데이터베이스

조인과 중첩 질의문

학습목표

- SQL 구문에 따라 다중 테이블에 대하여 조인을 수행하고, 정보를 검색할 수 있다.
- 조인문을 중첩 질의문으로 변환할 수 있다.

학습내용

- 조인 구문
- 중첩 질의문

조인 구문

간단한 조인

01 조인의 개념

하나의 SQL 질의문으로 여러 테이블에 저장된 데이터를
한번에 조회할 수 있는 기능

두 개 이상의 테이블 '결합'

조인 구문

간단한 조인

02 조인의 필요성

가정 ···> 하나의 SQL 질의문은 하나의 테이블만 검색할 수 있다.

예 | 사번이 103인 사원의 부서명을 구하시오.

- 사번이 103번인 사원의 부서번호 파악

```
USE MagicCorp
GO

SELECT DNO
FROM EMPLOYEE
WHERE ENO = 103
```

Results | Messages

	DNO
1	30

- 해당 부서번호와 같은 부서번호를 가진 부서명 검색

```
USE MagicCorp
GO

SELECT DNAME
FROM DEPARTMENT
WHERE DNO = 30
```

Results | Messages

	DNAME
1	Sales

가정

하나의 SQL 질의문은 하나의 테이블만 검색할 수 있다.

사용이 불편함

해결

간단한 조인 표기법으로 해결 가능

- FROM절에 조인에 참여하는 두 테이블을 기록 (콤마(,)로 구분)
- WHERE절에 조인 조건을 기술

조인 구문

간단한 조인

03 조인문 작성 방법

예 | 사번이 103인 사원의 부서명을 구하시오.

```
USE MagicCorp
GO

SELECT DNAME
FROM EMPLOYEE, DEPARTMENT
WHERE ENO= 103 and EMPLOYEE.DNO = DEPARTMENT.DNO
```

Results | Messages

	DNAME
1	Sales

- 사원 정보: EMPLOYEE
- 부서 정보: DEPARTMENT
- 조인 조건
  : EMPLOYEE.DNO = DEPARTMENT.DNO

조인 구문

간단한 조인

04 조인문 작성 시 유의 사항

컬럼 이름의 모호성

예 | 서로 다른 두 테이블의 컬럼(속성) 명이 같을 경우

```
USE MagicCorp
GO

SELECT DNAME
FROM EMPLOYEE, DEPARTMENT
WHERE ENO= 103 and DNO = DNO
```

```
Messages
Msg 209, Level 16, State 1, Line 4
Ambiguous column name 'DNO'.
Msg 209, Level 16, State 1, Line 4
Ambiguous column name 'DNO'.
```

어느 속성이 어느 테이블에 있는 것인지 DBMS(Data Base Management System)로 검색 불가

해결 방법

- 컬럼 이름 앞에 테이블 이름을 접두사로 사용
- 테이블 이름과 컬럼 이름은 점(.)으로 구분

예 | DEPARTMENT.DNO = EMPLOYEE.DNO

조인 구문

간단한 조인

04 조인문 작성 시 유의 사항

테이블 별명

- 테이블 이름이 긴 경우 이름 대신 별명을 사용하여 SQL문 작성
- FROM절에 테이블 이름 다음에 공백을 두고 별명을 정의

별명 작성 규칙

- 최대 30자까지 가능
- 테이블 이름과 별명은 하나의 SQL에서 혼용 불가
- 테이블의 별명은 해당 SQL문에서만 유효

예 | 사번이 103인 사원의 부서명을 구하시오.(별명 사용)

```
USE MagicCorp
GO

SELECT DNAME
FROM EMPLOYEE E, DEPARTMENT D
WHERE ENO= 103 and E.DNO = D.DNO
```

ENO=103 and E.DNO=D.DNO 입력

Results | Messages

	DNAME
1	Sales

조인 구문

간단한 조인

04 조인문 작성 시 유의 사항

테이블 별명

예 | 사번이 103인 사원의 부서명을 구하시오.(별명 사용)

```
USE MagicCorp
GO

SELECT DNAME
FROM EMPLOYEE E, DEPARTMENT D
WHERE ENO= 103 and E.DNO = DEPARTMENT.DNO
```

→ 별명을 정의한 후에는 별명만 사용 가능

```
Messages
Msg 4104, Level 16, State 1, Line 4
The multi-part identifier "DEPARTMENT.DNO" could not be bound.
```

조인 구문

다양한 조인들

01 카티션 프로덕트(Cartesian Product: X)

두 테이블에 속한 튜플들의 모든 가능한 쌍을 생성하는 연산

일반적 방법
- FROM절: 두 개 이상의 테이블명을 기록
- WHERE절: 조인 조건을 기술 안 함

예 | SELECT * FROM EMPLOYEE, DEPARTMENT

```
USE MagicCorp
GO
select *
from employee, department
```

SELECT * FROM EMPLOYEE, DEPARTMENT 입력

Results | Messages

	ENO	ENAME	JOB	MANAGER	HIREDATE	SALARY	COMMISSION	DNO	DNO	DNAME	LOC
1	101	e1	staff	1013	2007-03-01 00:00:00.000	300	NULL	20	10	Accounting	Sec
2	102	e2	deputy	1005	2007-04-02 00:00:00.000	250	80	30	10	Accounting	Sec
3	103	e3	section	1005	2005-02-10 00:00:00.000	500	100	30	10	Accounting	Sec
4	104	e4	chief	1008	2003-09-02 00:00:00.000	600	NULL	20	10	Accounting	Sec
5	105	e5	section	1005	2005-04-07 00:00:00.000	450	200	30	10	Accounting	Sec
6	106	e6	chief	1008	2003-10-09 00:00:00.000	480	NULL	30	10	Accounting	Sec
7	107	e7	chief	1008	2004-01-08 00:00:00.000	520	NULL	10	10	Accounting	Sec
8	108	e8	senior	1003	2004-03-08 00:00:00.000	500	0	30	10	Accounting	Sec
9	109	e9	ceo	NULL	1996-10-04 00:00:00.000	1000	NULL	20	10	Accounting	Sec
10	110	e10	section	1003	2005-04-07 00:00:00.000	500	NULL	10	10	Accounting	Sec
11	111	e11	staff	1007	2007-03-01 00:00:00.000	280	NULL	30	10	Accounting	Sec

조인 구문

다양한 조인들

02 동등 조인(Equi Join)과 자연 조인(Natural Join)

동등 조인 | 자연조인

동등 비교(=) 연산자만을 사용하는 조인

특징
- 조인 조건이 = 경우 사용
- 일반적으로 =을 많이 사용

동등 조인 | 자연조인

조인 조건을 명시 하지 않고 조인한다고 할 때,
두 테이블에 공통으로 나타나는 속성의 동등 조인

특징
- 동등 조인과 질의 결과의 구조(스키마)가 똑같지는 않음
- MS-SQL: 자연 조인을 명시적으로 지원하지 않음
- Oracle: 자연 조인을 지원하며, FROM절에 사용 가능
  - FROM 테이블명 NATURAL JOIN 테이블명
  - WHERE절에 조인 조건을 생략

조인 구문

다양한 조인들

03 세타 조인(Theta Join)

일반적인 조인 조건

예1 | 다른 사원의 봉급 보다 많은 봉급을 받는 사원들의 이름을 찾으시오.

조인 조건	<, >, <=, >=, != 등

```
USE MagicCorp
GO

SELECT E1.ENAME
FROM  EMPLOYEE E1, EMPLOYEE E2
WHERE E1.SALARY > E2.SALARY
```

Results | Messages

	ENAME
1	e3
2	e4
3	e5
4	e6
5	e7
6	e8
7	e9
8	e10
9	e13
10	e1
11	e3
12	e4
13	e5
14	e6
15	e7

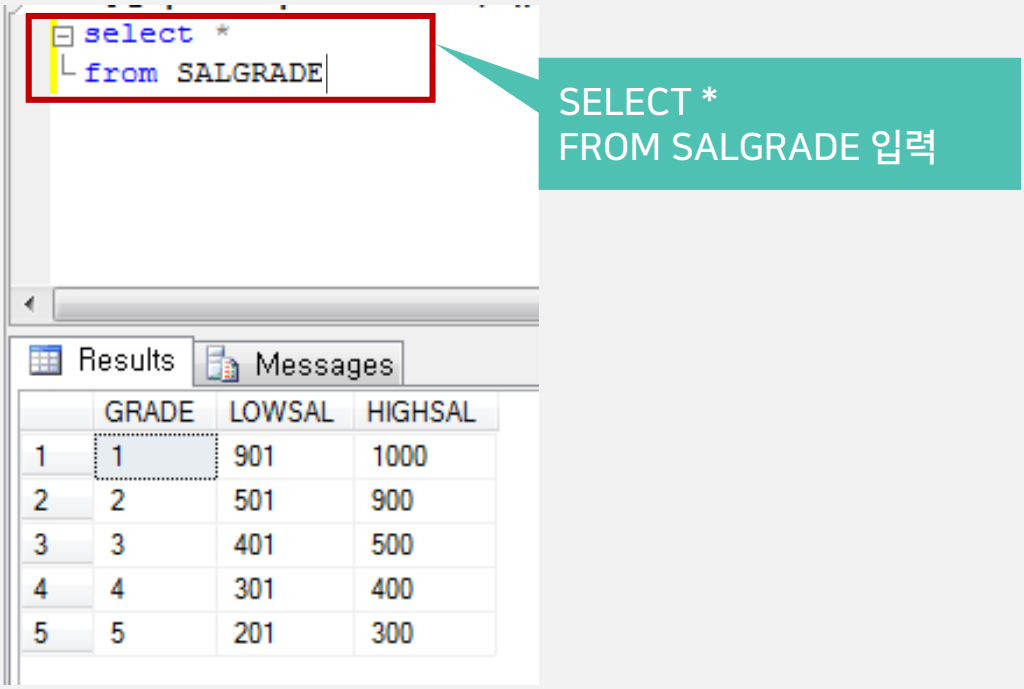
SELECT E1.ENAME
FROM EMPLOYEE E1,
EMPLOYEE E2
WHERE E1.SALRARY >
E2.SALARY 입력

조인 구문

다양한 조인들

03 세타 조인(Theta Join)

예2 | SALGRADE 테이블에서 각 봉급의 하한과 상한에 따른 등급(Grade)을 정하시오.



	GRADE	LOWSAL	HIGHSAL
1	1	901	1000
2	2	501	900
3	3	401	500
4	4	301	400
5	5	201	300

조인 구문

다양한 조인들

03 세타 조인(Theta Join)

예3 | 각 사원의 급여에 따라 사원 등급을 출력하시오.(SALGRADE 참조)

조인 조건

LOWSAL <= SALARY AND
SALARY <= HIGHSAL

```
select ENAME, GRADE
from EMPLOYEE, SALGRADE
WHERE lowsal <= SALARY  and SALARY <= HIGHSAL
```

Results | Messages

	ME	GRADE
		1
		2
		2
4	e13	2
5	e3	3
6	e5	3
7	e6	3
8	e8	3
9	e10	3
10	e1	5
11	e2	5
12	e11	5
13	e12	5
14	e14	5

조인 구문

다양한 조인들

03 세타 조인(Theta Join)

예4 | BETWEEN 연산자로 변환하시오.

```
select ENAME, GRADE
from EMPLOYEE, SALGRADE
WHERE  SALARY between LOWSAL and HIGHSAL
```

하한 상한

Results Messages

	ENAME	GRADE
1	e9	1
2	e4	2
3	e7	2
4	e13	2
5	e3	3

조인 구문

다양한 조인들

04 셀프 조인(Self Join)

하나의 테이블 내에 있는 컬럼끼리 연관시켜 조인이 필요한 경우

예 | 각 사원의 이름과 그 사원의 관리자 이름을 검색하시오

- 사원 이름: EMPLOYEE.NAME
- 사원의 관리자 번호: EMPLOYEEE.MANAGER
- 관리자 이름: EMPLOYEE.NAME(사원의 관리자 또한 사원에 해당)

```
USE MagicCorp
GO
select*
from employee
```

Results | Messages

	ENO	ENAME	JOB	MANAGER	HIREDATE
1	101	e1	staff	1013	2007-03-01 0
2	102	e2	deputy	1005	2007-04-02 0
3	103	e3	section	1005	2005-02-10 0
4	104	e4	chief	1008	2003-09-02 0
5	105	e5	section	1005	2005-04-07 0
6	106	e6	chief	1008	2003-10-09 0
7	107	e7	chief	1008	2004-01-08 0
8	108	e8	senior	1003	2004-03-08 0
9	109	e9	ceo	NULL	1996-10-04 0
10	110	e10	section	1003	2005-04-07 0

조인 구문

다양한 조인들

04 셀프 조인(Self Join)

조인 대상 테이블이 두 개인데, 동일한 테이블인 경우

- 물리적 테이블: 테이블이 1개
- 논리적 테이블: 동일 테이블을 서로 다른 테이블이라고 판단

이 외에는 다른 조인과 동일

Q 두 개의 동일한 테이블로 어떻게 SQL을 실행할까?

- 별명 활용하기

조인 구문

다양한 조인들

04 셀프 조인(Self Join)

예 | 각 사원의 이름과 그 사원의 관리자 이름을 검색하시오

```
SELECT E.ENAME as [employeeName], M.ENAME as [managerName]
 FROM EMPLOYEE E, EMPLOYEE M
WHERE E.MANAGER = M.ENO
```

Results | Messages

	employeeName	managerName
1	e1	e13
2	e2	e5
3	e3	e5
4	e4	e8
5	e5	e5
6	e6	e8
7	e7	e8
8	e8	e3
9	e10	e3
10	e11	e7
11	e12	e6
12	e13	e3
13	e14	e6

SELECT E.ENAME as [employeeName],
M.ENAME as [manageName]
FROM EMPLOYEE E, EMPLOYEE M
WHERE E.MANAGER=M.EMO 입력

조인 구문

다양한 조인들

05 다중 조인(M-Way Join)

이중 조인 (2-Way Join)	다중 조인 (M-Way Join)
조인에 참여하는 테이블의 수가 두 개인 경우	조인에 참여하는 테이블의 수가 여러 개인 경우
예 | 동등 조인, 자연 조인, 세타 조인, 셀프 조인	예 | 삼중 조인 등

- 세타 조인의 예제를 확장하여 각 사원별 이름, 급여, 등급, 부서명을 검색해보자.

- ✓ 사원 테이블의 부서 번호를 통해 부서 테이블을 참조하여 부서명을 가지고 와야 함
- ✓ 사원 테이블의 급여로부터 급여등급 테이블의 등급을 가지고 와야 함

3개의 테이블을 조인(3중 조인)

조인 구문

다양한 조인들

05 다중 조인(M-Way Join)

예 | 사원별 이름, 급여 등급, 부서명을 출력하시오.

이름 EMPLOYEE

급여 등급 SALGRADE

부서명 DEPARTMENT

조인 조건

EMPLOYEE와 SALGRADE
SALARY BETWEEN LOWSAL AND HIGHSAL
EMPLOYEE와 DEPARTMENT
DNO = DNO

```
select ENAME, GRADE, dname
from EMPLOYEE, SALGRADE, DEPARTMENT
WHERE  SALARY between LOWSAL and HIGHSAL
AND EMPLOYEE.DNO = DEPARTMENT.DNO
```

Results | Messages

	ENAME	GRADE	dname
1	e1	5	Human
2	e2	5	Sales
3	e3	3	Sales
4	e4	2	Human
5	e5	3	Sales
6	e6	3	Sales
7	e7	2	Accounting
8	e8	3	Sales
9	e9	1	Human
10	e10	3	Accounting
11	e11	5	Sales
12	e12	5	Human
13	e13	2	Human
14	e14	5	Accounting

select ENAME, GRADE, dname
from EMPLOYEE, SALFRADE,
DEPARTMENT
WHERE SALARY between LOWSAL
and HIGHSAL AND
EMPLOYEE.DNO=DEPARTMENT.DNO
입력

조인 구문

다양한 조인들

06 안시 조인(ANSI Join)

SQL을 표준화할 때 만든 ANSI 표준 문법

주요 특징

- 기존 SQL과 차이점
  - 조인 조건을 WHERE절로 표현하지 않고, FROM절에 표현
- T-SQL문법과 다소 상이하여, 지원되지 않는 부분이 존재

크로스 조인 (Cross Join)

내부 조인 (Inner Join)

외부 조인 (Outer Join)

조인 구문

다양한 조인들

06 안시 조인(ANSI Join)

크로스 조인(Cross Join)

- Cartesian Product의 다른 표현법

> FROM 테이블명 CROSS JOIN 테이블명

FROM EMPLOYEE CROSS JOIN DEPARTMENT 입력

```
USE MagicCorp
GO

SELECT EMPLOYEE.ENAME, DEPARTMENT.DNAME
FROM  EMPLOYEE CROSS JOIN DEPARTMENT
```

Results | Messages

	ENAME	DNAME
1	e1	Accounting
2	e2	Accounting
3	e3	Accounting
4	e4	Accounting
5	e5	Accounting
6	e6	Accounting
7	e7	Accounting
8	e8	Accounting
9	e9	Accounting
10	e10	Accounting
11	e11	Accounting
12	e12	Accounting
13	e13	Accounting
14	e14	Accounting
15	e1	Human

조인 구문

다양한 조인들

06 안시 조인(ANSI Join)

내부 조인(Inner Join)

- 일반적 조건의 안시 조인(ANSI join) 표기법

> FROM 테이블명 INNER JOIN 테이블명 ON 조인 조건

또는

> FROM 테이블명 JOIN 테이블명 ON 조인 조건

» FROM 테이블명 INNER JOIN 테이블명 ON 조인 조건

```
USE MagicCorp
GO

select ename, dname
from  employee INNER JOIN department  ON (employee.dno = department.dno )
```

Results | Messages

	ename	dname
1	e1	Human
2	e2	Sales
3	e3	Sales
4	e4	Human
5	e5	Sales
6	e6	Sales
7	e7	Accounting
8	e8	Sales
9	e9	Human
10	e10	Accounting
11	e11	Sales
12	e12	Human
13	e13	Human
14	e14	Accounting

FROM employee INNER JOIN department ON (employee. dno= department.dno) 입력

조인 구문

다양한 조인들

06 안시 조인(ANSI Join)

내부 조인(Inner Join)

FROM 테이블명 JOIN 테이블명 ON 조인 조건

```
USE MagicCorp
GO

select ename, dname
from  employee JOIN department ON employee.dno = department.dno
```

Results | Messages

	ename	dname
1	e1	Human
2	e2	Sales
3	e3	Sales
4	e4	Human
5	e5	Sales
6	e6	Sales
7	e7	Accounting
8	e8	Sales
9	e9	Human
10	e10	Accounting
11	e11	Sales
12	e12	Human
13	e13	Human
14	e14	Accounting

FROM employee JOIN department ON employee. dno= department.dno 입력

조인 구문

다양한 조인들

06 안시 조인(ANSI Join)

외부 조인(Outer Join)

일반적 조인

조인 조건을 만족하는
튜플들만이 조인 결과로 출력

VS

외부 조인

조인 조건을 만족하지 않는
튜플들까지 모두 결과로 출력

정보의 손실

명시적 표기법

- LEFT OUTER JOIN: 왼쪽 테이블에 있는 튜플들 모두 출력
- RIGHT OUTER JOIN: 오른쪽 테이블에 있는 튜플들 모두 출력
- FULL OUTER JOIN: 양쪽 테이블에 있는 튜플들 모두 출력

조인 구문

다양한 조인들

06 안시 조인(ANSI Join)

외부 조인(Outer Join)

예 | 각 사원의 이름과 해당 사원의 관리자 이름을 출력하시오.
(단, 관리자가 없는 사원들도 질의 결과에 포함)

```
USE MagicCorp
select member.ename as [empolyname], manager.ename as [managername]
from employee member LEFT OUTER JOIN employee manager
ON member.manager = manager.eno
```

Results | Messages

	empolyname	managername
1	e1	e13
2	e2	e5
3	e3	e5
4	e4	e8
5	e5	e5
6	e6	e8
7	e7	e8
8	e8	e3
9	e9	NULL
10	e10	e3
11	e11	e7
12	e12	e6
13	e13	e3
14	e14	e6

Select member.ename as [empolyname], manager.ename as [managername]
From employee member LEFT OUTER JOIN employee manager
ON member.manager = manager.eno 입력

중첩 질의문

중첩 질의문의 개념

01 중첩 질의문의 의미

하나의 SQL문의 결과를 다른 SQL문에 전달 하거나,
두 개의 SQL문을 하나의 SQL로 처리하는 방법

주요 특징

- 이론적으로 중첩 질의문은 조인 구문과 표현 능력이 동일
- 중첩 질의문의 필요성 = 조인의 필요성

중첩 질의문

중첩 질의문의 개념

02 중첩 질의문의 필요성

가정 ···> 하나의 SQL 질의문은 하나의 테이블만 검색할 수 있다.

예 | 사번이 103인 사원의 부서명을 구하시오.

- 사번이 103번인 사원의 부서번호 파악

```
USE MagicCorp
GO

SELECT DNO
FROM EMPLOYEE
WHERE ENO = 103
```

Results | Messages

	DNO
1	30

- 해당 부서번호와 같은 부서번호를 가진 부서명 검색

```
USE MagicCorp
GO

SELECT DNAME
FROM DEPARTMENT
WHERE DNO = 30
```

Results | Messages

	DNAME
1	Sales

가정

하나의 SQL 질의문은
하나의 테이블만 검색할 수 있다.

사용이 불편함

해결

조인문 또는 중첩 질의문 활용

- FROM절에 조인에 참여하는 두 테이블을 기록 (콤마(,)로 구분)
- WHERE절에 조인 조건을 기술

중첩 질의문

중첩 질의문의 개념

02 중첩 질의문의 필요성

- SQL문 안에 SQL문이 포함

```
USE MagicCorp
GO

SELECT DNAME
FROM DEPARTMENT
WHERE DEPARTMENT.DNO = (SELECT DNO
                        FROM EMPLOYEE
                        WHERE EMPLOYEE.ENO= 103
                        )
```

Results | Messages

	DNAME
1	Sales

중첩 질의문

단일행 서브 쿼리와 다중행 서브 쿼리

단일행 서브 쿼리		다중행 서브 쿼리
서브 쿼리의 결과로 하나의 튜플만 반환	VS	서브 쿼리의 결과로 여러 개의 튜플 반환

➡ 서브 쿼리의 검색 조건이 후보 키에 연관된 경우가 다수

Q 단일행 서브 쿼리와 다중행 서브 쿼리를 구분 해야 하는 이유는?

- 일반적인 비교 연산자(=, <, <=, >, >=, != 등)는 속성값 간의 비교 연산을 의미하기 때문
- 집합에 대한 비교 연산 불가

중첩 질의문

단일행 서브 쿼리와 다중행 서브 쿼리

01 단일행 서브 쿼리

예 | 사원 번호 110번과 같은 부서에 근무하는 사원들의 사원 번호와 부서번호를 검색하시오.

- 사원 번호가 기본 키이므로 사원 번호 110번은 한 명
  ➡ 단일행 서브쿼리
- =, <, <=, >, >=, != 등 사용 가능

```
USE MagicCorp
GO

SELECT ENO, DNO
FROM EMPLOYEE
WHERE EMPLOYEE.DNO = (SELECT DNO
                      FROM EMPLOYEE
                      WHERE EMPLOYEE.ENO= 110)
```

Results | Messages

	ENO	DNO
1	107	10
2	110	10
3	114	10

중첩 질의문

단일행 서브 쿼리와 다중행 서브 쿼리

02 다중행 서브 쿼리

단일행 비교 연산자 사용 시 오류 발생

예 | 봉급이 500 이상인 사원과 같은 부서에 근무하는 사원들의 이름, 봉급, 부서번호를 구하시오.

```
USE MagicCorp
GO

SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE DNO = (SELECT DNO
             FROM EMPLOYEE
             WHERE SALARY >= 500)
```

Results | Messages

```
Msg 512, Level 16, State 1, Line 2
Subquery returned more than 1 value. This is not permitted when the subquery follows =, !=, <, <=
```

중첩 질의문

단일행 서브 쿼리와 다중행 서브 쿼리

02 다중행 서브 쿼리

비교 연산자

비교 연산자: IN

- 속성 값이 여러 값들 중 하나이기만 하면 참
- '= OR'의 의미

중첩 질의문

단일행 서브 쿼리와 다중행 서브 쿼리

02 다중행 서브 쿼리

비교 연산자: IN

예 | 봉급이 500 이상인 사원과 같은 부서에 근무하는 사원들의 이름, 봉급, 부서번호를 구하시오.

```
SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE DNO IN(SELECT DNO
                    FROM EMPLOYEE
                    WHERE
                    SALARY>=500) 입력
```

```
USE MagicCorp
GO

SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE DNO IN (SELECT DNO
              FROM EMPLOYEE
              WHERE SALARY >= 500)
```

Results | Messages

	ENAME	SALARY	DNO
1	e1	300	20
2	e2	250	30
3	e3	500	30
4	e4	600	20
5	e5	450	30
6	e6	480	30
7	e7	520	10
8	e8	500	30
9	e9	1000	20
10	e10	500	10
11	e11	280	30
12	e12	300	20
13	e13	560	20
14	e14	250	10

중첩 질의문

단일행 서브 쿼리와 다중행 서브 쿼리

02 다중행 서브 쿼리

비교 연산자: ANY 또는 SOME

- 메인 쿼리 비교 조건에서 서브 쿼리의 결과와 하나라도 일치하면 참
- = ANY와 = SOME은 IN과 같은 의미

! IN과의 차이점: >, >=, <=, <과 같은 범위 비교와도 같이 사용 가능

예 | 부서번호 20에 근무하는 한 직원의 봉급 보다 많은 봉급을 받는 직원들의 이름, 봉급, 부서번호를 출력하시오.

```
SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
                WHERE SALARY >
                ANY(SELECT
                SALARY
                FROM EMPLOYEE
                 WHERE DNO=20) 입력
```

```
USE MagicCorp
GO

SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE SALARY > ANY (SELECT SALARY
                    FROM EMPLOYEE
                    WHERE DNO =20)
```

Results | Messages

	ENAME	SALARY	DNO
1	e3	500	30
2	e4	600	20
3	e5	450	30
4	e6	480	30
5	e7	520	10
6	e8	500	30
7	e9	1000	20
8	e10	500	10
9	e13	560	20

중첩 질의문

단일행 서브 쿼리와 다중행 서브 쿼리

02 다중행 서브 쿼리

비교 연산자: ANY 또는 SOME

> 메인 쿼리 비교 조건에서 서브 쿼리의 결과와 모두 일치하면 참

예 | 부서번호 10에 근무하는 모든 직원들의 봉급 보다 많은 봉급을 받는 직원들의 이름, 봉급, 부서번호를 출력하시오.

```
SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE SALARY > All(SELECT
                   SALARY
                   FROM
                   EMPLOYEE
                    WHERE DNO=10)
                   입력
```

```
USE MagicCorp
GO

SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE SALARY > ALL (SELECT SALARY
                    FROM EMPLOYEE
                    WHERE DNO =10)
```

Results | Messages

	ENAME	SALARY	DNO
1	e4	600	20
2	e9	1000	20
3	e13	560	20

중첩 질의문

단일행 서브 쿼리와 다중행 서브 쿼리

02 다중행 서브 쿼리

비교 연산자: ALL

> 메인 쿼리 비교 조건에서 서브 쿼리의 결과와 모두 일치하면 참

예 | 부서번호 10에 근무하는 모든 직원들의 봉급 보다 많은 봉급을 받는 직원들의 이름, 봉급, 부서번호를 출력하시오.

```
SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE SALARY > All(SELECT
                   SALARY
                   FROM
                   EMPLOYEE
                    WHERE DNO=10)
입력
```

```
USE MagicCorp
GO

SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE SALARY > ALL (SELECT SALARY
                    FROM EMPLOYEE
                    WHERE DNO =10)
```

Results | Messages

	ENAME	SALARY	DNO
1	e4	600	20
2	e9	1000	20
3	e13	560	20

중첩 질의문

단일행 서브 쿼리와 다중행 서브 쿼리

02 다중행 서브 쿼리

비교 연산자: EXISTS

[서브 쿼리의 결과가 하나라도 존재하면 참]

! NOT EXISTS: EXISTS와 상반되는 연산자

예 | 부서번호 10에 근무하는 모든 직원들의 봉급 보다 많은 봉급을 받는 직원들의 이름, 봉급, 부서번호를 출력하시오.

```
SELECT ENAME
FROM EMPLOYEE
WHERE EXISTS(SELECT *
                    FROM EMPLOYEE
                    WHERE SALARY+
                    COMMISSION>500) 입력
```

```
USE MagicCorp
GO

SELECT ENAME
FROM EMPLOYEE
WHERE EXISTS (SELECT *
              FROM EMPLOYEE
              WHERE SALARY+COMMISSION > 500)
```

Results | Messages

	ENAME
1	e1
2	e2
3	e3
4	e4
5	e5
6	e6
7	e7
8	e8
9	e9
10	e10
11	e11
12	e12
13	e13
14	e14

중첩 질의문

다중 컬럼 서브 쿼리

01 다중 컬럼 서브 쿼리의 의미

서브 쿼리의 결과가 여러 개의 속성들로 구성되어
주 쿼리의 조건과 비교하는 서브 쿼리

주요 특징

- 복수 개의 서브 쿼리들로 구성
- 메인 쿼리와 서브 쿼리의 비교 대상 칼럼을 분리
  - 개별적으로 비교한 후, AND 연산에 의해 최종 결과를 출력

중첩 질의문

다중 컬럼 서브 쿼리

02 다중 컬럼 서브 쿼리의 예

예 | 사원 번호 101인 사원과 동일 부서에 동일한 급여를 지급받는 직원을 구하시오.

```
SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE DNO IN(SELECT DNO
                FROM EMPLOYEE
                WHERE ENO=101)
                 AND SALARY
                 IN(SELECT SALARY
                 FROM EMPLOYEE
                 WHERE ENO=101)
                 입력
```

```
SELECT ENO, ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE DNO IN (SELECT DNO
              FROM EMPLOYEE
              WHERE ENO = 101)
AND SALARY IN (SELECT SALARY
              FROM EMPLOYEE
              WHERE ENO = 101)
```

Results | Messages

	ENO	ENAME	SALARY	DNO
1	101	e1	300	20
2	112	e12	300	20

중첩 질의문

상호 연관 서브 쿼리

비상호 연관 서브 쿼리		상호 연관 서브 쿼리
메인 쿼리 절에서 어떤 튜플을 검색하던, 서브 쿼리의 결과는 일정하게 출력	VS	메인 쿼리 절에서 어떤 튜플에 대한 조건을 비교하는가에 따라 서브 쿼리의 결과가 다르게 출력

01 비상호 연관 서브 쿼리

예 | 부서 테이블에서 어떤 튜플을 검색하던지 서브 쿼리의 결과는 일관되게 부서번호 30을 반환함

```
USE MagicCorp
GO

SELECT DNAME
FROM DEPARTMENT
WHERE DEPARTMENT.DNO = (SELECT DNO
                        FROM EMPLOYEE
                        WHERE EMPLOYEE.ENO= 103
                        )
```

Results | Messages

	DNAME
1	Sales

SELECT DNO
FROM EMPLOYEE
WHERE EMPLOYEE.ENO=103
입력

중첩 질의문

상호 연관 서브 쿼리

02 상호 연관 서브 쿼리

메인 쿼리 절과 서브 쿼리 간에 검색 결과를 교환하는 서브 쿼리

주요 특징

- 서브 쿼리의 WHERE 조건 절에서 메인 쿼리의 테이블과 연결
- 서브 쿼리의 조건 절에 메인 쿼리에서 사용하는 테이블의 속성이 명시

다시 말해서, 메인 쿼리에서 어떤 튜플에 대한 조건을 비교하는가에 따라서 서브 쿼리의 결과가 다르게 나타남

사용법

```
SELECT 속성리스트
FROM table1
WHERE table1.속성  비교연산자 (
                                SELECT 속성리스트
                                FROM table2
                                WHERE table2.속성 비교연산자 table1.속성)
```

중첩 질의문

상호 연관 서브 쿼리

02 상호 연관 서브 쿼리

주의사항

메인 쿼리 절에서 table1에 속한 튜플을 하나씩 접근하여
WHERE절 수행

서브 쿼리가 반복적으로 수행되므로 성능이 매우 저하될 가능성 존재

조인 구문 활용이 더 효율적

중첩 질의문

상호 연관 서브 쿼리

02 상호 연관 서브 쿼리

예 | 각 사원에 대하여 관리자와 동일 부서에서 근무하는 사원들의 이름, 급여, 부서번호를 출력하시오.

```
SELECT ENO, ENAME, SALARY, DNO
FROM EMPLOYEE E
WHERE  DNO IN (SELECT DNO
               FROM EMPLOYEE M
               WHERE E.MANAGER = M.ENO)
```

사원마다 관리자가 달라짐

Results | Messages

	ENO	ENAME	SALARY	DNO
1	101	e1	300	20
2	102	e2	250	30
3	103	e3	500	30
4	105	e5	450	30
5	106	e6	480	30
6	108	e8	500	30

조인 구문

중첩 질의문 작성 시 주의점

01 다중행 서브 쿼리 시 단일행 비교 연산자를 사용하는 경우

➡ 중첩 질의문 사용시 오류가 없도록 반복적으로 IN, ANY, ALL을 기본적으로 사용

```
USE MagicCorp
GO

SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE DNO = (SELECT DNO
             FROM EMPLOYEE
             WHERE SALARY >= 500)
```

SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE DNO=(SELECT DNO
FROM EMPLOYEE
WHERE
SALARY >=500) 입력

Results Messages

```
Msg 512, Level 16, State 1, Line 2
Subquery returned more than 1 value. This is not permitted when the subquery follows =, !=, <, <=
```

조인 구문

중첩 질의문 작성 시 주의점

02 서브 쿼리 내에서 ORDER BY 절 사용 불가

```
SELECT ENO, ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE DNO IN (SELECT DNO
              FROM EMPLOYEE
              WHERE ENO = 101
              ORDER BY DNO)
```

SELECT ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE DNO IN(SELECT DNO
FROM EMPLOYEE
WHERE ENO=101)
ORDER BY DNO 입력

Messages

```
Msg 1033, Level 15, State 1, Line 7
The ORDER BY clause is invalid in views, inline functions, derived tables, subqueries, and common table expressions, unless TOP or FO
```

조인 구문

중첩 질의문 작성 시 주의점

03 서브 쿼리의 결과가 NULL일 경우, 메인 쿼리의 결과 또한 NULL

서브 쿼리가 NULL 을 반환할 경우, 메인 쿼리에서 결과를 생성하고 싶으면 'NOT EXISTS'를 사용

```
SELECT ENO, ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE DNO IN (SELECT DNO
              FROM EMPLOYEE
              WHERE ENO = 500)
```

Results | Messages

ENO	ENAME	SALARY	DNO

```
SELECT ENO, ENAME, SALARY, DNO
FROM EMPLOYEE
WHERE NOT EXISTS (SELECT DNO
                  FROM EMPLOYEE
                  WHERE ENO = 500)
```

Results | Messages

	ENO	ENAME	SALARY	DNO
1	101	e1	300	20
2	102	e2	250	30
3	103	e3	500	30
4	104	e4	600	20
5	105	e5	450	30
6	106	e6	480	30

학습정리

1 조인 구문

✓ 간단한 조인

- 조인이란 하나의 SQL 질의문에 의해서 여러 테이블에 저장된 데이터를 한 번에 조회할 수 있는 기능
- SQL에서 간단한 조인 표기는 FROM절에 조인에 참여하는 두 테이블을 기록하고 WHERE절에 조인 조건을 기술

✓ 다양한 조인들

- ANSI 조인은 SQL을 표준화할 때 만든 ANSI 표준 문법
- 기존 SQL과의 차이점: 조인 조건을 WHERE절로 표현하지 않고 ON절에 표현함

2 중첩 질의문

✓ 하나의 SQL문의 결과를 다른 SQL문에 전달하여 두 개의 SQL문을 하나의 SQL문으로 처리

✓ 이론적으로 중첩 질의문은 조인 구문과 표현 능력이 동일

✓ 단일행 서브쿼리: 서브 쿼리의 결과로 하나의 튜플만이 반환

✓ 다중행 서브쿼리: 서브 쿼리의 결과로 여러 개의 튜플들이 반환

✓ 비상호 연관 서브 쿼리: 서브 쿼리의 결과가 메인 쿼리에서 검사하는 튜플에는 영향을 받지 않고 그 결과가 일정

✓ 상호 연관 서브 쿼리: 메인 쿼리 절과 서브 쿼리 간에 검색 결과를 교환하는 서브 쿼리