

16. SQL 뷰(view)

뷰는 하나 이상의 다른 테이블로부터 유도되어 만들어진 이름을 가진 '가상테이블'이다. 가상이라고 하는 이유는 실제 물리적으로 튜플을 갖는 테이블이 아니기 때문이다. DBMS는 뷰의 정의만을 시스템 카탈로그(데이터 사전)에 저장해 두었다가 실행시간에 참조한 테이블과 연동되어 뷰의 SQL 문장을 이용하여 테이블을 구축하여 보여 준다.

- 기본테이블 : 물리적으로 구현되어 실제 자료가 저장된다.
- 가상테이블 : 물리적으로 구현되지 않는다. 그 정의만 시스템에 저장된다.

1. 뷰의 생성 및 특징

다음은 뷰 생성 구문이다.

```

Create View 뷰_이름[(열_이름_리스트)]           //대괄호 []는 생략 가능한 부분
As Select 열_이름_리스트 From 테이블_이름
[Where 조건]
[Group By 열_이름][Having 조건]
[With Check Option];

```

- 기본테이블이 변경되면 뷰도 자동으로 변경된 내용을 보여주며,
- 거꾸로 뷰의 내용을 변경시키면 기본테이블의 내용이 변경된다. → 서로 거울 관계
- 기본테이블이 삭제되면 해당 기본테이블에서 생성된 모든 뷰도 같이 삭제된다.
- 한번 정의된 뷰는 기본테이블처럼 ALTER문으로 수정할 수 없다.(삭제 후 다시 생성)
- 뷰는 독자적인 인덱스를 가지지 못한다. 기본테이블에 생성된 인덱스를 이용한다.

// With Check Option 절

- 뷰에 대한 삽입이나 갱신 연산이 수행될 때 조건(점수>90)을 위배하면 실행을 거절한다.
- With Check Option 절은 Where 문의 조건을 만족하는 변경만 수용하도록 한다.

2 <http://cafe.daum.net/pass365>(홍재연)

// 수평 뷰 생성

학생

학번	학과	이름	학년	주소
1	전산	순돌	4	서울
2	토목	순돌	4	평양
3	건축	연재	1	개성
4	전산	갑순	2	수원
5	토목	순이	3	서울

기본키는 학번이다.

[질의 1] 학생 테이블로부터 학과가 전산과인 학생의 뷰를 생성하시오.

```
Create View 전산과
As Select * From 학생
Where 학과='전산';
```

↓ 실행 결과

전산과

학번	학과	이름	학년	주소
1	전산	순돌	4	서울
4	전산	갑순	2	수원

[질의 2] 학생 테이블로부터 학과는 전산 또는 토목이고 주소가 서울인 학생의 뷰를 생성하시오.

```
Create View 서울
As Select * From 학생
Where 학과 In('전산', '토목') And 주소='서울';
```

↓ 실행 결과

서울

학번	학과	이름	학년	주소
1	전산	순돌	4	서울
5	토목	순이	3	서울

• 수평 뷰는 이론적으로 검색뿐만 아니라 삽입, 삭제, 갱신이 가능한 뷰이다.(기본키 포함)

// 수직 뷰 생성

학생

학번	학과	이름	학년	주소
1	전산	순돌	4	서울
2	토목	순돌	4	평양
3	건축	연재	1	개성
4	전산	갑순	2	수원
5	토목	순이	3	서울

기본키는 학번이다.

[질의 1] 학생 테이블로부터 학번, 이름, 주소를 가지는 뷰를 생성하시오.

Create View 학생1(학번, 이름, 주소) // (학번, 이름, 주소)는 생략 가능

As Select 학번, 이름, 주소 From 학생;

↓ 실행 결과

학생1

학번	이름	주소
1	순돌	서울
2	순돌	평양
3	연재	개성
4	갑순	수원
5	순이	서울

학생1 : 기본키는 학번을 포함하는 뷰

- 학생1 : 기본키 학번을 포함하므로 삽입, 삭제, 갱신이 가능한 뷰이다.

[질의 2] 학생 테이블로부터 이름, 학년, 주소를 가지는 뷰를 생성하시오.

Create View 학생2(이름, 학년, 주소)

As Select 이름, 학년, 주소 From 학생;

↓ 실행 결과

학생2

이름	학년	주소
순돌	4	서울
순돌	4	평양
연재	1	개성
갑순	2	수원
순이	3	서울

학생2 : 기본키는 학번을 포함하지 않는 뷰

- 학생2 : 기본키 학번을 포함하지 않으므로 삽입, 삭제, 갱신이 불가능한 뷰이다.

4 <http://cafe.daum.net/pass365>(홍재연)

// 그룹 뷰 생성

- 그룹 뷰는 뷰 정의에서 Group By 절의 포함하는 뷰이다.
- 그룹 뷰는 관련 행을 그룹화하여, 각 그룹에 대해 데이터가 요약된 하나의 행을 생성한다.
- 그룹 뷰는 통계 뷰라고도 한다.

수강

학번	과목	점수	학점
1	디비	90	3
1	소공	80	2
2	디비	95	3
3	디비	75	3
3	소공	85	2
4	디비	90	3
5	디비	80	3

기본키는 학번과 과목이다.(합성키)

[질의] 수강 테이블로부터 각 학번별 수강 과목수, 총점, 평균을 가지는 뷰를 생성하시오.

```
Create View 통계(학번, 과목수, 총점, 평균)
As Select 학번, Count(*), Sum(점수), Avg(점수)
From 수강
Group By 학번
```

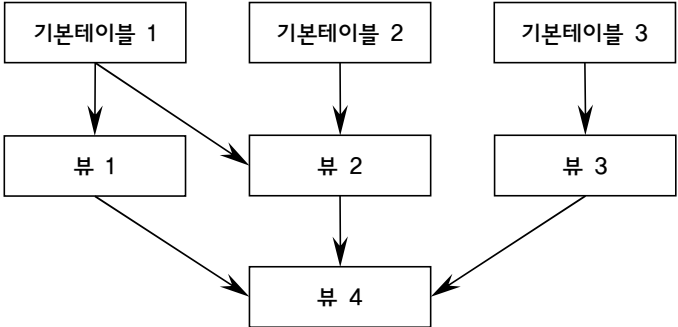
↓
↓ 실행 결과(학번을 기준으로 그룹화)
↓

통계

학번	과목수	총점	평균
1	2	170	85
2	1	95	95
3	2	160	80
4	1	90	90
5	1	80	80

- 그룹 뷰는 삽입, 삭제, 갱신이 불가능한 뷰이다.(검색은 제한이 없음)

// 기본테이블과 뷰의 관계



- 뷰1은 기본테이블1에서 정의되었고
 - 뷰2는 기본테이블1과 기본테이블2에서 정의되었고
 - 뷰3은 기본테이블3에서 정의되었고
 - 뷰4는 뷰1, 뷰2, 뷰3에서 정의되었음을 보여주고 있다.
- 뷰를 사용하면 여러 사용자가 여러 방식으로 기본테이블의 데이터를 볼 수 있다.

2. 뷰의 삭제

다음은 뷰 삭제 구문이다.

```
Drop view 뷰_이름 {Restrict | Cascade};
```

Restrict	<ul style="list-style-type: none"> • 삭제할 뷰에 종속된 뷰가 정의되어 있지 않을 때만 뷰를 삭제하도록 지시한다. • Restrict는 뷰 삭제에서 예기치 않은 부작용에 대한 예방책을 제공한다.
Cascade	<ul style="list-style-type: none"> • 그냥 삭제할 대상 뷰가 삭제되도록 지시한다. • 삭제할 뷰뿐만 아니라 • 삭제할 뷰에 종속되어 생성된 모든 뷰와 제약조건이 함께 제거된다.

- 뷰가 삭제될 때 삭제될 뷰가 기반으로 하는 기본테이블은 아무런 영향을 받지 않는다.
- 기본테이블이 삭제되면 이를 기반으로 생성된 뷰도 자동 삭제된다.(propagated destroy)

Drop view 우등생 Restrict:

→ '우등생' 뷰는 다른 곳에서 참조되고 있지 않을 때, 데이터베이스에서 삭제된다.

Drop view 우등생 Cascade:

→ '우등생' 뷰와 '우등생' 뷰를 이용하여 정의된 종속된 뷰도 자동으로 삭제된다.

3. 뷰의 조작

- ① 뷰에서 검색(Select)은 기본테이블처럼 똑같이 사용할 수 있다.
- ② 뷰에서 삽입, 삭제, 갱신은 상당한 제약이 있다. 기본키를 포함하고 있을 때 가능하다.

학생

학번	학과	이름	학년	주소
1	전산	순돌	4	서울
2	토목	순돌	4	평양
3	건축	연재	1	개성
4	전산	갑순	2	수원

기본키는 학번이다.

// 기본키(학번)를 포함하는 뷰

```
CREATE view 뷰1
As Select 학번, 이름, 주소
From 학생 Where 학년 = 4;
```

- 뷰1에 대한 삽입, 삭제, 갱신은 가능하다.
- 뷰1에 대한 연산은 곧 기본테이블에 대한 조작 연산이다.
- 하나의 레코드를 삽입하면 뷰에 없는 열의 값은 null이 된다.



학번	이름	주소
1	순돌	서울
2	순돌	평양

→ 2번 학생 순돌이가 제주로 이사를 갔을 때, 갱신 가능

• Update 뷰1 Set 주소 = '제주' Where 학번 = 2;

// 기본키(학번)를 포함하지 않은 뷰

```
CREATE view 뷰2
As Select 이름, 주소
From 학생 Where 학년 = 4;
```

- 뷰2에 대한 삽입, 삭제, 갱신은 불가능하다.
- 예 : 삽입은 기본키(학번)가 null이 되기 때문에 불가
- 예 : 삭제, 갱신은 그 대상을 정확히 알 수 없으므로 불가



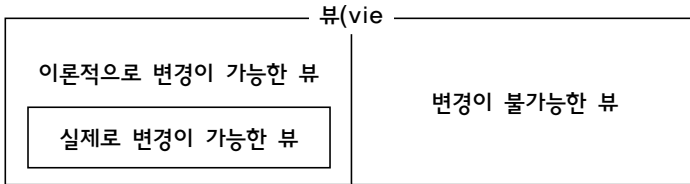
이름	주소
순돌	서울
순돌	평양

→ 2번 학생 순돌이가 제주로 이사를 갔을 때, 갱신 불가

• Update 뷰2 Set 주소 = '제주' Where 이름 = '순돌'; → 갱신 불가(같은 이름 충돌 발생)

4. 뷰의 제약

먼저, 일반적으로 뷰는 다음 3가지로 분류한다.



수강

학번	과목	점수	학점
1	디비	90	A
1	소공	80	B
2	디비	95	A
3	디비	75	C
3	소공	85	B

↓
 ↓ Create view 통계(학번, 과목수, 총점, 평균)
 ↓ As Select 학번, Count(*), Sum(점수), Avg(점수)
 ↓ From 수강 Group By 학번
 ↓

통계

학번	과목수	총점	평균
1	2	170	85
2	1	95	95
3	2	160	80

- 뷰가 데이터 통계 요약을 위해 정의되었을 때
- 변경 연산은 기본테이블에 많은 문제점을 발생시킨다.
- 예 : 평균을 이용하여, 각 과목이 몇 점인지? 알 수 없다.
- 학번 1 : 총점 = 디비 + 소공 = 90 + 80 = 170
- 학번 1 : 평균 = 총점 / 2 = 170 / 2 = 85

// 뷰에 대한 변경(삽입, 삭제, 갱신)이 불가능한 경우

- 뷰의 열이 상수나 산술식(연산자, 함수가 사용된 식)으로 구성되면 변경할 수 없다.
 → 삭제 연산은 허용되지만, 삽입과 갱신 연산은 이 열에 대해서 허용되지 않는다.
- 집계함수, Distinct, Group by, Having이 사용되어 정의된 뷰는 변경할 수 없다.
 → 뷰에 대한 변경을 어떤 방식으로 기본테이블에 적용해야 할지 결정할 수 없다
- 2개 이상의 테이블이 연관되어 정의된 뷰는 변경할 수 없다.(조인이 적용된 경우)
 → 뷰에 대한 변경이 기본테이블에 대한 변경으로 정확하게 대응되지 않을 수 있다.
- 변경할 수 없는 뷰를 기초로 정의된 뷰를 이용하여 데이터를 변경할 수 없다.

5. 뷰의 장단점

// 장점

독립성	<ul style="list-style-type: none"> • 뷰는 논리적 데이터 독립성을 어느 정도 제공할 수 있다. • 뷰가 정의된 기본테이블이 변해도 뷰가 사용된 응용프로그램은 영향받지 않는다.
보안성	<ul style="list-style-type: none"> • 뷰는 데이터에 대한 접근제어를 통해 보안을 제공할 수 있다. • 뷰에 표현되지 않은 데이터는 안전하게 보호된다. • 뷰를 생성할 때 포함되지 않은 속성은 노출되지 않으므로 데이터는 안전하게 보호된다.
편리성	<ul style="list-style-type: none"> • 각 사용자가 요구하는 데이터만을 뷰로 정의하여 다루면 데이터 관리가 단순해진다. • 자주 사용되는 복잡한 쿼리를 미리 뷰로 정의해두고, 추후 쿼리는 간단한 형태로 표현, 검색 등이 가능하다. • 뷰는 여러 사용자의 상이한 응용이나 요구를 지원할 수 있어서 데이터 관리를 단순하게 한다.
융통성	<ul style="list-style-type: none"> • 업무규칙 변경이 빈번하여 응용프로그램 수정이 자주 발생하는 경우 • 즉, 업무규칙을 뷰로 생성해 두고 응용프로그램은 뷰에 접근한다. • 데이터 모델이 변경되어도 응용프로그램에는 영향을 미치지 않도록 할 수 있다. • 업무규칙 변경 : 은행에서 이자를 구하는 공식이 변경되는 경우
임시성	<ul style="list-style-type: none"> • 뷰는 일시적인 작업 활용에 이용된다.

// 단점

- 뷰는 그 정의를 변경할 수 없다.(삭제 후 다시 생성해야 함)
 - 즉, ALTER view문을 사용할 수 없다.
- 뷰는 삽입, 삭제, 갱신 연산에 제약이 있다.
 - 하나의 테이블 위에서 뷰가 정의되었으나 기본키가 포함되지 않은 뷰는 변경할 수 없다.
- 뷰는 독자적인 인덱스를 가지지 못한다. 기본테이블에 생성된 인덱스를 이용한다.
- 뷰는 질의 처리 성능을 떨어뜨린다.
 - 사용자의 뷰 질의에 대해서 시스템은 해당 기본테이블에 대한 질의로 변경해야 한다.
 - 뷰를 통해 최종적으로 기본테이블에 접근해야 한다.
 - 만약, 뷰가 다중 테이블 질의로 정의된다면 조인을 수행해야 하므로 성능이 떨어진다.
 - 뷰 활용도를 높이려면, 수행속도 영향을 받지 않도록 조건을 잘 부여할 수 있어야 한다.
- 뷰 활용 : 보안성, 편의성, 융통성, 활용성, 임시성

// 인라인 뷰(inline view)

- 인라인 뷰(inline view)는 FROM절에 사용된 **부속질의어**이다.
- FROM절에 사용된 **부속질의어** 결과가 하나의 테이블에 대한 뷰처럼 사용된다.

수강

학번	과목	점수	학점
1	디비	90	A
1	소공	80	B
2	디비	95	A
3	디비	75	C
3	소공	85	B

```

Select *
From (Select * From 수강) E
Where E.학번 = 1;
    
```

학번	과목	점수	학점
1	디비	90	A
1	소공	80	B

- Create View 명령어로 만들지 않고 SQL문 라인에 직접 기술하므로 **인라인 뷰**라 한다.
- 인라인 뷰는 FROM절의 테이블을 대체할 수 있어서 **파생 테이블**이라고 한다.
- 인라인 뷰는 SQL문 라인에 직접 기술하므로 필요한 시점에만 사용되는 특징이 있다.

기출문제 분석

1. 뷰(view)에 대한 설명으로 옳지 않은 것은? [2015년 국가 7급]

- ① 뷰는 기본테이블(base table)에 대한 질의로 정의되는 가상테이블(virtual table)로 질의 처리 성능을 향상시킬 수 있다.
- ② 뷰를 통해 기본테이블에 대한 사용자의 접근을 제한함으로써 보안성을 높일 수 있다.
- ③ With Check Option을 사용하여 뷰를 정의하면, 뷰를 통해 삽입 또는 갱신되는 튜플(tuple)에 대해 제한을 둘 수 있다.
- ④ 집계함수의 결과를 애트리뷰트(attribute)로 사용하는 뷰에 튜플의 삽입이나 갱신이 불가능하다.

☞ 뷰(view)

- 뷰는 기본테이블(base table)에 대한 질의로 정의되는 가상테이블(virtual table)로 질의 처리 성능을 향상시킬 수 있다.(×)
 - 사용자의 뷰 질의에 대해서 시스템은 해당 기본테이블에 대한 질의로 변경해야 한다.
 - 뷰를 통해 최종적으로 기본테이블에 접근해야 한다.
 - 만약, 뷰가 다중 테이블 질의로 정의된다면 조인을 수행해야 하므로 성능이 떨어진다.
 - 뷰 활용도를 높이려면, 수행속도 영향을 받지 않도록 조건을 부여할 수 있어야 한다.

// 뷰에 대한 변경 연산(삽입, 삭제, 갱신)이 불가능한 경우

- 뷰의 열이 상수나 산술식(연산자, 함수가 사용된 식)으로 구성되면 변경할 수 없다.
- 집계함수, Distinct, Group by, Having이 사용되어 정의된 뷰는 변경할 수 없다.
- 2개 이상의 테이블이 연관되어 정의된 뷰는 변경할 수 없다.(조인이 적용된 경우)
- 변경할 수 없는 뷰를 기초로 정의된 뷰를 이용하여 데이터를 변경할 수 없다.

// With Check Option

```
CREATE View 우등생(학번, 이름, 점수)
    As Select 학번, 이름, 점수 From 학생
    Where 점수 > 90 With Check Option;
```

↓ With Check Option 절

- With Check Option 절은 Where 문의 조건을 만족하는 변경만 수용하도록 한다.
- 뷰에 대한 삽입이나 갱신 연산이 수행될 때 조건(점수>90)을 위배하면 실행을 거절한다.

2. SQL 뷰에 대한 설명으로 옳은 것은? [2023년 국가 9급]

- ① 복잡한 질의를 간단하게 표현할 수 있게 한다.
- ② 데이터 무결성을 보장하지만 독립성을 제공하지는 않는다.
- ③ 제거할 때는 DELETE문을 사용한다.
- ④ 동일한 데이터에 대해 하나의 뷰만 생성 가능하다.

☞ SQL 뷰

- ② 데이터 무결성을 보장하지만 독립성을 제공하지는 않는다.(x)
→ 뷰는 논리적 데이터 독립성을 어느 정도 제공할 수 있다.
- ③ 제거할 때는 DELETE문을 사용한다.(x)
→ 뷰의 삭제 : Drop view 뷰_이름 {Restrict | Cascade};
- ④ 동일한 데이터에 대해 하나의 뷰만 생성 가능하다.(x)
→ 동일한 데이터에 대해 다수의 뷰 생성 가능하다.

// 뷰의 장점

- 뷰는 논리적 데이터 독립성을 어느 정도 제공할 수 있다.
→ 뷰가 정의된 기본테이블이 변해도 뷰가 사용된 응용프로그램은 전혀 영향받지 않는다.
- 뷰는 데이터에 대한 접근제어를 통해 보안을 제공할 수 있다.
→ 뷰를 통해서만 자료를 접근하게 되면, 뷰에 표현되지 않은 데이터는 안전하게 보호된다.
- 동일 데이터에 대해 동시에 여러 사용자의 상이한 응용이나 요구를 지원해 준다.
→ 하나의 기본테이블에 대해 다양한 형태의 뷰를 생성할 수 있다.
- 사용자의 데이터 관리를 간단하게 해준다.
→ 각 사용자가 필요로 하는 데이터만을 뷰로 정의하여 다루면 데이터 관리가 단순해진다.

// 뷰의 단점

- 뷰는 그 정의를 변경할 수 없다.(삭제 후 다시 생성해야 함)
- 뷰는 삽입, 삭제, 갱신 연산에 제약이 있다.
→ 하나의 테이블 위에서 뷰가 정의되었으나 기본키가 포함되지 않은 뷰는 변경할 수 없다.
- 뷰는 독자적인 인덱스를 가지지 못한다. 기본테이블에 생성된 인덱스를 이용한다.
- 뷰는 질의 처리 성능을 떨어뜨린다.
→ 사용자의 뷰 질의에 대해서 시스템은 해당 기본테이블에 대한 질의로 변경해야 한다.
→ 뷰를 통해 최종적으로 기본테이블에 접근해야 한다.
→ 만약, 뷰가 다중 테이블 질의로 정의된다면 조인을 수행해야 하므로 성능이 떨어진다.
→ 뷰 활용도를 높이려면, 수행속도 영향을 받지 않도록 조건을 부여할 수 있어야 한다.
→ **뷰 활용** : 보안성, 편의성, 융통성, 활용성, 임시성

3. SQL에서 뷰(view)의 역할에 대한 설명으로 옳지 않은 것은? [2019년 국가 7급]

- ① 기본테이블(base table)들만으로 작성된 질의를 간소화시킬 수 있다.
- ② 사용자의 접근권한에 따라 동일한 기본테이블의 속성들을 선택적으로 제공할 수 있다.
- ③ 기본테이블들만으로 작성된 질의 처리 성능을 향상시키기 위해 개발되었다.
- ④ 기본테이블들의 물리적 구조를 변경시키지 않고 사용자가 원하는 새로운 가상테이블을 생성시킬 수 있다.

♣ SQL에서 뷰(view)

- 기본테이블들만으로 작성된 질의 처리 성능을 향상시키기 위해 개발되었다.(×)
 - 뷰와 기본테이블은 거울과 같은 관계이므로 질의 처리 성능이 향상되는 것은 아니다.
 - 뷰를 통해 최종적으로 기본테이블에 접근하므로(직접 접근하지 않고, 2번에 걸쳐 접근)
 - 뷰 활용도를 높이려면, 수행속도 영향을 받지 않도록 조건을 부여할 수 있어야 한다.
 - 뷰 활용 : 보안성, 편의성, 융통성, 활용성, 임시성
-

정답 : ③

4. SQL 뷰(view)에 대한 설명으로 옳지 않은 것은? [2006년 서울 9급]

- ① 뷰 생성할 때는 Create view 명령을 사용한다.
- ② 뷰 삭제할 때는 Drop view 명령을 사용한다.
- ③ 뷰 관련 검색은 Select view 명령을 사용한다.
- ④ 뷰는 보안을 제공하므로 데이터를 안전하게 보호한다.
- ⑤ 뷰는 하나의 기본테이블에 대해 다양한 형태의 뷰를 생성할 수 있다.

♣ SQL 뷰(view) - 명령어 구조

- 뷰의 검색은 기본테이블처럼 똑같이 사용한다. Select문에서는 키워드 view가 없다.
- 모든 검색(Select) 문은 기본테이블을 대상으로 하므로 키워드 view가 필요 없다.

뷰 생성 : Create view 뷰이름

뷰 삭제 : Drop view 뷰이름

뷰 검색 : Select 검색 대상 → Select문에서는 키워드 view가 없다.

정답 : ③

5. 다음 사원 테이블에 대해 뷰_사원1과 같은 뷰를 생성하는 SQL 질의어를 작성할 때, ㉠, ㉡에 들어갈 내용을 바르게 연결한 것은? [2021년 국가 7급]

사원

사번	이름	직급	부서
108002	김진수	부장	인사
105123	이수진	대리	연구
128372	박지훈	과장	영업
126721	김지수	부장	영업
132122	홍성대	대리	인사

뷰_사원1

사번	이름
105123	이수진

Create View 뷰_사원1(사번, 이름)
 (㉠) Select 사번, 이름
 From 사원
 Where (㉡);

- | | |
|------|-----------|
| ㉠ | ㉡ |
| ① IN | 직급 = '대리' |
| ② AS | 부서 = '연구' |
| ③ AS | 직급 = '대리' |
| ④ IN | 부서 = '연구' |

♣ 뷰(view)

- 부서 = '연구'인 사원의 사번과 이름을 구하는 뷰를 생성하는 문제이다.

Create View 뷰_사원1(사번, 이름)
 (㉠ As) Select 사번, 이름
 From 사원
 Where (㉡ 부서 = '연구');

// 뷰 생성 구문은 다음과 같다.

```
Create View 뷰_이름(열_이름_리스트)           //대괄호 []는 생략 가능한 부분
As Select 열_이름_리스트 From 테이블_이름
[Where 조건]
[Group By 열_이름] [Having 조건]
[With Check Option];
```

6. 학생(학번, 이름, 학년)과 등록(학번, 과목, 학점) 릴레이션에 대해 다음과 같이 SQL 뷰를 정의하였다. 다음 중 변경이 허용되는 뷰로 가장 옳은 것은? [2022년 군무원 7급]

- ① create view 과목성적(과목, 학점) as select 과목, avg(학점) from 등록 group by 과목;
- ② create view 학생성적(이름, 과목, 학점) as select 이름, 과목, 학점 from 학생, 등록 where 학생.학번 = 등록.학번;
- ③ create view 백점만점(학번, 과목, 학점) as select 학번, 과목, 학점 * 100/4.5 from 등록;
- ④ create view 고학년(학번, 이름, 학년) as select * from 학생 where 학년 >= 3;

♣ 변경이 허용되는 뷰

// 먼저, 주어진 릴레이션의 기본키는 다음과 같다. 밑줄친 속성이 각 릴레이션의 기본키이다.

학생(학번, 이름, 학년)

등록(학번, 과목, 학점)

- 주어진 문제는 각 릴레이션의 기본키가 무엇인지? 정확하게 제시하지 않았다.
- 각 릴레이션의 기본키를 제시하지 않았지만, 기본키가 무엇인지? 유추해서 풀어야 한다.

- ① create view 과목성적(과목, 학점) as select 과목, avg(학점) from 등록 group by 과목;(×)
→ Group by가 사용되어 정의된 뷰는 변경할 수 없다.
- ② create view 학생성적(이름, 과목, 학점) as select 이름, 과목, 학점 from 학생, 등록 where 학생.학번 = 등록.학번;(×)
→ 기본키를 포함하지 않는 뷰이다. 변경이 허용되지 않는 뷰이다. 기본키는 (학번, 과목)이다.
- ③ create view 백점만점(학번, 과목, 학점) as select 학번, 과목, 학점 * 100/4.5 from 등록;(×)
→ 상수나 산술식(연산자, 함수가 사용된 식)으로 구성된 뷰는 변경할 수 없다.
- ④ create view 고학년(학번, 이름, 학년) as select * from 학생 where 학년 >= 3:(○)
→ 기본키인 학번을 포함하고 있다. 변경이 허용되는 뷰이다.

// 뷰에 대한 변경(삽입, 삭제, 갱신)이 불가능한 경우

- 뷰의 열이 상수나 산술식(연산자, 함수가 사용된 식)으로 구성되면 변경할 수 없다.
→ 삭제 연산은 허용되지만, 삽입과 갱신 연산은 이 열에 대해서 허용되지 않는다.
- 집계함수, Distinct, Group by, Having이 사용되어 정의된 뷰는 변경할 수 없다.
→ 뷰에 대한 변경을 어떤 방식으로 기본테이블에 적용해야 할지 결정할 수 없다
- 2개 이상의 테이블이 연관되어 정의된 뷰는 변경할 수 없다.(조인이 적용된 경우)
→ 뷰에 대한 변경이 기본테이블에 대한 변경으로 정확하게 대응되지 않을 수 있다.
- 변경할 수 없는 뷰를 기초로 정의된 뷰를 이용하여 데이터를 변경할 수 없다.

7. 데이터베이스에서 뷰(view)에 대한 설명으로 옳은 것은? [2017년 서울 9급]

- ① 뷰는 테이블을 기반으로 만들어지는 가상테이블이며, 뷰를 기반으로 새로운 뷰를 생성할 수 없다.
- ② 뷰 삭제는 SQL 명령어 중 DELETE 구문을 사용하며, 뷰 생성에 기반이 된 기존 테이블 들은 영향을 미치지 않는다.
- ③ 뷰 생성에 사용된 테이블의 기본키를 구성하는 속성이 포함되어 있지 않은 뷰도 데이터의 변경이 가능하다.
- ④ 뷰 생성 시 사용되는 SELECT문에서 GROUP BY 구문은 사용 가능하지만, ORDER BY 구문은 사용할 수 없다.

☞ 데이터베이스에서 뷰(view)

-
- ① 뷰는 테이블을 기반으로 만들어지는 가상테이블이며, 뷰를 기반으로 새로운 뷰를 생성할 수 없다.(x)
→ 뷰는 테이블이나 다른 뷰를 기반으로 생성할 수 있다.
 - ② 뷰 삭제는 SQL 명령어 중 DELETE 구문을 사용하며, 뷰 생성에 기반이 된 기존 테이블들은 영향을 미치지 않는다.(x) → 뷰 삭제는 Drop view 구문을 사용
 - ③ 뷰 생성에 사용된 테이블의 기본키를 구성하는 속성이 포함되어 있지 않은 뷰도 데이터의 변경이 가능하다.(x) → 기본키가 포함되어야 데이터의 변경이 가능하다.

// 뷰(view) - Order By 구문

```

Create view 뷰_이름[열_이름_리스트] //대괄호 []는 생략 가능한 부분
As Select 열_이름_리스트 From 테이블_이름
[Where 조건]
[Group By 열_이름][Having 조건]
[With Check Option];
    
```

- 뷰 생성에서 사용되는 SELECT문에서 ORDER BY를 사용할 수 없다.
- 뷰 특징에서 뷰는 독자적인 인덱스를 가지지 못한다.(ORDER BY는 인덱스)
- 뷰는 기본적으로 기본테이블에 생성된 인덱스를 이용한다.
- 해서, 뷰 생성에서 사용되는 SELECT문에 ORDER BY를 사용할 수 없다.
- 당연, 표준 SQL에서 뷰 생성에서 ORDER BY를 사용할 수 없다.

- 그런데, 뷰 생성에 사용되는 SELECT문에 ORDER BY를 허용하기도 한다.
- 요즘, 오라클에서는 뷰 생성에 사용되는 SELECT문에 ORDER BY를 허용한다.
- 어떻게 보면, 참 어려운 문제입니다.

8. SQL의 부속질의(subquery)에 대한 설명으로 옳지 않은 것은? [2023년 국가 7급]

- ① 부속질의는 하나의 SQL문 안에 다른 SQL문이 중첩된 질의를 말한다.
- ② 연산자 ANY는 부속질의가 단일행을 반환할 때만 적용할 수 있다.
- ③ 스칼라(scalar) 부속질의는 결과 값을 단일한 스칼라 값으로 반환한다.
- ④ 인라인 뷰(inline view)는 FROM 절에서 사용되는 부속질의이다.

♣ 부속질의

- 연산자 ANY는 부속질의가 단일행을 반환할 때만 적용할 수 있다.(x)
→ 연산자 ANY는 부속질의가 다중행을 반환할 때만 적용할 수 있다.
- 부속질의문 : 하나의 SQL문 안에 또 다른 SQL문이 중첩된(nested) 것!

단일행 서브쿼리	<ul style="list-style-type: none"> • 서브쿼리 수행 결과 반환값은 오직 하나의 행만 가능(2개 이상이면 오류 발생) • 단일행연산자 :=, >, >=, <=, < • 즉, 메인쿼리 Where절에 비교연산자 =, >, >=, <=, < 사용 가능
다중행 서브쿼리	<ul style="list-style-type: none"> • 서브쿼리 수행 결과 반환값이 하나 이상인 경우 • 다중행연산자 : In, Exists, All, Any(Some) • In : 서브쿼리 실행 결과 중에 메인쿼리 값이 하나라도 존재하면 참 • Exists : 메인쿼리 행에 대해 서브쿼리 실행 결과가 하나라도 존재하면 참 • All : 메인쿼리 비교조건이 서브쿼리 검색 결과의 모든 값을 만족해야만 참 • Any : 메인쿼리 비교조건이 서브쿼리 검색 결과에 대해 적어도 하나만 만족하면 참

—〈실습〉—

학생			
학번	이름	학년	학과
1	홍재연	4	전산
2	이순신	3	토목
3	김유신	4	건축
4	강감찬	2	전산
5	홍하은	3	전산

수강		
학번	과목	점수
1	디비	90
1	소공	70
2	디비	70
3	소공	90
3	디비	80

단일행 서브쿼리	Select * From 수강 Where 학번 = (Select 학번 From 학생 Where 학번 = 1);	#메인쿼리(main query) #단일행 서브쿼리(subquery)
다중행 서브쿼리	Select * From 학생 Where 학번 In (Select 학번 From 수강 Where 과목 = '디비');	#메인쿼리(main query) #다중행 서브쿼리(subquery)

9. 데이터베이스에서 사용하는 뷰(view)에 대한 설명으로 옳은 것을 <보기>에서 모두 고르면?
[2018년 국회 9급]

-----<보 기>-----

- ㄱ. 뷰의 정의는 변경할 수 없다.
- ㄴ. 뷰는 삽입, 갱신, 삭제 연산에 제약이 있다.
- ㄷ. 뷰 위에 다른 뷰를 정의할 수 없다.
- ㄹ. 뷰가 정의된 테이블이 삭제되더라도 뷰는 남는다.
- ㅁ. 뷰는 물리적으로 구현되는 테이블이다.

- ① ㄱ, ㄴ ② ㄱ, ㅁ ③ ㄴ, ㄷ
- ④ ㄷ, ㄹ ⑤ ㄹ, ㅁ

♣ 뷰(view)

- ㄷ. 뷰 위에 다른 뷰를 정의할 수 없다.(×)
→ 뷰 위에 다른 뷰를 정의할 수 있다. 뷰는 또 다른 뷰를 정의하는데 사용될 수 있다.
- ㄹ. 뷰가 정의된 테이블이 삭제되더라도 뷰는 남는다.(×)
→ 뷰가 정의된 테이블이 삭제되면, 해당 뷰는 삭제된다.
→ 즉, 기본테이블이 삭제되면 해당 기본테이블에서 생성된 모든 뷰도 같이 삭제된다.

정답 : ①

10. 관계형 데이터베이스의 뷰(view)에 대한 장점으로 옳지 않은 것은? [2018년 계리직]

- ① 뷰는 데이터의 논리적 독립성을 일정 부분 제공할 수 있다.
- ② 뷰를 통해 데이터의 접근을 제어함으로써 보안을 제공할 수 있다.
- ③ 뷰에 대한 연산의 제약이 없어서 효율적인 응용프로그램의 개발이 가능하다.
- ④ 뷰는 여러 사용자의 상이한 응용이나 요구를 지원할 수 있어서 데이터 관리를 단순하게 한다.

♣ 관계형 데이터베이스 뷰(view)

- 뷰에 대한 연산의 제약이 없어서 효율적인 응용프로그램의 개발이 가능하다.(×)
→ 기본키를 포함하지 않는 뷰는 삽입, 삭제, 갱신 연산의 제약이 있다.

정답 : ③

11. 뷰(view)에 대한 설명으로 옳지 않은 것은? [2017년 국회 9급]

- ① 뷰는 독자적인 인덱스를 가질 수 없다.
- ② 뷰의 정의를 변경하기 위해 ALTER 문을 사용할 수 있다.
- ③ 뷰의 정의만 시스템 카탈로그에 저장하였다가 필요시 실행시간에 테이블을 구축한다.
- ④ 데이터에 대한 보안을 제공한다.
- ⑤ 뷰는 또 다른 뷰의 정의에 사용될 수 있다.

☞ 뷰(view)

- 뷰의 정의를 변경하기 위해 ALTER 문을 사용할 수 있다.(×)
→ 한번 정의된 뷰는 기본테이블처럼 ALTER문으로 수정할 수 없다.(삭제 후 다시 생성)
-

정답 : ②

12. 뷰(view)에 대한 설명으로 가장 옳은 것은? [2018년 서울 7급]

- ① 사용자들의 데이터 접근을 제어하여 보안성을 높일 수 있다.
- ② 단일 기본테이블에 집단 연산을 사용하지 않고 선택션과 프로젝션만을 사용하여 정의된 뷰에 대해서만 갱신할 수 있게 하는 것을 삽입 가능한 뷰라고 한다.
- ③ 뷰 매커니즘은 관계 모델에서 논리적 데이터의 종속성을 가진다.
- ④ 물리적으로 저장되어 있는 기본테이블과 다르게 가상테이블이므로, 갱신 연산에 제한을 받지 않는다.

☞ 뷰(view)

- ② 단일 기본테이블에 집단 연산을 사용하지 않고 선택션과 프로젝션만을 사용하여 정의된 뷰에 대해서만 갱신할 수 있게 하는 것을 삽입 가능한 뷰라고 한다.(×)
→ 뷰 생성은 읽기전용(read-only)과 갱신 가능(updatable) 상태로 정의할 수 있다.
→ 기본키가 포함되어 있을 때 갱신 가능(updatable)한 뷰가 된다.
 - ③ 뷰 매커니즘은 관계 모델에서 논리적 데이터 종속성을 가진다.(×)
→ 논리적 데이터 독립성을 가진다.
 - ④ 물리적으로 저장되어 있는 기본테이블과 다르게 가상테이블이므로, 갱신 연산에 제한을 받지 않는다.(×)
→ 기본키가 포함되어 있지 않은 뷰는 갱신 연산에 제한이 있다.
-

정답 : ①

13. 뷰(view)에 대한 설명으로 옳지 않은 것은? [2021년 국가 7급]

- ① 생성된 뷰는 새로운 독립적인 테이블로 물리 저장소에 저장된다.
- ② 데이터베이스 질의 시 사용자 편의성 및 재사용성, 보안성을 높이기 위한 기술이다.
- ③ 뷰의 질의연산은 제한을 받지 않지만 갱신연산은 제한을 받는다.
- ④ 하나의 테이블로 여러 개의 상이한 뷰를 정의하여 사용자의 요구에 따라 활용할 수 있다.

♣ 뷰(view)

-
- 생성된 뷰는 새로운 독립적인 테이블로 물리 저장소에 저장된다.(×)
→ 뷰는 물리적으로 구현되지 않는다. 그 정의만 시스템에 저장된다.
 - 뷰는 하나 이상의 다른 테이블로부터 유도되어 만들어진 이름을 가진 '가상테이블'이다.
 - DBMS는 뷰의 정의만을 시스템 카탈로그에 저장해 두었다가 실행시간에 참조한 테이블과 연동되어 테이블을 구축하여 보여 준다.
 - 가상이라고 하는 이유는 실제로 튜플을 갖지 않는 테이블이기 때문이다.
-

정답 : ①