사항 및 오류	125

대용량 작업 시 주의 사항	132
1. 대용량의 BCP	132
2. 대용량의 삭제 / 수정	133
3. 대용량의 인덱스 생성 작업	134
4. 파일 사이즈를 줄이는 작업	136
5. 대용량의 테이블 관리	136
6. 대용량 작업 중 서버가 장애를 입은 경우	136
기타 자주 발생하는 오류	138
1. administrator 계정의 패스워드를 변경한 뒤 서비스가 시작되지 않는 경우	138
2. 네트워크를 사용한 백업이 실패하는 경우	140
3. 다른 서버로 데이터베이스 복구 후 Broken Login 문제	142
4. 윈도우즈 2003 서버에서 SQL Server 성능 카운터가 로그에 기록되지 않는 경우	153
5. SQL Server 성능 카운터가 시스템 모니터에서 보이지 않는 경우	154
사용된 명령어 정리	157
• 저장 프로시저	157
• DBCC 명령어	158
• 유틸리티	159
• 시작 옵션 및 추적 플래그	160

Troubleshooting 방법론

마이크로소프트는 오류가 발생하거나 성능상의 문제가 발생하게 되면 다음과 같은 DETECT 라는 방법을 통해서 문제에 접근하고 해결하는 것을 권장하고 있습니다.

1단계	D	Discover the problem
2단계	E	Explore the Conditions
3단계	T	Track down Possible approaches
4단계	E	Execute the most likely approach
5단계	C	Check for Success
6단계	T	Tie up Loose ends

- 첫 번째 Discover the Problem 단계는 성능 문제나 장애 사항을 발견하는 첫 단계입니다. 이단계에서는 발생된 문제와 관련된 정보를 정확히 유지하여야 합니다.
- 두 번째 Explore the Conditions 단계는 발생된 문제의 주변 상황을 면밀히 검토하여 문제를 올바르게 정의하는 단계입니다.
- 세 번째 Track down Possible Approaches 단계는 앞의 단계에서 정의된 문제가 발생된 원인을 찾기 위해 범위를 좁혀 나가고 예상되는 원인 별로 가능한 방안을 검토하여 문제 해결의 전략을 수립하는 단계입니다.
- 네 번째 Execute the most likely approach 단계는 검토된 방안 중에서 가장 기본적인 방법부터 또는 가장 해결 가능성이 높은 방법부터 수행하여 문제 해결에 접근하는 단계입니다.
- 다섯 번째 Check for Success 단계는 문제의 해결여부를 점검하는 단계입니다.
- 마지막 여섯 번째 Tie up Loose ends 단계는 문제 해결을 위해 적용한 방법에 대한 추가적인 문제 발생 여부를 확인하고 잠재적인 문제점이 있는지 여부 등을 점검하며, 끝으로 발생단계부터 종결까지 문서화해서 차후 유사한 문제의 해결을 돕는 마무리 단계입니다.

이와 같은 DETECT 방법을 사용해 각 단계별로 지정된 절차를 통해 발생한 문제에 접근하고 해결 방법을 도출하는 경우 얻을 수 있는 장점은 다음과 같습니다.

먼저, 발생하는 문제와 해결방법을 정형화하고 단계별로 차근차근 접근함으로써 문제 해결에 자연스럽게 접근할 수 있습니다. 또한, 기존에 문제를 해결한 것을 바탕으로 새로운 문제가 발생했을 때 원인을 규명하는데 필요한 논리적 판단력과 대응력을 키우게 됩니다. 이러한 상황을 유발하게 된 원인이 무엇인지를 찾기 위해 윈도우 이벤트 로그나 SQL Server 오류 로그 등을 살펴보고 그 당시에 발생한 상황 등을 추적해 하나씩, 하나씩 정리 하면서 문제의 원인이 될 수 있는 부분과 될 수 없는 부분을 구별하게 되며, 그에 따른 해결방법을 정리해 놓음으로써 향후에 동일한 문제가 재발하거나 유사한 문제가 발생하는 경우 효율적으로 대처 할 수 있게 됩니다. 그리고 이러한 접근이 습관화가 되면, 발생한 문제에 대해서 어느 부분부터 접근하는 것이 문제를 보다 신속하고 정확하게 해결하는 것인지를 직관화하는 능력도 배양할 수 있습니다. 뿐만 아니라, 문제해결을 위해 본인이 습득해야 할 지식이 무엇인지를 명확하게 규명해 줍니다.

간단한 예로 데이터베이스가 갑자기 주의 대상(SUSPECT) 모드로 변경되어 엔터프라이즈 관리자에서 시커멓게 변해 버렸습니다. 이때 “이게 뭔 일일래?” 하는 생각이 들거나, 또는 “관리자에게 문의하세요” 라는 메시지를 받았는데 자신이 관리자인 경우입니다. 그런데 정작 자신은 본인에게나 다른 사람에게 도대체 뭘 어떻게 물어봐야 하는지도 모르는 상황이라면 어떻겠습니까?

이럴 때라도 DETECT 방법에 따라 차근차근 접근해 나가면서 각 단계별로 어떠한 작업을 해야 하고 그런 작업을 하기 위해서는 자신이 무엇을 모르고, 무엇을 알고 있는지를 인식할 수 있게 됩니다.

모르는 부분을 기록해두고 차근차근 학습함으로써 점차 문제 인지능력을 함양할 수 있게 됩니다. 이와 같이 단계별로 문제와 관련된 주변 환경의 지식을 습득함으로써 향후 문제가 발생했을 경우 빠르고 정확하게 발생한 문제를 해결해 나갈 수 있게 됩니다.

이제 윈도우즈 이벤트 로그와 SQL Server 오류 로그를 확인하는 방법을 살펴보고 이어서 DETECT 방법을 사용하여 발생한 문제에 접근하고 해결하는 과정을 통해서 문제해결을 위한 올바른 접근 방법을 체험해 보겠습니다.

이벤트로그와 SQL Server 오류 로그 확인하기

장애나 오류가 발생하게 되면 문제를 올바르게 정의하기 위해서 가장 먼저 해야 하는 일은 윈도우즈 이벤트 로그와 SQL Server 오류 로그를 면밀히 검토하는 일입니다. 이벤트 로그에는 해당 시스템에서 발생한 하드웨어, 소프트웨어와 관련되는 이벤트들이 기록됩니다. 기본적으로 시스템로그, 보안 로그, 응용 프로그램 로그 세가지 종류로 기록되며, 시스템로그에는 운영체제의 이벤트들이 기록되고 보안로그에는 감사된 내용에 따라 로그온 시도, 개체 액세스 등 보안 이벤트들이 기록되며, 응용 프로그램 로그에는 응용 프로그램이나 사용자 프로그램에서 발생한 이벤트 및 정보들이 기록됩니다. SQL Server는 서비스가 시작되면서 발생한 중요한 사항을 윈도우즈 응용 프로그램 로그와 SQL Server 오류 로그에 기록합니다. 따라서, 하드웨어나 운영체제에서 발생한 오류 사항은 이벤트 로그에서 확인하고, SQL Server 자체에서 발생한 오류 사항은 SQL Server 오류 로그에서 가장 먼저 확인해야 합니다. 물론, 오류가 발생했다고 해서 항상 이벤트 로그나 SQL Server 오류 로그에 정보가 기록되는 것은 아닙니다.

■ 이벤트 로그

이제 이벤트 로그를 통해서 운영체제나 SQL Server의 이벤트를 살펴 보겠습니다.

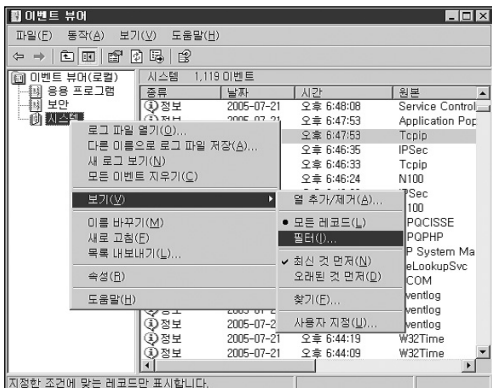


1) 이벤트 뷰어 시작하기

방법① : 시작→관리도구→이벤트 뷰어를 클릭합니다.

방법② : 시작→실행 창에서 eventvwr.msc 입력한 후 엔터 키를 누릅니다.

2) 시스템 로그를 선택한 후 다음과 같이 필터를 선택합니다.



- 3) [이벤트 유형]의 정보, 경고, 오류 체크박스 가운데에서 검색하지 않고자 하는 부분은 선택을 해제합니다. [이벤트 원본]에서 TCP/IP를 선택하고 [확인]버튼을 누릅니다.

시스템 등록 정보

일반 필터

이벤트 유형

☒ 정보(I) ☐ 성공 감사(S)

☒ 경고(W) ☐ 실패 감사(F)

☒ 오류(E)

이벤트 원본(O):

범주(C):

이벤트 ID(I):

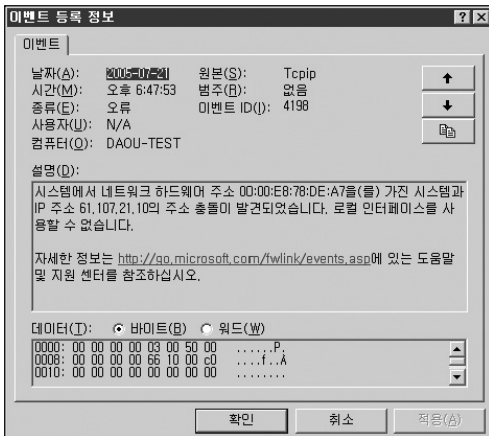
사용자(U):

컴퓨터(C):

시작(S): 2005-06-27

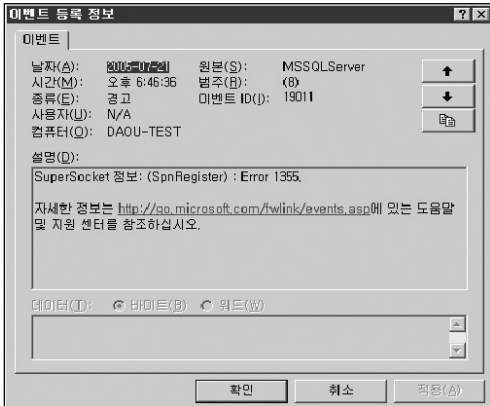
끝(F): 2005-07-25

- 4) 필터된 이벤트 중 오류 표시가 있는 이벤트를 더블클릭합니다. 다음의 경우는 IP 주소가 충돌이 발생했었다는 것을 보여주는 이벤트 로그입니다.



5) 이제 응용 프로그램 로그부분을 살펴 보겠습니다.

2)번과 3)번의 경우와 동일한 방법으로 필터의 [이벤트 원본]에서 MSSQLSERVER를 선택합니다. 경고가 표시된 이벤트를 더블클릭하여 다음과 같은 메시지를 확인합니다.

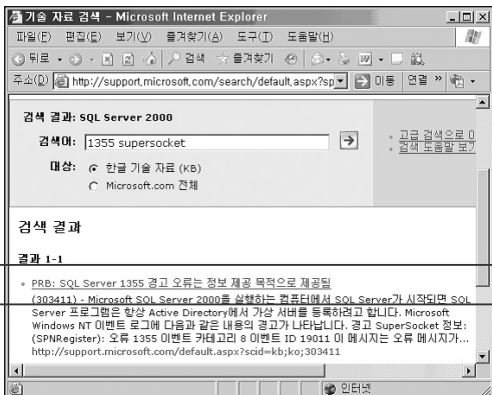
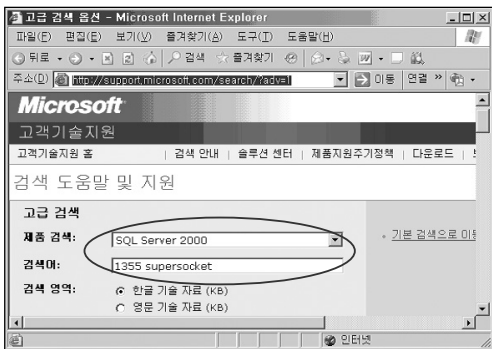


아마, 자주 본 메시지일 것입니다. 이 경우 해당 메시지를 복사해서 마이크로소프트 사이트에서 해당 메시지에 대한 검색을 수행합니다.

1) 인터넷 익스플로러를 실행하고 <http://support.microsoft.com/search/?adv=1> 주소로 이동을 합니다.

2) 제품검색 입력란에 [SQL Server 2000]을 선택합니다.

3) 검색어 입력란에 이벤트 로그에서 확인한 메시지 또는 이벤트 ID를 입력합니다.



- 4) 검색된 자료 가운데 이벤트와 가장 관련이 있을 것으로 예상되는 문서를 순서대로 선택하여 문서의 내용을 확인합니다.
- 5) 내용의 일부를 확인하면 다음과 같습니다.

Microsoft SQL Server 2000 을 실행하는 컴퓨터에서 SQL Server 가 시작되면 SQL Server 프로그램은 항상 Active Directory 에서 가상 서버를 등록하려고 합니다. Microsoft Windows NT 이벤트 로그에 다음과 같은 내용의 경고가 나타납니다.

경고 SuperSocket 정보 : (SPNRegister): 오류 1355

이벤트 카테고리 8

이벤트 ID 19011

이 메시지는 오류 메시지가 아닙니다. 이 텍스트는 SQL Server가 SPN(Service Principal Name)을 등록할 수 없음을 알리는 경고에 불과합니다.

이것은 사용되는 보안 메커니즘이 Kerberos가 아닌

Microsoft Windows NT Challenge\Response(NTLM) 인증임을 나타냅니다

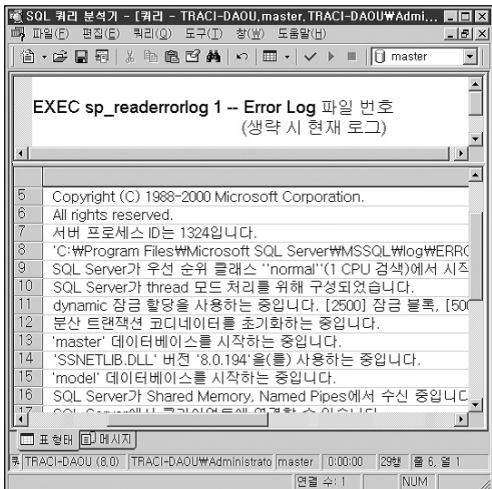
■ SQL Server 오류 로그

1) SQL Server 오류 로그 보기:

방법① : 엔터프라이즈 관리자 -> 관리 -> [SQL Server 로그] 선택

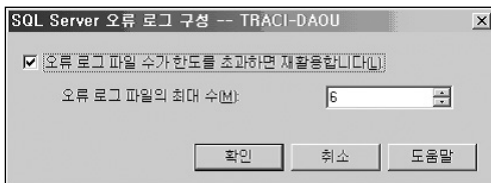
방법② : 쿼리 분석기 -> EXEC sp_readerrorlog 로그 파일 번호

방법③ : 쿼리 분석기 -> EXEC xp_readerrorlog 로그 파일 번호



2) SQL Server 오류 로그는 기본적으로 현재의 로그 파일 하나와 이전 6개의 오류 로그 파일을 유지합니다. 이것은 로그 파일들이 시간이 지남에 따라 덮어 써어진다는 것을 의미하며 필요한 시점의 로그를 확인하지 못할 수도 있습니다. 따라서, SQL Server에서 발생한 이벤트들을 적절하게 보존하기 위해서는 오류 로그 파일의 개수를 늘려 줄 것을 권고합니다.

오류 로그 파일 개수를 변경하려면 [SQL Server 로그]에서 오른쪽 버튼을 클릭한 다음에 [구성]을 선택하고 [오류 로그 파일의 최대 수] 부분에 원하는 수를 입력하면 됩니다.



- 3) SQL Server 서비스가 시작될 때마다 오류 로그 파일이 새로 만들어지기 때문에 정상적인 경우에는 오류 로그 파일의 크기가 지나치게 커지지 않습니다. 그러나 오류가 발생하게 되면 오류 파일이 수십~수백 메가 바이트 이상으로 커지기도 합니다. SQL Server 오류 로그 파일의 개수와 크기 정보는 xp_enumerrorlogs 확장 저장 프로시저를 사용하여 가능합니다.

	보관 #	날짜	로그 파일 크기(바이트)
1	0	08/18/2005 23:09	2011
2	1	08/18/2005 17:30	2218
3	2	08/18/2005 17:20	2218

오류 로그 파일의 크기가 지나치게 커진 경우에는 DBCC ERRORLOG 또는 sp_cycle_errorlog 시스템 저장 프로시저를 실행하면 SQL Server를 재시작하지 않고 즉시 새로운 로그 파일을 생성할 수 있습니다.

```
EXEC sp_cycle_errorlog
go
DBCC ERRORLOG
go
```


4) SQL Server 오류 로그의 내용은 다음과 같은 사항을 포함하고 있습니다.

- ① SQL Server 버전
- ② 운영체제 버전
- ③ 서비스 시작 시간
- ④ 서버 프로세스 ID
- ⑤ Error Log 파일 경로
- ⑥ 서버 실행 모드 (Thread/Fiber)
- ⑦ 수신 프로토콜 및 포트
- ⑧ 시스템 및 사용자 데이터베이스의 인스턴스 복구 프로세스 정보
- ⑨ 서버 옵션 변경 정보
- ⑩ SQL Server 에서 발생된 오류 등

ERRORLOG.1		
1	2005-07-24 19:20:52.68 server	Microsoft SQL Server 2000 - 8.00.760 (Intel X86)
2	Dec 17 2002 14:22:05	
3	Copyright (c) 1988-2003 Microsoft Corporation	
4	Enterprise Edition on Windows NT 5.2 (Build 3790: Service Pack 1)	
5	2005-07-24 19:20:52.68 server	Copyright (C) 1988-2002 Microsoft Corporation.
6	2005-07-24 19:20:52.68 server	All rights reserved.
7	2005-07-24 19:20:52.68 server	서버 프로세스 ID는 1912입니다.
8	2005-07-24 19:20:52.68 server	'C:\Program Files\Microsoft SQL Server\WMSSQL\WinM
9	2005-07-24 19:20:52.79 server	SQL Server가 우선 순위 클래스 "normal"(1 CPU 검색)에서 /
10	2005-07-24 19:20:53.14 server	SQL Server가 thread 모드 처리를 위해 구성되었습니다.
11	2005-07-24 19:20:53.15 server	dynamic 잠금 할당을 사용하는 중입니다. [2500] 잠금 블록, [
12	2005-07-24 19:20:53.18 server	분산 트랜잭션 코디네이터를 초기화하는 중입니다.
13	2005-07-24 19:20:55.46 spid3	'master' 데이터베이스를 시작하는 중입니다.
14	2005-07-24 19:20:55.89 server	'SSNETLIB.DLL' 버전 '8.0.766'을(를) 사용하는 중입니다.
15	2005-07-24 19:20:55.90 spid5	'model' 데이터베이스를 시작하는 중입니다.
16	2005-07-24 19:20:55.90 spid3	서버 이름은 'TRACI-DAOU'입니다.
17	2005-07-24 19:20:55.90 spid6	'msdb' 데이터베이스를 시작하는 중입니다.
18	2005-07-24 19:20:55.90 spid9	'pubs' 데이터베이스를 시작하는 중입니다.
19	2005-07-24 19:20:55.90 spid10	'Northwind' 데이터베이스를 시작하는 중입니다.
20	2005-07-24 19:20:55.95 server	SQL Server가 61.107.21.9: 14330에서 수신 중입니다.
21	2005-07-24 19:20:55.95 server	SQL Server가 127.0.0.1: 14330에서 수신 중입니다.
22	2005-07-24 19:20:55.96 server	SQL Server가 TCP, Shared Memory, Named Pipes에서 수
23	2005-07-24 19:20:55.96 server	SQL Server에서 클라이언트에 연결할 수 있습니다.
24	2005-07-24 19:20:56.26 spid5	tempdb 데이터베이스를 지우는 중입니다.
25	2005-07-24 19:20:57.81 spid5	'tempdb' 데이터베이스를 시작하는 중입니다.
26	2005-07-24 19:20:58.23 spid3	복구가 완료되었습니다.
27	2005-07-24 19:20:58.25 spid3	SQL global counter collection task is created.
28	2005-07-25 18:59:40.45 spid10	Master 데이터베이스(2000-08-20)의(를) 복구하는 과정이

[참고]

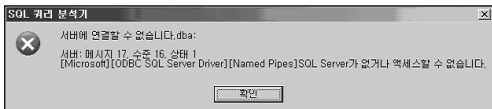
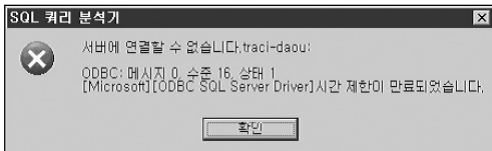
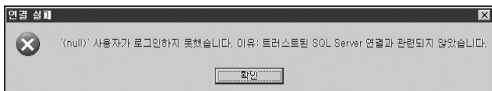
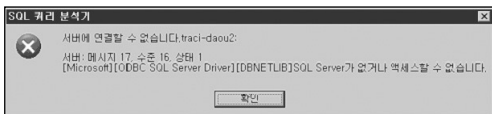
SQL Server 오류 로그와 관련한 자세한 사항은 SQL Server DBA 가이드(전현경 저) 나 온라인 설명서를 참고하기 바랍니다.

SQL Server 와 연결 불가의 경우

성능 문제나 장애사항을 발견하는 첫 번째 Discover 단계는 SQL Server와 클라이언트간에 연결이 실패하는 경우입니다. 연결 불가의 오류 메시지를 받은 상황입니다.

여러분이 첫 번째로 해야 될 일은 발생된 정확한 오류 메시지를 보관해 두는 것입니다.

메시지 박스로 발생되었다면 해당 메시지 박스를 Alt+PrintScrn 키를 누르거나 이미지 도구를 사용하여 다음과 같이 저장하기 바랍니다.



이제 두 번째 단계인 오류 사항을 올바르게 정의하는 Explore the condition 단계입니다.

가장 먼저 이벤트 로그와 SQL Server 오류 로그를 확인하여 발생한 오류와 관련되는 정보의 기록 여부를 점검합니다. 뿐만 아니라 발생한 오류 메시지의 정보를 가지고 마이크로소프트 기술 문서나 다양한 SQL Server 커뮤니티 사이트 등을 통해서 관련 정보를 수집하여 원인 분석과 해결을 위한 참고 자료로 활용합니다.

가장 먼저 둘러야 할 곳은 마이크로소프트사의 기술문서 검색 사이트입니다.

<http://support.microsoft.com/search/?adv=1>

이미 마이크로소프트사에서 해결한 상황이나 참고 자료를 검색함으로써 가장 빠르고 정확하게 문제를 진단하고 해결점에 도달할 수 있습니다. 이곳에서 관련 자료나 해결 방법을 얻지 못한 경우에는 다음의 커뮤니티 사이트들을 통해서 유사한 질의와 답변이 있었는지 점검하고, 원하는 답변이 없는 경우에는 질의를 남기면 전 세계 전문가들을 통해서 가능한 빠르게 답변을 받을 수도 있을 것입니다.

<http://support.microsoft.com/newsgroups/default.aspx>

<http://www.sqlservercentral.com/>

<http://www.databasejournal.com>

<http://www.SQLsecurity.com>

<http://www.SQLTeam.com>

<http://www.SQLcity.com>

<http://www.SQL-server-performance.com>

이와 같은 참고 자료와 함께 오류가 발생한 주변 상황을 면밀히 검토합니다. 기존에는 연결이 잘 되다가 어느 순간부터 연결이 잘 되지 않는 것인지 아니면 클라이언트를 구성하고 처음 연결 시 발생하는 문제인지 구별해야 합니다. 뿐만 아니라 해당 클라이언트만 좌측의 메시지 가운데 하나를 받고 연결이 안 되는 것인지 아니면 모든 클라이언트들에서 연결이 실패하는지 구별해야 합니다.

만약, 다른 클라이언트는 현재도 연결이 잘 되는데 특정 클라이언트에서만 연결이 안된다면 해당 클라이언트의 설정상의 문제일 가능성이 높고, 기존에 연결이 잘되던 클라이언트 여러 대가 앞에서 살펴본 연결 불가 메시지를 동일하게 받았다면 해당 클라이언트들과 서버 간의 네트워크상의 문제일 가능성이 높다는 것을 추리할 수 있습니다.

만약, 해당 클라이언트를 포함해서 동시에 모든 클라이언트가 연결되지 않는 상황이라면 모든 클라이언트가 공용으로 공유하는 부분이나 서버 측의 문제일 가능성이 높습니다.

이와 같은 주변환경을 조사함으로써 발생된 문제를 명확하게 정의하도록 합니다.

이제 여러 가지 케이스를 가지고 세 번째 단계인 문제를 해결하기 위해서 전략을 수립하고 범위를 좁혀 나가는 Track down possible approach 단계와 네 번째 단계인 Execute the most likely approach 단계로 넘어가 해결에 접근해 보도록 하겠습니다.

CASE 1. 다른 클라이언트들에서는 이상이 없는 상황에서 최초 구성된 클라이언트만 연결이 실패하는 경우

1. 가장 먼저 이벤트 로그와 시스템 로그를 확인합니다. IP 충돌이나 케이블 장애와 같은 해당 클라이언트의 네트워크상의 문제점이 있다면 시스템 로그에 기록이 남아 있을 것입니다. 클라이언트를 설치할 때 발생한 오류 사항은 응용 프로그램 로그에 기록될 수 있습니다. 이벤트 로그에 표시된 오류가 기록되어 있는 경우에는 적절한 해결을 시도합니다. 이상이 없는 경우에는, 운영체제 상에서 네트워크로 서버와 통신이 되는지를 먼저 점검합니다. 시작->실행에서 CMD를 입력하고 엔터 키를 눌러서 명령 프롬프트를 실행합니다. Ping 유틸리티를 실행해서 네트워크 연결 여부를 확인합니다.

- ① Ping <LOCAL ADDRESS> : 시스템에 TCP/IP 가 정상적으로 구성되었는지 확인합니다.
- ② Ping <동일 네트워크 machine의 IP ADDRESS> : 동일 네트워크상의 다른 컴퓨터와 통신할 수 있는지 확인합니다.
- ③ Ping <GATEWAY / SQL Server IP ADDRESS> : 라우터나 SQL Server와 통신할 수 있는지 확인합니다.
- ④ 통신이 불가능한 경우는 Tracert < GATEWAY / SQL Server IP ADDRESS>를 실행해서 어느 부분에서 연결이 되지 않는지 확인합니다.

해당 서버가 이상이 없다면 포트와의 연결 여부를 확인합니다. SQL Server와 클라이언트간의 통신 여부를 마이크로소프트가 제공하는 다양한 툴을 통해서 확인할 수 있습니다. 다음에서 소개하는 portqry 유틸리티는 SQL Server가 지정된 포트에서 정상적으로 수신 가능한지 여부를 확인할 수 있습니다. 먼저 portqry 유틸리티를 설치합니다. 다운로드 사이트는 다음과 같습니다.

〈PortqryV2.exe 다운로드 사이트〉

<http://support.microsoft.com/kb/310099/>

〈PortqryV2.exe 관련 기술 문서〉

<http://support.microsoft.com/kb/832919/>

명령 프롬프트에서 설치 경로(C:\PortqryV2)로 이동합니다.
다음의 매개 변수를 참고하여 portqry 유틸리티를 실행합니다.

```

C:\WINDOWS\system32\cmd.exe
C:\PortQryV2>PortQry -n traci-daou -p tcp -e 1433
Querying target system called:
traci-daou
Attempting to resolve name to IP address...
Name resolved to 61.107.21.9
querying...
TCP port 1433 (ms-sql-s service): LISTENING
  
```

〈1433 포트가 정상적으로 수신되는 경우〉

〈Portqry.exe 매개 변수〉

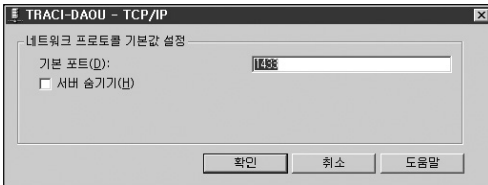
-n : 대상 호스트 이름 또는 IP 주소

-p : 대상 프로토콜 (tcp, udp)

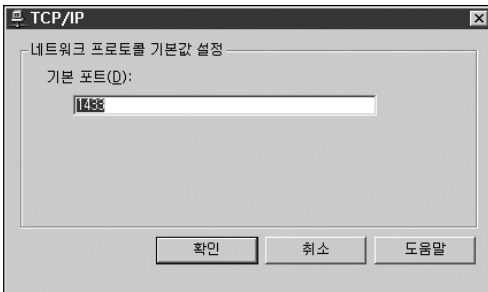
-e : 대상포트

해당 포트가 NOT LISTENING 인 경우 SQL Server가 해당 포트를 바인딩하지 못했거나 다른 포트를 사용하고 있을 수도 있습니다. 다른 클라이언트들은 연결이 가능한 상태이므로 해당 클라이언트의 MDAC 구성을 확인하는 것이 필요합니다.

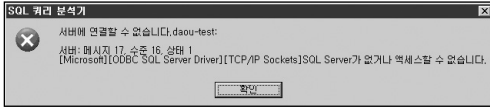
서버의 네트워크 구성 유틸리티를 실행하여 구성된 서버가 사용할 수 있는 프로토콜 및 해당 프로토콜의 포트 등의 구성과 해당 클라이언트의 MDAC 구성이 상호간 통신이 가능한지 여부를 확인합니다. 다음은 TCP/IP 네트워크 라이브러리의 경우입니다.



〈서버의 구성〉



〈클라이언트 구성〉



서버가 사용하는 포트와 클라이언트에 지정된 포트가 다르면 위와 같은 오류 메시지가 반환됩니다. 클라이언트가 접속하고자 하는 SQL Server의 포트가 지정되지 않은 경우 UDP 1434 포트를 통해 접속하고자 하는 SQL Server가 사용중인 포트를 질의합니다. 따라서 서버가 사용중인 포트가 클라이언트 네트워크 유틸리티나 연결 문자열에 정확하게 지정되었는지 또는 UDP 1434 포트가 사용 가능한지 등을 확인하여야 합니다.

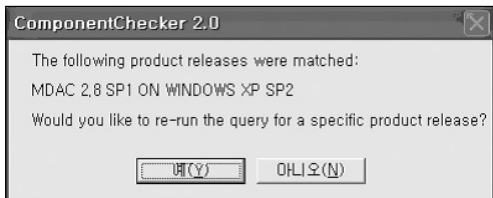
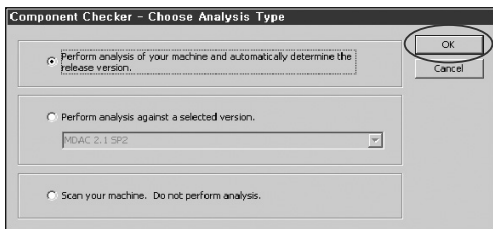
2. MDAC 구성에 이상이 없다면 MDAC 버전을 확인합니다. MDAC의 하위 버전은 상위 버전과 호환됩니다. 단 클라이언트 프로그램이 요구하는 기능에 따라서 최신 버전의 MDAC이 필요한 경우나 클러스터가 구성된 경우 해당 환경에서의 MDAC 설치와 관련된 사항을 기술문서를 통해서 확인한 후 설치하여야 합니다.

MDAC 버전을 확인하는 유틸리티와 MDAC을 버전 별로 다운로드 할 수 있는 사이트로 접속합니다.

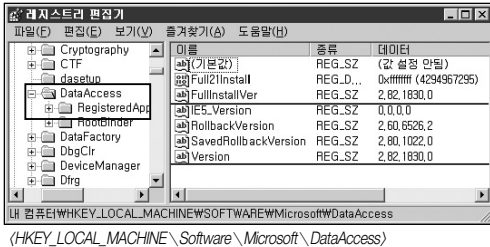
〈MDAC 다운로드 사이트〉

<http://msdn.microsoft.com/data/mdac/downloads/default.aspx>

MDAC 버전을 확인하는 Component Checker 유틸리티를 설치한 후 해당 클라이언트에서 실행하고 [OK] 버튼을 누르면 다음과 같이 버전을 확인할 수 있습니다.



또는 레지스트리에서도 확인이 가능합니다.

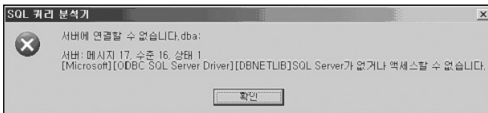


그러나 SQL Server간에 통신하거나 쿼리 분석기와 같은 SQL Server 클라이언트 도구를 사용하는 경우에는 상위 버전은 하위 버전과 통신이 불가능한 경우가 생기게 됩니다. 이런 경우는 상호 호환 가능한 서비스 팩을 설치해야 합니다.

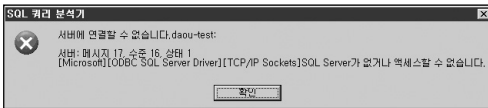
다음은 Windows 2003 Enterprise 서버에 SQL Server RTM 버전이 설치되어 있는 예입니다. 클라이언트의 MDAC 버전을 확인하면 2.82가 설치되어 있지만 SQL Server는 SSNETLIB 8.0.194를 사용하므로 상위 버전의 SQL Server에서는 접속하지 못하는 문제가 발생할 수 있습니다.



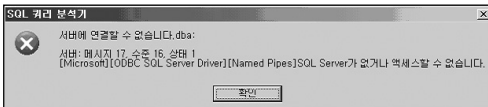
다음은 서비스 팩3 또는 서비스 팩4가 설치된 SQL Server에서 RTM 버전의 SQL Server에 접속하려고 할 때 발생하는 오류 메시지입니다



위의 경우는 SQL Server의 기본 사용 네트워크 라이브러리인 TCP/IP 와 명명된 파이프 (Named Pipe)로 각각 통신을 시도한 후 연결할 수 없음을 표시하는 오류 메시지입니다.



〈TCP/IP 네트워크 라이브러리로 연결 시도 시 오류 메시지〉



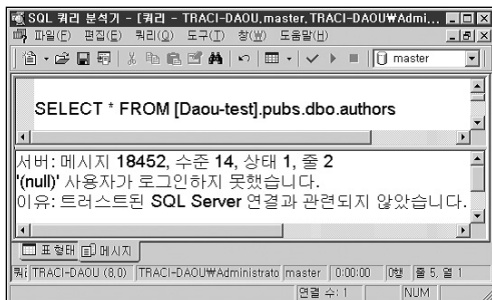
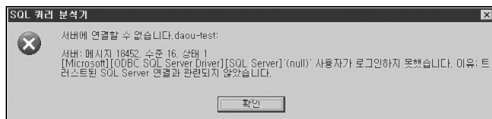
〈명명된 파이프로 연결 시도 시 오류 메시지〉

[주의]

만약 SQL Server 클라이언트 툴 (엔터프라이즈 관리자 또는 쿼리 분석기 등)을 사용하는 경우는 서버와 동일한 서비스 팩을 설치해야 하기도 합니다.

CASE 2. 특정 클라이언트에서만 연결이 잘 되다가 갑자기 실패하는 경우

CASE 1의 1단계와 2단계를 실행하여 해당 클라이언트의 운영체제와 하드웨어의 문제를 점검하고 이상이 없는 경우에는 연결이 실패하기 직전에 수행했던 작업을 점검해야 합니다. 최근에 설정을 변경하였거나 설치된 프로그램은 없는지도 점검해야 합니다. 액티브 디렉토리를 구성하여 중앙에서 계정관리를 하지 않는 경우 해당 클라이언트에서 사용하는 윈도우즈 계정의 패스워드를 변경하고, 접속하고자 하는 SQL Server에 있는 윈도우즈 계정의 패스워드를 변경하지 않으면 다음과 같은 오류 메시지를 발생립니다.



따라서, 클라이언트에서 최근에 변경된 사항을 중심으로 해결점에 접근합니다.

CASE 3. 일부 클라이언트들에서 연결이 실패하는 경우

기존에 연결이 잘되는 상황에서 일부 클라이언트에서만 연결이 되지 않는다면 스위치나 라우터 등 연결이 불가한 클라이언트들이 공통으로 사용하고 있는 부분에 초점을 맞추어 찾아 보아야 합니다.

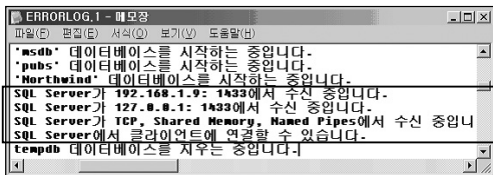
CASE 4. 모든 클라이언트들에서 연결이 실패하는 경우

1. 가장 먼저 운영체제의 실행 여부를 확인합니다. Power Supply 등의 하드웨어 문제로 다운되었을 수도 있습니다. 서버의 이벤트 로그와 SQL Server 오류 로그를 면밀히 살펴 보는 것이 필수입니다. 이어서 SQL Server 서비스의 실행 여부를 확인합니다. 이상이 없다면 서버의 로컬에서 쿼리분석기나 엔터프라이즈 관리자 등을 통한 연결 테스트를 진행합니다. 서버가 통신 포트를 바인딩하지 못한 것이 원인일수도 있습니다. 이벤트 로그를 통해 확인할 수 있으며 명령 프롬프트에서 아래 명령을 실행해서 수신중인 TCP 1433 포트(해당 서버의 SQL Server 사용 포트), UDP 1434 포트 등의 수신 대기를 확인합니다.

```
C:\w>netstat -an (수신 대기 포트 정보 확인)
```

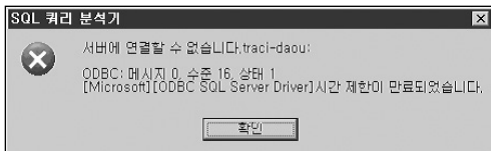
```
C:\w>netstat -anb (수신 대기 포트 정보 및 해당 응용 프로그램 확인)
```

SQL Server 오류 로그를 통해서도 수신 대기 중인 네트워크 라이브러리를 확인할 수 있습니다.



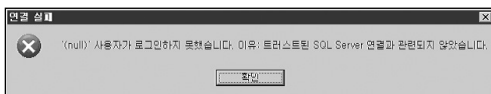
서버 자체의 네트워크 문제가 아니라면 라우터나 방화벽, DNS서버나 WINS 서버 등 SQL Server와 클라이언트 사이에 위치한 공통 사용 영역이 문제일 수도 있습니다. PING, TRACERT 등의 명령을 통해서 통신이 가능한 범위를 점검하여 문제의 원인이 되는 지점에 접근합니다. 방화벽을 사용하는 경우 SQL Server가 기본 구성의 포트를 사용하는 경우 TCP 1433 포트와 UDP 1434 포트를, 별도의 포트를 지정해서 사용하는 경우 각 인스턴스 별로 사용하는 포트를 방화벽에서 허용해야 합니다.

2. SQL Server가 로컬에서도 연결이 불가능한 경우는 CPU나 메모리, 네트워크 등이 과도하게 사용되는 경우 연결을 기다리다가 시간 제한에 걸려 연결이 불가능한 경우도 예상할 수 있습니다.

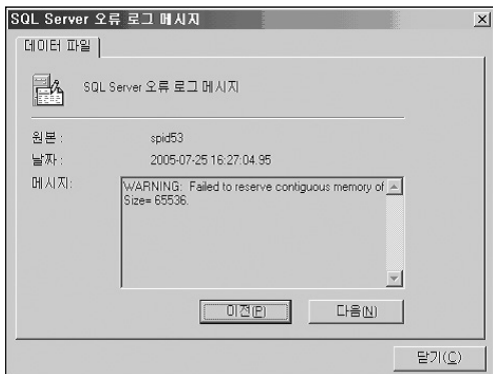


이런 경우는 시스템 모니터를 통하거나 작업관리자를 사용하여 현재 사용중인 자원의 상태를 모니터링할 수 있으며 잘못 설치된 하드웨어 드라이버 등이 원인이기도 합니다. 뿐만 아니라, 자원을 과도하게 사용하는 경우는 웜 바이러스 등에 감염된 것이 원인이 될 수도 있습니다. 이런 경우는 서버에 최신의 핫픽스와 서비스 팩의 적용을 미루거나 바이러스 백신 프로그램 등을 적절하게 업데이트하지 못한 결과입니다.

3. 시스템의 메모리 단편화 문제는 기존의 연결에서의 작업 수행은 가능하지만 새로운 연결은 실패하는 문제를 유발합니다. 메모리 단편화 문제는 SQL Server가 새로운 연결을 위해 필요로 하는 메모리의 연속적인 공간을 할당할 만한 메모리가 지원되지 않기 때문입니다. 윈도우즈 인증을 사용하는 경우 다음과 같은 오류 메시지와 함께 연결되지 못합니다. 그러나 SQL 인증을 사용하는 경우에는 정상적으로 연결이 이루어지기도 합니다.

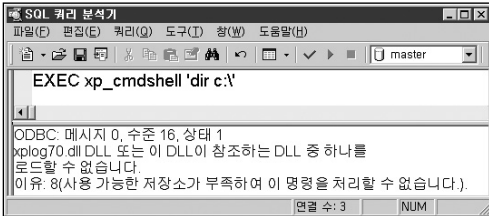


위의 오류 메시지는 윈도우즈 인증으로 인증을 요청할 때 패스워드가 일치하지 않는 경우 발생한 오류 메시지와 유사하기 때문에 주의하기 바랍니다. 반드시 SQL Server 오류 로그를 통해서 발생한 문제를 정확하게 정의하여야 합니다. 먼저 패스워드를 확인하여 윈도우즈 인증의 오류인지를 점검하고, SQL Server 오류 로그를 확인하여 관련된 메시지가 있는지 점검합니다. 메모리 단편화의 경우 다음과 같은 오류 메시지를 확인할 수 있습니다.

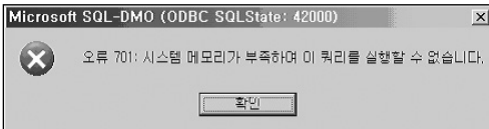


이런 경우에는 적절하게 작성되지 않은 응용 프로그램이나 SQL Server용 확장 저장 프로시저가 문제가 된다고 진단할 수 있습니다. SQL Server는 확장 저장 프로시저로 사용되는 DLL파일의 실행을 모두 SQL Server 오류 로그에 기록하므로 SQL Server가 시작된 뒤 실행된 DLL 파일을 검사합니다.

메모리 단편화의 경우 기존 연결의 경우도 특정한 작업을 실행하게 되면 다음과 같은 오류 메시지를 반환하기도 합니다.

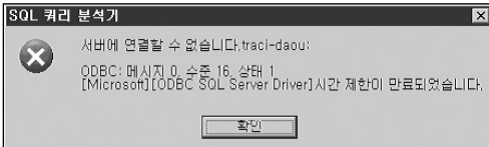


〈확장 저장 프로시저 XP_CMDSHELL 을 실행한 경우〉



〈엔터프라이즈 관리자에서 권한 탭을 클릭한 경우〉

4. 과도한 스레드(Thread)가 생성되어 서버의 구성 옵션인 "max worker threads"의 임계 값에 도달한 경우 해당 스레드들이 종료될 때까지 새로운 연결이 불가능한 경우도 있습니다. 이런 경우에도 역시 다음과 같은 오류 메시지가 반환되기도 합니다.



이러한 경우를 확인하기 위해 SQL Server 진단 유틸리티인 SQLDiag.exe를 실행하여 원인을 찾아 보겠습니다.

명령 프롬프트를 실행하여 SQL Server가 설치되어 있는 “C:\Program Files\Microsoft SQL Server\MSSQL\Binn” 폴더로 이동하여 SQLDiag.exe를 실행합니다.

SQLDiag.exe를 실행하면 기본적으로 오류 로그 폴더” C:\Program Files\Microsoft SQL Server\MSSQL\LOG”에 SQLdiag.txt 라는 결과 파일이 만들어집니다.

SQLDiag.exe 유틸리티는 다음 과정을 수행합니다.

- ① Error Log 수집
- ② EXEC sp_who2
- ③ EXEC sp_lock
- ④ EXEC sp_configure
- ⑤ EXEC xp_msver
- ⑥ EXEC sp_helpextendedproc
- ⑦ Sysprocesses 시스템 테이블 정보
- ⑧ DBCC INPUTBUFFER
- ⑨ 실행중인 프로세스 정보
- ⑩ 레지스트리 정보
- ⑪ DLL 버전 정보
- ⑫ 운영체제 및 H/W 정보 등

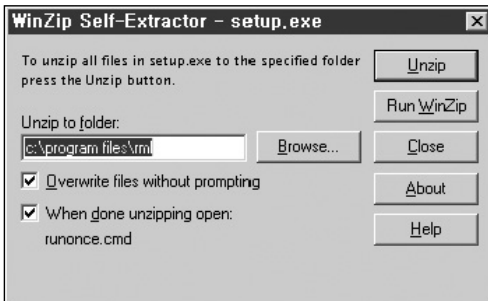
그러나 작업자 스레드의 개수가 “max worker threads”의 임계값(기본 설정 255개)에 도달하여 추가적인 스레드를 할당받을 수 없는 상태라면 새로운 연결을 필요로 하는 위의 ②~⑨번 작업까지의 결과를 출력하기 위해서 대기하게 됩니다. SQLdiag.txt를 살펴보면 ① Error Log 부분에서 17884 메시지를 발견하게 됩니다. 이것은 CPU당 하나씩 있는 SQL Server User Mode Scheduler가 새로운 스레드를 생성하지 못한다는 것을 의미합니다. 이 경우 SQL Server는 이와같은 상황을 교착상태로 처리합니다. 따라서, 추적 플래그(Trace flag) 1204를 지정하여 교착상태 발생과 관련해서도 관련된 정보를 수집할 수 있습니다. 마이크로소프트의 기술 문서에는 다양한 경우에 발생하는 17884 오류에 대한 설명과 핫픽스를 제공하고 있습니다. 만약, 단순히 동시 사용자가 많다면 “max worker threads” 임계 값을 적절하게 조정하여 이와 같은 현상을 방지할 수 있습니다.

다음은 마이크로소프트가 제공하는 OSTRESS 유틸리티를 사용하여 스레드 할당이 “max worker threads” 임계값에 도달하는 경우를 살펴 보겠습니다.

〈 OSTRESS 유틸리티 다운로드 사이트〉

<http://www.microsoft.com/downloads/details.aspx?FamilyId=5691AB53-893A-4AAF-B4A6-9A8BB9669A8B&displaylang=en>

위의 사이트에서 OSTRESS 유틸리티를 다운로드 받고 setup.exe를 실행합니다.



OSTRESS 유틸리티를 실행하기 위한 환경은 다음과 같습니다.

- Windows 2000 (Server/Professional), Windows XP, or Windows 2003 Server
- 128 MB RAM minimum,

시작-> 실행-> CMD를 입력한 후 엔터 키를 눌러서 명령 프롬프트를 실행합니다.

프로그램이 설치된 C:\Program files\rmf 폴더로 이동하여 OSTRESS.EXE /? 를 입력하여 실행 매개 변수를 확인합니다.

```
C:\WINDOWS\system32\cmd.exe

C:\Program Files\msl>OSTRESS /?
OSTRESS, A Generic ODBC-Based Stress/Replay Utility, Version 8.10.0010
Copyright (c) Microsoft Corporation 1997-2004. All rights reserved.

USAGE: ostress [-S server] [-d database] | [-D data source name]
        [-U login ID] [-P password] | [-E trusted connection]
        [-Q "single batch query"] | [-i query file with GO delimited batches]
        [-l login timeout (sec)] [-t query timeout (sec)]
        [-n number connections (one thread per connection)] [-r iterations]
        [-o output file] [-e error file] [-c control file (advanced options)]
        [-q quiet mode] | [-v verbose mode]
        [-m [stress | replay]] [-T trace flag]
        [-N No "OSTRESS exiting" message]

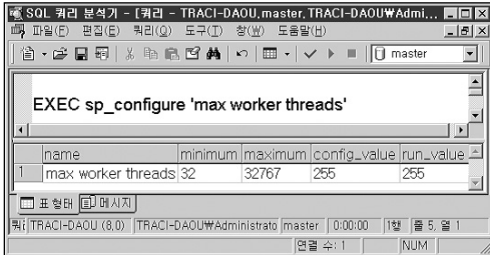
EXAMPLE: ostress -E -dpubs -Q"select * from authors"
```

〈OSTRESS 유틸리티 매개 변수〉

- S: 서버이름
- d: 데이터베이스 이름
- D: DSN 이름
- U: 로그인 아이디
- P: 패스워드
- E: 윈도우즈 인증 사용
- Q: 실행할 쿼리
- i : 쿼리가 저장된 파일
- o: 출력 파일
- n: 실행할 스레드 개수 (STRESS 테스트를 위해 동시 사용자 수를 가정하여 지정)

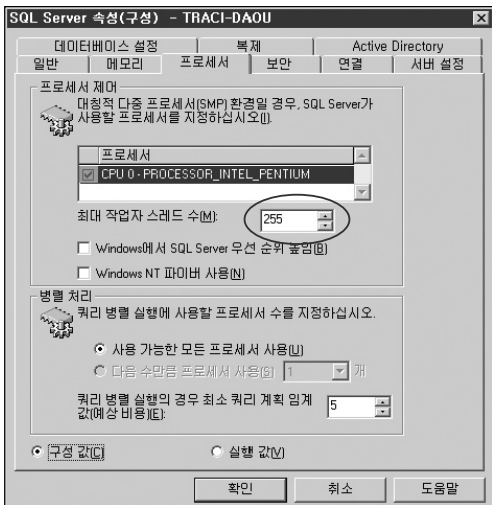
이제 작업자 스레드의 개수가 기본 설정 임계값인 255에 도달하게 되면 어떤 현상이 발생하는지 테스트해 보겠습니다.

① 쿼리 분석기를 실행하여 다음과 같이 쿼리를 입력합니다.

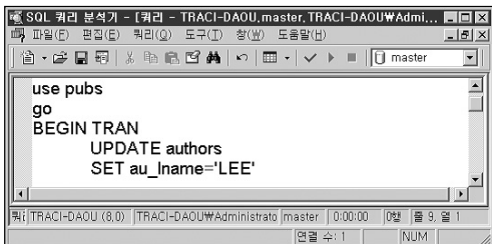


위의 결과를 살펴보면 “max worker threads”는 최소 32개에서 최대 32,767개까지 설정 가능하고 현재의 구성값은 255개이며, 현재 적용되어 있는 값도 255개라는 결과를 보여주고 있습니다.

다음과 같이 엔터프라이즈 관리자를 통해서도 확인 가능합니다. 서버의 속성 창을 실행해서 [프로세서] 탭을 클릭합니다. [최대 작업자 스레드 수] 부분의 값을 확인합니다.



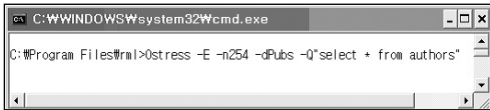
- ② 쿼리 분석기에서 다음과 같은 쿼리를 입력하고 실행합니다. pubs 데이터베이스의 authors 테이블의 각 행에 각각 배타적 잠금(Exclusive Lock)이 걸리게 됩니다.



- ③ 명령 프롬프트를 실행하고 OSTRESS가 설치된 경로로 이동합니다.



- ④ 이제 254개의 활성 스레드를 생성합니다. 실행되는 254개의 스레드는 모두 ②에서 실행된 트랜잭션의 잠금으로 대기하게 됩니다.

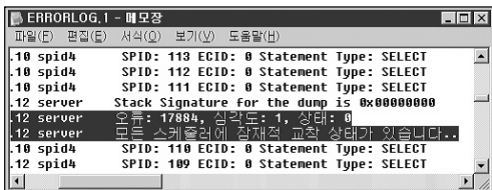


- ⑤ 이제 쿼리 분석기에서 추가로 새로운 창을 열어봅니다. 연결이 되기 위해서는 활성 스레드를 필요로 하는데 이미 255개의 작업자 스레드가 실행 중이므로 추가로 SQL Server에 연결하지 못함을 확인할 수 있습니다.

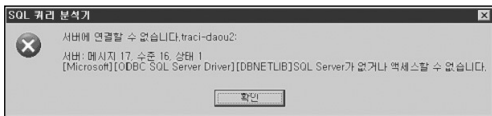
- ⑥ 쿼리 분석기 창으로 돌아가서 롤백을 실행하여 잠금을 해제합니다.

- ⑦ 명령 프롬프트에서 SQL Server가 설치된 "C:\Program File\Microsoft SQL Server\MSSQL\Binn" 폴더로 이동하여 SQLDiag.exe를 실행합니다.

⑧ SQLDiag의 출력 파일을 열어 보면 17884 메시지를 발견하게 됩니다.

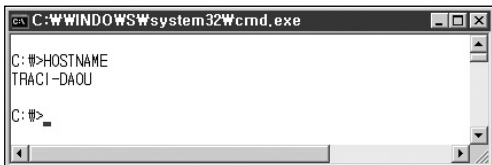


5. SQL Server를 설치한 이후 컴퓨터의 이름을 변경한 경우에도 로컬에서 Shared Memory를 이용한 접근이 아닌 경우 연결이 실패합니다. 클라이언트에서는 다음과 같은 오류 메시지를 받게 됩니다.

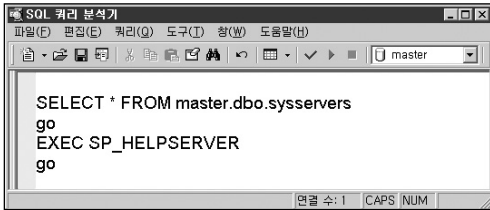


SQL Server의 기본 인스턴스의 이름은 설치된 컴퓨터의 이름과 같도록 구성되고 이 정보는 master 데이터베이스의 sys.servers 시스템 테이블에 기록됩니다. 기록된 인스턴스의 이름 정보는 master 데이터베이스의 sys.servers 시스템 테이블이나 sp_helpserver 시스템 저장 프로시저를 이용하여 확인할 수 있습니다.

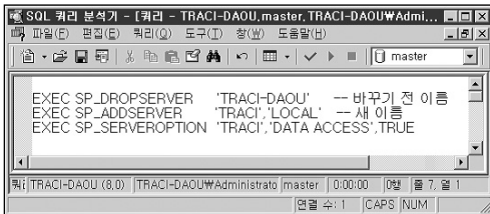
① 명령 프롬프트를 실행한 후 hostname 을 입력하고 엔터 키를 눌러서 현재 컴퓨터의 이름을 확인합니다.



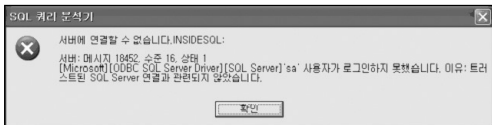
- ② master.dbo.sys.servers 시스템 테이블을 조회하거나 sp_helpserver 시스템 저장 프로시저를 실행하여 현재 인스턴스 이름의 등록 정보가 정확한지 확인합니다.



- ③ 컴퓨터 이름과 등록된 SQL Server 인스턴스 이름이 다른 경우 직접 master 데이터베이스의 sys.servers 시스템 테이블을 수정하거나 sp_dropserver, sp_addserver, sp_serveroption 시스템 저장 프로시저를 실행하여 기존에 등록된 서버를 삭제하고 새로 변경된 서버의 이름을 등록합니다. sp_serveroption 시스템 저장 프로시저는 sys.servers 시스템 테이블에 등록된 서버의 등록 옵션을 지정하는 프로시저입니다.



6. SQL Server의 인증 모드 설정이 윈도우즈 인증인 경우 SQL 로그인으로 인증을 요청하게 되면 다음과 같은 오류 메시지가 반환됩니다.

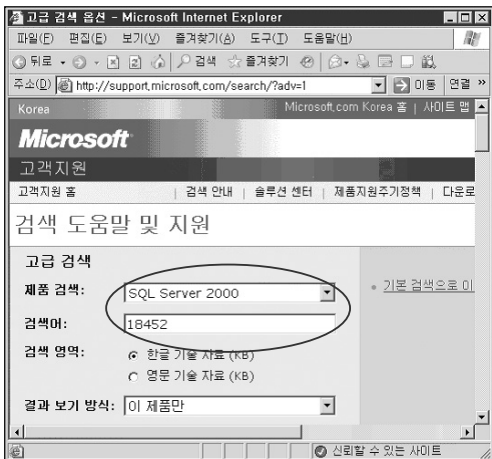


이번에는 마이크로소프트 사이트에서 관련 메시지를 가지고 검색해 보겠습니다.

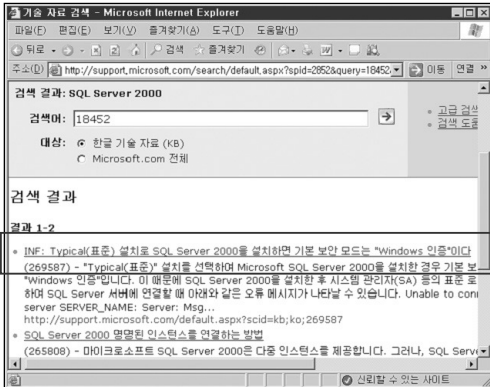
〈마이크로소프트 자료 검색 사이트 주소〉

<http://support.microsoft.com/search/?adv=1>

위에 표시된 사이트에 접속해 제품 검색에 “SQL Server 2000”을 선택하고 검색어 부분에 오류 메시지 번호 또는 발생한 메시지를 입력하여 검색합니다.



이제 화살표를 누르면 다음과 같은 결과가 반환됩니다.



INF: Typical(표준) 설치로 SQL Server 2000을 설치하면 기본 보안 모드는 "Windows 인증"이다 라는 부분을 클릭하여 내용을 보면 (INF란 단순 정보로 분류되었다는 의미입니다.)

```
Unble to connect to server SERVER_NAME:
Server: Msg 18452, Level 16, State 1[Microsoft] [ODBC SQL Server Driver]
[SQL Server]
Login failed for user 'sa' . Reason: Not associated with a trusted SQL Server
connection.
```

앞에서 보았던 것과 동일한 오류 메시지를 발견할 수 있습니다. 또한 이에 대한 해결방법으로 아래의 지시에 따라 인증모드를 변경함으로써 해결할 수 있다는 사항도 발견할 수 있습니다.

인증 모드를 “Windows NT Authentication Mode(only)”에서 “Mixed Mode”로 변경하려면 아래 단계를 수행합니다.

1. Enterprise Manager(엔터프라이즈 관리자)를 엽니다.
2. Server 그룹을 확장합니다.
3. 서버 이름을 마우스 오른쪽 단추로 누른 다음 Properties를 누릅니다.
4. Security 탭을 누릅니다.
5. Authentication에서 SQL Server and Windows 옵션 단추를 누릅니다.

또는 레지스트리를 통해서도 SQL Server의 인증 모드의 확인과 변경이 가능합니다. 레지스트리 편집기를 실행합니다. 다음의 레지스트리 키를 확인합니다.

HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\MSSQLServer\MSSQLServer\LoginMode



REG_DWORD 값이 1인 경우는 윈도우즈 인증만 가능한 설정이고 2인 경우는 혼합 모드로 윈도우즈 인증과 SQL 인증이 모두 가능한 상태입니다. 값을 변경하는 경우에는 SQL Server 서비스를 재시작해야 합니다.

[경고]

레지스트리 편집기를 잘못 사용하면 심각한 문제가 발생할 수 있으며 이 경우 문제를 해결하기 위해 운영 체제를 다시 설치해야 할 수도 있습니다.

위에 열거한 방법들을 통해 해결점에 도달했다고 판단하셨습니까? 물론 위에서 열거한 경우 외에 다양한 원인이 있을 수 있습니다.

이제 다섯 번째 단계인 Check for Success 단계로 들어가 보겠습니다. 성공 여부를 면밀히 점검해야 합니다. 동일한 오류 메시지가 여러 가지의 경우에 발생 가능하다는 것을 확인할 수 있었습니다. 또한, 단순히 눈에 보이던 현상만 해결되었다고 해서 문제가 완전히 종결된 것은 아닐 수도 있기 때문입니다. 문제가 해결된 것으로 판단된다면 원인을 찾아 내었을 것이고, 그렇다면 테스트 환경을 구축하고 발생했던 문제와 동일한 상황을 이끌어 내고 앞에서 성공한 방법으로 다시 한번 확인하기 바랍니다. 그리고, 잠재적인 다른 문제점들은 없는지 점검해야 합니다.

“모로 가도 서울만 가면 된다” 는 말은 아주 위험한 발상입니다. 오히려 나중에 더 큰 문제를 야기할 수도 있기 때문입니다. 해결은 하였으나 그 방법이 적절하지 않아 또 다른 문제를 야기하는 경우도 있을 수도 있고 그 방법보다 더 효율적인 방법이 있을 수도 있기 때문에 마지막 단계인 Tie up Loose ends 단계로 넘어가서 발생한 문제의 원인과 해결방법의 정확성을 점검하고 이것을 문서로 정리하여 향후 발생하는 문제에 대해 참고 자료로 활용토록 해야 합니다.

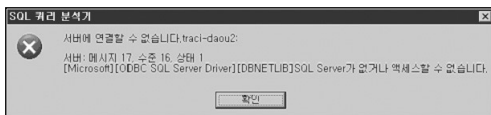
또한, 앞 단계에서 진행한 해결 방법과 원인의 진단이 정확했는지 인터넷의 관련 사이트 등을 통해서 유사한 문제와 해결 방법 등에 대한 사항을 꼼꼼히 점검해 보아야 합니다. 앞의 단계에서 해결점에 도달하지 못한 경우에도 인터넷에 공개되어 있는 사례를 수집하여 유용하게 참고자료로 활용할 수 있습니다.

자, 이제 여러분이 문제 해결 과정에서 정리한 자료와 인터넷 검색을 통해 얻어낸 관련 자료들을 모아 비고 정리하여 다음의 예와 같이 장애 일지나 오류 발생 일지와 함께 보관하면, 일련의 문제 해결 과정이 완전하게 마무리됩니다.

오 류 발 생 일 지

분류 번호	2005-0087	운영 팀	DB 팀
발생 일자	2005/07/01	운영 자	김 용 군
처리 일자	2005/07/01	처리 담당자	황 희 철
장애 구분	H/W() OS () DBMS (O) Network () 기타서버() 응용 프로그램()		
증 상			

- ① 개발 서버에서 엔터프라이즈 관리자 또는 쿼리 분석기로 운영 서버에 접속 시 다음과 같은 오류 메시지가 반환되고 연결되지 않음



- ② 개발 서버에서 운영 서버로의 Linked Server 연결 시 아래와 같은 오류 메시지가 반환되고 연결되지 않음

서버: 메시지 17, 수준 16, 상태 1, 줄 1
SQL Server가 없거나 액세스할 수 없습니다.

오류 발생 시 환경

- ① 개발 서버에 서비스 팩 3 적용
- ② 운영 서버는 현재 서비스 팩 1이 설치되어 있고 서비스 팩 3은 미 적용 상태임
- ③ 상기 사항 외에 운영 서버에 패치 등의 프로그램 설치 작업은 수행된 바 없음.

조치 사항

- ① 운영 서버에 서비스 팩 3 설치

주의 및 확인 사항

- ① 서비스 팩 설치 시 스탠바이 서버와 같은 읽기 전용 데이터베이스가 존재할 경우 설치가 실패함 (서비스 팩 3 설치 전에 읽기 전용 속성을 해제하여야 하며 스탠바이 데이터 베이스는 재구성하여야 함)

기 타 사 항

참고 자료 : 서비스 팩 프로그램 폴더의 sp3readme.htm

쿼리의 응답속도가 갑자기 느려지는 경우

평상시에는 이상이 없었는데 어느 날 갑자기 쿼리의 응답속도가 느려지는 경우를 아마 한번 겪은 경험했을 겁니다. 그런 경우 원인은 어떤 것이었나요? 해결은 어떻게 하셨나요? 물론 같은 시간 대에 백업 작업이나 배치 작업이 실행되고 있었다면 당연히 성능상에 영향이 있었을 겁니다. 이런 간단한 경우는 어떻게 해결하면 될까요? 업무량이 많은 시간대에 백업 작업이 실행된다는 것은 아무래도 효율적이지 못하니까 스케줄을 조정해 주는 것도 하나의 방법일 겁니다. 그런데, 그 시간대에 어떤 배치작업이 업무상 반드시 실행되어야 한다면 어떻게 하겠습니까? 만약 SQL Server가 CPU 4개가 작동하는 4-Way 시스템이라고 가정해 보겠습니다. 그렇다면 CPU의 “max degree of parallelism” 옵션을 4보다 작은 값으로 조정하여 비용이 많이 소모되는 작업을 한다 할지라도 해당 작업이 모든 CPU를 점유하지 않도록 하는 것도 하나의 방법이 될 수 있을 겁니다. 또는 해당 배치를 튜닝하여 부하를 줄이는 것도 방법이 될 수 있습니다.

그러나, 현재 상황은 그와 같은 일이 없다고 가정한 경우입니다.

갑자기 느려진 쿼리를 실행하는 프로세스 외에 이렇다 할 만한 대형 작업이 없는 경우라면 어떻게 해결해야 할까요?

다시, DETECT 방법을 이용해서 점검해보도록 하겠습니다.

첫 번째 Discover 단계는 쿼리의 응답속도가 갑자기 느려졌다는 것을 인지한 상황입니다.

두 번째 Explore 단계는 이와 같은 상황을 보다 명확히 정의해야 하는 작업입니다. 갑자기 느려지기 전에 어떤 일이 있었는지를 점검해 보도록 하겠습니다. 스케줄을 조사해보니 대량의 배치가 돌아서 상당한 양의 데이터가 새롭게 로딩되었습니다. 월말 배치가 실행되었다는 가정입니다. 약 5천만 행이 있던 테이블에 1천만 행 정도가 추가 되었습니다. 이러한 정보를 바탕으로 다음 단계로 넘어갑니다.

세 번째 단계인 Track down possible approach 단계와 네 번째 단계인 Execute the most likely approach 단계를 통해서 원인이 될 수 있는 사항들을 추측해 보고, 원인이 될 수 있는 각 CASE 별로 가능한 해결방법을 모색해 보도록 하겠습니다.

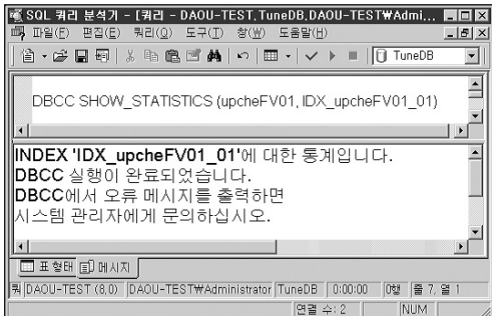
대략 다음과 같은 사항들이 원인일 가능성이 높다고 할 수 있습니다. 이제 다음의 각 CASE 를 통해서 원인을 분석하고 해결점을 도출해 보도록 하겠습니다.

1. 대량의 데이터가 로딩된 후 통계정보가 최신으로 업데이트 되지 않았거나 적절한 샘플링이 되지 않아 실행계획이 적절하지 않다.
2. 인덱스의 단편화가 심하게 발생되어 있다.
3. 현재 작업이 다른 프로세스에 의해 블로킹 당하고 있다.
4. 로그 파일의 파일 증가 설정값이 적절하지 않다.
5. 데이터 파일의 파일 증가 설정값이 적절하지 않다.
6. 현재 작업을 수행하는데 필요한 공간이 tempdb에 충분하게 확보되어 있지 않다.
7. 사용중인 디스크의 사용률이 100%에 근접해 있다.
8. 비정상적인 병렬 작업이 수행된다.
9. 바이러스등에 감염되어 서버의 상태가 불안정하다.
10. 기타 원인

CASE 1. 통계정보가 최신으로 업데이트 되지 않았거나 적절한 샘플링이 되지 않아 실행 계획이 적절하지 않은 경우

SQL Server는 데이터베이스 옵션으로 통계페이지가 생성되지 않은 경우 자동으로 통계를 생성하고, 자동으로 통계를 최신 상태로 업데이트를 해주는 옵션이 있으며 이것은 기본적으로 활성화되어 있습니다. 그렇지만 대량의 데이터가 로드된 후 즉시 다시 사용하고자 하는 경우 등에는 자동으로 업데이트가 되지 않아서 옵티마이저의 실행계획이 비효율적으로 작성되는 경우가 종종 발생합니다. 따라서, 다음과 같이 통계페이지의 최신 여부와 쿼리의 검색조건으로 지정된 컬럼의 통계페이지가 적절하게 유지 관리되고 있는지 확인해야 합니다.

DBCC SHOW_STATISTICS (테이블_이름, 인덱스_이름)



이와 같은 결과가 출력되는 해당 인덱스에 통계페이지가 생성되지 않은 경우입니다.

통계 정보가 최신으로 업데이트 되지 않았거나 인덱스만 존재하고 통계 정보가 생성되지 않은 경우 다음과 같은 명령을 실행하여 최신으로 업데이트합니다.

UPDATE STATISTICS 테이블_이름 인덱스_이름

통계를 업데이트 한 후 다시 DBCC SHOW_STATISTICS를 실행하면 다음과 같은 결과를 확인할 수 있습니다.

	Updated	Rows	Rows Sampled	Steps	Density
1	08 28 2005 12:04AM	30188024	297495	200	91.897102

	All density	Average Length	Columns
1	4.0436717E-4	6.0	upche_cd
2	3.8669759E-4	7.0	upche_cd, kl_gubn
3	3.0432135E-4	15.0	upche_cd, kl_gubn, kl_ymd

	RANGE_HI_KEY	RANGE_ROWS	EQ_ROWS	DISTINCT...	AVG...
1	000639	0.0	165.0	0	0.0
2	002267	1651.2125	165.0	0	12829
3	004372	1486.2125	330.0	0	15010
4	006740	1790.2125	26.0	0	10637
5	009316	1486.2125	46.0	0	15010
6	011450	1110.2125	165.0	0	14012
7	012211	826.21246	165.0	0	16678

①번이 최근에 통계 정보가 업데이트된 일자입니다.

①번으로 표시된 부분인 updated 항목이 데이터가 대량으로 로딩된 것이 반영된 시점인지 확인해서 통계 정보를 최신으로 유지해야 합니다.

뿐만 아니라 SQL Server는 통계의 분포도를 기록할 때 해당 값(STEP)을 200개 밖에 저장할 수 없고 통계를 수집하는 샘플링 비율에 따라 적절한 분포 정보를 유지하지 못해서 유효한 통계 정보를 생성할 수 없는 경우도 발생할 수 있습니다.

이럴 때는 옵티마이저가 유효한 실행계획을 생성할 수 있도록 샘플링 비율을 적절하게 조정해서 수동으로 통계페이지를 업데이트해야 합니다. 또한 인위적으로 분포의 스텝 값을 지정할 수 없으므로 쿼리에서 최적의 실행계획을 유도하는 옵티마이저 힌트를 사용해야만 하는 경우도 있습니다.

②번으로 표시된 부분이 나타내는 값은 일종의 버그입니다. density는 1보다 작아야 합니다. 위와 같이 잘못된 density 값이 표시될 때에는 WITH FULLSCAN 옵션으로 통계를 업데이트해야 정상적인 값이 표시됩니다.

〈통계 정보의 density 버그 참조 사이트〉

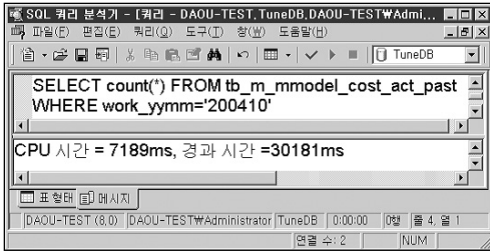
<http://support.microsoft.com/default.aspx?scid=kb;en-us;299518>

통계 정보의 업데이트는 비용 기반 옵티마이저에게 있어서 성능상 가장 중요한 요소 중 하나입니다. 대량의 배치 작업 후에는 반드시 적절한 샘플링 옵션을 지정하여 수동으로 업데이트해야 합니다.

CASE 2. 인덱스의 단편화가 심하게 발생 된 경우

인덱스의 단편화는 넓은 범위의 값을 검색할 때 성능상의 문제를 야기하게 됩니다.

인덱스는 구조상 다음 페이지 주소와 이전 페이지 주소를 저장하는 이중 링크로 구성되어 있으며 다음 페이지를 읽기 위해서는 링크의 정보를 따라 순차적으로 이동하게 됩니다. 따라서, 특정 범위의 데이터를 포함하고 있는 페이지들이 순서대로 정렬되어 있지 않고 뒤죽박죽 넓게 퍼져서 단편화가 심하게 발생되었다면 그렇지 않은 경우와 비교하면 모니터링 되는 페이지의 I/O는 동일해도 처리 속도는 느려지게 됩니다.



위와 같은 경우 특정범위 값을 카운트 하는데 약 30초가 걸렸습니다. 이것이 정상적인지를 확인하기 위해 단편화 정도를 점검해 보겠습니다. 단편화 정보를 확인하기 위해 DBCC SHOWCONTIG 명령어를 사용합니다. 테이블의 단편화를 조사해서 다음과 같은 결과를 얻었습니다.

DBCC SHOWCONTIG (테이블_이름 , 인덱스_이름)

테이블: 'tb_m_mmodel_cost_act_Past' (165575628); 인덱스 ID: 1,
 데이터베이스 ID: 10
 TABLE 수준 스캔이 수행되었습니다.

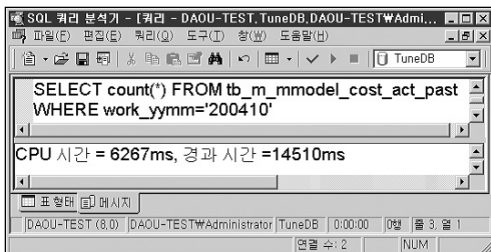
- 스캔한 페이지.....: 1888100
- 스캔한 익스텐트.....: 244450
- 전환된 익스텐트.....: 671143
- 익스텐트 당 평균 페이지 수.....: 7.7
- 스캔 밀도[최적:실제].....: 35.17% [236013:671144]
- 논리 스캔 조각화 상태: 2.35%
- 익스텐트 스캔 조각화 상태: 35.57%
- 페이지 당 사용 가능한 평균 바이트 수.....: 76.7

위에서 실행된 결과를 확인하면, 해당 테이블의 클러스터드 인덱스는 총 1,888,100 개의 페이지로 구성되었고, 페이지 링크를 따라 검색한 결과 익스텐트 전환을 671,143번 실행했습니다. 익스텐트는 8개의 페이지로 구성되어 있으므로 1,888,100개의 페이지라면 $1,888,100/8=236,013$ 개의 익스텐트로 구성되고 익스텐트 전환도 236,012번 실행해야 최적인데 익스텐트 전환을 671,143번 했으므로 마치 671,144개의 익스텐트를 가지고 있는 상태와 유사하다고 할 수 있습니다. DBCC DBREINDEX 명령어로 인덱스를 재구성하여 압축한 뒤 다시 단편화 정도를 검사한 결과가 다음과 같습니다.

TABLE 수준 스캔이 수행되었습니다.

- 스캔한 페이지.....: 1888097
- 스캔한 익스텐트.....: 240140
- 전환된 익스텐트.....: 240139
- 익스텐트 당 평균 페이지 수.....: 7.9
- 스캔 밀도[최적:실제].....: 98.28% [236013:240140]
- 논리 스캔 조각화 상태: 0.42%
- 익스텐트 스캔 조각화 상태: 0.11%
- 페이지 당 사용 가능한 평균 바이트 수.....: 76.7
- 평균 페이지 밀도(전체).....: 99.05%

기존에 비해 페이지도 몇 개 감소하고 익스텐트도 완벽하지는 않지만 거의 최적화가 되었습니다. 이와 같이 단편화를 제거한 후 다시 범위 검색 쿼리를 실행하면 다음과 같이 검색속도가 1/2로 상당히 단축되었음을 확인할 수 있습니다. 이와 같이 인덱스 재구성 작업은 단편화를 제거함으로써 검색 속도를 향상시킬 뿐만 아니라 저장공간도 효율적으로 관리할 수 있도록 해 줍니다. 물론, 인덱스를 재구성하는 동안 통계 정보는 100% 샘플링의 비율로 업데이트되었습니다.



CASE 3. 현재 작업이 다른 프로세스에 의해 블로킹 당하고 있는 경우

현재 실행중인 쿼리가 다른 프로세스가 먼저 걸어놓은 잠금 때문에 대기하고 있는 상태일 수도 있습니다. SQL Server의 기본 잠금 대기 시간은 무한대로 설정되어 있기 때문에 먼저 잠금을 걸어놓은 프로세스가 종료될 때까지 기다리기 때문에 응답이 지연되는 문제가 발생합니다. 이와 같은 상태를 점검하기 위해서는 sp_who2 시스템 저장 프로시저를 실행해서 BlkBy 컬럼의 값을 확인하거나 sp_lock 시스템 저장 프로시저 등으로 모니터링하는 것도 좋은 방법이고 SQL Server DBA 가이드에서 설명하고 있는 sp_blocker_pss80 저장 프로시저를 사용하는 것도 효율적인 방법입니다. sp_blocker_pss80 저장 프로시저는 아래의 명령을 모두 실행한 결과를 일목요연하게 모니터링할 수 있도록 해 줍니다.

- ① EXEC sp_who2 (master.dbo.sysprocess 의 정보)
- ② EXEC sp_lock (master.dbo.syslockinfo 의 정보)
- ③ DBCC SQLPERF(WAITSTATS)
- ④ DBCC INPUTBUFFER

다음의 사이트에서 스크립트 소스를 복사하여 사용하기 바랍니다.

〈sp_blocker_pss80 저장 프로시저소스 코드 참조 사이트〉
<http://support.microsoft.com/default.aspx?scid=kb;en-us;271509>

SQL 쿼리 분석기									
파일(F) 편집(E) 쿼리(Q) 도구(T) 창(W) 도움말(H)									
<div> <div> </div> <div> </div> <div> <div>master</div> </div> </div>									
EXEC sp_blocker_pss80									
<div> <div> <div>1</div> <div> <div>spid</div> <div>status</div> <div>blocked</div> <div>open_tran</div> <div>waitresource</div> </div> </div> <div> <div>1</div> <div>52</div> <div>sleeping</div> <div>0</div> <div>1</div> <div></div> </div> <div> <div>2</div> <div>53</div> <div>sleeping</div> <div>52</div> <div>0</div> <div>KEY: 6: 1977058079: 1</div> </div> </div>									
<div> <div> <div>spid</div> <div>ecid</div> <div>waittype</div> </div> <div> <div>1</div> <div>53</div> <div>0</div> <div>0x0003</div> </div> </div>									
<div> <div> <div>spid</div> </div> <div> <div>1</div> <div>52</div> </div> </div>									
<div> <div> <div>spid</div> <div>ecid</div> <div>dbid</div> <div>ObjId</div> <div>IndId</div> <div>Type</div> <div>Resource</div> <div>Mode</div> <div>St</div> </div> <div> <div>9</div> <div>52</div> <div>0</div> <div>6</div> <div>1977058079</div> <div>0</div> <div>TAB</div> <div></div> <div>IX</div> <div>GF</div> </div> <div> <div>10</div> <div>52</div> <div>0</div> <div>6</div> <div>1977058079</div> <div>1</div> <div>KEY (010086470766)</div> <div></div> <div>X</div> <div>GF</div> </div> <div> <div>11</div> <div>53</div> <div>0</div> <div>6</div> <div>1977058079</div> <div>1</div> <div>KEY (010086470766)</div> <div></div> <div>S</div> <div>W</div> </div> </div>									
<div> <div> <div>Wait Type</div> <div>Requests</div> <div>Wait Time</div> <div>Signal</div> <div>Wait T</div> </div> <div> <div>5</div> <div>LCK_M_U</div> <div>0.0</div> <div>0.0</div> <div>0.0</div> </div> <div> <div>6</div> <div>LCK_M_X</div> <div>2.0</div> <div>62.0</div> <div>0.0</div> </div> <div> <div>7</div> <div>LCK_M_IS</div> <div>0.0</div> <div>0.0</div> <div>0.0</div> </div> </div>									
<div> <div> <div>EventType</div> <div>Parameters</div> <div>EventInfo</div> </div> <div> <div>1</div> <div>Language Event 0</div> <div>update employees set lastname='김남'</div> </div> </div>									
<div> <div> <div>EventType</div> <div>Parameters</div> <div>EventInfo</div> </div> <div> <div>1</div> <div>Language Event 0</div> <div>select * from employees</div> </div> </div>									
<div> <div> <div>연결 수: 4</div> <div>NUM</div> </div> </div>									

(SP_BLOCKER_PSS80 저장 프로시저 실행 결과)

위의 그림은 sp_blocker_pss80 저장 프로시저의 실행 결과 창입니다.

- ①번 부분에서는 현재 SPID 53은 SPID 52에 의해 블로킹을 당하고 있어 대기하고 있는 상태를 보여 주고 있습니다. ②번 부분에서는 SPID 53은 인덱스 키(010086470766)에 공유 잠금(Shared Lock)을 걸기 위해 대기하고 있음을 나타내고 있습니다. ③번 부분은 DBCC SQLPERF(WAITSTATS)의 실행 결과로서 각 잠금의 요청 별로 대기 시간을 나타냅니다. ④번 부분은 DBCC INPUTBUFFER의 실행 결과이며 잠금을 걸고 있거나 블로킹을 당하고 있는 SPID가 실행중인 쿼리 문을 확인해 줍니다.

CASE 4. log file의 설정 값이 적절하지 않은 경우

트랜잭션 로그 파일은 트랜잭션을 시작하기 위해서 그리고, 트랜잭션 로그를 기록하여 트랜잭션의 ACID 속성을 보장하기 위해서 반드시 충분한 공간을 확보하고 있어야 합니다. 트랜잭션 로그가 더 이상 기록될 공간이 없게 되면 아래와 같은 오류 메시지가 반환되고 더 이상의 트랜잭션의 실행은 불가능하게 됩니다.

오류: 9002, 심각도: 17, 상태: 2

'%.s' 데이터베이스의 로그 파일이 꽉 찼습니다.

이러한 상황이 발생하지 않도록 DBA는 로그 파일을 적절한 크기로 관리해야 합니다. 뿐만 아니라 로그 파일의 자동 증가 설정을 통해 미처 DBA가 예상하지 못한 경우에 대비해야 합니다. 자동증가 설정이 부적절하면 쿼리의 응답속도가 갑자기 느려지는 현상이 발생할 수 있습니다.

처음 가정에서 대량의 배치가 실행된 직후 라는 가정을 하였습니다. 대량의 변경작업이 수행 되었으므로 당연히 많은 로그 파일 공간을 사용했을 것입니다. 먼저 다음 구문으로 로그 파일의 공간 사용 정보를 확인합니다.

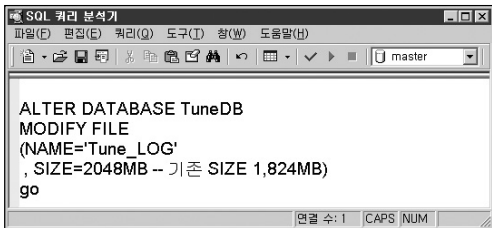
DBCC SQLPERF(LOGSPACE)

	Database Name	Log Size (MB)	Log Space Used (%)	Status
3	model	0.4921875	65.674606	0
4	msdb	2.2421875	30.008711	0
5	pubs	0.7421875	42.763157	0
6	Northwind	0.9921875	44.1437	0
7	TuneDB	128.86719	0.33343437	0

만약, 로그 파일의 사용률이 100%에 달했다면 파일증가 속성에 지정된 임계값의 크기로 확장하는 시간 동안 DML이나 DDL과 같은 쿼리의 실행은 대기 상태에 있게 됩니다. 이럴 때 확장 크기가 너무 크다면 확장하는 시간 동안, 너무 작다면 너무 빈번하게 확장을 시도하게 되어 두 경우 모두 쿼리가 대기하게 되므로 응답속도가 느려지는 결과를 초래하게 됩니다.

실제로 DBA의 실수로 로그 파일 증가 속성을 1MB로 지정해놓고 대형 인덱스를 재구성하여 약 1시간 정도면 종료 되어야 하는 작업이 10시간이 넘게 수행되는 경우도 목격하였고, 대형의 배치작업을 수행하다가 트랜잭션 로그 파일이 있는 디스크에 충분한 공간이 있음에도 불구하고 로그가 가득 찼다는 9002 오류를 유발하고 전체 작업이 롤백되는 현상도 목격하였습니다. 이와 같이 로그 파일의 사용률과 로그 파일의 확장 속성의 지정 여부도 쿼리의 응답속도에 영향을 미칠 수 있으므로 DBA는 항상 로그 파일의 사용률을 모니터링하고 미리 미리 충분한 로그 파일의 공간을 확보해야 합니다.

참고로 다음과 같은 구문으로 Wide-Ultra SCSI3/10,000 RPM의 RAID 1 환경에서 약 200MB를 확장하는데 약 14초 정도가 소요되었습니다.



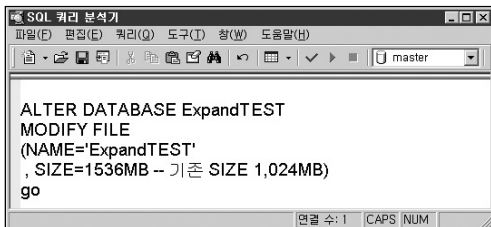
특히나 대형 테이블의 인덱스 생성 / 재구성 작업이나 대량의 배치작업을 위해서는 대형의 로그 파일이 필요하므로 이와 관련해서는 대용량 작업에서 발생할 수 있는 문제 부분에서 다루도록 하겠습니다.

〈트랜잭션 로그 참조 사이트〉

<http://support.microsoft.com/default.aspx?scid=kb;ko;317375>

CASE 5. data file의 설정 값이 적절하지 않은 경우

CASE 4의 경우를 이해하였다면 굳이 추가적인 설명이 필요하지는 않습니다. 단순 SELECT의 경우라면 데이터 파일에서 추가 공간을 할당 받을 필요가 없지만 DML 작업인 경우에는 데이터 파일의 공간을 필요로 하므로 파일의 사용률과 증가 속성이 중요한 요소가 됩니다. 참고로 다음과 같은 구문을 통해서 로그 파일을 테스트한 동일한 환경에서 500MB의 데이터 파일을 확장하는 테스트를 한 경우 약 29초의 시간이 소요되었습니다. 따라서, DBA는 자동 증가 옵션의 설정에 의지해서 데이터 파일과 로그 파일을 방치하게 되면 SQL Server의 응답 속도가 느려지는 문제가 발생하게 됩니다.



CASE 6. 현재 작업에 충분한 tempdb의 공간이 없는 경우

tempdb는 정렬(sort), 그룹핑(group by), 커서 사용, 임시테이블 및 테이블 변수 사용, JOIN 작업, SORT_IN_TEMPDB 옵션을 사용한 인덱스 생성, 데이터베이스의 복구 작업 등에서 사용됩니다. 따라서, 이와 같은 작업을 위해서는 반드시 충분한 tempdb의 공간이 확보되어야 하며, 다중의 사용자가 tempdb를 사용한다면 tempdb의 위치나 파일의 개수, 사이즈 등을 충분히 고려하여 구성하여야 합니다. 즉, DML 작업이 아니더라도 쿼리는 tempdb의 공간을 필요로 할 수 있기 때문에 DBA는 항상 tempdb의 상황을 모니터링하고 관리해야 합니다. tempdb의 공간 사용 정보는 sp_tempdbspace 시스템 저장 프로시저를 통해서 확인합니다.

	database_name	database_size	spaceused
1	tempdb	8.500000	.539062

tempdb의 I/O 경합을 줄이기 위해서는 SMP 서버의 경우 해당 CPU 개수만큼의 데이터 파일을 동일한 사이즈로 생성할 것을 권장합니다. 그럴 경우 각 CPU를 점유하고 있는 스레드들이 각각 다른 tempdb 데이터 파일을 액세스하므로 I/O의 경합을 줄일 수 있습니다. 뿐만 아니라 추적 플래그 T1118을 사용하면 tempdb가 익스텐트를 할당할 때 다중의 세그먼트가 사용하는 혼합(Mixed) 익스텐트를 할당하지 않고 유니폼(Uniform) 익스텐트만을 할당하므로 추가적으로 IO의 경합을 감소시킬 수 있습니다. 단, T1118 추적 플래그는 SQL Server 7.0에서는 사용을 삼가합니다.

〈tempdb 성능 관련 참조 사이트〉

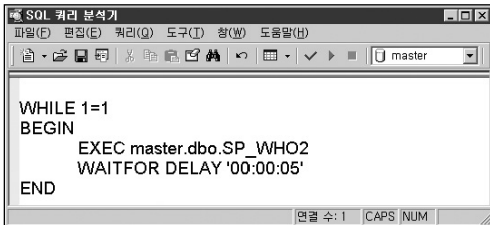
<http://support.microsoft.com/kb/328551>

CASE 7. 사용중인 디스크의 사용률이 100%에 근접한 경우

하드디스크의 경우 사용률이 전체 공간대비 약 85% 대에 이르면 불필요한 큐잉(queueing)을 유발할 수 있습니다. 따라서, 데이터의 증가에 대한 사이징을 미리 하셔서 매년 사업계획을 작성할 때 반드시 예산을 확보해야 합니다.

CASE 8. 비정상적인 parallel 작업이 수행 되는 경우

앞에서 다룬 SQL Server와 연결이 되지 않는 경우의 CASE 4 에서 다루었던 부분이기도 합니다. 마이크로소프트의 기술 문서에서 확인하면 다양한 경우에 스레드가 과도하게 할당되어 느려지거나 추가적인 스레드 할당을 하지 못하고 17884 오류를 발생하는 경우가 있습니다. 필자의 판단이지만 Hyper-Threading을 지원하는 SMP 서버에서 통계 페이지를 업데이트 하는 과정이나 대량의 DML작업에서도 비정상적으로 스레드의 개수를 늘리는 경우가 모니터링 되곤 합니다. 이런 경우에는 스레드의 개수가 “max worker threads” 옵션의 현재 구성 값에 도달해서 장시간 대기하다가 교착상태로 종결되는 경우도 있고, 교착상태를 발생시키지 않고 응답속도만 느려지는 경우도 있습니다. 비정상적으로 스레드의 개수가 늘어나는 현상을 모니터링 하기 위해서는 “max worker threads”를 충분한 개수로 조정해야 가능합니다. 사용자 connection의 정보는 성능 모니터의 SQL Server : General Statistics : User connections 카운터를 통해서 모니터링 할 수 있고, 활성화된 스레드의 정보는 다음과 같은 쿼리로 증가하는 상황을 확인할 수 있습니다.



〈5초마다 sp_who2 저장 프로시저를 실행하는 쿼리〉

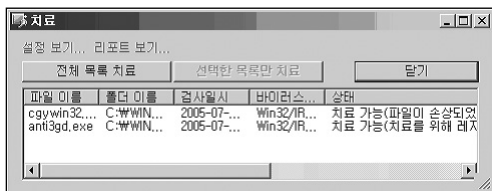
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	RUNNABLE	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	DEFWAKEUP	sa	DAOU-TESTSVR	.	Hugug
89	RUNNABLE	sa	DAOU-TESTSVR	.	Hugug
89	RUNNABLE	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	RUNNABLE	sa	DAOU-TESTSVR	.	Hugug
89	RUNNABLE	sa	DAOU-TESTSVR	.	Hugug
89	DEFWAKEUP	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
89	sleeping	sa	DAOU-TESTSVR	.	Hugug
		sa			Hugug
89	RUNNABLE		DAOU-TESTSVR	.	
89	sleeping	sa	DAOU-TESTSVR	.	Hugug

〈대량의 INSERT 작업 중 과도하게 THREAD를 확장해가는 초기 단계〉

좌측의 표는 8개의 CPU를 가진 서버가 INSERT 작업을 실행하면서 과도하게 스레드를 할당하는 초기 단계를 sp_who2 시스템 저장 프로시저를 통해서 모니터링한 결과입니다. 이와 같은 경우 마이크로소프트가 제공하는 핫픽스를 설치하여 해결되는지 여부를 확인합니다. 핫픽스로 해결되지 않을 경우 임시적이기는 하지만 “max degree of parallelism” 서버 옵션을 물리적인 CPU 의 개수 -1 정도의 값을 설정하는 방법도 고려합니다. 상당기간이 흐른 현재까지 동일한 문제가 재발하고 있지 않습니다. 쿼리에서 OPTION (MAXDOP n) 절을 사용하는 방법도 확인해보도록 합니다.

CASE 9. 바이러스 등에 감염되어 서버의 상태가 불안정한 경우

시스템이 바이러스에 감염된 경우에는 시스템 자체에 과부하가 걸려서 이상 현상을 나타내기도 하며 SQL Server 웹 바이러스 같은 바이러스 공격을 받는 경우에 쿼리의 응답속도가 느려지는 현상이 생기기도 합니다. 따라서, 윈도우즈 서버나, SQL Server의 보안 패치는 테스트 후 즉시, 서비스 팩도 테스트 실시 후 빠른 시간 내에 반영하도록 권장합니다. 뿐만 아니라 서버용 바이러스 백신을 설치하고 정기적으로 검사함으로써 서버를 더욱 안전하고 정상적인 상태로 유지할 수 있습니다. 단, SQL Server에 바이러스 백신을 설치하고 검사하는 작업의 결과가 예상하지 못한 오류를 발생하거나 SQL Server에 치명적인 결과를 초래할 수도 있습니다.



〈바이러스에 감염된 경우〉

따라서, 바이러스 백신 프로그램이 SQL Server에 적합한 것인지를 다음의 마이크로소프트 사이트에서 확인하고 해당 백신 판매업체와 충분히 상담하여야 합니다.

<http://www.microsoft.com/security/partners/antivirus.asp>

또한, SQL Server에서 바이러스 감염 여부를 검사할 때 발생할 수 있는 문제에 대해서도 사전 검토가 필요합니다. 다음의 사이트에서 SQL Server에서 바이러스를 검사하는 경우 발생할 수 있는 문제와 주의사항에 대해 설명하고 있습니다.

<http://support.microsoft.com/?kbid=309422>

시스템이 바이러스에 감염된 경우 외에도 설치된 응용 프로그램이나 하드웨어 드라이버 등의 문제가 운영체제를 불안정하게 하거나 많은 리소스를 사용하여 시스템 전체를 느리게 할 수도 있습니다. 이런 경우 문제가 되는 응용 프로그램이나 하드웨어 드라이버 등을 중심으로 점검하고, 정확한 진단과 해결이 어려운 경우에는 다음의 사이트를 참조하여 메모리 덤프를 생성해서 마이크로소프트 기술지원센터에 문의합니다.

〈메모리 덤프 생성 참조 사이트〉

<http://support.microsoft.com/default.aspx?scid=kb;ko;244139>

쿼리의 응답속도가 갑자기 느려지는 아홉 가지 경우를 살펴보았습니다. 지금까지 살펴본 이유 외에도 여러 가지가 쿼리의 응답속도가 갑자기 느려지는 원인이 될 수도 있습니다. 위에서 나열한 여러 가지 경우들 중에서 원인과 해결방법을 찾았다라도 반드시 나머지의 경우도 점검을 하도록 합니다. 여러 가지 문제가 복합적으로 작용해 원인이 될 수도 있고 지금은 장애의 원인으로 나타나지는 않았지만 어느 정도 지금의 상태가 지속된다면 또 다시 갑자기 쿼리의 응답속도가 느려지는 상황을 유발할 가능성이 잠재되어 있을 수도 있기 때문입니다.

다섯 번째 단계인 Check for Success를 수행하셨다면 마지막으로 Tie up loose ends 단계를 실시함으로써 확실한 마무리를 해야 합니다. 다시 한번 미비한 점은 없는지 살펴 보고 인터넷으로 관련 자료를 수집해 여러분이 처리한 해결방법과 비교해 보는 것도 좋은 방법입니다. 끝으로 조치한 사항에 대해서 다음과 같이 오류 발생 일지를 작성하고 일지의 “주의 및 확인사항” 항목에 기록된 대로 수시로 관련 자료의 유무를 확인한다면 아주 깔끔한 마무리라 할 수 있습니다.

오류 발생 일지

분류 번호	2005-0088	운영 팀	DB 팀
발생 일자	2005/07/02	운영 자	김 호
처리 일자	2005/07/02	처리 담당자	이 찬 일
장애 구분	H/W() OS () DBMS(O) Network () 기타서버() 응용 프로그램()		
증 상			

- ① 일일 배치의 2단계 작업 usp_loan_job1 실행의 응답속도가 갑자기 느려짐
- ② 나머지 작업은 이상 없음

오류 발생 시 환경

- ① 일일 배치의 1단계 작업에서 평상시 약 2배의 데이터가 로드됨
- ② 상기 사항 외에 운영 서버에 패치 등 프로그램 설치 작업 없었음
- ③ usp_loan_job1 실행 모니터링 결과 비정상적으로 약 160여 개의 스레드가 할당됨

조치 사항

- ① 마이크로소프트 기술 문서를 검색하여 동일한 경우의 진단 및 조치와 관련된 사항 확인
- ② 서버 구성 옵션 "max degree of parallelism" 의 값을 0에서 5로 조정

주의 및 확인 사항

- ① 서버 구성 옵션 "max degree of parallelism" 의 값을 5로 조정하여 다른 배치작업 또는 백업 작업등 부하가 큰 작업들의 성능 점검이 필요함
- ② OPTION 절의 MAXDOP 힌트 사용도 확인이 필요함
- ③ 해당 현상과 동일한 기술 문서가 있는 지 수시로 확인이 필요함

기 타 사 항

참고 자료 : 없음

노련한 Troubleshooter 가 되려면?

DETECT 방법에 따라 두 가지의 문제에 대해 접근 방법과 해결 방법에 대해서 접근해 보았습니다. 여러분은 진행되는 과정에서 본인이 어디까지 스스로 접근할 수 있었는지 판단해 보셨습니까? 생뚱 맞지만 예를 하나 들어 보겠습니다.

여러분은 살인사건 현장에 서 있습니다. 살인사건을 해결하려는 명탐정입니다. 어렸을 때 읽었던 추리 소설 속의 주인공인 에르퀼르 뵈와로나 설록 홈즈라고 상상하셔도 좋고, 수사반장 최불암씨 라고 생각해도 좋습니다. 살인 사건이 발생한 현장에서 여러분이 가장 먼저 해야 할 일은 무엇입니까?

아마 살인사건의 현장 보존부터 해야 할 것 같습니다. 일반인들이 살인현장에 접근해서 현장을 훼손하는 일을 막아야 합니다. 현장의 모든 것이 단서가 될 수 있기 때문입니다.

이어서, 주변에 단서가 될 수 있다고 의심이 가는 물건들을 수집하고 또한 현장사진도 찍어 두어야 합니다. 감식반이 해야 할 일입니다. 그리고, 검시 의의를 통해 어떤 흉기로 어떻게 상처를 입어 몇 시경에 살해당했는지 확인해야 됩니다. 이제 수집된 물품들에 대한 분석에 들어갑니다. 담배꽂초나 휴지 같은 것은 국립 과학 수사 연구소에 보내도록 합니다. 이어서, 수집된 물건들과 살해 현장, 그리고 주변의 목격자들을 조사합니다.

그리고, 살인의 동기를 조사해봐야 할겁니다. 원한에 의한 살인인지 단순 강도 살인인지, 아니면 또 다른 동기가 있는지 등을 철저히 조사해야 합니다. 원한에 의한 살인이라면 피해자와 원한관계에 있는 사람들을, 강도 살인이라면 유사한 범행수법으로 범행을 저지른 적이 있는 전과자들을 조사해 보아야 합니다. 아니면 단순 강도를 가장한 원한에 의한 살인인지도 의심해 봐야 합니다. 용의자들을 찾아내면 그들의 용의 점을 면밀히 분석해야 합니다. 그리고, 그들의 알리바이를 조사하는 것도 빼놓으면 안 됩니다. 이제 여러분의 추리력으로 범인과 확실한 증거를 찾았다면 범인을 검거하러 가야 합니다. 때로는 며칠씩 잠복 근무를 할 수도 있습니다. 범인이 숨어 있을 만한 곳도 추리해내야 합니다. 탐문이 필요하기도 합니다. 마침내 범인을 체포했다면 조서를 꾸밀 차례입니다. 6하 원칙에 의거하여 작성해야 하며 증거품들을 철저하게 챙기는 것도 잊으면 안됩니다. 이것으로 마무리 된 것은 아닙니다. 여러분은 엉뚱한 사람을 범인으로 지목했을 수도 있고 어딘가에 공범이 있는지도 모른다는 사실을 잊으면 안 됩니다.

이것은 우리가 SQL Server에서 발생한 오류나 성능상의 문제를 해결하는 DETECT 방식과 흡사합니다. 왜 필자가 DETECT 방법론을 설명했는지 이해가 되었을 것입니다.

이제는 노련한 Troubleshooter 가 되기 위해서 연마해야 할 사항들은 무엇이 있는지 살펴볼 것입니다.

명탐정의 수사를 돕기 위해 땀 흘리며 현장 보존을 담당하는 말단 경찰관부터 감식반, 검시의, 국립 과학수사 연구소의 연구원, 그리고 보고서를 작성하는 부하 수사관들을 대신해 지금 오류가 발생한 현장 주변엔 어떤 사람들이 있습니까?

하드웨어 전문가, 네트워크 전문가, 보안 전문가, 운영체제 전문가, SQL Server 전문가, 클라이언트 툴 전문가, 다른 응용 프로그램 서버 전문가, 이 가운데에서 여러분 혼자서 할 수 있는 역할은 어떤 것이 있습니까?

여러분은 SQL Server가 설치되어 있는 하드웨어 플랫폼과 스토리지에 대해서, 윈도우즈라는 운영체제, 뿐만 아니라 SQL Server와 클라이언트 툴에 대해서 어느 정도의 지식을 가지고 계십니까?

앞에서 처리해본 두 가지 경우를 처리하기 위해서 여러분의 주위에 몇 명의 도움이 필요합니까? 어떤 부분에 대해 얼마만큼 공부하면 여러분 주위에서 여러분을 도울 조력자의 숫자를 줄일 수 있습니까?

모든 분야에 전문가가 될 수는 없겠지만 발생한 오류에 대해서 이것이 하드웨어의 문제인지 운영체제의 문제인지, 아니면 SQL Server의 문제인지는 구별해 낼 수 있는 정도의 지식은 있어야 할겁니다. 뿐만 아니라, 여러분의 서버를 유지보수 하고 있는 업체 직원이 진단 프로그램을 돌려본 뒤 “하드웨어 문제는 아닙니다.” 라고 얘기하면 여러분은 “아! 그래요? “라고 하면서 하드웨어를 문제의 용의 선상에서 제외시키는 실수를 범해서도 안됩니다.

노련한 Troubleshooter가 되기 위해서는 서버를 구성하고 있는 전반적인 환경에 대한 기본적인 지식과 SQL Server에 대한 충분한 지식, 그리고 그와 같은 지식을 가지고 오류의 원인과 해결책을 찾아낼 수 있는 논리적인 추리력, 뿐만 아니라 그동안 해결했던 경험을 바탕으로 유사한 문제에 대해 적용할 수 있는 응용력과 창의력도 배양해야 합니다.

사건현장을 따라 다니던 헤이즈팅스 대위나 닥터 와트슨처럼 오류 발생 일지를 꼼꼼하게 기록해 두는 정도의 센스도 필수 사항입니다.

끝으로 다른 사람이 정리해 둔 기술 문서를 빠르고 효율적으로 검색하고 참고할 수 있는 인터넷 정보 검색사의 능력까지 보유한다면 더할 나위가 없겠습니다.

오류가 발견되면 그때마다 DETECT 방법에 따라 문제해결에 접근하고 그때그때 필요한 지식을 습득하다 보면 여러분의 지식과 경험은 피라미드처럼 견고히 쌓여 아주 노련한 Troubleshooter 가 될 것이라고 믿습니다.

손상된 사용자 데이터베이스 복구

데이터베이스 시스템 관리에 있어서 재난에 대비한 복구 전략의 수립은 필수적인 요소입니다. 그 중에서도 특히 중요한 요소는 백업과 복구 전략이라고 할 수 있습니다. 백업 및 복구 전략을 수립하는 목적은, 데이터 손실은 최소화하면서 가능한 한 빠른 시간 내에 업무를 정상화하는 것입니다. 즉, 복구는 목적이고 백업은 수단의 일부라는 사실을 유념해야 합니다. 간혹, 백업은 열심히 하면서 복구 훈련은 한번도 하지 않는 경우도 있습니다.

재난에 대비한 복구 전략을 수립하는 데 있어서, 가장 먼저 해야 할 일은 재난에 대한 정의를 올바르게 하는 일입니다. 재난의 정의에 빈틈이 생기면 그에 대한 대비 전략에도 빈틈이 생길 수 밖에 없기 때문입니다. 따라서, 어느 날 갑자기 대포동 미사일이나 ICBM이 테헤란로에 떨어지는 상황에서부터 디스크의 손상, DBA나 개발자의 실수에 이르기까지 모든 상황을 염두에 두어야 합니다. 서버룸에 이산화탄소 분사 장치로 화재 상황에 대비는 했지만 운영인력을 위해서 산소마스크는 준비해 두지 않는 곳이 많이 있습니다. 화재 발생시 시스템은 무사한데 운영인력이 없어서 장애가 발생할 수도 있는 상황인 것입니다.

SQL Server로 돌아 옵니다. SQL Server DBA는 SQL Server 백업과 복구의 기능과 특성을 버전 별로 정확히 알고 있어야 합니다. 그래야만 운영중인 SQL Server 버전에 따라서 올바른 복구 전략을 수립할 수 있습니다. 본 가이드에서는 SQL Server의 백업과 복구 기능에 대해서는 다루지 않습니다. SQL Server DBA 가이드(전현경 저)의 백업과 복구 부분을 참조하여 올바르게 숙지하기 바랍니다.

〈SQL Server DBA 가이드(전현경 저)〉

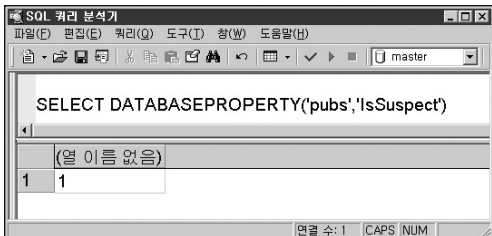
http://www.microsoft.com/korea/technet/sql/tuning_guide_developer03.asp

데이터베이스를 운영하는 중에 데이터베이스가 주의 대상(SUSPECT) 모드가 되는 경우가 있습니다. 데이터베이스가 손상을 입은 경우입니다. 데이터 페이지가 손상을 입었을 수도 있고 외장형 스토리지와의 연결이 정상적이지 않을 수도 있으며 로그 파일이 손상되었을 수도 있습니다. 데이터베이스가 주의대상 모드인 경우는 다음과 같이 확인할 수 있습니다.

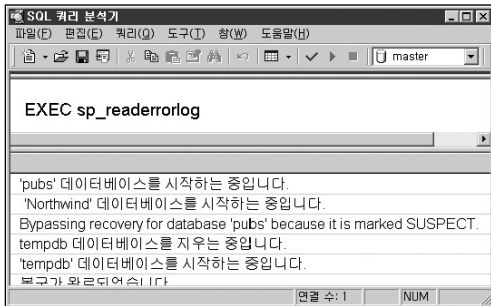
1) 엔터프라이즈 관리자에서 확인하는 경우



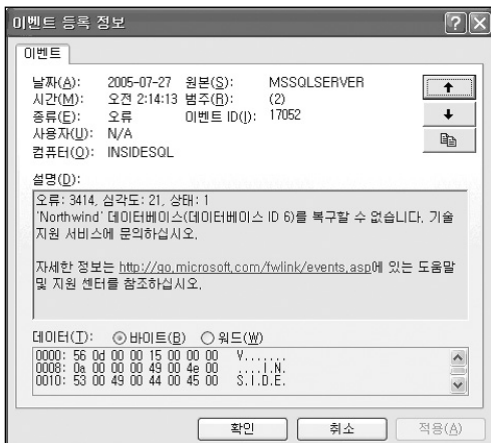
2) 시스템 함수로 확인하는 경우



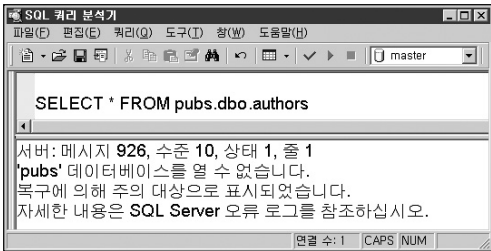
3) SQL Server 오류 로그에서 확인하는 경우



4) 이벤트 로그에서 확인하는 경우



5) 데이터베이스 사용 시점에 확인하는 경우



■ 데이터 파일 손상 시 복구 방법

사용자 데이터베이스가 주의 대상 모드로 변경된 경우에는 원인을 찾기 위해서 윈도우즈 이벤트 로그와 SQL Server 오류 로그를 면밀히 검토해야 합니다. SQL Server 오류 로그에서 다음과 같은 오류 메시지를 발견한 경우에 데이터베이스를 복구하는 방법을 살펴보겠습니다.

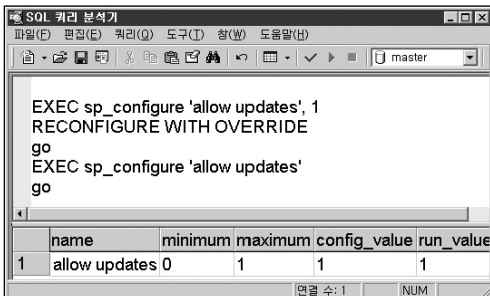


먼저 증상을 확인하기 위해서 해당 메시지와 관련된 마이크로소프트 기술문서를 검색합니다. 아래 링크의 KB828337 기술 문서를 확인하면 DBCC CHECKDB를 실행해서 데이터베이스의 오류나 일관성의 오류 등이 발생했는지 여부를 점검하도록 권장하고 있습니다.

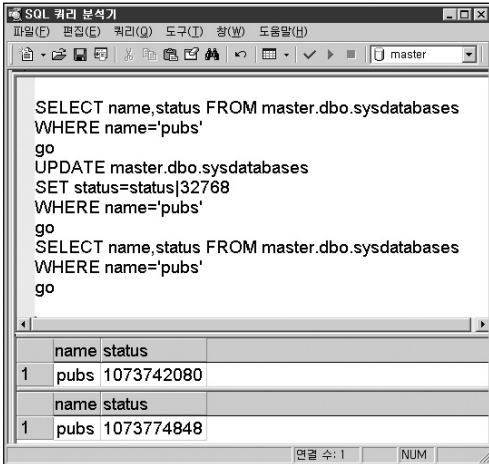
<http://support.microsoft.com/default.aspx?scid=kb;en-us;828337>

현재는 데이터베이스가 주의 대상 모드이므로 데이터베이스에 접근이 불가능한 상태라 해당 데이터베이스에서 아무런 작업도 할 수 없습니다. 이런 경우에는 먼저 응급 복구 (EMERGENCY) 모드로 변경해서 작업하도록 합니다. 모든 데이터베이스의 상태 정보는 master.dbo.sysdatabases 시스템 테이블의 status 컬럼과 status2 컬럼에 저장되어 있습니다. ALTER DATABASE 구문이나 sp_dboption 시스템 저장 프로시저를 사용하여 데이터베이스 구성 옵션을 변경하는 경우에도 바로 이 값이 변경됩니다. SQL Server 2005에서는 ALTER DATABASE ~ SET EMERGENCY 구문을 제공하고 있으나 SQL Server 2000에서는 직접 시스템 테이블을 수정해야 합니다. 시스템 테이블을 직접 수정할 수 있는 경우는 SQL Server 서비스를 /m 옵션으로 시작하는 단일 사용자 모드인 경우와 SQL Server의 구성 옵션인 "allow updates" 의 설정값을 1로 변경한 경우입니다.

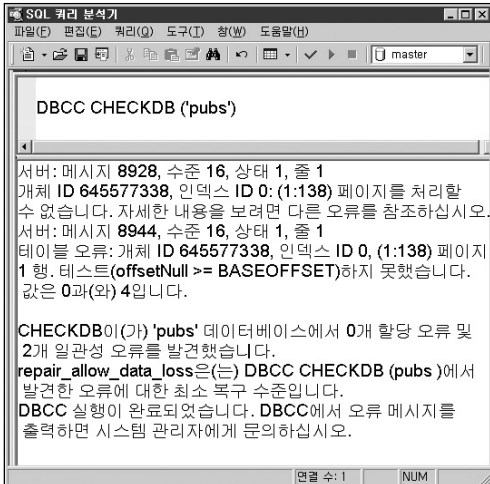
- 1) 사용자 데이터베이스 점검을 위하여 서비스를 단일 사용자 모드로 다시 시작하는 것은 비효율적이므로 다음과 같이 서버의 구성 옵션을 변경하고 점검 작업을 수행합니다.



- 2) master.dbo.sysdatabases 시스템 테이블의 status 컬럼을 응급 복구 모드로 변경합니다. 응급 복구 모드로의 변경은 기존 status 컬럼 값에 32768을 비트 OR 연산하여 변경합니다. 복구가 완료되면 그 값에 (~32768)을 비트 & 연산을 수행함으로써 원래 상태의 값으로 다시 변경할 수 있습니다. DBA는 데이터베이스의 설정 상태를 유지하기 위해 데이터베이스 별로 평상시 이 값의 정보를 관리자 노트에 기록해 놓아야 합니다.



- 3) 이제 응급 복구 모드로 변경되어 임시적으로 데이터베이스에 접근이 가능하므로 DBCC CHECKDB를 실행하여 원인을 분석합니다.



- 4) DBCC CHECKDB를 실행한 결과 2개의 일관성 오류가 발견되었다는 리포트를 받았습다. 이런 경우 SQL Server 2005는 해당 데이터베이스의 백업으로부터 손상된 페이지만 복구하는 기능을 제공합니다. 반면에 SQL Server 2000은 이와 같은 기능을 제공하지 않고 있습니다. 따라서, 일부 데이터페이지가 손상된 경우라도 해당 데이터베이스의 백업이나 데이터 파일의 백업을 이용해서 복구해야 합니다. 그러나 가용한 백업이 없다는 가정으로 최대한 데이터를 복구하기 위해서 좀 더 정밀하게 분석해 보겠습니다.

5) DBCC CHECKDB의 실행 결과를 자세히 살펴 보면 mytab이란 테이블은 총 99개 페이지에 297개 행이 점검 가능하였고 (1:138) 페이지의 하나의 행이 손상을 입어서 해당 페이지를 처리할 수 없다는 것을 알 수 있습니다. 해당 테이블과 관련된 추가적인 정보를 확인하기 위해서 아래 그림과 같이 sysindexes 시스템 테이블을 통해서 mytab의 테이블의 페이지 할당 정보를 확인하고 DBCC CHECKDB 처리 결과와 비교합니다.

	indid	dpages	reserved	used	rowmodctr
1	0	100	107	103	300
2	2	1	2	2	0

sysindexes 시스템 테이블과 DBCC CHECKDB의 실행 결과를 분석한 결과는 다음과 같습니다.

- ① mytab 테이블은 HEAP으로 구성되어 있고 Non-Clustered 인덱스가 1개 존재합니다. (indid 컬럼 값이 0=HEAP, 2~250 =Non-Clustered INDEX)
- ② 데이터는 총 100개 페이지를 사용하며 데이터는 300행입니다.
(DBCC CHECKDB의 '99개 페이지 297개 행이 있다'는 결과 메시지와 비교합니다.)
- ③ 사용중인 페이지는 데이터 100개 페이지, 해당 테이블의 IAM 1개 페이지 인덱스 1개 페이지, 인덱스 IAM 1개 페이지를 포함하여 총 103개의 페이지입니다.
- ④ 테이블에서 사용하고 있는 데이터 페이지 가운데 0x8A (138) 페이지가 첫 번째 페이지이며 이 가운데 1개 행이 손상을 입었습니다.

필요한 정보를 정리하면 ② 번에서 확인한 정보와 앞에서 실행한 DBCC CHECKDB의 실행 결과와 비교하여 mytab 테이블은 총 300행으로 100개의 페이지를 사용하고 있으며 1번 파일의 138번 페이지(1:138)에 3개의 행 가운데 하나의 행이 손상되었음을 알게 되었습니다. 손상된 데이터베이스를 복구하기 위해서 DBCC CHECKDB 또는 DBCC CHECKTABLE을 REPAIR_ALLOW_DATA_LOSS 복구 옵션과 함께 실행하게 되면 다음 구문에서 설명하는 바와 같이 손상된 1개 페이지의 할당이 취소되므로 3개 행 전체를 잃게 됩니다.

DBCC CHECKDB | CHECKTABLE

('테이블 이름' , REPAIR_FAST

| REPAIR_REBUILD

| REPAIR_ALLOW_DATA_LOSS)

REPAIR_FAST : 클러스터드 인덱스의 별도 키를 복구하는 것과 같이 사소 하고 시간이 소요되지 않는 복구 작업을 수행합니다. 이러한 복구는 데이터 손실의 위험 없이 빨리 실행할 수 있습니다.

REPAIR_REBUILD :

REPAIR_FAST에서 이루어지는 모든 복구 작업을 수행하고 인덱스 다시 작성과 같이 시간이 소요되는 복구를 포함합니다. 이러한 복구는 데이터 손실의 위험 없이 실행할 수 있습니다.

REPAIR_ALLOW_DATA_LOSS :

REPAIR_REBUILD에서 수행하는 모든 복구 작업을 수행하며 할당 오류, 구조적 행 오류나 페이지 오류, 손상된 텍스트 개체 삭제 를 수정하기 위한 행과 페이지의 할당 및 할당 취소가 포함 됩니다. 이러한 복구를 할 경우 일부 데이터가 손실될 수 있습니다. 복구 작업은 사용자가 변경 사항을 롤백할 수 있도록 사용자 트랜잭션 내에서 수행합니다. 복구가 롤백되어도 데이터베이스에는 오류가 그대로 포함되므로 백업에서 데이터베이스 를 복원해야 합니다. 제공된 복구 수준 때문에 오류 복구를 생략한 경우 해당 복구에 종속적인 모든 복구도 생략됩니다. 복구를 완료한 다음에 데이터베이스를 백업합니다.

따라서, 손상되지 않은 나머지 2개의 행을 복구하기 위해서 손상된 행을 추적합니다.
다음과 같이 EXEC sp_help 'mytab' 을 실행해서 테이블의 정보를 확인해 보았습니다.

	Name	Owner	Type	Created_datetime
1	mytab	dbo	user table	2004-09-26 17:11:43.977

	Column_name	Type	Computed	Length	Prec	Scale	Null
1	col1	int	no	4	10	0	no
2	col2	char	no	2000			no
3	col3	varchar	no	5			yes

	Identity	Seed	Increment	Not For Replication
1	col1	1	1	0

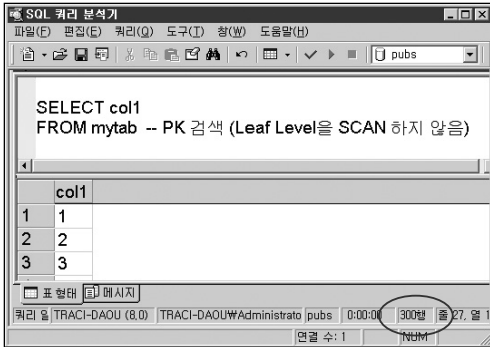
	RowGuidCol

	Data_located_on_filegroup
1	PRIMARY

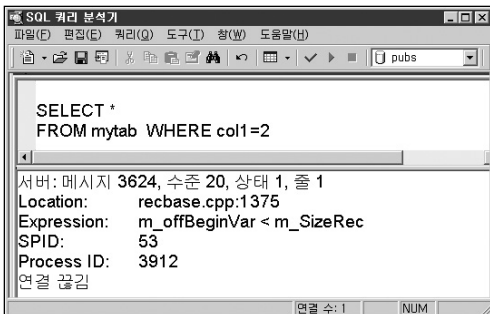
	index_name	index_description
1	PK__mytab__276EDEB3	nonclustered, unique, prima...

연결 수: 1 NUM

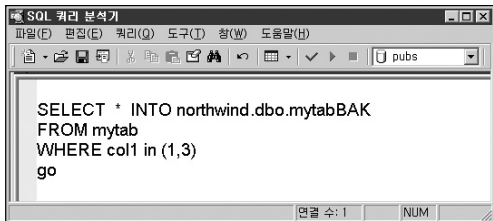
실행 결과에서 mytab 테이블은 col1, col2, col3 3개의 컬럼으로 구성되어 있고 col1 컬럼에 PK가 만들어져 있으며 IDENTITY(1,1) 속성이 있다는 정보도 확인했습니다. HEAP은 데이터가 INSERT 되는 대로 저장되므로 첫 번째 페이지에 존재할 수 있는 행의 IDENTITY 값의 빈틈이 없는 경우 col1의 값이 1~3일 가능성이 높음을 추리해낼 수 있습니다.
빈틈이 존재하는지 여부를 확인하기 위해서 먼저 col1 컬럼에 만들어져 있는 손상되지 않은 인덱스를 통해서 col1 컬럼의 값을 확인합니다. 아래와 같이 인덱스를 통해서 확인한 결과 col1 컬럼의 IDENTITY는 빈틈이 없이 1~300까지 존재함을 확인했습니다.



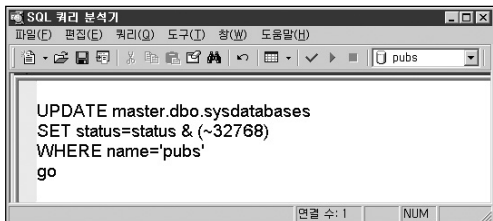
이제 첫 번째 페이지에 있을 것으로 추정되는 col1 컬럼의 값이 1~3인 행을 제외하고 검색합니다. 에러가 발생하지 않고 297행이 반환되었습니다. 그렇다면 위의 쿼리에 WHERE 조건절을 추가하고 col1=1, col1=2, col1=3을 차례로 입력하고 실행해서 손상된 행이 어느 행인지 확인합니다.



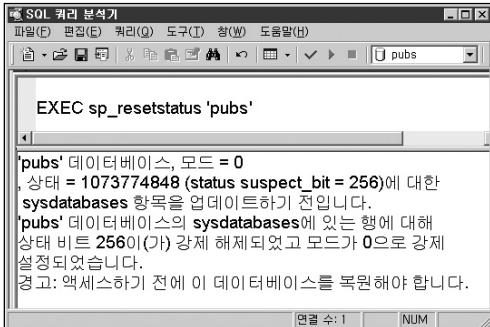
위와 같이 col1 컬럼의 값이 2일 때만 오류가 발생했으므로 손상된 행이 col1 컬럼의 값이 2임을 발견하였습니다. 이제 손상되지 않은 col1=1,3인 행을 SELECT * INTO 구문으로 다른 데이터베이스로 백업합니다.



이어서 DBCC CHECKDB 또는 DBCC CHECKTABLE을 REPAIR_ALLOW_DATA_LOSS 복구옵션과 함께 실행하기 위해서 다음과 같이 설정했던 응급 복구 모드를 해제하고, sp_resetstatus 시스템 저장 프로시저를 실행해서 주의 대상 모드를 해제합니다. master 데이터베이스의 sysdatabases 시스템 테이블에 있는 pubs 데이터베이스의 status 컬럼값은 0으로 리셋 됩니다.

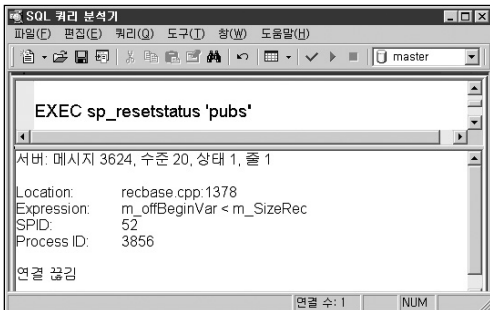


〈응급 복구 모드 해제〉



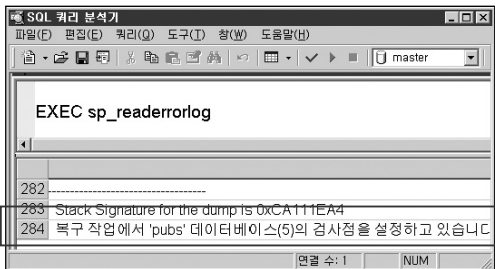
〈데이터베이스 리셋〉

이어서 DBCC DBRECOVER를 IGNOREERRORS 옵션과 함께 실행합니다. IGNOREERRORS 옵션은 데이터베이스를 복구하는 과정에서 오류가 발생해도 무시하고 복구를 계속 진행하겠다는 옵션입니다



〈데이터베이스 복구〉

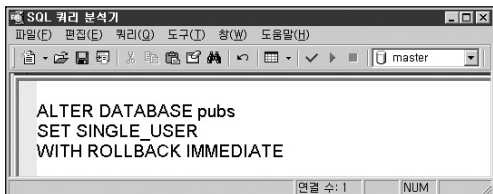
DBCC DBRECOVER ('pubs' , ignoreerrors)를 실행할 때 오류가 발생했지만 복구 작업이 계속 진행되었습니다. SQL Server 오류 로그를 통해서 복구 작업이 진행되었음을 확인합니다.



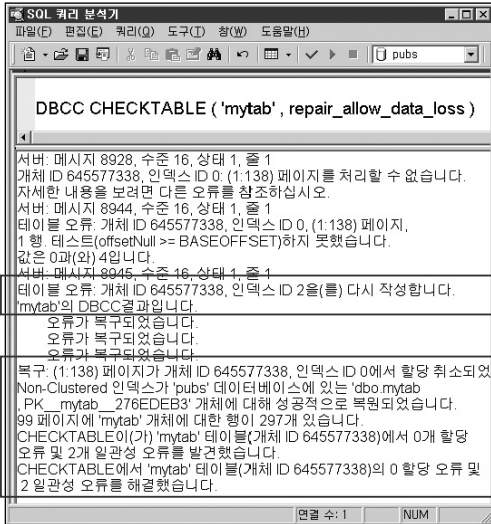
〈데이터베이스 복구 결과 확인〉

이제 DBCC CHECKDB 나 DBCC CHECKTABLE 문을 실행하여 손상된 페이지를 복구합니다. 이미 앞에서 mytab 테이블만 손상되었다는 것을 확인하였으므로 DBCC CHECKDB를 수행하지 않고 해당 테이블에 DBCC CHECKTABLE을 수행합니다.

DBCC CHECKDB 나 DBCC CHECKTABLE 을 REPAIR_ALLOW_DATA_LOSS 옵션과 함께 실행하기 위해서는 데이터베이스를 단일 사용자 모드로 변경해야 합니다.

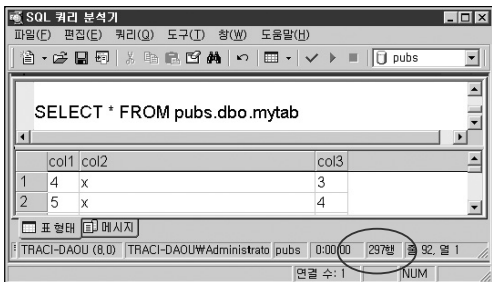


이어서 DBCC CHECKTABLE ('mytab', REPAIR_ALLOW_DATA_LOSS) 구문을 실행합니다.



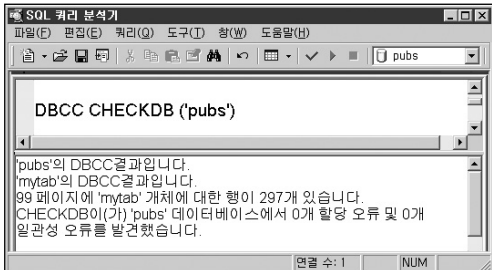
〈DBCC CHECKTABLE 수행 결과〉

위의 그림에서 DBCC CHECKTABLE을 REPAIR_ALLOW_DATA_LOSS 복구 옵션과 함께 실행하여 인덱스 ID 0 (HEAP)에서 손상된 페이지의 할당이 취소되고 인덱스 ID 2 가 다시 작성되었다는 메시지를 확인할 수 있습니다. 이어서 메시지가 나타내는 대로 mytab 테이블이 297개 행만으로 복구가 되었는지 SELECT * FROM mytab 쿼리를 실행하여 확인합니다.



〈DBCC 결과 확인을 위한 QUERY〉

이상이 없다면 DBCC CHECKDB('pubs')를 실행하여 추가로 발생한 문제가 없는지 확인 합니다.



〈해당 데이터베이스 재 점검〉

다른 데이터베이스에 백업해 놓은 col1=1,3 의 데이터를 다시 mytab에 로드하고 ALTER DATABASE pubs SET MULTI_USER WITH NO_WAIT 을 실행하여 단일 사용자 모드를 해제 하고 데이터베이스 옵션 설정을 원래 상태로 복원하면 모두 마무리가 됩니다.

[참고]

손상된 데이터베이스 복구 내용은 SQL Server PASS 2004 PSS LAB의 서버 및 데이터 복구 랩6의 내용입니다. 관련 파일은 다음 사이트에서 다운로드 하실 수 있습니다.

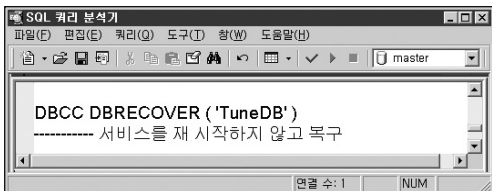
<http://www.microsoft.com/downloads/details.aspx?FamilyId=AEC18337-887F-4EC6-A858-81F84DE8082F&displaylang=en>

■ 데이터베이스 파일이 저장된 디스크와 운영체제의 연결이 비정상적인 경우 복구

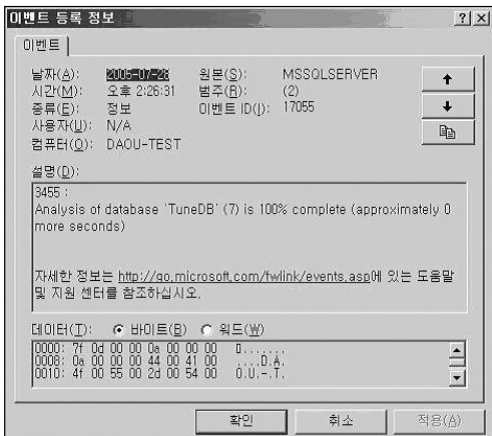
이번에는 데이터 파일이나 로그 파일이 포함된 디스크와 운영체제의 연결이 비정상적인 경우를 살펴 봅니다. 다음은 외장형 스토리지와 운영체제간의 연결 장애로 데이터베이스가 주의 대상 모드로 변경된 경우입니다. 역시, SQL Server 오류 로그와 윈도우즈 이벤트를 로그를 확인합니다.



이와 같은 현상이 발생하면 운영체제와 스토리지 시스템간의 연결을 확인합니다. 윈도우즈 디스크 관리자에서 [동작] 메뉴 선택->[디스크 다시 검사]를 실행하여 디스크를 다시 연결합니다.



연결한 다음에 DBCC DBRECOVER 명령을 통해서 서비스를 다시 시작하지 않고 데이터베이스를 복구합니다. 윈도우즈 이벤트 로그나 SQL Server 오류 로그에 다음과 같이 복구가 완료되었다는 로그를 확인합니다.



■ 로그 파일 손상 시 복구

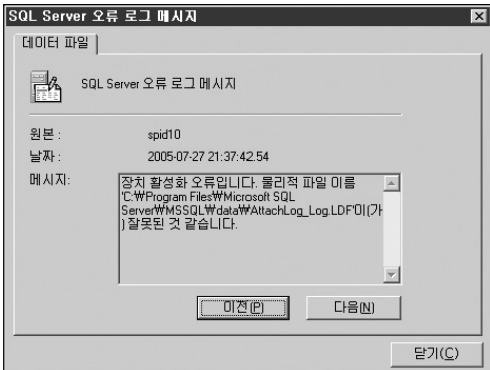
SQL Server의 트랜잭션 로그 파일은 트랜잭션을 시작하기 위해서뿐만 아니라 데이터베이스 손상 시에 데이터베이스가 손상된 시점까지의 데이터를 복구하기 위해서 반드시 필요한 필수 구성 요소입니다.

로그 파일은 데이터 파일과 다른 구조로 구성되어 있습니다. 로그 파일은 여러 개의 가상 로그 파일로 구성 되어 있으며 트랜잭션이 시작되면 각 가상의 로그 파일에 순환 방식으로 트랜잭션을 기록해 나갑니다. 이 가상의 로그 파일은 DBCC SHRINKFILE 명령 실행 시 로그 파일의 크기가 감소하는 단위이기도 합니다. 로그가 백업되어 가상의 로그 파일이 비워져 (TRUNCATE) 다시 사용 할 수 있게 되면 비로소 가상의 로그 파일 단위로 삭제가 가능하게 됩니다.

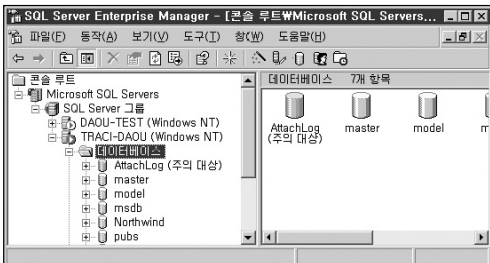
로그 파일은 가상 로그 파일을 전형적인 순차적 액세스 방식에 따라서 순서대로 트랜잭션 로그를 기록하므로, 여러 개의 물리적인 로그 파일은 성능상으로 별다른 기능을 하지 못하고 오히려 관리 부담만 가중시키게 됩니다. 따라서, 특별한 경우가 아니면 로그 파일은 하나만 유지하도록 권장합니다.

SQL Server 2000부터는 복구 모델이 최대(FULL)로 설정된 경우 로그 파일만 손상되지 않았다면 WITH NO_TRUNCATE 옵션으로 로그를 백업할 수 있기 때문에 데이터베이스가 손상된 시점까지 복구가 가능합니다. 대량 로그(BULK_LOGGED) 모델인 경우는 로그에 기록되지 않는 작업이 진행되면 로그백업을 수행해야만 데이터베이스가 손상되더라도 로그 파일이 손상되지 않은 경우 WITH NO_TRUNCATE 옵션으로 로그를 백업할 수 있습니다. 따라서, 복구 모델의 설정에 따라서 올바른 백업 전략을 수립하고 로그 파일은 RAID1 또는 RAID 1+0 으로 구성된 하드디스크에 위치시켜 손상될 확률을 줄여야 합니다.

그럼에도 불구하고 SQL Server 오류 로그에서 확인한 것과 다음 그림과 같이 로그 파일의 손상으로 데이터베이스가 주의 대상 모드가 된 경우 다음과 같이 작업을 수행합니다.



〈SQL Server 오류 로그〉



〈로그 파일의 손상으로 주의 대상 모드로 변경된 경우〉

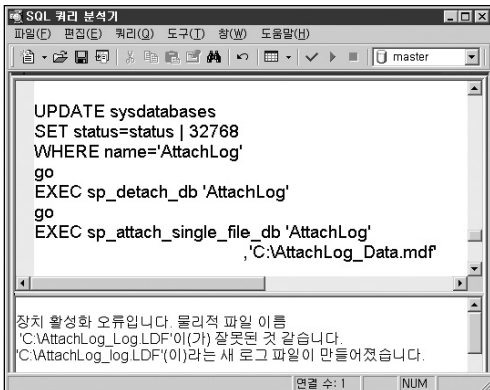
■ 로그 파일 손상 시 진행중인 트랜잭션이 없는 단일 로그 파일의 경우

1) 손상된 데이터베이스를 응급 복구 모드로 변경합니다.

2) sp_detach_db 시스템 저장 프로시저를 사용하여 데이터베이스를 분리합니다. 이때 어떤 오류메시지도 발생하지 않아야 합니다. 오류가 발생하면 손상된 트랜잭션이 존재하는 경우이며 이런 경우에는 DBCC REBUILD_LOG 명령을 사용해서 로그를 다시 생성해야만 하고 반드시 데이터의 무결성을 점검해야 합니다.

3) sp_attach_single_file_db 시스템 저장 프로시저를 사용하여 새로운 로그 파일을 생성하고 데이터베이스를 서버에 연결합니다.

sp_attach_single_file_db '데이터베이스 이름', '새로운 물리적 로그 파일 이름'



(sp_attach_single_file_db 시스템 저장 프로시저 사용 복구)

4) 엔터프라이즈 관리자나 master 데이터베이스의 sysdatabases 시스템 테이블의 status컬럼 값을 변경하여 데이터베이스 옵션 설정을 응급 복구 모드 변경 이전 상태로 재구성합니다.

■ 다중 로그 파일인 경우

다중 로그 파일인 경우나 손상된 트랜잭션이 존재해서 sp_detach_db 시스템 저장 프로시저를 실행해서 분리할 때 오류가 발생한 경우에는 DBCC REBUILD_LOG를 사용해야 합니다.

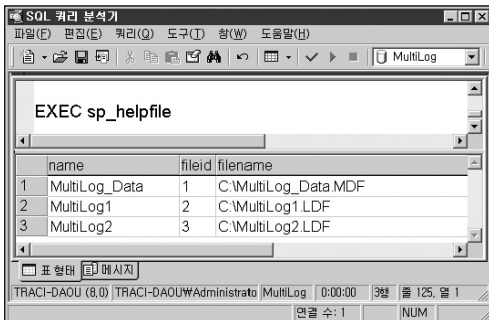
[참고]

다중 로그 파일인 경우 sp_attach_single_file_db 시스템 저장 프로시저를 사용할 수 없다는 기술 문서입니다.

<http://support.microsoft.com/default.aspx?scid=kb;en-us;271223>

DBCC REBUILD_LOG는 sp_attach_single_file_db 시스템 저장 프로시저를 사용하는 경우처럼 데이터베이스를 분리하게 되면 실행할 수 없습니다. 따라서 로그 파일이 손상된 것이 확인되면 데이터베이스를 응급 복구 모드로 변경하고 로그 파일의 개수를 확인합니다.

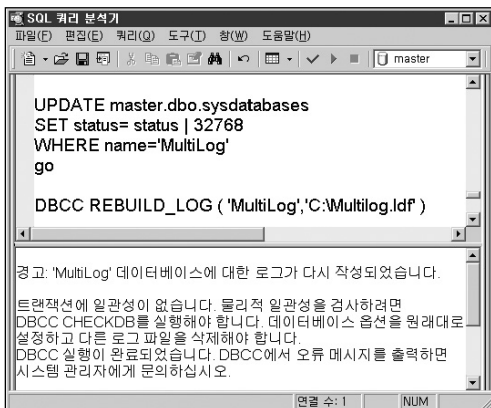
MultiLog 데이터베이스는 다음 그림과 같이 두 개의 로그 파일을 가지고 있습니다.



〈다중 로그 파일의 데이터베이스〉

따라서, 데이터베이스를 분리하지 않고 응급 복구 모드로 변경한 뒤 DBCC REBUILD_LOG 구문을 실행하여 새로운 로그 파일을 생성합니다.

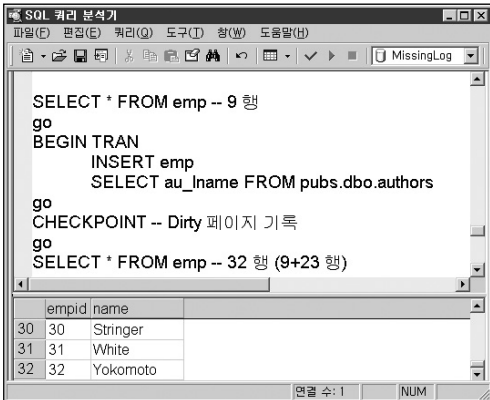
DBCC REBUILD_LOG ('데이터베이스_이름' , '새로운 물리적 로그 파일 이름')



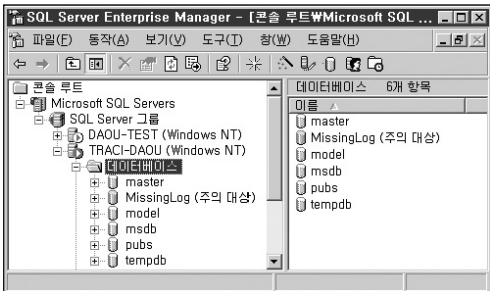
■ 손상된 트랜잭션이 존재하는 경우

로그 파일이 손상되기 전에 진행중인 트랜잭션이 있는 경우에는 데이터베이스를 분리하는 과정에서 오류가 발생해서 sp_attach_singl_file_db 시스템 저장 프로시저로 새로운 로그 파일을 생성하면서 연결할 수 없게 됩니다. 이런 경우에도 다중의 로그 파일과 마찬가지로 DBCC REBUILD_LOG를 통해서 새로운 로그 파일을 생성합니다. 이런 경우에는 데이터의 무결성의 손상이 의심됩니다.

다음과 같이 MissingLog 데이터베이스의 emp 테이블에는 데이터가 9행 있었는데 사용자 트랜잭션으로 23개 행이 추가되었습니다. 이어서 CHECKPOINT가 발생해서 추가로 삽입된 23개 행의 정보가 데이터 파일에 기록되었습니다.

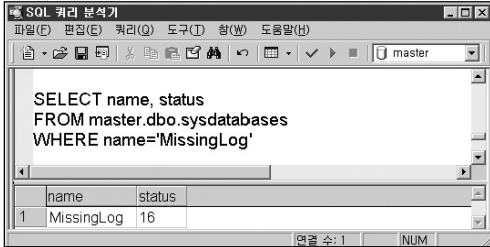


트랜잭션을 COMMIT하지 못한 시점에서 로그 파일이 손상되어 데이터베이스가 주의 대상 모드가 된 상황을 가정하였습니다.



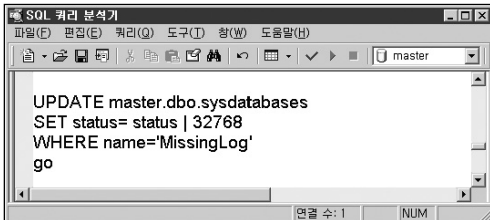
〈트랜잭션 진행 중 로그 파일이 손상된 경우〉

1) 이와 같은 경우 앞의 데이터 파일이 손상된 경우에서 살펴본 바와 같이 손상된 데이터베이스를 응급 복구 모드로 변경합니다. 이때 master.dbo.sysdatabases 시스템 테이블의 status 컬럼을 확인하고 기록합니다.



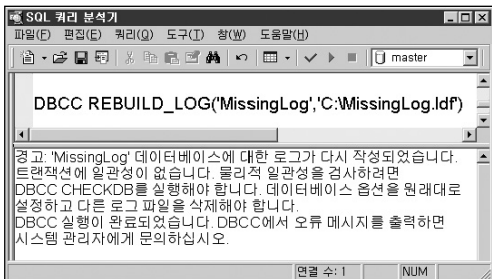
〈현재 데이터베이스 설정 상태 확인〉

2) 데이터베이스를 응급 복구 모드로 변경합니다.



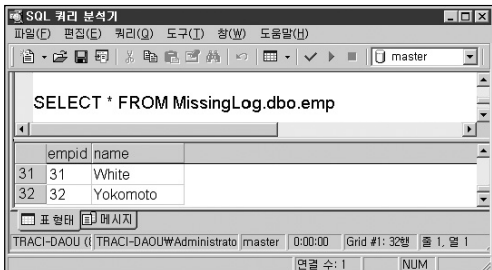
〈응급 복구 모드 변경〉

5) DBCC REBUILD_LOG를 실행하여 새로운 로그 파일을 생성합니다.



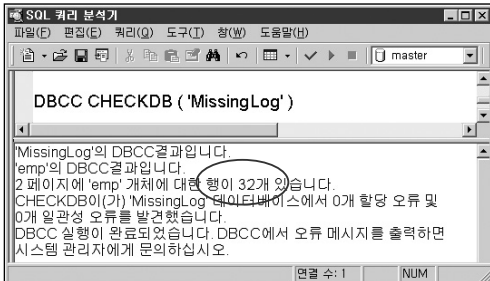
〈DBCC REBUILD_LOG를 실행한 경우〉

이제 SQL Server가 메시지를 통해 경고한 대로 데이터의 무결성을 점검합니다. 먼저, 작업 중이던 emp 테이블을 조회하면 트랜잭션이 COMMIT 되지 않았음에도 불구하고 32 행이 반환됩니다. 로그 파일의 손상으로 인스턴스 복구 프로세스를 진행할 수 없기 때문입니다.



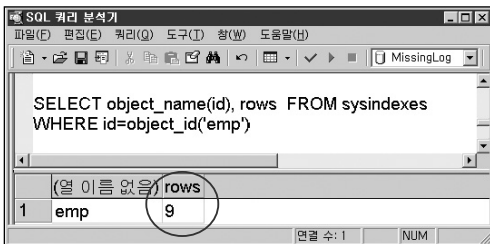
〈무결성 손상 여부 확인〉

DBCC CHECKDB를 실행해서 점검해 보더라도 동일하게 다음과 같이 32행이 존재한다는 정보가 반환됩니다.



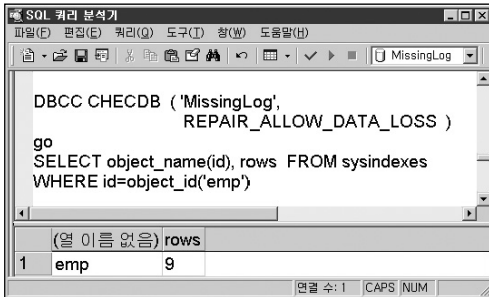
(DBCC CHECKDB 실행 결과)

그러나 일부 다른 정보에는 이전에 9행이 있었던 상태를 유지하고 있습니다. sysindexes 시스템 테이블을 검색해 보면 기존의 9행이 있는 정보를 반환합니다. 이와 같이 로그 파일이 손상되기 전에 진행중인 트랜잭션이 있는 경우는 DBCC CHECKDB 실행결과와 sysindexes 시스템 테이블의 검색결과를 비교하여 무결성이 손상된 테이블과 데이터를 확인해야 합니다.



(DBCC CHECKTABLE 실행 시 결과)

이때 DBCC CHECKDB를 REPAIR_ALLOW_DATA_LOSS 등으로 복구하면 데이터가 서로 다른 정보를 유지하던 상황으로 종료됩니다. 그러나 COMMIT 되지 않은 데이터가 존재하는 무결성이 손상된 상황으로 마무리가 되므로 손상된 로그 파일을 복구 하는 경우에는 로그 파일 손상 시에 발생한 작업등을 면밀히 점검하고 데이터를 수동으로 복구할지 여부를 검토한 뒤 실시해야 합니다. 이와 같은 상황에서는 emp 테이블의 id 컬럼값이 9보다 큰 데이터를 수동으로 삭제해야 합니다.



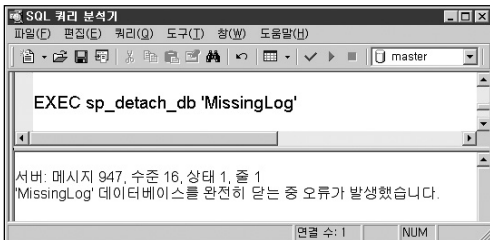
(무결성이 손상된 상황으로 마무리된 DBCC CHECKDB (REPAIR_ALLOW_DATA_LOSS))

[주의]

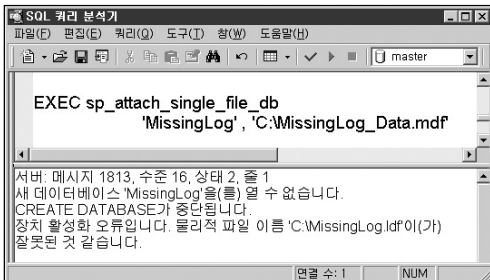
DBCC REBUILD_LOG를 사용해서 새로운 로그 파일을 생성한 경우에는 DBCC CHECKDB실행 결과와 sysindexes 시스템 테이블의 검색결과를 비교해서 무결성이 손상된 테이블과 데이터를 점검해야만 합니다.

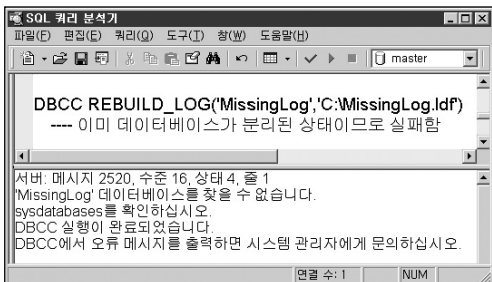
■ 데이터베이스 분리 중 오류가 발생한 경우

새로운 로그 파일 생성을 위해서 데이터베이스를 분리하는 경우 오류가 발생하면 DBCC REBUILD_LOG 명령을 통해서 로그 파일을 생성해야 한다고 했습니다. 그러나, DBCC REBUILD_LOG는 데이터베이스가 분리되기 전에 실시해야만 합니다.



따라서, `sp_attach_single_file_db`를 실행할 수도 없고 DBCC REBUILD_LOG도 실행할 수 없는 상황에 놓이게 됩니다.





이런 경우에는 다음과 같은 방법을 사용하여 새로운 로그 파일로 데이터베이스를 복구합니다.

- 1) 분리된 데이터 파일을 모두 임시 폴더로 이동하고 동일한 이름과 파일구성을 가지는 새로운 데이터베이스를 생성합니다.
- 2) SQL Server 서비스를 종료합니다.
- 3) 새롭게 생성된 데이터베이스의 파일을 모두 삭제하고 임시 폴더로 이동했던 손상된 데이터 파일을 원래 위치로 다시 이동합니다.
- 4) 이제 다시 SQL Server 서비스를 시작합니다. 또 다시 주의 대상 모드인 데이터베이스를 발견하게 됩니다. 앞에서 설명한 DBCC REBUILD_LOG 명령을 사용해 새로운 로그 파일을 생성합니다.
- 5) master.dbo.sysdatabases 의 status 컬럼을 변경하여 기존의 데이터베이스 옵션 설정 상태로 변경합니다.

손상된 시스템 데이터베이스 복구

시스템 데이터베이스가 손상된 경우 SQL Server 서비스가 시작하지 못하거나 일부 기능이 정상적으로 실행되지 않을 수 있습니다.

아래의 표는 각 시스템 데이터베이스가 손상되었을 경우 발생하는 문제점을 설명합니다.

손상 데이터베이스	손상 시 문제점	복구를 위한 작업
master	SQL Server 서비스 시작 실패	Rebuildm.exe 수행 후 백업을 사용한 복구
model	tempdb를 비롯한 데이터베이스 생성 실패 및 SQL Server 서비스 시작 실패	Rebuildm.exe 수행 후 백업을 사용한 복구 , sp_attach_db
msdb	SQL Agent 서비스 시작 실패	상동
tempdb	tempdb를 사용하는 작업 실패 (데이터 복구, 정렬, 집계 등)	서비스 재시작 등

1. master 데이터베이스 복구 및 이동

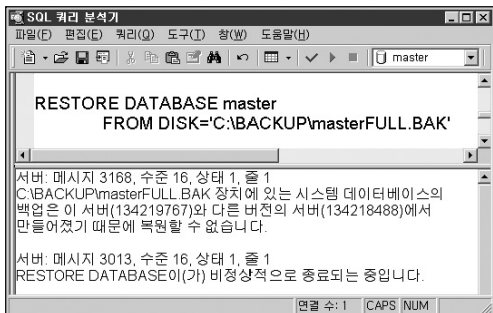
■ master 데이터베이스 복구

master 데이터베이스가 손상되면 SQL Server는 서비스 시작이 실패하며 따라서 복구 작업을 포함한 어떤 작업도 할 수 없습니다. 이렇게 master 데이터베이스가 손상되었다면 일단 SQL Server 서비스가 시작할 수 있도록 Rebuildm.exe 유틸리티를 사용해서 모든 시스템 데이터베이스를 초기화하거나 특정 시점의 master.mdf 파일과 mastlog.ldf 파일의 복사본을 사용하여 서비스를 시작할 수 있습니다. master 데이터베이스를 복구하고자 하는 경우에는 SQL Server 서비스를 반드시 단일 사용자 모드로 시작하여야 하며 현재 설치되어 있는 SQL Server의 빌드 번호와 백업된 master 데이터베이스의 빌드 번호가 일치해야만 합니다.

[경고]

시스템 데이터베이스 복구의 경우 현재 설치된 SQL Server의 빌드 번호와 백업된 시스템 데이터베이스의 빌드 번호가 반드시 일치해야만 복구가 가능합니다.

만약, 백업된 master 데이터베이스의 빌드 번호가 현재 설치되어있는 버전의 빌드 번호와 일치하지 않는 경우에는 다음과 같이 오류 메시지가 반환되고 복구 작업이 실패합니다.



SQL Server의 빌드 번호는 서비스 팩이나 보안 패치를 적용할 때 변경됩니다. 따라서, DBA는 SQL Server에 서비스 팩이나 보안 패치 등을 적용하는 경우에 반드시 시스템 데이터베이스를 백업해 두어야 합니다.

[주의]

시스템 데이터베이스의 빌드 번호는 SQL Server 서비스 팩과 보안 패치의 적용에 따라 변경됩니다. 따라서 DBA는 서비스 팩이나 보안 패치를 설치하는 경우 반드시 시스템 데이터베이스를 백업합니다. 뿐만 아니라 설치된 서비스 팩과 보안 패치 파일을 따로 저장하고 설치 정보를 문서화해서 보관합니다.

■ 서비스 팩이나 보안 패치의 빌드 번호를 확인하는 방법

① 서비스 팩이나 보안 패치 설치 파일의 빌드 번호 구성은 아래 그림과 같습니다.

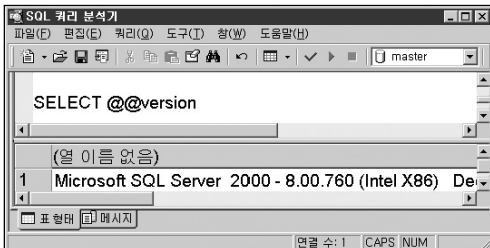
〈보안 패치의 빌드 번호 구성〉

빌드 넘버 : ProductName- KBArticleNumber-X.YY.ZZZZ-LangName.exe

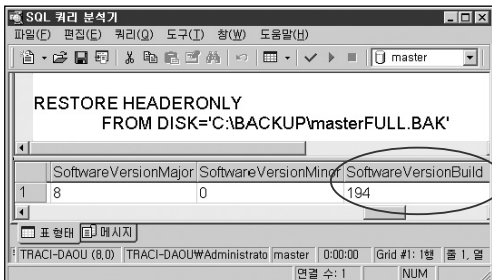
- 1) ProductName은 제품 이름과 버전 정보입니다.
- 2) KBArticleNumber는 관련된 Microsoft 기술 자료 문서의 ID입니다.
- 3) X는 주 버전 번호를 나타냅니다.
- 4) YY는 두 자리 수의 부 버전 번호를 나타냅니다.
- 5) ZZZZ 는 보안 패치 등 핫픽스 번호를 나타냅니다. (빌드 번호)
- 6) LangName 은 보안패치가 현지화된 언어의 세 자리 약어입니다.
예를 들어, KOR(한국어), ENU(영어), JPN (일본) 등이 있습니다.

예) SQL2000-KB815495-8,00,0818-KOR.exe
/ SQL2000-KB899761-v8,00,2040-x86x64-KOR.exe

② 현재 설치되어있는 SQL Server 빌드 번호 확인은 xp_msver, sp_server_info 등의 저장 프로시저나 @@version, SERVERPROPERTY('ProductVersion') 시스템 함수를 사용하여 가능합니다. 다음은 @@version 시스템 함수를 사용하여 확인하는 방법입니다.



- ③ 시스템 데이터베이스 백업의 빌드 번호는 RESTORE HEADERONLY 구문을 통해서 확인하게 됩니다. 다음은 master 데이터베이스 백업의 빌드 번호를 확인하는 경우입니다.



이제 SQL Server의 빌드 번호와 백업의 빌드 번호를 확인하였다면 일치 여부에 따라서 해당 되는 복구방법을 살펴보도록 하겠습니다.

CASE 1. 현재 설치되어있는 SQL Server 빌드 번호보다 master 데이터베이스 백업의 빌드 번호가 낮은 경우

이와 같은 경우는 먼저 SQL Server를 재 설치하고 master 데이터베이스를 복구합니다.

- 1) 백업된 시스템 데이터베이스 빌드 번호와 동일한 서비스 팩과 보안 패치를 준비합니다.
- 2) 복구하고자 하는 인스턴스를 [프로그램 추가/삭제] 애플릿을 통해 삭제합니다.

[참고]

SQL Server를 [프로그램 추가/삭제] 애플릿을 통해서 삭제할 수 없는 경우에는 <http://support.microsoft.com/default.aspx?scid=kb;ko;290991>의 문서를 참고하여 수동으로 제거하기 바랍니다.

- 3) 2)번에서 준비한 SQL Server 2000 및 서비스 팩, 보안 패치를 순서에 따라 설치합니다.
- 4) SQL Server 서비스를 단일 사용자 모드에서 시작하고 master 데이터베이스를 복구합니다.

CASE 2. 현재 설치되어있는 SQL Server 빌드 번호와 시스템 데이터베이스 백업의 빌드 번호가 일치하는 경우

- 1) Rebuildm.exe 유틸리티를 실행합니다.

방법① : 시작 버튼을 누른 뒤 실행창에서 Rebuildm.exe를 입력한 뒤 엔터 키를 누릅니다.

방법② : Windows 탐색기를 사용하여 "C:\Program Files\Microsoft SQLServer\80\Tools\Binn" 폴더로 이동하여 Rebuildm.exe를 더블 클릭합니다.

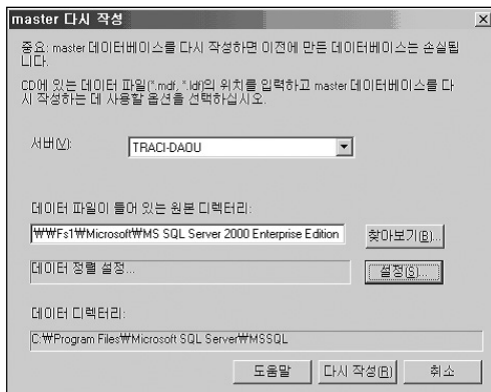
[주의]

SQL Server 프로그램 CD의 x86\DATA 폴더를 이용해서 Rebuildm.exe 유틸리티를 사용하는 경우 SQL Server 2000 프로그램 CD의 파일은 읽기 전용 속성을 변경하지 못하는 버그로 인하여 설치에 실패하게 됩니다. 따라서, SQL Server 프로그램 CD의 x86\DATA 폴더를 하드 디스크로 복사한 뒤 읽기 전용 속성을 해제하고 사용해야 합니다.

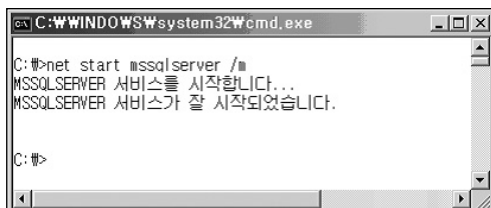
[참조 문서]

<http://support.microsoft.com/default.aspx?scid=kb;en-us;273572>

- 2) **[찾아보기]** 버튼을 눌러서 초기 시스템 데이터베이스 파일이 들어 있는 폴더의 경로를 지정합니다.



- 3) 데이터 정렬은 SQL Server 설치시의 정렬과 동일하지 확인합니다. 기본적으로 운영 체제의 언어에 따라 설정됩니다. 다시 작성 버튼을 눌러 재구성을 진행합니다.
- 4) 시스템 데이터베이스의 재구성이 완료되면 SQL Server 서비스를 종료합니다. 명령 프롬프트를 사용해서 SQL Server 서비스를 단일 사용자 모드로 시작합니다. /m 옵션을 추가하여 SQL Server 서비스를 시작하면 SQL Server 서비스가 단일 사용자 모드로 시작됩니다.

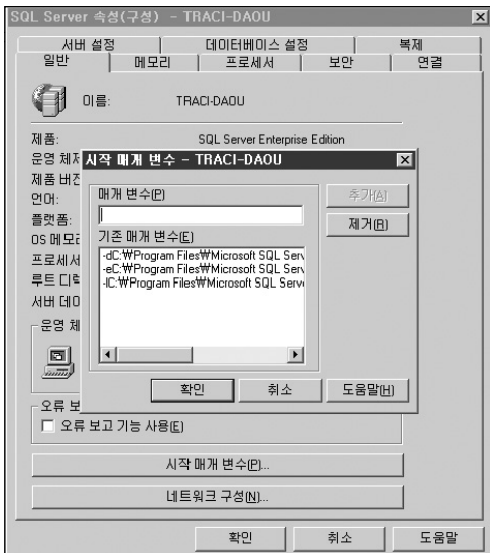


5) 쿼리 분석기로 연결한 뒤 master 데이터베이스의 백업을 사용하여 복구를 진행합니다.

■ 현재 설치된 SQL Server 의 master 데이터베이스의 위치를 변경하는 경우

현재 master 데이터베이스가 설치된 위치의 하드 디스크의 저장 공간이 부족한 경우 등의 원인으로 master 데이터베이스의 위치를 변경하고자 하는 경우에는 엔터프라이즈 관리자에서 시작 매개변수 정보를 변경하거나 레지스트리의 해당 값을 변경합니다. 먼저, 엔터프라이즈 관리자를 사용하는 경우입니다.

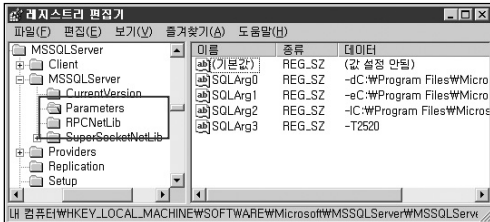
- 1) SQL Server 엔터프라이즈 관리자를 실행합니다.
- 2) [서버의 속성]을 클릭합니다.
- 3) [시작 매개변수] 버튼을 클릭합니다.
- 4) [기존 매개 변수] 부분의 -d 에 지정되어있는 경로에는 master 데이터베이스의 데이터 파일을 이동할 위치로 수정하고, -l 에 지정되어 있는 경로에는 master 데이터베이스의 로그 파일을 이동할 위치로 수정합니다.
- 5) -e 에 지정되어 있는 경로를 오류 로그 파일 폴더를 이동하고자 하는 위치로 수정합니다.
- 6) [확인] 버튼을 누르고 SQL Server 서비스를 종료합니다.
- 7) master.mdf 파일과 masterlog.ldf 파일을 이동하고자 하는 위치로 복사합니다.
- 8) SQL Server 서비스를 시작합니다.
- 9) 정상적으로 동작하는지 확인한 다음에 원본 파일을 삭제합니다.



(SQL Server 시작 매개 변수 입력 창)

레지스트리를 수정하여 master 데이터베이스의 위치를 변경하려는 경우 시작버튼을 누르고 실행창에서 Regedit.exe 또는 Regedt32.exe를 입력하고 [확인] 버튼을 눌러 레지스트리 편집기를 실행합니다.

- 1) "HKEY_LOCAL_MACHINE\Software\Microsoft\MSSQLServer\MSSQLServer" 로 다음과 같이 이동합니다.



- 2) SQLArg0 (데이터 파일 위치)/ SQLArg1 (오류 로그 파일 위치)/ SQLArg2 (로그 파일 위치) 부분의 [값 데이터]를 이동하고자 하는 경로로 변경합니다.
- 3) 레지스트리 편집기를 종료합니다.
- 4) master.mdf 파일과 masterlog.ldf 파일을 이동하고자 하는 위치로 복사합니다.
- 5) SQL Server 서비스를 시작합니다.
- 6) SQL Server가 정상적으로 동작하는지를 확인한 뒤 원본 파일을 삭제합니다.

2. model 데이터베이스 복구 및 이동

■ model 데이터베이스 복구

model 데이터베이스는 사용자 데이터베이스 및 tempdb를 포함한 시스템 데이터베이스를 만들기 위한 템플릿 역할을 수행합니다. 따라서, model 데이터베이스가 손상된 경우 사용자 데이터베이스를 생성할 수 없게 되고 tempdb도 만들어지지 않습니다. 뿐만 아니라 SQL Server 서비스가 정상적인 상태로 시작할 수 없게 됩니다. 만약, model 데이터베이스가 손상을 입은 상태에서 tempdb도 생성되지 않은 경우라면 tempdb를 사용하는 작업인 복구 프로세스를 진행할 수도 없습니다.

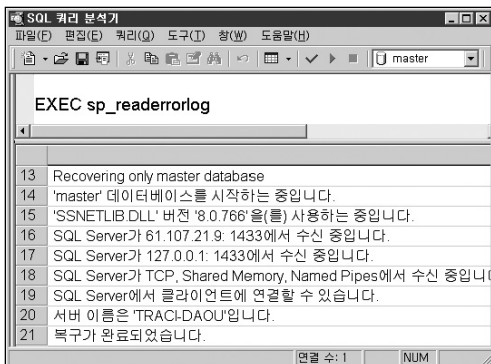
다음에서 model 데이터베이스가 손상되어 tempdb도 생성할 수 없는 경우에 model 데이터베이스가 복구되지 않는 상황을 살펴 보겠습니다. model 데이터베이스가 손상되면 SQL Server 서비스를 정상적으로 시작할 수 없으므로 다음과 같이 명령프롬프트를 실행하고 SQL Server가 설치된 경로로 이동하여 sqlservr.exe를 -c 옵션과 -T3608 추적 플래그를 사용하여 시작합니다.



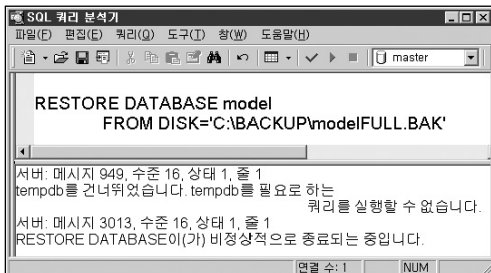
[참고]

-T3608 : master 데이터베이스를 제외한 데이터베이스의 인스턴스 복구 프로세스를 실행하지 않고 서비스를 시작하는 추적 플래그입니다.

T3608 추적 플래그를 지정하여 SQL Server 서비스를 시작하게 되면 SQL Server 오류 로그에서 master 데이터베이스를 제외한 데이터베이스의 인스턴스 복구 프로세스에 대한 로그가 기록되지 않는 것을 확인할 수 있습니다.



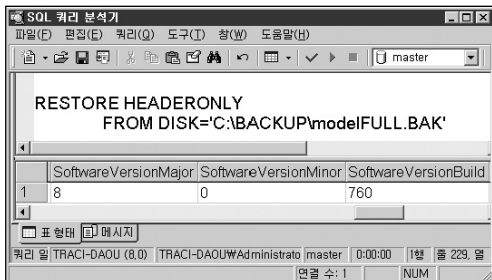
이 상태에서의 model 데이터베이스의 복구는 tempdb를 생성하지 못하였으므로 앞에서 설명한 대로 다음과 같은 오류 메시지를 남기고 실패하게 됩니다.



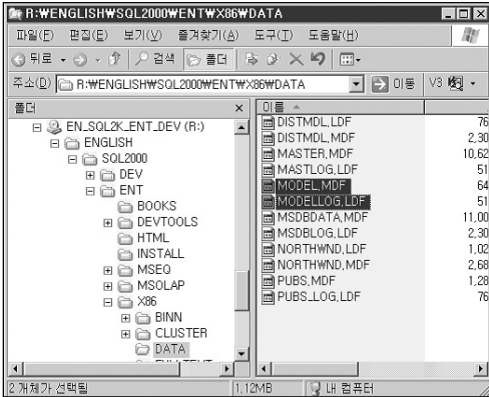
따라서, model 데이터베이스를 복구하기 위해서는 master 데이터베이스 복구에서 살펴본 바와 같이 Rebuildm.exe 유틸리티를 이용해서 시스템 데이터베이스를 모두 새로 구성한 뒤 기존 시스템 데이터베이스 백업을 통해서 master, model, msdb를 순서대로 복구하거나 model 데이터베이스만 손상된 상태라면 SQL Server 프로그램 CD의 x86\DATA 폴더의 model.mdf 파일과 model.ldf 파일 또는 기존에 파일 백업으로 복사해 놓은 model 데이터베이스 파일들을 SQL Server 시스템 데이터베이스가 위치한 폴더로 복사한 뒤 서비스를 시작하는 방법 등을 통해서 SQL Server 서비스를 정상적으로 시작한 뒤 진행해야 합니다.

다음은 Rebuildm.exe 유틸리티를 사용하지 않고 간단하게 SQL Server 프로그램 CD의 model 데이터베이스 파일을 복사해서 SQL Server 서비스를 시작하고 model 데이터베이스를 복구하는 예제입니다.

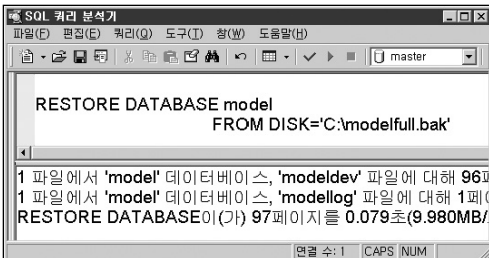
- 1) 복구 작업을 하기에 앞서서 현재 SQL Server의 빌드 번호와 model 데이터베이스 백업의 빌드 번호를 확인합니다.



- 2) 빌드 번호가 서로 일치한다면 해당 model 데이터베이스 백업으로 복구가 가능하므로 SQL Server 프로그램 CD에서 model 데이터베이스 파일을 시스템 데이터베이스 폴더로 복사합니다.

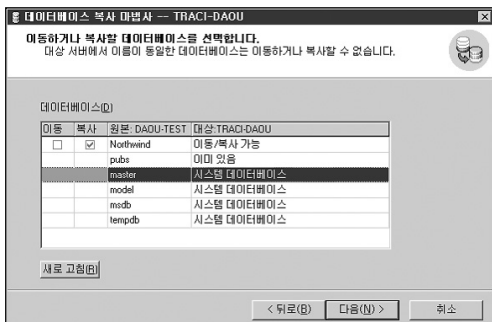


- 3) 이제 SQL Server 서비스를 시작하고 다음과 같이 복구를 진행합니다.

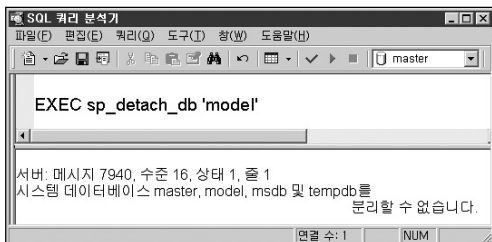


■ model 데이터베이스의 이동

사용자 데이터베이스의 이동은 데이터 복사 마법사, 백업 및 복구, sp_detach_db/sp_attach_db 시스템 저장 프로시저 등을 이용해서 가능합니다. 그렇지만 다음에서 시스템 데이터베이스의 경우 이동과 복사 모두 선택할 수 없는 것처럼 시스템 데이터베이스는 데이터베이스 복사 마법사를 이용할 수 없고 일반적인 상태에서는 sp_detach_db/ sp_attach_db 시스템 저장 프로시저도 사용할 수 없습니다.



〈데이터베이스 복사 마법사〉



〈sp_detach_db 시스템 저장 프로시저 사용의 경우〉

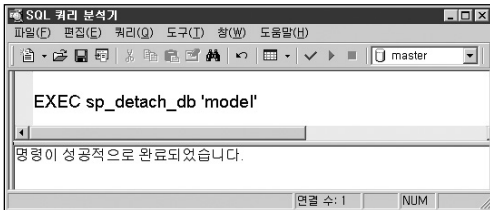
model 데이터베이스와 msdb 데이터베이스의 이동은 SQL Server 서비스를 서비스가 아닌 응용 프로그램으로 -c 옵션과 -T3608 추적 플래그를 함께 지정하여 시작한 뒤 sp_detach_db/sp_attach_db 시스템 저장 프로시저를 사용하여 분리하거나 연결해야 합니다.

수행 방법은 다음과 같습니다.

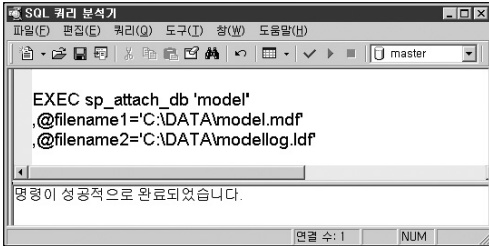
- 1) 명령프롬프트를 실행해서 SQL Server 프로그램이 설치된 폴더로 이동합니다.
sqlservr.exe -c -T3608 을 입력하고 실행합니다.



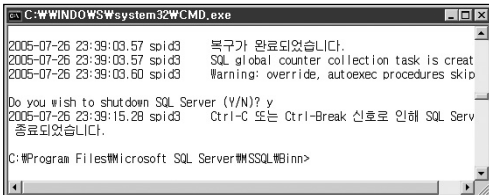
- 2) sp_detach_db 시스템 저장 프로시저를 실행하여 model 데이터베이스를 분리합니다.



- 3) 이동하고자 하는 위치로 model 데이터베이스 파일을 이동한 뒤 sp_attach_db 시스템 저장 프로시저를 실행해서 연결합니다.



- 4) SQL Server를 응용 프로그램으로 시작한 명령 프롬프트 창에서 Ctrl+C 키를 입력한 다음 Y 키를 눌러 SQL Server를 종료합니다. 다시 SQL Server 서비스를 정상적으로 시작합니다.



3. msdb 복구 및 이동

■ msdb 복구

msdb에는 SQL 에이전트 서비스와 관련된 메타데이터가 저장됩니다. 따라서, msdb가 손상된 경우 SQL 에이전트 서비스는 시작이 실패할 수 있습니다. 또한, 백업 및 복제 작업, DTS 패키지 등과 관련된 메타 데이터의 저장소이므로 손상 시 기존에 실시한 데이터베이스 백업의 정보와 복제작업, DTS 패키지 등이 손실됩니다. 그러나, msdb가 손상된 경우라 할 지라도 SQL Server 서비스는 정상적으로 동작합니다. 따라서, 가용한 msdb의 백업이 존재하는 경우는 SQL Server 서비스의 재 시작 없이 복구할 수 있습니다. 그렇지만 이전에 살펴본 시스템 데이터베이스 복구작업과 마찬가지로 SQL Server의 빌드 번호와 백업의 빌드 번호가 일치해야 복구할 수 있음을 기억하기 바랍니다. 빌드 번호의 확인과 복구 방법은 master 데이터베이스 복구를 참조하기 바랍니다.

msdb의 가용한 백업이 없는 경우에는 SQL Server 프로그램 CD에서 msdb 파일들을 복사해서 사용할 수 있습니다. 그러나, 이러한 경우 기존에 msdb에 저장된 작업, 운영자, 경고, DTS 패키지 등은 다시 생성해야만 합니다.

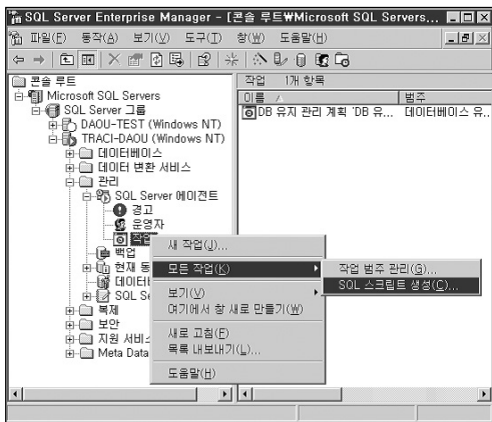
따라서, 재난에 대비하기 위해서는 msdb가 변경될 때마다 master 데이터베이스 및 model 데이터베이스와 함께 msdb도 백업할 뿐만 아니라 SQL Server 에이전트에서 생성한 작업, 운영자, 경고 등은 스크립트로 백업하고, DTS 패키지는 DTS 파일로 백업해 놓는 이중의 안전장치를 고려해야 합니다.

SQL Server 에이전트에 생성한 작업, 운영자, 경고를 엔터프라이즈 관리자를 통해서 스크립트로 생성하는 과정은 다음과 같습니다.

- 1) 경고 스크립트 생성 : 엔터프라이즈 관리자 --> 관리--> SQL Server 에이전트--> 경고 --> 오른쪽 버튼 클릭-->[모든 작업]-->[SQL 스크립트 생성]

2) 운영자 스크립트 생성 : 엔터프라이즈 관리자 --> 관리-->SQL Server 에이전트-->운영자-->오른쪽 버튼 클릭-->[모든 작업]-->[SQL 스크립트 생성]

3) 작업 스크립트 생성 : 엔터프라이즈 관리자 --> 관리-->SQL Server 에이전트-->작업-->오른쪽 버튼 클릭-->[모든 작업]-->[SQL 스크립트 생성]



〈엔터프라이즈 관리자를 이용한 작업 스크립트 생성〉

이와 같이 엔터프라이즈 관리자를 이용해서 스크립트를 생성하는 방법은 SQL-DMO를 사용하므로 SQL Server 에이전트의 작업으로 생성하고 스케줄을 이용해서 정기적으로 실행하는 것은 가능하지 않습니다. 그러나, 작업을 스크립트로 생성하는 쿼리를 마이크로소프트 사이트나 기타 커뮤니티 사이트에서 제공하고 있으므로 소개합니다.

〈AGENT JOB 스크립트 생성〉

<http://support.microsoft.com/default.aspx?scid=kb;en-us;321835>

```

--sp_OA params
DECLARE @cmd varchar(255)      -- Command to run
DECLARE @oSQLServer int        -- OA return object
DECLARE @hr int -- Return code  --User params
DECLARE @FileName varchar(200) -- File name to script jobs out
DECLARE @Server varchar(30)
    -- Server name to run script on. By default, local server. --SQL DMO Constants
DECLARE @ScriptType varchar(50)
DECLARE @Script2Type varchar(50) SET @ScriptType = '327'
    --Send output to file, Transact-SQL, script permissions, test for existence,
    -- used quotedcharacters.
SET @Script2Type = '3074'
    -- Script Jobs, Alerts, and use CodePage 1252.
    -- Set the following properties for your server
SET @FileName = 'c:\sqlJobs.sql'
SET @Server = @@SERVERNAME --CREATE The SQLDMO Object
EXEC @hr = sp_OACreate 'SQLDMO,SQLServer', @oSQLServer OUT
    --Set Windows Authentication
EXEC @hr = sp_OASetProperty @oSQLServer, 'LoginSecure', TRUE
    --Connect to the Server
EXEC @hr = sp_OAMethod @oSQLServer, 'Connect', NULL, @server
    --Script the job out to a text file
SET @cmd
    = 'Jobserver.Jobs.Script(' + @ScriptType + ', "' + @FileName + '" , ' + @
      Script2Type + ' )'
EXEC @hr = sp_OAMethod @oSQLServer, @cmd
    --Close the connection to SQL Server
    --If object is not disconnected, the processes will be orphaned.
EXEC @hr = sp_OAMethod @oSQLServer, 'Disconnect' --Destroy object created.
exec sp_OADestroy @oSQLServer

```

[참고]

테이블이나 프로시저 등의 데이터베이스 개체 생성 스크립트를 만드는 작업도 `sp_OA` 확장 저장 프로시저를 사용한 소스의 예제와 SQL Server 프로그램 경로의 `Upgrade` 폴더에 있는 `scptxfr.exe`를 사용하는 간단한 소스 예제 등을 다음의 커뮤니티 사이트에서 참조하여 활용하면 편리합니다. 회원 가입 후 참고하기 바랍니다.
<http://www.sqlservercentral.com/forums/shwmessage.aspx?forumid=5&messageid=202336>

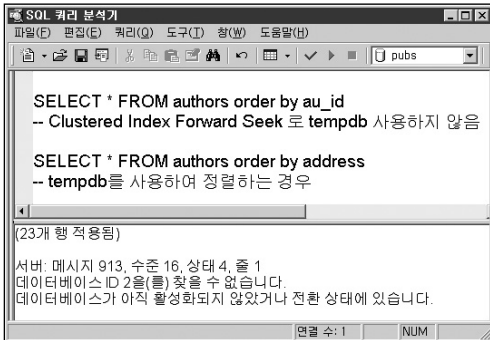
■ msdb 이동

시스템 데이터베이스는 복사 마법사를 이용할 수 없고 일반적인 상태에서 `sp_detach_db/sp_attach_db` 저장 프로시저도 사용할 수 없다는 것은 model 데이터베이스 이동에서 이미 살펴 보았습니다. msdb의 이동도 모델데이터베이스의 이동 방식과 동일합니다. 따라서, T3608 추적 플래그와 함께 응용 프로그램 모드로 시작하여 분리한 뒤 연결을 하여야 합니다. 주의 사항은 model 데이터베이스와 함께 이동시킬 경우 반드시 model 데이터베이스를 먼저 연결한 다음에 msdb를 연결해야 합니다.

4. tempdb 복구 및 이동

■ tempdb 복구

tempdb는 SQL Server에서 정렬, GROUP BY를 사용한 집계, 커서 사용, 임시테이블 및 테이블변수 사용, 일부 JOIN, SORT_IN_TEMPDB 옵션을 사용한 인덱스 생성, 데이터베이스의 복구 작업 등에서 사용됩니다. 따라서, 손상 시에는 이와 같은 작업들을 진행할 수 없게 됩니다. tempdb의 손상 시 tempdb를 사용하는 작업은 다음과 같이 913 오류를 발생하며 정상적으로 실행되지 않습니다.



[참고]

tempdb를 생성하지 않고 SQL Server를 시작하기 위해서는 명령 프롬프트에서 다음과 같이 시작합니다.

C:\Program Files\Microsoft SQL Server\MSSQL\bin\sqlservr.exe -c -T3609

다음과 같이 tempdb가 손상되어 주의 대상 상태인 경우 SQL Server 서비스를 재시작만 하더라도 복구되기도 하지만 그렇지 않은 경우도 발생합니다.

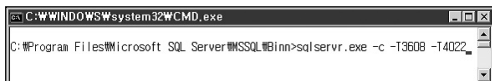


〈TEMPDB 가 손상된 경우〉

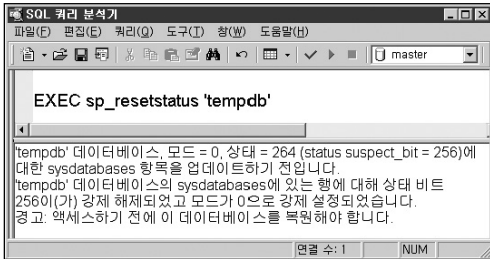
이런 경우 다음의 절차를 따라서 tempdb를 재 생성합니다.

1) 명령 프롬프트를 실행하고 SQL Server 프로그램이 설치된 경로로 이동합니다.

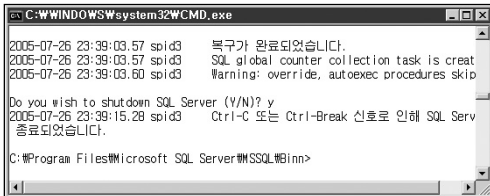
T3608 추적 플래그를 사용하여 응용 프로그램으로 시작합니다. 이때 T4022 추적 플래그도 함께 사용합니다. T4022 추적 플래그는 서비스가 시작될 때 자동으로 실행하도록 구성한 저장 프로시저를 실행되지 않도록 하는 옵션입니다. 상황에 따라서는 -f (최소 구성 시작) 옵션으로 시작해야 하는 경우도 있습니다.



- 2) 쿼리 분석기에서 `sp_resetstatus` 프로시저를 사용하여 주의 대상 모드를 해제합니다.



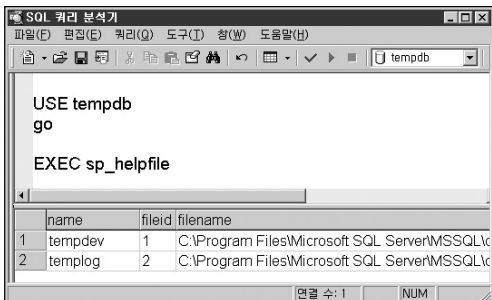
- 3) SQL Server를 응용 프로그램으로 시작한 명령 프롬프트 창에서 Ctrl+C 키를 입력한 다음에 Y 키를 눌러 SQL Server를 종료합니다. 다시 SQL Server 서비스를 정상적으로 시작합니다.



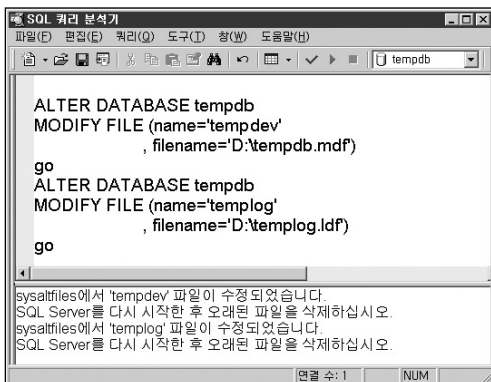
■ tempdb 이동

SQL Server 데이터베이스 중 유일하게 tempdb의 경우에는 ALTER DATABASE ~ MODIFY FILE 구문으로 데이터베이스의 파일의 물리적인 경로를 변경할 수 있습니다. 단, 변경한 다음에 SQL Server 서비스를 다시 시작해야만 변경된 위치에 새로운 tempdb가 생성됩니다. 이 경우 이전에 생성된 tempdb의 파일들은 자동으로 삭제되지 않기 때문에 불필요한 공간을 차지하게 되므로 수동으로 삭제합니다.

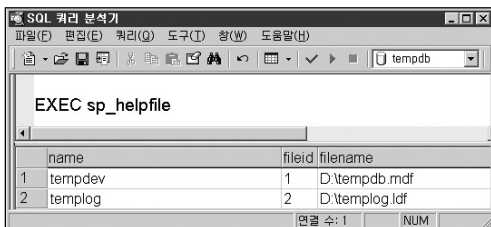
1) 기존 파일의 위치를 확인합니다.



2) ALTER DATABASE 구문으로 파일의 물리적 경로를 변경합니다.



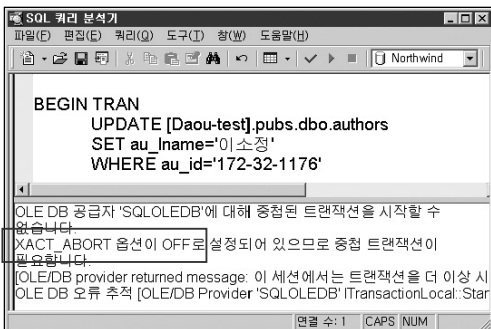
3) 메시지 창에 표시된 지시대로 SQL Server 서비스를 다시 시작하고 새로운 위치로 변경되었는지 확인한 다음에 기존의 파일을 삭제합니다.



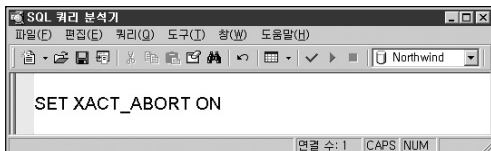
분산 트랜잭션을 시작할 수 없는 경우

■ SET XACT_ABORT 옵션을 활성화하지 않은 경우

SQL Server에서 분산 트랜잭션을 시작하기 위해서는 XACT_ABORT 세션 옵션을 반드시 ON으로 설정해야 합니다. XACT_ABORT 옵션을 ON으로 설정하면 T-SQL 문에서 런타임 오류가 발생할 경우 전체 트랜잭션이 종료된 후 롤백됩니다. OFF로 설정하면 오류를 발생 시킨 T-SQL 문만 롤백되고 작업이 계속 진행됩니다. 구문 오류와 같은 컴파일 오류는 XACT_ABORT 옵션 설정으로 영향을 받지 않습니다.

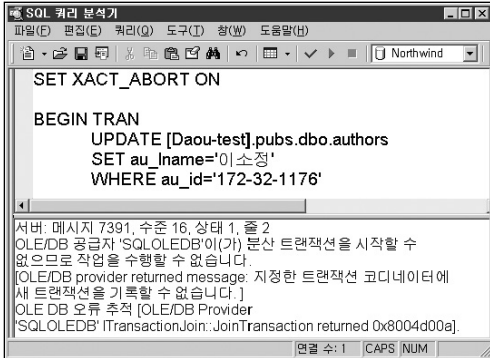


XACT_ABORT 옵션을 다음과 같이 활성화합니다.



■ 윈도우즈 2003 서버에서 DTC 사용 시 오류

윈도우즈 2003 서버에서 DTC를 사용하는 분산 트랜잭션을 실시하는 경우에는 다음과 같은 오류 메시지가 발생합니다.



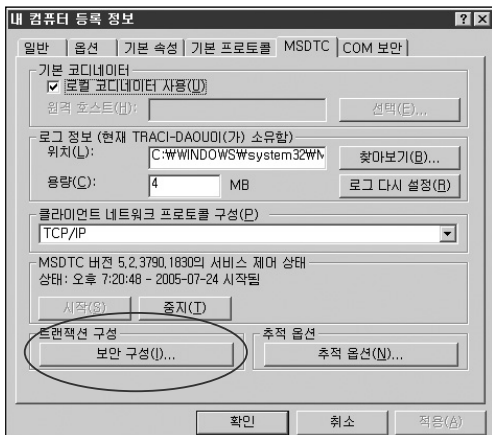
이것은 윈도우즈 2003 서버가 기본적으로는 로컬 호스트 내에서의 분산 트랜잭션만을 허용하고 다중의 호스트가 참여하는 분산 트랜잭션에 대해서는 구성되지 않았기 때문입니다. 이 경우 문제를 해결하기 위해서는 먼저 현재의 분산 트랜잭션과 관련된 구성 정보를 확인합니다. 구성 요소 서비스의 내 컴퓨터의 속성을 선택하고 MSDTC 탭을 선택하면 [로컬 코디네이터 사용]에 체크가 되어 있음을 확인합니다. 이어서 하단에 있는 [트랜잭션 구성의 보안구성] 버튼을 눌러 [네트워크 DTC 액세스]의 체크박스 상태를 확인합니다. 만약 이 체크박스가 선택되어 있지 않다면 다중 호스트의 분산 트랜잭션이 구성되어 있지 않다는 것을 의미합니다.

다른 호스트와의 분산 트랜잭션을 처리하기 위해서는 다음의 단계를 진행해야 합니다.

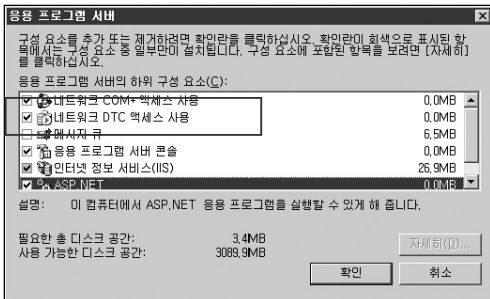
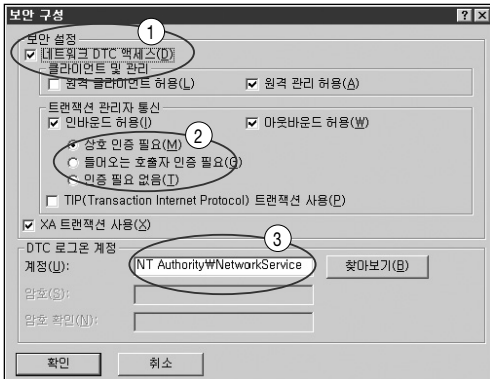
- 1) 네트워크 DTC 액세스 설정
- 2) RPC Security 구성 (트랜잭션 관리자 인증)
- 3) TCP/IP 이름 분해 구성 (Hosts/DNS)

먼저, 네트워크 DTC 액세스 설정입니다.

관리 도구에서 구성 요소 서비스를 실행하거나 시작에서 실행을 선택한 후 Dcomcnfg를 입력한 후 엔터 키를 누릅니다. 내 컴퓨터의 속성을 선택합니다. 다음과 같은 창이 실행되면 화면 하단의 트랜잭션 구성의 [보안구성] 버튼을 누릅니다.



다음과 같이 보안 구성 창에서 ①번으로 표시된 [네트워크 DTC 액세스]의 체크박스를 선택합니다. 창을 닫기 위해 확인 버튼을 누르면 MSDTC 서비스를 재시작합니다. 또는 제어판의 [프로그램 추가/삭제]에서 [윈도우즈 구성 요소 추가/삭제] 버튼을 누릅니다. 윈도우즈 구성 요소 중에서 응용 프로그램 서버를 선택하고 [자세히] 버튼을 누릅니다. 두 번째의 [네트워크 DTC 액세스 사용] 체크박스를 선택합니다. [확인], [다음], [마침] 버튼을 순서대로 눌러 설치를 마칩니다. 이제 1단계가 완료되었습니다.



두 번째 단계는 네트워크 DTC의 보안 구성 단계입니다. 이 단계는 서비스 팩의 설치 여부에 따라서 다릅니다.

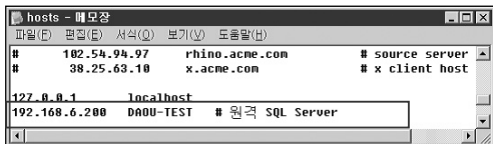
① 서비스 팩1 이 설치된 경우 위 그림의 보안 구성 창에서 ②번 부분의 트랜잭션 관리자 통신 영역의 [인증 필요 없음]을 선택합니다.

② 서비스 팩이 설치되지 않은 경우는 레지스트리 편집기를 실행해서 HKEY_LOCAL_MACHINE\SOFTWARE\Microsoft\MSDTC로 이동합니다. 편집 메뉴를 누르고 새로 만들기-> DWORD 값을 선택합니다. TurnOffRpcSecurity 라고 입력하고 값을 1로 할당합니다.

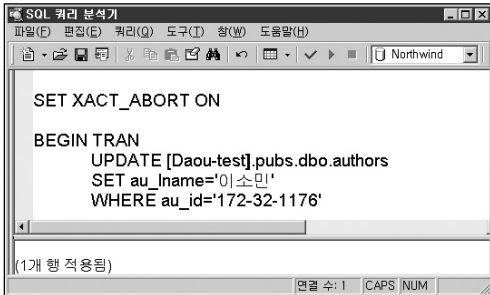


③ 두 경우 모두 보안 구성 창에서 ③번 부분의 DTC 로그온 계정이 NT Authority\Network Service임을 확인합니다.

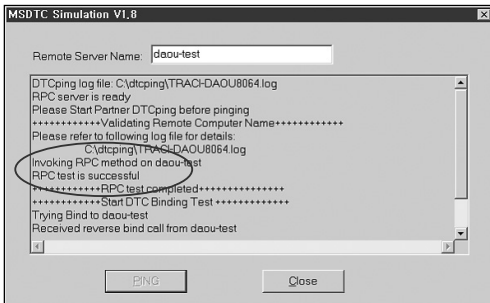
세 번째 단계에서는 분산 트랜잭션에 참여하는 호스트간에 호스트 이름으로 통신이 가능하도록 구성합니다. 회사 내에서 DNS 서버를 사용하는 경우에는 모두 DNS에 이름을 등록합니다. DNS 서버가 구성되지 않은 경우에는 %SystemRoot%\system32\Drivers\Etc 폴더의 hosts 파일에 상대방 서버의 호스트이름과 IP 주소를 등록합니다.



이와 같은 작업을 분산 트랜잭션을 사용할 모든 호스트에서 실시합니다. 작업이 완료되면, 다시 쿼리 분석기의 분산 트랜잭션 테스트를 진행하여 분산 트랜잭션이 정상적으로 진행되는지 확인합니다.

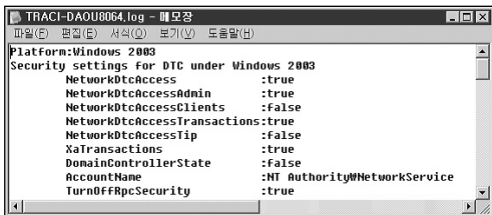


DTC의 통신 여부와 구성 상태는 마이크로소프트가 제공하는 DTCPing.exe 라는 유틸리티를 통해서 점검할 수 있습니다. DTCPing.exe를 다운로드 받아 테스트하려는 호스트에 복사합니다. DTCPing.exe를 각각의 호스트에서 실행하고 상대방 호스트 이름을 입력한 후 PING 버튼을 누르면 통신 여부를 보여주며 자세한 사항은 지정된 폴더에 생성되는 로그 파일을 통해 확인할 수 있습니다.



DTCPing 유틸리티의 로그는 다음 사항을 포함합니다.

① 대상 DTC 의 구성 정보



② 대상 호스트의 IP 구성

③ 대상 호스트의 NIC 구성

④ 대상 호스트의 host 파일과 lmhost 파일 구성 정보

⑤ DTC 통신 결과

〈DTCPing.exe 다운로드 사이트〉

<http://support.microsoft.com/default.aspx?scid=kb;ko;306843>

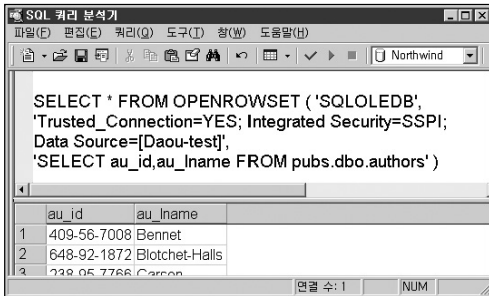
■ 분산 쿼리 사용시 주의 사항 및 오류

다른 서버로의 연결이 자주 발생하지 않는 경우는 OPENROWSET 함수나 OPENDATA SOURCE 함수 등 AD-HOC 쿼리를 사용하는 것이 효율적이지만 상시 연결을 필요로 하는 경우에는 Linked Server를 구성하여 사용하는 것이 효율적입니다. 먼저 Linked Server를 구성하였을 때 사용할 수 있는 OPENQUERY 함수와 Four-Part 이름을 사용한 쿼리의 성능상 문제를 비교하고 이와 관련해서 발생하는 오류를 점검합니다.

많은 사이트에서 개발자가 손쉽게 Query를 작성할 수 있다는 장점 때문에 Four-Part 이름을 사용하는 Query를 사용하지만 성능상 문제를 정확히 이해한 뒤 사용해야 합니다. Four-Part 이름을 사용한 Query는 로컬 서버의 리소스를 사용하고 OPENQUERY 함수는 Pass-through Query라고 해서 원격 서버의 리소스를 사용합니다. 즉 Pass-through Query는 Query자체를 원격 서버에 보내서 최적화 작업을 원격 서버가 실시하고 그에 따른 결과만을 Remote Scan 하여 반환 받지만, Four-Part 이름을 사용한 Query는 원격 서버가 동일한 SQL 서버인 경우에는 원격 서버에게 sp_tableinfo_rowset, sp_columns_rowset, sp_indexes_rowset, sp_check_constraints_rowset, sp_table_statistics_rowset 등의 시스템 저장 프로시저 실행을 요청해서 반환 받은 정보를 바탕으로 로컬 서버가 최적화를 한 뒤에 Remote Query를 실행하게 됩니다. 따라서, Four-Part Query는 Pass-through Query에 비해서 이론적으로 비효율적입니다. 그러나 모든 경우가 이와 같지는 않습니다. UPDATE 구문과 DELETE 구문을 예로 들면 Four-Part 이름을 사용한 Query가 오히려 좋은 성능을 냅니다. UPDATE와 DELETE를 실행하는 경우 OPENQUERY 함수는 OLEDB Provider의 기능의 제한으로 인해서 UPDATE 나 DELETE 구문을 전송하지 못하므로 워크테이블을 생성하는 방법으로 처리하게 됩니다. 이와 같은 경우는 Four-Part 이름을 사용한 Query를 사용하는 것이 효율적입니다. 따라서, UPDATE나 DELETE 구문을 사용하거나 원격의 저장 프로시저를 사용하는 경우, 쿼리의 길이가 8000 바이트를 넘는 경우 등은 Four-Part 이름을 사용한 Query를 사용하고 나머지 경우에는 OPENQUERY 함수를 사용하는 것이 성능상 유리할 수 있습니다. 오라클과 같은 이기종 DBMS인 경우에는 SQL 서버와 구문이 다른 옵티마이저 힌트를 주거나 오라클 함수의 실행과 같은 OLEDB Provider가 이해하지 못하는 작업은 Four-Part 이름을 사용한 Query는 불가능하며 검색 조건을 제대로 지정하더라도 테이블 스캔을 해서 성능을 저해할 수 있습니다. 따라서, 이와 같이 원격 서버로 Query를 실행하는 작업은 충분한 테스트를 진행한 뒤 어떤 방법을 사용할 것인지 결정해야 합니다.

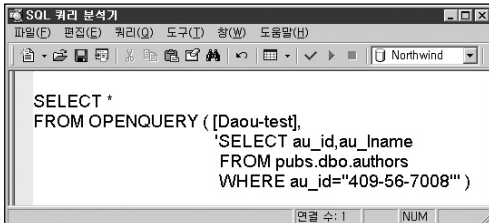
1) OPENROWSET 함수 사용 시 윈도우즈 인증

회사 내 보안을 위하여 계정 정보와 패스워드를 소스 내에 하드코딩하는 것은 바람직하지 않습니다. OPENROWSET 함수 사용시 윈도우즈 인증을 사용할 것을 권장합니다. 윈도우즈 인증을 통한 OPENROWSET 함수를 사용하는 경우는 다음 항목을 지정해야 합니다.
Trusted_Connection=YES; Integrated Security=SSPI ;Data Source=[대상 인스턴스 이름]

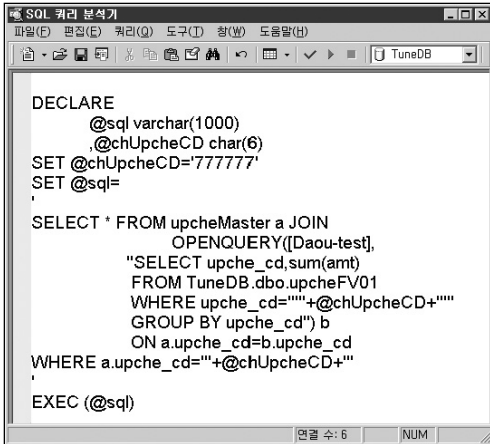


2) Linked Server를 이용한 SELECT / INSERT

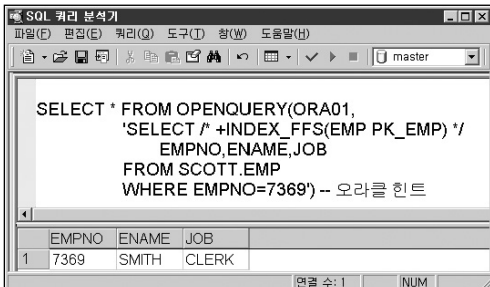
원격 서버에 SELECT 또는 INSERT 구문을 실행하는 경우 OPENQUERY 함수를 사용하는 것이 효율적입니다. 그러나 다음 예제 가운데 하나와 같이 집계연산을 포함하여 조인을 하는 등의 복잡한 쿼리는 검색조건이 효율적으로 지정되도록 주의하여야 합니다.



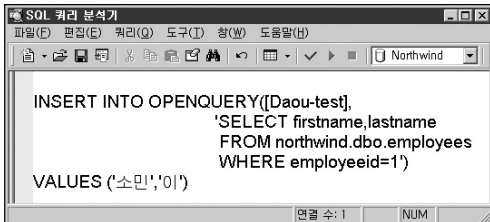
(OPENQUERY 함수를 이용한 SELECT)



〈검색조건을 효율적으로 지정한 동적 쿼리〉



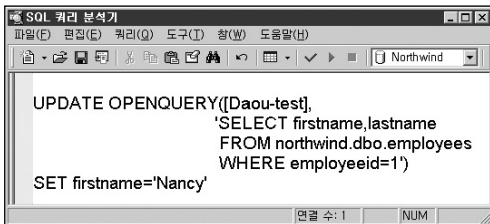
〈OPENQUERY 함수를 이용한 오라클 힌트 사용〉



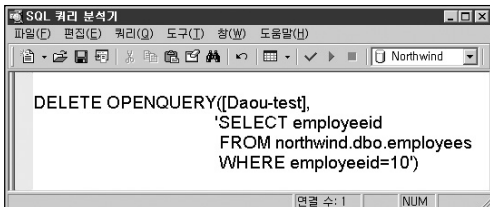
(OPENQUERY 함수를 이용한 INSERT)

3) Linked Server를 이용한 UPDATE / DELETE

원격 서버에 UPDATE 또는 DELETE 구문을 실행하는 경우 Four-Part 이름을 사용한 Query가 효율적입니다. 다음과 같은 OPENQUERY 함수를 사용한 UPDATE와 DELETE의 사용을 자제합니다.



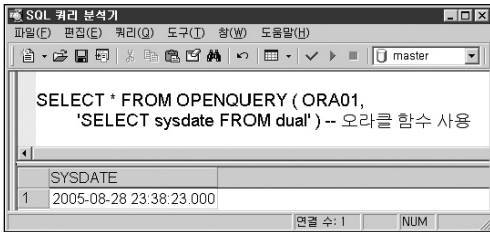
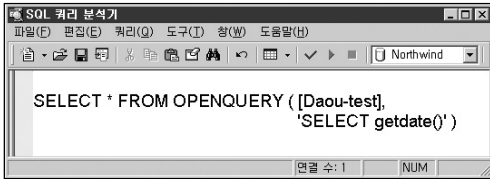
(OPENQUERY 함수를 이용한 UPDATE)



(OPENQUERY 함수를 이용한 DELETE)

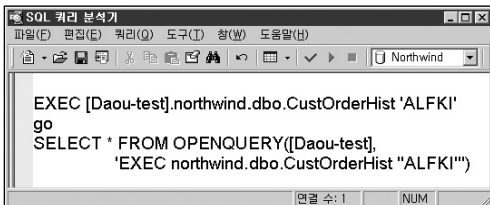
4) Linked Server를 사용한 함수 실행

원격 서버의 함수를 실행하고자 하는 경우에는 다음과 같이 OPENQUERY 함수를 사용합니다. Four-Part 이름을 사용해서 원격 서버에 대한 함수 호출은 지원되지 않습니다.

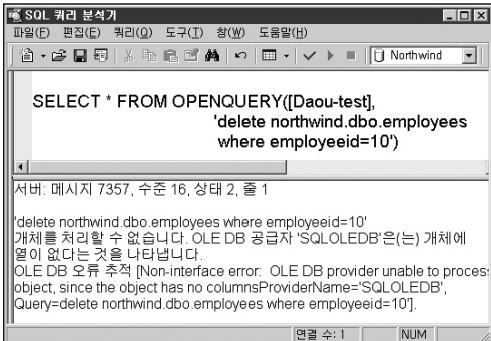


5) Linked Server를 사용한 저장 프로시저 실행

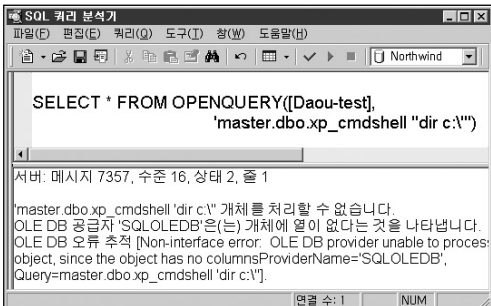
원격 서버의 저장 프로시저를 실행하고자 하는 경우 충분한 검증을 필요로 합니다. Four-Part 이름을 사용해서 원격의 저장 프로시저를 호출하는 것이 다소 효율적인 경우가 많습니다.



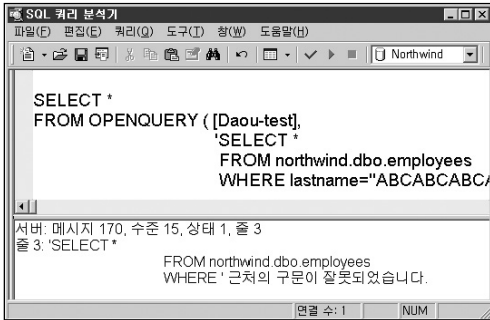
- 6) DML 문장을 직접 OPENQUERY 함수 안에 작성하는 것은 바람직하지 않습니다. 정상적으로 수행되지 않고 다음과 같이 오류를 발생하게 됩니다.



- 7) OPENQUERY 함수를 사용해서 원격 서버의 확장 저장 프로시저를 실행할 수 없습니다.



8) OPENQUERY 함수를 사용해서 8000자 이상의 쿼리를 전송하지 못합니다.



[참고]

Linked Server를 사용하여 OPENQUERY를 실행하는 중 오류가 발생하는 경우 DBCC TRACEON (7300, 3604) 구문을 실행하면 보다 자세한 오류 정보를 반환 받을 수 있습니다.

대용량 작업 시 주의 사항

대용량 데이터베이스에서 작업을 하려면 사전에 상당한 주의를 필요로 합니다.

대량의 배치를 실행하는 경우나 대량의 수정이나 삭제 작업 등은 충분한 저장공간을 필요로 할 뿐만 아니라 대량의 트랜잭션 로그 파일 공간을 필요로 합니다. 따라서, 작업 전에 데이터 파일과 로그 파일의 크기를 적절하게 구성해야 합니다. 이와 같은 사전 준비가 적절하지 않으면 어려움을 겪을 수 있습니다. 다음에서 대용량 작업 시 주의 사항을 살펴 봅니다.

1) 대용량의 BCP

데이터 파일과 로그 파일은 미리 충분한 공간을 할당합니다. 로그 파일의 경우 로그가 백업된 뒤 TRUNCATE 를 수행해서 파일을 줄일 수 있는 공간은 CHECKPOINT의 LSN(로그 시퀀스 번호) 과 최근에 발생한 COMMIT 의 LSN, 가장 오래된 활성 트랜잭션의 LSN 가운데 가장 오래된 LSN 가운데에서 가장 적은 값의 LSN이 기준이 됩니다. 따라서, 대형의 트랜잭션이 발생 중에는 해당 트랜잭션 이후의 작업은 설사 COMMIT 되고 백업되었다고 할지라도 로그를 잘라낼 수 없어서 로그 파일 크기를 줄일 수 없습니다. 해당 디스크에 빈 공간이 부족한 경우에는 임시로 로그 파일을 추가합니다. 작업이 종료되면 로그를 백업하고 임시로 만든 로그 파일을 삭제합니다. 또는, 데이터베이스의 복구 모델을 대량 로그(Bulk_logged)로 변경하여 작업하는 것도 해결책이 될 수 있습니다. 대량 로그 모드에서는 로그 파일에는 BCP 작업이 시작되었다는 정보와 익스텐트 할당 등의 정보만 기록되고 BCP 작업의 세부 변경 사항은 기록되지 않습니다. 그렇지만 BCP 작업으로 변경되거나 할당된 익스텐트의 정보가 각 데이터 파일의 BCM 이라는 페이지에 기록되고 로그 백업 시 해당 BCM의 정보를 가지고 변경된 데이터도 백업하게 됩니다.

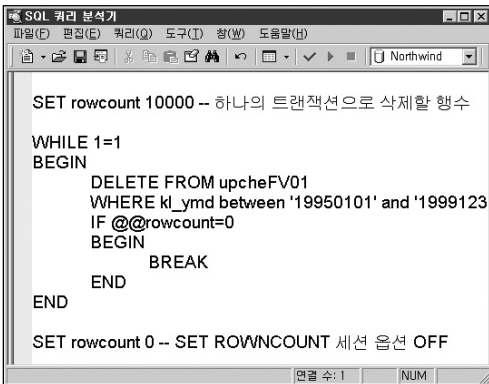
또는 -b 옵션을 사용해서 적절한 값을 입력하는 것을 고려합니다. -b 옵션은 COMMIT 되는 행수를 지정합니다. 다음과 같이 지정하면 1000 행마다 COMMIT을 수행합니다.



그러나 이런 경우는 1000행마다 COMMIT되므로 인덱스의 단편화가 하나의 트랜잭션으로 작업한 경우보다 훨씬 심하게 발생하게 됩니다.

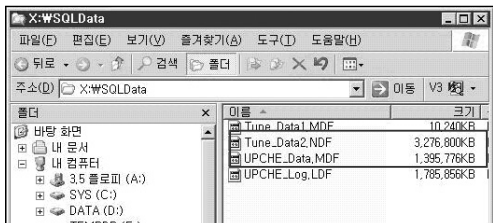
2) 대용량의 삭제 / 수정

대량의 삭제 시 로그 파일의 사용을 줄이고 여러 번에 나누어 다음과 같이 삭제할 수 있습니다. SQL Server 2005는 DELETE 와 UPDATE 구문에서도 TOP 키워드를 사용할 수 있어서 편리합니다.

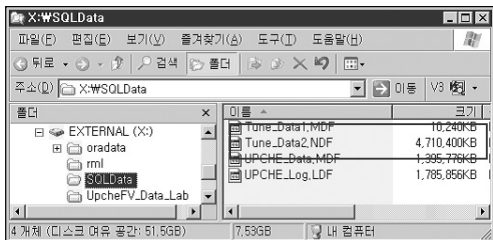


3) 대용량 인덱스 생성 작업

A. 클러스터드 인덱스를 생성하는 작업은 테이블 크기의 약 120% 공간을 추가로 필요로 합니다.



다음은 1,429MB 크기의 테이블에서 클러스터드 인덱스를 생성하는 경우입니다. 인덱스를 생성하기 전 데이터 파일 크기는 약 3,200MB인 것이 인덱스를 생성하는 동안 약 1,400MB 증가했음을 확인할 수 있습니다.



B. 데이터 파일에 여유 공간이 충분하지 않은 경우나 I/O의 경합을 감소시키고자 하는 경우에는 SORT_IN_TEMPDB 옵션을 사용합니다. 테이블이 저장되어 있는 디스크에서는 읽기작업이 수행되고 tempdb가 있는 디스크에서는 정렬을 위한 쓰기 작업이 진행되어 I/O가 분산됩니다.

- C. 인덱스 생성 작업은 MAXDOP 쿼리 힌트를 지원하지 않으므로 병렬 작업의 범위를 지정하기 위해서는 “Max Degree Of Parallelism” 서버 옵션의 값을 수정하여야 합니다. 이 값을 적절하게 지정하여 인덱스 생성 작업이 전체 CPU를 점유하지 않도록 조치합니다. SQL Server 2005는 인덱스 생성과 재구성 작업 시에 MAXDOP 쿼리 힌트를 지원합니다.
- D. 클러스터드 인덱스를 생성할 때는 테이블에 배타적 잠금이 걸립니다. 그러므로 작업이 종료될 때까지 해당 테이블에 대한 모든 작업이 블로킹됩니다. 그러므로 서비스 중에 작업하는 것은 피해야 합니다.
- E. 년클러스터드 인덱스를 생성할 때는 테이블에 공유 잠금이 걸립니다. 따라서, 해당 인덱스가 만들어지는 테이블에서는 읽기 작업만 가능합니다. 그러므로 이 작업 역시 서비스 중에는 하지 않는 것이 좋습니다.

[참고]

SQL Server 2000은 인덱스 생성이나 재구성 시 온라인 작업을 지원하지 않습니다. 온라인으로 인덱스를 생성하거나 재구성하는 기능은 SQL Server 2005부터 지원합니다. SQL Server 2000에서 지원하는 DBCC INDEXDEFRAG 명령은 온라인으로 작업되지만 병렬 작업을 지원하지 않으며 인덱스에 사용된 페이지의 물리적인 순서를 논리적인 순서대로 배열하는 작업을 합니다.

- F. 인덱스를 재구성할 때 삭제 후 다시 생성하지 않도록 합니다. 그렇게 하지 않으면 삭제와 재생성으로 이중의 시간이 소요됩니다. CREATE INDEX ~WITH DROP_EXISTING 이나 DBCC DBREINDEX를 사용하면 작업시간을 단축하고 저장 공간도 적게 사용할 수 있습니다.

4) 파일 사이즈를 줄이는 작업

데이터베이스의 데이터 파일이 적절한 크기를 초과하여 지나치게 커졌을 경우 DBA는 해당 파일을 DBCC SHRINKFILE 명령어를 통해서 줄이게 됩니다. 이런 경우에 작업중인 테이블이나 페이지 등에 배타적 잠금을 걸게 되어 업무 동시성을 저해합니다. 따라서, 업무 시간 중의 작업을 삼가해야 합니다. 커다란 데이터 파일 하나를 적은 크기의 데이터 파일 여러 개로 분리해서 관리하는 것이 효율적입니다.

5) 대용량의 테이블 관리

대용량의 테이블인 경우 인덱스도 대용량으로 생성되어 생성이나 재구성 작업이 효율적이지 못합니다. 또한 현재 운영에 필요하지 않는 오래된 데이터를 다른 데이터베이스나 테이블로 이동하는 작업을 진행할 경우에도 오랜 시간을 필요로 합니다. 따라서 대용량 테이블이나 인덱스의 경우 파티션 적용을 고려할 수 있습니다. SQL Server 2005는 테이블과 인덱스의 파티션을 지원합니다. SQL Server 2000은 파티션은 지원하지 않고 유사한 효과를 얻을 수 있는 분할된 뷰를 제공합니다. 온라인 설명서의 “분할된 뷰”를 참고하기 바랍니다.

6) 대용량 작업 중 서버가 장애를 입은 경우

대용량 작업 중 정전과 같은 돌발 사고로 SQL Server가 다시 시작되는 경우입니다. SQL Server가 수정이 발생한 메모리의 데이터 페이지들을 하드 디스크의 데이터 페이지로 저장하는 방법은 크게 두 가지가 있는데 한 가지는 데이터 버퍼 캐시 공간의 사용률에 따라서 Lazy Writer 라는 프로세스가 동작하고, 다른 한 가지는 SQL Server 서비스가 시작될 때 수행되는 인스턴스 복구 프로세스의 효율성을 위해서 CHECKPOINT 라는 프로세스가 동작합니다. 따라서, 대량으로 로드되던 데이터들은 서버가 다운되기 전에 이미 상당한 량의 데이터를 하드 디스크의 데이터 페이지에 저장했거나 저장공간을 할당받은 상태입니다. 그러나 트랜잭션은 COMMIT 하지 못하였으므로 데이터의 무결성을 유지하기 위해 SQL 서버는 서비스 시작 시점에 인스턴스 복구 프로세스를 진행하여, COMMIT 되었으나 데이터 파일에 기록되지 않은 트랜잭션을 롤포워드(ROLL FORWARD)하여 기

록하고 COMMIT되지 않았으나 데이터 파일에 저장된 데이터나 할당 정보에 대해서 롤백 작업을 진행합니다.

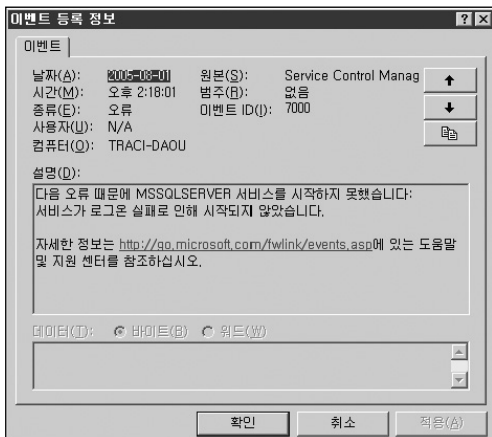
이런 경우 롤백이 완료될 때까지 데이터베이스는 오랜 시간 동안 사용할 수 없습니다.

SQL Server 2005는 인스턴스 복구 프로세스가 진행될 때 롤포워드 작업이 종료되면 즉시 데이터베이스를 사용할 수 있도록 하여 데이터베이스를 효율적으로 운영할 수 있습니다. 반면에 SQL Server 2000에서는 인스턴스 복구 프로세스가 종료될 때까지 기다리거나 대용량의 배치 외에 다른 작업이 진행되지 않았다면 본 가이드의 **[로그 파일 손상 시 복구]** 부분을 참조하여 로그 파일을 삭제하고 새로 생성하는 방법으로 해당 데이터베이스를 빠르게 사용할 수도 있습니다. 그러나, 해당 로그 파일에 롤백하고자 하는 배치 작업 외에 다른 트랜잭션이 기록된 경우 트랜잭션 로그의 유실로 데이터의 무결성이 손상을 입게 될 수 있으므로 주의하여야 합니다. 해당 배치 작업만 실행된 경우라 할지라도 DBCC CHECKDB를 실행하고 sysindexes 시스템 테이블을 검사하여 일관성 오류가 발생한 테이블이 존재하는지 여부와 논리적인 데이터의 무결성 여부를 반드시 확인해야 합니다.

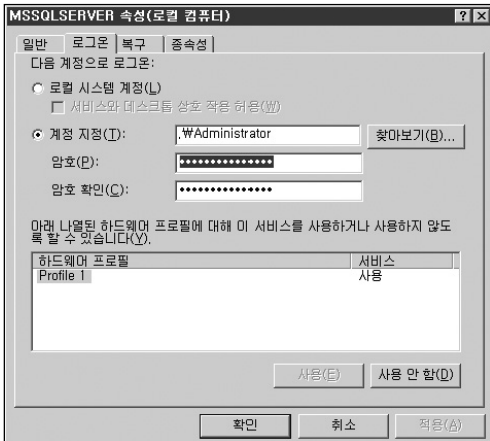
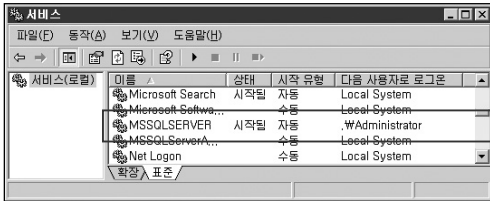
기타 자주 발생하는 오류

■ Administrator 계정의 패스워드를 변경한 뒤 서비스가 시작되지 않는 경우

Administrator 계정의 패스워드를 변경한 뒤 서비스가 시작되지 않는 경우가 종종 발생합니다. 이런 경우 이벤트 뷰어의 시스템 이벤트를 로그를 확인합니다.



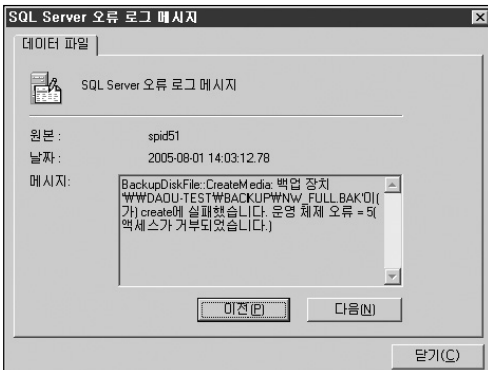
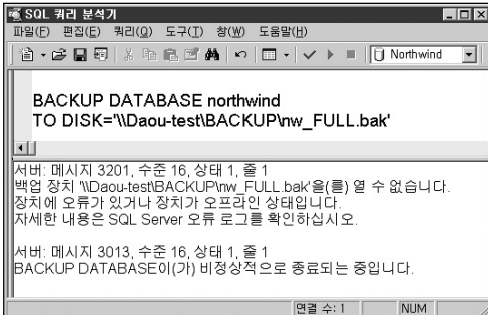
서비스 로그온 계정의 인증 실패로 인하여 서비스를 시작하지 못했다는 메시지가 기록되어 있는지 확인합니다. 관리 도구에서 서비스를 선택합니다. MSSQLSERVER를 선택하고 더블 클릭하여 서비스의 속성 창을 엽니다.



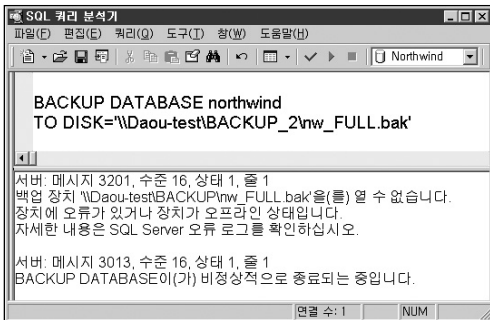
로그온 탭을 선택하고 [계정 지정]의 암호 칸에 변경된 패스워드를 입력합니다. SQLSERVERAGENT 서비스뿐만 아니라 서비스의 로그인 계정이 administrator로 지정된 모든 서비스는 위와 같이 패스워드를 변경하고 해당 서비스를 재 시작해야 합니다.

■ 네트워크를 사용한 백업이 실패하는 경우

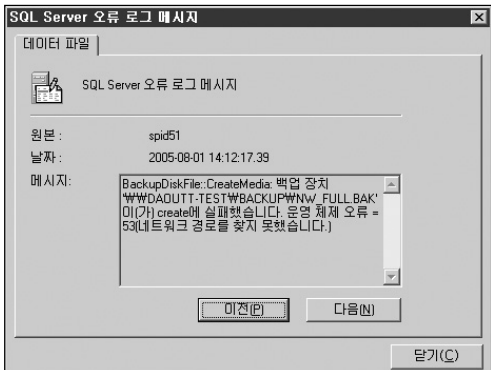
네트워크 드라이브로의 디스크 백업을 하는 경우 다음과 같은 오류 메시지가 발생하는 경우가 있습니다.



오류 메시지에서 지시하는 대로 SQL Server 오류 로그를 확인하면 운영 체제 오류로 인하여 액세스가 거부되었다는 메시지를 확인할 수 있습니다. 이와 같은 오류는 공유 설정이나 NTFS 파일 시스템의 권한 설정에서 비롯될 수 있습니다. 따라서, SQL Server 서비스의 로그인 계정이 백업 대상 파일 시스템과 공유에 적절한 권한이 있는지 확인합니다. 인증 실패가 원인일 수도 있습니다. 인증이 실패하는 경우에는 해당 파일서버에 동일한 계정의 존재 여부와 패스워드의 일치 여부를 확인해야 합니다. 로그인 계정이 로컬 시스템 계정인 경우에도 네트워크를 통해서 아무런 작업도 할 수 없습니다. 또한, 다음과 같이 네트워크 경로가 잘못 지정되어도 유사한 오류 메시지가 반환됩니다.



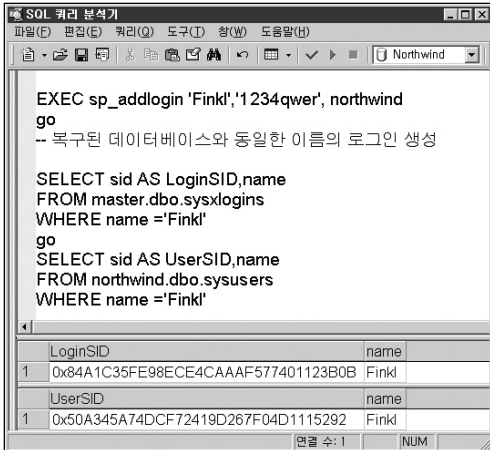
이 예제에서는 SQL Server 오류 로그에 네트워크 경로를 찾을 수 없다는 메시지를 확인할 수 있습니다.



■ 다른 서버로 데이터베이스 복구 후 Broken Login 문제

데이터베이스를 다른 서버로 복구한 경우에는 복구한 데이터베이스의 사용자 계정과 연결되는 SQL Server 로그인 계정이 없으므로 동일하게 로그인 계정을 생성해 주어야 합니다.

그러나, 기존의 서버에 있던 사용자 계정과 동일한 이름의 로그인 계정을 생성하더라도 해당 로그인 계정은 해당 데이터베이스에 접속할 수 있는 유효한 사용자 계정이 될 수 없습니다. SQL Server는 서버에 접속할 수 있는 로그인 계정(LoginID)과 각각의 데이터베이스에 접속할 수 있는 데이터베이스 사용자 계정(UserID)으로 2단계에 걸친 인증을 받기 때문에 로그인 계정이 있다고 하더라도 적절한 사용자 계정과 연결이 되지 않은 경우에는 SQL Server에 로그인하지 못합니다. 연결된 적절한 사용자 계정이 없는 경우에는 Guest 계정을 통해서 해당 데이터베이스에 연결할 수 있기는 하지만 Guest 계정의 사용은 보안상 취약하므로 사용자 데이터베이스에서는 사용을 삼가합니다. 2단계에 걸친 인증을 위한 로그인 계정과 각 데이터베이스의 사용자 계정은 로그인 계정을 만들 때 생성된 SID 값으로 연결되어 있습니다.

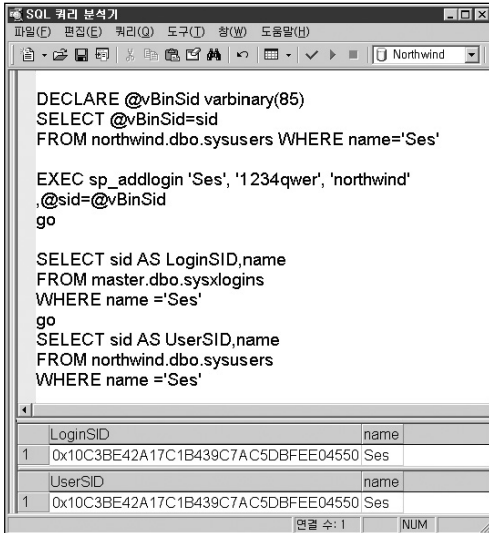


〈다른 서버로 복구된 데이터베이스의 사용자 계정〉

위의 그림에서 나타내는 바와 같이, 다른 서버로 복구된 데이터베이스의 사용자 계정에는 기존 서버에서 생성된 로그인 계정의 SID 정보를 sysusers 시스템 테이블에 저장하고 있어서, 이름만 같은 로그인 계정을 생성하더라도 SID가 다르므로 SQL Server는 전혀 별개의 로그인 계정으로 취급합니다.

이와 같은 경우에는 다음과 같은 방법으로 해결할 수 있습니다.

방법① 복구된 데이터베이스의 sysusers 시스템 테이블에서 해당 사용자 계정의 SID 정보를 지정하여 로그인 계정을 생성합니다.



The screenshot shows a window titled "SQL 쿼리 분석기" (SQL Query Analyzer) with a menu bar (파일(F), 편집(E), 쿼리(Q), 도구(T), 창(W), 도움말(H)) and a toolbar. The query text is as follows:

```
DECLARE @vBinSid varbinary(85)
SELECT @vBinSid=sid
FROM northwind.dbo.sysusers WHERE name='Ses'

EXEC sp_addlogin 'Ses', '1234qwer', 'northwind'
, @sid=@vBinSid
go

SELECT sid AS LoginSID,name
FROM master.dbo.sysxlogins
WHERE name = 'Ses'
go
SELECT sid AS UserSID,name
FROM northwind.dbo.sysusers
WHERE name = 'Ses'
```

Below the query text is a results grid with two sections. The first section shows the results of the first SELECT statement:

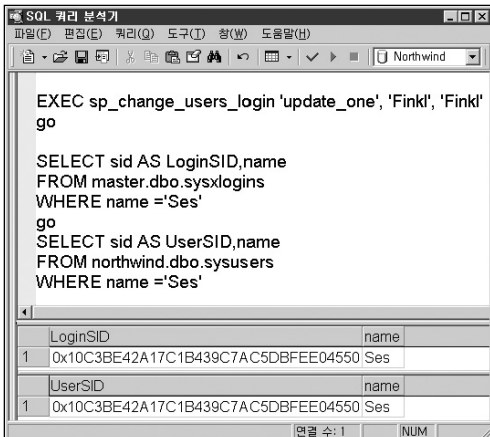
	LoginSID	name
1	0x10C3BE42A17C1B439C7AC5DBFEE04550	Ses

The second section shows the results of the second SELECT statement:

	UserSID	name
1	0x10C3BE42A17C1B439C7AC5DBFEE04550	Ses

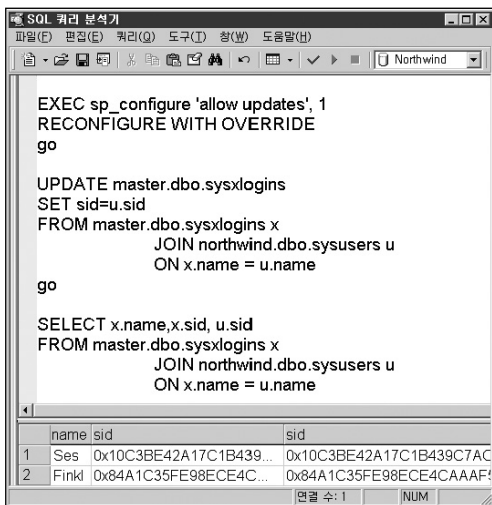
At the bottom right of the grid, it says "연결 수: 1" (Number of connections: 1) and "NUM".

방법② 사용자 계정과 동일한 이름의 로그인 계정을 생성하고 sp_change_users_login 시스템 저장 프로시저를 사용하여 SID 값을 동일하게 변경합니다. 유의할 사항은 이와 같이 조치하면, master.dbo.sysxlogins 시스템 테이블에 저장된 SID값으로 sysusers 시스템 테이블의 SID 컬럼의 값이 변경된다는 점입니다. 따라서, STANDBY 서버의 경우에는, STANDBY 데이터베이스가 읽기 전용 상태이므로 이 방법을 사용할 수 없습니다.



〈sp_change_users_login 시스템 저장프로시저를 사용하여 Login 계정의 SID로 사용자 계정의 SID를 변경한 경우〉

방법③ 사용자 계정과 동일한 이름의 로그인 계정을 생성하고 master 데이터베이스의 sysxlogins 시스템 테이블의 SID를 복구된 데이터베이스의 sysusers 시스템 테이블의 값으로 JOIN을 사용하여 수정합니다.



방법④ 마이크로소프트 기술 문서에서 제공하는 저장 프로시저를 생성하여 백업한 원본 서버의 SID와 패스워드를 추출하여 로그인을 생성합니다. 먼저 원본 서버의 master 데이터베이스에서 다음 두개의 저장 프로시저를 생성합니다. 이어서 sp_help_revlogin 저장 프로시저를 실행해서 결과창에 나타난 스크립트를 로그인을 옮기고자 하는 서버에서 실행합니다.

[관련 문서]

HOWTO: SQL Server 인스턴스 간의 로그인 및 암호 전송

<http://support.microsoft.com/KB/246133>

① sp_hexadecimal 저장 프로시저 생성 스크립트

```

----- Begin Script, Create sp_hexadecimal procedure -----
USE master
GO
IF OBJECT_ID ('sp_hexadecimal') IS NOT NULL
    DROP PROCEDURE sp_hexadecimal
GO
CREATE PROCEDURE sp_hexadecimal
    @binvalue varbinary(256),
    @hexvalue varchar(256) OUTPUT
AS
DECLARE @charvalue varchar(256)
DECLARE @i int
DECLARE @length int
DECLARE @hexstring char(16)
SELECT @charvalue = '0x'
SELECT @i = 1
SELECT @length = DATALENGTH (@binvalue)
SELECT @hexstring = '0123456789ABCDEF'
WHILE (@i <= @length)
BEGIN
    DECLARE @tempint int
    DECLARE @firstint int
    DECLARE @secondint int
    SELECT @tempint = CONVERT(int, SUBSTRING(@binvalue,@i,1))

```

```

SELECT @firstint = FLOOR(@tempint/16)
SELECT @secondint = @tempint - (@firstint*16)
SELECT @charvalue = @charvalue +
    SUBSTRING(@hexstring, @firstint+1, 1) +
    SUBSTRING(@hexstring, @secondint+1, 1)
SELECT @i = @i + 1
END
SELECT @hexvalue = @charvalue
GO
----- End Script , Create sp_hexadecimal procedure -----

```

② sp_help_revlogin 저장 프로시저 생성 스크립트

```

----- Begin Script, Create sp_help_revlogin procedure -----
USE master
GO
IF OBJECT_ID('sp_help_revlogin') IS NOT NULL
    DROP PROCEDURE sp_help_revlogin
GO
CREATE PROCEDURE sp_help_revlogin @login_name sysname = NULL AS
DECLARE @name sysname
DECLARE @xstatus int
DECLARE @binpwd varbinary(256)
DECLARE @txtpwd sysname
DECLARE @tmpstr varchar(256)
DECLARE @SID_varbinary varbinary(85)
DECLARE @SID_string varchar(256)

IF (@login_name IS NULL)

```



```

DECLARE login_curs CURSOR FOR
  SELECT sid, name, xstatus, password FROM master..sysxlogins
  WHERE srvid IS NULL AND name <> 'sa'
ELSE
  DECLARE login_curs CURSOR FOR
  SELECT sid, name, xstatus, password FROM master..sysxlogins
  WHERE srvid IS NULL AND name = @login_name
OPEN login_curs
FETCH NEXT FROM login_curs INTO @SID_varbinary, @name,
@xstatus, @binpwd
IF (@@fetch_status = -1)
BEGIN
  PRINT 'No login(s) found,'
  CLOSE login_curs
  DEALLOCATE login_curs
  RETURN -1
END
SET @tmpstr = '/* sp_help_revlogin script '
PRINT @tmpstr

SET @tmpstr = '** Generated '
+ CONVERT (varchar, GETDATE()) + ' on ' + @@SERVERNAME + ' */'
PRINT @tmpstr
PRINT ''
PRINT 'DECLARE @pwd sysname'
WHILE (@@fetch_status <> -1)
BEGIN
  IF (@@fetch_status <> -2)

```

```

BEGIN
PRINT ''
SET @tmpstr = '-- Login: ' + @name
PRINT @tmpstr
IF (@xstatus & 4) = 4
BEGIN -- NT authenticated account/group
IF (@xstatus & 1) = 1
BEGIN -- NT login is denied access
SET @tmpstr = 'EXEC master..sp_denylogin '' + @name + ''
PRINT @tmpstr
END
ELSE BEGIN -- NT login has access
SET @tmpstr = 'EXEC master..sp_grantlogin '' + @name + ''
PRINT @tmpstr
END
END
ELSE BEGIN -- SQL Server authentication
IF (@binpwd IS NOT NULL)
BEGIN -- Non-null password
EXEC sp_hexadecimal @binpwd, @txtpwd OUT
IF (@xstatus & 2048) = 2048
SET @tmpstr = 'SET @pwd = CONVERT (varchar(256), ' + @txtpwd + ')'
ELSE
SET @tmpstr = 'SET @pwd = CONVERT (varbinary(256), ' + @txtpwd + ')'
PRINT @tmpstr
EXEC sp_hexadecimal @SID_varbinary, @SID_string OUT
SET @tmpstr = 'EXEC master..sp_addlogin '' + @name
+ '', @pwd, @sid = ' + @SID_string + ', @encryptopt = '

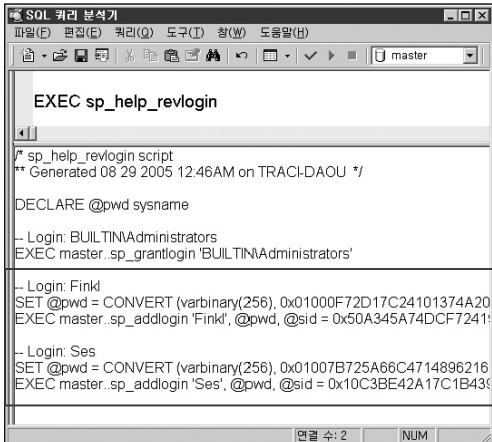
```

```

END
ELSE BEGIN
    -- Null password
    EXEC sp_hexadecimal @SID_varbinary,@SID_string OUT
    SET @tmpstr = 'EXEC master..sp_addlogin "' + @name
        + '"', NULL, @sid = ' + @SID_string + ', @encryptopt = '
    END
    IF (@xstatus & 2048) = 2048
        -- login upgraded from 6.5
        SET @tmpstr = @tmpstr + "'skip_encryption_old'"
    ELSE
        SET @tmpstr = @tmpstr + "'skip_encryption'"
    PRINT @tmpstr
    END
END
FETCH NEXT FROM login_curs INTO @SID_varbinary, @name,
    @xstatus, @binpwd
END
CLOSE login_curs
DEALLOCATE login_curs
RETURN 0
GO
----- End Script , Create sp_help_revlogin procedure -----

```

- ③ 원본 서버에서 sp_help_revlogin 저장 프로시저를 실행하여 다음과 같이 기존의 SID값과 패스워드가 동일한 로그인을 생성할 수 있는 스크립트를 작성하여 동일한 로그인을 생성하고자 하는 서버에서 실행합니다.



```
SQL 쿼리 분석기
파일(F) 편집(E) 쿼리(Q) 도구(T) 창(W) 도움말(H)
[Icons] master
EXEC sp_help_revlogin

/* sp_help_revlogin script
** Generated 08 29 2005 12:46AM on TRACI-DAOU */

DECLARE @pwd sysname

-- Login: BUILTIN\Administrators
EXEC master..sp_grantlogin 'BUILTIN\Administrators'

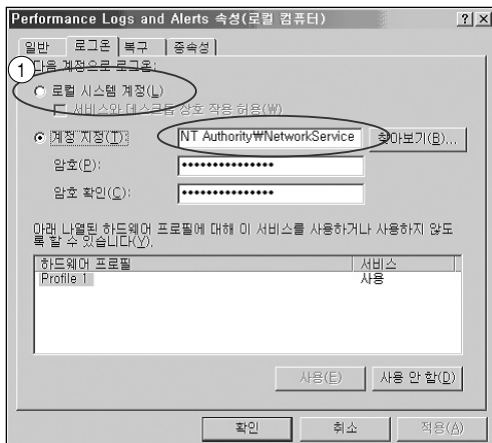
-- Login: Finkl
SET @pwd = CONVERT (varbinary(256), 0x01000F72D17C24101374A20
EXEC master..sp_addlogin 'Finkl', @pwd, @sid = 0x50A345A74DCF7241

-- Login: Ses
SET @pwd = CONVERT (varbinary(256), 0x01007B725A66C4714896216
EXEC master..sp_addlogin 'Ses', @pwd, @sid = 0x10C3BE42A17C1B439

연결 수: 2 NUM
```

■ 윈도우즈 2003 서버에서 SQL Server 성능 카운터가 로그에 기록되지 않는 경우

윈도우즈 2003 서버에서 운영되는 SQL Server 성능 모니터링을 위해서 시스템 모니터의 성능 로그를 수집하는 경우 디폴트 상태에서는 SQL Server의 성능 카운터가 기록되지 않습니다. 이런 문제가 발생하는 원인은 윈도우즈 2003 서버의 [Performance Logs and Alerts] 서비스의 로그인 계정이 기본적으로 다음과 같이 [NT Authority\NetworkService] 계정으로 지정되어 있기 때문입니다.



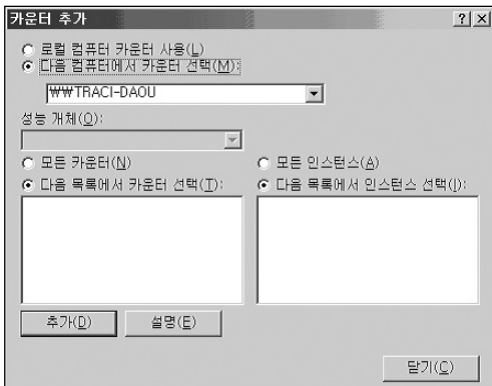
[NT Authority\NetworkService] 계정은 SQL Server의 카운터 로그를 시스템 모니터의 카운터 로그 파일(.blg)에 기록할 수 있는 권한을 가지고 있지 않습니다. 따라서, [Performance Logs and Alerts] 서비스의 로그인 계정을 그림①번의 [로컬 시스템 계정]을 선택하여 변경한 뒤 사용합니다.

[참조 문서]

[http://support.microsoft.com/default.aspx?scid=kb;en-us;839506\)](http://support.microsoft.com/default.aspx?scid=kb;en-us;839506)

■ SQL Server 성능 카운터가 시스템 모니터에서 보이지 않는 경우

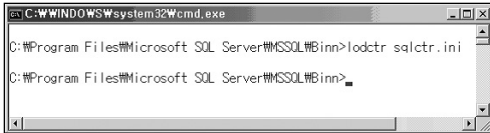
자주 발생하는 경우는 아니지만 시스템 모니터에서 SQL Server 성능 카운터가 보이지 않거나 다음과 같이 성능 카운터 전체가 보이지 않는 경우가 있습니다.



이런 경우 성능 카운터와 관련된 레지스트리 정보나 파일이 손상된 것이 원인인 경우가 많습니다. 다음의 순서로 확인합니다.

- ① SQL Server 성능 카운터가 보이지 않는 경우 관련 파일을 점검합니다.

"C:\Program Files\Microsoft SQL Server\MSSQL\Binn" 폴더의 sqlctr.ini 와 sqlctr80.dll 파일을 점검합니다. 이 파일이 손상된 경우 현재 설치된 SQL Server의 빌드 번호와 동일한 빌드 번호의 SQL Server 프로그램 CD나 서비스 팩 파일에서 복사합니다. sqlctr.ini 파일은 x86\bin 폴더에 sqlctr80.dll 파일은 x86\system 폴더에 있습니다. 그리고 sqlctr.ini 파일이 있는 "C:\Program Files\Microsoft SQL Server\MSSQL\Binn" 폴더로 이동하여 다음과 같이 lodctr 명령으로 성능 카운터를 로드하고 컴퓨터를 재시작 합니다.



- ② 윈도우즈 서버의 성능 카운터가 보이지 않는 경우 관련 파일을 점검합니다.

Winnt\system32 또는 Windows\system32 폴더의 Perf009.dat, Perfh009.dat 파일과 Perf012.dat, Perfh012.dat 파일을 점검합니다. Perf009.dat 파일이 손상된 경우에는 처음 그림과 같이 모든 성능 카운터가 나타나지 않게 됩니다. 009나 012는 해당 언어를 표시하며 012로 표시된 파일이 손상되면 성능 카운터 설명 등의 한글 표시가 되지 않습니다. 이와 같은 파일이 삭제되었거나 손상된 경우는 윈도우즈 서버 프로그램 CD의 i386 폴더의 Perf009.da_, Perfh009.da_, Perf012.da_, Perfh012.da_ 파일들을 다음과 같이 EXPAND 명령으로 압축을 풀어서 교체합니다.



- ③ 레지스트리 정보를 점검합니다.

HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Perflib에 009와 012 서브키가 존재하는지 확인합니다.

HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services에서 각 서비스 별로 해당 성능관련 서브키가 존재하는지 확인합니다.



<HKEY_LOCAL_MACHINE\Software\Microsoft\Windows NT\CurrentVersion\Perflib>



<HKEY_LOCAL_MACHINE\SYSTEM\CurrentControlSet\Services\MSSQLSERVER>

해당 서비스가 없거나 손상된 경우 시스템 상태 백업이나 레지스트리 편집기에서 내보내기 메뉴로 백업한 파일을 통해서 복원합니다. 시스템 상태 복원이나 레지스트리의 수정은 심각한 문제가 발생할 수 있으며 문제를 해결하기 위해 운영 체제를 다시 설치해야 할 수 있으므로 주의하여 수행해야 합니다.

사용된 명령어 정리

■ 저장 프로시저

저장 프로시저 이름	설 명
sp_readerrorlog	SQL Server 오류 로그를 반환합니다.
sp_cycle_errorlog	SQL Server를 재 시작하지 않고 새로운 오류 로그 파일만 생성합니다.
sp_helpserver	master.dbo.sys.servers 시스템 테이블에 등록된 정보를 반환합니다.
sp_dropserver	master.dbo.sys.servers 시스템 테이블에서 서버를 삭제합니다.
sp_addserver	master.dbo.sys.servers 시스템 테이블에 서버를 등록합니다.
sp_serveroption	master.dbo.sys.servers 시스템 테이블에서 등록된 서버의 옵션을 변경합니다.
sp_blocker_pss80	잠금 정보와 블로킹하는 프로세스와 블로킹 당하는 프로세스의 정보를 반환합니다. http://support.microsoft.com/default.aspx?scid=kb;en-us;299518
sp_tempdbspapce	tempdb의 공간 사용 정보를 반환합니다.
sp_configure	서버의 구성 옵션을 변경합니다.
sp_attach_db	데이터베이스를 SQL Server에 연결합니다.
sp_attach_single_file_db	로그 파일이 손실된 데이터베이스를 SQL Server에 연결합니다.
sp_detach_db	데이터베이스를 분리합니다.
sp_resetstatus	데이터베이스에서 주의 대상 플래그를 해제합니다.
sp_helpfile	현재 데이터베이스 파일의 물리적 이름 및 특성을 반환합니다.
sp_change_users_login	로그인 계정과 사용자 계정의 연결을 설정합니다.
sp_hexadecimal	Binary값이나 Decimal값을 16진수 형태의 varchar 타입으로 변경한 값을 반환합니다. http://support.microsoft.com/KB/246133
sp_help_revlogin	원본과 동일한 SID와 패스워드를 가지는 로그인을 생성하는 스크립트를 반환합니다. http://support.microsoft.com/KB/246133

■ DBCC 명령어

DBCC 명령	설 명
DBCC ERRORLOG	새로운 SQL Server 오류 로그를 생성합니다.
DBCC SHOW_STATISTICS	인덱스의 통계 정보를 보여 줍니다. 온라인 설명서 참조
DBCC SHOWCONTIG	인덱스의 단편화 정보를 보여 줍니다. 온라인 설명서 참조
DBCC DBREINDEX	인덱스를 재구성합니다. 온라인 설명서 참조
DBCC INDEXDEFRAG	인덱스의 페이지를 재정렬합니다. 온라인 설명서 참조
DBCC SQLPERF(WAITSTATS)	각 대기 유형별로 대기 시간을 확인합니다.
DBCC SQLPERF(LOGSPACE)	로그 파일의 사용 공간을 확인합니다. 온라인 설명서 참조
DBCC INPUTBUFFER	해당 프로세스의 명령문을 확인합니다. 온라인 설명서 참조
DBCC CHECKDB	지정한 데이터베이스에서 모든 개체의 할당과 구조적 무결성을 검사합니다. 온라인 설명서 참조
DBCC CHECKTABLE	지정한 테이블에서 할당과 구조적 무결성을 검사합니다. 온라인 설명서 참조
DBCC DBRECOVER	데이터베이스를 재시작하지 않고 복원합니다.
DBCC REBUILD_LOG	로그 파일 손상 시에 새로운 로그 파일을 생성합니다.
DBCC TRACE	추적 플래그를 설정합니다. 온라인 설명서 참조
DBCC TRACESTATUS	추적 플래그 설정 여부를 보여 줍니다. 온라인 설명서 참조

자세한 설명은 온라인 설명서와 SQL Server DBA 가이드를 참조하기 바랍니다.

■ 유틸리티

유틸리티 이름	설명
Portqry.exe	윈도уз 서버에서 사용중인 포트 상태를 점검합니다. http://support.microsoft.com/kb/310099
Componet Checker (cc_pkg.exe)	MDAC 버전 정보 및 파일 정보를 점검합니다. http://msdn.microsoft.com/data/mdac/downloads/default.aspx
SQLDiag.exe	SQL Server 진단 툴입니다. C:\Program Files\Microsoft SQL Server\MSSQL\Binn
Ostress.exe	커넥션 및 스트레스 테스트를 합니다. http://www.microsoft.com/downloads/details.aspx?FamilyId=5691AB53-893A-4AAF-B4A6-9A8BB9669A8B&displaylang=en
Rebuildm.exe	시스템 데이터베이스를 재구성합니다. C:\Program Files\Microsoft SQL Server\MSSQL\Binn
DTCPing.exe	MSDTC의 구성 정보와 상태 정보를 점검합니다. http://support.microsoft.com/default.aspx?scid=kb;ko;306843

■ 시작 옵션 및 추적 플래그

시작 옵션 및 매개 변수	설 명
/d	master 데이터베이스 데이터 파일의 위치를 지정합니다.
/l	master 데이터베이스 로그 파일의 위치를 지정합니다.
/e	SQL Server 오류 로그 파일의 위치를 지정합니다.
/f	최소 구성으로 시작합니다.
/m	단일 사용자 모드로 시작합니다.
/c	SQL Server를 서비스로 시작하지 않고 응용 프로그램으로 시작합니다.
/g	확장 저장 프로시저, OLE 자동화 개체, 분산 쿼리 등을 위한 메모리 공간을 MB단위로 지정합니다.
/x	CPU 시간과 캐시 적중률 통계를 유지할 수 없도록 합니다. 해당 정보를 모니터링할 필요가 없는 경우에 지정하면 성능이 다소 향상됩니다.
-T	추적 플래그를 설정합니다.
T1118	모든 데이터베이스에 유니폼 익스텐트만 할당합니다. (7.0에서는 인덱스 생성 시 등에 오류가 발생합니다.)
T1204	교착 상태를 모니터링하여 SQL Server 오류 로그에 기록합니다.
T1807	UCN 경로를 이용하여 네트워크 드라이브에 데이터 파일 및 로그 파일을 생성할 수 있습니다.
T2520	DBCC PAGE 등 문서화 되지 않은 DBCC 명령어의 매개 변수를 DBCC HELP를 통해서 확인 합니다.
T3604	DBCC 실행 결과를 화면에 출력합니다.
T3605	DBCC 실행 결과를 SQL Server 오류 로그에 기록합니다.
T3607	모든 데이터베이스에서 인스턴스 복구 프로세스를 생략합니다.
T3608	master 데이터베이스를 제외한 모든 데이터베이스에서 인스턴스 복구 프로세스의 실행을 생략합니다.
T3609	tempdb 생성을 생략합니다.
T4013	새로운 연결이 생성되는 경우 해당 정보를 SQL Server 오류 로그에 기록합니다.
T4220	Startup procedure의 실행을 비활성화합니다.
T7300	OLEDB의 보다 상세한 오류 메시지를 반환 받을 수 있습니다.
T8602	인덱스 힌트를 적용하지 않습니다.
T8755	잠금 힌트를 적용하지 않습니다.

마치면서

"이 세상에 진리는 존재하지 않는다, 객관성을 가장한 주관적 해석만이 존재할 뿐이다"라는 말이 있습니다. 지금은 최선의 방법이라고 생각할지 모르지만 하룻밤이 지나고 나면 더 적절한 방법을 발견하거나 추가적인 조치가 미비했음을 인지할 때가 있습니다. 점차 엔진이 발달하고 핫픽스가 개발되어짐에 따라서 어제의 방법이 더 이상 최선의 선택이 아닐 수도 있다는 사실을 유념하시기 바랍니다. 이 조그마한 지식을 바탕으로 끊임없이 발전하는 여러분을 기대합니다.

비매품

SQL Server Troubleshooting 가이드

- 저자 : 이종인
- 감수 : 하성희
- 기획 : 임무호 (주)다우데이타시스템
- Contents 관련문의 : jilee@daoudata.co.kr

- 발행 : 한국마이크로소프트(유)
- 제작 : (주)다우데이타시스템
- 초판 발행일 : 2005년 9월

본 책에 실린 글과 그림, 사진 및 프로그램 코드의 저작권 및 배포권은 한국마이크로소프트(유)와 (주) 다우데이타시스템에 있으며 저작권자의 동의 없이는 사용할 수 없습니다.

Microsoft®
SQL Server™
Troubleshooting 가이드

Microsoft®

한국마이크로소프트(유)

서울특별시 강남구 대치동 892번지 포스코센터 서관 5층

전화 : 080-985-2000

인터넷 : <http://www.microsoft.com/korea/sql>

SQL-200509-01