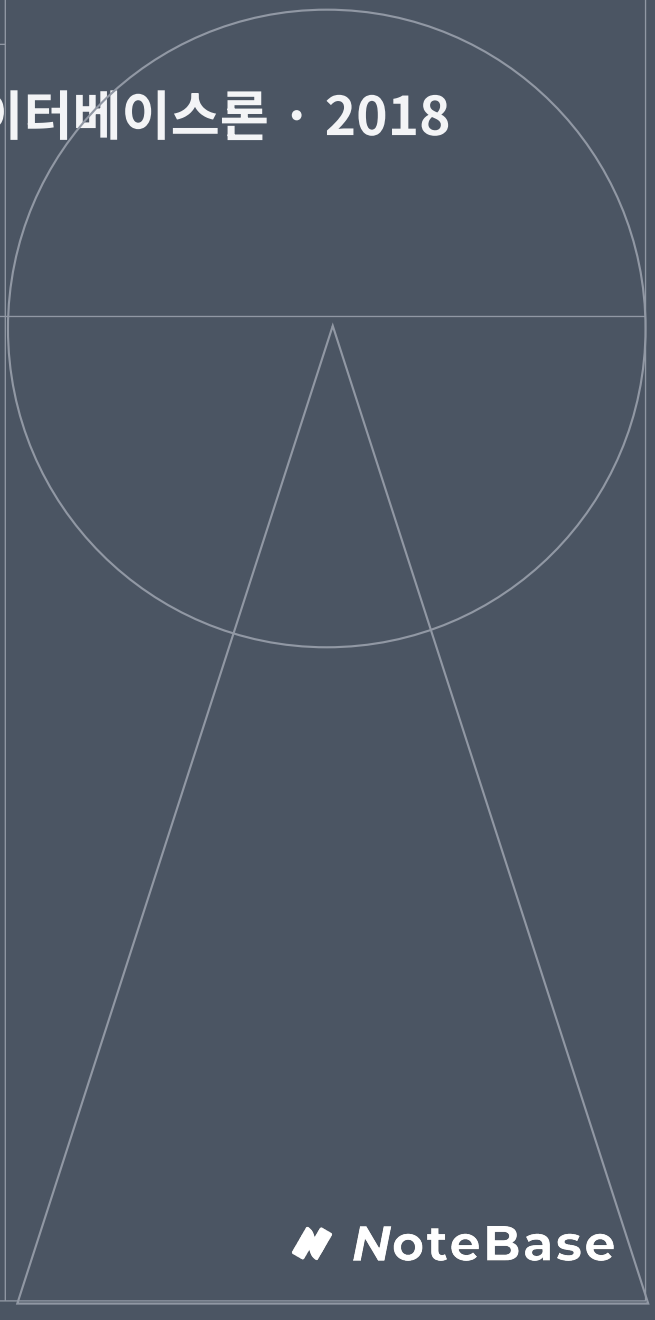
기출OX 해설집

공무원/경찰/경력채용 · 데이터베이스론 · 2018



NoteBase

기출OX 해설집

공무원/경찰/경력채용 · 데이터베이스론 · 2018

공무원/경찰/경력채용 · 데이터베이스론 · 2018

문 1. 데이터베이스 시스템의 주요 구성요소에 대한 설명 중 옳은 것은 모두 몇 개인가?

㉠ 질의어 처리기 - 대화식으로 입력된 고급 질의문을 처리한다.
㉡ DML 예비컴파일러 - 응용 프로그램 내에 삽입된 DML 명령문을 추출한다.
㉢ 시스템 카탈로그 - 스키마 정보 등의 메타데이터를 저장한다.
㉣ 트랜잭션 관리기 - 동시성 제어 및 회복 작업을 수행한다.

① 1개
② 2개
③ 3개
④ 4개

정답 4번

간단한 해설

질의 처리, 삽입 DML 추출, 메타데이터 저장, 동시성 제어·회복은 각각 제시된 구성요소의 기능이다. **네 설명 모두 역할이 올바르게 연결**되어 있다.

NoteBase

보기별 해설

㉠ O: 질의어 처리기는 사용자가 입력한 고급 질의문을 해석하고 실행 가능한 처리 절차로 변환하여 수행한다. 대화식 SQL 질의도 그 대상이며, **사용자의 질의를 실제 데이터 접근 작업으로 연결**하는 역할을 한다.

㉡ O: DML 예비컴파일러는 C 등의 호스트 언어로 작성된 응용 프로그램에서 **삽입된 DML 문장을 분리·추출**하여 처리할 수 있게 한다. 일반 호스트 언어 컴파일러가 그대로 처리하기 어려운 데이터베이스 명령을 구별하는 단계이다.

㉢ O: 시스템 카탈로그는 테이블·열·자료형·제약조건 등의 스키마 정보를 보관한다. 이는 사용자 업무 데이터 자체가 아니라 **데이터의 구조와 관리에 관한 메타데이터**이며, DBMS가 질의 처리와 객체 관리 등에 활용한다.

㉣ O: 트랜잭션 관리기는 여러 트랜잭션의 동시 실행을 조정하고 장애 발생 시 데이터베이스를 일관된 상태로 회복하도록 관리한다. **동시성 제어와 회복**은 트랜잭션의 격리성·원자성·지속성 등을 유지하는 핵심 기능이다.

㉠·㉡·㉢·㉣이 모두 옳으므로 총 4개이며, ④와 일치한다.

NoteBase

문 2. 데이터베이스 저장장치와 파일 시스템에 대한 설명 중 가장 적절하지 않은 것은?

① 디스크 접근 시간은 탐구시간(seek time)과 회전지연시간, 데이터 전송시간으로 구성된다.

② 데이터를 여러 개의 디스크에 스트리핑(striping)하면 중복을 통해 높은 신뢰성을 확보할 수 있다.

③ 버퍼는 디스크 블록의 복사본을 저장하기 위해 사용하는 메인 메모리의 일부분이다.

④ 검색키를 기초로 정렬된 순서를 갖는 파일은 순차파일(sequential file)이다.

정답 2번

간단한 해설

스트리핑은 데이터를 여러 디스크에 나누어 저장하는 것이며 **복사본이나 패리티를 만드는 중복 기법 자체는 아니다**. 신뢰성을 확보하려면 별도의 중복 저장 구조가 필요하다.

선지별 해설

① O: 회전식 디스크에서 데이터를 읽으려면 헤드를 해당 트랙으로 이동시키는 seek time, 원하는 섹터가 헤드 아래로 올 때까지 기다리는 회전지연시간, 실제 데이터를 옮기는 전송시간이 필요하다. 디스크 접근 시간을 설명하는 기본 구성 요소이다.

② X: 스트리핑은 데이터를 여러 디스크에 나누어 배치하여 병렬 입출력을 가능하게 하는 기법이다. **분산 배치 자체는 데이터 중복을 만들지 않는다**. 신뢰성을 높이려면 미러링이나 패리티 같은 별도의 중복 정보가 필요하며, 스트리핑만 사용하는 RAID 0에는 디스크 고장에 대비한 복구용 중복 정보가 없다.

③ O: 데이터베이스 버퍼는 디스크에서 읽은 블록·페이지의 복사본을 보관하는 메인 메모리 영역이다. 필요한 데이터가 버퍼에 있으면 디스크를 다시 읽지 않고 사용할 수 있어 입출력 횟수를 줄인다.

④ O: 순차파일은 검색키 값을 기준으로 레코드를 정렬하여 저장하는 파일 조직이다. 키 순서로 이어지는 레코드를 처리하거나 일정 범위의 레코드를 차례로 읽는 데 적합하다.

NoteBase

문 3. 해싱(hashing)에 관한 설명 중 가장 적절하지 않은 것은?

① 이상적인 해시 함수는 모든 탐색키 값을 서로 다른 버킷에 대응시킨다.

② 해싱을 이용하여 특정 탐색 조건을 만족하는 레코드들을 빠르게 접근할 수 있다.

③ 해시 함수가 탐색키를 균등하게 분배하면 버킷 오버플로가 발생할 확률이 증가한다.

④ 충돌(collision)이란 2개의 상이한 레코드가 같은 버킷으로 해싱되는 것을 의미한다.

정답 3번

간단한 해설

균등 분배는 일부 버킷으로 레코드가 집중되는 현상을 줄인다. 따라서 **오버플로 가능성을 증가시키는 것이 아니라 편중에 따른 위험을 낮춘다**.

선지별 해설

① O: 여기서 '이상적'은 저장할 서로 다른 탐색키가 충돌하지 않도록 각기 다른 버킷에 대응하는 목표 상태로 해석한다. 다만 **유한한 버킷에 모든 가능한 키를 충돌 없이 배정할 수 있다는 일반 보장은 아니다**. 실제 해시 함수에서는 균등한 분산이 중요하며, 특정 키 집합에 대한 충돌 없는 완전 해싱과 구별해야 한다.

② O: 탐색키에 해시 함수를 적용하여 해당 버킷을 찾으면 전체 레코드를 순서대로 살피지 않고 후보에 접근할 수 있다. 특히 **키 값이 같은지를 찾는 동등 조건 검색**에 적합하며, 키의 순서를 이용하는 범위 검색에 동일한 장점이 보장되지는 않는다.

③ X: 탐색키가 균등하게 분포하면 특정 버킷으로 레코드가 몰리는 현상이 줄어든다. 따라서 다른 조건이 같을 때 **분포의 편중에 따른 버킷 오버플로 위험은 감소**한다. 전체 레코드 수와 버킷 용량에 따라 오버플로 자체가 완전히 없어지는 것은 아니다.

④ O: 서로 다른 레코드가 해시 결과 같은 버킷에 배정되면 충돌이 발생한다. **같은 버킷으로의 대응**이 충돌이며, 그 버킷의 저장 용량까지 초과하는 오버플로와는 구별한다.

NoteBase

문 4. B-트리와 B^+-트리에 대한 설명 중 가장 적절한 것은?

① 같은 수의 킷값을 가지는 B-트리와 B^+-트리의 레벨은 항상 같다.

② B-트리에서 모든 리프 노드는 같은 레벨에 있다.

③ B^+-트리에서 인덱스 세트에 킷값이 있으면 그 킷값의 검색이 보장된다.

④ B^+-트리에서 킷값의 삭제는 항상 인덱스 세트와 순차 세트에서 동시에 일어난다.

정답 2번

간단한 해설

B-트리는 **모든 리프 노드를 같은 레벨에 유지**한다. B^+-트리의 내부 키는 탐색용 경계값이므로 실제 데이터의 존재 확인이나 삭제를 리프 노드의 처리와 구분해야 한다.

선지별 해설

① X: 트리의 높이는 키 개수만으로 정해지지 않는다. 노드에 담을 수 있는 키·포인터 수와 노드의 채움 정도, 레코드를 배치하는 방식에 따라 달라지므로, 같은 수의 키를 저장하더라도 B-트리와 B^+-트리의 레벨이 항상 같지는 않다.

② O: B-트리는 **모든 리프 노드가 같은 레벨에 있는 균형 다원 탐색 트리**이다. 삽입·삭제 시 노드 분할, 재분배, 병합 등을 통해 이 균형 조건을 유지한다.

③ X: B^+-트리의 인덱스 세트에 있는 키는 탐색 방향을 정하는 경계값이다. 데이터 항목을 삭제한 뒤에도 경계값으로 유효하면 내부 노드에 남을 수 있으므로, 내부 노드에서 키를 발견했다는 사실만으로 해당 데이터의 존재가 보장되지는 않는다. **실제 검색 성공 여부는 순차 세트인 리프 노드에서 확인**해야 한다.

④ X: 데이터 항목의 삭제는 리프 노드의 순차 세트에서 이루어지며, 인덱스 세트의 변경이 항상 함께 필요한 것은 아니다. 경계값 조정이나 재분배·병합이 필요할 때 관련 내부 노드를 수정한다. 따라서 두 세트에서 항상 동시에 삭제한다는 설명은 틀리다.

NoteBase

문 5. 네트워크를 통해 데이터베이스에 접속하는 환경에서는 데이터베이스 응용 시스템이 두 부분 또는 세 부분으로 나누어지는 계층 구조를 가진다. 다음 설명 중 가장 적절하지 않은 것은?

① 2-계층구조(two-tier architecture)에서 ODBC나 JDBC는 서버와 클라이언트 사이의 교신에 사용된다.

② 3-계층구조(three-tier architecture)에서 클라이언트는 일반적으로 전처리 시스템 역할을 한다.

③ 3-계층구조에서는 응용프로그램 서버(application server)가 데이터베이스와 교신한다.

④ 3-계층구조에서 비즈니스 로직(business logic)은 여러 클라이언트에 분산되어 있다.

정답 4번

간단한 해설

3-계층 구조는 **클라이언트의 표현 계층, 응용 서버의 비즈니스 로직, 데이터베이스 계층**을 분리한다. 핵심 업무 로직은 여러 클라이언트가 아니라 중간 응용 서버 계층에서 처리한다.

선지별 해설

① O. 2-계층 구조에서는 클라이언트 응용프로그램이 데이터베이스 서버에 직접 접속한다. **ODBC·JDBC 같은 데이터베이스 접근 인터페이스**와 드라이버를 이용해 질의를 보내고 결과를 받는다. 이들은 그 자체로 단일 네트워크 전송 프로토콜이라는 뜻은 아니다.

② O. 3-계층 구조의 클라이언트는 주로 **사용자 인터페이스와 입력·출력**을 담당하는 전단부 역할을 한다. 여기서 전처리 시스템이라는 표현은 요청을 받아 중간 계층에 전달하고 결과를 표시하는 역할로 이해하며, 모든 업무 규칙을 클라이언트가 처리한다는 뜻이 아니다.

③ O. 중간 계층의 **응용프로그램 서버**가 업무 처리를 수행하고 필요한 데이터 조회·변경을 데이터베이스 서버에 요청한다. 데이터베이스의 결과는 응용 서버를 거쳐 클라이언트에 전달된다.

④ X. 전형적인 3-계층 구조는 **비즈니스 로직을 중간 응용 서버 계층**에 분리한다. 이를 여러 클라이언트에 분산시키는 것이 특징이라는 설명은 잘못이다. 클라이언트에 일부 입력 검사가 존재하더라도 핵심 업무 로직의 계층 분리라는 원칙은 달라지지 않는다.

따라서 가장 적절하지 않은 것은 비즈니스 로직을 여러 클라이언트에 분산한다고 한 ④이다.

NoteBase

문 6. 두 릴레이션 R, S에 대한 관계 대수식 중 등호 관계가 항상 '참'인 것은?

① $R - S = S - R$

② $R \div S = S \div R$

③ $R \bowtie S = S \bowtie R$

④ $R \ltimes S = S \ltimes R$

정답 3번

간단한 해설

자연 조인은 공통 속성 값이 같은 튜플을 결합하므로 **두 릴레이션의 순서를 바꾸어도 결과가 같다**.
차집합·디비전·왼쪽 세미 조인은 입력의 좌우 역할이 달라 교환법칙이 성립하지 않는다.

선지별 해설

① X: 차집합은 **왼쪽 릴레이션에만 있는 튜플**을 남긴다. 같은 속성 A를 갖는 R={(1)}, S={(2)}라면 R−S={(1)}이고 S−R={(2)}이므로 교환법칙이 성립하지 않는다.

② X: 디비전은 나누는 릴레이션의 **모든 튜플과 대응하는 값**을 나누어지는 릴레이션에서 찾는 연산이다. 두 입력의 역할이 다르며, R(A,B)를 S(B)로 나누면 결과는 A 속성을 갖지만 순서를 바꾼 연산은 같은 조건으로 정의되지 않을 수도 있다. 따라서 교환법칙이 성립하지 않는다.

③ O: 자연 조인은 두 릴레이션의 **공통 속성 값이 같은 튜플을 결합**한다. 이 조건은 R과 S의 순서를 바꾸어도 같으므로 같은 결과 릴레이션을 얻는다. 관계 모델에서 속성의 표시 순서는 릴레이션의 동일성을 바꾸지 않는다.

④ X: 왼쪽 세미 조인은 다른 릴레이션과 조인되는 튜플 중 **왼쪽 릴레이션의 속성만**반환한다. R⋉S는 R의 속성을, S⋉R는 S의 속성을 반환하므로 두 결과가 항상 같지는 않다.

NoteBase

문 7. 다음 두 릴레이션 R과 S를 애트리뷰트 B로 조인(join), 애트리뷰트 B로 완전외부조인(full outer join), 외부합집합(outer union)하였다. 각 결과 릴레이션의 카디널리티(cardinality)로 가장 적절한 것은?

R

A	B
w	2
x	2
y	3
z	4

S

B	C
1	p
2	q

① 조인: 1, 완전외부조인: 5, 외부합집합: 4

② 조인: 2, 완전외부조인: 5, 외부합집합: 6

③ 조인: 2, 완전외부조인: 3, 외부합집합: 4

④ 조인: 2, 완전외부조인: 3, 외부합집합: 6

정답 2번

간단한 해설

B=2에서 2×1개의 튜플이 결합하므로 조인은 **2개**이다. 완전외부조인은 미대응 튜플 3개를 더해 **5개**, 외부합집합은 양쪽 튜플을 보존해 **6개**이므로 ②이다.

NoteBase

계산 과정

카디널리티는 **릴레이션에 포함된 튜플의 수**이다. 그림에서 R은 (w,2), (x,2), (y,3), (z,4)의 4개 튜플이고, S는 (1,p), (2,q)의 2개 튜플이다.

B 조인: 두 릴레이션에서 같은 B값은 2이다. R의 B=2 튜플은 2개이고 S의 B=2 튜플은 1개이므로, **2×1=2개의 결합**이 만들어진다. 결과를 (A,B,C)로 쓰면 (w,2,q), (x,2,q)이다. B값이 같아도 A값이 다르므로 두 결과 튜플은 중복이 아니다.

완전외부조인: 조인 결과 2개에 어느 쪽에도 대응하지 못한 튜플을 추가한다. R의 (y,3), (z,4)는 C를 NULL로, S의 (1,p)는 A를 NULL로 채운다. 따라서 **2+2+1=5개**이다.

외부합집합: 서로 다른 속성 부분을 NULL로 보완하면서 양쪽의 튜플을 합친다. R 쪽은 (w,2,NULL), (x,2,NULL), (y,3,NULL), (z,4,NULL), S 쪽은 (NULL,1,p), (NULL,2,q)가 된다. **B값이 같다는 이유로 두 튜플을 결합하지 않으므로** 모두 6개이다.

따라서 조인 2개, 완전외부조인 5개, 외부합집합 6개인 ②가 정답이다.

NoteBase

문 8. (가)의 SQL 문은 테이블 PROF와 STUDENT의 정의를 일부 보인 것이다. (나)의 STUDENT 테이블에서 잘못된 행(row)을 모두 고른 것은?(단, (가)에서 제시된 내용만 고려할 것)

(가) SQL

```
CREATE TABLE PROF (
    ...
    PRIMARY KEY(PID));
CREATE TABLE STUDENT (
    ...
    PRIMARY KEY(SNO),
    FOREIGN KEY(ADVISOR) REFERENCES PROF(PID),
    CHECK(YEAR > 0 AND YEAR < 5));
```

(나) 테이블

PROF

PID	PNAME
A	김교수
B	이교수
C	박교수

STUDENT

	SNO	SNAME	YEAR	ADVISOR
㉠	1	일학생	3	A
㉡	2	이학생	2	NULL
㉢	3	삼학생	1	D
㉣	4	NULL	2	B
㉤	5	오학생	5	C
㉥	NULL	육학생	4	B

(테이블에서 NULL은 널 값을 의미함)

① ㉠㉡㉣

② ㉡㉢㉤

③ ㉢㉤㉥

④ ㉣㉤㉥

정답 3번

간단한 해설

잘못된 행은 **존재하지 않는 외래키 D, 범위를 벗어난 YEAR 5, 기본키 SNO의 NULL**을 가진 행이다. 일반 열이나 별도 NOT NULL이 없는 외래키의 NULL은 구별해야 한다.

NoteBase

보기별 해설

㉠ O: SNO가 1로 다른 행과 중복되지 않고 NULL도 아니다. YEAR는 3으로 0보다 크고 5보다 작으며, ADVISOR의 A도 PROF의 PID에 존재한다. **제시된 세 제약을 모두 만족**한다.

㉡ O: SNO 2와 YEAR 2는 기본키·CHECK 조건을 만족한다. ADVISOR에는 별도 NOT NULL 조건이 없으므로 **외래키 열의 NULL 자체는 위반이 아니다**. 외래키라고 해서 반드시 값이 있어야 하는 것은 아니다.

㉢ X: ADVISOR가 D이지만 PROF의 PID에는 A·B·C만 존재한다. NULL이 아닌 외래키 값은 참조 대상 키와 일치해야 하므로 **존재하지 않는 D를 참조하여 참조 무결성을 위반**한다.

㉣ O: SNO 4는 유효하고 YEAR 2와 ADVISOR B도 제약을 만족한다. SNAME에는 제시된 NOT NULL 조건이 없으므로 **이름이 NULL이라는 이유만으로 잘못된 행이 되지 않는다**.

㉤ X: YEAR가 5이므로 YEAR가 0보다 크고 5보다 작아야 한다는 CHECK 조건에서 후자를 만족하지 못한다. 경계값 5는 포함되지 않으므로 **YEAR의 CHECK 제약 위반**이다.

㉥ X: SNO는 STUDENT의 기본키로 선언되어 있다. 기본키는 중복뿐 아니라 NULL도 허용하지 않으므로 **SNO의 NULL이 개체 무결성을 위반**한다. YEAR 4와 ADVISOR B가 유효해도 이 위반은 해소되지 않는다.

잘못된 행은 ㉢·㉤·㉥이므로 ③과 일치한다.

NoteBase

문 9. SQL 뷰(view)에 대한 다음 설명 중 옳은 것으로만 이루어진 것은?

㉠ 뷰는 데이터의 접근을 제어함으로써 보안을 제공할 수 있다.
㉡ 뷰는 데이터 독립성 제공과 무관하다.
㉢ 두 개 이상의 테이블을 조인하여 뷰를 생성할 수 있다.
㉣ ALTER VIEW 구문을 이용하여 뷰 정의를 변경한다.
㉤ 뷰에 대한 검색 및 갱신 연산은 아무런 제약이 없다.

① ㉠㉢
② ㉠㉡
③ ㉡㉣
④ ㉢㉤

정답 1번

간단한 해설

뷰는 **접근 제어와 논리적 데이터 독립성**을 지원하며 여러 테이블의 조인으로 정의할 수 있다. 다만 갱신에는 제약이 있고, 정의를 바꾸는 구문은 DBMS에 따라 다르다.

NoteBase

보기별 해설

㉠ O. 뷰로 필요한 행·열만 노출하고 적절한 접근 권한을 부여하면 **사용자가 볼 수 있는 데이터를 제한**할 수 있다. 따라서 뷰는 보안 수단으로 활용될 수 있다.

㉡ X. 뷰는 사용자에게 제공하는 외부 관점을 기초 테이블의 구조와 분리하여 **논리적 데이터 독립성**을 지원할 수 있다. 기초 구조가 바뀌어도 뷰의 인터페이스를 유지하도록 정의를 조정할 수 있으므로 독립성과 무관하지 않다.

㉢ O. 뷰의 정의에는 **여러 테이블을 조인한 SELECT 질의**를 사용할 수 있다. 다만 조인으로 뷰를 만들 수 있다는 사실이 그 뷰를 제한 없이 갱신할 수 있다는 뜻은 아니다.

㉣ DBMS에 따라 다름. SQL Server 등은 **ALTER VIEW로 정의 질의를 변경**할 수 있다. 반면 PostgreSQL에서는 정의 질의 변경에 CREATE OR REPLACE VIEW를 사용하고 ALTER VIEW는 다른 속성 변경에 사용한다. 따라서 ALTER VIEW가 모든 DBMS에서 불가능하다고 단정해서는 안 된다. 이 항목이 포함된 선택지는 독립적으로 틀린 ㉡도 포함하므로 정답이 될 수 없다.

㉤ X. 뷰에는 접근 권한 등의 제약이 있고, 특히 **갱신 가능성은 뷰 정의와 DBMS 규칙**에 제한된다. 집계·그룹화 등이 포함된 뷰는 일반적으로 직접 갱신하기 어려우며, 모든 뷰의 검색·갱신에 아무 제약이 없다는 설명은 틀렸다.

따라서 옳은 진술인 ㉠·㉢으로만 이루어진 선택지는 ①이다.

NoteBase

문 10. 다음 EMP 테이블에 ㉠~㉣의 SQL 문을 차례대로 모두 실행한 최종 결과로 가장 적절한 것은?

EMP

NAME	DEPT	SALARY
김직원	1	200
이직원	2	100
박직원	2	300

```
㉠ INSERT INTO EMP VALUES ('정직원', 2, 200);
㉡ UPDATE EMP SET DEPT = 1
   WHERE NAME LIKE '박%';
㉢ INSERT INTO EMP VALUES ('최직원', 3, 400);
㉣ SELECT DEPT, AVG(SALARY) AS ASAL
   FROM EMP
   GROUP BY DEPT HAVING COUNT(*) >= 2
   ORDER BY DEPT ASC;
```

① DEPT: 1, ASAL: 250
DEPT: 2, ASAL: 150

② DEPT: 2, ASAL: 150
DEPT: 1, ASAL: 250

③ DEPT: 1, ASAL: 250
DEPT: 2, ASAL: 150
DEPT: 3, ASAL: 400

④ DEPT: 3, ASAL: 400
DEPT: 2, ASAL: 150
DEPT: 1, ASAL: 250

정답 1번

간단한 해설

박직원을 1부서로 옮긴 뒤 부서별로 집계하면 **1부서 평균은 250, 2부서 평균은 150**이다. 3부서는 1명이라 HAVING 조건에서 제외되며, 남은 결과는 부서 번호 오름차순으로 출력된다.

NoteBase

계산 과정

㉠ 삽입: 정직원이 DEPT=2, SALARY=200으로 추가된다. 이 단계에서 2부서에는 이직원 100, 박직원 300, 정직원 200이 있다.

㉡ 갱신: NAME LIKE '박%'는 이름이 '박'으로 시작하는 행을 찾는다. 따라서 **박직원의 DEPT가 2에서 1로 변경**되고 급여 300은 그대로 유지된다.

㉢ 삽입: 최직원이 DEPT=3, SALARY=400으로 추가된다. 최종 그룹은 1부서의 김직원·박직원, 2부서의 이직원·정직원, 3부서의 최직원이다.

㉣ 그룹 집계: GROUP BY DEPT로 부서별로 묶고 **HAVING COUNT(*) ≥ 2**를 적용한다. 1·2부서는 각각 2명이므로 남고, 3부서는 1명이므로 제외된다. 1부서의 평균 급여는 $\frac{200+300}{2}=250$, 2부서는 $\frac{100+200}{2}=150$이다. 문제에서 급여의 별도 단위는 제시하지 않았다.

정렬 및 결과: ORDER BY DEPT ASC는 부서 번호를 오름차순으로 정렬한다. 따라서 최종 출력은 **(DEPT, ASAL)=(1,250), (2,150)** 순서이며 ①과 일치한다. ②는 행 순서가 반대이고, ③·④는 제외되어야 할 3부서까지 포함한다.

NoteBase

문 11. 데이터베이스 커서(cursor)를 사용할 때 명령어 순서로 가장 적절한 것은?

① DECLARE CURSOR - FETCH - OPEN - CLOSE

② DECLARE CURSOR - OPEN - FETCH - CLOSE

③ FETCH - DECLARE CURSOR - OPEN - CLOSE

④ FETCH - OPEN - DECLARE CURSOR - CLOSE

정답 2번

간단한 해설

커서는 처리 대상을 선언하고 연 뒤에 행을 가져와야 한다. **FETCH는 열린 커서에 대해 수행**하며, 처리가 끝나면 커서를 닫는다.

올바른 순서

DECLARE CURSOR → OPEN → FETCH → CLOSE

DECLARE CURSOR: 커서의 이름과 처리할 조회문을 선언한다. 어떤 결과를 커서로 다룰 것인지 먼저 정의해야 커서를 열 수 있다.

OPEN: 선언된 조회문을 실행하여 커서를 사용할 수 있는 상태로 연다. **행을 가져오는 FETCH보다 먼저 실행**해야 한다.

FETCH: 열린 커서의 결과에서 행을 가져와 처리한다. 여러 행을 차례로 처리하려면 필요한 만큼 반복한다.

CLOSE: 행 처리를 마친 뒤 커서를 닫고 현재 결과 집합에 연결된 자원을 해제한다. 커서 선언 자체를 제거하는 명령과는 구별된다.

NoteBase

문 12. 데이터베이스 설계 단계에서 필요한 작업에 대한 설명으로 가장 적절하지 않은 것은?

① 요구조건 분석 단계 - 데이터 처리과정에서의 특별한 제약조건 수집

② 개념적 설계 - ER 다이어그램으로 특정 DBMS에서 제공되는 정보 구조 표현

③ 논리적 설계 - 트랜잭션의 전체적 골격을 개발하고 인터페이스 정의

④ 물리적 설계 - 인덱싱 기법, 레코드 집중화 및 파일 조직방법 선택

정답 2번

간단한 해설

개념적 설계의 ER 다이어그램은 **특정 DBMS에 종속되지 않는** 정보 구조를 표현한다. 특정 DBMS가 지원하는 데이터 모델로 옮기는 작업은 논리적 설계에 해당한다.

선지별 해설

① O: 요구조건 분석에서는 사용자가 필요로 하는 데이터와 처리 작업을 조사하고, 업무 규칙이나 처리 과정에서 지켜야 할 제약조건을 수집한다. 이러한 요구사항이 이후 설계의 기준이 된다.

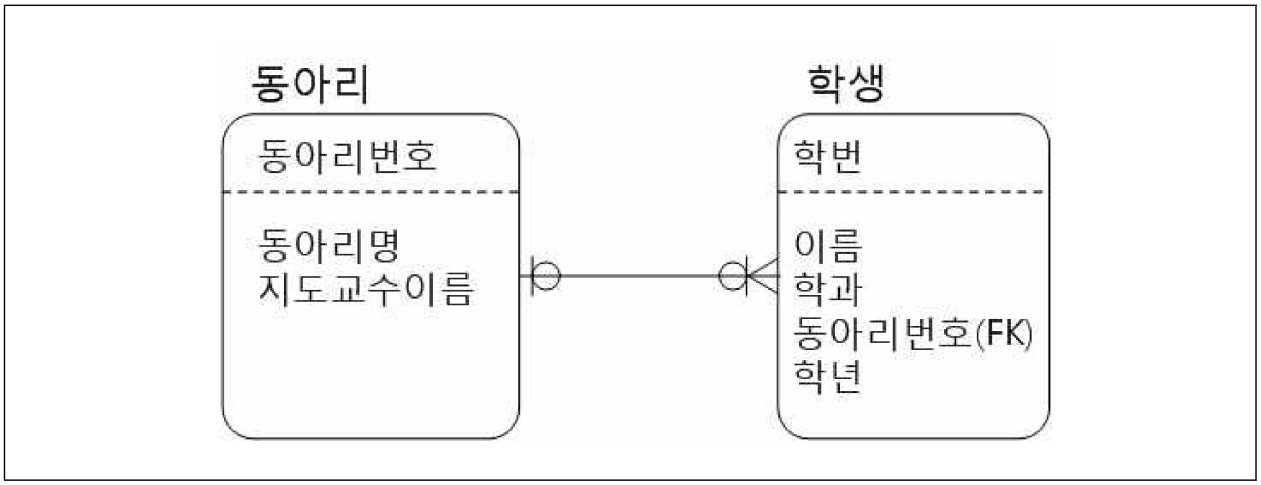
② X: 개념적 설계는 개체·속성·관계와 제약조건을 ER 다이어그램 같은 고수준 모델로 표현하며, **특정 DBMS에 독립적**이다. 이를 선택한 DBMS가 지원하는 데이터 모델의 구조로 변환하는 것은 논리적 설계에 해당한다.

③ O: 논리적 설계에서는 개념 스키마를 목표 데이터 모델로 변환하는 작업과 함께 **트랜잭션 인터페이스를 구체화**한다. 앞 단계에서 파악한 입력·출력과 기능을 바탕으로 응용 프로그램의 처리 골격과 데이터 접근 인터페이스를 설계하는 취지이다. 실행 가능한 세부 프로그램을 완성하는 단계와는 구별된다.

④ O: 물리적 설계에서는 실제 저장과 접근의 효율을 고려하여 파일 조직과 인덱스를 선택한다. 함께 접근하는 레코드를 가까이 배치하는 집중화도 **물리적 저장 구조**를 결정하는 작업이다.

NoteBase

문 13. 다음 ER 다이어그램은 정보공학(IE) 표기 방식으로 표현한 것이다. 이에 대한 설명 중 가장 적절하지 않은 것은?



① 학생 개체(entity)가 부모 개체이다.

② 학생은 최대 하나의 동아리에만 참여할 수 있다.

③ 특정 동아리에는 참여 학생이 한 명도 없을 수 있다.

④ 동아리 개체와 학생 개체 사이의 관계는 비식별 관계이다.

정답 1번

간단한 해설

학생의 동아리번호(FK)가 동아리의 키를 참조한다. 따라서 **부모는 동아리, 자식은 학생**이며 이를 반대로 설명하면 부적절하다.

NoteBase

선지별 해설

① X: 학생이 아니라 **동아리가 부모 개체**이다. 동아리의 식별자인 동아리번호를 학생이 외래키로 참조하므로, 참조되는 쪽이 동아리이고 참조하는 자식 쪽이 학생이다.

② O: 동아리 쪽 끝의 원과 한 줄 표시는 학생 한 명에 대응하는 동아리가 0개 또는 1개라는 뜻이다. 따라서 참여하지 않는 학생은 있을 수 있지만, **한 학생이 참여할 수 있는 동아리는 최대 하나**이다.

③ O: 학생 쪽 끝의 원과 까마귀발 표시는 동아리 하나에 대응하는 학생이 0명 이상일 수 있다는 뜻이다. 최소 참여 수가 0이므로 **참여 학생이 없는 동아리도 허용**된다.

④ O: 학생의 식별자 영역에는 학번이 있고, 동아리번호(FK)는 그 아래 일반 속성 영역에 있다. 즉 부모의 키가 자식의 식별자에 포함되지 않아 **학번만으로 학생을 식별하는 비식별 관계**이다. 관계의 식별 여부는 외래키의 존재 자체가 아니라 자식 식별자에 포함되는지로 판단한다.

NoteBase

문 14. 제 3 정규형인 릴레이션을 보이스코드 정규형(BCNF)으로 변환하기 위한 작업으로 가장 적절한 것은?

① 원자 값이 아닌 도메인을 분해

② 부분 함수 종속 관계 제거

③ 이행적 함수 종속 관계 제거

④ 결정자가 후보키가 아닌 함수 종속 제거

정답 4번

간단한 해설

BCNF는 **모든 비자명한 함수 종속의 결정자가 슈퍼키**여야 한다. 이 조건을 위반하는 종속을 기준으로 릴레이션을 분해하여 3정규형보다 강한 조건을 만족시킨다.

핵심 풀이

BCNF의 엄밀한 조건은 모든 **비자명한 함수 종속 X → Y에서 X가 슈퍼키**여야 한다는 것이다. 3정규형에는 종속되는 속성이 후보키 구성 속성인 경우 등의 예외가 허용되지만, BCNF는 이러한 예외 없이 결정자가 전체 튜플을 식별할 수 있어야 한다.

따라서 슈퍼키가 아닌 결정자로 인한 위반을 찾고, 그 종속에 따라 릴레이션을 **무손실 분해**하여 각 릴레이션이 BCNF를 만족하도록 한다. 함수 종속이라는 사실 자체를 임의로 없애는 것이 아니라, 위반이 발생하지 않도록 스키마를 분해하는 작업이다.

선지의 “결정자가 후보키가 아닌 함수 종속 제거”는 이 정규화 단계를 가리키는 전형적인 요약이다. 다만 정확한 기준은 후보키의 최소성까지 요구하는 것이 아니라 **슈퍼키인지 여부**이므로, 후보키를 포함한 더 큰 결정자라는 이유만으로 BCNF 위반은 아니다. 원자 값으로의 정리는 1정규형, 비주요 속성의 부분 종속 제거는 2정규형, 이행 종속 해소는 일반적인 3정규형 설명과 관련된다.

따라서 3정규형에서 BCNF로 강화하는 작업에 가장 해당하는 것은 ④이다.

NoteBase

문 15. 사용자 Hong이 생성한 학과 테이블에 대해 다음과 같은 권한부여 명령이 수행되었다. 가장 적절하지 않은 것은?

Hong: GRANT SELECT, INSERT, UPDATE ON 학과
TO Park WITH GRANT OPTION;
Park: GRANT SELECT, UPDATE ON 학과 TO Lee;
Hong: REVOKE UPDATE ON 학과 FROM Park CASCADE;

① Park은 학과 테이블에 대해 SELECT권한이 있다.

② Park은 학과 테이블에 대해 INSERT권한이 있다.

③ Lee는 학과 테이블에 대해 SELECT권한이 있다.

④ Lee는 학과 테이블에 대해 UPDATE권한이 있다.

정답 4번

간단한 해설

CASCADE는 회수되는 권한에 의존해 다른 사용자에게 부여된 권한까지 연쇄적으로 회수한다. 따라서 **Park의 UPDATE를 회수하면 Lee의 UPDATE도 사라지지만**, SELECT와 INSERT처럼 회수 대상이 아닌 권한은 그대로 남는다.

선지별 해설

① O: Hong은 Park에게 SELECT·INSERT·UPDATE를 부여한 뒤 **UPDATE만 회수**했다. 따라서 Park의 SELECT 권한은 유지되어 학과 테이블을 조회할 수 있다.

② O: INSERT는 Hong이 Park에게 부여한 권한이며 마지막 REVOKE의 회수 대상에 포함되지 않는다. 따라서 Park의 **INSERT 권한은 유지**된다.

③ O: Park은 WITH GRANT OPTION으로 받은 SELECT를 Lee에게 부여할 수 있다. Hong이 회수한 것은 UPDATE이므로 **Lee의 SELECT 권한도 유지**된다.

④ X: Lee의 UPDATE 권한은 Hong→Park→Lee로 이어지는 부여 경로에 의존한다. Hong이 Park의 UPDATE를 **CASCADE로 회수**하면 그 권한에 의존하여 Lee에게 부여된 UPDATE도 함께 회수된다. 문제에 제시된 경로만 고려하면 Lee에게 UPDATE 권한은 남지 않는다.

NoteBase

문 16. 다음 중 무결성 규정(integrity rule)에 포함될 요소로 가장 적절하지 않은 것은?

① 규정 이름 - 규정을 참조할 때 사용하는 식별자

② 규정 대상 - 트랜잭션 사용자 및 데이터 접근 권한 명세

③ 제약 조건 - 데이터가 만족해야 될 제약조건

④ 위반 조치 - 제약조건 위반이 발견되었을 때의 대응조치 명세

정답 2번

간단한 해설

무결성 규정의 대상은 **규칙을 적용할 데이터**이다. 사용자와 접근 권한을 정하는 것은 데이터의 유효성을 검사하는 무결성 규정이 아니라 보안·권한 관리의 문제이다.

선지별 해설

① O: 규정 이름은 개별 무결성 규정을 구별하고 참조하기 위한 식별자이다. 제약을 관리하거나 오류가 발생한 규정을 확인할 때 사용할 수 있다.

② X: 규정 대상은 **무결성 규정이 적용되는 데이터의 범위**를 명세한다. 예를 들어 특정 릴레이션이나 속성 등이 대상이 된다. 트랜잭션 사용자와 데이터 접근 권한의 명세는 누가 어떤 작업을 할 수 있는지를 정하는 보안·권한 관리에 해당한다.

③ O: 제약 조건은 대상 데이터가 만족해야 하는 규칙이다. 예를 들어 값의 허용 범위, 키의 유일성, 참조 관계의 일관성 등을 조건으로 정한다.

④ O: 위반 조치는 제약 조건을 위반하는 변경이 발생했을 때 어떻게 대응할 것인지를 명세한다. 위반한 변경을 거부하거나, 정해진 참조 동작에 따라 관련 데이터를 처리하는 방식 등이 있다.

NoteBase

문 17. 데이터 무결성을 보장하기 위해 데이터베이스 시스템이 지원해야 하는 트랜잭션 특성으로 가장 적절하지 않은 것은?

① 원자성(atomicity)

② 지속성(durability)

③ 고립성(isolation)

④ 효율성(efficiency)

정답 4번

간단한 해설

트랜잭션의 기본 특성인 **ACID는 원자성·일관성·고립성·지속성**이다. 효율성은 성능 목표이지 이 네 가지 무결성 관련 특성 중 하나가 아니다.

개념 구분

원자성(Atomicity): 트랜잭션의 작업은 **전부 반영되거나 전혀 반영되지 않아야 한다**. 일부 변경만 남아 불완전한 상태가 되는 것을 막는 성질이다.

일관성(Consistency): 올바른 트랜잭션은 데이터베이스의 무결성 제약을 보존하여 **일관된 상태에서 다른 일관된 상태로**이행시킨다. 이는 ACID의 C에 해당한다.

고립성(Isolation): 동시에 실행되는 트랜잭션이 서로 부적절하게 간섭하지 않도록 실행을 격리하는 성질이다. 완전한 고립성은 직렬 실행과 동등한 결과를 목표로 하며, 실제 시스템에서는 격리 수준에 따라 허용되는 동시 실행 현상이 달라질 수 있다.

지속성(Durability): 성공적으로 커밋한 트랜잭션의 결과는 **장애 후에도 보존**되어야 한다. 실행 도중의 전부 또는 전무를 뜻하는 원자성과 달리, 완료가 확정된 결과의 유지에 초점이 있다.

효율성(Efficiency): 처리 속도나 자원 사용에 관한 성능 목표이다. 중요한 설계 고려 사항이지만 ACID의 네 가지 트랜잭션 특성에는 포함되지 않으므로 ④가 해당한다.

NoteBase

문 18. 데이터베이스 로그(log)를 이용한 회복(recovery) 기법에 대한 설명 중 가장 적절하지 않은 것은?

① 지연 갱신 회복 기법은 UNDO와 REDO 연산이 필요하지 않다.

② 지연 갱신 회복 기법을 사용하면 로그 레코드에 변경 이전 값(old value)을 저장할 필요 없다.

③ 즉시 갱신 회복 기법은 UNDO 연산이 필요하다.

④ 즉시 갱신 회복 기법을 사용하면 로그 레코드에 변경 이전 값(old value)이 저장된다.

정답 1번

간단한 해설

지연 갱신은 미완료 변경을 데이터베이스에 반영하지 않으므로 UNDO는 불필요하지만, 완료된 변경의 누락을 복구할 **REDO는 필요할 수 있다**.

선지별 해설

① X: 지연 갱신은 미완료 트랜잭션의 변경을 데이터베이스에 반영하지 않아 UNDO가 필요하지 않다. 그러나 완료된 변경이 장애로 아직 저장되지 않았을 수 있으므로 **REDO는 필요할 수 있다**. 두 연산이 모두 불필요하다는 설명이 틀렸다.

② O: 지연 갱신에서는 완료되지 않은 트랜잭션 때문에 디스크의 이전 값이 덮어써지지 않는다. 취소를 위해 되돌릴 변경이 없으므로 **UNDO용 변경 이전 값을 로그에 저장할 필요가 없다**. 완료된 변경을 재수행할 수 있도록 변경 이후 값은 기록한다.

③ O: 즉시 갱신은 트랜잭션이 완료되기 전에도 변경 내용을 데이터베이스에 반영할 수 있다. 이후 그 트랜잭션이 실패하면 **이미 반영된 미완료 변경을 UNDO로 취소**해야 하므로, 회복 기법에 UNDO 기능이 필요하다.

④ O: 즉시 갱신에서 이전 값이 덮어써진 뒤에도 취소할 수 있으려면 **변경 이전 값을 로그에 보존**해야 한다. 전형적인 UNDO/REDO 로그 방식은 이전 값과 이후 값을 함께 기록하여 취소와 재수행에 각각 사용한다.

NoteBase

문 19. 다음은 어느 트랜잭션에 포함된 lock()와 unlock() 연산이 나타난 순서를 보인 것이다. 2단계 로킹 규약(2 phase locking protocol)을 준수한 것은?

① lock(x); lock(y); lock(z); unlock(y); unlock(x); unlock(z);

② lock(x); unlock(x); lock(y); lock(z); unlock(y); unlock(z);

③ lock(y); unlock(y); lock(x); unlock(x); lock(z); unlock(z);

④ lock(y); lock(x); unlock(y); unlock(x); lock(z); unlock(z);

정답 1번

간단한 해설

2단계 로킹은 잠금을 얻는 성장 단계와 잠금을 해제하는 축소 단계로 나뉜다. **한 번이라도 잠금을 해제한 뒤에는 새로운 잠금을 획득할 수 없다**.

선지별 해설

① O: lock(x), lock(y), lock(z)로 필요한 잠금을 모두 획득한 다음 unlock(y), unlock(x), unlock(z)로 해제한다. **첫 해제 이후에는 새로운 잠금 획득이 없으므로** 성장 단계와 축소 단계가 분리되어 있다. 잠금 해제 순서가 획득 순서의 역순이어야 하는 것은 아니다.

② X: unlock(x)로 잠금을 해제한 뒤 lock(y)와 lock(z)를 수행한다. 첫 해제부터는 축소 단계이므로 새 잠금을 얻을 수 없다. y와 z의 잠금도 첫 unlock 이전에 획득해야 한다.

③ X: unlock(y) 뒤에 lock(x)가 있고, unlock(x) 뒤에도 lock(z)가 있다. 잠금 획득과 해제를 번갈아 수행하는 방식은 2단계 로킹이 아니다. 필요한 잠금 획득을 마친 후 해제를 시작하도록 바꾸어야 한다.

④ X: x와 y의 잠금을 해제한 뒤 lock(z)를 수행하므로 축소 단계에서 다시 잠금을 얻는다. lock(z)를 첫 unlock(y)보다 앞에 두어야 2단계 로킹 규약을 준수한다.

NoteBase

문 20. 다음 내용이 설명하는 데이터베이스 응용 분야로 가장 적절한 것은?

대규모 데이터베이스를 반자동으로 분석하여 유용한 패턴을 찾는 과정을 의미하며, 과거의 예를 바탕으로 한 값의 예측이나 구매 행위 간의 연관성 분석, 데이터 객체 간의 자동 클러스터링 등 다양한 응용 사례가 있다.

① 데이터 마트
② 데이터 웨어하우스
③ 데이터 마이닝
④ 블록체인

정답 3번

간단한 해설

대규모 데이터에서 **패턴을 발견하고 예측·연관 분석·군집화를 수행**하는 과정은 데이터 마이닝이다.

핵심 풀이

데이터 마이닝은 대규모 데이터에서 **유용한 패턴과 관계를 발견**하는 분석 과정이다. 과거 사례를 이용한 예측, 함께 구매되는 상품 사이의 연관 규칙 탐색, 유사한 객체를 묶는 군집화가 모두 이에 해당한다.

데이터 웨어하우스는 분석을 위해 여러 출처의 데이터를 통합한 저장 기반이고, 데이터 마트는 특정 부서나 주제의 분석 요구에 맞춘 데이터 집합이다. 이들은 데이터 마이닝에 활용될 수 있지만, 제시문이 직접 설명하는 패턴 발견 과정 자체와는 구별된다. 블록체인은 블록을 연결하여 기록을 관리하는 분산원장 기술이다.

따라서 제시된 예측·연관성 분석·클러스터링을 포괄하는 것은 ③ 데이터 마이닝이다.

NoteBase