1회차

1. 다음은 어떤 데이터 모델링에 대한 설명인가?

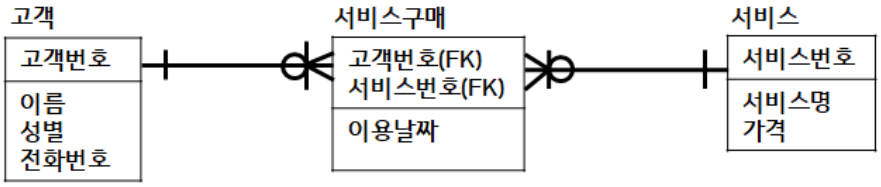
추상화 수준이 높고 업무 중심적이고 포괄적인 수준의 모델링 진행. 전사적 데이터 모델링

① 논리적 데이터 모델링
② 물리적 데이터 모델링
③ 개괄적 데이터 모델링
④ 개념적 데이터 모델링

2. 다음중 엔터티의 특징이 아닌 것은?
① 반드시 해당 업무에서 필요하고 관리되어야 하는 정보이어야 한다.
② 유일한 식별자에 의해 식별이 가능해야 한다.
③ 엔터티는 반드시 속성이 있어야 한다.
④ 정규화 이론에 근간하여 정해진 주식별자는 함수적 종속성을 가져야 한다.

3. 속성의 분류 중 속성의 특성에 따른 분류가 아닌 것은?
① 기본속성
② 파생속성
③ 일반속성
④ 설계속성

4. 아래 ERD에 대한 설명으로 가장 적절하지 않은 것은?



① 한 명의 고객은 여러 개의 서비스를 구매할 수 있다.
② 한 명의 고객은 서비스를 구매하지 않을 수 있다.
③ 하나의 서비스 구매는 반드시 한 명의 고객에 의해 주문된다.
④ 하나의 서비스 구매 이력은 고객 정보가 없을 수 있다.

5. 다음은 식별자의 특징 중 무엇을 설명하고 있는가?

사원번호 없는 회사직원은 있을 수 없음

① 유일성
② 최소성
③ 불변성
④ 존재성

6. 다음 엔터티를 제 3차 정규화를 수행했을 때 도출되는 엔터티의 수는(도서대출 포함)?
(단, 하나의 대출번호에 대해 하나의 도서만 대출할 수 있다고 가정)

도서대출

대출번호
대출자번호(FK)
대출자명
대출자직업
대출일자
대출도서번호
대출도서명
출판사명
출판년월
대표저자명
반납일자

① 1
② 2
③ 3
④ 4

7. 관계에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?

> 회원은 반드시 개인회원 또는 법인회원으로 회원가입을 한다.
> 회원 가입 후 개인회원 또는 법인회원으로 로그인하여 서비스를 이용할 수 있다.

① 관계는 존재적 관계와 행위에 의한 관계로 나누어 볼 수 있다.
② 부서와 사원 엔터티 간의 '소속' 관계는 존재적 관계이다.
③ 고객과 주문내역 엔터티 간의 '주문' 관계는 행위에 의한 관계이다.
④ 개인회원 또는 법인회원 둘 중 하나로 주문 가능할 경우 고객과 주문 엔터티는 상호포함적 관계이다.

8. 트랜잭션의 특징 중 보기는 무엇을 설명하고 있는가?

> 트랜잭션이 실행되기 전의 데이터베이스 내용이 잘못 되어 있지 않다면
> 트랜잭션이 실행된 이후에도 데이터베이스의 내용에 잘못이 있으면 안된다.

① 원자성
② 일관성
③ 고립성
④ 지속성

9. NULL에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?
① 정해지지 않은 값을 의미한다.
② NULL과의 모든 비교(IS NULL 제외)는 알 수 없음을 반환한다.
③ NULL로만 구성된 컬럼을 COUNT한 결과는 공집합이다.
④ 공백문자 혹은 숫자 0과는 다른 의미를 갖는다.

10. 다음 식별자에 대한 설명으로 가장 적절한 것은?

엔터티 내의 여러 인스턴스 중 하나를 유일하게 구분할 수 있으나, 대표성을 가지지 못하는 식별자

① 보조식별자
② 인조식별자
③ 본질식별자
④ 복합식별자

11. SELECT 문에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?
① 오라클에서는 GROUP BY절 위에 HAVING절을 명시할 수 있다.
② ORDER BY절은 문법 순서도 맨 마지막에 위치하며, 실행 순서 역시 마지막이다.
③ FROM 절은 모든 DBMS에서 생략 가능하다.
④ SELECT절에 DISTINCT는 항상 SELECT 바로 뒤에 위치한다.

12. SQL의 종류와 해당되는 명령어를 연결한 것 중 <u>틀린</u> 것은?
① DML - TRUNCATE
② TCL - COMMIT
③ DCL - GRANT
④ DDL - ALTER

13. SQL 문을 실행했을 때 오류가 발생하는 부분으로 가정 적절한 것은?
① SELECT DEPTNO, ROUND(AVG(SAL)) AS ROUND_VALUE
② FROM EMP E
③ WHERE ROUND_VALUE >= 2000
④ GROUP BY DEPTNO;

14. 다음의 함수 실행 결과 중 <u>틀린</u> 것은?
① SUBSTR('WWW.HDATALAB.CO.KR', -5) : 'CO.KR'
② LPAD('X',5,'X') : 'XXXXXX'
③ INSTR('WWW.HDATALAB.CO.KR','.', 5, 2) : 16
④ LTRIM('AABABAA', 'A') : 'BABAA'

15. 함수의 실행 결과로 적절하지 <u>않은</u> 것은?
① CEIL(3.5) : 4
② FLOOR(3.5) : 3
③ ROUND(12345.678, -2) : 12350
④ SIGN(120) : 1

16. 함수의 실행 결과 중 다른 하나는?

COMM
NULL
500

① NVL(COMM, 100)
② NULLIF(COMM, 100)
③ COALESCE(COMM, 100)
④ ISNULL(COMM, 100)

17. 다음 쿼리의 실행 결과로 알맞은 것은?

<TAB1>

SAL
3000
3500
4000
4200
6000

```
SELECT SUM(DECODE(SIGN(SAL-4000), 1, 1, 0))
  FROM TAB1;
```

① 0
② 1
③ 2
④ 3

18. 다음 SQL 중 실행 결과가 다른 것은?

<TAB1>

JUMIN(CHAR(13))
9012231223456
9606272349876

①
```
SELECT DECODE(SUBSTR(JUMIN, 7, 1), '1', '남자', '여자')
  FROM TAB1;
```
②
```
SELECT CASE WHEN SUBSTR(JUMIN, 7, 1) = 1 THEN '남자' ELSE '여자' END
  FROM TAB1;
```
③
```
SELECT CASE SUBSTR(JUMIN, 7, 1) WHEN 1 THEN '남자' ELSE '여자' END
  FROM TAB1;
```
④
```
SELECT CASE SUBSTR(JUMIN, 7, 1) WHEN '1' THEN '남자' ELSE '여자' END
  FROM TAB1;
```

19. 아래 SQL의 실행 결과로 가장 적절한 것은?(단, DBMS는 ORACLE로 가정)

```
SELECT TO_CHAR(TO_DATE('2024/08/24 10:00', 'YYYY/MM/DD HH24:MI')
        - 30/24/60, 'YYYY.MM.DD HH24:MI:SS')
  FROM DUAL;
```

① 2024.07.25 10:00:00
② 2024.08.24 06:00:00
③ 2024.08.24 09:30:00
④ 2024.08.24 09:59:30

20. 다음 SQL 중 실행 결과가 <u>다른</u> 것은?

①
```
SELECT JOB,
       COUNT(CASE WHEN DEPTNO = 10 THEN 1 ELSE 0 END) AS "10번부서원수",
       COUNT(CASE WHEN DEPTNO = 20 THEN 1 ELSE 0 END) AS "20번부서원수",
       COUNT(CASE WHEN DEPTNO = 30 THEN 1 ELSE 0 END) AS "30번부서원수"
  FROM EMP
 GROUP BY JOB;
```

②
```
SELECT JOB,
       COUNT(CASE WHEN DEPTNO = 10 THEN 1 END) AS "10번부서원수",
       COUNT(CASE WHEN DEPTNO = 20 THEN 1 END) AS "20번부서원수",
       COUNT(CASE WHEN DEPTNO = 30 THEN 1 END) AS "30번부서원수"
  FROM EMP
 GROUP BY JOB;
```

③
```
SELECT JOB,
       SUM(CASE WHEN DEPTNO = 10 THEN 1 ELSE 0 END) AS "10번부서원수",
       SUM(CASE WHEN DEPTNO = 20 THEN 1 ELSE 0 END) AS "20번부서원수",
       SUM(CASE WHEN DEPTNO = 30 THEN 1 ELSE 0 END) AS "30번부서원수"
  FROM EMP
 GROUP BY JOB;
```

④
```
SELECT JOB,
       NVL(SUM(CASE WHEN DEPTNO = 10 THEN 1 END),0) AS "10번부서원수",
       NVL(SUM(CASE WHEN DEPTNO = 20 THEN 1 END),0) AS "20번부서원수",
       NVL(SUM(CASE WHEN DEPTNO = 30 THEN 1 END),0) AS "30번부서원수"
  FROM EMP
 GROUP BY JOB;
```

21. 다음 SQL 수행 결과로 알맞은 것은?

<TAB1>

CODE
A1C1
A2C2
B1A0
B2C1
C1A1
D1B2

```
SELECT COUNT(CODE)
  FROM TAB1
 WHERE CODE LIKE '_1%'
       OR CODE LIKE '%A%';
```

① 4
② 5
③ 7
④ 8

22. 다음 SQL 문장 중 실행 결과가 다른 하나는?

① SELECT COUNT(*) FROM EMP WHERE DEPTNO = 10 OR DEPTNO = 20 AND JOB = 'CLERK';
② SELECT COUNT(*) FROM EMP WHERE (DEPTNO = 10 OR DEPTNO = 20) AND JOB = 'CLERK';
③ SELECT COUNT(*) FROM EMP WHERE DEPTNO = 10 OR (DEPTNO = 20 AND JOB = 'CLERK');
④ SELECT COUNT(*) FROM EMP WHERE (DEPTNO = 10 OR DEPTNO = 20 AND JOB = 'CLERK')

23. 아래 SQL 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2	COL3
100	NULL	100
NULL	200	400
0	300	NULL

```
SELECT SUM(COL2) + SUM(COL3) FROM TAB1;
SELECT SUM(COL2) + SUM(COL3) FROM TAB1 WHERE COL1 > 0;
SELECT SUM(COL2) + SUM(COL3) FROM TAB1 WHERE COL1 IS NOT NULL;
SELECT SUM(COL2) + SUM(COL3) FROM TAB1 WHERE COL1 IS NULL;
```

① 500, 100, 400, 600
② 500, NULL, 400, 600
③ 1000, 100, 400, 600
④ 1000, NULL, 400, 600

24. 다음 중 GROUP BY절 대한 설명 중 <u>틀린</u> 것은?

① GROUP BY절에는 컬럼 별칭을 사용할 수 없다.

② GROUP BY절에는 SUM, COUNT 함수를 사용할 수 없다.

③ GROUP BY절에 명시되지 않은 컬럼을 그룹함수 없이 SELECT절에 사용할 수 없다.

④ GROUP BY절에 나열되는 컬럼 순서에 따라 SELECT절의 그룹함수의 연산 결과가 달라질 수 있다.

25. 아래 SQL 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2	COL3
100	NULL	100
NULL	200	400
0	300	NULL

```
SELECT COUNT(*) AS RESULT FROM TAB1 WHERE COL1 = 200 GROUP BY COL1;
SELECT SUM(COL2) AS RESULT FROM TAB1 WHERE COL1 = 100;
SELECT COUNT(COL1) AS RESULT FROM TAB1 WHERE COL3 = 400;
```

①

RESULT
0

RESULT
0

RESULT
0

②

RESULT

RESULT

RESULT
0

③

RESULT

RESULT

RESULT

④

RESULT

RESULT

RESULT
0

26. 다음 중 실행이 불가한 구문은?

①
```
SELECT COL1, COL2, COL3 C1
  FROM TAB1 T
 ORDER BY 1, COL2, COL3;
```

②
```
SELECT COL1, COL2, COL3 C1
  FROM TAB1 T
 ORDER BY 1, 2, 3;
```

③
```
SELECT COL1, COL2, COL3 C1
  FROM TAB1 T
 ORDER BY 1, COL2, C1;
```

④
```
SELECT COL1, COL2, COL3 C1
  FROM TAB1 T
 ORDER BY COL1, COL1, T.C1;
```

27. 다음 SQL 구문의 실행 결과로 가장 적절한 것은?

<TAB1>

SAL
1300
800
2500
300

```
SELECT SAL
  FROM TAB1
 ORDER BY TO_CHAR(SAL);
```

① 1300 2500 300 800
② 300 800 1300 2500
③ 300 1300 800 2500
④ 2500 1300 800 300

28. 아래의 SQL 의 결과로 알맞은 것은?

<TAB1>

COL1	COL2
1	A
2	
3	B
4	C

<TAB2>

COL1	COL2
1	A
2	
3	B

```
SELECT SUM(A.COL1)
  FROM TAB1 A, TAB2 B
 WHERE A.COL2 <> B.COL2;
```

① 8
② 10
③ 12
④ 30

29. 아래 SQL 실행 결과로 알맞은 것은?

<TAB1>

EMPNO	NAME
1000	SCOTT
2000	SMITH
3000	FORD
4000	TIGER

<TAB2>

NO	RULE_NAME
1	%O%
2	F%

```
SELECT COUNT(*)
  FROM TAB1, TAB2
 WHERE NAME LIKE RULE_NAME;
```

① 0
② 1
③ 2
④ 3

30. 아래와 같은 테이블 TAB1, TAB2가 있을 때 아래의 SQL의 결과 건수를 알맞게 나열한 것은?

<TAB1>

COL1	COL2	KEY1
A	100	BB
B	200	CC
C	300	DD
NULL	400	EE

<TAB2>

COL1	COL2	KEY2
A	100	AA
B	200	BB
C	300	BB
NULL	400	EE

```
SELECT * FROM TAB1 A INNER JOIN TAB2 B ON (A.KEY1 = B.KEY2)
SELECT * FROM TAB1 A LEFT OUTER JOIN TAB2 B ON (A.KEY1 = B.KEY2)
SELECT * FROM TAB1 A FULL OUTER JOIN TAB2 B ON (A.KEY1 = B.KEY2)
SELECT * FROM TAB1 A CROSS JOIN TAB2 B
SELECT * FROM TAB1 A NATURAL JOIN TAB2 B
```

① 2 4 6 8 3
② 2 5 6 16 4
③ 3 5 6 16 3
④ 3 6 8 12 4

31. 아래와 같은 테이블 데이터가 있다. 각 SQL에 대한 결과값이 잘못된 것은?

<TAB1>

N1	V1
1	A
2	
3	B
4	C

<TAB2>

N1	V1
1	A
2	
3	B

① SELECT *
FROM TAB1
WHERE V1 IN (SELECT V1 FROM TAB2);

N1	V1
1	A
3	B

② SELECT *
FROM TAB1
WHERE V1 NOT IN (SELECT V1 FROM TAB2);

N1	V1
4	C

③ SELECT *
FROM TAB1 A
WHERE EXISTS (SELECT 'X'
FROM TAB2 B
WHERE A.V1 = B.V1);

N1	V1
1	A
3	B

④ SELECT *
FROM TAB1 A
WHERE NOT EXISTS (SELECT 'X'
FROM TAB2 B
WHERE A.V1 = B.V1);

N1	V1
2	
4	C

32. 아래와 같은 테이블이 있다. 다음 SQL 실행 결과로 올바른 것은?

<EMPLOYEES>

EMPNO	DEPTNO	HIREDATE
1	NULL	1998/12/23
2	10	2001/05/06
3	20	2020/01/16
4	30	2020/02/01
5	40	2020/02/28
6	50	2021/06/07

```
SELECT COUNT(DEPTNO)
  FROM EMPLOYEES
 WHERE DEPTNO <= ALL(SELECT DEPTNO
                       FROM EMPLOYEES
                      WHERE HIREDATE >= TO_DATE('2020/02', 'YYYY/MM'));
```

① 2
② 3
③ 4
④ 5

33. 다음 SQL 중 정상 수행이 불가한 것은?

①
```
SELECT *
  FROM TAB1 A
 WHERE NOT EXISTS (SELECT 'X'
                     FROM TAB2 B
                    WHERE A.NO = B.NO);
```

②
```
SELECT A.NO, (SELECT B.NAME
                FROM TAB2 B)
  FROM TAB1 A
 WHERE A.NO = B.NO;
```

③
```
SELECT NO
  FROM TAB1 A JOIN TAB2 B
 USING (NO);
```

④
```
UPDATE TAB1 A
   SET A.NAME = (SELECT B.NAME
                   FROM TAB2 B
                  WHERE A.NO = B.NO);
```

34. 아래의 SQL 의 출력 결과 중 알맞은 것은?

<TAB1>

COL1	COL2
A	10
B	20
C	30

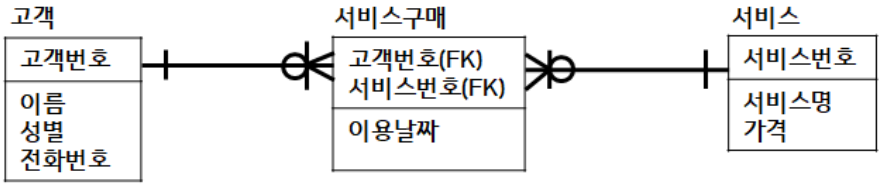
<TAB2>

COL1	COL2
B	20
C	30
D	40

```
SELECT COUNT(*)
  FROM TAB1
 WHERE EXISTS (SELECT 1
                 FROM TAB2
                WHERE TAB2.COL1 = 'A');
```

① NULL
② 0
③ 1
④ 3

35.다음 ERD를 보고, 고객의 성별로 서비스 이용횟수와 이용금액의 총합을 출력하는 SQL로 적절하지 <u>않은</u> 것은?



① SELECT 고객.성별, COUNT(서비스.서비스번호) AS CNT, SUM(서비스.가격) AS SUM_PRICE
FROM 고객, 서비스구매, 서비스
WHERE 고객.고객번호= 서비스구매.고객번호
AND 서비스구매.서비스번호 = 서비스.서비스번호
GROUP BY 고객.성별;

② SELECT 고객.성별, COUNT(고객.고객번호) AS CNT, SUM(서비스.가격) AS SUM_PRICE
FROM 고객 LEFT OUTER JOIN 서비스구매
ON 고객.고객번호= 서비스구매.고객번호 LEFT OUTER JOIN 서비스
ON 서비스구매.서비스번호 = 서비스.서비스번호
GROUP BY 고객.성별;

③ SELECT 고객.성별, COUNT(I.SNO) AS CNT, SUM(I.가격) AS SUM_PRICE
FROM 고객 INNER JOIN (SELECT 서비스구매.고객번호 AS GNO, 서비스.서비스번호 AS SNO, 서비스.가격
FROM 서비스구매 INNER JOIN 서비스
ON 서비스구매.서비스번호 = 서비스.서비스번호) I

```
    ON 고객.고객번호= I.GNO
  GROUP BY 고객.성별;
```

④
```
SELECT 고객.성별,
       COUNT((SELECT 서비스번호
                FROM 서비스
               WHERE 서비스구매.서비스번호 = 서비스.서비스번호)) AS CNT,
       SUM((SELECT 가격
              FROM 서비스
             WHERE 서비스구매.서비스번호 = 서비스.서비스번호)) AS SUM_PRICE
  FROM 고객 INNER JOIN 서비스구매
    ON 고객.고객번호= 서비스구매.고객번호
  GROUP BY 고객.성별;
```

36. 아래 SQL 중 수행이 정상적이지 <u>않은</u> 것은?

<EMP>

COLUMN	DATA_TYPE
EMPNO	NUMBER(4)
ENAME	VARCHAR2(10)
SAL	NUMBER(7,2)
DEPTNO	NUMBER(2)

<EMPLOYEES>

COLUMN	DATA_TYPE
EMPLOYEE_ID	VARCHAR2(7)
NAME	VARCHAR2(10)
SALARY	NUMBER(8,2)
DEPARTMENT_ID	VARCHAR2(5)

①
```
SELECT DEPTNO, SUM(SAL) AS SUM_SAL
  FROM EMP
 GROUP BY DEPTNO
 UNION ALL
SELECT DEPARTMENT_ID, SUM(SALARY)
  FROM EMPLOYEES
 GROUP BY DEPARTMENT_ID;
```

②
```
SELECT ENAME, SAL
  FROM EMP
 MINUS
SELECT NAME, SALARY
  FROM EMPLOYEES;
```

③ SELECT TO_CHAR(EMPNO) AS EMPNO, ENAME, SAL
FROM EMP
UNION
SELECT EMPLOYEE_ID, NAME, SALARY
FROM EMPLOYEES
ORDER BY EMPNO;

④ (SELECT TO_CHAR(EMPNO) AS EMPNO, ENAME, SAL
FROM EMP
UNION
SELECT EMPLOYEE_ID AS EMPNO, NAME, SALARY
FROM EMPLOYEES)
ORDER BY EMPNO;

37. 아래 쿼리중 결과값이 다른 하나는?

① SELECT DNAME, JOB, COUNT(*) AS CNT, SUM(SAL) AS TOTAL_SAL
FROM SCOTT.EMP A, SCOTT.DEPT B
WHERE A.DEPTNO = B.DEPTNO
GROUP BY ROLLUP(DNAME,JOB)
ORDER BY DNAME, JOB;

② SELECT DNAME, JOB, COUNT(*) AS CNT, SUM(SAL) AS TOTAL_SAL
FROM SCOTT.EMP A, SCOTT.DEPT B
WHERE A.DEPTNO = B.DEPTNO
GROUP BY GROUPING SETS((DNAME,JOB), DNAME, NULL)
ORDER BY 1, 2;

③ SELECT DNAME, JOB, COUNT(*) AS CNT, SUM(SAL) AS TOTAL_SAL
FROM SCOTT.EMP A, SCOTT.DEPT B
WHERE A.DEPTNO = B.DEPTNO
GROUP BY CUBE(DNAME, JOB)
ORDER BY 1, 2;

④ SELECT DNAME, JOB, COUNT(*) AS CNT, SUM(SAL) AS TOTAL_SAL
FROM SCOTT.EMP A, SCOTT.DEPT B
WHERE A.DEPTNO = B.DEPTNO
GROUP BY DNAME, JOB
UNION ALL
SELECT DNAME, '' AS JOB, COUNT(*) AS CNT, SUM(SAL) AS TOTAL_SAL
FROM SCOTT.EMP A, SCOTT.DEPT B
WHERE A.DEPTNO = B.DEPTNO
GROUP BY DNAME
UNION ALL
SELECT '' AS DNAME, '' AS JOB, COUNT(*) AS CNT, SUM(SAL) AS TOTAL_SAL
FROM SCOTT.EMP A, SCOTT.DEPT B
WHERE A.DEPTNO = B.DEPTNO
ORDER BY 1, 2;

38. 다음 SQL 실행 결과로 가장 알맞은 것은?

<주문내역>

주문번호	일자	주문금액	지점
1	2024.01.01	1000	A
2	2024.01.02	1000	A
3	2024.01.02	2000	A
4	2024.01.03	3000	A
5	2024.01.04	3000	B
6	2024.01.05	4000	B

```
SELECT 주문번호, 일자,
       SUM(주문금액) OVER(ORDER BY 일자) AS 주문금액,
       DENSE_RANK() OVER(PARTITION BY 지점 ORDER BY 주문금액) AS 순위
  FROM 주문내역;
```

①

주문번호	일자	주문금액	순위
1	2024.01.01	1000	1
2	2024.01.02	2000	1
3	2024.01.02	4000	2
4	2024.01.03	7000	3
5	2024.01.04	10000	1
6	2024.01.05	14000	2

②

주문번호	일자	주문금액	순위
1	2024.01.01	1000	1
2	2024.01.02	2000	1
3	2024.01.02	4000	3
4	2024.01.03	7000	4
5	2024.01.04	10000	1
6	2024.01.05	14000	2

③

주문번호	일자	주문금액	순위
1	2024.01.01	1000	1
2	2024.01.02	4000	1
3	2024.01.02	4000	2
4	2024.01.03	7000	3
5	2024.01.04	10000	1
6	2024.01.05	14000	2

④

주문번호	일자	주문금액	순위
1	2024.01.01	1000	1

2	2024.01.02	4000	1
3	2024.01.02	4000	3
4	2024.01.03	7000	4
5	2024.01.04	10000	1
6	2024.01.05	14000	2

39. 다음 SQL 실행 결과로 가장 알맞은 것은?

<TAB1>

NAME	SAL	DNAME
홍길동	300	아시아지부
박길동	400	아시아지부
최길동	500	아시아지부
김길동	450	남유럽지부
이길동	550	남유럽지부

```
SELECT DNAME,
        FIRST_VALUE(NAME) OVER(PARTITION BY DNAME ORDER BY SAL) AS VALUE1,
        LAST_VALUE(NAME) OVER(PARTITION BY DNAME ORDER BY SAL) AS VALUE2
  FROM TAB1;
```

①

DNAME	VALUE1	VALUE2
아시아지부	홍길동	최길동
아시아지부	박길동	최길동
아시아지부	최길동	최길동
남유럽지부	김길동	이길동
남유럽지부	이길동	이길동

②

DNAME	VALUE1	VALUE2
아시아지부	홍길동	홍길동
아시아지부	박길동	박길동
아시아지부	최길동	최길동
남유럽지부	김길동	김길동
남유럽지부	이길동	이길동

③

DNAME	VALUE1	VALUE2
아시아지부	홍길동	최길동
아시아지부	홍길동	최길동
아시아지부	홍길동	최길동
남유럽지부	김길동	이길동
남유럽지부	김길동	이길동

④

DNAME	VALUE1	VALUE2
아시아지부	홍길동	홍길동
아시아지부	홍길동	박길동
아시아지부	홍길동	최길동
남유럽지부	김길동	김길동
남유럽지부	김길동	이길동

40. 다음 SQL 실행 결과 중 다른 하나는?

<EXAM>

이름	성적
홍길동	98
박길동	60
최길동	72
김길동	80
이길동	80

① SELECT 성적
FROM EXAM
ORDER BY 성적 DESC
FETCH FIRST 3 ROWS ONLY;

② SELECT TOP(2) WITH TIES 성적
FROM EXAM
ORDER BY 성적 DESC;

③ SELECT 성적
FROM (SELECT 성적, RANK() OVER(ORDER BY 성적 DESC) AS RN
FROM EXAM)
WHERE RN <= 2;

④ SELECT 성적
FROM (SELECT 성적, ROWNUM AS RN
FROM EXAM
ORDER BY 성적 DESC)
WHERE RN <= 3;

41. 아래의 계층형 SQL 에서 리프 데이터이면 1, 그렇지 않으면 0 을 출력하고 싶을 때 사용하는 키워드로 알맞은 것은?

```
SELECT LEVEL,
       EMPNO,
       MGR,
       _____________ AS 리프노드여부
  FROM SCOTT.EMP
 START WITH MGR IS NULL
CONNECT BY PRIOR EMPNO = MGR;
```

① CONNECT_BY_ISLEAF
② CONNECT_BY_ISCYCLE
③ SYS_CONNECT_BY_PATH
④ CONNECT_BY_ROOT

42. 아래 SQL 실행 결과로 가장 알맞은 것은?

<사원>

사원번호	이름	입사일자	매니저사원번호
10000	나사장	2010-01-01	
10001	박전무	2010-01-01	10000
10002	김전무	2010-12-01	10000
10003	홍상무	2011-12-01	10002
10004	김상무	2012-12-01	10002
10005	김사원	2013-01-01	10003
10006	홍사원	2014-12-01	10003
10007	박사원	2015-01-01	10004
10008	최사원	2016-12-01	10004
10009	박인턴	2017-01-01	10007
10010	강인턴	2018-12-01	10008

```
SELECT E.*, LEVEL
  FROM 사원 E
 START WITH 매니저사원번호 IS NULL
CONNECT BY PRIOR 사원번호 = 매니저사원번호 AND EXTRACT(MONTH FROM 입사일자) >= 7
 ORDER SIBLINGS BY 이름;
```

①

사원번호	이름	입사일자	매니저사원번호	LEVEL
10000	나사장	2010-01-01 0:00		1
10002	김전무	2010-12-01 0:00	10000	2
10003	홍상무	2011-12-01 0:00	10002	3
10006	홍사원	2014-12-01 0:00	10003	4

10004	김상무	2012-12-01 0:00	10002	3
10008	최사원	2016-12-01 0:00	10004	4
10010	강인턴	2018-12-01 0:00	10008	5

②

사원번호	이름	입사일자	매니저사원번호	LEVEL
10000	나사장	2010-01-01 0:00		1
10002	김전무	2010-12-01 0:00	10000	2
10004	김상무	2012-12-01 0:00	10002	3
10008	최사원	2016-12-01 0:00	10004	4
10010	강인턴	2018-12-01 0:00	10008	5
10003	홍상무	2011-12-01 0:00	10002	3
10006	홍사원	2014-12-01 0:00	10003	4

③

사원번호	이름	입사일자	매니저사원번호	LEVEL
10002	김전무	2010-12-01 0:00	10000	2
10004	김상무	2012-12-01 0:00	10002	3
10008	최사원	2016-12-01 0:00	10004	4
10010	강인턴	2018-12-01 0:00	10008	5
10003	홍상무	2011-12-01 0:00	10002	3
10006	홍사원	2014-12-01 0:00	10003	4

④

사원번호	이름	입사일자	매니저사원번호	LEVEL
10002	김전무	2010-12-01 0:00	10000	2
10004	김상무	2012-12-01 0:00	10002	3
10008	최사원	2016-12-01 0:00	10004	4
10010	강인턴	2018-12-01 0:00	10008	5
10003	홍상무	2011-12-01 0:00	10002	3
10006	홍사원	2014-12-01 0:00	10003	4
10006	홍사원	2014-12-01 0:00	10003	4

43. 다음 SQL 구문 결과와 같은 결과를 갖는 SQL은?

```
SELECT COUNT(DECODE(DEPTNO,10,1)) AS "10",
       COUNT(DECODE(DEPTNO,20,1)) AS "20",
       COUNT(DECODE(DEPTNO,30,1)) AS "30"
  FROM EMP;
```

① SELECT *
FROM (SELECT EMPNO, DEPTNO FROM EMP)
PIVOT (COUNT(EMPNO) FOR DEPTNO IN (10,20,30));

② SELECT *
FROM (SELECT EMPNO, JOB, DEPTNO FROM EMP)
PIVOT (COUNT(DEPTNO) FOR EMPNO IN (10,20,30));

③ SELECT *
FROM EMP
UNPIVOT (COUNT(DEPTNO) FOR DEPTNO IN (10,20,30));

④ SELECT *
FROM EMP
UNPIVOT (COUNT(*) FOR DEPTNO IN (10,20,30));

44. 다음 출력 결과로 가장 알맞은 것은?

```
SELECT REGEXP_COUNT('abc1004 zz1234', '\d{2}+') AS C1,
       REGEXP_COUNT('abc1004-zz1234-100', '\d{2,}+') AS C2
  FROM DUAL;
```

① 2, 1
② 2, 3
③ 8, 1
④ 8, 3

45. 다음 출력 결과로 가장 알맞은 것은?

<TAB1>

COL1
AAXYXY-Z
XXYYZ
XY-
XY-Z
XY-+?

```
SELECT COUNT(COL1)
  FROM TAB1
 WHERE REGEXP_LIKE(COL1, '[XY-]+Z?');
```

① 2
② 3
③ 4
④ 5

46. 다음 SQL 실행 결과로 알맞은 것은?

<EMPLOYEE>

ID	NAME	SAL	DNAME
0001	홍길동	1000	아시아지부
0002	박길동	2000	아시아지부
0003	최길동	3000	아시아지부
0004	이길동	4000	남유럽지부
0005	김길동	6000	남유럽지부
0006	김길동	8000	남유럽지부

```
UPDATE EMPLOYEE E1
    SET SAL = (SELECT MAX(SAL)
                 FROM EMPLOYEE E2
                WHERE E1.DNAME = E2.DNAME)
 WHERE SAL <= (SELECT AVG(SAL)
                 FROM EMPLOYEE);
```

①

ID	NAME	SAL	DNAME
0001	홍길동	NULL	아시아지부
0002	박길동	NULL	아시아지부
0003	최길동	NULL	아시아지부
0004	이길동	NULL	남유럽지부
0005	김길동	6000	남유럽지부
0006	김길동	8000	남유럽지부

②

ID	NAME	SAL	DNAME
0001	홍길동	3000	아시아지부
0002	박길동	3000	아시아지부
0003	최길동	3000	아시아지부
0004	이길동	4000	남유럽지부
0005	김길동	6000	남유럽지부
0006	김길동	8000	남유럽지부

③

ID	NAME	SAL	DNAME
0001	홍길동	3000	아시아지부
0002	박길동	3000	아시아지부
0003	최길동	3000	아시아지부
0004	이길동	8000	남유럽지부
0005	김길동	6000	남유럽지부
0006	김길동	8000	남유럽지부

④

ID	NAME	SAL	DNAME
0001	홍길동	4000	아시아지부
0002	박길동	4000	아시아지부
0003	최길동	4000	아시아지부
0004	이길동	8000	남유럽지부
0005	김길동	6000	남유럽지부
0006	김길동	8000	남유럽지부

47. 아래 SQL 실행 결과로 가장 적절한 것은?

<TAB1>

NO	COL1	COL2
1	A	1
2	B	2
3	C	2
4	D	3

```
INSERT INTO TAB1 VALUES(5, 'E', 3);
COMMIT;
UPDATE TAB1 SET COL2 = 3 WHERE NO = 2;
SAVEPOINT SAVE1;
INSERT INTO TAB1 VALUES(6, 'F', 5);
DELETE TAB1 WHERE NO = 4;
ROLLBACK TO SAVE1;
UPDATE TAB1 SET COL2 = 2 WHERE NO = 1;
ROLLBACK;
COMMIT;

SELECT SUM(COL2) FROM TAB1;
```

① 8
② 11
③ 12
④ 13

48. 비교연산자의 어느 한쪽이 VARCHAR 유형 타입인 경우 문자 유형 비교에 대한 설명 중 가장 알맞지 <u>않은</u> 것은?
① 서로 다른 문자가 나올 때까지 비교한다
② 길이가 다르다면 짧은 것이 끝날 때까지만 비교한 후에 길이가 긴 것이 크다고 판단한다
③ 길이가 같고 다른 것이 없다면 같다고 판단한다
④ 길이가 다르다면 작은 쪽에 SPACE 를 추가하여 길이를 같게 한 후에 비교한다

49. 다음 중 실행이 불가능한 SQL을 고르시오.

< TAB1 >

COL1	NUMBER(4)
COL2	NUMBER(5,2)
COL3	VARCHAR2(10)
COL4	CHAR(10)

① INSERT INTO TAB1 VALUES(1000, 12.345, 100, 'ABC');
② INSERT INTO TAB1 VALUES(1000, 123.45, '100', 'ABC');
③ INSERT INTO TAB1 VALUES(1000, 123.456, '2024-01-01', NULL);
④ INSERT INTO TAB1 VALUES(1000, 1234.5, '100', ' ');

50. 유저와 권한 중 권한에 대한 설명 중 가장 올바르지 <u>않은</u> 것은?
① 사용자가 실행하는 모든 DDL 문장은 그에 해당하는 적절한 권한이 있어야만 문장을 실행 할 수 있다.
② DBA 권한을 가진 유저만이 권한을 부여 할 수 있다
③ 테이블의 소유자는 해당 테이블의 DML 권한을 다른 유저에게 부여 할 수 있다.
④ 권한 부여를 편리하게 관리하기 위해 만들어진 권한의 집합인 ROLE 이 있다

2회차

1. 데이터 모델링을 할 때 유의해야 할 사항으로 가장 적절하지 않은 것은?
   ① 같은 정보를 저장하지 않도록 하여 중복성을 최소화한다.
   ② 데이터 간의 상호 연관관계를 명확하게 정의하여 일관성 있게 데이터가 유지되도록 한다.
   ③ 데이터의 정의를 프로세스와 분리하여 유연성을 높인다.
   ④ 사용자가 처리하는 프로세스에 따라 매핑이 될 수 있도록 프로그램과 테이블 간의 연계성을 높인다.

2. 엔터티 분류 중 발생시점에 따른 분류가 아닌 것은?
   ① 기본엔터티
   ② 중심엔터티
   ③ 사건엔터티
   ④ 행위엔터티

3. 속성에 대한 설명으로 가장 적절하지 않은 것은?
   ① 업무상 인스턴스로 관리하고자 하는 더 이상 분리되지 않는 최소 데이터 단위를 나타낸다.
   ② 하나의 엔터티는 두 개 이상의 속성을 갖는다.
   ③ 하나의 인스턴스에서 각각의 속성은 하나 이상의 속성값을 가질 수 있다.
   ④ 속성은 정해진 주식별자에 함수적 종속성을 가져야 한다.

4. 데이터 모델링의 관계에 대한 설명으로 가장 적절하지 않은 것은?
   ① 관계는 존재에 의한 관계와 행위에 의한 관계로 구분될 수 있으나 ERD에서는 존재와 행위를 구분하지 않는다.
   ② 부서와 사원 엔터티 간의 '소속' 관계는 존재적 관계의 사례이다.
   ③ 관계의 페어링은 하나의 엔터티와 다른 엔터티 간의 레코드 연결 방식을 나타낸다.
   ④ 관계의 표기법은 관계명, 관계차수, 선택성 3가지 개념을 사용한다.

5. 데이터 모델링에서 식별자 관계에 대한 설명 중 가장 적절하지 않은 것은?
   ① 식별 관계는 하나의 엔터티의 기본키를 다른 엔터티가 기본키의 하나로 공유하는 관계이다.
   ② IE 표기법에서는 식별 관계는 실선으로, 비식별 관계는 점선으로 표시한다.
   ③ 비식별 관계는 부모 엔터티가 소멸하면 자식 엔터티도 종속적으로 삭제되는 관계이다.
   ④ 자식 엔터티의 식별자가 부모 엔터티의 주식별자를 상속받아 생성하는 것보다 별도의 주식별자를 생성하는 것이 더 유리하다고 판단되는 경우 비식별자 관계로 연결해야 한다.

6. 데이터 모델링의 정규화에 대한 설명으로 가장 적절하지 않은 것은?
   ① 정규화는 모델의 일관성을 확보하고 중복을 제거하여 모델의 독립성을 확보하는 과정이다.
   ② 개념 모델링 단계에서의 엔터티를 상세화 하는 과정이다.
   ③ 제1정규형은 모든 인스턴스가 반드시 하나의 값을 가져야 함을 의미한다.
   ④ 제3정규형을 만족하는 엔터티의 일반속성은 주식별자 전체에 종속적이다.

7. 관계(Relationship)와 조인(Join)에 대한 설명으로 가장 적절하지 않은 것은?
① 조인(Join)이란 식별자를 상속하고, 상속된 속성을 매핑키로 활용하여 데이터를 결합하는 것을 의미한다.
② 부모의 식별자를 자식의 일반속성으로 상속하면 식별 관계, 부모의 식별자를 자식의 식별자에 포함하면 비식별 관계라고 할 수 있다.
③ 엔터티 간의 관계를 통해 데이터의 중복을 피하고, 각 데이터 요소를 한 번만 저장하여 유지관리의 복잡성을 줄일 수 있다.
④ 관계는 엔터티간의 논리적 연관성을 의미한다.

8. 트랜잭션에 대한 설명 중 가장 적절하지 않은 것은?
① 트랜잭션에 의한 관계는 선택적인 관계 형태를 가진다.
② 원자성이란 하나의 트랜잭션의 작업이 모두 성공하거나 모두 취소되어야 하는 특징을 말한다.
③ 순차적으로 수행되는 작업 A와 B가 하나의 트랜잭션일 경우 A만 실행되고 시스템 장애가 발생했다면 A를 ROLLBACK해야 한다.
④ 하나의 트랜잭션으로 구성된 작업은 부분 COMMIT이 불가하다.

9. NULL에 대한 설명으로 가장 적절하지 않은 것은?
① 정해지지 않은 값을 의미한다.
② 논리모델 설계 시 각 컬럼별로 NULL을 허용할 지를 결정한다.
③ IE 표기법에서는 각 컬럼별 NULL 허용 여부를 알 수 없다.
④ 바커 표기법에서는 속성 앞에 별(*)을 사용하여 널 허용 속성을 표현한다.

10. 본질식별자와 인조식별자에 대한 설명으로 가장 적절하지 않은 것은?
① 인조식별자를 사용하면 불필요하게 발생하는 중복데이터를 막을 수 있다.
② 인조식별자를 사용하면 본질식별자를 사용할 때와 비교하여 추가적인 인덱스가 필요해진다.
③ 인조식별자는 대체로 본질식별자가 복잡한 구성을 가질 때 만들어진다.
④ 자동으로 증가하는 일련번호 같은 형태는 인조식별자에 해당한다.

11. 다음 SQL 중 항상 오류가 발생하는 구문으로 가장 적절한 것은?
①
```
SELECT T.COL1 C1, T.COL2 AS C2
  FROM TABLE1 T
 WHERE T.COL1 = 4;
```
②
```
SELECT TABLE1.COL1, SUM(TABLE1.COL2)
  FROM TABLE1
 GROUP BY TABLE1.COL1
 ORDER BY COL1;
```
③
```
SELECT T.COL1 C1, TABLE1.COL2 AS C2
  FROM TABLE1 T
 WHERE TABLE1.COL2 = 'A'
 ORDER BY 1, 2;
```
④
```
SELECT T.COL1 C1, T.COL2 AS "C1"
  FROM TABLE1 T
 WHERE T.COL2 IN ('A', 'B')
 ORDER BY 1, "C1";
```

12. SELECT 문에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?

① GROUP BY절에는 컬럼별칭을 사용할 수 없다.
② WHERE절에는 그룹함수를 사용한 조건 전달이 불가하다.
③ HAVING절에서는 그룹함수가 없는 일반 조건을 사용할 수 있다.
④ SELECT문의 6개 절 중에서 SELECT절이 가장 마지막에 실행된다.

13. SQL 문을 실행했을 때 오류가 발생하는 부분으로 가장 적절한 것은?

① SELECT T.COL1, COL2, SUM(COL3) AS "SUM VALUE"
② FROM TAB1 T
③ GROUP BY COL1, COL2
④ ORDER BY COL3;

14. 아래의 SQL 에 대해서 결과값이 <u>다른</u> 것은?

① SELECT CONCAT ('RDBMS', ' SQL') FROM DUAL;
② SELECT 'RDMBS' || ' SQL' FROM DUAL;
③ SELECT 'RDBMS' + ' SQL';
④ SELECT 'RDBMS' & ' SQL' FROM DUAL;

15. 아래 SQL에서 밑줄 친 자리에 쓰인 함수의 결과가 <u>다른</u> 하나는?

```
SELECT _________(5.47) FROM DUAL;
```

① TRUNC
② CEIL
③ FLOOR
④ ROUND

16. 다음 SQL의 수행 결과로 가장 적절한 것은?

<TAB1>

JUMIN
7510231111111

```
SELECT TO_CHAR(TO_DATE(SUBSTR(JUMIN, 1, 6), 'RRMMDD'), 'YYYY-MM-DD')
  FROM TAB1;
```

① 1975-10-23
② 2075-10-23
③ 1975-10-23 00:00:00
④ 2075-10-23 00:00:00

17. 다음 SQL의 수행 결과로 알맞은 것은?

<TAB1>

COL1
1
2
3
NULL

```
SELECT ISNULL(COL1, 3)
  FROM TAB1;
```

①

COL1
1
2
3
NULL

②

COL1
1
2
3
3

③

COL1
1
2
NULL
NULL

④

COL1
1
2
NULL
3

18. 아래 SQL의 실행 결과로 알맞은 것은?
(단, 문자타입인 경우 ''를 붙여서 표현, 숫자타입인 경우 숫자만 전달)

```
SELECT LTRIM(' AB DE ') AS C1,
       INITCAP('ABCDE') AS C2,
       TO_CHAR('123', '999.99') AS C3
  FROM DUAL;
```

①

C1	C2	C3
'AB DE '	'Abcde'	'123.00'

②

C1	C2	C3
' AB DE '	'Abcde'	123.00

③

C1	C2	C3
'ABDE'	'abcde'	'999.99'

④

C1	C2	C3
'AB DE '	'Abcde'	999.99

19. 다음 SQL 수행 결과로 가장 알맞은 것은?

<TAB1>

COL1	COL2	COL3
NULL	10	20
10	NULL	30
20	30	NULL

```
SELECT COALESCE(COL1, COL2, COL3) RESULT
  FROM TAB1;
```

①

RESULT
NULL
NULL
NULL

②

RESULT
10
10
20

③

RESULT
NULL
10
20

④

RESULT
10
30
NULL

20. 아래 SQL의 실행 결과로 알맞은 것은?

<TAB1>

NO	SAL
1	100
2	200
3	300
4	400
NULL	100

<TAB2>

NO	SAL
1	100
3	200
3	300
NULL	300
5	400
5	400

```
SELECT SUM(SAL)
  FROM TAB1
 WHERE NO NOT IN (SELECT NO
                    FROM TAB2
                   GROUP BY NO);
```

① 0
② NULL
③ 600
④ 700

21. 아래 수행 결과로 알맞은 것은?

<TAB1>

COL1	COL2
A	10
B	NULL
B	20
NULL	30
NULL	40
C	20

```
SELECT COUNT(COL1) AS RESULT
  FROM TAB1
 WHERE COL2 >= 30
 GROUP BY COL1;
```

①

RESULT

②

RESULT
NULL

③

RESULT
0

④

RESULT
1

22. 다음 SQL 실행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
10	100
10	200
20	400
30	200
30	300
NULL	100
NULL	500

```
SELECT COL1 AS C1, SUM(COL2) AS C2
  FROM TAB1
 GROUP BY COL1
HAVING SUM(COL2) >= 400;
```

①

C1	C2
10	300
20	400
30	500
NULL	600

②

C1	C2
20	400
30	500
NULL	600

③

C1	C2
20	400
30	500

④

C1	C2
20	400
NULL	500

23. 다음 중 <u>틀린</u> 설명은? (단, DBMS는 ORACLE)

① ORDER BY COMM 시 NULL이 마지막에 배치된다.

② ORDER BY COMM NULLS FIRST 시 NULL이 맨 앞에 배치된다.

③ ORDER BY COMM DESC 시 NULL이 맨 앞에 배치된다.

④ ORDER BY COMM DESC NULLS LAST 시 NULL이 맨 앞에 배치된다.

24. 아래 SQL 수행 결과로 가장 알맞은 것은?

<TAB1>

COL1	COL2
SMITH	10
ALLEN	20
SCOTT	30
FORD	40

```
SELECT COL1
  FROM TAB1
 ORDER BY CASE WHEN MOD(COL2, 3) = 0 THEN 'A'
               ELSE 'B'
          END, COL1;
```

①

COL1
SMITH
ALLEN
SCOTT
FORD

②

COL1
SCOTT
ALLEN
FORD
SMITH

③

COL1
SCOTT
SMITH
ALLEN
FORD

④

COL1
SCOTT
SMITH
FORD
ALLEN

25. 다음 SQL 구문의 결과는?

<TAB1>

COL1	COL2
1	A
2	B
3	
4	C

<TAB2>

COL1	COL2
1	A
2	B
3	B
4	

```
SELECT COUNT(A.COL2)
  FROM TAB1 A JOIN TAB2 B
    ON A.COL2 = B.COL2;
```

① 0

② 1

③ 2

④ 3

26. 다음 SQL 구문의 결과는?

<TAB1>

COL1	COL2
1	A
2	B
3	
4	C

<TAB2>

COL1	COL2
1	A
2	B
3	B
4	

```
SELECT COUNT(TAB1.COL1)
  FROM TAB1, TAB2
 WHERE TAB1.COL2 = TAB2.COL2(+);
```

① 2
② 3
③ 4
④ 5

27. 다음 표준조인에 대한 설명 중 가장 적절하지 <u>않은</u> 것은?
① CROSS JOIN인 두 테이블의 조인 컬럼의 값과 상관없이 항상 모든 경우의 수를 출력한다.
② FULL OUTER JOIN은 LEFT OUTER JOIN 결과와 RIGHT OUTER JOIN 결과를 UNION한 것과 같다.
③ NATURAL JOIN시 같은 이름의 컬럼이 여러 개인 경우 USING절을 사용하여 원하는 컬럼을 선택할 수 있다.
④ INNER JOIN은 줄여서 JOIN으로 전달할 수 있다.

28. SQL의 실행 결과로 가장 적절한 것은?

<TAB1>

NO	CODE
1	A
2	B
3	
4	B
6	D
7	E

<TAB2>

NO	QTY
1	2
2	4
4	6
4	8
5	10
5	10
5	10

```
SELECT COUNT(TAB1.NO) FROM TAB1 LEFT OUTER JOIN TAB2 ON TAB1.NO = TAB2.NO;
SELECT COUNT(DISTINCT TAB1.CODE) FROM TAB1 RIGHT OUTER JOIN TAB2 ON TAB1.NO = TAB2.NO;
```

① 7, 2
② 7, 3
③ 9, 2
④ 9, 3

29. 아래와 같은 테이블 데이터가 있다. SQL에 대한 결과로 가장 알맞은 것은?

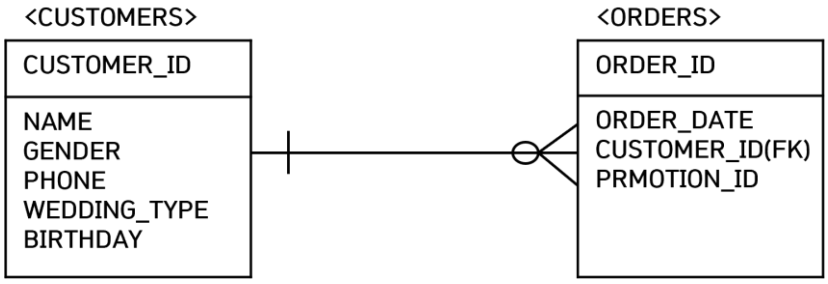
<TAB1>

COL1	COL2
A	10
A	20
A	30
A	30
B	30
B	40

```
SELECT SUM(T1.COL2)
  FROM TAB1 T1
 WHERE T1.COL2 = (SELECT MAX(COL2)
                    FROM TAB1 T2
                   WHERE T1.COL1 = T2.COL1);
```

① 에러
② NULL
③ 90
④ 100

30. 아래 ERD를 참고하여, 다음 SQL을 작성하였을 때 이들 중 실행 결과가 다른 하나는?



① SELECT DISTINCT C.CUSTOMER_ID
FROM CUSTOMERS C, ORDERS O
WHERE C.CUSTOMER_ID = O.CUSTOMER_ID
AND C.WEDDING_TYPE = 'Y'
AND C.GENDER = 'F'
AND TRUNC((SYSDATE - BIRTHDAY)/365) BETWEEN 40 AND 50;

② SELECT C.CUSTOMER_ID
FROM CUSTOMERS C, (SELECT O.CUSTOMER_ID, COUNT(O.PROMOTION_ID) AS ORD_CNT
FROM ORDERS O
GROUP BY O.CUSTOMER_ID
HAVING COUNT(O.PROMOTION_ID) >= 1) I
WHERE C.CUSTOMER_ID = I.CUSTOMER_ID
AND C.WEDDING_TYPE = 'Y'
AND C.GENDER = 'F'
AND TRUNC((SYSDATE - BIRTHDAY)/365) BETWEEN 40 AND 50;

③ SELECT C.CUSTOMER_ID
FROM CUSTOMERS C
WHERE C.WEDDING_TYPE = 'Y'
AND C.GENDER = 'F'
AND TRUNC((SYSDATE - BIRTHDAY)/365) BETWEEN 40 AND 50
AND C.CUSTOMER_ID IN (SELECT O.CUSTOMER_ID
FROM ORDERS O);

④ SELECT C.CUSTOMER_ID
FROM CUSTOMERS C
WHERE C.WEDDING_TYPE = 'Y'
AND C.GENDER = 'F'
AND TRUNC((SYSDATE - BIRTHDAY)/365) BETWEEN 40 AND 50
AND EXISTS (SELECT 'X'
FROM ORDERS O
WHERE C.CUSTOMER_ID = O.CUSTOMER_ID);

31. 다음 SQL 수행 결과로 알맞은 것은?

<EMP>

EMPNO	ENAME	HIREDATE
7369	SMITH	1980/12/17
7499	ALLEN	1981/02/20
7521	WARD	1981/02/22
7566	JONES	1981/04/02
7698	BLAKE	1981/05/01
7782	CLARK	1981/06/09

```
SELECT E1.EMPNO, E1.ENAME, E1.HIREDATE, COUNT(E2.EMPNO) AS CNT
  FROM EMP E1 LEFT OUTER JOIN EMP E2
    ON E1.HIREDATE > E2.HIREDATE
 GROUP BY E1.EMPNO, E1.ENAME, E1.HIREDATE
 ORDER BY E1.HIREDATE;
```

①

EMPNO	ENAME	HIREDATE	CNT
7499	ALLEN	1981/02/20	1
7521	WARD	1981/02/22	2
7566	JONES	1981/04/02	3
7698	BLAKE	1981/05/01	4
7782	CLARK	1981/06/09	5

②

EMPNO	ENAME	HIREDATE	CNT
7369	SMITH	1980/12/17	0
7499	ALLEN	1981/02/20	1
7521	WARD	1981/02/22	2
7566	JONES	1981/04/02	3
7698	BLAKE	1981/05/01	4
7782	CLARK	1981/06/09	5

③

EMPNO	ENAME	HIREDATE	CNT
7369	SMITH	1980/12/17	5
7499	ALLEN	1981/02/20	4
7521	WARD	1981/02/22	3
7566	JONES	1981/04/02	2
7698	BLAKE	1981/05/01	1

④

EMPNO	ENAME	HIREDATE	CNT
7369	SMITH	1980/12/17	5
7499	ALLEN	1981/02/20	4
7521	WARD	1981/02/22	3
7566	JONES	1981/04/02	2
7698	BLAKE	1981/05/01	1
7782	CLARK	1981/06/09	0

32. 서브쿼리 설명으로 가장 적절한 것은?

① 단일 행 서브쿼리는 서브쿼리의 실행 결과가 항상 한 건 이하인 서브쿼리로 IN, ALL 등의 비교 연산자를 사용하여야 한다.

② 다중 행 서브쿼리 비교 연산자는 단일 행 서브쿼리의 비교 연산자로도 사용할 수 있다.

③ 연관 서브쿼리는 주로 메인쿼리에 값을 제공하기 위한 목적으로 사용한다.

④ 서브 쿼리는 항상 메인쿼리에서 읽힌 데이터에 대해 서브쿼리에서 해당 조건이 만족하는지를 확인하는 방식으로 수행된다.

33. 아래 결과를 출력하기 위해 빈칸에 들어갈 문장으로 적절한 것은?

<고객>

고객번호	이름	포인트
1000	홍길동	10000
1001	박길동	9000
1002	최길동	13000
1003	이길동	2000

<상품>

상품번호	상품명	최소포인트	최대포인트
1	A	1	2000
2	B	2001	8000
3	C	8001	12000
4	D	12001	20000

고객번호	상품명
1000	C
1001	C
1002	D
1003	A

① SELECT 고객.고객번호, 상품.상품명
FROM 고객 INNER JOIN 상품
ON 고객.포인트 >= 상품.최소포인트;

② SELECT 고객.고객번호, 상품.상품명
FROM 고객 INNER JOIN 상품
ON 고객.포인트 <= 상품.최대포인트;

③ SELECT 고객.고객번호, 상품.상품명
FROM 고객 JOIN 상품
ON 고객.포인트 BETWEEN 상품.최소포인트 AND 상품.최대포인트;

④ SELECT 고객.고객번호, 상품.상품명
FROM 고객, 상품
ON 고객.포인트 >= 상품.최소포인트(+);

34. 다음 SQL 실행 결과로 가장 알맞은 것은?

<TAB1>

COL1	COL2	COL3
1	A	10
2	A	15
3	B	10
4	B	30
5	B	20

<TAB2>

COL1	COL2	COL3
6	A	10
7	A	30
8	B	10
9	B	30
10	A	20

```
SELECT SUM(COL1)
  FROM TAB1 T1
 WHERE COL3 >= (SELECT AVG(COL3)
                  FROM TAB2 T2
                 WHERE T2.COL2 = T1.COL2);
```

① NULL
② 0
③ 9
④ 23

35. 테이블이 아래와 같을 때, 다음 집합연산자 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
10	A
20	B
30	C
30	D
40	E

<TAB2>

COL1	COL2
10	A
20	A
50	F
NULL	A

<TAB3>

COL1	COL2
30	C
30	D
40	E
NULL	A

```
SELECT *
  FROM (SELECT COL1, COL2
          FROM TAB1
          UNION
         SELECT COL1, COL2
          FROM TAB2)
 MINUS
SELECT COL1, COL2
 FROM TAB3;
```

①

COL1	COL2
10	A
20	B
50	A

②

COL1	COL2
10	A
20	A
20	B
50	F

③

COL1	COL2
10	A
20	A
20	B
50	A
NULL	A

④

COL1	COL2
10	A
10	A
20	A
20	B
50	A
NULL	A

36. 아래 SQL의 결과로 가장 알맞은 것은?

<JUMUN>

JUMUN_NO	PRODUCT_NO	GOGAK_NO	QTY	JUMUN_DATE
1000	1	0001	3	2024/05/24 11:00:56
1001	2	0001	1	2024/05/25 12:01:56
1002	1	0002	2	2024/05/26 09:13:12
1003	1	0002	1	2024/05/27 10:24:30
1004	2	0002	2	2024/05/28 13:01:10

```
SELECT PRODUCT_NO, GOGAK_NO, SUM(QTY) AS TOTAL_QTY
  FROM JUMUN
 GROUP BY GROUPING SETS(PRODUCT_NO, GOGAK_NO, ());
```

①

PRODUCT_NO	GOGAK_NO	TOTAL_QTY
NULL	0001	4
NULL	0002	5
1	NULL	6
2	NULL	3
NULL	NULL	9

②

PRODUCT_NO	GOGAK_NO	TOTAL_QTY
1	NULL	6
2	NULL	3
1	0001	3
1	0002	3
2	0001	1
2	0002	2

③

PRODUCT_NO	GOGAK_NO	TOTAL_QTY
1	NULL	6
2	NULL	3
1	0001	3
1	0002	3
2	0001	1
2	0002	2
NULL	NULL	9

④

PRODUCT_NO	GOGAK_NO	TOTAL_QTY
NULL	0001	4
NULL	0002	5
1	NULL	6
2	NULL	3
1	0001	3
1	0002	3
2	0001	1
2	0002	2
NULL	NULL	9

37. 다음 중 RANK, DENSE_RANK, ROW_NUMBER 결과로 가장 적절한 것은?

<STUDENT>

NO	NAME	JUMSU
1	홍길동	100
2	박길동	90
3	최길동	90
4	이길동	80
5	구길동	70

```
SELECT NO,
        RANK() OVER(ORDER BY JUMSU DESC) AS RANK1,
        DENSE_RANK() OVER(ORDER BY JUMSU DESC) AS RANK2,
        ROW_NUMBER() OVER(ORDER BY JUMSU DESC) AS RANK3
  FROM STUDENT;
```

①

NO	RANK1	RANK2	RANK3
1	1	1	1
2	2	2	2
3	2	2	3
4	3	3	4
5	4	4	5

②

NO	RANK1	RANK2	RANK3
1	1	1	1
2	2	2	2
3	2	2	3
4	4	3	4
5	5	4	5

③

NO	RANK1	RANK2	RANK3
1	1	1	1
2	2	2	2
3	2	2	2
4	3	4	3
5	4	5	4

④

NO	RANK1	RANK2	RANK3
1	1	1	1
2	2	2	2
3	2	3	2
4	4	4	3
5	5	5	4

38. 부서내에서의 급여의 비율을 출력하는 구문으로 가장 적절하지 <u>않은</u> 것은?

① SELECT ENAME, SAL, DEPTNO,
ROUND(SAL/(SELECT SUM(SAL)
FROM EMP E2
WHERE E1.DEPTNO = E2.DEPTNO) * 100, 2) AS SAL_RATIO
FROM EMP E1
ORDER BY DEPTNO, SAL DESC;

② SELECT ENAME, SAL, DEPTNO,
ROUND(RATIO_TO_REPORT(SAL) OVER(PARTITION BY DEPTNO) * 100, 2) AS SAL_RATIO
FROM EMP E1
ORDER BY DEPTNO, SAL DESC;

③ SELECT E1.ENAME, E1.SAL, E1.DEPTNO,
ROUND(E1.SAL/I.SUM_SAL * 100, 2) AS SAL_RATIO
FROM EMP E1, (SELECT DEPTNO, SUM(SAL) AS SUM_SAL
FROM EMP
GROUP BY DEPTNO) I
WHERE E1.DEPTNO = I.DEPTNO
ORDER BY DEPTNO, SAL DESC;

④ SELECT ENAME, SAL, DEPTNO,
ROUND(PERCENT_RANK() OVER(PARTITION BY DEPTNO ORDER BY SAL) * 100, 2) AS SAL_RATIO
FROM EMP E1
ORDER BY DEPTNO, SAL DESC;

39. 다음 SQL문의 실행 결과로 가장 알맞은 것은?

<고객>

고객번호	이름	포인트
1	홍길동	100
2	박길동	110
3	최길동	200
4	이길동	220
5	구길동	80
6	안길동	150

```
SELECT *
  FROM 고객
 ORDER BY 포인트 DESC
OFFSET 2 ROWS
 FETCH FIRST 2 ROWS ONLY;
```

①

고객번호	이름	포인트
6	안길동	150
2	박길동	110

②

고객번호	이름	포인트
4	이길동	220
3	최길동	200

③

고객번호	이름	포인트
4	이길동	220
3	최길동	200
6	안길동	150
2	박길동	110

④

고객번호	이름	포인트
1	홍길동	100
2	박길동	110

40. 아래 실행 결과를 출력하는 SQL로 가장 적절한 것은?

<사원>

사원번호	이름	상위관리자코드
1000	홍길동	NULL
1001	박길동	1000
1002	최길동	1001
1003	이길동	1001
1004	구길동	1002
1005	안길동	1003
1006	송길동	1000
1007	강길동	1006
1008	공길동	1006

```
SELECT 사원번호, 이름, LEVEL
  FROM 사원
  START WITH 사원번호 IN (1006, 1001)
CONNECT BY PRIOR 상위관리자코드 = 사원번호;
```

①

사원번호	이름	LEVEL
1001	박길동	1
1006	송길동	1

②

사원번호	이름	LEVEL
1001	박길동	1
1006	송길동	1
1000	홍길동	2

③

사원번호	이름	LEVEL
1001	박길동	1
1006	송길동	1
1000	홍길동	2
1000	홍길동	2

④

사원번호	이름	LEVEL
1001	박길동	1
1006	송길동	1
1002	최길동	2
1003	이길동	2
1007	강길동	2
1008	공길동	2

41. 계층형 질의에서 CONNECT BY 절에 사용되며, 현재 읽은 컬럼을 지정하는 구문은 무엇인가?

① START WITH

② PRIOR

③ NOCYCLE

④ ORDER SIBLINGS BY

42. 다음 빈칸에 들어갈 문장으로 가장 적절한 것은?

<판매>

지역	Q1	Q2
서울	10	20
경기	30	40

SELECT

FROM 판매

_____

<결과>

지역	구분	판매량
서울	Q1	10
서울	Q2	20
경기	Q1	30
경기	Q2	40

① UNPIVOT (판매량 FOR 구분 IN (Q1, Q2))

② UNPIVOT (판매량 FOR 구분 FOR (Q1, Q2))

③ PIVOT (판매량 FOR 구분 IN (Q1, Q2))

④ PIVOT (판매량 FOR 구분 FOR (Q1, Q2))

43. 다음 SQL 실행 결과로 가장 알맞은 것은?

<TAB1>

COL1
AXXX.AYYY.AXAX.
AAXYXY.XYYY.AXYAXY.

```
SELECT REGEXP_REPLACE(COL1, 'A(X|Y)+\.') FROM TAB1;
```

①

COL1
AX
XYYY

②

COL1
AX
AXYYY.AXY

③

COL1
NULL
XYYY

④

COL1
NULL
AXYYY.AXY

44. 다음 SQL 실행 결과로 가장 알맞은 것은?

```
SELECT REGEXP_SUBSTR('ORA-00600 Oracle SQL-Server 50', '[^0-9]+') "REGEXPR_SUBSTR"
  FROM DUAL;
```

① 50
② 00600 50
③ ORA-
④ ORA- Oracle SQL-Server

45. 다음 SQL중 입력오류가 발생할 문장으로 가장 적절한 것은?

```
CREATE TABLE TAB1(
COL1    VARCHAR(10) PRIMARY KEY,
COL2    NUMBER NOT NULL,
COL3    CHAR(10) NOT NULL,
COL4    DATE NOT NULL);
```

① INSERT INTO TAB1 VALUES(1, 10, 'AAA', SYSDATE);
② INSERT INTO TAB1 VALUES('0001', '20', 'BBB', SYSTIMESTAMP);
③ INSERT INTO TAB1 VALUES('0002', '30', '1000', CURRENT_DATE);
④ INSERT INTO TAB1 VALUES('0003', 40, 'CCC', '2024/01/01');

46. 아래 SQL 실행 결과로 가장 적절한 것은?

```
CREATE TABLE TAB1(COL1 NUMBER, COL2 NUMBER);

INSERT INTO TAB1 VALUES(1,10);
INSERT INTO TAB1 VALUES(2,20);
INSERT INTO TAB1 VALUES(3,30);
COMMIT;

ALTER TABLE TAB1 ADD (COL3 NUMBER);

INSERT INTO TAB1 VALUES(4,40,100);
UPDATE TAB1 SET COL2 = 50 WHERE COL1 = 1;
DELETE TAB1 WHERE COL1 = 3;

ALTER TABLE TAB1 DROP COLUMN COL1;

ROLLBACK;

SELECT SUM(COL2 + COL3) FROM TAB1;
```

① 110
② 120
③ 130
④ 140

47. 다음 문장이 차례대로 수행된 이후의 데이터 값으로 가장 적절한 것은?

<TAB1>

COL1	NUMBER
COL2	VARCHAR2(10)
COL3	NUMBER

<TAB1>

COL1	COL2	COL3
1	A	10
2	B	20
3	C	30

```
ALTER TABLE TAB1 ADD COL4 CHAR(5);
ALTER TABLE TAB1 MODIFY COL4 DEFAULT 'AAA';
INSERT INTO TAB1 VALUES(4, 'D', 40, NULL);
INSERT INTO TAB1(COL1, COL2, COL3) VALUES(5,'E',50);
```

①

COL1	COL2	COL3	COL4
1	A	10	NULL
2	B	20	NULL
3	C	30	NULL
4	D	40	NULL
5	E	50	NULL

②

COL1	COL2	COL3	COL4
1	A	10	AAA
2	B	20	AAA
3	C	30	AAA
4	D	40	AAA
5	E	50	AAA

③

COL1	COL2	COL3	COL4
1	A	10	NULL
2	B	20	NULL
3	C	30	NULL
4	D	40	NULL
5	E	50	AAA

④

COL1	COL2	COL3	COL4
1	A	10	NULL
2	B	20	NULL
3	C	30	NULL
4	D	40	AAA
5	E	50	AAA

48. 다음 설명 중 가장 적절하지 <u>않은</u> 것은? (단, DBMS는 오라클)

① 데이터 타입을 변경할 경우에는 반드시 빈 컬럼이어야 한다.

② 컬럼 사이즈는 언제든지 늘릴 수 있다.

③ 컬럼 추가 시 DEFAULT값을 선언하면 기존 데이터의 새로운 컬럼 값은 DEFAULT값이 된다.

④ 컬럼은 동시에 여러 개를 삭제할 수 없다.

49. 제약조건에 대한 설명 중 가장 적절하지 <u>않은</u> 것은?

① PRIMARY KEY는 여러 컬럼으로 구성하여 생성할 수 있다.

② UNIQUE 제약조건에는 NULL값을 허용하지 않는다.

③ CREATE TABLE AS SELECT문으로 테이블 복제 시 NOT NULL속성은 복제된다.

④ FOREIGN KEY는 부모-자식 관계 중 자식 테이블에 생성한다.

50. 다음 중 유저가 갖는 권한에 대한 설명으로 가장 적절한 것은?

```
SYSTEM) GRANT SELECT, INSERT ON SCOTT.EMP TO HR WITH GRANT OPTION;
SYSTEM) GRANT CREATE VIEW TO HR WITH ADMIN OPTION;
HR) GRANT SELECT, INSERT ON SCOTT.EMP TO HONG;
HR) GRANT CREATE VIEW TO HONG;
SYSTEM) REVOKE INSERT ON SCOTT.EMP FROM HR;
SYSTEM) REVOKE CREATE VIEW FROM HR;
```

① HR 계정에게 부여된 SCOTT.EMP에 대한 SELECT 권한도 함께 회수된다.
② HR 유저에게 부여된 CREATE TABLE 권한 회수 시 HONG에게 부여된 CREATE TABLE 권한도 함께 회수되었다.
③ HR이 HONG에게 부여한 EMP 테이블의 조회 권한은 SYSTEM 계정에서 직접 회수가 가능하다.
④ HR 유저에게 부여된 EMP 테이블 입력 권한 회수 시 HONG에게 부여된 권한도 함께 회수되었다.

3회차

1. 다음 중 아래에서 설명하는 데이터모델의 개념으로 가장 적절한 것은?

학생이라는 엔터티에서 학년이라는 속성 값의 범위는 1~4 사이의 정수이며, 주민번호 속성은 13자리 이내 문자열로 정의할 수 있다.

① 도메인
② 릴레이션
③ 시스템카탈로그
④ 속성사전

2. 엔터티 - 인스턴스 - 속성 - 속성값에 대한 관계 설명 중 틀린 것을 고르시오.
① 한 개의 엔터티는 두 개 이상의 인스턴스의 집합이어야 한다.
② 한 개의 엔터티는 두 개 이상의 속성을 갖는다.
③ 하나의 속성은 하나 이상의 속성값을 가진다.
④ 하나의 엔터티의 인스턴스는 다른 엔터티의 인스턴스간의 관계인 Paring을 가진다.

3. 속성에 대한 설명으로 가장 적절하지 않은 것은?
① 하나의 속성에 여러 개의 값이 있는 다중값일 경우 별도의 엔터티를 이용하여 분리한다.
② 정해진 주식별자에 함수적 종속성을 가져야 한다.
③ 업무상 인스턴스로 관리하고자 하는 더 이상 분리되지 않는 최소의 데이터 단위를 말한다.
④ 엔터티에 속한 속성은 엔터티에 대한 추상적인 값을 갖는다.

4. 엔터티간 1:1, 1:M 과 같이 관계의 기수성을 나타내는 것을 무엇이라 하는가?
① 관계명
② 관계차수
③ 관계선택성
④ 관계정의

5. 다음 주식별자에 대한 설명 중 가장 적절하지 않은 것은?
① 주식별자에 의해 엔터티 내의 모든 인스턴스들이 유일하게 구분되어야 한다.
② 주식별자로 지정되더라도 속성 값으로 NULL이 들어갈 수 있다.
③ 주식별자를 구성하는 속성의 수는 유일성을 만족하는 최소의 수가 되어야 한다.
④ 지정된 주식별자의 값은 자주 변하지 않는 것이어야 한다.

6. 다음이 설명하는 정규화로 가장 적절한 것은?

테이블의 컬럼이 원자성(한 속성이 하나의 값을 갖는 특성)을 갖도록 테이블을 분해하는 단계

① 제 1 정규화
② 제 2 정규화
③ 제 3 정규화
④ 제 4 정규화

7. 다음이 설명하는 관계로 가장 적절한 것은?

두 엔터티나 두 속성 간에 동시에 발생할 수 없는 관계를 의미합니다. 즉, 하나의 엔터티나 속성이 특정한 경우 다른 엔터티나 속성은 해당 경우가 될 수 없음을 나타냅니다.

① 상호종속적
② 상호포괄적
③ 상호배타적
④ 상호일관적

8. 다음 중 트랜잭션에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?
① 두 엔터티의 관계가 서로 필수적일 때 하나의 트랜잭션을 형성한다.
② 두 엔터티가 서로 독립적 수행이 가능하다면 선택적 관계로 표현한다.
③ 하나의 트랜잭션은 부분 COMMIT이 가능하다.
④ 하나의 트랜잭션에는 여러 SELECT, INSERT, DELETE, UPDATE 등이 포함될 수 있다.

9. NULL에 대한 설명으로 <u>틀린</u> 것은?
① 값이 존재하지 않거나 확정되지 않은 값을 의미한다.
② NULL과의 비교연산은 FALSE(거짓)를 리턴한다.
③ NULL과의 수치연산은 NULL 값을 리턴한다.
④ 공백과 같은 ASCII 값을 가진다.

10. 다음이 설명하는 식별자로 가장 적절한 것은?

다른 엔터티 참조 없이 엔터티 내부에서 스스로 생성되는 식별자

① 보조식별자
② 인조식별자
③ 본질식별자
④ 내부식별자

11. 다음 중 DBMS 특징이 <u>아닌</u> 것은?
① DBMS에 저장된 데이터는 다른 사용자에게 공유될 수 없다.
② 데이터 무결성을 유지 할 수 있다.
③ 실시간 접근, 자료의 계속적인 변화의 적용에 유리하다.
④ 인증된 사용자만이 참조 할 수 있는 보안기능이 제공된다.

12. 테이블 생성 시 주의 할 사항으로 적절하지 않은 것은?
① 컬럼 뒤에 데이터 유형은 꼭 지정되어야 한다.
② 테이블명과 컬럼명은 숫자로 시작해도 무관하다.
③ 테이블 생성시 대소문자 구분은 하지 않는다.
④ 소유자가 다를 경우 같은 이름의 테이블을 생성할 수 있다.

13. 다음 설명 중 틀린 하나는?
① ORDER BY 절에 SELECT 절에 정의되지 않은 컬럼을 사용할 수 있다.
② 테이블 별칭을 선언하면 컬럼 앞의 구분자는 반드시 테이블명 대신 테이블 별칭을 사용한다.
③ GROUP BY 절을 사용하는 경우 ORDER BY 절에는 GROUP BY절에 정의되지 않은 컬럼을 사용할 수 있다.
④ SELECT 문은 ORDER BY 절이 가장 나중에 실행된다.

14. 다음 중 DISTINCT에 대한 설명으로 가장 적절하지 않은 것은?
① DISTINCT 뒤에 나열되는 컬럼들의 중복값을 한 번만 출력하기 위해 사용한다.
② SELECT 문에서만 사용 가능하다.
③ DISTINCT 뒤에 나열되는 컬럼의 순서에 따라 결과 집합의 수가 달라진다.
④ DISTINCT 뒤에 * 를 사용할 수 있다.

15. 다음 SQL 수행 결과로 가장 적절한 것은?

```
SELECT ROUND(TO_DATE('2024-02-20 14:00:00', 'YYYY-MM-DD HH24:MI:SS'), 'MONTH') AS D1,
       TRUNC(TO_DATE('2024-09-12 09:00:00', 'YYYY-MM-DD HH24:MI:SS')) AS D2
 FROM DUAL;
```

①

D1	D2
2024-03-01 00:00:00	2024-09-01 00:00:00

②

D1	D2
2025-01-01 00:00:00	2024-09-01 00:00:00

③

D1	D2
2024-03-01 00:00:00	2024-09-12 00:00:00

④

D1	D2
2024-03-20 00:00:00	2024-09-12 00:00:00

16. 다음 함수 사용시 결과값이 올바르지 않은 것은?
① CEIL(-12.345) = -13
② FLOOR(-12.345) = -13
③ MOD(8,3) = 2
④ SIGN(0) = 0

17. 다음 함수의 결과로 가장 적절한 것은?

```
SELECT LTRIM('ORACLE', 'A') AS C1,
       SUBSTR('SQL-SERVER', 3, 3) AS C2,
       LENGTH(REPLACE('SQL-SERVER', 'E')) AS C3
 FROM DUAL;
```

①

C1	C2	C3
ORCLE	L-S	8

②

C1	C2	C3
ORACLE	L	8

③

C1	C2	C3
ORACLE	L-S	8

④

C1	C2	C3
ORACLE	L-S	2

18. 다음 SQL중 실행 결과가 <u>다른</u> 하나는?

①
```
SELECT DECODE(DEPTNO, 10, DECODE(JOB, 'CLERK', 'A', 'B'), 20, 'C', 'D')
  FROM EMP;
```

②
```
SELECT CASE WHEN (DEPTNO = 10 AND JOB = 'CLERK') THEN 'A' ELSE 'B'
            WHEN DEPTNO = 20 THEN 'C' ELSE 'D'
       END
  FROM EMP;
```

③
```
SELECT CASE WHEN DEPTNO = 10 THEN CASE WHEN JOB = 'CLERK' THEN 'A' ELSE 'B' END
            WHEN DEPTNO = 20 THEN 'C' ELSE 'D'
       END
  FROM EMP;
```

④
```
SELECT CASE DEPTNO WHEN 10 THEN CASE WHEN JOB = 'CLERK' THEN 'A' ELSE 'B' END
                           WHEN 20 THEN 'C' ELSE 'D'
       END
  FROM EMP;
```

19. 다음 중 정상적으로 실행되지 <u>않는</u> 문장은? (단, DBMS는 오라클)

① SELECT 100 + '1' FROM DUAL;

② SELECT TO_DATE('20240101', 'YYYYMMDD') - 10 FROM DUAL;

③ SELECT TO_DATE('11', 'DD') + 10 FROM DUAL;

④ SELECT NVL(100, 'NULL') FROM DUAL;

20. 다음 SQL 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2	COL3
A	NULL	100
A	200	300
B	100	NULL
B	300	200
NULL	100	0

```
SELECT COUNT(COL1) RESULT FROM TAB1 WHERE COL3 < 100 GROUP BY COL1;
SELECT COUNT(COL1) RESULT FROM TAB1 WHERE COL1 IS NOT NULL GROUP BY COL1
HAVING SUM(COL2) > 500;
```

①

RESULT
NULL

RESULT
NULL

②

RESULT
0

RESULT
0

③

RESULT
0

RESULT

④

RESULT

RESULT

21. 아래 SQL의 실행 결과로 알맞은 것은?

<TAB1>

COL1	COL2	COL3
A	1	NULL
B	2	10
B	NULL	20
NULL	3	30

```
SELECT SUM(COL2 + COL3) FROM TAB1 WHERE COL1 IS NOT NULL;
```

① 0
② NULL
③ 12
④ 33

22. 다음 SQL 문장 중 COL1 값이 널(NULL)이 아닌 경우를 찾아내는 문장으로 가장 적절한 것은?

① SELECT * FROM TAB1 WHERE COL1 IS NOT NULL;
② SELECT * FROM TAB1 WHERE COL1 <> NULL;
③ SELECT * FROM TAB1 WHERE COL1 != NULL;
④ SELECT * FROM TAB1 WHERE COL1 NOT NULL;

23. 다음 중 에러가 나지 <u>않는</u> 문장은? (단, DBMS는 오라클)

① SELECT COL1 컬럼1, AVG(COL2) 평 균
FROM TAB1;

② SELECT COL1 컬럼1, AVG(COL2) 평균
FROM TAB1
GROUP BY COL2;

③ SELECT COL1 AS 컬럼1, AVG(COL2) AS 평균
FROM TAB1
WHERE AVG(COL2) >= 100
GROUP BY 컬럼1;

④ SELECT COL1 AS 컬럼1, AVG(COL2) AS 평균
FROM TAB1
GROUP BY COL1
HAVING AVG(COL2) >= 100;

24. 다음 수행 결과로 가장 적절한 것은?

<EMP>

EMPNO	NUMBER
ENAME	VARCHAR2(10)
SAL	NUMBER
JUMIN	CHAR(13)
DEPTNO	NUMBER

EMPNO	ENAME	SAL	JUMIN	DEPTNO
1000	SMITH	2500	8012011234567	10
1001	ALLEN	2000	9011072212345	10
1002	FORD	3400	9506232221234	20
1003	SCOTT	3800	9801181112345	20
1004	KING	4000	9908091234432	30

```
SELECT EMPNO
  FROM EMP
 ORDER BY TO_CHAR(TO_NUMBER(SUBSTR(JUMIN, 3, 2)));
```

①

EMPNO
1003
1002
1004
1001
1000

②

EMPNO
1003
1001
1000
1002
1004

③

EMPNO
1001
1003
1000
1002
1004

④

EMPNO
1000
1001
1002
1003
1004

25. 아래 SQL 수행 결과로 가장 적절한 것은? (단, DBMS는 오라클)

<EMP>

ENAME	DEPTNO	SAL
SMITH	10	2000
SCOTT	10	1800
FORD	10	3200
KING	20	4000
JAMES	20	5000
ADAMS	20	NULL

```
SELECT ENAME, DEPTNO, SAL
  FROM EMP
 ORDER BY DEPTNO, SAL DESC;
```

①

ENAME	DEPTNO	SAL
SCOTT	10	1800
SMITH	10	2000
FORD	10	3200
KING	20	4000
JAMES	20	5000
ADAMS	20	NULL

②

ENAME	DEPTNO	SAL	ENAME
SCOTT	10	1800	SCOTT
SMITH	10	2000	SMITH
FORD	10	3200	FORD
ADAMS	20	NULL	ADAMS
KING	20	4000	KING
JAMES	20	5000	JAMES

③

ENAME	DEPTNO	SAL
FORD	10	3200
SMITH	10	2000
SCOTT	10	1800
ADAMS	20	NULL
JAMES	20	5000
KING	20	4000

④

ENAME	DEPTNO	SAL	ENAME
JAMES	20	5000	JAMES
KING	20	4000	KING
ADAMS	20	NULL	ADAMS
FORD	10	3200	FORD
SMITH	10	2000	SMITH
SCOTT	10	1800	SCOTT

26. 아래 SQL 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
1	A
2	B
3	C
4	D

<TAB2>

COL1	COL2
1	A
2	B
2	B
4	C
5	C

```
SELECT COUNT(TAB1.COL1) AS CNT
  FROM TAB1 LEFT OUTER JOIN TAB2
    ON TAB1.COL2 = TAB2.COL2
   AND TAB1.COL1 = TAB2.COL1;
```

① 3
② 4
③ 5
④ 6

27. 다음 FROM 절의 JOIN 형태에 대한 설명 중 올바르지 <u>못한</u> 것은?
① INNER JOIN은 WHERE 절에서 사용하던 JOIN 조건을 FROM 절에서 정의하겠다는 표시이다.
② INNER JOIN 사용 시, USING 조건절이나 ON 조건절을 반드시 사용해야 한다.
③ RIGHT OUTER JOIN 결과와 LEFT OUTER JOIN 결과는 항상 다르다.
④ RIGHT OUTER JOIN, LEFT OUTER JOIN에서 OUTER는 생략 가능하다.

28. 다음 중 SELECT절에 사용하는 서브쿼리인 스칼라 서브쿼리에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?
① 하나의 로우에 해당하는 스칼라 서브쿼리 결과 건수는 1건 이하여야 한다.
② 하나의 로우에 해당하는 스칼라 서브쿼리 결과가 0건이면 생략된다.
③ 스칼라 서브쿼리는 반드시 한 컬럼만 출력이 가능하다.
④ 메인쿼리와 스칼라 서브쿼리의 연결 조건이 필요하다면 반드시 스칼라 서브쿼리에 정의해야 한다.

29. 다음 서브쿼리 결과로 가장 적절한 것은?

<EMP>

NAME	HIREDATE	SAL
TURNER	1981/09/08	1500
ADAMS	1987/05/23	1100
JAMES	1981/10/03	1000
FORD	1981/12/03	3000
MILLER	1982/01/23	1300

```
SELECT SUM(SAL)
  FROM EMP
 WHERE HIREDATE > (SELECT HIREDATE
                     FROM TAB1
                    WHERE NAME = 'JAMES');
```

① 3800
② 4100
③ 5400
④ 6900

30. 다음 서브쿼리 결과로 가장 적절한 것은?

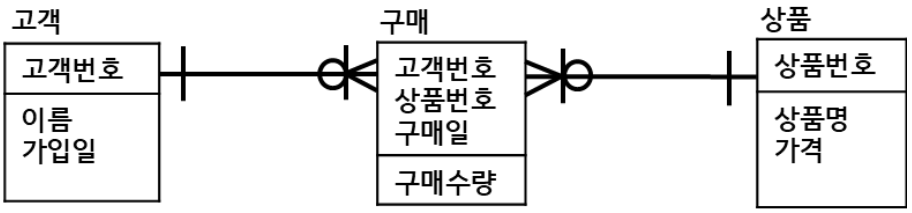
<EMPLOYEES>

NAME	HIREDATE	SAL	DEPTNO
TURNER	1981/09/08	1500	10
ADAMS	1987/05/23	1100	10
JAMES	1981/10/03	1000	20
FORD	1981/12/03	3000	20
MILLER	1982/01/23	3000	20

```
SELECT SUM(SAL)
  FROM EMPLOYEES
 WHERE (DEPTNO, SAL) IN (SELECT DEPTNO, MAX(SAL)
                           FROM EMPLOYEES
                          GROUP BY DEPTNO);
```

① 1500
② 3000
③ 4500
④ 7500

31. 다음 ERD를 보고 고객별로 가장 최근에 구매한 상품명(다수가능)을 출력하는 SQL로 가장 적절한 것은?



① SELECT G.고객번호, G.구매일, P.상품명
FROM 구매 G, 상품 P, (SELECT 고객번호, MAX(구매일) 최근구매일
FROM 구매
GROUP BY 고객번호) I
WHERE G.고객번호 = I.고객번호
AND G.구매일 = I.최근구매일
AND G.상품번호 = P.상품번호;

② SELECT G.고객번호, G.구매일, P.상품명
FROM 고객 G, 상품 P, (SELECT 고객번호, MAX(구매일) 최근구매일
FROM 구매
GROUP BY 고객번호) I
WHERE G.고객번호 = I.고객번호
AND G.상품번호 = P.상품번호;

③ SELECT C.이름, G.구매일, P.상품명
FROM 고객 C, 구매 G, 상품 P, (SELECT 고객번호, MIN(구매일) 최근구매일
FROM 구매
GROUP BY 고객번호) I
WHERE G.고객번호 = I.고객번호
AND G.구매일 = I.최근구매일
AND G.상품번호 = P.상품번호
AND C.고객번호 = G.고객번호;

④ SELECT C.이름, G.구매일, P.상품명
FROM 고객 C, 구매 G, 상품 P
WHERE G.상품번호 = P.상품번호
AND C.고객번호 = G.고객번호;

32. 다음 쿼리의 수행 결과로 적절한 것은?

<TAB1>

NAME	CODE	FARE
AAA	0001	500
BBB	0001	100
CCC	0002	300
DDD	0004	200

<TAB2>

CODE	STATUS
0001	CLOSED
0002	OPEN
0003	CLOSED
0004	OPEN

```
DELETE FROM TAB1
 WHERE CODE IN (SELECT CODE
                  FROM TAB2
                 WHERE STATUS = 'OPEN');

SELECT SUM(FARE)
  FROM TAB1;
```

① 100
② 400
③ 600
④ 800

33. 다음 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
A	100
B	200
C	300
D	400

<TAB2>

CODE	FARE
0001	250
0002	350

```
SELECT SUM(COL2)
  FROM TAB1
 WHERE COL2 < ANY(SELECT FARE
                    FROM TAB2);
```

① 100
② 300
③ 600
④ 1000

34. 다음 집합 연산자에 대한 설명 중 <u>틀린</u> 것은 무엇인가? (단, DBMS는 오라클)

① UNION 연산자는 조회 결과에 대한 합집합을 나타내며 정렬된 결과를 출력해준다.
② UNION ALL 연산자는 조회 결과를 정렬하고 중복되는 데이터를 한 번만 표현한다.
③ INTERSECT 연산자는 조회 결과에 대한 교집합을 의미한다.
④ MINUS 연산자는 조회 결과에 대한 차집합을 의미한다.

35. 아래 쿼리 결과와 같은 결과를 갖는 빈칸에 들어갈 문장으로 가장 적절한 것은?

```
SELECT DEPTNO, SUM(SAL) AS SUM_SAL
  FROM EMP
 GROUP BY DEPTNO
 UNION ALL
SELECT NULL DEPTNO, SUM(SAL) AS SUM_SAL
  FROM EMP;

SELECT DEPTNO, SUM(SAL) AS SUM_SAL
  FROM EMP
 GROUP BY _____;
```

① ROLLUP(DEPTNO)
② ROLLUP(SAL)
③ ROLLUP(DEPTNO, SAL)
④ ROLLUP(DEPTNO, ())

36. 순위관련 WINDOW 함수에 대한 설명 중 가장 적절하지 <u>않은</u> 것은?
① RANK함수는 동일한 값에 대해서는 동일한 순위를 부여한다.
② DENSE_RANK 함수는 RANK 함수처럼 동일한 값에 대해 동일한 순위를 부여하나, 동순위가 여럿 존재하더라도 다음 순위가 이어진다.
③ PERCENT_RANK 함수는 각 값의 누적된 순위를 부여할 수 있다.
④ RANK 함수가 동일한 값에 대해서는 동일한 순위를 부여하는데 반해, ROW_NUMBER 함수는 고유한 순위를 부여한다.

37. 다음 출력 결과를 갖도록 하는 빈칸의 문장으로 가장 적절한 것은?

<EMP>

EMPNO	ENAME	DEPTNO	SAL
7934	MILLER	10	1300
7782	CLARK	10	2450
7839	KING	10	5000
7369	SMITH	20	800
7876	ADAMS	20	1100
7566	JONES	20	2975
7788	SCOTT	20	3000
7902	FORD	20	3000

<RESULT>

EMPNO	ENAME	DEPTNO	SAL	RESULT
7934	MILLER	10	1300	3750
7782	CLARK	10	2450	8750
7839	KING	10	5000	7450
7369	SMITH	20	800	1900
7876	ADAMS	20	1100	4875
7566	JONES	20	2975	7075
7788	SCOTT	20	3000	8975
7902	FORD	20	3000	6000

```
SELECT EMPNO, ENAME, DEPTNO, SAL,
        SUM(SAL) OVER(PARTITION BY DEPTNO ORDER BY SAL
                                  _____) AS RESULT
  FROM EMP;
```

① RANGE BETWEEN UNBOUNDED PRECEDING AND 1 FOLLOWING
② RANGE BETWEEN 1 PRECEDING AND 1 FOLLOWING
③ ROWS BETWEEN UNBOUNDED PRECEDING AND 1 FOLLOWING
④ ROWS BETWEEN 1 PRECEDING AND 1 FOLLOWING

38. 다음 중 Top N Query에 대한 설명 중 <u>틀린</u> 것은?

① 윈도우 함수를 사용하여 상위 N개에 대한 값을 추출할 수 있으나 단일 Query로 표현 불가하다.
② ROWNUM을 사용한 방식은 ROWNUM 할당 전에 먼저 순서대로 데이터를 정렬한 뒤 ROWNUM을 부여 후 추출하는 것이 좋다.
③ FETCH 절을 사용하면 단일 Query로도 정렬 순서대로의 상위 N개에 대한 값을 추출할 수 있다.
④ SQL-Server의 TOP(N) 쿼리를 사용하면 정렬 순서대로 상위 N개 데이터만 출력 가능하다.

39. 아래 실행 결과를 출력하는 SQL로 가장 적절한 것은?

<사원>

사원번호	이름	상위관리자코드	지역
1000	홍길동	NULL	서울
1001	박길동	1000	경기
1002	최길동	1001	경기
1003	이길동	1001	인천
1004	구길동	1002	서울
1005	안길동	1003	서울
1006	송길동	1000	경기
1007	강길동	1006	경기
1008	공길동	1006	인천

```
SELECT 사원번호, 이름, LEVEL
    FROM 사원
  WHERE 지역 = '경기'
  START WITH 상위관리자코드 IS NULL
CONNECT BY 상위관리자코드 = PRIOR 사원번호;
```

①

사원번호	이름	LEVEL
1001	박길동	2
1002	최길동	3
1006	송길동	2
1007	강길동	3

②

사원번호	이름	LEVEL
1000	홍길동	1
1001	박길동	2
1002	최길동	3
1006	송길동	2
1007	강길동	3

③ 공집합

④

사원번호	이름	LEVEL
1000	홍길동	1

40. 다음 수행 결과로 가장 적절한 것은?

<DEPARTMENT>

DEPTNO	DNAME	PART	BUILD
101	컴퓨터공학과	100	정보관
102	멀티미디어공학과	100	멀티미디어관
103	소프트웨어공학과	100	소프트웨어관
201	전자공학과	200	전자제어관
202	기계공학과	200	기계실험관
203	화학공학과	200	화학실습관
301	문헌정보학과	300	인문관
100	컴퓨터정보학부	10	
200	메카트로닉스학부	10	
300	인문사회학부	20	
10	공과대학		
20	인문대학		

```
SELECT DEPTNO, DNAME, LEVEL, CONNECT_BY_ROOT(DNAME) AS ROOT
  FROM DEPARTMENT
 START WITH PART IS NULL
CONNECT BY PRIOR PART = DEPTNO;
```

① 공집합

②

DEPTNO	DNAME	LEVEL	ROOT
10	공과대학	1	공과대학
20	인문대학	1	인문대학

③

DEPTNO	DNAME	LEVEL	ROOT
10	공과대학	1	공과대학
100	컴퓨터정보학부	2	공과대학
101	컴퓨터공학과	3	공과대학
102	멀티미디어공학과	3	공과대학
103	소프트웨어공학과	3	공과대학
200	메카트로닉스학부	2	공과대학
201	전자공학과	3	공과대학
202	기계공학과	3	공과대학
203	화학공학과	3	공과대학

④

DEPTNO	DNAME	LEVEL	ROOT
10	공과대학	1	공과대학
100	컴퓨터정보학부	2	공과대학
101	컴퓨터공학과	3	공과대학
102	멀티미디어공학과	3	공과대학
103	소프트웨어공학과	3	공과대학
200	메카트로닉스학부	2	공과대학
201	전자공학과	3	공과대학
202	기계공학과	3	공과대학
203	화학공학과	3	공과대학
20	인문대학	1	인문대학
300	인문사회학부	2	인문대학
301	문헌정보학과	3	인문대학

41. 다음 SQL의 실행 결과를 얻기 위한 빈칸에 들어갈 값으로 가장 적절한 것은?

<TAB1>

성별	2023	2024
남자	10	20
여자	30	40

<결과>

성별	연도	판매량
남자	2023	10
남자	2024	20
여자	2023	30
여자	2024	40

```
 SELECT *
   FROM TAB1
UNPIVOT (_____ FOR _____ IN ("2023", "2024"));
```

① 판매량, 연도
② 성별, 연도
③ 성별, 판매량
④ 판매량, 판매량

42. 다음 SQL 실행 결과로 알맞은 것은?

```
SELECT LENGTH(REGEXP_REPLACE('REGEXP \. ESCAPE CHARACTER, A|B : A OR B', '[A-z|0-9\. ]'))
  FROM DUAL;
```

① 2
② 3
③ 4
④ 5

43. DML에 대한 설명으로 가장 적절한 것은?
① DELETE 사용 시 FROM 문구는 생략이 불가능하다.
② 일부 데이터 DELETE 시 WHERE 절은 반드시 붙이지 않아도 된다.
③ 반드시 COMMIT 또는 ROLLBACK을 수행하여 TRANSACTION을 종료해야 한다.
④ UPDATE 사용 시 동시에 여러 컬럼 수정은 불가능하다.

44. COMMIT 이후의 데이터 상태로 옳지 <u>않은</u> 것은?
① 데이터에 대한 변경 사항이 데이터베이스에 영구 저장된다.
② 변경된 행에 대한 잠금이 풀리고 다른 사용자들이 행을 조작할 수 있다.
③ 이전 데이터는 영원히 되돌릴 수 없다.
④ COMMIT을 수행한 사용자만 결과를 볼 수 있다.

45. 컬럼 변경 시 주의 사항으로 옳지 <u>않은</u> 것은?
① 컬럼의 크기를 늘릴 수는 있지만 줄일 수는 없다.
② 컬럼이 NULL 값만 가지고 있으면 데이터 유형을 변경할 수 있다.
③ 컬럼에 NULL 값이 없을 경우에만 NOT NULL 제약조건을 추가할 수 있다.
④ 컬럼의 DEFAULT 값을 바꾸면 변경 작업 이후 발생하는 행 삽입에만 영향을 미친다.

46. 제약조건의 설명 중 가장 적절하지 <u>않은</u> 것은?
① 기본키는 테이블에 저장된 행 데이터를 고유하게 식별하기 위한 키이다.
② 테이블에 저장된 행 데이터를 고유하게 식별하기 위해 고유키를 정의한다.
③ 기본키는 고유키와 외래키 제약을 합쳐놓은 것이다.
④ NOT NULL 은 NULL 값의 삽입을 금지한다.

47. 아래와 같이 테이블 및 데이터가 생성된 경우 추가 실행이 불가능한 문장은?
(단, 보기 순서대로 실행됨을 가정)

```
CREATE TABLE TAB1(COL1 NUMBER, COL2 NUMBER);
INSERT INTO TAB1 VALUES(100, 100);
COMMIT;
```

① ALTER TABLE TAB1 ADD (COL3 NUMBER, COL4 VARCHAR2(10));
② ALTER TABLE TAB1 ADD COL5 NUMBER NOT NULL;
③ ALTER TABLE TAB1 MODIFY COL2 DEFAULT 100 NOT NULL;
④ ALTER TABLE TAB1 DROP COLUMN COL4;

48. 다음 문장의 수행 후 TAB2의 조회 결과로 가장 적절한 것은?

<TAB1>

NO	NAME
1	A
2	B
3	C
4	D

<TAB2>

NO	NAME	CLASS_NO
1000	SMITH	1
1001	ALLEN	2
1002	FORD	3
1004	SCOTT	3

```
ALTER TABLE TAB2 ADD FOREIGN KEY(CLASS_NO) REFERENCES TAB1(NO) ON DELETE SET NULL;
DELETE FROM TAB1 WHERE NAME = 'C';
```

① DELETE 실행 오류

②

NO	NAME	CLASS_NO
1000	SMITH	1
1001	ALLEN	2

③

NO	NAME	CLASS_NO
1000	SMITH	1
1001	ALLEN	2
1002	FORD	NULL
1004	SCOTT	NULL

④

NO	NAME	CLASS_NO
1000	SMITH	1
1001	ALLEN	2
1002	FORD	3
1004	SCOTT	3

49. 다음 설명 중 가장 적절하지 <u>않은</u> 것은?

① 외래키를 생성 한 경우 부모 테이블의 참조키 컬럼을 삭제할 수 없다.

② 제약 조건 추가 시 제약조건 이름을 명시하지 않을 수 있다.

③ 이미 존재하는 컬럼에 대해 NOT NULL 제약조건 추가 시 반드시 MODIFY로 처리한다.

④ NULL값이 삽입되어 있는 경우 UNIQUE 제약조건을 추가할 수 없다.

50. 권한에 대한 설명으로 가장 적절한 것은?

① 권한은 테이블 소유자만이 부여할 수 있다.

② 테이블에 대한 조회 권한 부여 시 즉시 반영되지 않고 재접속을 해야 조회가 가능하다.

③ 롤에 있는 권한을 회수한 이후 롤을 부여받은 유저는 해당 권한을 갖지 않게 된다.

④ WITH ADMIN OPTION을 통해 부여받은 테이블 조회 권한을 다른 유저에게 부여할 수 있다.

4회차

1. 다음이 설명하는 모델링의 특징으로 가장 적절한 것은?

누구나 이해하기 쉽게 하기 위해 대상에 대한 애매모호함을 제거하고 정확하게 현상을 기술하는 것을 의미한다.

① 명확화 ② 추상화 ③ 단순화 ④ 그룹화

2. 엔터티 분류 중 유형과 무형에 따른 분류가 아닌 것은?
① 유형엔터티
② 개념엔터티
③ 사건엔터티
④ 행위엔터티

3. 다음 중 속성에 대한 설명 중 가장 적절하지 않은 것은?
① 기본속성에 대한 예를 들자면 입사날짜이다.
② 설계속성은 업무상 필요한 데이터 외에 업무를 규칙화하기 위하여 기본속성을 변형하는 속성이다.
③ 파생속성은 다른 속성에 영향을 받아 발생하는 속성이다.
④ 파생속성은 가급적 많이 정의할수록 좋은 속성이다.

4. 다음 중 관계를 구성하는 요소가 아닌 것은?
① 관계명
② 차수(Cardinality)
③ 선택성(Optionality)
④ 관계정의

5. 다음 중 아래 개념이 설명하는 관계로 가장 적절한 것은?

부모 엔터티의 주식별자를 상속받아 자식 엔터티에서 외부식별자이면서 주식별자로 사용하는 관계

① 식별관계
② 비식별관계
③ 필수관계
④ 선택관계

6. 다음 엔터티는 어떤 정규화를 위배한 것인가? (단, 고객번호+상품명이 PK임)

<이용이력>

고객번호	상품명	가격	이용날짜
1000	A	1000	2024/01/01
1000	B	1500	2024/01/31
1001	A	1000	2024/02/15
1001	C	2000	2024/03/20

① 제 1 정규화
② 제 2 정규화
③ 제 3 정규화
④ 제 4 정규화

7. 관계(Relationship)와 조인(Join)에 대한 설명으로 가장 적절하지 않은 것은?
① 관계란 서로 다른 엔터티의 인스턴스들 간의 존재하는 논리적인 연관성을 의미한다.
② 정규화를 거쳐 분리된 엔터티는 서로 관계를 맺지 않아도 된다.
③ 관계를 맺는 엔터티를 다시 연결하는 과정을 조인이라고 한다.
④ 행위 관계는 엔터티 간의 어떤 행위가 있는 것을 의미한다.

8. 트랜잭션의 특징 중 옳지 않은 것은?
① 일관성(consistency)
② 원자성(atomicity)
③ 지속성(durability)
④ 중복성(duplication)

9. NULL에 대한 설명으로 틀린 것은?
① COUNT는 NULL을 세지 않는다.
② NULL + 100은 NULL을 리턴한다.
③ NULL이 포함된 컬럼의 SUM 값은 항상 NULL이다.
④ NULL과의 비교 연산은 FALSE를 리턴한다.

10. 다음의 주식별자가 나타내는 특징을 가장 잘 설명한 것은?

주식별자를 구성하는 속성 중에서 유일성을 만족하는 최소한의 속성으로 구성한다.

① 유일성
② 최소성
③ 단일성
④ 존재성

11. 테이블에 대한 설명 중 가장 적절한 것은?
① 같은 테이블을 동시에 두 계정으로 소유할 수 있다.
② 테이블명은 중복될 수 없지만, 소유자가 다른 경우 같은 이름으로 생성 가능하다.
③ 하나의 행의 하나의 컬럼에는 둘 이상의 값이 삽입될 수 있다.
④ 테이블 생성 시 각 컬럼의 데이터 유형을 정의할 수 있고, 생성 이후에는 변경이 불가하다.

12. 관계형 데이터베이스에 대한 특징 중 가장 적절하지 않은 것은?
① 데이터의 무결성을 보장할 수 있다.
② 데이터를 분류, 정렬, 탐색하는 속도가 빠르다.
③ 기존의 작성된 스키마를 수정하기 어렵다.
④ 데이터베이스의 부하를 분석하기 쉽다.

13. 다음 중 SELECT문에 대한 설명으로 가장 적절하지 않은 것은? (단, DBMS는 오라클)

① 실행 순서는 FROM > WHERE > GROUP BY > HAVING > SELECT > ORDER BY 순 이다.

② FROM 절은 생략 불가하다.

③ GROUP BY 절은 NULL 그룹을 출력하지 않는다.

④ 일반적으로 SELECT 절에서는 * 와 컬럼명을 동시에 사용할 수 없다.

14. 다음 중 오류가 발생하는 문장으로 가장 적절한 것은?

①
```
SELECT COL1, T.COL2, T.COL3
  FROM TAB1 T
 WHERE COL1 >= 100;
```

②
```
SELECT T.COL2, SUM(T.COL1)
  FROM TAB1 T
 WHERE COL1 >= 100
 GROUP BY COL2;
```

③
```
SELECT T.COL2, SUM(T.COL1) AS SUM_VALUE
  FROM TAB1 T
 GROUP BY T.COL2
 ORDER BY SUM_VALUE, T.COL2;
```

④
```
SELECT T.COL2, SUM(T.COL1), SUM(T.COL3)
  FROM TAB1 T
 GROUP BY T.COL2
 ORDER BY T.COL3;
```

15. 다음 함수의 출력 결과로 가장 적절하지 않은 것은?

① DATE_FORMAT('2024-01-01','%Y-%m-%d') : 2024-01-01

② FLOOR(3.5) = 3

③ POWER(3,3) = 9

④ LAST_DAY(TO_DATE('2024-01-01','YYYY-MM-DD')) : 2024-01-31

16. 다음 SQL문의 실행 결과로 가장 적절한 것은?

<EMP>

EMPNO	DEPTNO	JOB
0001	10	CLERK
0002	10	MANAGER
0003	20	CLERK
0004	30	ANALYST

```
SELECT DECODE(DEPTNO, 10, DECODE(JOB, 'CLERK', 'A', 'B'), 'C') AS RESULT
  FROM EMP
 ORDER BY EMPNO;
```

①

RESULT
A
B
C
C

②

RESULT
A
B
A
C

③

RESULT
A
A
B
B

④

RESULT
A
A
B
C

17. 아래 SQL의 결과로 가장 적절한 것은?

```
SELECT ROUND(TO_DATE('2024-08-24 12:00:00','YYYY-MM-DD HH24:MI:SS')) COL1,
       ROUND(TO_DATE('2024-08-24 12:00:01','YYYY-MM-DD HH:MI:SS'))  COL2
  FROM DUAL;
```

①

COL1	COL2
2024-08-24	2024-08-25

②

COL1	COL2
2024-08-25	2024-08-25

③

COL1	COL2
2024-08-25	2024-08-26

④

COL1	COL2
2024-08-26	2024-08-26

18. 다음 SQL의 실행 결과에 대한 해석으로 가장 적절한 것은?

```
SELECT NEXT_DAY(ADD_MONTHS(HIREDATE, 6),'월요일')
  FROM EMP;
```

① 각 직원의 입사날짜로부터 6일 후 첫 번째 월요일에 해당하는 날짜
② 각 직원의 입사날짜로부터 6일 후 두 번째 월요일에 해당하는 날짜
③ 각 직원의 입사날짜로부터 6개월 후 첫 번째 월요일에 해당하는 날짜
④ 각 직원의 입사날짜로부터 6개월 후 두 번째 월요일에 해당하는 날짜

19. 다음 SQL 실행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
NULL	100
100	200
200	100

```
SELECT SUM(NULLIF(COL1,100) + COL2) FROM TAB1;
```

① 300
② 400
③ 500
④ 600

20. 아래 SQL 실행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
A	1
B	2
C	3
D	

<TAB2>

COL1	COL2
A	2
B	3
C	

```
SELECT COUNT(COL1)
  FROM TAB1
 WHERE NOT EXISTS(SELECT 'X'
                    FROM TAB2
                   WHERE TAB1.COL2 = TAB2.COL2);
```

① 1
② 2
③ 3
④ 4

21. 아래 SQL 실행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2	COL3
A	1000	10
B	1200	10
C	3000	20
D	2000	30
E	4000	20
F	3600	10

```
SELECT COUNT(COL1)
  FROM TAB1
WHERE COL2 NOT BETWEEN 2000 AND 3000
   AND COL3 NOT IN (10, 20);
```

① NULL
② 0
③ 1
④ 2

22. 아래 SQL 실행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2	COL3
100	A	10
200	A	20
300	A	30
0	B	10
400	NULL	20
500	B	30
300	B	40

```
SELECT COL2, SUM(COL3) AS C1, MIN(COL3) AS C2, MAX(COL3) AS C3
  FROM TAB1
 WHERE COL1 > 100
 GROUP BY COL2;
```

①

COL2	C1	C2	C3
NULL	20	20	20
A	50	20	30
B	70	30	40

②

COL2	C1	C2	C3
NULL	NULL	NULL	NULL
A	60	10	30
B	80	10	40

③

COL2	C1	C2	C3
A	50	20	30
B	70	30	40

④

COL2	C1	C2	C3
A	60	10	30
B	80	10	40

23. 다음의 정렬 결과로 가장 적절한 것은?

<TAB1>

ID
AA
ABC
BI
B
BAA

```
SELECT ID
  FROM TAB1
 ORDER BY CASE WHEN ID > 'B' THEN 'A' ELSE ID END, ID;
```

①

ID
AA
ABC
B
BI
BAA

②

ID
B
AA
BI
ABC
BAA

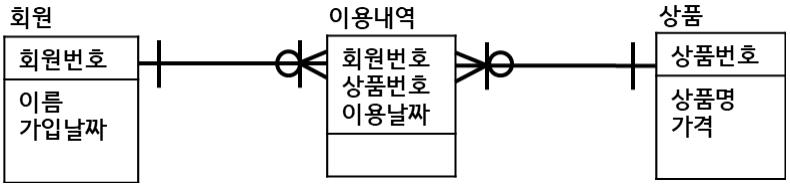
③

ID
BAA
BI
AA
ABC
B

④

ID
BI
BAA
AA
ABC
B

24. 다음 ERD를 참고하여 모든 회원의 상품 이용횟수와 총 이용가격을 출력하는 SQL로 가장 적절한 것은?



① SELECT 회원.회원번호, COUNT(상품.상품번호), SUM(상품.가격)
FROM 회원, 이용내역, 상품
WHERE 회원.회원번호 = 이용내역.회원번호
AND 이용내역.상품번호 = 상품.상품번호
GROUP BY 회원.회원번호;

② SELECT 회원.회원번호, COUNT(상품.상품번호), SUM(상품.가격)
FROM 회원 LEFT OUTER JOIN 이용내역
ON 회원.회원번호 = 이용내역.회원번호 JOIN 상품
ON 이용내역.상품번호 = 상품.상품번호
GROUP BY 회원.회원번호;

③ SELECT 회원.회원번호, COUNT(이용내역.상품번호), SUM(상품.가격)
FROM 회원 LEFT OUTER JOIN 이용내역
ON 회원.회원번호 = 이용내역.회원번호 LEFT OUTER JOIN 상품
ON 이용내역.상품번호 = 상품.상품번호
GROUP BY 회원.회원번호;

④ SELECT 회원.회원번호, COUNT(상품.상품번호), SUM(상품.가격)
FROM 회원, 이용내역, 상품
WHERE 회원.회원번호 = 이용내역.회원번호(+)
AND 이용내역.상품번호 = 상품.상품번호
GROUP BY 회원.회원번호;

25. 다음의 오라클 표준을 ANSI로 가장 잘 표현한 것은?

```
SELECT E.ENAME, E.SAL, E.DEPTNO, D.DNAME
  FROM EMP E, DEPT D
 WHERE E.DEPTNO = D.DEPTNO
   AND E.SAL > 3000
 ORDER BY E.ENAME;
```

① SELECT E.ENAME, E.SAL, E.DEPTNO, D.DNAME
FROM EMP E JOIN DEPT D
ON E.DEPTNO = D.DEPTNO
AND E.SAL > 3000
ORDER BY E.ENAME;

② SELECT E.ENAME, E.SAL, E.DEPTNO, D.DNAME
FROM EMP E INNER JOIN DEPT D
USING DEPTNO
AND E.SAL > 3000
ORDER BY E.ENAME;

③ SELECT E.ENAME, E.SAL, E.DEPTNO, D.DNAME
FROM EMP E INNER JOIN DEPT D
ON E.DEPTNO = D.DEPTNO
WHERE E.SAL > 3000
ORDER BY E.ENAME;

④ SELECT E.ENAME, E.SAL, E.DEPTNO, D.DNAME
FROM EMP E LEFT JOIN DEPT D
ON E.DEPTNO = D.DEPTNO
WHERE E.SAL > 3000
ORDER BY E.ENAME;

26. 다음 중 실행 오류가 발생하는 조인 문법은?

① SELECT *
FROM TAB1 LEFT JOIN TAB2
ON TAB1.COL1 = TAB2.COL1;

② SELECT *
FROM TAB1 JOIN TAB2
ON (TAB1.COL1 = TAB2.COL1);

③ SELECT *
FROM TAB1 INNER JOIN TAB2
USING (COL1);

④ SELECT *
FROM TAB1 NATURAL JOIN TAB2
USING (COL1);

27. 서브쿼리에 대한 설명으로 가장 적절한 것은?

① FROM 절에 사용하는 서브쿼리를 스칼라 서브쿼리라고 한다.
② 단일행 서브쿼리는 비교 연산자 사용이 불가하다.
③ 서브쿼리가 메인쿼리 컬럼을 가지고 있을 경우 비연관 서브쿼리라고 한다.
④ 연관 서브쿼리는 메인쿼리가 먼저 수행된 후에 서브쿼리에서 조건이 맞는지 확인할 때 주로 사용한다.

28. 다음 출력 결과로 가장 알맞은 것은?

<TAB1>

COL1	COL2
A	10
B	20
C	20
D	NULL

<TAB2>

COL1	COL2
10	AAA
20	BBB
30	CCC

<TAB3>

COL1	COL2
A	100
B	200
C	300
C	400

```
SELECT COL1, COL2,
        (SELECT COL2 FROM TAB2 WHERE TAB1.COL2 = TAB2.COL1) AS RESULT1,
        (SELECT MAX(COL2) FROM TAB3 WHERE TAB1.COL1 = TAB3.COL1) AS RESULT2
  FROM TAB1;
```

①

COL1	COL2	RESULT1	RESULT2
A	10	AAA	100
B	20	BBB	200
C	20	BBB	400

②

COL1	COL2	RESULT1	RESULT2
A	10	AAA	100
B	20	BBB	200
C	20	BBB	400
D	NULL	0	0

③

COL1	COL2	RESULT1	RESULT2
A	10	AAA	100
B	20	BBB	200
C	20	BBB	400
D	NULL	NULL	NULL

④

COL1	COL2	RESULT1	RESULT2
A	10	AAA	100
B	20	BBB	200
C	20	BBB	400
D	NULL	NULL	0

29. 아래 SQL 중 결과가 다른 하나는?

① SELECT P.CODE, P.NAME, P.PRICE
FROM PRODUCT P, (SELECT CODE, MAX(PRICE) AS MAX_PRICE
FROM PRODUCT
GROUP BY CODE) I
WHERE P.CODE = I.CODE
AND P.PRICE = I.MAX_PRICE;

② SELECT CODE, NAME, PRICE
FROM PRODUCT
WHERE (CODE, PRICE) IN (SELECT CODE, MAX(PRICE)
FROM PRODUCT
GROUP BY CODE);

③ SELECT CODE, NAME, PRICE
FROM PRODUCT P1
WHERE PRICE = (SELECT MAX(PRICE)
FROM PRODUCT P2
WHERE P1.CODE = P2.CODE);

④ SELECT CODE, NAME, (SELECT PRICE
FROM PRODUCT
WHERE P1.CODE = P2.CODE)
FROM PRODUCT P1;

30. 아래 SQL에 대한 실행 결과로 가장 알맞은 것은?

<TAB1>

COL1	COL2
A	10
B	20
C	20
D	40

<TAB2>

COL1	COL2
10	100
10	200
20	300

```
SELECT SUM(TAB2.COL2)
  FROM TAB1, TAB2
 WHERE TAB1.COL2 = TAB2.COL1;
```

① 600
② 700
③ 800
④ 900

31. 다음 SQL 수행 결과로 가장 적절한 것은?

<EMP>

ENAME	HIREDATE	SAL
CLARK	1981/06/09	1000
TURNER	1981/09/08	2000
KING	1981/11/17	3000
SCOTT	1987/04/19	4000

```
SELECT SUM(E2.SAL) AS RESULT
  FROM EMP E1, EMP E2
 WHERE E1.HIREDATE >= E2.HIREDATE
 GROUP BY E1.ENAME;
```

①

RESULT
1000
2000
3000
4000

②

RESULT
3000
6000
10000

③

RESULT
1000
3000
6000
10000

④

RESULT
10000
6000
3000
1000

32. 다음 수행 결과로 가장 적절한 것은?

<PRODUCT>

NAME	CODE
A	0001
B	0002
C	0003
D	NULL

<PROMOTION>

CODE	FARE
0001	100
0002	150
0003	200
0004	250

```
SELECT NAME, NVL((SELECT FARE
                    FROM PROMOTION P2
                   WHERE P1.CODE = P2.CODE), 100) PRO_FARE
  FROM PRODUCT P1;
```

①

NAME	PRO_FARE
A	100
B	150
C	200

②

NAME	PRO_FARE
A	100
B	150
C	200
D	NULL

③

NAME	PRO_FARE
A	100
B	150
C	200
D	100

④

NAME	PRO_FARE
A	100
B	150
C	200
D	0

33. 집합 연산자 사용시 주의사항에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?

① 두 집합의 컬럼 수가 일치해야 한다.

② 두 집합의 각 컬럼의 데이터 유형이 일치해야 한다.

③ 두 집합 중 위의 집합의 컬럼명을 전체 집합의 컬럼명으로 가져간다.

④ 위 집합의 컬럼의 사이즈보다 아래 집합의 컬럼 사이즈가 큰 경우 연산이 불가하다.

34. 아래 쿼리 결과와 같은 결과를 갖는 빈칸에 들어갈 문장으로 가장 적절한 것은?

```
SELECT DEPTNO, NULL AS JOB, SUM(SAL) AS SUM_SAL
  FROM EMP
 GROUP BY DEPTNO
 UNION ALL
SELECT NULL, JOB, SUM(SAL) AS SUM_SAL
  FROM EMP
 GROUP BY JOB
 UNION ALL
SELECT NULL, NULL, SUM(SAL) AS SUM_SAL
  FROM EMP;

SELECT DEPTNO, JOB, SUM(SAL) AS SUM_SAL
  FROM EMP
 GROUP BY _____;
```

① ROLLUP(DEPTNO, JOB)

② GROUPING SETS(DEPTNO, JOB)

③ GROUPING SETS(DEPTNO, JOB, NULL)

④ CUBE(DEPTNO, JOB)

35. 아래 SQL 문장 실행 결과로 가장 적절한 것은?

<EMP>

EMPNO	ENAME	DEPTNO	SAL
7934	MILLER	10	1300
7782	CLARK	10	2450
7839	KING	10	5000
7369	SMITH	20	800
7876	ADAMS	20	1100
7566	JONES	20	2975
7788	SCOTT	20	3000
7902	FORD	20	3000

```
SELECT EMPNO, ENAME, DEPTNO, SAL,
        LAG(SAL, 2, 0) OVER(PARTITION BY DEPTNO ORDER BY SAL) LAG_VALUE
  FROM EMP;
```

①

EMPNO	ENAME	DEPTNO	SAL	LAG_VALUE
7934	MILLER	10	1300	NULL
7782	CLARK	10	2450	NULL
7839	KING	10	5000	1300
7369	SMITH	20	800	NULL
7876	ADAMS	20	1100	NULL
7566	JONES	20	2975	800
7788	SCOTT	20	3000	1100
7902	FORD	20	3000	2975

②

EMPNO	ENAME	DEPTNO	SAL	LAG_VALUE
7934	MILLER	10	1300	0
7782	CLARK	10	2450	0
7839	KING	10	5000	1300
7369	SMITH	20	800	0
7876	ADAMS	20	1100	0
7566	JONES	20	2975	800
7788	SCOTT	20	3000	1100
7902	FORD	20	3000	2975

③

EMPNO	ENAME	DEPTNO	SAL	LAG_VALUE
7934	MILLER	10	1300	NULL
7782	CLARK	10	2450	1300
7839	KING	10	5000	2450
7369	SMITH	20	800	NULL
7876	ADAMS	20	1100	800
7566	JONES	20	2975	1100
7788	SCOTT	20	3000	2975
7902	FORD	20	3000	3000

④

EMPNO	ENAME	DEPTNO	SAL	LAG_VALUE
7934	MILLER	10	1300	0
7782	CLARK	10	2450	1300
7839	KING	10	5000	2450
7369	SMITH	20	800	0
7876	ADAMS	20	1100	800
7566	JONES	20	2975	1100
7788	SCOTT	20	3000	2975
7902	FORD	20	3000	3000

36. MIN 함수와 동일하게 사용할 수 있는 WINDOW 함수는?

① LAG

② RATIO_TO_REPORT

③ LEAD

④ FIRST_VALUE

37. 다음 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
A	10
B	20
C	30
D	40

```
SELECT RATIO_TO_REPORT(COL2) OVER() AS C1,
       CUME_DIST() OVER(ORDER BY COL2) AS C2,
       ROUND(PERCENT_RANK() OVER(ORDER BY COL2),2) AS C3
  FROM TAB1;
```

①

C1	C2	C3
0.1	0.25	0
0.2	0.5	0.33
0.3	0.75	0.67
0.4	1	1

②

C1	C2	C3
0.1	0.1	0
0.2	0.3	0.33
0.3	0.6	0.67
0.4	1	1

③

C1	C2	C3
0.1	0.1	0.1
0.2	0.3	0.3
0.3	0.6	0.6
0.4	1	1

④

C1	C2	C3
0.1	0.25	0.1
0.2	0.5	0.3
0.3	0.75	0.6
0.4	1	1

38. 다음 수행 결과로 가장 적절한 것은?

<EMP>

EMPNO	ENAME	DEPTNO	SAL
7934	MILLER	10	1300
7782	CLARK	10	2450
7839	KING	10	5000
7369	SMITH	20	800
7876	ADAMS	20	1100
7566	JONES	20	2975
7788	SCOTT	20	3000
7902	FORD	20	3000

```
SELECT SUM(SAL)
  FROM (SELECT ENAME, DEPTNO, SAL,
                    NTILE(2) OVER(PARTITION BY DEPTNO ORDER BY SAL) AS GN
          FROM EMP)
 WHERE (DEPTNO = 10 AND GN = 1) OR (DEPTNO = 20 AND GN = 2);
```

① 7300
② 8625
③ 9750
④ 10275

39. 다음 결과를 얻기 위한 SQL 문장으로 적절하지 <u>않은</u> 것은?

<EXAM>

이름	성적
홍길동	98
박길동	80
최길동	80
김길동	75
이길동	70

<결과>

이름	성적
박길동	80
최길동	80
김길동	75

①
```
SELECT 이름, 성적
  FROM EXAM
 ORDER BY 성적 DESC
OFFSET 1 ROW
 FETCH FIRST 3 ROWS ONLY;
```

② SELECT TOP(3) WITH TIES 이름, 성적
FROM EXAM
ORDER BY 성적 DESC;

③ SELECT 이름, 성적
FROM (SELECT 성적, ROW_NUMBER() OVER(ORDER BY 성적 DESC) AS RN
FROM EXAM)
WHERE RN BETWEEN 2 AND 4;

④ SELECT 이름, 성적
FROM (SELECT 성적, DENSE_RANK() OVER(ORDER BY 성적 DESC) AS RN
FROM EXAM)
WHERE RN BETWEEN 2 AND 3;

40. 다음과 같은 데이터 상황에서 계층형 질의절을 완성하기 위해 필요한 표현식은?

<EMP>

EMPNO	NAME	MGR
0001	SMITH	0002
0002	ALLEN	0001

```
SELECT *
  FROM EMP
 START WITH EMPNO = '0001'
CONNECT BY MGR = PRIOR EMPNO;
```

① CONNECT_BY_ROOT
② SYS_CONNECT_BY_PATH
③ NOCYCLE
④ ORDER SIBLINGS BY

41. 다음 수행 결과로 가장 적절한 것은?

<DEPARTMENT>

DEPTNO	DNAME	PART	BUILD
101	컴퓨터공학과		정보관
102	멀티미디어공학과	100	멀티미디어관
103	소프트웨어공학과	100	소프트웨어관
201	문헌정보학과	200	인문관
100	컴퓨터정보학부		
200	인문사회학부		

```
SELECT SYS_CONNECT_BY_PATH(DNAME, '-')
  FROM DEPARTMENT
 WHERE DEPTNO = 103
  START WITH PART IS NULL
CONNECT BY PART = PRIOR DEPTNO ;
```

① -소프트웨어공학과
② -컴퓨터정보학부-소프트웨어공학과
③ -소프트웨어공학과-컴퓨터정보학부
④ -컴퓨터정보학부

42. PIVOT과 UNPIVOT에 대한 설명으로 가장 적절한 것은?
① PIVOT은 교차표 형태의 데이터를 TIDY 데이터로 변경하는 문법이다.
② UNPIVOT은 LONG 데이터를 WIDE 데이터로 변환하는 기법이다.
③ UNPIVOT시 쌓을 컬럼을 지정할 수 없다.
④ PIVOT 시 FOR 앞에는 반드시 집계함수(SUM, AVG 등)의 형태여야 한다.

43. 다음 SQL 실행 결과로 가장 적절한 것은?

```
SELECT REGEXP_REPLACE('031-234-4567', '\d+', 'XXX', 1, 2) FROM DUAL;
```

① 031-234-4567
② XX1-234-4567
③ 031-XXX-4567
④ 031-234-XXX

44. 다음 SQL 실행 결과로 가장 적절한 것은?

```
SELECT REGEXP_SUBSTR('123-234-4545-233', '((\d+)-(\d+))-((\d+)-(\d+))', 1, 1, NULL, 4) FROM DUAL;
```

① 233
② 234
③ 4545-233
④ 4545

45. 다음 SQL 중 수행이 불가능 한 것은?

<TAB1>

NO	NUMBER
NAME	VARCHAR2(10)
JUMIN	CHAR(13)
DEPTNO	NUMBER

<TAB2>

NO	NUMBER
NAME	VARCHAR2(10)
JUMIN	CHAR(13)

① INSERT INTO TAB2 SELECT NO, NAME, JUMIN FROM TAB1;
② INSERT INTO TAB1(NO, NAME, JUMIN) AS SELECT * FROM TAB2;
③ INSERT INTO TAB1 SELECT NO, NAME, JUMIN, NULL FROM TAB2;
④ INSERT INTO TAB2 (SELECT NO, NAME, JUMIN FROM TAB1);

46. 다음 설명 중 가장 적절하지 <u>않은</u> 것은?
① COMMIT을 한 이후에는 ROLLBACK을 수행해도 이전 값으로 돌아갈 수 없다.
② INSERT 한 이후 컬럼 추가 시 INSERT 값은 자동 저장되어 ROLLBACK 할 수 없다.
③ ROLLBACK을 한 명령을 다시 ROLLBACK으로 취소할 수 없다.
④ SAVEPOINT 지점이 COMMIT 이전일때 해당 SAVEPOINT까지 ROLLBACK 시도 시 COMMIT이후 까지만 ROLLBACK 된다.

47. 테이블 생성 시 주의 사항으로 옳지 <u>않은</u> 것은?
① 테이블 생성시 대소문자 구분은 하지 않아도 된다.
② 문자 데이터 유형은 별도로 크기를 지정하지 않아도 된다.
③ 날짜 유형은 별도로 크기를 지정하지 않아도 된다.
④ 컬럼에 대한 제약조건을 추가하는 경우 CONSTRAINT를 사용한다.

48. 테이블 복제에 대한 설명 중 가장 적절하지 <u>않은</u> 것은?
① CREATE TABLE 테이블명 AS SELECT 문으로 테이블 복제가 가능하다.
② 테이블 복제 시 컬럼 순서 및 데이터 유형도 복제된다.
③ 테이블에 생성한 PRIMARY KEY 도 함께 복제된다.
④ PRIMARY KEY 나 UNIQUE 설정 없이 부여된 NOT NULL 속성은 함께 복제된다.

49. 외래키에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?
① 외래키 생성 시 참조 테이블의 참조키에 반드시 기본키 또는 고유키가 생성되어 있어야 한다.
② 외래키 생성 후 부모 테이블은 자식 데이터가 있을 경우 삭제 불가하다.
③ ON DELETE CASCADE 옵션으로 외래키 생성 시 부모 데이터만 삭제되고 자식데이터는 그대로 남는다.
④ 자식 테이블은 UPDATE, INSERT 시 제약을 받는다.

50. 뷰에 대한 설명 중 가장 적절하지 <u>않은</u> 것은?
① 뷰는 가상의 테이블이기에 물리적으로 구현되어 있지 않으며 저장공간을 차지하지 않는다.
② 원본 테이블이 노출되지 않으므로 데이터를 안전하게 보호할 수 있다.
③ 뷰를 참조하는 또 다른 뷰는 생성 불가하다.
④ 기본 테이블이 삭제되면 그 테이블을 참조하여 만든 뷰 역시 삭제된다.

5회차

1. 다음이 설명하는 모델링 개념으로 가장 적절한 것은?

사용자 관점의 데이터베이스 스키마를 통합하여 데이터베이스의 전체 논리적 구조를 정의하는 단계로 전체 데이터베이스의 개체, 속성, 관계, 데이터 타입 등을 정의한다.

① 외부 스키마
② 개념 스키마
③ 논리 스키마
④ 내부 스키마

2. 엔터티에 대한 설명으로 가장 적절한 것은?
① 누락된 프로세스의 경우 이후 추가하거나 수정할 수 없다.
② 사용되지 않는 고립 엔터티는 제거를 고려한다.
③ 다른 엔터티와 관계를 꼭 갖지 않아도 된다.
④ 주로 약어를 사용하여 엔터티 이름을 정한다.

3. 다음 중 단일 속성이 아닌 것은?
① 이름
② 성별
③ 핸드폰번호
④ 주소

4. 두 개의 엔터티 사이의 관계 도출 시 고려사항이 아닌 것은?
① 두 개의 엔터티 사이에 정보의 조합이 발생되는가?
② 두 개의 엔터티 사이에 관련 있는 연관규칙이 존재하는가?
③ 업무기술서, 장표에 관계연결에 대한 규칙이 서술되어 있는가?
④ 업무기술서, 장표에 관계연결을 가능하게 하는 명사가 있는가?

5. 다음 중 주식별자의 특징이 아닌 것은?
① 유일성
② 최소성
③ 불변성
④ 종속성

6. 다음 정규화에 대한 설명으로 가장 적절한 것은?

이행적 종속을 없애도록 테이블을 분리하는 단계

① 제 1 정규화
② 제 2 정규화
③ 제 3 정규화
④ 제 4 정규화

7. 관계에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?
① 주문항목은 반드시 주문이 있어야만 존재할 수 있으므로 존재 관계를 갖는다.
② 학생과 수업 간의 관계는 등록이라는 행위를 통해 형성되므로 행위 관계를 갖는다.
③ 관계를 맺는다는 의미는 자식의 식별자를 부모에 상속하는 일이다.
④ 관계는 존재에 의한 관계와 행위에 의한 관계로 분류한다.

8. 트랜잭션의 특성 그 의미와 맞는 것은?
① 일관성 : 정의된 연산들은 모두 성공적으로 실행되던지 아니면 전혀 실행되지 않은 상태로 남아 있어야 한다.
② 원자성 : 트랜잭션이 실행되기 전 데이터의 내용이 잘못되면 실행 이후에도 내용이 잘못 되어 있지 않다.
③ 고립성 : 트랜잭션이 실행되는 도중 다른 트랜잭션의 영향을 받아 잘못된 결과를 만들어서는 안 된다
④ 연관성 : 트랜잭션이 성공적으로 수행되면 그 트랜잭션이 갱신한 데이터베이스의 내용은 영구적으로 저장된다.

9. 다음 중 NULL만으로 구성된 컬럼을 계산하여 NULL이 리턴되지 <u>않는</u> 함수는?
① COUNT
② SUM
③ AVG
④ MIN

10. 다음 식별자에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?
① 업무에 의해 만들어지는 식별자를 본질식별자라고 한다.
② 꼭 필요하지 않지만 관리의 편이성 등의 이유로 인위적으로 만들어지는 식별자를 인조식별자라 한다.
③ 본질식별자가 단순한 구성을 가질 때 주로 인조식별자를 생성한다.
④ 인조식별자를 사용하면 중복 데이터 발생 가능성이 있어 데이터 품질이 저하된다.

11. 다음이 설명하는 데이터 무결성의 종류는?

테이블의 기본키를 구성하는 컬럼은 NULL 값이나 중복값을 가질 수 없다.

① 개체 무결성
② 참조 무결성
③ 도메인 무결성
④ NULL 무결성

12. 다음이 설명하는 용어로 가장 적절한 것은?

데이터베이스의 속성(ATTRIBUTE, 또는 필드)이 가질 수 있는 값들의 범위 또는 집합을 의미한다.

① 릴레이션
② 도메인
③ 튜플
④ 키

13. 다음 중 SQL 분류로 적절하지 <u>않은</u> 것은?
① DDL - ALTER
② TCL - ROLLBACK
③ DCL - COMMIT
④ DML - UPDATE

14. 다음의 컬럼 별칭 정의가 가장 적절하지 <u>않은</u> 것은?
① SELECT ENAME 이름, SAL AS Salary
② SELECT ENAME, SAL, DEPTNO AS DEPT NUMBER
③ SELECT ENAME, SAL, JOB AS "JOB**"
④ SELECT ENAME, SAL, COMM AS SAL_COMM

15. 다음 날짜 연산의 결과로 가장 적절한 것은?

```
SELECT ADD_MONTHS(TO_DATE('2024/08/24 10:00:00', 'YYYY/MM/DD HH24:MI:SS') - 10, -3) + 10/24/60
  FROM DUAL;
```

① 2024/05/14 10:10:00
② 2024/05/14 11:00:00
③ 2024/05/24 10:00:10
④ 2024/05/24 10:10:00

16. 다음의 연산 결과가 <u>다른</u> 하나는?
① SELECT TRUNC(TO_DATE('2024/05/14', 'YYYY/MM/DD'), 'MONTH') FROM DUAL;
② SELECT ROUND(TO_DATE('2024/05/14', 'YYYY/MM/DD'), 'MONTH') FROM DUAL;
③ SELECT TO_DATE('2024/05', 'YYYY/MM') FROM DUAL;
④ SELECT LAST_DAY(TO_DATE('2024/05/14', 'YYYY/MM/DD')) FROM DUAL;

17. 아래 함수 결과로 가장 적절하지 <u>않은</u> 것은?
① UPPER('Sql Developer') : SQL DEVELOPER
② SUBSTR('Sql Developer', -5, 2) : lo
③ TO_CHAR(1000,'9,999') : 1,000
④ RTRIM('ABCAA', 'A') : BC

18. 다음 함수의 결과가 다른 것은?

COMM
0
NULL
100

① NVL(COMM, 100)
② NVL2(COMM, COMM, 100)
③ NULLIF(COMM, 100)
④ ISNULL(COMM, 100)

19. 다음 결과를 출력하는 SQL 문으로 가장 적절한 것은?

<TAB1>

COL1	COL2
A	10
B	10
C	10
D	20
E	20

<결과>

CNT10	CNT20
3	2

① SELECT SUM(DECODE(COL2, 10, 1, 0)) AS CNT10,
SUM(DECODE(COL2, 20, 1, 0)) AS CNT20
FROM TAB1;
② SELECT COUNT(DECODE(COL2, 10, 1, 0)) AS CNT10,
COUNT(DECODE(COL2, 20, 1, 0)) AS CNT20
FROM TAB1;
③ SELECT SUM(DECODE(COL2, 10, 'O', 'X')) AS CNT10,
SUM(DECODE(COL2, 20, 'O', 'X')) AS CNT20
FROM TAB1;
④ SELECT COUNT(DECODE(COL2, 10, 'O', 'X')) AS CNT10,
COUNT(DECODE(COL2, 20, 'O', 'X')) AS CNT20
FROM TAB1;

20. 아래 SQL 수행 결과로 가장 적절한 것은?

<EMP>

ENAME	SAL
SMITH	800
ALLEN	2000
KING	5000
BLAKE	3500
CLARK	1500

```
SELECT COUNT(ENAME) FROM EMP WHERE ENAME NOT LIKE '_L%E%';
```

① 1
② 2
③ 3
④ 4

21. 아래 SQL 중 출력 결과가 다른 것은?

①
```
SELECT *
  FROM EMP
 WHERE DEPTNO IN (10,20)
   AND JOB = 'CLERK'
    OR SAL > 3000;
```

②
```
SELECT *
  FROM EMP
 WHERE DEPTNO = 10 OR DEPTNO = 20
   AND JOB = 'CLERK'
    OR SAL > 3000;
```

③
```
SELECT *
  FROM EMP
 WHERE (DEPTNO = 10 AND JOB = 'CLERK')
    OR (DEPTNO = 20 AND JOB = 'CLERK')
    OR SAL > 3000;
```

④
```
SELECT *
  FROM EMP
 WHERE (DEPTNO = 10 OR DEPTNO = 20)
   AND JOB = 'CLERK'
  OR SAL > 3000;
```

22. 다음 GROUP BY에 대한 설명 중 가장 적절하지 않은 것은?

① GROUP BY 절에는 컬럼 별칭을 사용할 수 없다.
② GROUP BY 뒤의 컬럼 순서에 따라 출력되는 그룹의 수가 달라진다.
③ GROUP BY 뒤의 첫 번째 값이 UNIQUE한 경우 뒤에 어떤 컬럼을 추가적으로 명시해도 그룹의 수는 변화없다.
④ GROUP BY 절에 사용하지 않은 컬럼은 SUM과 같은 집계함수와 함께 SELECT절에 사용 가능하다.

23. ORDER BY에 대한 설명으로 가장 적절한 것은?

① ORDER BY 절을 사용하지 않으면 입력된 데이터의 순서대로 출력된다.

② ORDER BY 절의 기본 정렬 순서는 내림차순이다.

③ ORDER BY 절에는 컬럼 별칭을 사용할 수 없다.

④ ORDER BY 절에는 컬럼명과 숫자를 동시에 사용할 수 없다.

24. 다음 출력 결과를 얻기 위한 SQL 문장으로 가장 적절한 것은?

<EMP>

EMPNO	ENAME	MGR	DEPTNO
7369	SMITH	7566	20
7499	ALLEN	7654	30
7521	WARD	7566	30
7566	JONES	7654	20
7654	MARTIN		30

<결과>

EMPNO	ENAME	MANAGER_NAME
7369	SMITH	JONES
7499	ALLEN	MARTIN
7521	WARD	JONES
7566	JONES	MARTIN
7654	MARTIN	

① SELECT E1.EMPNO, E1.ENAME, E2.ENAME AS MANAGER_NAME
FROM EMP E1, EMP E2
WHERE E1.MGR = E2.EMPNO
ORDER BY E1.EMPNO;

② SELECT E1.EMPNO, E1.ENAME, E2.ENAME AS MANAGER_NAME
FROM EMP E1, EMP E2
WHERE E1.MGR = E2.EMPNO(+)
ORDER BY E1.EMPNO;

③ SELECT E1.EMPNO, E1.ENAME, E2.ENAME AS MANAGER_NAME
FROM EMP E1, EMP E2
WHERE E1.EMPNO = E2.MGR
ORDER BY E1.EMPNO;

④ SELECT E1.EMPNO, E1.ENAME, E2.ENAME AS MANAGER_NAME
FROM EMP E1, EMP E2
WHERE E1.EMPNO = E2.MGR(+)
ORDER BY E1.EMPNO;

25. 다음 두 테이블의 NATURAL JOIN을 수행한 결과의 출력 건수는?

<TAB1>

COL1	COL2
A	10
B	20
C	30
D	10
E	NULL

<TAB2>

COL2	COL3
10	1
20	2
20	3
NULL	4

① 2
② 3
③ 4
④ 5

26. 다음 ANSI 문법을 오라클 문법으로 바꾼 것으로 가장 적절한 것은?

```
SELECT *
  FROM TAB1 JOIN TAB2
    ON TAB1.COL1 = TAB2.COL1
      LEFT OUTER JOIN TAB3
    ON TAB1.COL2 = TAB3.COL2;
```

①
```
SELECT *
  FROM TAB1, TAB2, TAB3
 WHERE TAB1.COL1 = TAB2.COL1
   AND TAB1.COL2 = TAB3.COL2;
```

②
```
SELECT *
  FROM TAB1, TAB2, TAB3
 WHERE TAB1.COL1 = TAB2.COL1(+)
   AND TAB1.COL2 = TAB3.COL2;
```

③
```
SELECT *
  FROM TAB1, TAB2, TAB3
 WHERE TAB1.COL1 = TAB2.COL1
   AND TAB1.COL2 = TAB3.COL2(+);
```

④
```
SELECT *
  FROM TAB1, TAB2, TAB3
 WHERE TAB1.COL1 = TAB2.COL1(+)
   AND TAB1.COL2 = TAB3.COL2(+);
```

27. 학생과 교수 테이블을 사용하여 지도교수가 없는 학생 정보도 출력을 하고자 할 때, 아래 빈칸으로 가장 적절한 것은?

```
SELECT 학생.이름 학생명, 교수.이름 교수명
  FROM 학생 _________ 교수
     ON 학생.교수번호 = 교수.교수번호;
```

① LEFT OUTER JOIN
② RIGHT OUTER JOIN
③ FULL OUTER JOIN
④ INNER JOIN

28. 5개의 테이블 조인 시 조인 조건의 최소 개수는?

① 1
② 2
③ 3
④ 4

29. 다음 수행 결과를 차례대로 나열한 것은?

<TAB1>

COL1	COL2	COL3
NULL	1	1
1	1	NULL
1	NULL	1

```
SELECT COUNT(*) C1, COUNT(COL1) C2, SUM(COL2) C3 , SUM(COL1+COL2+COL3) C4
  FROM TAB1;
```

① 0, 2, 2, 0
② 3, 2, 2, 0
③ 3, 2, 2, NULL
④ 3, 2, NULL, NULL

30. 다음 중 순수 관계 연산자로 적절하지 <u>않은</u> 것은?

① SELECT
② DIVISION
③ JOIN
④ PRODUCT

31. 다음 SQL의 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
A	10
B	20
C	30
D	40

<TAB2>

COL1	COL2
A	10
C	20
NULL	30

```
SELECT SUM(COL2)
  FROM TAB1
 WHERE COL1 NOT IN (SELECT COL1
                      FROM TAB2
                     WHERE COL1 IS NOT NULL);
```

① 0
② 40
③ 50
④ 60

32. 다음 SQL의 실행 결과로 가장 적절한 것은?

<TAB1>

COL1
10
20
30
40

<TAB2>

COL1
35
45

```
SELECT SUM(COL1)
  FROM TAB1
 WHERE COL1 <= ALL(SELECT COL1
                     FROM TAB2);
```

① 10
② 30
③ 60
④ 100

33. 서브쿼리에 대한 설명 중 가장 적절하지 <u>않은</u> 것은?

① < ALL(10, 20, 30)은 10보다 작다와 같다.
② 다중행 서브쿼리는 동등비교, 대소비교 연산자를 사용할 수 없다.
③ 스칼라 서브쿼리는 반드시 각 행마다 단 하나의 행이 리턴되어야 한다.
④ 다중컬럼 서브쿼리의 SELECT절에 사용된 컬럼은 메인쿼리 SELECT절에 전달할 수 있다.

34. 다음 결과를 얻기 위한 아래 빈칸으로 가장 적절한 것은?

<EMP>

EMPNO	ENAME	MGR
0001	SMITH	
0002	ALLEN	0001
0003	FORD	0002
0004	SCOTT	0003

<결과>

사원명	상위관리자명
SMITH	SMITH
ALLEN	SMITH
FORD	ALLEN
SCOTT	FORD

```
SELECT ENAME 사원명,
       _____ 상위관리자명
  FROM EMP E1;
```

① (SELECT ENAME FROM EMP E2)

② (SELECT NVL(E2.ENAME,E1.ENAME) FROM EMP E2)

③ NVL((SELECT E2.ENAME FROM EMP E2 WHERE E1.EMPNO = E2.MGR), E1.ENAME)

④ NVL((SELECT E2.ENAME FROM EMP E2 WHERE E1.MGR = E2.EMPNO), E1.ENAME)

35. 아래 쿼리와 중 결과가 <u>다른</u> 하나는?

<TAB1>

COL1	COL2
A	10
B	20
C	30
D	40
E	NULL

<TAB2>

COL1	COL2
A	10
E	NULL

① SELECT * FROM TAB1 MINUS SELECT * FROM TAB2;

②
```
SELECT *
  FROM TAB1, TAB2
WHERE TAB1.COL1 != TAB2.COL1;
```

③
```
SELECT *
  FROM TAB1
WHERE TAB1.COL1 NOT IN (SELECT COL1
                        FROM TAB2);
```

④ SELECT *
FROM TAB1
WHERE NOT EXISTS (SELECT COL1
FROM TAB2
WHERE TAB1.COL1 = TAB2.COL1);

36. 아래 쿼리 결과와 같은 결과를 갖는 빈칸에 들어갈 문장으로 가장 적절한 것은?

```
SELECT DEPTNO, NULL AS JOB, SUM(SAL) AS SUM_SAL
  FROM EMP
 GROUP BY DEPTNO
 UNION ALL
SELECT DEPTNO, JOB, SUM(SAL) AS SUM_SAL
  FROM EMP
 GROUP BY DEPTNO, JOB
 UNION ALL
SELECT NULL, NULL, SUM(SAL) AS SUM_SAL
  FROM EMP;

SELECT DEPTNO, JOB, SUM(SAL) AS SUM_SAL
  FROM EMP
 GROUP BY ____________________;
```

① ROLLUP(DEPTNO, JOB)
② GROUPING SETS(DEPTNO, JOB)
③ GROUPING SETS(DEPTNO, JOB, NULL)
④ CUBE(DEPTNO, JOB)

37. 다음 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
A	10
A	20
A	20
B	30
B	40
B	40

```
SELECT SUM(COL2) OVER(PARTITION BY COL1 ORDER BY COL2 ASC) AS RESULT
  FROM TAB1;
```

①

RESULT
10
30
50
80
120
160

②

RESULT
10
30
50
30
70
110

③

RESULT
10
50
50
30
110
110

④

RESULT
10
50
70
80
110
150

38. 다음 수행 결과로 가장 적절한 것은?

<EMP>

EMPNO	ENAME	DEPTNO	SAL
7934	MILLER	10	2000
7782	CLARK	10	2000
7839	KING	10	5000
7369	SMITH	20	800
7876	ADAMS	20	2500
7566	JONES	20	2500
7788	SCOTT	20	3000
7902	FORD	20	3000

```
SELECT SUM(SAL)
  FROM (SELECT ENAME, DEPTNO, SAL,
               RANK() OVER(PARTITION BY DEPTNO ORDER BY SAL DESC) AS RANK_VALUE
          FROM EMP)
 WHERE RANK_VALUE = 2;
```

① 2000
② 4000
③ 9000
④ 10000

39. 다음 SQL 결과로 가장 적절한 것은?

<EXAM>

이름	성적
홍길동	98
박길동	80
최길동	80
김길동	75
이길동	75
황길동	70

```
SELECT TOP(3) WITH TIES 이름, 성적
  FROM EXAM
 ORDER BY 성적 DESC;
```

①

이름	성적
홍길동	98
박길동	80

②

이름	성적
홍길동	98
박길동	80
최길동	80

③

이름	성적
홍길동	98
박길동	80
최길동	80
김길동	75

④

이름	성적
홍길동	98
박길동	80
최길동	80
김길동	75
이길동	75

40. 다음 SQL의 수행 결과로 가장 적절한 것은?

<EMP>

EMPNO	ENAME	MGR	SAL
7839	KING		5000
7369	SMITH	7839	800
7499	ALLEN	7369	1600
7521	WARD	7499	1250
7566	JONES	7499	2975
7654	MARTIN	7839	1250

```
SELECT SUM(E2.SAL)
  FROM EMP E1 INNER JOIN EMP E2
    ON E1.MGR = E2.EMPNO
   AND E1.SAL > E2.SAL;
```

① 1600
② 1975
③ 2400
④ 4575

41. 계층형 질의에 대한 설명 중 가장 적절하지 <u>않은</u> 것은?
① 행끼리 연결관계가 있을 경우 연결관계를 표현하는 기법이다.
② 행의 연결관계는 CONNECT BY에 전달한다.
③ LEVEL은 항상 가장 먼저 시작하는 행이 1을 갖는다.
④ 같은 레벨일 경우 추가 정렬을 원한다면 ORDER BY에 컬럼명을 나열하면 된다.

42. 다음 출력 결과를 얻기 위한 SQL로 가장 적절하지 <u>않은</u> 것은?

<CUSTOMERS>

CUSTOMER_ID	CUST_FIRST_NAME	CUST_JOB_NAME
C26306	Kathy	농업인
C31762	Kathy	농업인
C15506	Sachin	종합소매업
C32830	Farrah	금융업
C32291	Harry Mean	금융업

<결과>

CNT1	CNT2	CNT3
2	2	1

①
```
SELECT "'금융업'" AS CNT1, "'농업인'" AS CNT2, "'종합소매업'" AS CNT3
  FROM (SELECT CUSTOMER_ID, CUST_FIRST_NAME, CUST_JOB_NAME FROM CUSTOMERS)
 PIVOT (COUNT(CUST_FIRST_NAME) FOR CUST_JOB_NAME IN ('금융업','농업인','종합소매업'));
```

② SELECT "'금융업'" AS CNT1, "'농업인'" AS CNT2, "'종합소매업'" AS CNT3
FROM (SELECT CUST_JOB_NAME FROM CUSTOMERS)
PIVOT (COUNT(CUST_JOB_NAME) FOR CUST_JOB_NAME IN ('금융업','농업인','종합소매업'));

③ SELECT "'금융업'" AS CNT1, "'농업인'" AS CNT2, "'종합소매업'" AS CNT3
FROM (SELECT CUSTOMER_ID, CUST_JOB_NAME FROM CUSTOMERS)
PIVOT (COUNT(CUSTOMER_ID) FOR CUST_JOB_NAME IN ('금융업','농업인','종합소매업'));

④ SELECT "'금융업'" AS CNT1, "'농업인'" AS CNT2, "'종합소매업'" AS CNT3
FROM (SELECT CUST_FIRST_NAME, CUST_JOB_NAME FROM CUSTOMERS)
PIVOT (COUNT(CUST_FIRST_NAME) FOR CUST_JOB_NAME IN ('금융업','농업인','종합소매업'));

43. 다음 SQL 실행 결과로 가장 적절한 것은?

```
SELECT REGEXP_REPLACE('ABCCCAC BC BBBC', 'AB|C+') FROM DUAL;
```

① AC B BBB
② A B BBB
③ CCC BB
④ AC BC BBBC

44. 이메일 주소에서 이메일 아이디만 추출하기 위한 빈칸 표현으로 가장 적절하지 <u>않은</u> 것은?

```
SELECT REGEXP_SUBSTR('HDATALAB_1@NAVER.COM', '__________') FROM DUAL;
```

① (\w|_)+
② [A-z_0-9]+
③ ([[:alnum:]]|_)+
④ \w|\d+

45. 아래 SQL 수행 결과로 가장 적절한 것은?

<TAB1>

NO	NAME	PRICE
1	아메리카노	1000
2	카페라떼	2000

<TAB2>

NO	NAME	PRICE
1	아메리카노	1500
2	카페라떼	2500
3	카페모카	3000

```
MERGE INTO TAB1
USING TAB2
    ON (TAB1.NO = TAB2.NO)
 WHEN MATCHED THEN
        UPDATE
            SET TAB1.PRICE = TAB2.PRICE
        DELETE
            WHERE TAB1.PRICE <= 1000
 WHEN NOT MATCHED THEN
        INSERT VALUES(TAB2.NO, TAB2.NAME, TAB2.PRICE);

SELECT SUM(PRICE) FROM TAB1;
```

① 3000
② 5000
③ 5500
④ 7000

46. 아래 문장들을 순서대로 수행했을 경우 최종 데이터 형태로 가장 적절한 것은?
(단, 에러가 나는 문장은 무시한다.)

```
CREATE TABLE TAB1(NO NUMBER, NAME VARCHAR2(10));
INSERT INTO TAB1 VALUES(1, 'A');
INSERT INTO TAB1 VALUES(2, 'B');
COMMIT;
UPDATE TAB1 SET NAME = 'C' WHERE NO = 2;
SAVEPOINT SAVE1;
INSERT INTO TAB1 VALUES(3, 'C');
COMMIT;
DELETE FROM TAB1 WHERE NO = 1;
ROLLBACK TO SAVE1;
ROLLBACK;
```

①

NO	NAME
2	C

②

NO	NAME
2	C
3	C

③

NO	NAME
1	A
2	B
3	C

④

NO	NAME
1	A
2	C
3	C

47. 다음 설명 중 가장 적절하지 <u>않은</u> 것은?

① TRUNCATE는 데이터 삭제 시 즉시 반영되는 DML이다.
② DELETE는 COMMIT을 하기 전까지는 삭제된 데이터를 삭제 전으로 다시 되돌릴 수 있다.
③ DROP은 테이블의 구조를 삭제하기 때문에 삭제된 이후에 테이블 조회가 불가능하다.
④ TRUNCATE는 일부 데이터만을 삭제할 수 없다.

48. 다음 제약조건 생성 명령어로 가장 적절하지 <u>않은</u> 것은?

① CREATE TABLE TAB1(NO NUMBER PRIMARY KEY, NAME VARCHAR2(10) NOT NULL);
② CREATE TABLE TAB1(NO NUMBER, NAME VARCHAR2(10), JDATE DATE,
PRIMARY KEY(NO, NAME));
③ CREATE TABLE TAB1(NO NUMBER, NAME VARCHAR2(10) NOT NULL);
ALTER TABLE TAB1 ADD PRIMARY KEY(NO);
④ CREATE TABLE TAB1(NO NUMBER PRIMARY KEY, NAME VARCHAR2(10) NOT NULL);
ALTER TABLE TAB1 ADD JDATE DATE NOT NULL DEFAULT SYSDATE ;

49. 테이블 구조 변경에 대한 설명 중 가장 적절하지 <u>않은</u> 것은?

① 컬럼을 동시에 여러 개 추가 시 반드시 괄호로 묶어 전달해야 한다.
② 데이터의 존재여부와 상관없이 컬럼 삭제는 항상 가능하다.
③ 데이터가 존재할 경우 데이터 유형 수정은 항상 불가하다.
④ 기본키 생성 시 기본키 컬럼에 인덱스가 자동 생성된다.

50. 뷰에 대한 설명 중 가장 적절한 것은?

① 뷰 생성 후 뷰에 컬럼 정의를 변경할 수 있다.
② 뷰를 통해 원본 테이블에 데이터를 변경할 수 없다.
③ 뷰에 직접 인덱스를 생성할 수 있다.
④ 서브쿼리나 조인을 사용한 복잡한 쿼리에 대한 뷰를 생성할 수 있다.

6회차

1. 다음은 어떤 데이터 모델링에 대한 설명인가?

개념적 모델링의 결과를 토대로 세부속성, 식별자, 관계 등을 표현하는 단계 데이터 구조를 정의하기 때문에 비슷한 업무나 프로젝트에서 동일한 형태의 데이터 사용 시 재사용 가능

① 논리적 데이터 모델링
② 물리적 데이터 모델링
③ 개괄적 데이터 모델링
④ 개념적 데이터 모델링

2. 다음 중 중심엔터티로 가장 적절하지 <u>않은</u> 것은?
① 주문
② 청구
③ 계약
④ 고객

3. 다음 중 속성에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?
① 한 개의 엔터티는 2개 이상의 인스턴스 집합이어야 한다.
② 한 개의 엔터티는 1개 이상의 속성을 갖는다.
③ 한 개의 속성은 1개의 속성값을 갖는다.
④ 속성은 엔터티에 속한 자세하고 구체적인 값을 가져야 한다.

4. 다음 중 관계에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?
① 엔터티의 정의에 따라 관계가 변경되기도 한다.
② 존재관계와 행위관계는 ERD에 표현되지 않는다.
③ 한 엔터티의 레코드가 다른 엔터티의 레코드와 어떻게 연결되는지를 표현한 것이 관계이다.
④ 사원은 반드시 소속 부서가 있어야 한다면 이는 완전 1 대 1 관계를 갖는다.

5. 다음이 설명하는 KEY로 가장 적절한 것은?

유일성과 최소성을 만족하는 키들의 집합

① 기본키
② 후보키
③ 슈퍼키
④ 대체키

6. 정규화에 대한 설명으로 가장 적절하지 않은 것은?
① 최소한의 데이터만을 하나의 엔터티에 넣는식으로 데이터를 분해하는 과정을 말한다.
② 물리모델링 단계에서 진행한다.
③ 정규화를 하지 않아 발생하는 현상을 이상현상이라 한다.
④ 이행적 종속을 없애도록 테이블을 분리하는 것은 제 3 정규화이다.

7. 관계에 대한 설명으로 가장 적절하지 않은 것은?
① 엔터티 안에 인스턴스가 개별적으로 관계를 가지는 것을 관계의 페어링이라고 한다.
② 관계의 차수는 하나의 엔터티와 다른 엔터티 간의 레코드 연결 방식을 의미한다.
③ 고객이 여러 계좌를 소유할 수 있다면 두 엔터티는 M 대 N 관계를 갖는다.
④ 관계는 페어링의 집합을 의미한다.

8. 필수적, 선택적 관계에 대한 설명으로 가장 적절하지 않은 것은?
① 두 엔터티의 관계가 필수적일 때 하나의 트랜잭션을 형성한다.
② 두 엔터티가 서로 독립적 수행이 가능하다면 선택적 관계로 표현한다.
③ 바커표기법에서는 실선과 점선으로 필수적, 선택적 관계를 구분한다.
④ IE표기법에는 필수적, 선택적 관계를 구분하지 않는다.

9. NULL에 대한 설명으로 가장 적절하지 않은 것은?
① NULL은 0과 공백과는 다른 개념이다.
② NULL을 포함한 산술연산 결과는 항상 NULL이다.
③ IE표기법에서는 NULL 허용여부를 따로 표시하지 않는다.
④ NULL로만 구성된 집합의 COUNT 결과는 NULL이다.

10. 다음은 식별자 중 무엇을 설명하고 있는가?

엔터티 내에서 각 인스턴스를 구분할 수 있는 구분자지만, 대표성을 가지지 못해 참조 관계 연결을 할 수 없는 식별자

① 주식별자
② 보조식별자
③ 내부식별자
④ 본질식별자

11. 다음 중 테이블에 대한 설명으로 가장 적절한 것은?
① 테이블명은 숫자로 시작할 수 있다.
② 테이블명에는 모든 특수기호(!, _, -, ?등)를 사용할 수 없다.
③ 테이블 소유자를 명시하지 않으면 테이블을 생성하는 유저 소유가 된다.
④ 본인 소유의 테이블에 대해 SELECT, INSERT, UPDATE, DELETE 권한을 다른 사용자에게 임의로 권한을 부여할 수 없다.

12. 다음 중 SQL 분류로 적절하지 않은 것은?
① DDL - TRUNCATE ② DML - MERGE
③ DCL - GRANT ④ DCL - ROLLBACK

13. SELECT문에 대한 설명 중 가장 적절하지 않은 것은? (단, DBMS는 오라클)

① SELECT절에서는 *와 컬럼을 동시에 사용할 수 없다.
② DISTINCT는 단 한번만 사용 가능하다.
③ 컬럼 별칭은 항상 AS를 사용하여 정의 가능하다.
④ 테이블 별칭은 AS를 사용하지 않는다.

14. 다음 날짜 함수 결과로 적절하지 않은 것은?

단, 오늘 날짜는 2024년 11월 17이다.

① TO_DATE('12','MM') : '2024/12/01'
② TO_DATE('2024/12','YYYY/MM') : '2024/12/01'
③ TO_DATE('2024','YYYY') : '2024/01/01'
④ TO_DATE('12','DD') : '2024/11/12'

15. 다음 중 함수 수행 결과가 다른 것은?

COL1	COL2
10	10
20	10
NULL	20

① NVL(COL1, 20)
② COALESCE(COL1, COL2)
③ ISNULL(COL1, 20)
④ DECODE(COL1, NULL, 20)

16. 다음 함수의 결과로 적절하지 않은 것은?

① SIGN(-100) : -1
② MOD(7, 3) : 1
③ CEIL(-2.7) : -3
④ FLOOR(2.5) : 2

17. 다음 함수의 결과로 적절하지 않은 것은?

① SUBSTR('SQL SERVER TOOL', 4, 5) : ' S'
② RPAD('', 4, '*') : NULL
③ RTRIM('SQL SERVER TOOL', 'L') : 'SQL SERVER TOO'
④ LENGTH(REPLACE('SQL SERVER TOOL', 'TOOL')) : 11

18. 다음 SQL 중 실행이 불가능 한 구간을 고르시오.

① SELECT COL1 C1, SUM(COL3) AS SUM
② FROM TABLE1 T1
③ WHERE SUM(COL1) >= 100
④ GROUP BY COL1, COL2;

19. 아래 SQL의 실행 결과로 알맞은 것은?

<TAB1>

COL1	COL2	COL3
10	1	NULL
10	2	10
NULL	1	20
0	3	30
20	3	30

```
SELECT SUM(COL1 + COL2)
  FROM TAB1
 WHERE COL3 != 30;
```

① NULL
② 12
③ 13
④ 24

20. 다음 문장 수행 결과로 적절한 것은?

<TAB1>

COL1	COL2
A	NULL
A	10
B	20
C	30
C	NULL
C	20
NULL	10
NULL	30

```
SELECT COL1, SUM(COL2) AS CSUM
  FROM TAB1
 GROUP BY COL1
HAVING COUNT(COL2) = 2;
```

①

COL1	CSUM
A	10

②

COL1	CSUM
C	50

③

COL1	CSUM
C	50
NULL	40

④

COL1	CSUM
A	10
C	50
NULL	30

21. 다음 문장 중 실행이 가능한 것은?

①
```
SELECT T.COL1, T.COL2, SUM(T.COL3) TOTAL
  FROM TAB1 T
 WHERE T.COL1 != 10
 GROUP BY T.COL2;
```
②
```
SELECT COL1, COL2, SUM(COL3) TOTAL
  FROM TAB1
 GROUP BY COL1, COL2
HAVING SUM(COL3) > 100;
```
③
```
SELECT COL1, SUM(COL2) TOTAL
  FROM TAB1
 GROUP BY COL1
 ORDER BY COL2;
```
④
```
SELECT COL1, COL2, SUM(COL2) TOTAL
  FROM TAB1
 WHERE TOTAL > 100
 GROUP BY COL1, COL2, COL3
 ORDER BY TOTAL;
```

22. 아래 SQL 수행 결과로 가장 적절한 것은?

<TAB1>

TEL
234-4567
1229-9384
567-8839
2569-4856

```
SELECT SUBSTR(TEL, 1, INSTR(TEL,'-')-1) AS NO
  FROM TAB1
 ORDER BY NO;
```

①

NO
234
1229
567
2569

②

NO
234
567
1229
2569

③

NO
1229
234
2569
567

④

NO
1229-
234-
2569-
567-

23. 아래 SQL 수행 결과로 가장 적절한 것은?

<상품>

코드	후기
100	50
101	0
102	10
103	45
104	60
105	120
106	55

```
SELECT CASE WHEN 후기 < 20 THEN '< 20'
            WHEN 후기 < 100 THEN '< 100'
             ELSE '>= 100'
       END AS 후기수,
       COUNT(후기) AS 상품수
  FROM 상품
 GROUP BY CASE WHEN 후기 < 20 THEN '< 20'
               WHEN 후기 < 100 THEN '< 100'
                ELSE '>= 100'
          END,
          CASE WHEN 후기 < 20 THEN 1
               WHEN 후기 < 100 THEN 2
                ELSE 3
          END
 ORDER BY CASE WHEN 후기 < 20 THEN 1
               WHEN 후기 < 100 THEN 2
               ELSE 3
          END;
```

①

후기수	상품수
< 20	2
< 100	4
>= 100	1

②

후기수	상품수
< 100	4
< 20	2
>= 100	1

③

후기수	상품수
>= 100	1
< 100	4
< 20	2

④

후기수	상품수
>= 100	1
< 20	2
< 100	4

24. 아래 SQL 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
A	10
B	20
NULL	30
D	20

<TAB2>

COL1	COL2
A	20
B	20
C	10
NULL	30

```
SELECT COUNT(*)
  FROM TAB1 CROSS JOIN TAB2;
```

① 1
② 2
③ 9
④ 16

25. 아래 SQL 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
A	10
B	20
NULL	30
D	20

<TAB2>

COL1	COL2
A	20
B	20
C	10
NULL	30

```
SELECT SUM(TAB2.COL2)
  FROM TAB1 LEFT OUTER JOIN TAB2
 USING (COL1);
```

① NULL
② 40
③ 50
④ 80

26. 아래 결과를 얻기 위한 SQL로 가장 적절한 것은?

<고객>

고객번호	이름	포인트
100	홍길동	80
101	박길동	110
102	최길동	150
103	김길동	500
104	안길동	900

<상품>

상품번호	상품명	최소포인트	최대포인트
1000	양말	101	200
1001	세탁세제	201	300
1002	식기세트	301	400
1003	전기담요	401	500
1004	세탁기	501	1000

<결과>

이름	상품명
홍길동	NULL
박길동	양말
최길동	양말
김길동	전기담요
안길동	세탁기

①
```
SELECT 이름, 상품명
  FROM 고객, 상품
 WHERE 포인트 >= 최소포인트;
```
②
```
SELECT 이름, 상품명
  FROM 고객, 상품
 WHERE 포인트 BETWEEN 최소포인트 AND 최대포인트;
```
③
```
SELECT 이름, 상품명
  FROM 고객, 상품
 WHERE 포인트 >= 최소포인트(+);
```
④
```
SELECT 이름, 상품명
  FROM 고객, 상품
 WHERE 포인트 BETWEEN 최소포인트(+) AND 최대포인트(+);
```

27. 조인에 대한 설명으로 가장 적절한 것은?

① NATURAL JOIN 시 ON절을 사용할 수 없다.

② FULL OUTER JOIN은 LEFT OUTER JOIN과 RIGHT OUTER JOIN 결과의 합집합(UNION ALL) 결과와 같다.

③ USING절에는 괄호를 생략할 수 있다.

④ ORACLE에서는 FROM절의 테이블 순서에 따라 LEFT OUTER JOIN 결과가 달라진다.

28. 아래 SQL 수행 결과로 알맞은 것은?

<CAR>

CAR_ID	MODEL	RENT_FEE	CAR_TYPE
1	소나타	60000	SEDAN
2	K5	65000	SEDAN
3	셀토스	80000	SUV
4	K9	220000	LUXURY
5	EV6	95000	ELECTRIC

<CAR_RENTAL_HISTORY>

RENTAL_ID	CAR_ID	START_DATE	END_DATE
1000	1	2024/01/02 10:30:00	2024/01/05 14:45:00
1001	2	2024/07/12 08:00:00	2024/07/18 11:35:00
1002	4	2024/07/25 09:15:00	2024/07/27 10:45:00
1003	5	2024/02/15 08:45:00	2024/02/20 11:20:00
1004	1	2024/07/05 14:00:00	2024/07/10 16:50:00

```
SELECT COUNT(RENTAL_ID) AS RESULT
  FROM CAR_RENTAL_HISTORY
 WHERE CAR_ID IN (SELECT CAR_ID
                    FROM CAR
                   WHERE CAR_TYPE = 'SEDAN')
   AND TO_CHAR(START_DATE, 'YYYYMM') = '202407';
```

① 0
② 1
③ 2
④ 3

29. 다음 SQL의 수행 결과로 가장 적절한 것은?

<TAB1>

COL1	COL2
A	10
B	20
C	30
E	50
NULL	40

<TAB2>

COL1	COL2
A	10
C	20
D	30
NULL	40

```
SELECT SUM(COL2)
  FROM TAB1
 WHERE NOT EXISTS (SELECT COL1
                     FROM TAB2
                    WHERE TAB1.COL1 = TAB2.COL1);
```

① 40
② 70
③ 80
④ 110

30. 아래 SQL 수행 결과로 알맞은 것은?

<TAB1>

COL1
AB
ABC
BC
BCD
CAB

```
SELECT COUNT(COL1) AS C1
  FROM TAB1
 WHERE COL1 LIKE (SELECT '%'||MIN(COL1)||'%'
                    FROM TAB1);
```

① 0
② 1
③ 2
④ 3

31. 다음 중 항상 오류가 발생하는 것은?

①
```
SELECT *
  FROM TAB1
 WHERE (COL1, COL2) = (SELECT COL1, MAX(COL2)
                         FROM TAB2
                        GROUP BY COL1
                       HAVING MAX(COL2) > 100);
```
②
```
SELECT COL1, (SELECT COL3
                FROM TAB2
               WHERE TAB1.COL2 = TAB2.COL2)
  FROM TAB1;
```
③
```
SELECT TAB1.COL1, TAB1.COL2, I.AVGSAL
  FROM TAB1, (SELECT COL1, AVG(COL2) AS AVGSAL
                FROM TAB1
               GROUP BY COL1) I
 WHERE TAB1.COL1 = I.COL1
   AND TAB1.COL2 > I.AVGSAL;
```

④

```
SELECT T1.COL1, T2.AVGSAL
  FROM TAB1 T1
 WHERE COL1 < (SELECT AVG(COL1) AS AVGVAL
                 FROM TAB1 T2
                WHERE COL2 = T1.COL2);
```

32. 아래 SQL 수행 결과로 알맞은 것은?

<EMP>

EMPNO	ENAME	COMM	MGR
1000	SMITH	200	1001
1001	ALLEN	100	1002
1002	FORD	200	1003
1003	WARD	300	1004
1004	KING	200	NULL

```
UPDATE EMP E1
   SET COMM = (SELECT (E1.COMM + E2.COMM)/2
                 FROM EMP E2
                WHERE E1.MGR = E2.EMPNO
                  AND E1.COMM > E2.COMM);

SELECT SUM(COMM)
  FROM EMP;
```

① 300
② 400
③ 800
④ NULL

33. 서브쿼리에 대한 설명으로 가장 적절하지 <u>않은</u> 것은?
① 연관 서브쿼리는 서브쿼리에 메인 테이블의 컬럼이 존재하는 경우이다.
② 다중 컬럼 서브쿼리는 모든 비교연산에 대한 처리가 가능하다.
③ 단일 행 서브쿼리는 대소비교 연산자의 전달이 가능하다.
④ > ALL 연산자는 서브쿼리 내 최댓값보다 큰 조건을 완성한다.

34. 아래의 SQL 의 출력 결과 중 알맞은 것은?

<TAB1>

COL1	COL2
A	25
B	20
C	40
D	10

<TAB2>

COL1	COL2
A	30
A	20
B	20
B	10
C	30
C	40
D	10
D	20

```
SELECT SUM(COL2)
  FROM TAB1
 WHERE COL2 < ANY(SELECT COL2
                    FROM TAB2
                   WHERE TAB1.COL1 = TAB2.COL1);
```

① 25
② 35
③ 55
④ 95

35. 다음 중 수행 불가능한 SQL은?

<TAB1>

COL1	NUMBER(10)
COL2	NUMBER(6)

<TAB2>

COL1	NUMBER(10)
COL2	NUMBER(6)

①
```
SELECT COL2, COL1
  FROM TAB1
 UNION ALL
SELECT COL1, COL2
  FROM TAB2;
```
②
```
SELECT COL1, COL2
  FROM TAB1
 UNION
SELECT COL1, COL2
  FROM TAB2
 ORDER BY COL1;
```
③
```
(SELECT COL1, COL2
  FROM TAB1
 UNION
SELECT COL1, COL2
  FROM TAB2)
ORDER BY COL1;
```
④
```
SELECT COL1, COL2
  FROM TAB1
 ORDER BY COL1
 UNION ALL
SELECT COL1, COL2
  FROM TAB2
 ORDER BY COL1;
```

36. 아래 결과를 얻기 위한 빈칸에 들어갈 문장으로 가장 적절한 것은?

<EMP>

ENAME	JOB	DNAME	SAL
CLARK	MANAGER	ACCOUNTING	2450
KING	PRESIDENT	ACCOUNTING	5000
MILLER	CLERK	ACCOUNTING	1300
JONES	MANAGER	RESEARCH	2975
FORD	ANALYST	RESEARCH	3000
ADAMS	CLERK	RESEARCH	1100
SMITH	CLERK	RESEARCH	800
SCOTT	ANALYST	RESEARCH	3000
WARD	SALESMAN	SALES	1250
TURNER	SALESMAN	SALES	1500
ALLEN	SALESMAN	SALES	1600
JAMES	CLERK	SALES	950
BLAKE	MANAGER	SALES	2850
MARTIN	SALESMAN	SALES	1250

<결과>

DNAME	JOB	SUMSAL
ACCOUNTING	CLERK	1300
ACCOUNTING	MANAGER	2450
ACCOUNTING	PRESIDENT	5000
ACCOUNTING		8750
RESEARCH	ANALYST	6000
RESEARCH	CLERK	1900
RESEARCH	MANAGER	2975
RESEARCH		10875
SALES	CLERK	950
SALES	MANAGER	2850
SALES	SALESMAN	5600
SALES		9400
	ANALYST	6000
	CLERK	4150
	MANAGER	8275
	PRESIDENT	5000
	SALESMAN	5600
		29025

```
SELECT DNAME, JOB, SUM(SAL) AS SUMSAL
  FROM EMP
 GROUP BY _____
 ORDER BY 1, 2;
```

① CUBE(DNAME, JOB)
② CUBE(DNAME, JOB, NULL)
③ ROLLUP(DNAME,JOB)
④ GROUPING SETS(DNAME,JOB, NULL)

37. 다음 SQL 실행 결과로 가장 알맞은 것은?

<실적>

지점	연도	월	실적금액
A	2024	01	2000
A	2024	02	1000
A	2024	03	2000
A	2024	04	3000
B	2024	01	2000
B	2024	02	4000
B	2024	03	2000

```
SELECT SUM(실적금액)
  FROM (SELECT 실적.*,
               LAG(실적금액) OVER(PARTITION BY 지점
                                  ORDER BY 월) AS VAL
          FROM 실적)
 WHERE 실적금액 < VAL;
```

① 3000
② 5000
③ 7000
④ 9000

38. 다음 SQL 실행 결과로 가장 알맞은 것은?

<직원>

사번	이름	부서번호	급여
1000	SMITH	01	5000
1001	ALLEN	01	5000
1002	SCOTT	01	3000
1003	CLARK	01	2000
1004	BLAKE	02	4000
1005	JAMES	02	2000
1006	ADAMS	02	2000

```
SELECT RANK() OVER(PARTITION BY 부서번호 ORDER BY 급여 DESC) AS 순위1,
       DENSE_RANK() OVER(PARTITION BY 부서번호 ORDER BY 급여 DESC) AS 순위2,
       ROW_NUMBER() OVER(PARTITION BY 부서번호 ORDER BY 급여 DESC) AS 순위3
  FROM 직원;
```

①

순위1	순위2	순위3
1	1	1
1	1	2
2	3	3
3	4	4
1	1	1
2	2	2
2	2	3

②

순위1	순위2	순위3
1	1	1
2	1	2
3	2	3
4	3	4
1	1	1
2	2	2
3	2	3

③

순위1	순위2	순위3
1	1	1
1	1	1
3	2	2
4	3	3
1	1	1
2	2	2
2	2	3

④

순위1	순위2	순위3
1	1	1
1	1	2
3	2	3
4	3	4
1	1	1
2	2	2
2	2	3

39. 다음 SQL 실행 결과로 가장 알맞은 것은?

<포인트>

고객번호	포인트	날짜
1000	500	2024/01/01 14:34:00
1000	300	2024/01/03 19:30:00
1000	300	2024/01/05 09:50:10
1000	100	2024/01/07 11:10:00
1001	400	2024/01/09 14:00:00
1001	200	2024/01/10 15:30:00
1001	100	2024/01/11 16:34:00

```
SELECT p.*, SUM(포인트) OVER(PARTITION BY 고객번호
                             ORDER BY 날짜
                             ROWS BETWEEN 1 PRECEDING AND CURRENT ROW ) as 누적포인트
  FROM 포인트 p
```

①

고객번호	포인트	날짜	누적포인트
1000	500	2024/01/01 14:34:00	500
1000	300	2024/01/03 19:30:00	300
1000	300	2024/01/05 09:50:10	300
1000	100	2024/01/07 11:10:00	100
1001	400	2024/01/09 14:00:00	400
1001	200	2024/01/10 15:30:00	200
1001	100	2024/01/11 16:34:00	100

②

고객번호	포인트	날짜	누적포인트
1000	500	2024/01/01 14:34:00	500
1000	300	2024/01/03 19:30:00	800
1000	300	2024/01/05 09:50:10	600
1000	100	2024/01/07 11:10:00	400
1001	400	2024/01/09 14:00:00	400
1001	200	2024/01/10 15:30:00	600
1001	100	2024/01/11 16:34:00	300

③

고객번호	포인트	날짜	누적포인트
1000	500	2024/01/01 14:34:00	500
1000	300	2024/01/03 19:30:00	800
1000	300	2024/01/05 09:50:10	1100
1000	100	2024/01/07 11:10:00	1200
1001	400	2024/01/09 14:00:00	400
1001	200	2024/01/10 15:30:00	600
1001	100	2024/01/11 16:34:00	700

④

고객번호	포인트	날짜	누적포인트
1000	500	2024/01/01 14:34:00	500
1000	300	2024/01/03 19:30:00	1100
1000	300	2024/01/05 09:50:10	1100
1000	100	2024/01/07 11:10:00	1200
1001	400	2024/01/09 14:00:00	400
1001	200	2024/01/10 15:30:00	600
1001	100	2024/01/11 16:34:00	700

40. 다음 결과를 얻기 위한 SQL 결과로 가장 적절한 것은?

<EXAM>

이름	성적
홍길동	98
김길동	80
이길동	80
최길동	72
박길동	60

<결과>

이름	성적
홍길동	98
김길동	80

①

```
SELECT TOP(2) 이름, 성적
  FROM EXAM
 ORDER BY 성적 DESC;
```

②

```
SELECT TOP(3) 이름, 성적
  FROM EXAM
 ORDER BY 성적 DESC;
```

③

```
SELECT TOP(2) WITH TIES 이름, 성적
  FROM EXAM
 ORDER BY 성적 DESC;
```

④

```
SELECT TOP(3) WITH TIES 이름, 성적
  FROM EXAM
 ORDER BY 성적 DESC;
```

41. 다음 수행 결과로 가장 적절한 것은?

<DEPARTMENT>

DEPTNO	DNAME	PART	BUILD
101	컴퓨터공학과	100	정보관
102	멀티미디어공학과	100	멀티미디어관
103	소프트웨어공학과	100	소프트웨어관
201	전자공학과	200	전자제어관
202	기계공학과	200	기계실험관
203	화학공학과	200	화학실습관
301	문헌정보학과	300	인문관
100	컴퓨터정보학부	10	
200	메카트로닉스학부	10	
300	인문사회학부	20	
10	공과대학		
20	인문대학		

```
SELECT LEVEL, DNAME
  FROM DEPARTMENT D
 START WITH DEPTNO = 10
CONNECT BY PRIOR D.DEPTNO = D.PART
 ORDER BY 1, 2;
```

①

LEVEL	DNAME
1	공과대학
2	컴퓨터정보학부
3	컴퓨터공학과
3	멀티미디어공학과
3	소프트웨어공학과
2	메카트로닉스학부
3	전자공학과
3	기계공학과
3	화학공학과
1	인문대학
2	인문사회학부
3	문헌정보학과

②

LEVEL	DNAME
1	공과대학
1	인문대학

③

1	공과대학
2	메카트로닉스학부
2	컴퓨터정보학부
3	기계공학과
3	멀티미디어공학과
3	소프트웨어공학과
3	전자공학과
3	컴퓨터공학과
3	화학공학과

④

LEVEL	DNAME
1	공과대학

42. 아래 결과를 얻기 위한 빈칸에 들어갈 문장으로 가장 적절한 것은?

<학생>

학번	이름	클래스	점수
1	홍길동	A	90
2	박길동	A	80
3	송길동	A	70
4	김길동	B	85
5	구길동	B	80
6	이길동	B	75

<결과>

학번	이름	클래스	점수	학번	이름	클래스	점수
1	홍길동	A	90				
2	박길동	A	80	1	홍길동	A	90
3	송길동	A	70	1	홍길동	A	90
3	송길동	A	70	2	박길동	A	80
4	김길동	B	85				
5	구길동	B	80	4	김길동	B	85
6	이길동	B	75	4	김길동	B	85
6	이길동	B	75	5	구길동	B	80

```
SELECT S1.*, S2.*
  FROM 학생 S1, 학생 S2
 WHERE _________________
 ORDER BY S1.학번, S2.학번;
```

① S1.점수 > S2.점수
② S1.점수 < S2.점수
③ S1.점수 > S2.점수(+) AND S1.클래스 = S2.클래스(+)
④ S1.점수 < S2.점수(+) AND S1.클래스 = S2.클래스(+)

43. 다음 출력 결과를 얻기 위한 SQL로 가장 적절한 것은?

<학생>

학생번호	이름	학년	성별	성적
1000	홍길동	4	남	90
1001	박길동	4	여	80
1002	최길동	3	남	90
1003	유길동	3	남	70
1004	신길동	3	여	90
1005	김길동	2	남	100
1006	간길동	2	남	60
1007	문길동	2	여	80
1008	송길동	1	여	60
1009	구길동	1	남	90

<결과>

	남	여
1	90	60
2	80	80
3	80	90
4	90	80

①
```
SELECT *
  FROM (SELECT 학년, 성별, 성적 FROM 학생)
  PIVOT (AVG(성적) FOR 학년 IN (1,2,3,4));
```
②
```
SELECT *
  FROM (SELECT 학년, 성별, 성적 FROM 학생)
UNPIVOT (AVG(성적) FOR 학년 IN (1,2,3,4));
```
③
```
SELECT *
  FROM (SELECT 학년, 성별, 성적 FROM 학생)
  PIVOT (AVG(성적) FOR 성별 IN ('남','여'));
```
④
```
SELECT *
  FROM (SELECT 학년, 성별, 성적 FROM 학생)
UNPIVOT (AVG(성적) FOR 성별 IN ('남','여'));
```

44. 다음 SQL 실행 결과로 가장 적절한 것은?

```
SELECT REGEXP_SUBSTR('ORA-00933: SQL command not properly ended', '[A-z]+', 1, 3) FROM DUAL;
```

① null
② ORA
③ SQL
④ command

45. 다음 SQL 실행 결과로 가장 적절한 것은?

<TAB1>

A12345A
12345BA
AA
B00A

```
SELECT COUNT(ID)
  FROM TAB1
 WHERE REGEXP_LIKE(ID, '^A|B.+A$');
```

① 0
② 1
③ 2
④ 3

46. 다음 UPDATE문장 중 수행이 불가능한 문장은? (단, DBMS는 오라클)

①
```
UPDATE EMP
   SET SAL = 5000, COMM = 300
 WHERE ENAME = 'SMITH';
```
②
```
UPDATE EMP
   SET (SAL, COMM) = (5000, 300)
 WHERE ENAME = 'SMITH';
```
③
```
UPDATE EMP
   SET (SAL, COMM) = (SELECT MAX(SAL), MAX(COMM)
                      FROM EMP)
 WHERE ENAME = 'SMITH';
```
④
```
UPDATE EMP E1
   SET SAL = (SELECT MAX(SAL)
              FROM EMP
             WHERE DEPTNO = E1.DEPTNO)
 WHERE ENAME = 'SMITH';
```

47. 다음의 문장 수행 결과로 알맞은 것은?

<TAB1>

COL1	COL2
A	10
A	20
B	30
B	40

```
UPDATE TAB1
    SET COL2 = 20
 WHERE COL1 = 'B';
COMMIT;
INSERT INTO TAB1 VALUES('C', 50);
SAVEPOINT AAA;
DELETE TAB1 WHERE COL1 = 'A';
INSERT INTO TAB1 VALUES('C', 60);
ROLLBACK TO AAA;
ROLLBACK;

SELECT SUM(COL2) FROM TAB1;
```

① 40
② 70
③ 120
④ 180

48. 다음 문장을 차례대로 수행한 결과로 알맞은 것은?

```
CREATE TABLE TAB1(COL1 NUMBER, COL2 VARCHAR2(10));
INSERT INTO TAB1 VALUES(1, 'A');
INSERT INTO TAB1 VALUES(2, 'B');
COMMIT;

ALTER TABLE TAB1 ADD COL3 NUMBER;
ALTER TABLE TAB1 MODIFY COL3 DEFAULT 100;

INSERT INTO TAB1(COL1, COL2) VALUES(3, 'C');
INSERT INTO TAB1 VALUES(4, 'D', NULL);
COMMIT;

SELECT SUM(COL3)
  FROM TAB1;
```

① 100
② 200
③ 300
④ 400

49. 외래키에 대한 설명 중 적절하지 <u>않은</u> 것은?

① 외래키를 생성하기 위해서는 반드시 참조키에 기본키 또는 고유키가 먼저 생성되어야 한다.
② 외래키 설정 후에는 참조키에 생성된 제약조건을 삭제할 수 없다.
③ 외래키 설정 후 부모 테이블 삭제 시 CASCADE 옵션을 사용하면 자식 테이블을 함께 삭제할 수 있다.
④ 외래키를 ON DELETE CASCADE 옵션과 함께 생성하면 부모 데이터 삭제 시 부모키를 참조하고 있는 자식 데이터도 함께 삭제할 수 있다.

50. 사용자와 권한에 대한 설명으로 가장 적절한 것은?

① 자신이 소유한 테이블에 대해서는 조회 권한을 다른 사용자에게 언제나 부여할 수 있다.
② SYSTEM 계정에서 테이블을 생성하는 경우 항상 SYSTEM 소유로 생성된다.
③ 동시에 여러 테이블에 대한 조회 권한을 부여할 수 있다.
④ 권한을 동시에 여러 사용자에게 부여할 수 없다.

1회차 답안 및 해설

1. ④

모델링의 3 단계는 개념적, 논리적, 물리적 모델링이 있으며, 이 중 추상화 수준이 가장 높고 업무를 분석하는 단계는 개념적 모델링이다.

2. ④

엔터티는 현실 세계를 반영한 식별 가능한 객체나 사물을 나타내는데, 두 개 이상의 인스턴스와 둘 이상의 속성을 가져야 한다. 또한 유일한 식별자에 의해 식별이 가능해야 하며, 주요 속성이 주식별자에 대해 함수적 종속성을 가져야 한다.

3. ③

일반속성은 엔터티 구성 방식에 따른 분류(PK, FK, 일반속성)에 속한다.

4. ④

고객과 서비스 구매의 관계의 차수는 1 이므로 하나의 고객이 여러 서비스를 구매할 수 있다. 또한 고객 -> 서비스 구매는 선택적 관계이므로 어떤 고객은 서비스를 구매하지 않아도 되지만, 반대는 필수적 관계이므로 서비스 구매는 구매한 고객이 반드시 정의되어야 한다.

5. ④

존재성이란 주식별자가 지정되면 반드시 값이 존재해야 하며 널을 허용하지 않는 특성을 말한다. 따라서 사원 번호가 없을 수 없다는 것은 널을 허용하지 않는 특징이므로 존재성에 해당한다.

6. ③

제 3 차 정규화는 이행적 종속을 없애도록 엔터티를 분리하는 것을 말한다. 대출번호 -> 대출자 번호 -> 대출자 명이 성립되므로 대출자 번호와 대출자 명(대출자 직업 포함)을 분리한다. 같은 방식으로 대출 도서 명, 출판사 명, 출판년월, 대표 저자 명을 도서 엔터티로 나누게 되면, 총 도서 대출, 대출자, 대출 도서 3 개의 엔터티로 나눌 수 있다.

7. ④

개인 회원 또는 법인 회원 중 하나만 주문이 가능한 경우 두 엔터티의 관계는 상호배타적 관계이다.

8. ②

트랜잭션 전후의 데이터가 일관적인지를 설명한 내용이다. 원자성은 모두 성공하거나 모두 실패해야 한다는 특징이며, 고립성은 트랜잭션 실행 도중 다른 트랜잭션에 영향을 받지 않는다는 특성이다. 또한, 지속성은 트랜잭션이 성공적으로 수행되면 갱신한 데이터베이스 내용이 영구적으로 저장되는 특징을 말한다.

9. ③

NULL 로 구성된 집합의 COUNT 결과는 0 이다.

10. ①

주식별자의 반대는 보조식별자로, 인스턴스를 구분할 수 있지만 대표성을 가지지 못하는 식별자를 말한다. 본질식별자는 업무적으로 존재하는 식별자, 인조식별자는 업무에서 사용되지는 않지만 인위적으로 만든 식별자를 의미하며 복합식별자는 단일 식별자의 반대로 여러 속성으로 구성된 식별자를 의미한다.

11. ③

FROM 절은 ORACLE 에서는 생략 불가하다. GROUP BY 절과 HAVING 절은 순서가 바뀌어도 상관없으나 가급적 순서대로 사용한다.

12. ①

TRUNCATE 는 테이블의 구조는 남기고 데이터만 전체 삭제하는 명령어로, DML 과는 다르게 자동 확정(AUTO COMMIT)되기 때문에 DDL 에 속한다.

13. ③

WHERE 절은 SELECT 절보다 먼저 수행되므로 SELECT 절에서 정의한 컬럼 별칭을 사용할 수 없다.

14. ②

LPAD 를 사용하여 'X' 왼쪽에 'X'를 삽입하여 총 5 글자(5Bytes)로 리턴한다. INSTR 함수를 사용하여 5 번째 위치에서부터 두 번째로 발견된 '.'의 위치는 16 이다(맨 왼쪽 글자부터 위치가 1 이다).

15. ③

ROUND(12345.678, -2)는 정수 두 번째 자리(십의 자리)에서 반올림하여 맞추기 때문에 12300 이 리턴된다.

16. ②

NVL, COALESCE, ISNULL 모두 널을 만나면 그 다음 인수 값을 리턴하는 널 치환 함수이다. NULLIF(COMM, 100)은 COMM 값과 100이 같으면 NULL을, 다르면 COMM 값을 리턴하므로 NULL, 500이 출력된다.

17. ③

SAL > 4000 이면 SAL - 4000 은 양수이므로 SIGN 을 취한 값이 1 이 리턴된다. DECODE 문을 해석하면 결국 SAL 이 4000 보다 큰 경우 1 을, 그렇지 않은 경우 0 을 리턴하므로 최종 출력값은 2 이다.

18. ③

CASE 문 축약형 문법(비교 대상이 CASE 와 WHEN 사이에 있는)은 반드시 비교 대상과 비교 상수의 데이터 타입이 일치해야 한다. SUBSTR 결과는 항상 문자로 출력되므로 숫자 상수와 비교하는 3 번 지문은 에러가 발생하여 출력되지 않는다.

19. ③

오라클에서 날짜 연산 단위는 DAY(일)이다. 따라서 30/24/60 은 30*1/24/60 인데 1/24/60 는 1 분을 의미한다. (1(하루)를 24 로 나누면 1 시간, 다시 60 으로 나누면 1 분이 된다.) 따라서 30 분 이전을 출력하되, TO_CHAR 에서 요구하는 날짜 포맷으로 출력하면 2024.08.24 09:30:00 이 된다.

20. ①

1 번은 CASE 문에 의해 10 번 부서원의 경우 1, 그 외 부서원들은 0 을 생성하고, 이를 모두 COUNT 하면 1 과 0 을 모두 세므로 10 번 부서원만 셀 수 없다. 따라서 2 번 보기처럼 ELSE 를 생략하여 조건에 맞지 않는 값을 NULL 로 리턴한다면 NULL 은 COUNT 하지 않으므로 조건에 맞는 대상만 셀 수 있다. SUM 의 경우는 조건에 맞을 경우 1, 그렇지 않을 경우 0 으로 출력하면 조건에 맞는 대상의 개수를 구할 수 있다.

21. ②

'_1%' 조건은 두 번째 값이 1 인 문자열을 나타내므로 A1C1, B1A0, C1A1, D1B2 총 4 개이고, '%A%' 조건은 A 를 포함하는 문자열이므로 A1C1, A2C2, B1A0, C1A1 총 4 개이다. OR 조건은 두 집합의 합집합(중복값은 한 번만)이므로 총 8 개에서 두 조건을 모두 만족하는 A1C1, B1A0, C1A1 을 제외하면 총 5 개이다.

22. ②

연산자는 NOT > AND > OR 순서대로의 우선순위를 가지고 있다. 따라서 1, 3, 4 번의 경우 DEPTNO = 20 AND JOB = 'CLERK' 조건이 먼저 실행되고, 이 결과에 DEPTNO = 10 조건에 만족하는 집합을 합하여 리턴되므로 2 번과의 결과가 달라진다.

23. ④

SUM(COL2)는 NULL 만 제외하고 총합을 리턴하므로 500 이 출력된다. 하지만 2 번 지문의 경우 COL1 > 0 조건에 만족하는 행은 첫 번째 행 하나이므로 SUM(COL2) + SUM(COL3) = NULL + 100 이 되며, 이때는 NULL 이 리턴된다.

24. ④

GROUP BY 절에 나열되는 순서는 그룹의 수에 영향을 주지 않는다. 즉, GROUP BY A, B 와 GROUP BY B, A 모두 A, B 값이 모두 같을 때만 한 그룹으로 묶기 때문에 출력 순서만 달라질 뿐 총 그룹 연산 수행 결과는 같다.

25. ④

첫 번째 문장은 COL1 = 200 조건에 만족하는 값이 없고, 공집합을 GROUP BY 하면 아무것도 출력되지 않는다. 따라서 공집합에 대한 COUNT 결과는 0 이 아닌 공집합이 출력되므로 컬럼명만 출력되는 형태로 리턴된다. 두 번째 문장은 첫 번째 행에 대한 SUM(COL2)를 출력하게 되는데, 첫 번째 행의 COL2 값이 NULL 이므로 NULL 이 출력된다. NULL 로만 구성된 데이터의 SUM, AVG, MIN, MAX 등의 결과는 NULL 이 리턴되며 COUNT 만 0 으로 출력된다. 따라서 세 번째 문장의 출력값은 0 이 된다.

26. ④

단일 DEPTH를 갖는 SELECT 문의 컬럼 별칭은 테이블명이나 테이블 별칭이 앞에 붙을 수 없다. 아래와 같은 경우 테이블 별칭이 붙을 수 있다.

```
SELECT *
  FROM (SELECT EMPNO AS NN, ENAME
              FROM EMP) I
 ORDER BY I.NN;
```

27. ①

TO_CHAR에 의해 SAL 값이 문자로 변환되며, 문자 값의 대소 비교는 값의 왼쪽부터 비교하므로 왼쪽 값이 가장 작은 값의 크기가 제일 작다. 따라서 1300 < 2500 < 300 < 800 순이 되며, 오름차순 정렬은 1300, 2500, 300, 800 순서로 배치된다.

28. ③

NULL은 같지 않다에 참이 리턴되지 않는다. 따라서 TAB1 기준 COL2가 A인 경우는 1개, NULL인 경우 0개, B인 경우 1개, C인 경우 2개가 리턴된다. 따라서 이들의 COL1에 대한 총 합은 12가 된다.

29. ④

TAB1을 RULE_NAME에 매핑되는 규칙으로 TAB2에 조인을 수행했기 때문에, 각 행마다 LIKE 조건이 일치하면 TAB2 값이 출력된다. 즉, SCOTT은 %O%와, FORD는 %O%와 F% 둘 다 매핑되어 행이 두 개로 출력된다. 따라서 총 행의 수는 3건이 리턴된다.

30. ③

INNER JOIN은 값이 같으면 모두 출력되므로 TAB1의 KEY1의 값이 BB인 경우 2건, CC와 DD는 일치하는 값이 없으므로 생략, EE는 한 건이 출력되어 총 3건이 리턴된다. LEFT OUTER JOIN은 TAB1의 KEY1의 값이 모두 출력돼야 하므로 앞의 INNER JOIN 결과에 CC와 DD가 추가적으로 리턴되어 총 5건이 된다. FULL OUTER JOIN은 LEFT OUTER JOIN 결과와 RIGHT OUTER JOIN 결과의 합집합(중복 값은 한 번만 출력)이므로 총 6건이며 CROSS JOIN의 경우는 모든 발생 가능한 조합이므로 4X4, 16건이 리턴된다. NATURAL JOIN에 의해 COL1, COL2 값이 같은 값만 출력되며 이때 NULL은 같다고 볼 수 없기 때문에 제외되어 총 3건이 출력된다.

31. ②

1번. NULL은 IN 연산자에 의해 출력되지 않기 때문에 V1이 A, B인 경우만 리턴된다. 2번. NOT IN 연산자는 서브쿼리 결과에 NULL이 포함될 경우 전체가 거짓이 되므로 아무것도 출력되지 않는다. V1 NOT IN ('A', NULL, 'B', 'C') => NOT (V1 = 'A' OR V1 = NULL OR V1 = 'C') => V1 != 'A' AND V1 != NULL AND V1 != 'C' V1 != NULL은 거짓이므로 이 조건으로 인해 전체 조건이 거짓이 된다. 3번. EXISTS 연산자는 서브쿼리 결과가 참이면 메인 쿼리 결과가 리턴된다. 따라서 TAB1의 각 행마다 A.V1 = B.V1이 만족하면 TAB1 결과가 출력되므로 V1이 A, B인 행이 선택된다. 4번. NOT EXISTS 연산자는 반대로 서브쿼리 결과가 거짓이면 출력된다. 따라서 NULL끼리 같은 조건은 항상 거짓이므로 TAB1의 V1이 NULL인 경우와 C가 최종 리턴된다.

32. ②

서브쿼리 결과가 먼저 실행되어 2020년 2월 1일보다 입사일이 큰 행의 DEPTNO(30, 40, 50)가 출력되고 ALL이 작다와 결합되어 이들 중 30보다 작거나 같은 조건으로 전달된다. 따라서 DEPTNO가 10, 20, 30인 EMPNO의 수는 총 3개이다.

33. ②

2번 지문의 경우 스칼라 서브쿼리 결과가 여러 행인 경우 출력이 불가하다. 또한, 메인 쿼리 절에 사용된 WHERE A.NO = B.NO 조건은 실행이 불가한데 메인 쿼리에서는 FROM 절에 A 테이블만 선언되었기 때문에 B 테이블은 인식할 수 없기 때문이다.

34. ②

서브쿼리 결과가 항상 거짓이므로 조건에 만족하는 행이 없다. 이 경우 COUNT는 NULL이 아닌 0을 리턴한다.

35. ②

고객과 서비스 구매 관계에서 고객은 필수, 서비스 구매는 선택적 관계이므로 LEFT OUTER JOIN을 하면 서비스 구매를 하지 않은 고객의 서비스 구매 정보가 NULL로 출력된다. 따라서 이때, 고객의 성별로 서비스 구매나 서비스 테이블의 컬럼을 COUNT할 경우 NULL은 세지 않기 때문에 정상적으로 구매를 한 고객 수가 리턴될 텐데 2번의 경우 고객 테이블의 고객 번호를 세기 때문에 성별 고객 수가 출력된다.

36. ①

1번 보기의 경우 DEPTNO와 DEPARTMENT_ID 컬럼의 데이터 유형이 다르기 때문에 에러가 발생한다. 3, 4번 보기의 경우 집합 연산 결과에 정렬을 하는 구문이므로 정상적으로 수행된다.

37. ③

ROLLUP(A, B)는 A별, (A, B)별, 전체 그룹 연산 결과가 출력, CUBE(A, B)는 A별, B별, (A, B)별, 전체 그룹 연산 결과 출력 GROUPING SET은 나열한 것만 출력된다.

38. ③

SUM의 경우 ORDER BY 사용 시 ORDER BY 컬럼 순서대로 누적 합을 계산하는데, 이때 범위는 RANGE가 기본이다. RANGE란 ORDER BY 절에 명시된 컬럼의 값이 같을 경우 하나의 그룹으로 묶어서 누적 합을 계산하는 범위를 말한다. 따라서 2024.01.02 값이 두 개이므로 각각 1000과 2000이 먼저 3000으로 결합되어 두 번째와 세 번째 행의 누적 합이 둘 다 4000이 된다. DENSE_RANK의 경우 동순위 발생 뒤 순위가 연속적으로 출력되므로 주문번호가 3인 행의 순위는 2위가 된다.

39. ④

FIRST_VALUE, LAST_VALUE는 ORDER BY 컬럼 순서대로 범위 내 가장 처음 값과 마지막 값을 리턴하는 함수인데, 이때 기본 범위는 RANGE UNBOUNDED PRECEDING AND CURRENT ROW이다. 따라서 같은 DNAME 내 SAL 순서대로 가장 앞에 있는 이름은 아시아지부의 경우 홍길동, 남유럽지부의 경우 김길동이 된다. 하지만 각 행마다 LAST_VALUE를 구할 때 범위가 처음부터 현재 행까지만 고려해서 마지막 값을 리턴하기 때문에 항상 현재 행의 값이 마지막 값이 되므로 각 행의 값이 리턴된다.

40. ④

성적이 높은 순서대로 3명을 뽑는 쿼리로 적절한 것은 1, 2, 3번 보기이다. 2번의 경우 WITH TIES를 사용하여 동순위까지 출력하므로 98점과 함께 80점 두 명이 모두 출력된다. 하지만 ROWNUM의 경우 SELECT 절이 ORDER BY 절보다 먼저 수행되므로 SELECT 절에서의 ROWNUM은 ORDER BY 결과를 반영하지 못한다. 따라서 성적이 높은 순서대로 RN의 값이 형성되지 않기 때문에 원하는 결과를 얻을 수 없다.

41. ①

각 행이 리프 노드 데이터인지 여부를 출력해주는 계층형 질의절의 가상 함수는 CONNECT_BY_ISLEAF이다.

42. ②

매니저 사원번호가 NULL인 지점을 시작으로 레벨 1을 부여, 나사장의 사원번호를 매니저 사원번호로 가지면서 하반기 입사자인 행을 찾아 레벨 2를 부여한다. 이렇게 연결된 행으로부터 CONNECT 절의 조건을 만족하는 행을 계속 이어 나간다. 이때, START WITH 절은 CONNECT BY에 있는 조건에 따라 생략되지 않기 때문에 나사장이 상반기 입사자라도 출력된다.

43. ①

10, 20, 30번 부서원의 수를 각 컬럼별로 표현한 것은 LONG -> WIDE로의 변환을 수행하는 PIVOT의 결과로 대체할 수 있다. PIVOT 절에서 IN 앞의 컬럼은 서로 다른 컬럼으로 분해할 대상이 명시되어야 하므로 DEPTNO가 들어가야 한다.

44. ②

둘 다 숫자(\d)가 2회 이상 나열된 단어의 수를 찾는 문제이므로 C1은 2, C2는 3개가 리턴된다.

45. ④

[XY-]는 X 또는 Y 또는 -와 같다. 따라서 X, Y, - 중 하나가 여러 번 반복되면서 그 뒤에 Z값이 오거나 오지 않는 문자열 배열을 갖는 행은 전체이므로 총 5개가 출력된다.

46. ③

전체 평균(4000)보다 급여가 낮은 직원들의 급여만 수정되므로 김길동 두 명의 급여는 수정되지 않는다. SET절은 각 행의 급여 수정 시 부서명을 확인하여 서브쿼리에서 해당 부서의 최대 급여를 찾아 수정하는 구문이다. 따라서 아시아지부는 모두 3000으로, 남유럽지부는 8000으로 수정된다.

47. ②

첫 INSERT문장은 COMMIT했기 때문에 영구 저장된다. 그 이후 UPDATE, INSERT, DELETE를 차례대로 하지만 SAVE1 지점으로 롤백하므로 INSERT와 DELETE는 실행 취소된다. 첫 UPDATE문장과 마지막 UPDATE문장은 모두 롤백되어 결과적으로는 첫 INSERT문장만 실행된다. 따라서 남은 행의 SUM(COL2) 결과는 11이다.

48.④

길이가 다른 경우 길이가 같을 때까지 비교하여 모두 값이 같다면 길이가 큰 문자열을 더 큰 값으로 판단한다.

49. ④

NUMBER(5,2) 타입은 총 5자리 중 2자리를 소수점 자리로 사용하므로 정수 자리는 최대 3자리만 삽입 가능하다. 또한, 소수점 자리는 세 자리 이상 삽입은 가능하나 입력되는 것은 둘째 자리까지만 입력된다.

50. ②

테이블 소유자의 경우 소유한 테이블에 대한 조회, 수정 권한을 다른 유저에게 부여할 수 있다. 또한 WITH ADMIN OPTION 등을 통해 권한을 위임 받은 유저도 해당 권한에 대해 권한 부여가 가능하다.

2회차 답안 및 해설

1. ④

프로그램과 테이블간의 연계성을 낮추어야 한다. 연계성이 높을 경우 사소한 업무 변화에 대해서도 데이터 모델링의 큰 변화를 가져와 유연성이 떨어진다.

2. ③

발생 시점에 따라서는 기본, 중심, 행위 엔터티로 분류한다.

3. ③

하나의 인스턴스는 속성마다 반드시 하나의 속성값을 가져야한다.(속성의 원자성)

4. ③

관계의 페어링이란 엔터티 안의 인스턴스가 개별적으로 관계를 가지는 것을 의미한다. 하나의 엔터티와 다른 엔터티 간의 레코드 연결 방식을 나타내는 것은 관계의 차수이다.

5. ③

비식별 관계는 자식 엔터티가 부모 엔터티의 생명 주기와 독립적으로 존재할 수 있으므로 부모 엔터티가 소멸하더라도 자식 엔터티가 독립적으로 유지될 수 있다. 따라서 부모 엔터티의 인스턴스가 자식 엔터티의 인스턴스보다 먼저 소멸하는 경우 비식별자 관계로 연결해야 한다.

6. ②

제1정규화 : 속성의 원자성(한 속성이 하나의 값을 갖는 특성)을 갖도록 엔터티를 분해하는 단계
제2정규화 : 제1정규화를 진행한 엔터티에 대해 완전 함수 종속을 갖도록 분해하는 단계
제3정규화 : 제2정규화를 진행한 엔터티에 대해 이행적 종속을 갖지 않도록 분해하는 단계
-> 일반속성은 주식별자 전체에 종속적이다

7. ②

부모의 식별자를 자식의 일반속성으로 상속하면 비식별 관계, 부모의 식별자를 자식의 식별자에 포함하면 식별관계라고 할 수 있다.

8. ①

두 엔터티의 관계가 서로 필수적일 때 하나의 트랜잭션을 형성할 수 있다. 두 엔터티가 서로 독립적으로 수행이 가능하다면 선택적 관계로 정의한다.

9. ④

NULL 속성에 대해 바커 표기법에서는 동그라미가 NULL 허용 속성을 의미한다.

10. ①

인조식별자를 사용하면 불필요하게 발생하는 중복데이터를 막을 수 있는 것이 아니라 중복 데이터의 발생 가능성을 높이기 때문에 데이터 품질의 저하를 발생시킬 수 있는 단점을 가지고 있다.

11. ③

FROM절에서 테이블 별칭을 선언한 경우는 반드시 테이블 별칭만으로 컬럼을 구분해야 한다.
따라서 TABLE1.COL2가 잘못된 표현이다.

12. ④

SELECT문 수행 순서는 FROM > WHERE > GROUP BY > HAVING > SELECT > ORDER BY 순이다.

13. ④

ORDER BY절에는 GROUP BY에 사용하지 않은 컬럼을 명시할 수 없다.

14. ④

||(연결연산자)는 오라클에서의 문자열 결합 방식이며, SQL SERVER에서는 +를 사용하여 문자열을 결합 할 수 있다. & 연산자로 문자열 결합은 불가하다.

15. ②

TRUNC는 소수점 이하 버림으로 결과값인 5, CEIL은 값보다 큰 최소정수로 6이 출력된다. FLOOR는 값보다 작은 최대 정수가 리턴되므로 5, ROUND는 소수점 첫번째 자리에서 반올림하여 5가 출력된다.

16. ①

주민번호의 앞 6자리를 사용하여 날짜변환 시 RR 포맷을 사용하면 두 자리 연도가 1~49 사이면 2000년대를, 50~99이면 1900년도의 4자리 연도로 출력, YY를 사용하면 2000년대를 출력한다.

17. ②

ISNULL(대상, 대체값) 함수는 대상이 NULL이면 대체값으로 치환하는 함수로서 COL1이 NULL이면 3으로 대체하라는 쿼리이기 때문에 1,2,3,3이 나오게 된다.

18. ①

LTRIM 함수에 제거문자열을 전달하지 않을 경우 왼쪽에서 공백을 제거한다. INITCAP 함수는 첫 문자만 대문자로 나머지 문자는 소문자로 반환하는 함수이며, TO_CHAR 함수에 의해 소수점 둘째 자리까지, 정수자리는 세자리로 표현하여 리턴한다.

19. ②

COALESCE 함수는 대상들 중 널이 아닌 첫 번째 값을 출력하므로 첫 번째 행부터 10, 10, 20 이 출력된다.

20. ②

NOT IN문의 서브쿼리 결과 중 NULL이 포함되는 경우 데이터가 출력되지 않는다.
NULL은 논리적으로 비교할 수 없는 연산이기 때문에 NULL을 비교하는 연산자로 인해 전체조건이 거짓이 된다.
조건에 만족하는 값이 없으므로 SUM 결과는 NULL이 된다.

21. ③

WHERE절의 비교 연산 결과 COL2의 30과 40만 해당되고 그에 해당하는 COL1은 모두 NULL이다. GROUP BY에 의해 COL1이 NULL인 한 그룹이 생성되지만 NULL은 COUNT하지 않기 때문에 0이 출력된다.

22. ②

GROUP BY 후 SUM(COL2) 연산 결과 (10, 300), (20, 400), (30, 500), (NULL, 600) 그룹이 출력된다. 이들 중 HAVING 조건에 만족하는 그룹은 (20, 400), (30, 500), (NULL, 600) 이다.

23. ④

ORDER BY 시 DESC NULLS LAST 하면 NULL값이 맨 뒤에 배치된다.

24. ②

CASE문 결과를 1차 정렬로 한 뒤, 이 값이 같을 경우 COL1의 값으로 2차 정렬을 수행한다. 이렇게 정렬된 COL1의 값은 SCOTT, ALLEN, FORD, SMITH 순으로 출력된다.

25. ④

NULL 끼리는 동등비교 조건에 참으로 리턴되지 않는다. TAB1의 COL2값이 A인 경우 TAB2의 1건, B인 경우 2건, C의 경우 조건에 만족하는 값이 없으므로 INNER JOIN에서는 생략된다. 따라서 총 3건이 출력된다.

26. ④

(+) 가 붙은 반대편 테이블이 기준이 되는 테이블로 TAB1 테이블을 기준 테이블로 LEFT OUTER JOIN이 수행된다. 즉, 조인 조건이 일치하지 않아도 TAB1은 생략되서는 안되므로 INNER JOIN의 결과에 TAB1의 COL2가 NULL, C인 경우 추가적으로 출력되므로 총 5건이 나온다.

27. ③

NATURAL JOIN은 USING, ON, WHERE 절에서 조건 정의가 불가하다.

28. ①

LEFT OUTER JOIN의 결과 TAB1의 NO값이 1, 2, 4, 4, 6, 7, 3인 총 7개의 행이 나오게 되고, RIGHT OUTER JOIN의 결과로 TAB1의 CODE값이 A, B, B, B, NULL, NULL, NULL이 출력되어 DISTINCT 수는 총 2건이다.(NULL은 세지 않는다)

29. ④

서브쿼리 조건절은 TAB1의 각 행의 COL1을 확인하여 같은 값을 갖는 행들 중 COL2의 최대값과 일치하는 행을 찾아 COL2의 총 합을 묻는 질의절이다. 즉, COL1별 COL2 값이 최대인 행들의 COL2의 총 합을 리턴하는 문장이므로 A그룹에서는 30,30, B그룹에서는 40이 리턴되어 총 100이 출력된다.

30. ②

2번 보기를 제외한 모든 지문은 기혼 40대 여성 중 구매이력이 있는 고객의 고객아이디를 출력하는 문장이다. 실제 구매이력은 ORDERS에 있기 때문에 ORDERS의 CUSTOMER_ID 값에 존재하는지를 확인하여 구매자의 고객번호를 특정할 수 있다. 이는 INNER JOIN, IN, EXISTS 연산자로 구현 가능한데, 2번 보기의 경우 인라인 뷰의 결과가 PROMOTION_ID 가 1 이상인 상품을 구매한 고객 아이디만 특정하기 때문에 전체 상품 구매를 기준으로 출력하는 다른 보기와는 결과값이 다르게 출력된다.

31. ②

셀프조인을 사용하여 EMP에서의 각 직원별로 입사일이 빠른 직원의 수를 계산하는 질의절이다. 이 때, LEFT OUTER JOIN을 수행하였기 때문에 입사일이 가장 빠른 SMITH의 경우도 CNT가 0으로 출력된다.

32. ②

① 단일 행 서브쿼리는 단일 행 비교연산자인 =, <>, >, >=, <, <=의 연산자를 주로 사용한다.
③ 메인쿼리에 값을 제공하기 위한 목적으로 사용하는 쿼리는 비연관 서브쿼리이다.
④ 연관 서브쿼리는 일반적으로 메인쿼리가 먼저 수행된 후에 서브쿼리에서 조건이 맞는지 확인하고자 할 때 사용하기 때문에 항상 서브쿼리 조건이 만족하는지를 확인하는 방식이라고 볼 수 없다.

33. ③

결과표를 보면 고객이 보유한 포인트에 맞춰서 상품을 출력한 것을 알 수 있다. 따라서 고객이 보유한 포인트가 상품 테이블의 최소포인트와 최대포인트 사이에 있는 조건을 갖는 쿼리는 3번이다.

34. ③

T2.COL2 값의 그룹별로 COL3의 평균보다 T1.COL3의 값이 큰 행을 찾고, 이들의 T1.COL1의 총 합을 구하는 질의절이다. T2에서 COL2의 값이 A인 그룹의 AVG(COL3)은 20이므로 T1에서 A그룹이면서 COL3의 값이 20보다 크거나 같은 대상을 찾으면 0건이 출력된다. 마찬가지로 T2에서 B그룹의 AVG(COL3)을 구하면 20이고, T1의 B그룹중 COL3의 값이 20보다 크거나 같은 행은 COL1의 값이 4,5인 행이므로 최종 결과는 9가 리턴된다.

35. ②

먼저 TAB1과 TAB2의 UNION 결과는 아래와 같다. 이들 중 TAB3의 결과를 빼면 2번 결과가 같다.

10	A
20	A
20	B
30	C
30	D
40	E
50	F
	A

36. ①

GROUPING SETS(PRODUCT_NO, GOGAK_NO, ())에서 PRODUCT_NO별 SUM(QTY) 결과, GOGAK_NO별 SUM(QTY)연산 결과가 출력된 것과 ()으로 인해 SUM(QTY)의 전체 총 합이 출력된 것을 찾는 문제이다.

37. ②

RANK는 동점일 경우 같은 등수로 표시하고 다음 순위는 동점인 순위의 수만큼 밀리므로 12245가 출력되지만
DENSE_RANK는 동점일 경우 동순위를 부여 뒤, 다음 순위가 바로 이어지므로 12234가 리턴된다.
ROW_NUMER는 동점일 경우를 인정하지 않고 순서대로 나열하므로 12345가 최종 출력된다.

38. ④

PERCENT_RANK는 특정 값의 상대적 비율이 아닌, 그 값의 위치를 백분율로 리턴하는 함수이기 때문에 급여의 비율을 출력하기 위한 표현으로 적절하지 않다.

39. ①

고객 테이블에서 포인트가 높은 순서대로 정렬을 한 후 2개의 행을 건너 뛰고 3번째부터 2개의 행을 뽑는다.

40. ③

START WITH 조건이 1006과 1001이므로 두 행이 1레벨이 되고, 해당 행의 상위관리자코드를 사원번호로 갖는 행을 찾으면 둘 다 홍길동이 출력된다. 따라서 정답은 3번이 된다.

41. ②

PRIOR에 대한 설명이다. 각 행별로 연결 시 PRIOR 컬럼을 먼저 읽고, 뒤에 있는 컬럼과 일치하는 행을 찾기 때문이다.

42. ①

WIDE -> LONG 데이터로 변환하는 과정이므로 UNPIVOT이 적절하다. UNPIVOT은 IN으로 LONG 데이터로 변환할 대상을 지정한다.

43. ②

A 뒤에 X 또는 Y가 여러 개 오며 그 뒤에 .이 오는 문자열을 찾아 모두 지우는 쿼리문이다.

44. ③

[^0-9]+ 는 숫자가 아닌 값이 여러 개 반복되는 문자열을 의미한다. REGEXP_SUBSTR은 이 패턴에 해당하는 값을 처음부터 찾아 단 하나의 문자열을 추출하므로 ORA-만 추출된다.

45. ④

4번에서 COL4의 값은 문자상수이므로 날짜 변환 후 입력을 해야 한다. DBMS의 기본 날짜 포맷이 YYYY/MM/DD가 아닌 경우는 이 문장은 에러가 발생한다.

46. ④

처음 INSERT 3개의 문장은 COMMIT을 수행했으므로 영구 저장된다. 이후 COL3을 추가하면 이미 입력된 세 개의 행에 대해 NULL을 갖게 된다. 그 뒤 수행하는 INSERT, UPDATE, DELETE 문장은 이어서 실행하는 ALTER TABLE DROP COLUMN 문장으로 인해 자동 확정된다.(DDL AUTO COMMIT). 따라서 ROLLBACK을 수행해도 취소되지 않는다.

47. ③

COL4 컬럼 추가 시, 기존 세 개의 행의 값은 NULL이 삽입된다. 그 이후 DEFAULT 값을 변경해도 이전에 삽입된 행은 반영되지 않고, 이후 삽입되는 행에 대해 적용된다. COL3에 DEFAULT 값이 설정되어 있다 하더라도 NULL을 직접 입력하면 NULL이 삽입되며, 마지막 INSERT 문장처럼 COL3의 값이 아예 입력되지 않을 경우만 DEFAULT VAULE로 삽입된다.

48. ①

CHAR, VARCHAR 타입일 경우 데이터가 있어도 서로 변경가능 하기에 반드시 빈 컬럼일 필요는 없다.

49. ②

UNIQUE 제약조건에서는 NULL을 허용한다.

50. ④

중간관리자가 WITN GRANT OPTION으로 부여 받은 권한을 제 3자에게 부여한 경우, 관리자가 제 3자의 권한을 직접 회수할 수 없다. 하지만 중간관리자 권한을 회수하면 제 3자에게 부여한 권한도 함께 회수된다. 반대로 WITH ADMIN OPTION으로 부여할 경우 중간관리자 권한 회수 시 제 3자에게 부여한 권한은 함께 회수되지 않는다.

3회차 답안 및 해설

1. ①

도메인은 속성값이 갖는 범위를 의미한다.

2. ③

하나의 속성은 한 개의 속성값을 가져야 한다.

3. ④

속성은 엔터티에 속한 엔터티에 대한 자세하고 구체적인 정보를 나타낸다.

4. ②

관계의 차수에 대한 설명이다.

5. ②

주식별자는 NULL값이 들어갈 수 없다.

6. ①

제 1 정규화는 한 속성이 하나의 값을 갖도록 이를 분해하는 단계를 말한다.

7. ③

두 엔터티나 두 속성 간에 동시에 발생할 수 없는 관계는 상호 배타적 관계이다.

8. ③

하나의 트랜잭션은 부분 COMMIT이 불가하다.

9. ④

공백과는 다른 개념이므로 공백과 다른 ASCII 값을 갖는다.

10. ④

다른 엔터티 참조 없이 엔터티 내부에서 스스로 생성되는 식별자는 내부식별자이다.

11. ①

DBMS는 데이터를 중앙 집중화하여 여러 응용프로그램이 데이터를 공유하고 사용할 수 있도록 하는 소프트웨어이다.

12. ②

테이블명과 컬럼명은 반드시 문자로 시작해야 한다. 또한, 소유자가 다른 경우 같은 이름의 테이블을 생성할 수 있다. 즉, SCOTT.TABLE1과 HR.TABLE1은 서로 다른 테이블로 존재할 수 있다.

13. ③

GROUP BY 절에 명시되지 않은 컬럼은 ORDER BY 절에 사용할 수 없다.

14. ③

DISTINCT COL1, COL2의 경우 두 컬럼의 값이 모두 같은 집합을 중복값으로 간주, 하나만 출력하기 때문에 DISTINCT COL2, COL1의 결과와 순서만 다를뿐 집합의 수는 동일하다.

15. ③

ROUND와 TRUNC는 각각 날짜의 반올림과 버림을 수행할 수 있다. 두 번째 인수 생략 시 "일" 단위로의 반올림/버림이 진행되며, 'MONTH'의 경우 "월" 단위로의 반올림/버림이 진행된다. 즉, "일" 단위에서 반올림을 진행하게 된다.

16. ①

CEIL은 값보다 크면서 가장 작은 정수인 올림값을 리턴한다. 따라서 -12.345보다 값이 크면서 가장 작은 정수는 -12이다. 반대로 FLOOR는 값보다 작으면서 가장 큰 정수인 내림값을 리턴하므로 FLOOR(-12.345) 값은 -13이다.

17. ③

LTRIM은 왼쪽에서부터 특정 문자열를 지우며, 중간에 있는 문자열은 삭제되지 않는다. 따라서 LTRIM('ORACLE','A'은 왼쪽에 A가 없으므로 ORACLE 그대로 리턴된다. SUBSTR('SQL-SERVER', 3, 3)은 세 번째 위치에서 3개 문자열을 추출하기 때문에 L-S가 출력된다. 또한, REPLACE에 의해 E가 삭제된 문자열의 길이는 8이 된다.

18. ②

2번은 에러가 발생하는 문장으로 마지막 WHEN절에만 ELSE를 사용할 수 있다.

19. ④

오라클에서는 묵시적 형 변환으로 인해 숫자로 변환 가능한 문자값과 숫자값의 연산이 가능하다. 또한 "일"만 있는 경우 날짜로 변환하면, 현재 날짜의 연도와 월을 따른다. NVL의 경우 첫 번째와 두 번째 인수의 데이터 유형이 일치해야 한다.

20. ③

첫 번째 문장은 조건에 만족하는 COL1값이 NULL이므로 NULL 그룹이 리턴된다. 따라서 COUNT를 하면 0이 출력되며, 두 번째 문장은 HAVING 조건에 만족하는 그룹이 없으므로 공집합이 출력되어 COUNT 결과가 NULL이 된다.

21. ③

COL1이 NULL이 아닌 값은 위의 세 행인데, 이들의 COL2+COL3의 값은 순서대로 NULL, 12, NULL이 된다. 따라서 총 합은 12가 출력된다.

22. ①

NULL은 일반적인 비교연산을 수행할 수 없고 IS NULL, IS NOT NULL로 비교해야 한다.

23. ④

① 컬럼 별칭에 공백이 있는 경우 쌍따옴표로 묶어서 전달해야 한다.
② GROUP BY절에 있지 않은 컬럼을 SELECT절에 그룹함수 없이 전달할 수 없다.
③ WHERE절에는 그룹함수를 사용할 수 없고 그룹함수를 사용한 비교식은 HAVING절에 사용 가능하다.

24. ②

SUBSTR(JUMIN, 3, 2)은 태어난 월을 추출하기 때문에, 순서대로 12, 11, 06, 01, 08의 문자 유형으로 출력된다.
이를 TO_NUMBER를 사용하여 숫자 형태로 변환하면 12, 11, 6, 1, 8이 되는데, 이를 다시 TO_CHAR로 변환하게 되면 문자값의 비교 규칙에 따라 1 < 11 < 12 < 6 < 8 순서대로 출력된다. (문자는 가장 왼쪽부터 비교하여 값이 작을수록 작은값이 된다)

25. ③

DEPTNO가 작은순서대로, DEPTNO가 같은 경우 SAL이 큰 순서대로 정렬하여 출력한다.
NULL이 마지막에 출력되는게 기본 순서이지만 DESC로 내림차순 정렬하면 NULL이 젤 먼저 출력된다.

26. ③

조인 결과는 아래와 같다

TAB1_COL1	TAB1_COL2	TAB2_COL1	TAB2_COL2
1	A	1	A
2	B	2	B
2	B	2	B
4	D		
3	C		

따라서 COUNT(TAB1.COL1)의 결과는 5 개이다

27. ③

두 테이블 조인 시, 조인 조건에 의해 생략되는 쪽이 둘 다 없을 경우 LEFT OUTER JOIN / RIGHT OUTER JOIN 결과는 같다.

28. ②

하나의 로우에 해당하는 스칼라 서브쿼리 결과 건수가 0건이더라도 메인쿼리절에서 생략을 하지 않는 한 NULL로 출력된다.

29. ③

JAMES의 입사일보다 늦게 입사한 직원들의 급여 총 합을 구하는 문제이다. JAMES의 입사일인 1981/10/03 보다 늦게 입사한 직원은 ADAMS, FORD, MILLER 이며 이들의 급여 총합은 5400이다.

30. ④

다중컬럼 서브쿼리를 사용하여 각 부서별로 최대급여를 받는 직원들의 급여 총합을 출력하고 있다.
10번 부서는 1500, 20번 부서는 3000이지만 FORD와 MILLER 둘 다 출력되므로 총 합은 7500이다.

31. ①

먼저 구매테이블을 통해 고객별 최근 구매일을 파악한 뒤(인라인뷰), 해당 고객과 구매일에 구매한 상품번호를 파악하기 위해 다시 구매테이블과 조인이 필요하다.

그 뒤 상품테이블과 조인하여 해당 상품의 상품명을 알아내면 된다.

32. ③

TAB2의 STATUS가 OPEN인 CODE는 0002와 0004이므로 TAB1에서 이들을 삭제하면 AAA,BBB만 남는다. 이들의 FARE 총합은 600이다.

33. ③

ANY는 작다와 만나면 값들 중 최댓값을 리턴한다. 따라서, 메인쿼리 WHERE절은 COL2 < 350 이 되므로 A, B, C들의 COL2의 총 합은 600이 된다.

34. ②

UNION ALL은 중복된 데이터를 모두 출력하며 정렬은 발생하지 않는다.

35. ①

ROLLUP은 전체 소계를 함께 출력한다. 즉, ROLLUP(A) => GROUP BY A 결과에 전체 소계 출력

36. ③

PERCENT_RANK는 값이 아닌 행의 상대 위치를 0~1 사이값으로 반환하는 함수이다.

37. ④

누적합의 범위가 각 행마다 이전행과 현재행, 다음행을 연산하고 있으므로(JONES 기준 1100 + 2975 + 3000 = 7075) 1 PRECEDING AND 1 FOLLOWING 이며, SAL이 같은 SCOTT과 FORD의 누적합이 각각 다르게 계산되었으므로 ROWS가 적절하다.

38. ④

TOP(N)은 WITH TIES 를 사용하면 N개보다 더 많은 데이터 추출이 가능하다.

39. ①

WHERE절은 출력 대상을 결정하기 때문에 서울 지역인 홍길동은 출력하지 않는다.

40. ②

PRIOR의 위치가 PART에 있으므로 가장 최상위 학과(PART IS NULL)를 먼저 출력하고, 두 행의 PART를 DEPTNO로 갖는 행을 찾지만 해당 행이 없으므로 최상위 학과인 공과대학과 인문대학만 출력된다.

41. ①

첫 번째 밑줄은 10, 20, 30, 40 값이 쌓여 하나의 컬럼을 이룰 때 컬럼명을 나타내는 자리이므로 판매량,
두 번째 밑줄은 2023, 2024를 넣을 컬럼명을 의미하므로 연도가 적절하다.

42. ①

[A-z|0-9\.] 패턴은 영문 또는 | 또는 숫자 또는 \ 또는 . 그리고 공백을 모두 지칭하는 패턴이다. 따라서 이들을 모두 지우면 ,과 :만 남게 된다.

43. ③

UPDATE로 동시 여러 컬럼 수정 가능하다. DELETE 시 FROM은 생략이 가능하다.
DML 시 COMMIT 또는 ROLLBACK으로 트랜잭션을 종료하지 않으면 변경된 행의 잠금이 발생하여
다른 사용자의 사용에 제한이 생긴다.

44. ④

데이터베이스는 공유 저장 공간이므로 COMMIT 하여 영구 저장된 데이터는 다른 사용자에게 공유된다.

45. ①

컬럼 크기를 늘리는 것은 언제든지 가능하며 반대로 줄이는 것은 해당 컬럼의 최대 길이만큼 줄일 수 있다.

46. ③

기본키는 고유키와 NOT NULL 제약조건을 합쳐 놓은 것과 같다. 즉, 중복될 수 없으며 NULL이 삽입될 수 없다.

47. ②

새로운 컬럼 추가 시 기존 데이터의 새 컬럼 데이터는 NULL로 삽입된다. 따라서 DEFAULT 값 선언 없이는 NOT NULL 속성을 갖는 컬럼 추가는 불가능하다.

48. ③

ON DELETE SET NULL 옵션에 의해 TAB1 데이터 삭제 시 자식 데이터의 외래키 컬럼은 NULL로 수정된다.

49. ④

NULL값이 있더라도 중복된 값만 없다면 UNIQUE 제약조건을 추가할 수 있다.

50. ③

롤에 있는 권한을 회수하는 경우 롤을 부여받은 유저도 해당 권한을 즉시 잃게 된다. 4번 문장은 WITH GRANT OPTION에 대한 설명이다.

4회차 답안 및 해설

1. ①

모델링의 특징은 단순화, 추상화, 명확화가 있다. 이 중 누구나 이해하기 쉽게 대상에 대한 애매모호함을 제거하고 정확하게 기술하는 것은 명확화에 해당된다.

2. ④

행위 엔터티는 발생 시점에 따른 분류에 속한다.

3. ④

파생속성은 데이터 정합성을 유지하기 위해 가급적 적게 정의하는 것이 좋다.

4. ④

관계는 관계명, 차수, 선택성으로 구성된다.

5. ①

부모 엔터티의 주식별자를 상속받아 자식 엔터티에서 외부식별자이면서 주식별자로 사용하는 관계는 식별관계이다.

6. ②

기본키(고객번호+상품명) 중 상품명에 의해 가격이 결정되므로 완전 함수 종속성을 위배하였다. 따라서 제 2 정규화 규칙을 위반하였다.

7. ②

정규화에 의해 분리된 두 테이블은 서로 관계를 맺는다.

8. ④

트랜잭션의 특징으로는 일관성, 원자성, 지속성, 고립성 등이 있다.

9. ③

NULL이 포함된 컬럼의 SUM 값은 NULL을 무시하고 연산한 결과가 리턴된다.

10. ②

주식별자를 구성하는 속성 중에서 유일성을 만족하는 최소한의 속성으로 구성하는 특성은 최소성이다.

11. ②

하나의 테이블은 반드시 한 계정의 소유여야 한다. 테이블명은 소유자가 다른 경우 같은 이름으로 생성 가능하다. 하나의 행의 하나의 컬럼에는 반드시 하나의 값만 삽입되어야 하며, 테이블 생성 시 정의한 컬럼의 데이터 유형은 테이블 생성 이후 ALTER 명령어로 변경 가능하다.

12. ④

데이터베이스의 크기나 복잡한 관계 및 다양한 쿼리 패턴 등의 이유로 부하를 분석하기가 어렵다.

13. ③

GROUP BY는 NULL로만 구성된 NULL 그룹을 리턴한다. 일반적으로 SELECT 절에서는 * 또는 컬럼명을 사용한다.

14. ④

GROUP BY 절에 포함되지 않은 컬럼으로 정렬할 수 없다. 또한, 컬럼 별칭을 ORDER BY 절에 사용할 경우 컬럼 별칭 앞에 테이블명이나 테이블 별칭을 붙일 수 없다.

15. ③

POWER 는 거듭제곱을 출력하는 함수이다. 따라서 POWER(3,3) 는 3의 3 거듭제곱, 27이 리턴된다.
LAST_DAY는 지정된 날짜가 속한 달의 마지막 날짜를 리턴한다.

16. ①

DEPTNO가 10이면서 JOB이 CLERK인 경우 A, DEPTNO가 10이면서 JOB이 CLERK가 아닌 경우는 B를 리턴하며, DEPTNO가 10이 아닌 경우는 모두 C를 리턴한다.

17. ②

HH24는 24시간 표현식이고, HH는 12시간 표현식이다. 시간은 정오를 기준으로 반올림 시 자리수가 바뀌는데, 둘 다 정오를 나타내므로 반올림 시 "일(DAY)"의 자리가 바뀌며 24년 8월 25일이 리턴된다.

18. ③

ADD_MONTHS는 지정한 날짜에서 n개월 이후 날짜를 출력하는 함수이다. NEXT_DAY(날짜,요일) 는 지정한 날짜 뒤의 첫 번째 지정요일에 해당하는 날짜를 리턴한다.

19. ①

NULLIF(COL1, 100)은 COL1 값이 100이면 NULL을 리턴하고, 같지 않으면 COL1값을 리턴한다. 따라서 COL1값은 순서대로 NULL, NULL, 200이 리턴되므로 COL2와의 합은 순서대로 NULL, NULL, 300이 되어 총 300이 출력된다.

20. ②

NOT EXISTS는 서브쿼리 조건이 거짓인 경우 메인쿼리의 결과가 출력된다. 따라서 TAB2의 COL2와 일치하지 않는 값은 1이며, NULL은 TAB1과 TAB2가 모두 존재하지만 EQUAL(=) 연산 결과가 항상 거짓이므로, NOT EXISTS에 의해 출력된다. 따라서 COUNT 결과는 2이다.

21. ②

2000과 3000사이 값이 아니면 2000미만, 3000초과가 된다. 이들 중 COL3이 10, 20과 일치하지 않는 행은 존재하지 않는다. 따라서 COUNT 결과는 0이다. 조건에 만족하지 않더라도 COUNT는 NULL이 아닌 0을 리턴한다.

22. ①

COL1 > 100 조건에 만족하는 값은 COL1이 200, 300, 400, 500, 300인 행이다. 이들을 COL2에 의해 그룹핑을 하면 A, B, NULL그룹이 리턴되며, NULL일 때의 SUM(COL3), MIN(COL3)은 모두 20이 리턴된다.

23. ③

문자 B보다 큰 ID는 BI와 BAA이다. 따라서 ID 는 CASE문에 의해 순서대로 AA, ABC, A, B, A 변환되어 두번째 정렬 기준인 ID 값과 함께 정렬된다. 문자 정렬은 왼쪽부터 비교하여 값이 같을 때까지 비교하여 더 큰 값이 큰 문자열이 되므로 최종 정렬 결과는 정렬 결과는 A(BAA), A(BI), AA, ABC, B 가 된다.

24. ③

ERD를 보면, 회원과 이용내역은 1대 다 관계이고 회원 -> 이용내역이 선택적 관계이다. 즉, 한 고객이 이용을 하지 않을 수 있기 때문에 상품을 이용하지 않은 고객까지 출력을 원한다면 OUTER JOIN이 수행돼야 한다. 또한 회원 엔터티 기준으로 이용내역이 선택적 관계이면 상품관계 또한 회원 엔터티 입장에서 선택적이다(이용내역이 없으면 이용 상품도 없기 때문) 따라서 상품 테이블도 OUTER JOIN이 수행돼야 한다. 따라서 가장 적절한 것은 3번이다. 4번의 경우 두 번째 조인조건에서 상품.상품번호 뒤에도 (+)를 붙여야 상품 테이블에도 OUTER JOIN이 연산된다.

25. ③

조인 조건에 성립하는 행만 출력하는 INNER JOIN이 수행되었으므로 조인조건을 ON절에, 일반조건을 WHERE절에 각각 명시하면 된다. USING을 사용하는 경우 반드시 괄호로 묶어서 전달해야 한다.

26. ④

NATURAL JOIN은 USING이나 ON절을 사용할 수 없다.

27. ④

서브쿼리가 메인쿼리 컬럼을 가지고 있을 경우 연관 서브쿼리라고 하며, 주로 메인쿼리가 먼저 수행된 후에 서브쿼리에서 조건이 맞는지 확인할 때 사용한다.

28. ③

RESULT1은 TAB2의 COL1의 값이 TAB2의 COL1과 같은 행의 COL2값을 리턴한다. 하지만 TAB1의 COL2가 NULL인 경우 조건에 일치하지 않아도 생략되지 않고 NULL로 출력되는데, 이는 메인쿼리의 WHERE절이 정의되지 않아 전체 행이 출력되기 때문이다. 마찬가지로 RESULT2도 TAB1의 COL1이 D인 경우 TAB1.COL1 = TAB3.COL1 조건에 일치하지 않지만 NULL로 출력된다.

29. ④

1, 2, 3 번 보기는 모두 코드별 최고가를 찾은 쿼리이다. 따라서 코드별로 최고가가 아닌 행들은 생략되는데, 4번 지문은 메인쿼리에 WHERE절이 없으므로 전체 행이 출력된다.

30. ④

A는 100, 200을 B는 300, C는 300을 리턴한다. 따라서 총 합은 900 이다.

31. ③

셀프조인을 통해 각 행마다 입사일이 작거나 같은 모든 행을 출력하는 쿼리로, 각 행마다의 누적 급여 총 합을 출력하게 된다.

32. ③

스칼라 서브쿼리는 OUTER JOIN을 수행하지 않아도 연결 조건에 만족하지 않는 행도 출력된다. 즉, D가 출력되는데, NVL로 NULL을 100으로 치환하고 있으므로 D의 PRO_FARE는 100으로 리턴된다.

33. ④

두 집합의 컬럼 사이즈는 달라도 집합 연산이 가능하다.

34. ③

DEPTNO별, JOB별, 전체 소계를 출력하는 함수는 GROUPING SETS(DEPTNO, JOB, NULL) 이다.

35. ②

같은 DEPTNO내에서 SAL이 낮은 순서대로 이전 이전 값을 가져오는 문장이다. 가져올 값이 없을 경우 0으로 리턴한다.

36. ④

FIRST_VALUE를 사용하여 최소값을 리턴할 수 있다.

37. ①

RATIO_TO_REPORT는 총 합 기준 COL2의 값의 크기에 대한 차지 비율을 출력, CUME_DIST는 COL2 순서대로 각 행의 상대적 누적 위치 출력, PERCENT_RANK는 COL2 순서대로 각 행의 누적 분위수(0~1 사이)를 출력한다.

38. ③

그룹으로 나눌 때 명학히 나눠지지 않으면 앞 그룹의 크기를 더 크게 나누므로 10번 부서의 1번 그룹원은 MILLER, CLARK이며, 20번 부서의 2번 그룹원은 SCOTT과 FORD 이므로 1300 + 2450 + 3000 + 3000 = 9750 이 된다.

39. ②

TOP 쿼리는 성적이 높은 순서대로 동순위 포함하여 3명을 출력하므로 홍길동, 박길동, 최길동이 출력된다.

40. ③

위 데이터는 SMITH와 ALLEN이 서로 순환구조를 가지기 때문에 출력이 불가하다. 따라서 출력을 원할 경우 CONNECT BY 뒤에 NOCYCLE 옵션을 전달해야 한다.

41. ②

최상위 학과인 컴퓨터정보학부와 인문사회학부로부터 시작하여 하위 학과들을 연결한 후 루트노드로부터의 연결과정을 '-'로 이어서 출력하는 과정이다. DEPTNO가 103번인 소프트웨어공학과의 상위학과는 컴퓨터정보학부이다.

42. ④

PIVOT 시에는 IN절에 나열한 값들을 갖는 행들이 결합되어 출력되므로 반드시 FOR 앞에는 집계함수 형태로 전달되어야 한다.

43. ③

문자열 처음부터 찾아서 두 번째로 발견되는 숫자의 연속 문자열을 XXX로 치환하면 031-XXX-4567 이 리턴된다.

44. ③

서브그룹을 추출하는 문제이다. 서브그룹 순서는 ((A)-(B))-((C)-(D))에서 A-B -> A -> B -> C-D -> C -> D 순서대로 정해진다. 따라서 4번째 서브그룹은 뒤에 두 숫자집합의 결합인 4545-233 가 된다.

45. ②

INSERT SELECT 구문에는 AS가 붙지 않는다. 양쪽 테이블의 컬럼수가 다르므로 INSERT 시 적절하게 컬럼수를 맞춰서 입력해야 한다.

46. ④

4번 선지의 COMMIT이후 까지만 ROLLBACK 된다. 가 틀린 표현입니다.
COMMIT 이후까지 ROLLBACK되는 것이 아니라 ROLLBACK 명령어 자체가 에러가 발생합니다.

간단하게 예를 설명하면

1 SELECT * FROM EMP_T1;

2 SAVEPOINT SP1;

3 DELETE FROM EMP_T1 WHERE JOB = 'CLERK';

4 COMMIT;

5 INSERT INTO EMP_T1 VALUES(1111,'PARK','CELRK',1112,SYSDATE, 8000, NULL, 20);

6 ROLLBACK TO SP1;

이렇게 있다고 가정할 때 6번의 ROLLBACK TO SP1; 을 실행하게 되면 COMMIT이후 5번의 문장이 ROLLBACK되는 것이 아니라 ROLLBACK 에러 문구가 발생합니다. 즉 5번 문장은 실행이 그대로 되어 있습니다.

47. ②

문자 유형은 반드시 최대 크기를 지정해야 한다.

48. ③

PRIMARY KEY 나 UNIQUE 설정 없이 부여된 NOT NULL 속성은 함께 복제되며 PRIMARY KEY 나 UNIQUE로 인해 만들어진 NOT NULL 속성은 복제되지 않는다.

49. ③

ON DELETE CASCADE 옵션으로 외래키 생성 시 부모 데이터 삭제 시 자식 데이터도 함께 삭제된다.

50. ③

이미 정의되어 있는 뷰는 다른 뷰의 정의에 기초가 될 수 있다.

5회차 답안 및 해설

1. ②

스키마의 3단계 구조인 외부, 개념, 내부 스키마 중 개념 스키마에 대한 설명이다.

2. ②

엔터티의 이름을 정할 때 약어 사용은 자제하고 현업에서 사용하는 용어를 사용한다. 또한 사용되지 않는 고립 엔터티는 추후 제거를 고려한다.

3. ④

복합 속성이란 속성값이 여러 의미로 구성되어 분해가 가능한 속성을 말하며, 주소의 경우 시, 구, 동 등으로 분해가 가능하므로 복합 속성이라고 볼 수 있다.

4. ④

두 개의 엔터티 사이 관계 도출시 업무기술서, 장표에 관계연결을 가능하게 하는 동사가 있는지를 고려해야 한다.

5. ④

주식별자의 특징은 유일성, 최소성, 불변성, 존재성 이다.

6. ③

이행적 종속을 없애도록 테이블을 분리하는 단계는 제 3 정규화이다.

7. ③

관계를 맺는다는 의미는 부모의 식별자를 자식에 상속하고, 상속된 속성을 매핑키(조인키)로 활용하는 것이다.

8. ③

순서대로 원자성, 일관성, 고립성, 지속성을 나타내는 표현이다.

9. ①

COUNT는 0이 리턴된다.

10. ③

본질식별자가 복잡한 구성을 가질 때 주로 인조식별자를 생성한다.

11. ①

개체 무결성에 대한 설명이다. 참조 무결성은 왜래키 값이 NULL이거나 참조 테이블의 기본키 값과 동일해야 하는 특성이고, 도메인 무결성은 주어진 속성 값이 정의된 도메인에 속한 값이어야 하는 특성을 말한다. NULL 무결성은 특정 속성에 대해 NULL을 허용하지 않는 특징을 의미한다.

12. ②

도메인은 엔터티에서의 각 속성에 허용되는 값의 범위를 의미한다.

13. ③

COMMIT, ROLLBACK은 TCL로 분류되고 GRANTM REVOKE는 DCL로 분류된다.

14. ②

컬럼 별칭 정의 시 _를 제외한 특수기호를 포함하는 경우 쌍따옴표로 묶어서 전달해야 한다. 또한, 컬럼 별칭에 공백(띄어쓰기)을 포함하는 경우 쌍따옴표로 묶어서 전달해야 한다.

15. ①

2024/08/24 10:00:00 날짜로부터 10일 이전, 3개월 이전은 2024/05/14 10:00:00 이다.
10/24/60는 10분을 의미하므로
10분을 더하면 2024/05/14 10:10:00 이 된다.

16. ④

①, ②, ③ 모두 2024/05/01 이 출력되며, ④는 5월의 마지막 날인 2024/05/31이 리턴된다.

17. ④

RTRIM('ABCAA', 'A') 는 오른쪽에서부터 A를 연속적으로 지우고 지울 대상이 아닌 문자를 만나면 더 이상 글자를 지우지 않는다.

18. ③

NULLIF는 두 값이 같으면 NULL, 다르면 첫 번째 인수값을 리턴하는 함수이다.

19. ①

특정 대상의 수를 세려면 COUNT는 세고자 하는 값이 어떤 값이든 존재(1이나 'X' 등)하면 되고, 반대로 세지 않을 값은 반드시 NULL이어야 한다. SUM은 세고자 하는 값이 1이고, 나머지는 NULL이거나 0이면 된다. 따라서 해당 조건에 만족하는 문장은 1번이다.

20. ③

두 번째 글자가 L이면서 그 뒤에 E를 하나 포함하는 이름을 제외하면 SMITH, KING, CLARK 가 출력된다.

21. ②

2번 문장은 연산자 우선순위(NOT > AND > OR)에 의해 (DEPTNO = 20 AND JOB = 'CLERK') 조건이 먼저 수행되므로 다른 문장들과 연산 순서가 달라진다.

22. ②

GROUP BY 뒤의 컬럼 순서는 그룹을 결정하는데 영향을 주지 않는다. 결과적으로 GROUP BY에 나열되는 컬럼들의 값들이 모두 같으면 하나의 그룹이 되기 때문에 GROUP BY A, B 나 GROUP BY B,A이나 두 컬럼의 값이 같은 그룹은 동일하다.

23. ①

ORDER BY 절을 사용하지 않으면 기본적으로 테이블에 입력된 행 순서대로 출력된다.

24. ②

SELECT절을 보면 E2.ENAME이 매니저이름이다. 따라서 E1의 MGR(매니저번호)을 갖는 E2에서의 EMPNO가 E1 사원의 매니저라고 볼 수 있다. 또한, 매니저가 없는 MARTIN도 출력을 하기 위해서는 LEFT OUTER JOIN이 필요하므로 오라클 표준으로 E1의 반대쪽 컬럼에 (+) 기호를 붙여주면 된다.

25. ③

NATURAL JOIN은 컬럼명이 같은 컬럼끼리 값이 같은 조인을 완성한다. 따라서 양쪽 테이블의 COL2의 값이 같은 대상끼리 연결되는데, TAB1의 COL2가 10인 경우 1개, 20인 경우 2개, 30인 경우는 생략되며, NULL은 같다고 볼 수 없으므로 생략된다. 따라서 최종 출력되는 건수는 1 + 2+ 1 = 4건이다.

26. ③

TAB1과 LEFT OUTER JOIN인 테이블은 TAB3이므로 TAB1과 TAB3의 관계에만 TAB3의 컬럼에 (+)를 붙여주면 된다.

27. ①

왼쪽에 배치된 학생 테이블 기준, 지도 교수가 없어도 교수 정보를 NULL로 출력하려면 LEFT OUTER JOIN이 수행되어야 한다.

28. ④

N개 테이블 조인 시 최소 N-1개의 조인 조건이 필요하다.

29. ③

COUNT(*)는 전체 컬럼의 값이 NULL일 경우도 항상 전체 행의 수를 출력한다. COL1+COL2+COL3은 각 행마다 NULL을 하나 이상 포함하므로 모든 NULL을 포함한 산술연산의 결과는 NULL이 리턴된다.

30. ④

순수 관계 연산자는 릴레이션의 구조와 특성을 이용하는 연산자로 SELECT, PROJECT, JOIN, DIVISION이 있다.

31. ④

TAB2에서 COL1의 값이 NULL이 아닌 값은 A와 C이다. TAB1에서 이들을 제외한 COL2의 값은 20과 40이므로 총 합은 60이다.

32. ③

ALL은 작다와 만나면 값들 중 최솟값을 리턴한다. 따라서 TAB1에서 COL1의 값이 35보다 작거나 같은 값은 10,20,30이므로 총 합은 60이 된다.

33. ④

다중컬럼 서브쿼리는 메인쿼리의 FROM 절이 아닌 WHERE 절에서 비교 상수로서 사용되므로 메인쿼리에서 사용할 수 없다.

34. ④

각 직원의 상위관리자명을 출력하기 위한 스칼라 서브쿼리를 찾는 문제이다. 각 직원의 MGR은 상위관리자 번호이므로 이 번호를 사번으로 가지고 있는 자가 상위관리자가 된다. 또한, NVL의 경우 스칼라 서브쿼리 밖에서 실행해야 스칼라 서브쿼리 결과가 NULL일 때 치환할 수 있다.

35. ②

보기 2번의 경우 각 행마다 COL1의 값과 같지 않은 값을 COL2에 선택하기 때문에 TAB1의 B, C, D의 TAB2의 두 행이 모두 선택되어 출력된다.

36. ①

DEPTNO별, DEPTNO + JOB별, 전체 소계를 출력하는 함수는 ROLLUP(DEPTNO, JOB) 이다.

37. ③

A, B 그룹별 COL2 순서대로 누적합을 구하면 되는데, COL2의 값이 같으면 하나의 그룹으로 묶여 함께 계산된다. 따라서 A 그룹은 10, 50, 50, B 그룹은 30, 110, 110이 리턴된다.

38. ②

RANK는 동일한 값을 갖는 경우 동일한 순위를 부여하지만 그 다음 순위가 앞의 동일한 순위의 개수만큼 밀리게 된다. 따라서 10번 부서의 2등은 MILLER, CLARK이며, 20번 부서의 2등은 존재하지 않는다. 따라서 총 SAL의 합은 4000이 된다.

39. ②

TOP(3) WITH TIES 은 성적이 높은 순서대로 3명을 출력하는데, 마지막 순위에 동순위가 있다면 추가 출력된다. 현재는 98점이 1등, 80점 두 명이 공동 2등이므로 이들 셋만 출력된다.

40. ③

셀프조인을 사용하여 각 직원의 상위관리자 급여보다 높은 급여를 받는 직원을 출력한 뒤 각 직원의 상위관리자 급여의 총 합을 리턴하는 문장이다. ALLEN의 급여가 상위관리자인 SMITH의 급여보다 크므로 SMITH의 급여인 800과 JONES의 급여가 상위관리자인 ALLEN 급여보다 높기 때문에 ALLEN의 급여인 1600을 더하면 2400이다.

41. ④

같은 레벨일 경우 추가 정렬을 원한다면 ORDER SIBLINGS BY 뒤에 정렬 컬럼을 전달한다.

42. ①

1번 문장은 FROM절의 서브쿼리 결과 중 PIVOT절에 사용되지 않은 컬럼의 값이 출력되어 CUSTOMER_ID 별로 금융업, 농업인, 종합소매업 종사자 수를 세게 된다.

43. ②

AB|C+ 는 "AB" 또는 C가 1회 이상 연속된 문자열을 나타내므로 이를 모두 삭제하면 A B BBB만 남게된다.

44. ④

\w|\d+에서 + 범위는 \d에만 적용된다. 따라서 한 글자의 단어 또는 숫자의 연속을 나타내므로 H 만 출력된다.

45. ④

MERGE문에 의해 TAB1.NO = TAB2.NO 조건이 만족하면 먼저 UPDATE한 후 UPDATE 결과에서 DELETE 조건이 만족할 경우 값이 삭제된다.

46. ④

ROLLBACK TO SAVE1은 결과적으로 에러가 발생하여 무시된다. 따라서 COMMIT 이전 트랜잭션은 모두 반영되므로 4번이 정답이다.

47. ①

TRUNCATE는 테이블 구조는 남기고 데이터를 전체 삭제하며 즉시 반영되므로 DDL에 속한다.

48. ④

4번 문장은 NOT NULL과 DEFAULT의 순서를 서로 변경해야 정상 처리된다.

49. ③

VARCHAR와 CHAR 타입간의 변경은 데이터 존재 유무와 상관없이 변경 가능하다.

50. ④

뷰의 정의를 변경할 수 없고, 뷰를 통한 원본 테이블 데이터를 변경할 수 있느나 여러 제한이 생긴다. 뷰에는 인덱스를 직접 생성할 수 없고, 조인이나 서브쿼리를 사용한 복잡한 쿼리를 갖는 뷰는 생성이 가능하다.

6회차 답안 및 해설

1. ①

모델링의 3단계는 개념적, 논리적, 물리적 모델링이 있으며, 이들 중 세부 속성 및 식별자를 정의하는 단계는 논리적 모델링 단계이다.

2. ④

중심엔터티는 기본엔터티로부터 파생되는 형태의 엔터티이다. 주문, 청구, 계약, 매출 등이 이에 속한다.
고객은 기본엔터티이다.

3. ②

한 개의 엔터티는 두 개 이상의 속성을 가져야 엔터티의 기본 조건이 성립된다.

4. ③

관계란 엔터티들간의 연관성을 나타내는 개념이며, 한 엔터티의 레코드가 다른 엔터티의 레코드와 어떻게 연결되는지를 표현한 것은 관계의 차수를 말한다.

5. ②

유일성과 최소성을 만족하는 여러 키 중 하나가 기본키가 되고, 나머지가 대체키가 된다.

6. ②

정규화는 논리모델링 단계에서 진행한다.

7. ③

고객이 여러 계좌를 소유할 수 있다면 두 엔터티는 1 대 N 관계를 갖는다.

8. ④

IE표기법에는 원을 사용하여 필수적, 선택적 관계를 구분한다.
선택적 관계에는 관계선 끝에 원을 그리고, 필수적 관계에는 원을 그리지 않는다.

9. ④

NULL로만 구성된 집합의 COUNT 결과는 0이다.

10. ②

보조식별자에 대한 설명이다.

11. ③

테이블 생성 시 소유자를 명시하지 않으면 CREATE TABLE을 실행하는 유저 소유의 테이블이 된다. 자신 소유의 테이블의 SELECT, INSERT, UPDATE, DELETE 권한은 다른 사용자에게 언제나 부여 가능하다.

12. ④

DCL(Data Control Language)에는 GRANT, REVOKE 가 포함된다. ROLLBACK은 TCL 이다.

13. ③

컬럼 별칭은 AS를 사용하거나 생략 가능하고, 오라클의 경우 테이블 별칭은 AS를 사용할 수 없다.

14. ③

TO_DATE('2024','YYYY') 함수의 결과는 현재 날짜의 월이 반영되어 '2024/11/01'이 리턴된다.

15. ④

NVL(COL1,20), ISNULL(COL1,20)은 모두 COL1의 값이 NULL인 경우만 20으로 치환된다.
COALESCE(COL1,COL2)는 COL1의 값이 널이 아니면 COL1값을, NULL인 경우 COL2의 값을 반환한다.
DECODE(COL1,NULL,20)는 COL1의 값이 널이면 20을 리턴하지만, 널이 아닌 값은 모두 널로 변환된다.

16. ③

CEIL(-2.7)은 -2.7보다 크면서 가장 작은 정수를 리턴한다. 즉 올림값을 반환하므로 -2.7을 올리면 -3이 아닌 -2가 된다.

17. ①

SUBSTR('SQL SERVER TOOL',4,5) 결과는 네번째에 위치한 공백부터 5자를 추출하므로 ' SERV'이 리턴된다.

18. ③

SUM과 같은 형태의 컬럼별칭도 사용가능하나 가급적 사용하지 않는 것이 좋다. WHERE절에는 집계함수를 허용하지 않기 때문에 WHERE절은 수행 불가하다.

19. ②

COL3 != 30 조건에 만족하는 행은 2, 3번 행이다. 따라서 (10+2)와 (NULL+1)를 먼저 연산하고 SUM을 하면 12와 NULL을 합하게 되는데, SUM은 NULL을 무시하므로 12가 리턴된다.

20. ③

GROUP BY를 수행한 그룹은 NULL을 포함한다. 따라서 COL2의 수가 2개인 그룹은 C 와 NULL그룹이 출력된다.

21. ②

① GROUP BY절에 정의하지 않은 컬럼은 SELECT절에 집계함수 없이 사용이 불가하다.

③ GROUP BY절에 정의하지 않은 컬럼으로 정렬할 수 없다.

④ WHERE절에 컬럼 별칭을 사용할 수 없다.

22. ③

INSTR 함수를 사용하여 '-'의 위치보다 1 작은 값을 구한 뒤, 그만큼을 TEL의 처음부터 추출하면 전화번호의 앞자리가 추출된다. 문자 값이므로 문자정렬 기준대로 1229 < 234 < 2569 < 567 순으로 정렬된다.

23. ①

후기가 20미만인 상품('< 20')은 총 2개, 20개 이상 100개 미만('< 100')인 경우 4, 100개 이상('>= 100')이면 1이다. ORDER BY절에서 정렬 순서는 후기 그룹을 1, 2, 3으로 매핑 하였기에 오름차순으로 정렬하면 < 20 -> < 100 -> >=100 순서로 정렬된다.

24. ④

CROSS JOIN은 조인컬럼의 값과 상관없이 두 테이블의 발생 가능한 모든 연결 조합을 출력하므로 총 4*4, 16건이 출력된다.

25. ②

TAB1 기준 LEFT OUTER JOIN시에는 TAB1 데이터는 생략없이 모두 출력된다.

따라서 조인 결과는 다음과 같다.

TAB1.COL1	TAB2.COL2	TAB2.COL1	TAB2.COL2
A	10	A	20
B	20	B	20
NULL	30	NULL	NULL
D	20	NULL	NULL

26. ④

상품을 받을 수 없는 홍길동도 출력되어야 하므로 고객 테이블 기준으로 아우터 조인을 수행해야 하므로 최소포인트와 최대포인트 컬럼 모두 (+)가 붙어야 한다.

27. ①

② FULL OUTER JOIN은 두 집합의 UNION 결과와 같다.

③ USING절에는 괄호를 생략할 수 없다.

④ ORACLE은 FROM절 나열 순서에 따라 조인 결과가 달라지지 않는다.

28. ③

서브쿼리를 사용하여 SEDAN의 CAR_ID를 확인 후 해당 차에 대한 2024년 07월 렌트 기록 수를 묻고 있으므로 총 2건이다.

29. ④

NOT EXISTS는 서브쿼리의 조건이 참인 경우 생략, 거짓인 경우를 출력하는데 TAB2에 COL1에 포함되는 경우는 A, C 이므로 이를 제외한 B, E, NULL이 출력된다.

30. ④

MIN(COL1)은 'AB'이므로 COL1 LIKE '%AB%'이 수행된다. 따라서 AB, ABC, CAB 총 3건이 출력된다.

31. ④

WHERE절에 사용된 서브쿼리 내 출력값은 메인쿼리로 전달될 수 없다.

32. ②

UPDATE 쿼리는 상사보다 COMM이 높은 경우 부하직원과 상사직원의 COMM의 중간값으로 수정하는 문장이다. 따라서 SMITH와 WARD만 각각 150 250으로 UPDATE 되며 나머지는 조건에 만족하지 않아 NULL로 UPDATE된다.

33. ②

다중 컬럼 서브쿼리는 대소비교 연산에 대한 처리가 불가능하다.

34. ②

ANY는 작다와 만나면 서브쿼리 결과중 최댓값을 리턴한다. 각 COL1의 값이 일치하면서 TAB2의 COL2값 중 최댓값보다 작은 조건을 만족하는 TAB1의 대상은 A와 D이다.

35. ④

집합 연산자 사용 시 각 쿼리에 정렬을 수행할 수 없다. 마지막에 사용된 ORDER BY절은 전체 수행 결과에 대한 정렬 결과를 리턴한다.

36. ①

DNAME별, JOB별, (DEPTNO,JOB)별, 전체 소계를 출력한 결과이므로 CUBE(DNAME, JOB)가 적절하다.
CUBE는 NULL을 사용하지 않아도 전체 소계가 출력된다.
ROLLUP(DNAME, JOB)은 DNAME별, (DNAME,JOB)별, 전체 소계만 출력되며,
GROUPING SETS(DNAME,JOB, NULL)은 나열된 집합인 DNAME별, JOB별, 그리고 전체 소계가 출력된다.

37. ①

같은 지점 내 지난달보다 실적이 감소한 달의 실적금액의 총 합을 구하는 문제이므로 A지점은 2월의 1000, B지점은 3월의 2000이 리턴된다.

38. ④

RANK는 동 순위 뒤의 순위가 연속적이지 않으므로 순위1의 결과는 1,1,3,4,1,2,2이다.
DENSE_RANK는 동 순위 뒤의 순위가 연속적이므로 순위2의 결과는 1,1,2,3,1,2,2이다.
ROW_NUMBER는 크기 상관없이 연속적인 숫자를 중복 없이 출력하므로 1,2,3,4,1,2,3이 출력된다.

39. ②

고객번호별로 날짜가 작은순서대로 포인트 누적합을 구하지만, 누적합 범위는 이전행과 현재행이다. 따라서 두번째 행은 500과 300의 합인 800이, 세번째 행은 300과 300의 합인 600이 나와야 한다.

40. ①

동순위가 존재하지만 성적이 높은순으로 2개의 행만 선택하였으므로 WITH TIES없이 TOP(2)를 수행해야 한다.

41. ③

START WITH PART = 10 조건으로 인해 공과대학부터 시작하여 하위 계층을 출력하는 문제이다.
CONNECT BY PRIOR D.DEPTNO = D.PART 조건에 의해 공과대학의 DEPTNO(10) 값을 PART로 갖는 행을 찾아 레벨 2를 부여한다. 즉, 컴퓨터정보학부와 메카트로닉스학부 가 레벨 2가 된다.

42. ④

같은 클래스내 S1.점수보다 높은 점수를 갖는 경우만 매칭되어 출력되고 있다. 또한, 각 클래스 내 성적이 가장 높은 사람도 출력하기 위해서는 아우터 조인 수행이 필요하다.

43. ③

교차표 형태로의 출력은 PIVOT을 통해 가능하다. 이때, FOR에는 최종 출력될 컬럼을 표현하는 자리이므로 성별이 적당하다.

44. ④

대소를 구분하지 않고 영문으로만 구성된 단어를 처음부터 찾아 3번째 발견되는 문자열을 추출하는것이므로 command가 출력된다.

45. ④

A로 시작하거나 A로 끝나되, B와 A사이에 한자 이상의 글자(문자, 공백, 숫자)를 포함하는 문자열을 가진 ID는 A12345A, AA, B00A 이다.

46. ②

다중컬럼 UPDATE는 반드시 서브쿼리를 통해서만 가능하다.

47. ②

ROLLBACK TO AAA에 의해 DELETE문과 그 뒤에 수행된 INSERT문이 실행 취소된다.
이어서 ROLLBACK을 또 수행하므로 마지막 COMMIT 이후 모든 트랜잭션은 롤백된다.
따라서 COMMIT 이후 INSERT문까지 취소되어 UPDATE만 반영된다.

48. ①

DEFAULT 값 선언 시 기존 데이터는 변경되지 않는다. 이후 입력된 값에 대해서도 NULL이 입력될 경우는NULL이 입력되고 DEFAULT값이 입력되지 않는다.

49. ③

부모테이블을 삭제하려면 CASCADE CONSTRAINTS로 외래키를 함께 삭제하거나, 자식테이블을 먼저 삭제하고 부모테이블을 삭제해야한다.

50. ①

본인 소유 테이블에 대해서는 언제나 권한 부여가 가능하다. SYSTEM 계정에서 테이블을 생성하더라도 소유자를 지정할 수 있다. 또한 동시에 여러 테이블에 대해 권한을 부여할 수 없지만 같은 권한을 여러 사용자에게 동시에 부여할 수 있다.