Vitamin Quiz 모범 답안

3장.

P102

SELECT prodName, company, amount FROM productTbl WHERE cost >= 8 ;

P109

CREATE VIEW uv_over10
AS
SELECT * FROM productTbl WHERE cost >= 10;

P111

CREATE PROCECURE over10Proc
AS
SELECT * FROM productTbl WHERE cost >= 10;

P122

디스크가 꽉 찰 때까지 파일 크기가 늘어나도록 지정한다. SQL Server 2008에서 무제한 증가가 허용된 로그 파일의 최대 크기는 2TB(테라바이트)이고 데이터 파일의 최대 크기는 16TB(테라바이트)이다.

P133

직접 3장 <실습 11>을 참조해서 수행

4장

P144

프로토타입 모델(Prototype model): 요구된 기능, 성능 등에 대한 핵심적인 사항들을 먼저 프로토타입으로 모형화하고 이를 분석한 결과를 토대로 소프트웨어를 개발하는 모델

점증적 모델(Incremental model): 소프트웨어를 두개 이상의 build, version, 또는release들로 나누어 이들을 하나씩 점차적으로 개발해 나감으로써 최종 release가 완성된 소프트웨어가 되도록 하는 모델

나선형 모델(Spiral model): 요구분석, 설계, 구현, 통합, 시험 등의 각 작업을 리스크의 분석과 해소 방안을 수립하여 수행하는 모델

혼합형 모델(Hybrid model): 두가지 이상의 프로세스 모델들을 혼합한 모델

6장

P223

```
SELECT * FROM userTbl
  WHERE height = ANY (SELECT height FROM userTbl WHERE addr = '서울' AND mobile1 = '011')
```

P225

```
SELECT DISTINCT mobile1 FROM userTbl WHERE mobile1 IS NOT NULL ORDER BY mobile1
```

P232

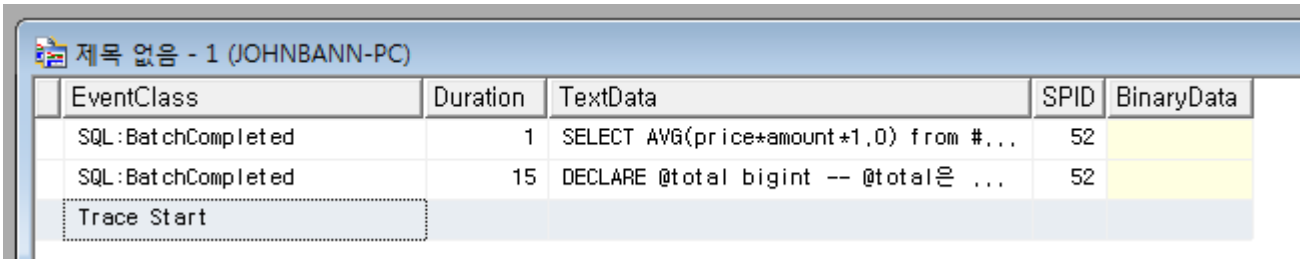
```
SELECT userid AS [사용자아이디], SUM(price*amount*1.0) / COUNT(userid) AS [평균구매액]
  FROM buyTbl GROUP BY userid
```

P239

SELECT AVG(price*amount*1.0) from #tmpTbl

```
DECLARE @total bigint -- @total은 총구매액을 누적시킬변수
DECLARE @priceXamount int -- @priceXamount는 각행의 가격과 수량을 곱한값을 저장할 변수
DECLARE @num int -- @num 은구매테이블의 순번열의 증가
DECLARE @count int -- @count는 전체행 숫자
SET @total=0
SET @num=1
SELECT @count=COUNT(*) FROM #tmpTbl
SET @count = @count + 1
WHILE @count <> @num
BEGIN
        SELECT @priceXamount = price * amount FROM  #tmpTbl
                WHERE num = @num
   SET @num = @num + 1
   SET @total = @total + @priceXamount
END
```

PRINT @total / (@count - 1.0)


제목 없음 - 1 (JOHNBANN-PC)

EventClass	Duration	TextData	SPID	BinaryData
SQL:BatchCompleted	1	SELECT AVG(price*amount*1.0) from #...	52	
SQL:BatchCompleted	15	DECLARE @total bigint -- @total은 ...	52	
Trace Start				

P248

```
WITH cte_buyTbl(userid, maxPurchase)
AS
  ( SELECT userid, MAX(price*amount) FROM buyTbl GROUP BY userid)
SELECT AVG(maxPurchase*1.0) AS [각 사용자별 최고 구매액의 평균] FROM cte_buyTbl
```

P272

```
SELECT price, amount, CONVERT(DECIMAL(10,2), CONVERT(FLOAT, price)/amount)
        AS [단가/수량] FROM buyTbl
```

P278

```
UPDATE maxTbl SET
        col1 = STUFF( (SELECT col1 FROM maxTbl), 1, 1000000,
                (SELECT REVERSE((SELECT col1 FROM maxTbl)))),
        col2 = STUFF( (SELECT col2 FROM maxTbl), 1, 1000000,
                (SELECT REVERSE((SELECT col2 FROM maxTbl))))
```

P283

```
USE AdventureWorks
SELECT ROW_NUMBER( ) OVER(ORDER BY HireDate ASC) [입사 순위],
  LoginID, JobTitle, HireDate
FROM HumanResources.Employee
ORDER BY HireDate ASC

SELECT ROW_NUMBER( ) OVER(ORDER BY HireDate ASC, LoginID ASC)[입사 순위],
  LoginID, JobTitle, HireDate
FROM HumanResources.Employee
ORDER BY HireDate ASC

SELECT JobTitle,
```

```
 ROW_NUMBER( ) OVER( PARTITION BY JobTitle
     ORDER BY HireDate ASC, LoginID ASC)[직종별 입사 순위],
  LoginID,  HireDate
FROM HumanResources.Employee
ORDER BY JobTitle, HireDate ASC

SELECT DENSE_RANK( ) OVER(ORDER BY HireDate ASC)[입사 순위],
  LoginID, JobTitle, HireDate
FROM HumanResources.Employee
ORDER BY HireDate ASC

SELECT RANK( ) OVER(ORDER BY HireDate ASC)[입사 순위],
  LoginID, JobTitle, HireDate
FROM HumanResources.Employee
ORDER BY HireDate ASC

SELECT NTILE(2) OVER(ORDER BY HireDate ASC) [반번호],
  LoginID, JobTitle, HireDate
FROM HumanResources.Employee
ORDER BY HireDate ASC

SELECT NTILE(4) OVER(ORDER BY HireDate ASC) [반번호],
  LoginID, JobTitle, HireDate
FROM HumanResources.Employee
ORDER BY HireDate ASC
```

P290

1쇄 문제 수정 : INNER JOIN을 사용해서, AdventureWorks의 HumanResources.Employee와 Person.Person을 조인해 보도록 하자.

```
USE AdventureWorks
SELECT E.BusinessEntityID, P.FirstName, P.MiddleName, P.LastName
FROM HumanResources.Employee E
  INNER JOIN Person.Person P
     ON E.BusinessEntityID = P.BusinessEntityID
```

P293

```
SELECT S.stdName, S.addr, C.clubName, C.roomNo
FROM stdTbl S , stdclubTbl SC, clubTbl C
```

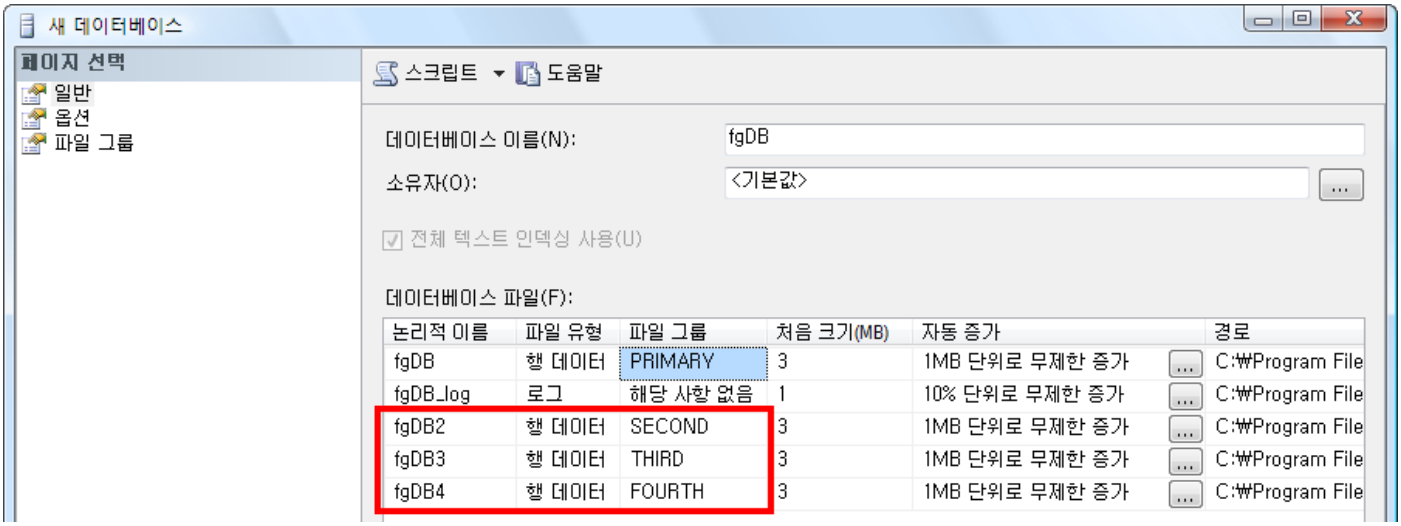
```
        WHERE S.stdName = SC.stdName AND SC.clubName = C.clubName
ORDER BY S.stdName

SELECT C.clubName, C.roomNo, S.stdName, S.addr
FROM  stdTbl S, stdclubTbl SC, clubTbl C
  WHERE SC.stdName = S.stdName AND SC.clubName = C.clubName
ORDER BY C.clubName
```

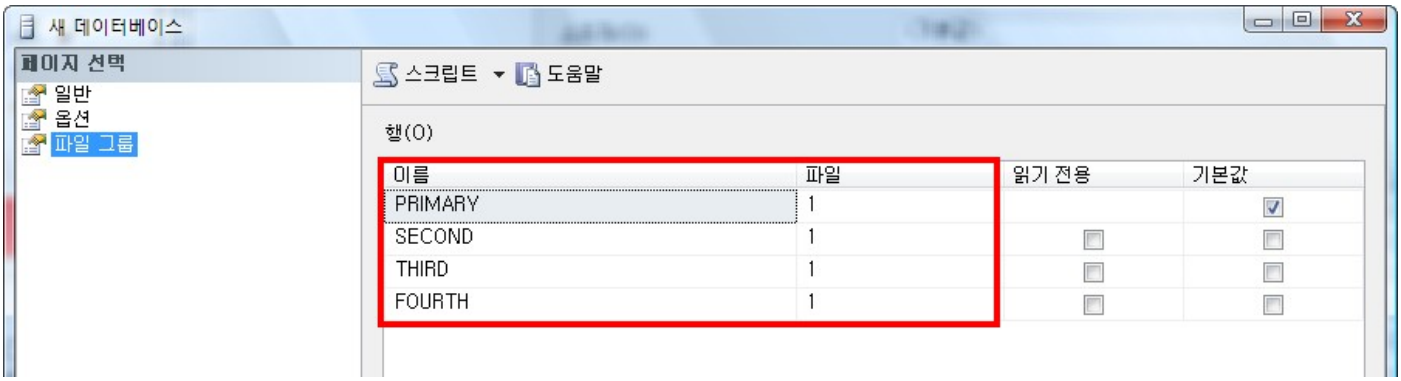
7장

P331

동일하게 수행됨.

P353





P387

전제 조건

- 데이터베이스 이름 : moveDB
- 파일 그룹 및 데이터 파일 : Primary(moveDB1.mdf, moveDB1.mdf) , Secondary(moveDB3.mdf, moveDB4.mdf)
- 폴더 이동 (폴더는 기존에 존재해야 함) : C:₩BEFORE₩ ➔ C:₩AFTER₩

```
CREATE DATABASE  moveDB
ON  PRIMARY
( NAME = moveDB1,
FILENAME = N'C:₩BEFORE₩moveDB1.mdf' ),
( NAME = moveDB2,
FILENAME = N'C:₩BEFORE₩moveDB2.mdf'),
FILEGROUP [SECONDARY]
( NAME = moveDB3,
FILENAME = N'C:₩BEFORE₩moveDB3.mdf'),
( NAME = moveDB4,
FILENAME = N'C:₩BEFORE₩moveDB4.mdf')
 LOG ON
( NAME = moveDB_log, -- 로그 파일
FILENAME = N'C:₩BEFORE₩moveDB_log.ldf')
GO

ALTER DATABASE moveDB
        MODIFY FILE
                ( NAME = moveDB1,
                  FILENAME = 'C:₩AFTER₩moveDB1.mdf' )
ALTER DATABASE moveDB
        MODIFY FILE
                ( NAME = moveDB2,
                  FILENAME = 'C:₩AFTER₩moveDB2.mdf' )
ALTER DATABASE moveDB
        MODIFY FILE
                ( NAME = moveDB3,
                  FILENAME = 'C:₩AFTER₩moveDB3.mdf' )
ALTER DATABASE moveDB
        MODIFY FILE
                ( NAME = moveDB4,
                  FILENAME = 'C:₩AFTER₩moveDB4.mdf' )
```

```
ALTER DATABASE moveDB
        MODIFY FILE
                ( NAME = moveDB_log, -- 데이터 파일
                  FILENAME = 'C:₩AFTER₩moveDB_log.ldf' )
GO

ALTER DATABASE moveDB SET OFFLINE;
GO
-- C:₩BEFORE₩의 파일을 C:₩AFTER₩로 옮겨야 함.

ALTER DATABASE moveDB SET ONLINE
GO
```

8장

P406

먼저 현재 구매 테이블의 ID값을 확인

```
USE tableDB;
DBCC CHECKIDENT('buyTbl');
```

순번을 원하는 번호로 입력 (예로 다음에 10번부터 들어가게 하려면)

```
DBCC CHECKIDENT('buyTbl', RESEED, 9);
```

P410

1쇄 문제 수정 : 앞 <실습2>의 3번에서 데이터를 마저 입력해 보도록 하자.

[그림 8-1]을 보고 먼저 회원테이블(userTbl)의 데이터를 모두 입력한 후에, 구매테이블(buyTbl)을 입력

P414

개체탐색기에서 해당 테이블을 선택한 후, 마우스 오른쪽 버튼을 클릭하고 [디자인]을 선택 □

Ctrl 키를 누른 상태에서 여러 개의 열을 선택한 후에, 마우스 오른쪽 버튼을 클릭하고 [기본 키 설정]을 선택하면 한 번에 여러 개의 열이 기본 키로 설정됨.

P419

제약조건의 내용이 충돌되지 않는다면 여러 개의 체크제약조건이 설정 가능.

예로 키가 0~200으로 설정하기 위해서 두 제약조건을 사용할 수 있음.

```
USE sqlDB;
-- 제약조건 1
```

```
ALTER TABLE userTbl
        ADD CONSTRAINT CK_height
        CHECK
    (height >= 0)
GO
-- 제약조건 2
ALTER TABLE userTbl
        ADD CONSTRAINT CK_height2
        CHECK
    (height <= 200)
```

위처럼 두 개의 체크 제약조건은 가능하지만, 추가로 아래의 제약조건 3은 오류가 발생됨.

```
-- 제약조건 3
ALTER TABLE userTbl
        ADD CONSTRAINT CK_height3
        CHECK
    (height <= 100)
```

이미 제약조건1,2번에 의해서 0~200의 조건이지만, 추가로 100이하를 지정하는 것이므로 충돌됨.

P428

(1) 쿼리창을 열고 임시 테이블 생성.

```
CREATE TABLE ##imsiTbl(id int);
CREATE TABLE #imsiTbl(id int);
```

(2) 새 쿼리창(세션)을 열고 아래와 같이 수행하면

```
USE tempDB;
DROP TABLE ##imsiTbl; ➔ 성공 (전역 테이블이므로 테이블을 삭제 가능)
DROP TABLE #imsiTbl; ➔ 실패 (지역 테이블이므로 테이블 존재를 다른 세션에서는 알수 없음)
```

P432

오류가 발생됨.

예제)

```
USE tempDB;
CREATE TABLE test1 ( a NCHAR(10));
INSERT INTO test1 VALUES ('뇌를자극하는시리즈');
```

```
ALTER TABLE test1
        ALTER COLUMN a NCHAR(5);
---------오류 메시지--------
메시지 8152, 수준 16, 상태 13, 줄 1
문자열이나 이진 데이터는 잘립니다.
문이 종료되었습니다.
```

P453

새로운 로그인과 사용자를 생성한 후에, 그 사용자의 디폴트 스키마를 HumanResources로 지정하면 됨.
(해당 하는 상세한 내용은 16장 설명됨.)

P456

THIRD 인스턴스를 먼저 실행한 후에,

```
EXEC sp_addlinkedserver
   @server='THIRD',
   @srvproduct='SQLServer',
    @provider='SQLNCLI',
   @datasrc='BRAIN₩THIRD'
```

P459

- 일반 쿼리문보다 속도가 아주 약간 느려질 수 있다. (하지만, 거의 차이는 없는 걸로 봐도 무방하다.)
- 별도로 뷰를 관리해야 하므로, 관리차원에서도 약간 부담이 늘어난다.

P465

(1) 뷰와 관련있는 개체 카탈로그 뷰 : sys.objects, sys.tables, sys.views
(2) sys.sql_dependencies 카탈로드 뷰는 온라인 도움말 참조

9장

P493

(1) Unique 만 설정하면 비클러스터형 인덱스로 생성됨.

```
CREATE TABLE  tbl6
        (       a INT UNIQUE  ,
                b INT ,
                c INT ,
                d INT
```

)

GO

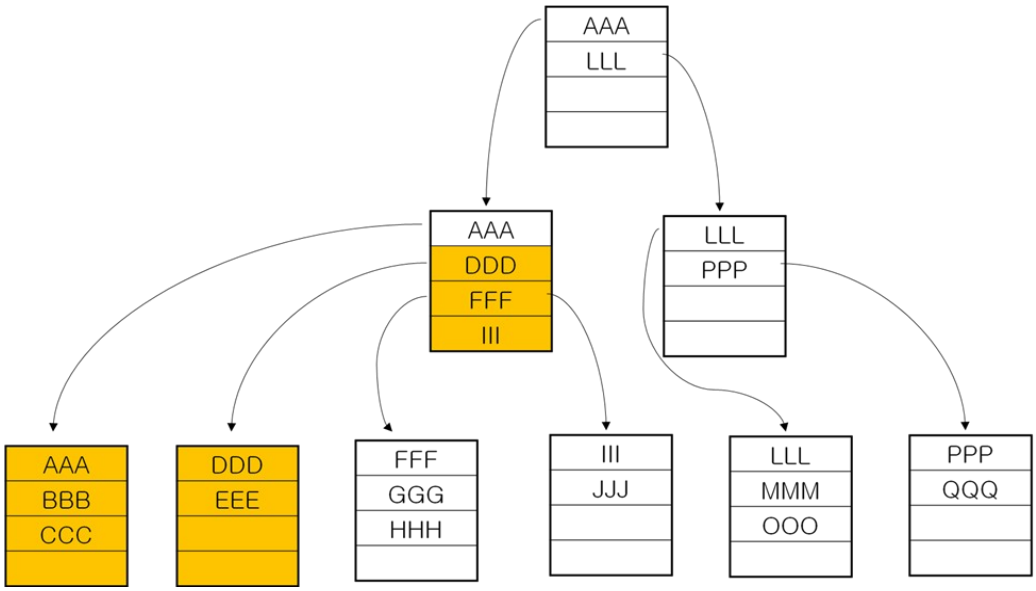
EXEC sp_helpindex tbl6;

결과 메시지

	index_name	index_description	index_keys
1	UQ__tbl6__3BD0198F15502E78	nonclustered, unique, unique key located on PRIMA...	a

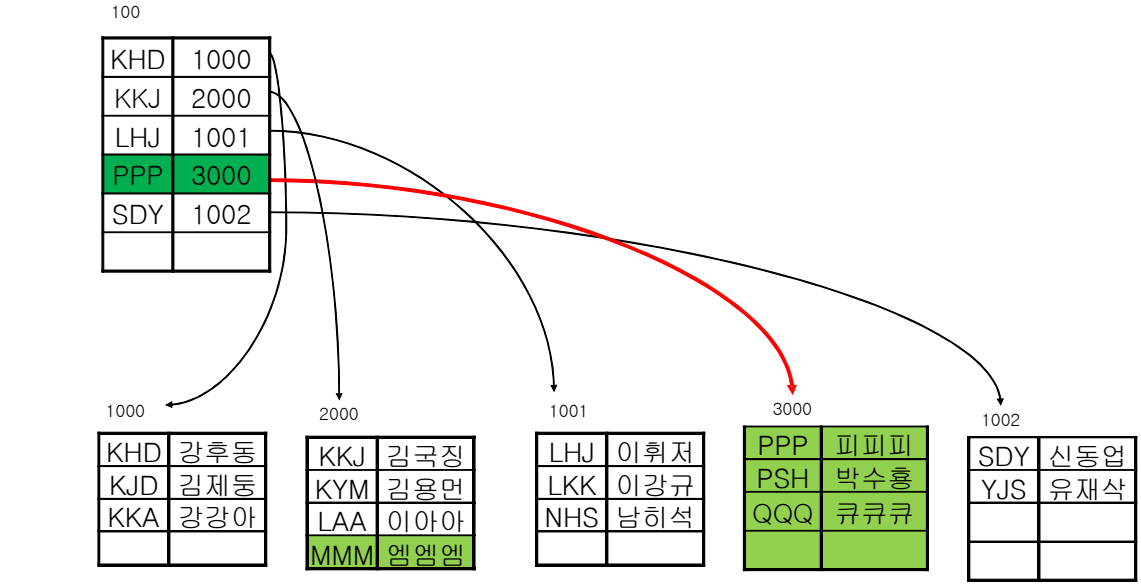
(2) Primary Key, Unique 외의 제약조건은 Clustered, Nonclustered 를 붙일 수 없음.

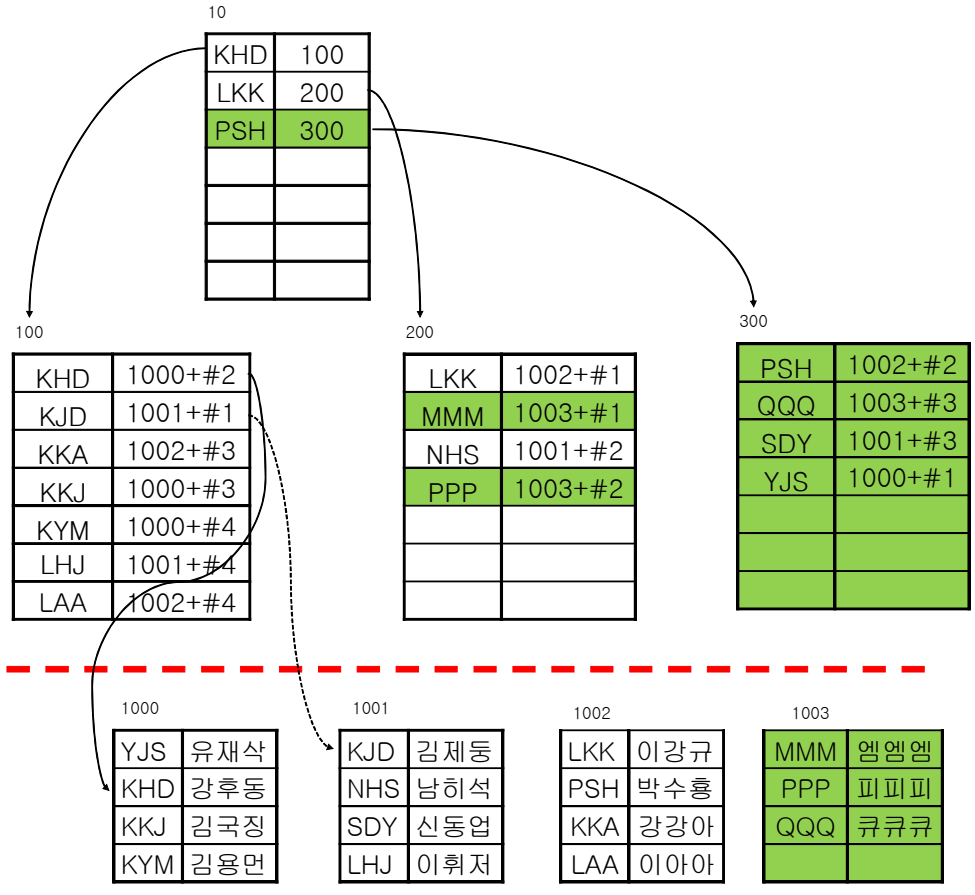
P497



P503

MMM(엠엠엠), PPP(피피피), QQQ(큐큐큐) 3개가 입력된다고 가정하면





P528

[그림 9-40]을 보면

우선, 100을 찾기 위해 루트페이지, 001페이지를 읽고
또, 200을 찾기 위해 루트페이지, 002페이지를 읽고
또, 300을 찾기 위해 루트페이지, 002페이지를 읽었으므로
검색은 3회가 되고, 각 검색당 2페이지씩 읽었으므로 총 6페이지를 읽은 것이다.

P544

*** 포괄열이 있는 인덱스 생성 이전**

```
USE indexDB;
Select * INTO Cust_Include from AdventureWorks.Sales.Customer ORDER BY rowguid;
EXEC sp_IndexInfo Cust_Include;
```

결과 메시지

	테이블이름	인덱스이름	인덱스타입	페이지개수	크기(KB)	행개수
1	Cust_Include	NULL	HEAP	156	1248	19820

*** 포괄열이 있는 인덱스 생성 이후**

```
CREATE NONCLUSTERED INDEX idx_cust_include
                    ON Cust_Include (CustomerID)
                    INCLUDE (AccountNumber,TerritoryID);
EXEC sp_IndexInfo Cust_Include;
```

결과 메시지

	테이블이름	인덱스이름	인덱스타입	페이지개수	크기(KB)	행개수
1	Cust_Include	NULL	HEAP	156	1248	19820
2	Cust_Include	idx_cust_include	NONCLUSTERED	89	712	19820

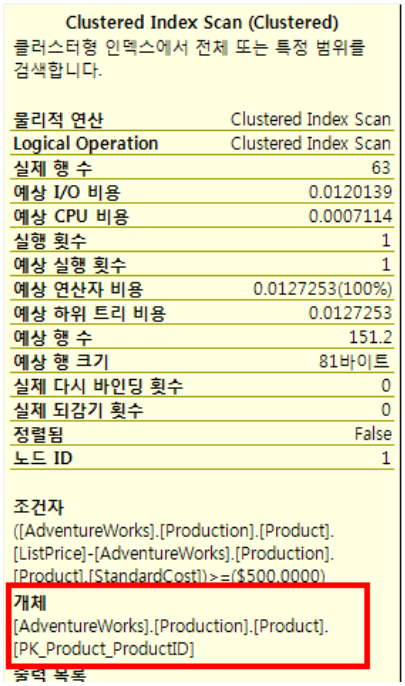
P551

```
CREATE VIEW vProfit WITH SCHEMABINDING  AS
        SELECT ProductID, Name , (ListPrice - StandardCost) AS Profit FROM Production.Product
        WHERE (ListPrice - StandardCost) >= 1000;
GO
CREATE UNIQUE CLUSTERED INDEX vIdx_vProfit ON vProfit (ProductID);
```

(1) SELECT ProductID,Name,(ListPrice - StandardCost) FROM Production.Product

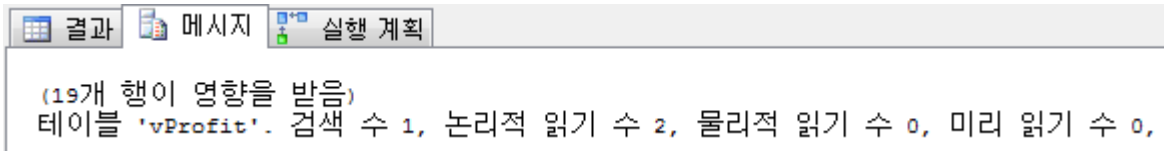
WHERE (ListPrice - standardcost) >= 500

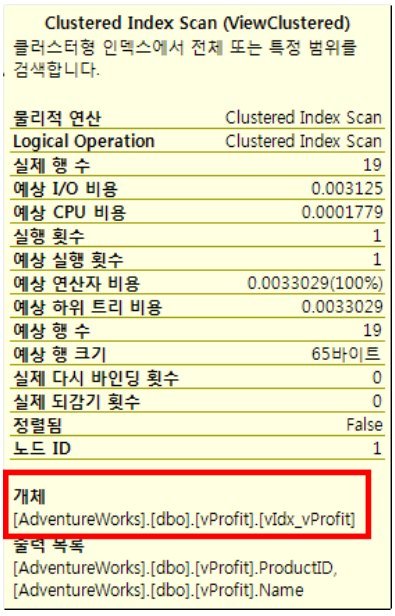
⇨ **500~999 까지는 인덱싱된 뷰가 생성이 안되어 있으므로, 인덱싱된 뷰 사용하지 않음**



(2) SELECT ProductID, Name FROM Production.Product WHERE (ListPrice - StandardCost) >= 1000

⇨ **인덱싱된 뷰를 사용하기 때문에 2페이지만 읽음**





10장

P571

```
BEGIN TRY
	BEGIN TRAN
		UPDATE bankBook SET money = money - 600 WHERE NAME = N'존밴이';
		UPDATE bankBook SET money = money - 10  WHERE NAME = N'존밴이'; -- 수수료
		UPDATE bankBook SET money = money + 600 WHERE NAME = N'당탕이';
	COMMIT TRAN
END TRY
BEGIN CATCH
	PRINT N'잔고가 부족합니다. 계좌를 확인해 보세요'
	ROLLBACK TRAN
END CATCH;
```

P595

P591의 잠금힌트를 사용해서 직접 실습

P601

<실습8>의 1번의 쿼리창1에서

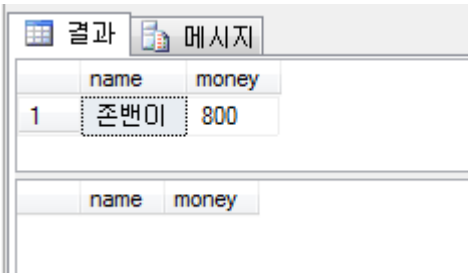
```
SET DEADLOCK_PRIORITY HIGH
```

를 설정하고 다시 실습하면 쿼리창2의 쿼리문이 희생됨.

P606

```
-- SECOND은행에서 통장 테이블만 개설하고 당탕이 계좌는 입력 하지 않음.
USE tempDB;
CREATE TABLE bankBook
	(name NVARCHAR(10),
	 money INT ,
	 CONSTRAINT CK_money
	 CHECK  (money >= 0)
 )
GO

-- 기본 은행에서 다시 테이블 생성 후, 존밴이 통장 개설
USE tempDB;
```

```
DROP TABLE bankbook

CREATE TABLE bankBook
        (name NVARCHAR(10),
         money INT ,
         CONSTRAINT CK_money
         CHECK  (money >= 0)
 )
GO
INSERT INTO bankBook VALUES (N'존밴이', 1000);

-- 기본 은행에서 SECONDBANK에 '당탕이'의 계좌 자체가 없는 상태에서 아래를 실행하면,
BEGIN DISTRIBUTED TRANSACTION
        UPDATE bankBook SET money = money - 200 WHERE NAME = '존밴이';
        UPDATE SECONDBANK.tempDB.dbo.bankBook SET money = money + 200
                WHERE NAME = '당탕이';
COMMIT TRANSACTION

SELECT * FROM bankBook;
SELECT * FROM SECONDBANK.tempDB.dbo.bankBook;
```



'존밴이'의 계좌에서는 돈이 빠지고 '당탕이'는 아예 계좌가 없으므로 당연히 돈이 아예 안 들어 간다. 결국 200원이 공중에서 사라졌다. 이를 해결하기 위해서는 아래와 같이 수정할 수 있다.

```
BEGIN TRY
        BEGIN DISTRIBUTED TRANSACTION
                UPDATE bankBook SET money = money - 200 WHERE NAME = '존밴이';
                IF ((SELECT COUNT(*) FROM SECONDBANK.tempDB.dbo.bankBook
                        WHERE  NAME = '당탕이') >= 1)
                        UPDATE SECONDBANK.tempDB.dbo.bankBook
                                SET money = money + 200 WHERE NAME = '당탕이';
                ELSE
```

```
                    RAISERROR('',16,1)
        COMMIT TRANSACTION
END TRY
BEGIN CATCH
        ROLLBACK TRANSACTION
END CATCH;
```

11장

P629

(1) sp_pkeys는 PRIMARY KEY 제약 조건을 사용하여 명시적으로 정의된 열에 대한 정보를 반환한다.

```
EXEC sp_pkeys @table_name = N'buyTbl', @table_owner = N'dbo'
```

(2) 기본 키 테이블 이름은 제공되고 외래 키 테이블 이름이 NULL인 경우 sp_fkeys는 외래 키를 포함한 모든 테이블을 지정된 테이블로 반환한다.

```
EXEC sp_fkeys @pktable_name = N'userTbl', @pktable_owner = N'dbo'
```

sp_tables 및 sp_databases 는 도움말을 참조할 것.

12장

P671

(1) KEYSET

```
USE master ;
RESTORE DATABASE sqlDB FROM DISK ='c:₩sqldb2008.bak' WITH REPLACE ;
USE sqlDB;

DECLARE userTbl_cursor1 CURSOR GLOBAL KEYSET
  FOR SELECT * FROM userTbl;

OPEN userTbl_cursor1;

FETCH NEXT FROM userTbl_cursor1;

UPDATE userTbl SET addr = N'미국';
FETCH NEXT FROM userTbl_cursor1; -- 변경된 '미국'으로 주소가 보임

INSERT INTO userTbl VALUES('ZZZ',N'마징가',1960,N'일본','000','0000000','999',NULL);
```

```
FETCH NEXT FROM userTbl_cursor1; -- 이 부분을 계속 반복해도 '마징가'는 안보임

CLOSE userTbl_cursor1;
DEALLOCATE userTbl_cursor1;
```

(2) DYNAMIC

```
USE master ;
RESTORE DATABASE sqlDB FROM DISK ='c:₩sqldb2008.bak' WITH REPLACE ;
USE sqlDB;

DECLARE userTbl_cursor2 CURSOR GLOBAL DYNAMIC
  FOR SELECT * FROM userTbl;

OPEN userTbl_cursor2;

FETCH NEXT FROM userTbl_cursor2;

UPDATE userTbl SET addr = N'미국'; -- 변경된 '미국'으로 주소가 보임
FETCH NEXT FROM userTbl_cursor2;

INSERT INTO userTbl VALUES('ZZZ',N'마징가',1960,N'일본','000','0000000','999',NULL);

FETCH NEXT FROM userTbl_cursor2; -- 이 부분을 계속 반복하면 '마징가'가 마지막에 보임

CLOSE userTbl_cursor2;
DEALLOCATE userTbl_cursor2;
```

P676

데이터의 인출이 확인 됨.

```
USE master ;
RESTORE DATABASE sqlDB FROM DISK ='c:₩sqldb2008.bak' WITH REPLACE ;
USE sqlDB;

CREATE TABLE keysetTbl(id INT, txt CHAR(5));
INSERT INTO  keysetTbl VALUES(1,'AAA');
```

```
INSERT INTO  keysetTbl VALUES(2,'BBB');
INSERT INTO  keysetTbl VALUES(3,'CCC');

DECLARE keysetTbl_cursor CURSOR GLOBAL FORWARD_ONLY  DYNAMIC TYPE_WARNING
FOR SELECT id, txt FROM keysetTbl;

DECLARE @result CURSOR
EXEC sp_describe_cursor @cursor_return = @result OUTPUT,
        @cursor_source = N'GLOBAL', @cursor_identity = N'keysetTbl_cursor'
FETCH NEXT from @result
WHILE (@@FETCH_STATUS <> -1)
    FETCH NEXT FROM @result

OPEN keysetTbl_cursor;
FETCH NEXT FROM keysetTbl_cursor;

INSERT INTO keysetTbl VALUES(4,'XXX');
FETCH NEXT FROM keysetTbl_cursor;  -- 여러 번 실행하면 'XXX'가 보임.
```

13장

P692

(1) <실습3> 에서 uv_devliver 뷰를 삭제 했다면, 다시 <실습3>의 0 ~ 2까지를 수행한다.

(2) UPDATE uv_deliver SET userid = 'DDY' WHERE userid='DTI'; -- 작동 실패(제약조건으로 인함)

```
-- 오류 메시지 --
메시지 547, 수준 16, 상태 0, 줄 1
UPDATE 문이 FOREIGN KEY 제약 조건 "FK__buyTbl__userid__0519C6AF"과(와) 충돌했습니다. 데이터베이스 "sqlDB", 테이블 "dbo.userTbl", column 'userID'에서 충돌이 발생했습니다.
문이 종료되었습니다.
```

```
CREATE TRIGGER trg_update
ON uv_deliver
INSTEAD OF UPDATE
AS
BEGIN
        -- 제약조건을 잠깐 작동중지시킴
```

```
        ALTER TABLE buyTbl NOCHECK CONSTRAINT FK__buyTbl__userid__0519C6AF

        UPDATE userTbl SET userid = (SELECT userid FROM inserted)
                WHERE userid= (SELECT userid FROM deleted)

        UPDATE buyTbl SET userid = (SELECT userid FROM inserted)
                WHERE userid= (SELECT userid FROM deleted)

    -- 제약조건을 다시 활성화시킴
    ALTER TABLE buyTbl CHECK CONSTRAINT FK__buyTbl__userid__0519C6AF
END;

UPDATE uv_deliver SET userid = 'JVI' WHERE userid='JBI'; -- 작동 성공
```

P711

```
USE triggerDB;
SELECT name, is_recursive_triggers_on FROM sys.databases
        WHERE name = 'triggerDB'; -- 0으로 나옴

ALTER DATABASE triggerDB
        SET RECURSIVE_TRIGGERS ON;

SELECT name, is_recursive_triggers_on FROM sys.databases
        WHERE name = 'triggerDB'; -- 1 로 나옴

CREATE TRIGGER trg_recuAA
ON recuAA
AFTER  INSERT
AS
        IF ( (SELECT trigger_nestlevel() ) >= 32 )
                RETURN
        DECLARE @id INT
        SELECT @id = trigger_nestlevel() -- 현재 트리거 레벨값
        PRINT '트리거레벨==> ' + CAST(@id AS CHAR(5))
        INSERT INTO recuAA VALUES ('직접재귀트리거')
GO

INSERT INTO recuAA VALUES ('처음입력값');
```

SELECT * FROM **recuAA**;

14장

P740

P733을 참조해서 직접 '중지 단어'를 추가해서 실습

15장

P755

SELECT * FROM indexXmlTbl
WHERE xmlInfo.exist('/row[@LastName="Kim"]') = 1; ➔ 인덱스 사용함

SELECT * FROM noIndexXmlTbl
WHERE xmlInfo.exist('/row[@LastName="Kim"]') = 1; ➔ 인덱스 사용 안함

16장

P808

```
USE master;
RESTORE DATABASE sqlDB FROM DISK = 'C:₩sqlDB2008.bak' WITH REPLACE

CREATE LOGIN auditUser
WITH    PASSWORD = 'password',
                DEFAULT_DATABASE = [sqlDB],
                CHECK_POLICY = OFF,
                CHECK_EXPIRATION = OFF;
GO
CREATE LOGIN badUser
WITH    PASSWORD = 'password',
                DEFAULT_DATABASE = [sqlDB],
                CHECK_POLICY = OFF,
                CHECK_EXPIRATION = OFF;
GO
USE sqlDB;
CREATE USER auditUser FOR LOGIN auditUser;
```

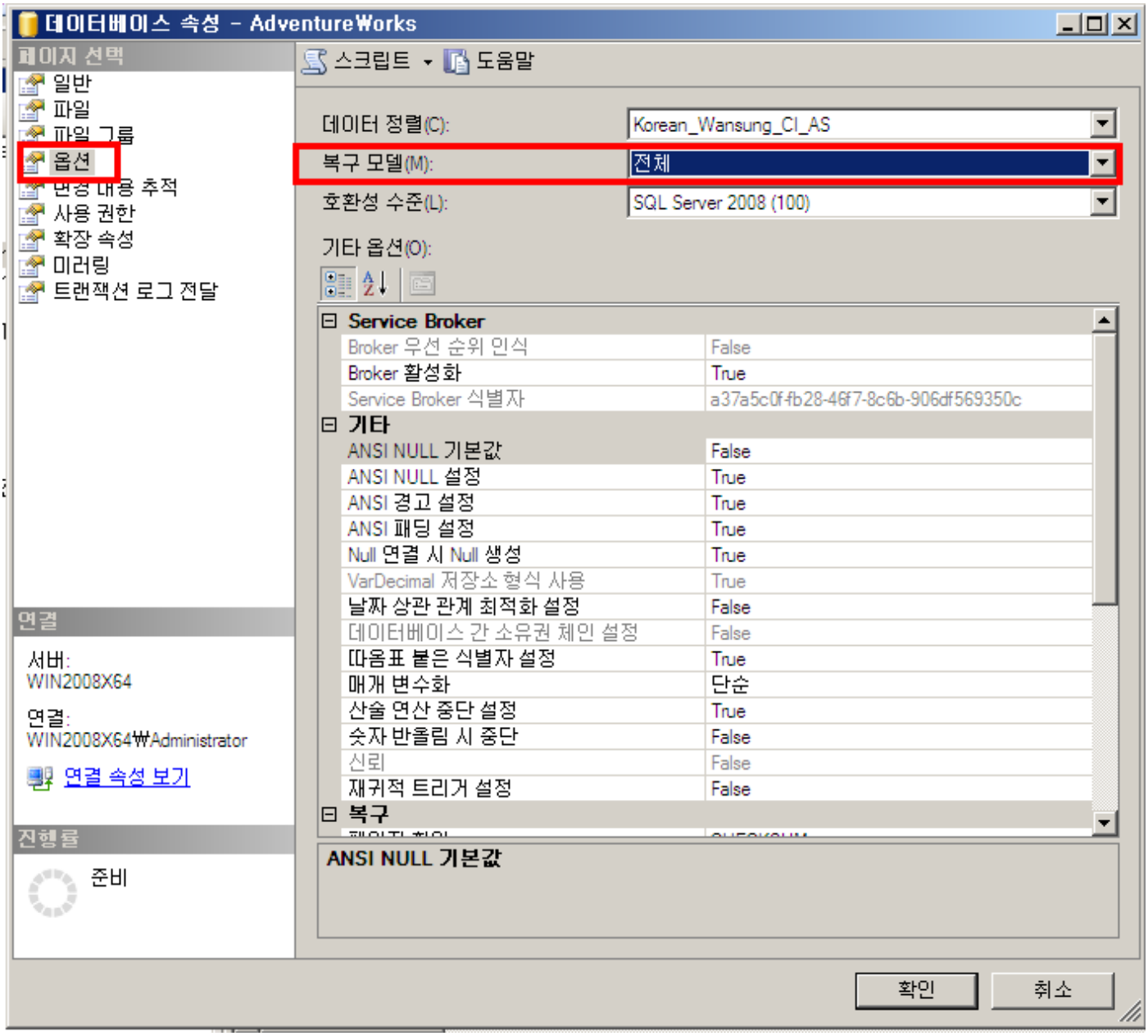
```
GRANT ALL ON userTbl TO auditUser;
GRANT ALL ON buyTbl TO auditUser;

------------------------
-- C:₩감사파일₩  폴더 생성
------------------------

-- 서버 감사 생성
USE master;
CREATE SERVER AUDIT [myServerAudit]
TO FILE
(       FILEPATH = N'C:₩감사파일'
        ,MAXSIZE = 0 MB
        ,MAX_ROLLOVER_FILES = 2147483647
        ,RESERVE_DISK_SPACE = OFF
)
WITH
(       QUEUE_DELAY = 1000
        ,ON_FAILURE = CONTINUE
)
GO

-- 서버 감사 사양 생성
USE master;
CREATE SERVER AUDIT SPECIFICATION [myServerAuditSpec]
FOR SERVER AUDIT [myServerAudit]
ADD (SUCCESSFUL_LOGIN_GROUP),
ADD (FAILED_LOGIN_GROUP),
ADD (LOGOUT_GROUP)
GO

-- 데이터베이스 감사 사양 생성
USE sqlDB;
CREATE DATABASE AUDIT SPECIFICATION [myDBAuditSpec]
FOR SERVER AUDIT [myServerAudit]
ADD (INSERT ON OBJECT::[dbo].[userTbl] BY [auditUser]),
ADD (SELECT ON OBJECT::[dbo].[userTbl] BY [auditUser]),
ADD (DELETE ON OBJECT::[dbo].[userTbl] BY [auditUser])
GO
```

```
-- 서버 감사 시작
USE master;
ALTER SERVER AUDIT myServerAudit
WITH (STATE = ON);

-- 서버 감사 사양 시작
USE master;
ALTER SERVER AUDIT SPECIFICATION myServerAuditSpec
WITH (STATE = ON);

-- 데이터베이스 감사 사양 시작
USE sqlDB;
ALTER DATABASE AUDIT SPECIFICATION myDBAuditSpec
WITH (STATE = ON);

-------------------------
-- 여기서 여러가지 작업 수행
-------------------------

SELECT * FROM sys.fn_get_audit_file('C:₩감사파일₩*',default,default);
```

18장

P876

AdventureWorks 의 데이터베이스 속성에서 변경




```
USE AdventureWorks;

-- 전체백업
BACKUP DATABASE AdventureWorks TO DISK='C:₩adv.bak';

-- 차등백업
BACKUP DATABASE AdventureWorks TO DISK='C:₩adv.bak' WITH DIFFERENTIAL;

-- 로그백업
BACKUP LOG AdventureWorks TO DISK='C:₩adv.bak';
```

P879

```
sp_addumpdevice [ @devtype = ] 'device_type'
      , [ @logicalname = ] 'logical_name'
      , [ @physicalname = ] 'physical_name'
    [ , { [ @cntrltype = ] controller_type |
        [ @devstatus = ] 'device_status' }
    ]
```

P915

```
CREATE DATABASE [testDB3] ON  PRIMARY
( NAME = 'testDB3'
, FILENAME = 'c:₩데이터파일₩testDB3.mdf' )
FOR ATTACH ATTACH_REBUILD_LOG
```

P935

백업은 현재의 데이터베이스를 별도의 장치에 보관하는 것을 말하며, 보관된 데이터는 복원시키기 전에는 사용할 수가 없는 상태가 된다.

스냅숏은 현재 데이터베이스의 상태를 보관하는 용도로 사용하지만, 보관된 데이터도 실시간으로 읽을 수가 있다. 하지만, 스냅숏은 백업의 대신으로 사용하면 오히려 큰 문제를 일으킬 소지가 있다.

19장

P943

(1) 비용이 가장 많이 들더라도, 서비스가 중단되지 않아야 하는 방법

- 서버 클러스터링 + 데이터베이스 미러링 + 디스크 미러링

(2) 서비스가 잠시 중단되어도 큰 문제가 없고, 되도록 최소의 비용으로 구성이 가능한 방법

- 로그 전달 또는 데이터베이스 미러링

(3) 성능을 최대로 유지할 수 있는 방법

- 로그 전달 + 디스크 미러링(RAID 0)

(4) 가장 짧은 서비스 중단을 보장하는 방법

- 데이터베이스 미러링

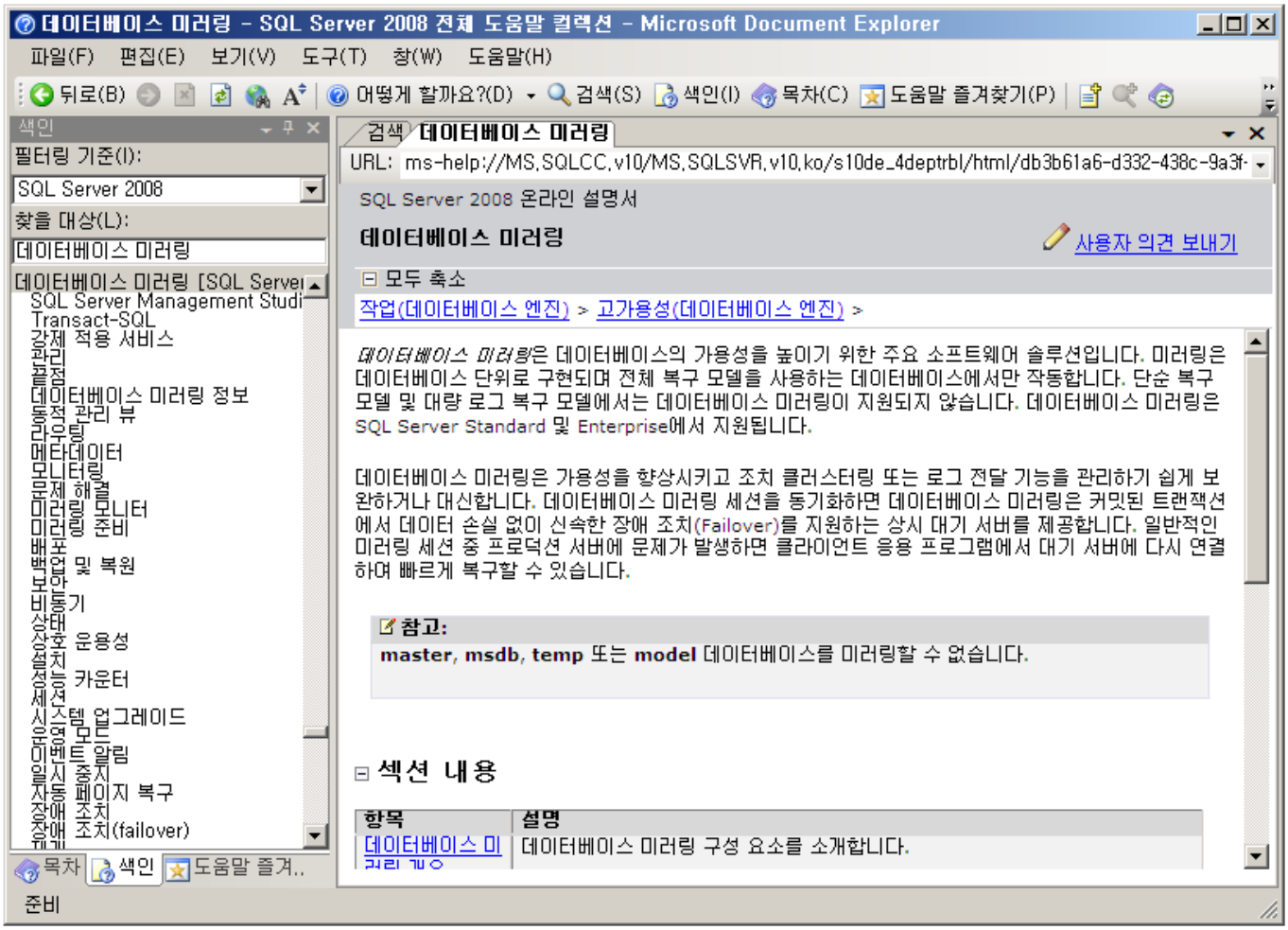
P956

로그전달의 사용법은 교재를 참조.

장점은 다른 것에 비해서 프로세스를 적게 차지하며, SQL Server 2008 이전 버전에서도 제공됨.

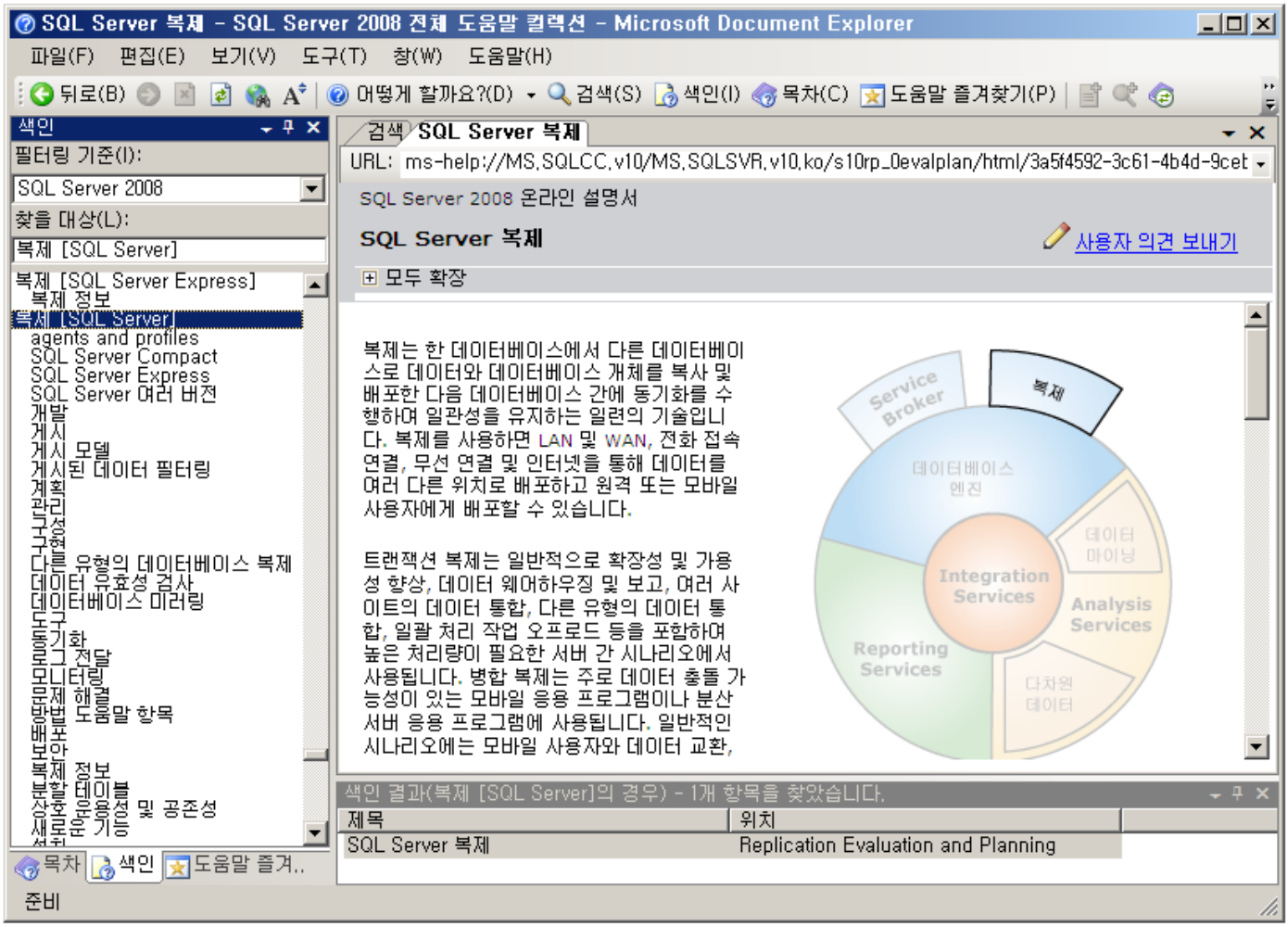
P974

도움말의 색인에서 "데이터베이스 미러링 [SQL Server]"부분 참조



P1010

도움말의 색인에서 "복제 [SQL Server]"부분 참조



20장

P1031

(1) <실습3>의 2-⑤ 이후에 오라클에서 아래 코드 입력

```
CREATE TABLE oraTbl (id NUMBER(3), NAME VARCHAR(10));
INSERT INTO oraTbl VALUES (1, '나일번');
INSERT INTO oraTbl VALUES (2, '나이번');
INSERT INTO oraTbl VALUES (3, '나삼번');
```

(2) SQL Server에 sqlDB4 데이터베이스 생성

(3) <실습4>의 1-① 에서 [태스크] → [데이터베이스 가져오기]를 클릭

(4) 그 이후는 내보내기와 비슷하므로 직접 실습

P1035

MS Access도 다른 DB와 동일한 개념으로 사용하면 되므로, 직접 실습해 볼 것.

21장

P1046

<실습2>의 2-④ 쿼리문을 아래처럼 수정

```
DECLARE @backupDate DATETIME
DECLARE @fileName NVARCHAR(50)
SELECT @backupDate = GETDATE();
SET @fileName = 'c:₩백업폴더₩sqlDB-LOG-'
        + CAST(YEAR(@backupDate) AS CHAR(4)) + '-'
        + CAST(MONTH(@backupDate) AS VARCHAR(2)) + '-'
        + CAST(DAY(@backupDate) AS VARCHAR(2))
        + '.bak'
BACKUP LOG sqlDB TO disk = @fileName;
```

그 외는 필요에 따라 적절히 수정하면 됨.