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 않은 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 다른 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 다른 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?

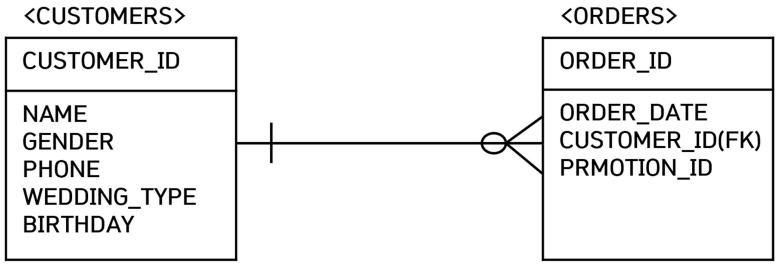
<TAB1>

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 않은 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.

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, 50~99 이면 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.