데이터베이스론	국가 전산 7급	2019년 8월 17일

♤ 합격선/최종합격인원(74.16점/33명) - 선발예정인원 30명 ♤

1. 관계형 데이터베이스에서 참조 무결성 제약조건을 만족하도록 외래키를 생성하고자 한다. 이와 관련된 SQL 명령문의 제약조건에 대한 설명으로 옳지 않은 것은? [2019년 국가 7급]

① 'on delete set null'은 참조되는 테이블의 행이 삭제되면, 참조하는 테이블의 행에 있는 외래키 열에 null을 저장한다.

② 'on delete cascade'는 참조되는 테이블의 행이 삭제되면, 참조하는 테이블의 행에 있는 외래키 열을 삭제한다.

③ 'on delete set default'는 참조되는 테이블의 행이 삭제되면, 참조하는 테이블의 행에 있는 외래키 열에 사전에 정의된 default 값을 저장한다.

④ 'on delete no action'은 참조되는 테이블의 행을 삭제하려고 할 때, 참조하는 테이블의 행이 존재할 경우 삭제 명령이 수행되지 못하도록 한다.

♤ SQL 명령문의 제약조건

- 'on delete cascade'는 참조되는 테이블의 행이 삭제되면, 참조하는 테이블의 행에 있는 **외래키 열을 삭제**한다.(×)
  → 참조하는 테이블의 관련 **행**이 **모두 삭제**되어야 참조 무결성이 유지된다.

// 참조 무결성 유지

주어진 문제에 대한 비슷한 유형의 예를 들면 다음과 같다.

[학생]

학번	이름	*학과명*	학년	
100	산	전산과	4	← 기본키 : 학번
200	해	수학과	1	← 외래키 : 학과명
300	강	전산과	2	
400	별	물리과	4	

[학과]

학과명	인원	
전산과	90	
수학과	80	
물리과	75	← 기본키 : 학과명
화학과	85	

학생 테이블의 외래키 "학과명"은 학과 테이블의 "학과명"을 참조

- 학생 테이블에서 기본키는 학번이고, 외래키는 학과명이다.
- 학생 테이블의 외래키 학과명은 학과 테이블의 학과명을 참조하고 있다.
- 학과 테이블에서 **전산과**를 **삭제**하는 경우
- 학생 테이블에서 **학과명**이 **전산과**인 튜플은 **모두 삭제**되어야 참조 무결성이 유지된다.

```
Create Table 학생
(
  학번          char(4) Not Null,
  이름          char(10),
  학과명        char(15),
  학년          integer,
  Primary key(학번),                              → 기본키 "학번"
  Foreign Key(학과명) References 학과(학과명)   → 외래키 "학과명"
    On Delete Cascade        → 참조되는 테이블의 행이 동시에 삭제된다.
    On Delete Set Null       → 테이블 행이 삭제되면, 참조하는 테이블 행의 외래키에 Null 저장
    On Delete Set Default    → 테이블 행이 삭제되면, 참조하는 테이블 행의 외래키에 Default
    On Delete No Action,     → 테이블 행을 삭제하려고 할 때, 삭제 명령이 수행되지 못하도록
  Check(학년≥1 And 학년≤4)
);

Create Table 학과
(
  학과명 char(15) Not Null,
  인원 integer,
  Primary key(학과명) → 기본키 "학과명"
);
```

// ON DELETE { NO ACTION | CASCADE | SET NULL | SET DEFAULT }

No Action	참조되는 테이블의 행을 삭제하려고 할 때, 참조하는 테이블의 행이 존재할 경우 삭제 명령이 수행되지 못하도록 한다.
Cascade	참조되는 테이블의 행이 삭제되면, 참조하는 테이블의 관련 행이 모두 삭제된다.
Set Null	참조되는 테이블의 행이 삭제되면, 참조하는 테이블 행의 외래키에 Null이 저장된다. 이 제약조건을 실행하려면 외래키 열이 Null을 허용해야 한다.
Set Default	참조되는 테이블의 행이 삭제되면, 참조하는 테이블의 행에 있는 외래키 열에 사전에 정의된 Default 값을 저장한다.

• 참조된 행이 부모 테이블에서 삭제될 경우에 테이블의 행에 수행될 동작을 지정한다.
• 기본값은 No Action이다.

정답 : ②

2. 다음 과제 테이블은 학년별 동아리에 가입한 학생 수와 제출한 과제 수를 저장하고 있다. '학생이 10명 이상 가입한 동아리에 대하여 동아리와 제출한 총 과제 수를 출력하시오'를 수행하기 위한 SQL문으로 옳은 것은? [2019년 국가 7급]

과제			
학년	동아리	학생수	과제수
1	A	20	20
1	B	10	20
1	C	5	8
2	A	15	10
2	B	15	20
3	A	5	15
3	B	12	20
3	C	8	15

① Select 동아리, Sum(과제수) From 과제
Group By 동아리 Having Sum(학생수) >= 10;
② Select 동아리, Sum(과제수) From 과제
Where Sum(학생수) >= 10;
③ Select 동아리, Sum(과제수) From 과제
Having Sum(학생수) >= 10;
④ Select 동아리, Sum(과제수) From 과제
Where 학생수 >= 10 Group By 동아리;

♤ SQL문

• '학생이 10명 이상 가입한 동아리에 대하여 동아리와 제출한 총 **과제 수**를 출력하시오'

↓

↓ 질의문에 대한 SQL문

↓

```
Select 동아리, Sum(과제수) From 과제     //동아리, 총 과제수를 출력
Group By 동아리                          //동아리별로 그룹화
Having Sum(학생수) >= 10;                //학생이 10명 이상 가입
```

정답 : ①

3. 다음 관계형 데이터베이스의 세 가지 기능적 요소에 대한 설명에서 ㉠~㉢에 들어갈 용어를 바르게 연결한 것은? [2019년 국가 7급]

○ (㉠)는(은) SQL에서 삽입, 삭제, 갱신과 같은 데이터 변경문을 실행할 때 미리 명시된 조건을 만족하는 경우 특정한 동작을 자동으로 수행할 수 있도록 한다.
○ (㉡)는(은) 데이터베이스 내에 존재하는 작업순서가 정해진 수행 단위로서 DBMS에서 컴파일된 후 실행된다.
○ (㉢)는(은) 데이터베이스에서 데이터를 신속하게 탐색할 수 있도록 만든 데이터 구조이다.

	㉠	㉡	㉢
①	인덱스	트리거(trigger)	주장(assertion)
②	주장	인덱스	저장 프로시저(stored procedure)
③	주장	인덱스	트리거
④	트리거	저장 프로시저	인덱스

♤ **관계형 데이터베이스 기능**

// 트리거(trigger)
- SQL 서버나 데이터베이스에 **어떤 이벤트가 발생**했을 때, 트리거가 발생될 수 있다.
- 트리거는 명시된 조건을 만족하는 경우 특정한 동작을 **자동으로 수행**할 수 있도록 한다.
- 트리거는 데이터베이스 **무결성**을 유지하기 위한 일반적이고 강력한 도구이다.
- 트리거는 제약조건을 위반했을 때 수행할 동작을 명시하는 것이다.

// 주장(assertion)
- 주장은 **제약조건을 위반하는 연산은 수행되지 않도록** 하는 것이다.
- 주장은 트리거보다 좀 더 **일반적인 무결성** 제약조건에 적용한다.
- 주장은 2개 이상의 테이블에 영향을 미치는 제약조건을 명시하기 위해 사용될 수 있다.

// 저장 프로시저
- 저장 프로시저는 데이터베이스 내에 존재하는 **작업순서가 정해진 수행** 단위이다.
- 저장 프로시저는 일련의 쿼리를 하나의 함수처럼 실행하기 위한 쿼리 집합이다.
- 저장 프로시저는 DBMS에서 컴파일된 후 데이터베이스시스템 내부에 저장되어 실행된다.
- 저장 프로시저는 데이터베이스 서버와 같은 프로세스 공간에서 실행된다.

// 인덱스
- 인덱스는 데이터베이스에서 데이터를 **신속하게 탐색**할 수 있도록 만든 데이터 구조이다.
- 인덱스는 하나의 필드 또는 여러 필드를 기반으로 하여 인덱스를 만들 수 있다.

정답 : ④

4. 데이터베이스 보안과 관련한 설명으로 옳지 않은 것은? [2019년 국가 7급]

① SQL 삽입(injection) 공격은 공격자가 악의적으로 만든 SQL 명령을 응용프로그램이 수행하도록 하는 것이다.

② 데이터베이스 관리자(DBA)가 각 사용자에게 데이터베이스에 대한 접근권한을 부여하거나 취소할 때 grant 명령을 사용한다.

③ 데이터베이스에 대한 권한은 역할(role)에도 부여할 수 있다.

④ 전자서명(digital signature)은 공개키 암호화 기법의 특성을 이용하여 인증을 수행한다.

♤ **데이터베이스 보안**

// 권한 부여, 거부, 제거

Grant	• 데이터베이스 사용자에게 사용 권한을 명시적으로 **부여**한다. • 사용 권한 부여(사용 권한 허가)
Deny	• 데이터베이스 사용자에게 사용 권한을 갖지 못하도록 명시적으로 **거부**한다. • 사용 권한 거부(사용 권한 금지, 사용 권한 차단) • **Deny**는 **Grant**보다 우선한다.
Revoke	• 데이터베이스 사용자에게 기존의 **Grant** 또는 **Deny** 권한을 **제거**한다. • 이전에 허가(**Grant**)되거나 거부(**Deny**)된 사용 권한을 제거한다. • **(명령)을 취소하다.** 권한 제거(권한 취소, 권한 철회)

// 권한 종류(연산)

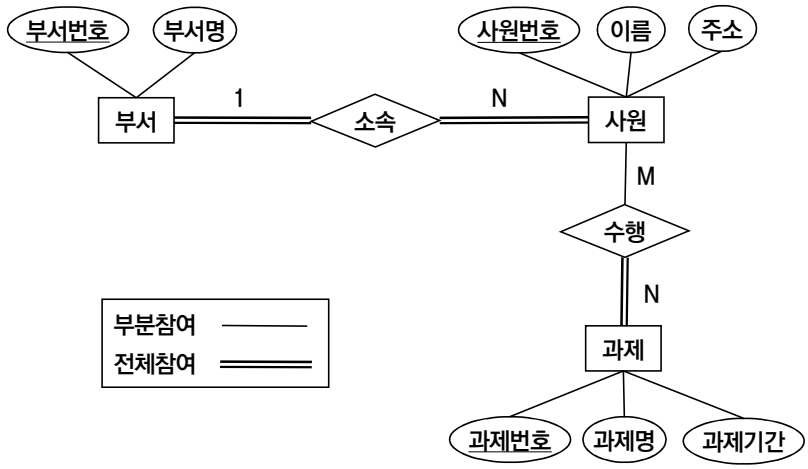
• select : 테이블 내용을 탐색할 수 있다.(조회)
• insert : 테이블에 새로운 인스턴스를 추가할 수 있다.
• update : 테이블에 존재하는 인스턴스를 수정할 수 있다.
• delete : 테이블에 존재하는 인스턴스를 삭제할 수 있다.
• alter : 테이블의 정의를 수정할 수 있다.(테이블에 속성 추가 또는 삭제)
• index : 탐색 속도 향상을 위해 칼럼에 인덱스 생성 또는 제거할 수 있다.
• execute : 데이터베이스에 저장되어 있는 저장 프로시저를 호출할 수 있다.
• all privileges : 위에서 설명한 모든 권한을 포함

//예제 : 권한 부여----------

• Grant Select, Insert On 학생 To 홍재연;
→ 홍재연에게 테이블 학생에 대한 Select, Insert 권한을 부여
• Grant Execute On proc_Name To 홍재연;
→ 홍재연에게 저장 프로시저 proc_Name에 대한 Execute 권한을 부여

정답 : ②

5. 다음 ERD(entity-relationship diagram)에 대한 설명으로 옳지 않은 것은? (단, 속성 이름에 대한 밑줄은 기본키이다) [2019년 국가 7급]



① 주어진 ERD를 릴레이션으로 사상(mapping)하면, 4개의 릴레이션이 생성된다.
② 모든 사원은 적어도 하나 이상의 과제를 수행하여야 한다.
③ 모든 부서는 적어도 1명 이상의 사원이 존재하여야 하며, 사원도 반드시 하나의 부서에 소속되어야 한다.
④ 사원은 2개의 부서에 동시에 소속될 수 없다.

♤ ERD(entity-relationship diagram)

- 모든 사원은 적어도 **하나 이상**의 과제를 수행하여야 한다.**(×)**
  → 모든 사원은 적어도 **0개 이상**의 과제를 수행하여야 한다.
  → 이유는, 사원과 과제 수행은 부분참여 관계이다.**(단일선)**
- **부분참여**는 개체 집합에서 일부 개체만 관계에 참여해도 된다는 것이다.
- 부분참여이므로, 모든 사원이 반드시 과제 수행에 참여할 필요는 없다.

// 관계형 데이터베이스 스키마 생성

부서	부서번호	부서명		
사원	사원번호	이름	주소	*부서번호*
과제	과제번호	과제명	과제기간	
수행	사원번호	과제번호		

정답 : ②

6. 다음 함수종속성 집합 FD의 최소커버(minimal cover) FDmin는? [2019년 국가 7급]

FD = {Y→X, Z→XYW}

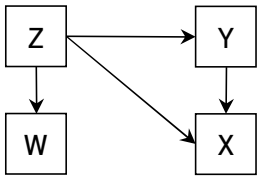
---

① FDmin = {Y→X, Z→X, Z→Y, Z→W}
② FDmin = {Y→X, Z→Y, Z→W}
③ FDmin = {Y→X, Z→X, Z→W}
④ FDmin = {Y→X, Z→X, Z→Y}

♤ 함수종속성 집합 FD의 최소커버(minimal cover, 최소덮개, 최소집합)

- 함수종속성 집합 FD의 최소커버는 **불필요한 종속성을 제거**하는 것이다.(최소집합)
- 불필요한 종속성 제거는 다른 종속성으로 대치 가능할 때 종속성을 제거하는 것이다.
- **예** : {A→B, AB→C}는 {A→B, A→C}의 조합으로 대치 가능
- 함수종속성 집합 F의 최소집합은 모든 종속성은 가능한 작다는 뜻이다.
- 왼쪽 속성들은 꼭 필요한 것만 존재해야 하고, 오른쪽은 속성이 하나만 존재해야 한다.

// 먼저, 주어진 문제는 다음과 같은 릴레이션 구조이다.



차번호(Z)	배기량(W)	면허번호(Y)	주소(X)
1	1000	1234	서울
2	2000	1234	서울
3	2000	3333	평양
4	3000	4444	제주

- FD = {Y→X, Z→XYW}
  ↓
  ↓ 우측 속성을 분리해서 적으면
  ↓
- FD = {Y→X, Z→X, Z→Y, Z→W}
  ↓
  ↓ 이행성 규칙 {Z→Y, Y→X}의 조합으로 Z→X는 대치 가능하다.
  ↓ 해서, Z→X는 제거한다.
  ↓
- FDmin = {Y→X, Z→Y, Z→W}

정답 : ②

7. 다음 인덱스 기법에 대한 설명으로 옳은 것만을 모두 고르면? [2019년 국가 7급]

ㄱ. B-트리 전체를 노드 내의 키 값에 따라 순차검색하기 위해서는 트리의 각 노드를 전위순회(preorder traversal) 한다.

ㄴ. 밀집(dense) 인덱스는 희소(sparse) 인덱스에 비해 액세스 시간은 빠르지만 더 많은 공간을 필요로 한다.

ㄷ. B-트리에서 오버플로가 발생하여 리프노드가 분할될 때, 중간키 값이 부모노드뿐만 아니라 새로 분할된 노드에도 저장된다.

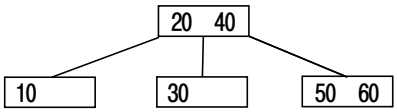
ㄹ. 1,000명의 사원 정보가 저장된 '사원' 테이블의 '부서' 필드에 대하여, 30명이 소속된 '총무과' 사원에 대한 비트맵 인덱스를 구성할 경우 1,000비트가 필요하다.

---

① ㄱ, ㄷ ② ㄱ, ㄹ ③ ㄴ, ㄷ ④ ㄴ, ㄹ

♤ **인덱스 기법**

ㄱ. B-트리 전체를 노드 내의 키 값에 따라 **순차검색**하기 위해서는 트리의 각 노드를 **전위순회**(preorder traversal) 한다.**(×)** → **중위순회**



중위순회(좌,중,우) : 10 20 30 40 50 60

ㄷ. **B-트리**에서 오버플로가 발생하여 리프노드가 분할될 때, **중간키 값**이 부모노드뿐만 아니라 새로 분할된 노드에도 저장된다.**(×)** → **B^+-트리**

// 비트맵 인덱스

- 인덱스 값을 0과 1로 변환하여 저장한다.(비트맵)
- 성별에 대한 비트맵 인덱스(사원이 5명인 경우)

사번	이름	성별
1	홍재연	남
2	홍하은	여
3	김영미	여
4	홍연재	남
5	김하은	여

남 : 10010
여 : 01101

- 성별과 무관하게 비트맵 인덱스 크기는 5가 된다.(사원이 5명인 경우)
- 사원 수가 1000명이면, 비트맵 인덱스 크기는 1000bit가 된다.
- 비트가 1로 설정돼 있으면, 상응하는 레코드가 해당 키 값을 포함하고 있음을 의미한다.

정답 : ④

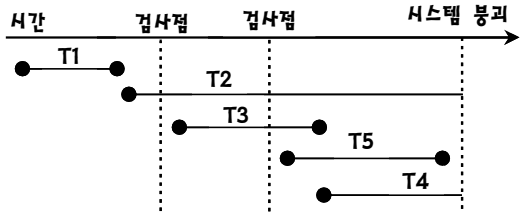
8. 다음은 시스템 고장이 발생할 때 트랜잭션 T1, T2, T3, T4, T5를 복구하기 위해 고장 전에 마지막으로 기록된 DBMS 로그이다. 이 로그를 사용하여 즉시갱신 회복 기법의 undo-redo 알고리즘을 수행할 때, 회복 과정에 대한 설명으로 옳지 않은 것은? (단, <checkpoint>는 검사점 기록 로그 레코드이며, <T1, A, 200, 400>은 'T1이 데이터 항목 A의 현재 값 200을 400으로 갱신한다'를 의미하는 로그 레코드이다. 표현되지 않은 로그의 다른 속성은 고려하지 않는다)

로그 번호	로그 레코드
1	<T1 start>
2	<T1, A, 200, 400>
3	<T1 commit>
4	<T2 start>
5	<T2, B, 900, 1900>
6	<checkpoint>
7	<T3 start>
8	<T3, A, 400, 700>
9	<checkpoint>
10	<T5 start>
11	<T5, C, 100, 5000>
12	<T3 commit>
13	<T4 start>
14	<T4, D, 700, 1300>
15	<T5 commit>
16	<T4, E, 1300, 1500>

--시스템 고장 발생-

① T1은 어떠한 undo와 redo 연산도 수행하지 않는다.
② 데이터 항목 A와 B의 값은 각각 700과 900으로 갱신된다.
③ 데이터 항목 C와 D의 값은 각각 5000과 700으로 갱신된다.
④ redo 연산은 T5, T3 순서로 수행된다.

♤ 검사점을 이용한 즉시 갱신 회복 기법



- T1 : 검사점 이전 완료(no action)
- T2, T4 : 미완료(undo 연산)
- T3, T5 : 검사점 이후 완료(redo 연산)

- redo 연산은 redo-list에 있는 로그 기록된 순으로 수행(T3, T5 순서로 수행)

정답 : ④

9. 〈보기 1〉에서 공급업체와 부품, 카탈로그 테이블을 생성하는 SQL문을 수행한 후 튜플을 삽입하여 세 테이블의 상태가 다음과 같을 때, 〈보기 2〉의 SQL문을 수행한 결과로 옳은 것은? [2019년 국가 7급]

-----<보기 1>----------

```
Create Table 공급업체
(
  업체번호 Int Not Null,
  업체명 Varchar(20),
  Primary Key(업체번호));
```

```
Create Table 부품
(
  부품번호 Int Not Null,
  부품명 Varchar(20),
  색상 Varchar(20),
  Primary Key(부품번호));
```

```
Create Table 카탈로그
(
  업체번호 Int Not Null,
  부품번호 Int Not Null,
  가격 Int,
  Primary Key(업체번호, 부품번호),
  Foreign Key(업체번호) References 공급업체(업체번호),
  Foreign Key(부품번호) References 부품(부품번호)
);
```

공급업체

업체번호	업체명
1	공급A
2	공급B
3	공급C

부품

부품번호	부품명	색상
1	부품1	빨강
2	부품2	파랑
3	부품3	노랑

카탈로그

업체번호	부품번호	가격
1	3	10000
2	1	20000
3	2	30000

-----<보기 2>----------

```
Select 업체번호, 업체명 From 공급업체
Where Not Exists
      (Select 부품.부품번호 From 부품 Where 부품.색상='빨강' And Exists
            (Select * From 카탈로그
            Where 카탈로그.부품번호=부품.부품번호
                  And 카탈로그.업체번호=공급업체.업체번호
            )
      );
```

---

①

업체번호	업체명

②

업체번호	업체명
2	공급B

③

업체번호	업체명
1	공급A
3	공급C

④

업체번호	업체명
1	공급A
2	공급B
3	공급C

♤ SQL문

공급업체	
업체번호	업체명
1	공급A
2	공급B
3	공급C

부품		
부품번호	부품명	색상
1	부품1	빨강
2	부품2	파랑
3	부품3	노랑

카탈로그		
업체번호	부품번호	가격
1	3	10000
2	1	20000
3	2	30000

↓SQL문 실행 결과는?

```
Select 업체번호, 업체명 From 공급업체
    Where Not Exists
        (Select 부품.부품번호 From 부품 Where 부품.색상='빨강' And Exists
            (Select * From 카탈로그
             Where 카탈로그.부품번호=부품.부품번호
               And 카탈로그.업체번호=공급업체.업체번호
            )
        );
```

① Select 부품.부품번호 From 부품 Where 부품.색상='빨강'

부품.부품번호
1

② Select * From 카탈로그
Where 카탈로그.부품번호=부품.부품번호 And 카탈로그.업체번호=공급업체.업체번호

공급업체.업체번호	업체명	부품.부품번호	부품명	색상	카탈로그.업체번호	카탈로그.부품번호	가격
1	공급A	3	부품3	노랑	1	3	10000
2	공급B	1	부품1	빨강	2	1	20000
3	공급C	2	부품2	파랑	3	2	30000

↓연산 결과 : ① AND ②

공급업체.업체번호	업체명	부품.부품번호	부품명	색상	카탈로그.업체번호	카탈로그.부품번호	가격
2	공급B	1	부품1	빨강	2	1	20000

↓ Select 업체번호, 업체명 From 공급업체 Where Not Exists

업체번호	업체명
1	공급A
3	공급C

정답 : ③

10. 다음 SQL문에 대하여 질의 최적화를 수행하고자 한다. 이에 대한 설명으로 옳지 않은 것은?

```
Select 학생.이름, 교수.이름 From 학생, 학과, 교수
Where 학생.학과코드 = 학과.학과코드
  And 교수.교수코드 = 학과.학과장코드
  And 학과.학과코드 = 'CS';
```

① 카탈로그는 해당 테이블에 대한 통계 정보를 저장하고 있어 비용 계산에 활용될 수 있지만, 통계 정보를 실시간으로 갱신하려면 부하가 커서 주기적으로 갱신하기도 한다.

② 조인연산 학생 $\bowtie_{학과코드=학과코드}$ 학과는 기본적으로 중첩루프(nested loop)를 이용하여 구현하지만, '학과.학과코드'에 인덱스가 구축되어 있다면 비용을 감소시킬 수 있다.

③ 경험적 질의 최적화 기법에서는 주어진 SQL문의 Where절에서 조인연산보다 학과.학과코드='CS'를 먼저 수행하도록 한다.

④ | R | 을 릴레이션 R의 투플의 수로 정의할 때, $R\bowtie_{A=B}S$에 대하여 A가 R의 기본키이면 $|(R\bowtie_{A=B}S)| \le |S|$ 이고 조인 선택률(js)은 $js > 1 / |R|$ 이다.

♤ **질의 최적화 – 조인 선택률(js, join selectivity)**

- **조인 선택률**(js) : 카티션곱 크기(투플 수)에 대비한 조인 결과 크기의 비율

$$js = \frac{조인\ 결과\ 크기}{카티션곱\ 크기} = \frac{|R\bowtie_{A=B}S|}{|R\times S|} = \frac{|R\bowtie_{A=B}S|}{|R| * |S|}$$

- 릴레이션 R의 투플 수가 a이고 S의 투플 수가 b이면, $0 \le |R\bowtie_{조인조건}S| \le a*b$
- 즉, $0 \le |(R\bowtie_{A=B}S)| \le |R| * |S|$

- 질의 최적화 : 질의문에서 **조인 선택률이 작은 것**부터 먼저 수행한다.
- 질의 최적화 : **가장 제한적인 선택 연산**을 먼저 실행되도록 한다.(학과.학과코드='CS')

- 카티션곱(cartesian product, ×)

이름
홍재연
이순신
장길산

×

과목
디비
소공

=

이름	과목
홍재연	디비
홍재연	소공
이순신	디비
이순신	소공
장길산	디비
장길산	소공

(3 * 2 = **6**)

정답 : ④