데이터베이스프로그래밍 2 차퀴즈 문제

2025/1 학기 담당교수 공학박사 장석구

7.1 변수

7.1-1 PL/SQL 에서 변수 선언 시 사용되는 올바른 구문은? ★★☆☆☆

① NUMBER v_num;
② v_num := NUMBER;
③ DECLARE v_num := 10;
④ v_num NUMBER := 10;

4번

7.1-3 다음 중 PL/SQL 에서 변수 선언 시 발생할 수 있는 오류로 잘못 설명된 것은? ★★★☆☆

① 변수명은 대소문자를 구분하지 않으므로 중복 선언이 불가하다.
② NOT NULL 제약이 있는 변수는 반드시 초깃값을 지정해야 한다.
③ 변수명에 큰따옴표("")를 사용하면 대소문자를 구분할 수 있다.
④ := 연산자는 논리 비교를 위한 연산자이다.

4번

7.1-4 다음 중 PL/SQL 에서 변수의 초기값을 지정하지 않아도 오류가 발생하지 않는 경우는? ★★★☆☆

① v_total NUMBER NOT NULL;
② v_flag BOOLEAN;
③ v_name VARCHAR2(10) NOT NULL;
④ v_code VARCHAR2(5) NOT NULL;

2번

7.1-5 다음 중 PL/SQL 에서 변수에 값을 할당할 때 발생할 수 있는 오류에 대한 설명으로 옳지 않은 것은? ★★★★☆

① 변수의 크기를 초과하는 값을 대입하면 오류가 발생한다.
② := 연산자 대신 = 연산자를 사용해도 값이 할당된다.
③ NOT NULL 제약을 가진 변수에 NULL 을 대입하면 오류가 발생한다.
④ 변수 선언 없이 값을 할당하면 컴파일 오류가 발생한다.

2번

7.1-6 다음 중 리터럴에 대한 설명으로 가장 적절한 것은? ★★☆☆☆

① 리터럴은 값이 저장된 메모리 위치를 참조한다.
② 리터럴은 고정된 메모리 블록으로 관리된다.
③ 리터럴은 변수처럼 값이 변경될 수 있다.
④ 리터럴은 그 자체로 값을 의미하며 저장소와는 관계가 없다.

4번

7.2 상수

7.2-7 다음 중 PL/SQL 에서 상수를 선언하는 올바른 형식은? ★★☆☆☆

① c_rate NUMBER CONSTANT := 0.05;
② c_rate CONSTANT := 0.05;
③ c_rate := CONSTANT NUMBER 0.05;
④ CONSTANT c_rate NUMBER := 0.05;

1번

7.2-8 다음 중 PL/SQL 에서 상수에 대한 설명으로 틀린 것은? ★★★☆☆

① 상수는 선언과 동시에 반드시 초기값을 지정해야 한다.
② 상수는 값을 한 번만 할당할 수 있다.
③ 상수는 값을 변경할 수 없다.
④ **상수는 BEGIN 블록 내에서만 선언할 수 있다.**

4번

7.2-9 PL/SQL 에서 상수를 선언하고 이후 값을 변경하려 할 때 발생하는 오류로 올바른 것은? ★★★★☆

① ORA-06550: syntax error in declaration
② ORA-00942: table or view does not exist

③ ORA-06550: PLS-00363: 식은 피할당자로 사용할 수 없습니다 3번
④ ORA-06502: numeric or value error

7.2-10 다음 코드 실행 시 발생하는 결과로 옳은 것은? ★★★☆☆

```
DECLARE
  c_tax CONSTANT NUMBER := 0.1;
BEGIN
  c_tax := 0.2;
END;
```

2번
① 정상 실행된다.
② 컴파일 오류가 발생한다.
③ 런타임 시 NULL 값이 출력된다.
④ BEGIN 블록에서만 상수 값을 변경할 수 있다.

7.2-11 PL/SQL 상수 선언 시 반드시 포함되어야 하는 요소는? ★★☆☆☆

① DEFAULT 절
② NULL 허용 선언
③ := 연산자를 이용한 초기값 지정 3번
④ 주석 처리

문제해설(③): 상수는 선언 시 반드시 := 연산자를 사용하여 초기값을 지정해야 하며, 이를 생략할 경우 컴파일 오류가 발생한다.
① DEFAULT 는 사용할 수 있으나 필수 아님
② NULL 허용 여부는 상수와 무관
④ 주석은 가독성을 위한 요소이지 필수 아님

7.3 리터럴

7.3.1 문자형 리터럴

7.3.1-12 다음 중 리터럴에 대한 설명으로 가장 적절한 것은? ★★☆☆☆

① 리터럴은 선언 후 값의 변경이 가능한 저장소이다.
② 리터럴은 프로그램 내에서 상수 이름으로 사용된다. 4번
③ 리터럴은 변수나 상수에 저장되는 메모리 위치를 참조한다.
④ 리터럴은 소스 코드에 직접 표현된 고정된 값이다.

7.3.1-13 다음 PL/SQL 코드에서 사용된 리터럴이 아닌 것은 무엇인가? ★★★☆☆

```
DECLARE
  c_pi CONSTANT NUMBER := 3.14;
  v_text VARCHAR2(50) := 'Circle Area';
BEGIN
  DBMS_OUTPUT.PUT_LINE(v_text);
END;
```

3번(4번도 답 아닌가?)
① 3.14
② 'Circle Area'
③ c_pi
④ 'v_text'

7.3.1-14 다음 중 리터럴과 관련된 오라클 PL/SQL 규칙에 대한 설명으로 옳지 않은 것은? ★★★☆☆

① 작은따옴표로 감싼 문자열은 문자형 리터럴이다.
② 문자형 리터럴은 줄 바꿈을 포함할 수 있다.
③ 리터럴 NULL 은 데이터 타입에 따라 의미가 다를 수 있다. 4번
④ '0' ~ '9' 문자형 리터럴은 수치형 리터럴과 호환된다.

7.3.1-15 PL/SQL 에서 문자형 리터럴을 표현하는 방법으로 부적절한 것은? ★★★☆☆

① 'Scott''s cat'
② Q'[Scott's cat]'
③ "Scott's cat"
④ Q'|Scott's cat|'

3번

7.3.1-16 PL/SQL 에서 다음 구문에 대한 설명 중 옳지 않은 것은? ★★★★☆

```
DECLARE
  v_msg VARCHAR2(1000);
BEGIN
  v_msg := Q'{This is John's message}';
  DBMS_OUTPUT.PUT_LINE(v_msg);
END;
```

① 접두어 Q 는 긴 문자열 리터럴을 쉽게 표현할 수 있다.
② 시작 구분자 {가 사용되었기 때문에 끝 구분자는 반드시 }이다.
③ Q 접두어를 사용할 때 작은따옴표를 두 번 반복할 필요가 없다.
④ Q 접두어는 반드시 대문자 Q 를 사용해야 한다.

4번

7.3.2 수치형 리터럴

7.3.2-17 다음 중 NUMBER 형 리터럴에 해당하는 것은? ★★☆☆☆

① 3.14F
② 2.71D
③ +100
④ -5F

3번

7.3.2-18 다음 중 FLOAT 형 리터럴의 예로 적절하지 않은 것은? ★★☆☆☆

① 1.0F
② +2F
③ -3.14f
④ 0.5

4번

7.3.2-19 다음 중 부동 소수점 리터럴에 대한 설명으로 틀린 것은? ★★★☆☆

① FLOAT 형은 4 바이트 IEEE 754 형식으로 저장된다.
② DOUBLE 형은 NUMBER 형보다 더 정밀하다.
③ FLOAT 형 리터럴은 정밀도가 낮아 오차가 발생할 수 있다.
④ DOUBLE 형 리터럴은 숫자 뒤에 D 또는 d 를 붙여 표현한다.

2번

7.3.2-20 다음 중 부동 소수점 리터럴에 대한 설명으로 알맞은 것은? ★★★☆☆

① 부동 소수점 리터럴은 항상 정확한 값을 저장한다.
② FLOAT 형 리터럴은 NUMBER 형 변수에 대입하면 정밀도가 보존된다.
③ FLOAT 형 리터럴은 약간의 오차를 포함할 수 있다.
④ NUMBER 형 리터럴은 IEEE 754 포맷으로 저장된다.

3번

7.3.2-21 다음 중 TO_CHAR(9.95F, '99.0')의 결과로 옳은 것은? ★★★★☆

① 10.0
② 9.95
③ 9.9
④ 9.94999981

3번

7.3.3-날짜형 리터럴

7.3.3-22 다음 중 ANSI 표준 날짜형 리터럴의 올바른 형식은? ★★☆☆☆

① TO_DATE('2024-01-01', 'YYYY-MM-DD')
② DATE'2024/01/01'
③ DATE'2024-01-01'
④ TIMESTAMP'2024-01-01'

3번

7.3.3-23 다음 중 TIMESTAMP 리터럴의 ANSI 형식으로 올바른 것은? ★★☆☆☆

① TIMESTAMP'2024-01-01 14:30:00'
② TO_TIMESTAMP('2024-01-01 14:30:00', 'YYYY-MM-DD HH24:MI:SS')
③ TIMESTAMP'2024-01-01T14:30:00'
④ TIMESTAMP'2024-01-01'

1번

7.3.3-24 다음 중 TIMESTAMP WITH TIME ZONE 형식의 올바른 ANSI 표준 리터럴은? ★★★☆☆

① TIMESTAMP'2024-01-01 12:00:00+09:00'
② TO_TIMESTAMP_TZ('2024-01-01 12:00:00 +09:00', 'YYYY-MM-DD HH24:MI:SS TZH:TZM')
③ TIMESTAMP WITH TIME ZONE '2024-01-01 12:00:00 +09:00'
④ TIMESTAMP'2024-01-01 12:00:00 +09:00'

4번

7.3.3-25 다음 중 INTERVAL YEAR TO MONTH 형식 리터럴로 옳지 않은 것은? ★★★☆☆

① INTERVAL '123-4' YEAR TO MONTH
② INTERVAL '123' YEAR
③ INTERVAL '50' MONTH
④ INTERVAL '12345' YEAR

4번

7.3.3-26 다음 중 INTERVAL DAY TO SECOND(3) 형식의 리터럴 표현으로 올바른 것은? ★★☆☆☆

① INTERVAL '04:05:12' DAY TO SECOND(3)
② INTERVAL '4 5:12:10.222' DAY TO SECOND(3)
③ INTERVAL '4.5 12:10' DAY TO SECOND
④ INTERVAL '5:12:10' SECOND TO MINUTE(3)

2번

8.1 연산자와 피연산자

8.1-1 PL/SQL 에서 표현식(Expression)에 대한 설명으로 가장 적절한 것은? ★★☆☆☆

① 연산자만으로 구성된 연산 실행 단위
② 조건에 따라 분기되는 논리 제어문
③ 하나 이상의 값, 상수, 변수, 연산자, 함수 등이 조합되어 하나의 결과를 생성하는 구조
④ 피연산자를 생략해도 결과가 도출되는 계산식

3번

8.1-2 다음 중 PL/SQL 표현식의 '연산자(operator)'에 해당하는 것은? ★★☆☆☆

① 변수, 상수, 리터럴
② BETWEEN, IS NULL, NOT
③ 숫자 리터럴, 함수 호출 결과
④ 문자열, 날짜, 정수

2번

8.1-3 다음 중 '단항 연산자'에 해당하지 않는 것은 무엇인가요? ★★☆☆☆

① + (부호 연산자)
② IS NULL
③ NOT
④ AND

4번

8.1-4 다음 중 표현식에서 '피연산자(operand)'가 될 수 없는 것은 무엇인가요? ★★★☆☆

① 숫자 리터럴
② 함수 호출 결과

4번

③ 변수
④ 연산자

8.2 연산자우선순위

8.2-5 다음 중 PL/SQL 연산자 우선순위가 가장 높은 것은 무엇인가요? ★★★☆☆

① 곱셈(*)
② 비교 연산자(=, <>, >= 등)
③ 지수 연산자(**)
④ 논리합(OR)

3번

8.2-6 다음 식의 계산 결과로 올바른 계산 순서를 고르세요: 2 + 3 * 4 - 5 ★★☆☆☆

① (2 + 3) * (4 - 5)
② ((2 + 3) * 4) - 5
③ 2 + (3 * 4) - 5
④ 2 + 3 * (4 - 5)

3번

8.2-7 다음 중 연산자 우선순위가 낮은 쪽부터 나열된 것은? ★★★☆☆

① ** → AND → +
② := → OR → NOT
③ * → + → =
④ IS NULL → AND → :=

2번

8.2-8 다음 중 '+' 연산자보다 우선순위가 가장 높은 연산자는 무엇인가요? ★★★☆☆

① *
② **
③ :=
④ +

2번

8.2-9 PL/SQL 에서 연산자 우선순위가 동일한 연산자끼리 묶인 것은? ★★☆☆☆

① +, -, *
② =, <, >, BETWEEN
③ OR, AND
④ :=, IS NULL

2번

8.3 연산자의 기능 설명

8.3-10 다음 중 문자열을 연결하는 데 사용하는 PL/SQL 연산자는 무엇인가요? ★☆☆☆☆(단)

① +
② &
③ ||
④ :=

3번

8.3-11 다음 중 v_Name IS NULL 조건이 의미하는 것은? ★☆☆☆☆

① v_Name 값이 공백("")인지 검사한다.
② v_Name 값이 'NULL' 문자열인지 검사한다.
③ v_Name 값이 존재하지 않음을 검사한다.
④ v_Name 값이 0 인지 검사한다.

3번

8.3-12 다음 중 두 값이 다름을 검사하는 연산자가 아닌 것은? ★★☆☆☆

① !=
② <>

4번

③ ~=
④ :=

8.3-13 다음 중 LIKE 연산자의 설명으로 가장 적절한 것은? ★★★☆☆

① 두 수치 값이 같은지를 비교한다.
② NULL 여부를 검사한다.
③ 지정한 패턴과 문자열이 일치하는지 검사한다.
④ 값이 특정 범위 안에 있는지 검사한다.

3번

8.3-14 다음 중 v_Salary BETWEEN 3000 AND 5000 의 의미와 동일한 표현은? ★★★☆☆

① v_Salary < 3000 OR v_Salary > 5000
② v_Salary >= 3000 AND v_Salary <= 5000
③ v_Salary = 3000 OR v_Salary = 5000
④ v_Salary IN (3000, 5000)

2번

8.4 연산자의 종류

8.4.1 산술연산자

8.4.1-15 산술 연산자에 해당하지 않는 것은? ★☆☆☆☆

① +
② AND
③ -
④ *

2번

8.4.2 논리연산자

8.4.2-16 논리 연산자 중 'NOT x'가 TRUE 를 반환하는 경우는? ★☆☆☆☆

① x 가 TRUE 일 때
② x 가 FALSE 일 때
③ x 가 NULL 일 때
④ x 가 TRUE 또는 FALSE 일 때 모두

2번

8.4.2-17 다음 중 논리 연산자 AND 의 결과로 TRUE 가 되는 경우는? ★★☆☆☆

① 두 피연산자 중 하나가 TRUE 일 때
② 두 피연산자 모두 TRUE 일 때
③ 두 피연산자 모두 FALSE 일 때
④ 두 피연산자 중 하나가 NULL 일 때

2번

8.4.2-18 다음 중 '논리 연산자에서 NULL 처리 규칙'으로 올바른 설명은? ★★★★☆

① NULL 과의 모든 연산 결과는 NULL 이다.
② 논리 연산에서 NULL 은 TRUE 로 간주한다.
③ 논리 연산자 AND, OR 는 NULL 을 특별히 다르게 처리한다.
④ NOT 연산자는 NULL 을 TRUE 로 변환한다.

1번

8.4.3 Short-Circuit Evaluation

8.4.3-19 Short-Circuit Evaluation 의 정의로 옳은 것은? ★★☆☆☆

① 모든 논리 연산식을 끝까지 계산하는 방식
② 앞 연산 결과로 전체 결과가 결정되면 뒤 연산을 평가하지 않는 방식
③ 산술 연산에서 나눗셈 0 예외를 방지하는 기법
④ 연산자를 줄여서 코드를 간략하게 만드는 최적화

2번

8.4.3-20 다음 중 Short-Circuit Evaluation 의 예외 상황으로 옳은 것은? ★★★☆☆

① 모든 연산이 평가되어야 예외가 발생한다.
② 앞 연산 결과가 전체 결과를 결정하지 않을 때 예외가 발생할 수 있다.

2번

③ 뒤 연산이 평가되지 않아도 예외가 발생한다.
④ 항상 예외가 발생하지 않는다.

8.4.3-21 Short-Circuit Evaluation 을 고려할 때 개발자가 주의해야 할 점으로 옳은 것은? ★★★★☆

① 모든 조건문의 모든 연산을 무조건 수행하도록 한다.
② 앞 조건 평가에 따라 뒤 연산이 실행되지 않을 수 있음을 고려해야 한다.
③ 예외가 발생할 수 없으므로 예외 처리 코드를 생략해도 된다.
④ 산술 연산자 대신 논리 연산자를 사용한다.

2번

8.4.4 연결연산자

8.4.4-22 연결 연산자(||)의 역할로 올바른 것은? ★★☆☆☆

① 두 숫자를 덧셈하는 연산자
② 두 문자열을 하나로 합치는 연산자
③ 두 논리값을 AND 연산하는 연산자
④ 두 문자열을 비교하는 연산자

2번

8.4.4-23 NULL 값을 연결 연산자(||)로 처리할 때 올바른 설명은? ★★★☆☆

① NULL 과 연결하면 결과도 항상 NULL 이 된다.
② NULL 과 연결해도 NULL 은 무시되어 결과 문자열이 만들어진다.
③ NULL 과 연결하면 예외가 발생한다.
④ NULL 은 빈 문자열로 자동 변환된다.

2번

8.4.4-24 다음 중 Oracle 에서 문자열 연결 방법으로 옳은 것은? ★★☆☆☆

① + 연산자와 CONCAT 함수
② || 연산자와 CONCAT 함수
③ & 연산자와 CONCAT 함수
④ + 연산자와 || 연산자

2번

8.4.4-25 다음 코드 실행 결과로 올바른 것은?

```
BEGIN
  DBMS_OUTPUT.PUT_LINE('Hello, ' || NULL || 'World!');
END;
```

① Hello, NULLWorld!
② Hello, World!
③ Hello, World!NULL
④ NULL

2번

8.4.4-26 다음 중 연결 연산자(||) 사용 시 암묵적 형 변환이 일어나는 경우는? ★★★☆☆

① 두 피연산자가 모두 숫자일 때
② 숫자와 문자열이 연결될 때 숫자가 문자열로 변환되어 연결된다.
③ 두 피연산자가 모두 NULL 일 때
④ 문자열과 논리값이 연결될 때

2번

8.4.5 비교연산자

8.4.5-27 다음 중 연결 연산자(||) 사용 시 암묵적 형 변환이 일어나는 경우는? ★★★☆☆

① 두 피연산자가 모두 숫자일 때
② 숫자와 문자열이 연결될 때 숫자가 문자열로 변환되어 연결된다.
③ 두 피연산자가 모두 NULL 일 때
④ 문자열과 논리값이 연결될 때

2번

8.4.5-28 다음 중 PL/SQL 에서 := 연산자의 용도는? ★★☆☆☆

① 논리 연산 수행
② 값의 비교
③ 변수에 값을 대입
④ 데이터베이스에 값 저장

3번

8.4.5-29 다음 중 PL/SQL 에서 예외 처리를 위한 키워드가 아닌 것은? ★★☆☆☆

① EXCEPTION
② WHEN
③ RAISE
④ TRIGGER

4번

8.4.5-30 PL/SQL 에서 커서를 사용하는 주요 이유는 무엇인가? ★★★☆☆

① SQL 문을 컴파일하기 위해
② 테이블을 삭제하기 위해
③ 다중 행 질의를 처리하기 위해
④ 트랜잭션을 커밋하기 위해

3번

8.4.5-31 다음 중 명시적 커서에 해당하는 것은? ★★★☆☆

① 시스템 커서
② SQL%ROWCOUNT
③ 사용자 정의 커서
④ 자동 커서

3번

8.4.5-32 PL/SQL 에서 IF...ELSIF...ELSE 문에서 ELSIF 가 유효하지 않은 경우는? ★★☆☆☆

① ELSIF 뒤에 조건이 없는 경우
② IF 문이 없는 경우
③ ELSE 문이 없는 경우
④ THEN 없이 ELSIF 만 사용할 경우

4번

8.4.5-33 다음 중 PL/SQL 에서 레코드 타입을 선언할 때 사용하는 키워드는? ★★★☆☆

① TYPE
② RECORD
③ ROWTYPE
④ TABLE

2번

8.4.5-34 %TYPE 속성의 주요 용도는 무엇인가? ★★★☆☆

① 변수 초기화
② 변수의 자료형을 자동 지정
③ 테이블 생성
④ 커서 정의

2번

8.4.5-35 다음 중 LOOP 문을 강제로 종료하는 데 사용하는 문장은? ★★☆☆☆

① SKIP
② BREAK
③ CONTINUE
④ EXIT

4번

8.4.5-36 다음 중 %ROWTYPE 속성의 특징으로 옳은 것은? ★★★☆☆

① 테이블 구조가 변경되면 오류 발생
② 특정 열의 자료형만 복사

③ 테이블 전체 행 구조를 참조
④ 자료형은 수동으로 지정해야 함

3번

8.4.6 BOOLEAN 표현식

8.4.6-37 BOOLEAN 표현식의 결과로 나올 수 없는 값은? ★★☆☆☆

① TRUE
② FALSE
③ NULL
④ 0

4번

8.4.6-38 다음 중 BOOLEAN 표현식의 사용 예로 가장 적절한 것은? ★★★☆☆

① 변수에 문자열을 연결할 때
② 루프 조건에서 반복 여부를 판단할 때
③ 레코드 구조를 선언할 때
④ 커서를 열 때

2번

8.4.6-39 다음 중 BOOLEAN 표현식에서 NOT 연산의 결과로 옳은 것은? ★★★☆☆

① NOT TRUE → FALSE
② NOT FALSE → FALSE
③ NOT NULL → TRUE
④ NOT TRUE → NULL

1번

8.4.6-40 PL/SQL 에서 BOOLEAN 표현식이 사용될 수 없는 구문은? ★★★☆☆

① IF 문
② WHILE 문
③ SELECT 문
④ LOOP 문 조건절

3번

8.4.6-41 다음 중 PL/SQL 의 BOOLEAN 데이터 타입에 대한 설명으로 옳지 않은 것은? ★★★☆☆

① TRUE, FALSE, NULL 값을 가질 수 있다.
② AND, OR, NOT 같은 논리 연산자와 함께 사용된다.
③ 출력 시 DBMS_OUTPUT.PUT_LINE 을 그대로 사용할 수 있다.
④ 제어문의 조건 표현식으로 활용할 수 있다.

3번

8.4.7 CASE 표현식

8.4.7-42 다음 중 단순 CASE 표현식의 특징으로 옳은 것은? ★★☆☆☆

① 선택자 없이 조건식들로만 구성된다.
② 여러 조건이 참일 경우 마지막 조건이 적용된다.
③ WHEN 다음에 반드시 논리형(TRUE/FALSE)만 올 수 있다.
④ 선택자와 선택값을 비교하여 일치하는 결과를 선택한다.

4번

8.4.7-43 다음 중 조사 CASE 표현식에 대한 설명으로 틀린 것은? ★★★☆☆

① WHEN 다음에 오는 항목은 조건식이다.
② 선택자가 없어도 사용 가능하다.
③ 조건식은 반드시 TRUE, FALSE, NULL 중 하나를 반환한다.
④ ELSE 절은 생략할 수 없으며 반드시 기술해야 한다.

3번

8.4.7-44 다음 CASE 표현식 실행 결과로 옳은 것은? ★★★☆☆

```
DECLARE
  v_BOOL BOOLEAN := TRUE;
  v_TRUE BOOLEAN := TRUE;
  v_STR STRING(100);
```

```
BEGIN
  v_STR := CASE v_BOOL
           WHEN TRUE THEN 'TRUE1'
           WHEN v_TRUE THEN 'TRUE2'
           ELSE 'OTHER'
          END;
  DBMS_OUTPUT.PUT_LINE(v_STR);
END;
```

4번

① TRUE1
② TRUE2
③ OTHER
④ 실행 오류 발생

8.4.7-45 다음 중 CASE 표현식에 대한 설명으로 옳지 않은 것은? ★★☆☆☆

① CASE 표현식은 값 또는 조건식에 따라 결과를 선택한다.
② 단순 CASE 는 선택자에 대한 값 비교를 한다.
③ 조사 CASE 는 WHEN 다음에 비교할 선택값이 온다.
④ 여러 WHEN 이 참일 경우 먼저 나타난 것이 적용된다.

3번

8.4.7-46 단순 CASE 표현식에서 선택자와 일치하는 선택값이 없고 ELSE 절도 생략된 경우 결과는? ★★★☆☆

① 첫 번째 WHEN 의 결과가 출력된다.
② 마지막 WHEN 의 결과가 출력된다.
③ NULL 이 반환된다.
④ 오류가 발생한다.

3번

8.5 PL/SQL 에서 내장 SQL 함수의 사용

8.5-47 다음 중 PL/SQL 에서 내장 SQL 함수에 대한 설명으로 옳지 않은 것은? ★★☆☆☆

① PL/SQL 에서는 LENGTH, TO_CHAR 같은 단일 행 함수를 사용할 수 있다.
② PL/SQL 에서는 SUM, COUNT 같은 집계 함수를 직접 사용할 수 없다.
③ DECODE 함수는 PL/SQL 에서 직접 사용할 수 있으며 CASE 보다 성능이 좋다.
④ PL/SQL 에서는 CASE 표현식을 이용해 조건 분기를 처리할 수 있다.

3번

8.5-48 다음 중 PL/SQL 에서 DECODE 함수를 사용할 수 있는 방법으로 적절한 것은? ★★☆☆☆

① CASE 문을 이용하여 DECODE 와 동일한 로직을 구현한다.
② DBMS_SQL 패키지로 DECODE 를 직접 호출한다.
③ DECODE 를 직접 PL/SQL 블록 내에서 사용한다.
④ EXCEPTION 블록에서 DECODE 를 감싼다.

1번

8.5-49 다음 중 PL/SQL 에서 사용할 수 없는 함수는? ★★☆☆☆

① UPPER
② SUM
③ LENGTH
④ TO_DATE

2번

8.5-50 다음 중 PL/SQL 에서 DECODE 함수가 오류를 발생시키는 이유로 가장 적절한 것은? ★★★☆☆

① PL/SQL 은 SQL 과 문법이 완전히 다르기 때문이다.
② DECODE 는 PL/SQL 블록에서 사용 시 자동으로 NULL 을 반환하기 때문이다.
③ DECODE 는 SQL 전용 함수이며, PL/SQL 에서는 CASE 표현식으로 대체해야 하기 때문이다.
④ PL/SQL 에서는 문자열 관련 함수를 전혀 사용할 수 없기 때문이다.

3번

8.5-51 PL/SQL 에서 집계 함수와 분석 함수가 직접 사용되지 않는 이유로 적절한 것은? ★★★☆☆

① PL/SQL 은 트랜잭션을 지원하지 않기 때문이다.
② 집계와 분석은 데이터 집합 처리에 해당하며, PL/SQL 은 단일 행 처리에 초점이 맞춰져 있기 때문이다.
③ 분석 함수는 데이터베이스의 물리적 설계와만 관련되어 있기 때문이다.
④ PL/SQL 은 SQL 기능을 포함하지 않기 때문이다.

2번

9 SQL 실행

9.1 SELECT 문의사용

9.1-1 다음 중 PL/SQL 에서 SELECT 문 사용 시 INTO 절에 대한 설명으로 옳지 않은 것은? ★★★☆☆

① SELECT 문의 결과를 저장하기 위해 반드시 INTO 절이 필요하다.
② INTO 절에는 SELECT 되는 칼럼의 개수보다 적은 수의 변수를 지정할 수 있다.
③ SELECT 문에서 여러 개의 칼럼을 조회하는 경우, 동일 개수의 출력 변수를 INTO 절에 나열해야 한다.
④ SELECT 문의 결과가 변수에 저장되도록 하는 역할을 한다.

2번

9.1-2 다음 중 SELECT 문에서 PL/SQL 입력 변수 사용에 대한 설명으로 옳은 것은? ★★★☆☆

① PL/SQL 에서 입력 변수는 반드시 : 기호를 앞에 붙여야 한다.
② SELECT 절에 입력 변수를 사용할 수 없다.
③ WHERE 절의 리터럴 값은 변수로 대체할 수 있다.
④ 입력 변수는 PL/SQL 에서 예약어여도 사용할 수 있다.

3번

9.1-3 다음 중 PL/SQL 에서 %ROWTYPE 사용의 장점으로 가장 적절한 것은? ★★★☆☆

① 여러 테이블의 칼럼을 한 변수로 통합하여 조회할 수 있다.
② SELECT 문의 WHERE 절에만 사용할 수 있다.
③ 테이블 구조가 바뀌어도 코드 수정을 최소화할 수 있다.
④ 레코드 단위로 데이터를 저장할 수 없다.

3번

9.1-4 다음 중 PL/SQL 에서 SELECT 절에 변수를 사용할 때 주의할 점으로 옳은 것은? ★★★☆☆

① SELECT 절에 변수를 사용하면 해당 변수에 조회 결과가 저장된다.
② SELECT 절에 변수를 사용할 수 없다.
③ SELECT 절에서 사용된 변수는 변수의 값이 반환된다.
④ SELECT 절에 변수를 사용하면 컴파일 오류가 발생한다.

3번

9.1-5 다음 중 PL/SQL 에서 BULK COLLECT INTO 구문을 사용하는 목적은 무엇인가? ★★★☆☆

① 한 행씩 데이터를 처리하여 성능을 높이기 위함이다.
② 배열 형태로 데이터를 대량으로 조회하기 위함이다.
③ SELECT 문에서 오류 발생 시 자동 복구를 하기 위함이다.
④ 커서를 열지 않고 데이터를 조회하기 위함이다.

2번

9.2 INSERT 문의 사용

9.2-6 다음 중 INSERT 문에서 %ROWTYPE 을 사용하는 이유로 가장 적절한 것은? ★★★☆☆

① 테이블명을 동적으로 지정하기 위해서
② 칼럼별 DEFAULT 값을 자동으로 적용하기 위해서
③ PL/SQL 변수를 일일이 선언하지 않고 레코드 단위로 처리하기 위해서
④ 트리거를 자동으로 실행시키기 위해서

3번

9.2-7 다음 중 PL/SQL 에서 INSERT 문에 사용할 수 없는 항목은 무엇인가? ★★☆☆☆

① SYSDATE
② PL/SQL 변수
③ %ROWTYPE 으로 선언한 레코드 변수
④ 테이블명에 사용된 변수

4번

9.2-8 다음 중 INSERT 문 실행 후 변경된 건수를 확인하는 데 사용하는 Oracle 의 내장 변수는? ★★☆☆☆

① SQL%FOUND
② SQL%ISOPEN
③ SQL%ROWCOUNT
④ SQL%NOTFOUND

3번

9.2-9 다음 중 INSERT 문에서 칼럼명을 변수로 사용하려고 할 때 발생하는 오류 원인은? ★★★☆☆

① 변수명 충돌
② SQL 구문 내 함수 사용 불가
③ 정적 SQL 의 문법 및 의미 검사 수행
④ 데이터 타입 불일치

3번

9.2-10 다음 중 %ROWTYPE 을 사용하는 경우 발생할 수 있는 결과로 옳은 것은? ★★★☆☆

① 테이블의 DEFAULT 값이 자동으로 필드에 반영된다.
② NULL 값을 가지는 필드도 함께 INSERT 된다.
③ 레코드 변수는 SELECT 문에만 사용된다.
④ INSERT 문에서는 레코드 변수의 일부분만 사용할 수 없다.

2번

9.3 UPDATE 문의 사용

9.3-11 다음 중 PL/SQL 에서 UPDATE 문 실행 후 변경된 행 수를 출력하는 데 사용하는 Oracle 내장 변수는? ★★☆☆☆

① SQL%FOUND
② SQL%ISOPEN
③ SQL%ROWCOUNT
④ SQL%NOTFOUND

3번

9.3-12 다음 중 PL/SQL 에서 %ROWTYPE 레코드 변수를 UPDATE 문에 사용하는 올바른 방식은? ★★★☆☆

① SET 다음에 칼럼별로 변수 필드를 지정해야 한다.
② SET ROW = 레코드변수 형식으로만 사용할 수 있다.
③ UPDATE 문에서는 %ROWTYPE 은 사용할 수 없다.
④ WHERE 절에서 여러 개의 레코드 변수를 사용할 수 있다.

2번

9.3-13 다음 중 PL/SQL 에서 UPDATE 문을 사용할 때 주의해야 할 점으로 가장 적절한 것은? ★★★★☆

① UPDATE 문에서는 테이블명과 칼럼명을 변수로 지정할 수 있다.
② 레코드 변수에서 지정하지 않은 필드 값은 원래 값이 유지된다.
③ 레코드 변수에서 값을 지정하지 않으면 해당 칼럼은 NULL 로 업데이트될 수 있다.
④ SET ROW 를 사용할 때 여러 개의 레코드 변수를 나열할 수 있다.

3번

9.3-14 다음 중 PL/SQL 에서 UPDATE 문을 효율적으로 작성하기 위한 방법으로 가장 적절한 것은? ★★★☆☆

① UPDATE 전에 모든 칼럼 값을 SELECT 하여 레코드 변수에 저장한 뒤 수정할 칼럼만 갱신한다.
② UPDATE 문에서 테이블명을 변수로 받아 동적으로 지정한다.
③ UPDATE 문을 IF 문 안에서 반복해서 실행하여 성능을 향상시킨다.
④ 레코드 변수를 UPDATE 문에서 WHERE 절과 SET 절에 동시에 사용할 수 있다.

1번

9.3-15 다음 중 PL/SQL 에서 FORALL 구문을 사용해야 하는 UPDATE 문의 특징으로 가장 적절한 것은? ★★★☆☆

① 레코드 변수로 특정 필드만 갱신할 때 사용된다.
② 다수의 행을 반복적으로 UPDATE 할 때 성능 향상을 위해 사용된다.
③ UPDATE 문에서 RETURNING 절을 사용할 때 필수적이다.
④ %ROWTYPE 을 사용할 수 없는 경우에만 사용된다.

2번

9.4 MERGE 문의 사용

9.4-16 MERGE 문의 가장 핵심 기능으로 올바른 것은? ★★★☆☆

① 조건에 맞는 데이터가 없으면 DELETE 를 실행한다.
② 조건에 맞는 데이터가 있으면 UPDATE, 없으면 INSERT 를 실행한다.

2번

③ 항상 INSERT 와 DELETE 를 동시에 실행한다.
④ 테이블 전체 데이터를 한 번에 삭제한다.

9.4-17 다음 중 MERGE 문의 UPDATE 절에서 사용할 수 없는 데이터 타입은? ★★☆☆☆

① 단일 칼럼 변수
② PL/SQL 입력 변수
③ 레코드 변수
④ 상수 값

3번

9.4-18 MERGE 문에서 PL/SQL 변수를 사용하여 조건을 지정할 때 올바른 구문은? ★★★☆☆

① ON (a.empno = 9000)
② ON (a.empno = v.empno)
③ ON (a.empno = :empno)
④ ON (a.empno = 'v.empno')

2번

9.4-19 다음 중 MERGE 문에서 키워드, 테이블명, 칼럼명 등에 변수 사용이 불가능한 이유로 옳은 것은? ★★★☆☆

① 문법적으로 허용하지 않기 때문이다.
② 변수 사용 시 성능이 저하되기 때문이다.
③ 테이블명과 칼럼명은 보안상 고정되어야 한다.
④ 변수 사용 시 실행 시간이 늘어나기 때문이다.

1번

9.4-20 MERGE 문에서 다수 행을 효율적으로 처리하기 위해 사용하는 방법은? ★★★☆☆

① 배열 처리와 FORALL 문을 함께 사용한다.
② 다중 MERGE 문을 반복해서 실행한다.
③ 레코드 변수를 UPDATE 절에 사용한다.
④ INSERT 절에만 변수를 사용한다.

1번

9.5 DELETE 문의 사용

9.5-21 DELETE 문의 기본 기능으로 올바른 것은? ★★☆☆☆

① 지정한 조건에 맞는 행을 삭제하고 결과 행을 반환한다.
② 지정한 조건에 맞는 행을 삭제하며, 기본적으로 결과 행은 반환하지 않는다.
③ 테이블 전체를 초기화한다.
④ 삭제 후 반드시 SELECT 문을 실행해야 한다.

2번

9.5-22 DELETE 문에서 PL/SQL 변수를 사용하는 방법으로 옳은 것은? ★★★☆☆

① DELETE FROM emp WHERE empno = 9000;
② DELETE FROM emp WHERE empno = v_empno;
③ DELETE FROM emp WHERE empno = 'v_empno';
④ DELETE FROM emp WHERE empno = :v_empno;

2번

9.5-23 다음 중 DELETE 문에서 변수로 사용할 수 없는 유일한 것은 무엇인가요? ★★☆☆☆

① WHERE 조건절의 값
② PL/SQL 변수
③ DELETE 문에 사용되는 테이블명
④ WHERE 키워드

3번

9.5-24 DELETE 문에서 삭제 후 삭제 건수를 출력하는 방법으로 적절한 것은? ★★★☆☆

① DBMS_OUTPUT.PUT_LINE('삭제 건수: ' || SQL%ROWCOUNT);
② DBMS_OUTPUT.PUT_LINE('삭제 건수: ' || SQLROWCOUNT);
③ DBMS_OUTPUT.PUT_LINE('삭제 건수: ' || SQLCOUNT);
④ DBMS_OUTPUT.PUT_LINE('삭제 건수: ' || SQLCOUNTROWS);

1번

9.5-25 DELETE 문에서 다수의 행을 효율적으로 삭제하기 위한 기법으로 옳은 것은? ★★★☆☆

① FORALL 문을 사용한 배열 처리
② 반복문 내에서 DELETE 문을 여러 번 실행
③ 레코드 변수를 WHERE 절에 직접 사용하는 방법
④ DELETE 문에 RETURNING 절을 반드시 포함하는 방법

1번

9.6 시퀀스 사용

9.6-26 PL/SQL 에서 시퀀스를 사용하는 두 가지 방식은 무엇인가? ★★☆☆☆

① 직접 PL/SQL 문장 내에서 사용 / SQL 문에 포함하여 사용
② 테이블에 직접 입력 / 뷰에 포함하여 사용
③ 함수 호출 / 프로시저 호출
④ PL/SQL 에서만 사용 / SQL 에서만 사용

1번

9.6-27 시퀀스를 직접 PL/SQL 문장 내에서 사용하는 방식의 장점은? ★★★☆☆

① 더 복잡하지만 성능이 좋다.
② 더 간단하고 성능 면에서 유리하다.
③ 항상 SQL 문을 함께 작성해야 한다.
④ 오직 SQL 문에만 포함해서 사용 가능하다.

2번

9.6-28 다음 중 시퀀스를 직접 사용하는 예로 옳은 코드는? ★★★☆☆

① v_seq_value := emp.seq.NEXTVAL;
② SELECT emp.seq.NEXTVAL INTO v_seq_value FROM DUAL;
③ v_seq_value := 'emp.seq.NEXTVAL';
④ INSERT INTO emp VALUES(emp.seq.NEXTVAL);

1번

9.6-29 시퀀스를 SQL 문에 포함하여 사용하는 방식의 단점은? ★★★☆☆

① 더 효율적이다.
② PL/SQL 변수 선언이 불가능하다.
③ 비효율적이다.
④ 시퀀스 값을 반환하지 않는다.

3번

9.6-30 다음 중 시퀀스 사용 시 올바른 출력 방법은? ★★☆☆☆

① DBMS_OUTPUT.PUT_LINE('시퀀스값: ' || TO_CHAR(v_seq_value));
② DBMS_OUTPUT.PUT_LINE('시퀀스값: ' || v_seq_value);
③ PRINT '시퀀스값: ' || v_seq_value;
④ OUTPUT('시퀀스값: ' || TO_CHAR(v_seq_value));

1번

9.7 DML 문의 결과값을 PL/SQL 변수로 반환하는 방법

9.7-31 RETURNING 절의 주요 역할은 무엇인가? ★★★☆☆(단)

① DML 문에서 처리된 값을 PL/SQL 변수로 반환한다.
② DML 문의 실행을 취소한다.
③ DML 문의 실행 계획을 출력한다.
④ DML 문에서 오류를 자동 수정한다.

1번

9.7-32 RETURNING 절과 항상 같이 사용되는 서브절은? ★★☆☆☆

① WHERE
② INTO
③ FROM
④ VALUES

2번

9.7-33 다음 중 RETURNING 절 사용 시 반드시 지켜야 할 조건은? ★★★☆☆

① 반환할 값의 개수와 INTO 변수의 개수가 같아야 한다.
② 반환할 값은 컬럼명만 가능하다.
③ INTO 절은 선택사항이다.
④ RETURNING 절은 MERGE 문에서도 사용 가능하다.

1번

9.7-34 아래 PL/SQL 코드에서 RETURNING 절의 역할로 가장 적절한 것은? ★★★☆☆

```
INSERT INTO emp(empno, ename, hiredate, deptno)
VALUES (9000, '홍길동', c_hiredate, 40)
RETURNING empno, ename, hiredate
INTO v_empno, v_ename, v_hiredate;
```

① 삽입된 행의 empno, ename, hiredate 값을 변수에 저장한다.
② 삽입 행을 삭제한다.
③ 테이블의 모든 데이터를 반환한다.
④ 변수에 기본값을 할당한다.

1번

9.7-35 MERGE 문에서 RETURNING 절을 사용할 수 있는가? ★☆☆☆☆

① 사용할 수 있다.
② 사용할 수 없다.
③ 일부 조건에서만 사용할 수 있다.
④ 사용 시 컴파일 오류 없이 무시된다.

2번

9.8 트랜잭션 제어

9.8-36 데이터베이스 트랜잭션이 갖는 ACID 특성 중 '원자성(Atomicity)'의 설명으로 옳은 것은? ★★★☆☆

① 트랜잭션의 일부 작업이 실패해도 나머지 작업은 정상 처리된다.
② 트랜잭션의 작업이 모두 성공하거나 모두 실패하여 데이터베이스가 변경되지 않는다.
③ 트랜잭션 수행 전후에 데이터베이스가 유효한 상태를 유지한다.
④ 트랜잭션의 성공 결과가 영구적으로 데이터베이스에 반영된다.

2번

9.8-37 트랜잭션의 '고립성(Isolation)' 특성에 대한 설명으로 적절한 것은? ★★☆☆☆

① 트랜잭션이 실행 전후에 데이터베이스가 항상 유효한 상태를 유지한다.
② 성공한 트랜잭션의 변경은 영구적으로 반영된다.
③ 동시 실행된 트랜잭션들이 마치 순차적으로 실행된 것과 같은 결과를 보장한다.
④ 트랜잭션은 일부분만 실행될 수 없다.

3번

9.8-38 다음 중 트랜잭션 제어문(TCL)에 해당하지 않는 것은? ★★★☆☆

① COMMIT
② ROLLBACK
③ SAVEPOINT
④ SELECT

4번

9.8-39 오라클 데이터베이스에서 트랜잭션이 시작되는 시점으로 올바른 것은? ★★☆☆☆

① 세션 접속 직후 또는 명시적 COMMIT, ROLLBACK 실행 직후
② 트랜잭션 미사용 모드에서 자동으로 시작된다.
③ 데이터 변경 시점이 아닌 세션 종료 시점에 시작된다.
④ 오라클에서는 트랜잭션이 항상 수동으로 시작되어야 한다.

1번

9.8-40 트랜잭션의 원자성 제어에 사용하는 명령어로 옳은 것은? ★★★☆☆

① SELECT
② COMMIT, ROLLBACK, SAVEPOINT
③ SET TRANSACTION
④ GRANT

2번

9.8-41 커밋(COMMIT)에 대한 설명으로 옳지 않은 것은? ★★☆☆☆

① 커밋은 트랜잭션의 변경 사항을 영구 저장한다.
② 커밋 이후 다른 사용자가 변경 사항을 조회할 수 있다.
③ 트랜잭션의 일부만 선택적으로 커밋할 수 있다.
④ COMMIT 과 COMMIT WORK 는 기능적으로 동일하다.

3번

9.8-42 다음 중 트랜잭션의 독립성에 관한 설명으로 옳은 것은? ★★☆☆☆

① 한 세션에서 커밋하면 다른 세션의 변경 사항도 커밋된다.
② 두 명이 각각 SQLPlus 를 사용하면 서로 다른 세션으로 트랜잭션이 독립적이다.
③ 한 사람이 두 개의 SQLPlus 를 실행하면 하나의 세션이다.
④ 커밋은 세션 간에 자동으로 동기화된다.

2번

9.8-43 오라클에서 DDL 실행 시 자동으로 발생하는 트랜잭션 제어는? ★★★☆☆

① 명시적 ROLLBACK
② 명시적 COMMIT
③ 묵시적 COMMIT
④ SAVEPOINT 설정

3번

9.8-44 ROLLBACK 에 대한 설명으로 옳지 않은 것은? ★★☆☆☆

① 최근 커밋 또는 롤백 이후부터 변경 사항을 취소한다.
② 세션 접속 직후부터 변경한 사항도 롤백 대상이 될 수 있다.
③ 트랜잭션 일부만 롤백하는 것은 불가능하다(세이브포인트 제외).
④ ROLLBACK 과 ROLLBACK WORK 는 서로 다른 명령이다.

1번 헷갈림
4번

9.8-45 SAVEPOINT 의 역할로 옳은 것은? ★★★☆☆

① 트랜잭션의 특정 지점에 표시를 하여 그 이후부터 롤백할 수 있게 한다.
② 트랜잭션 시작 이전으로 돌아가게 한다.
③ 트랜잭션을 분리하여 독립적인 커밋을 가능하게 한다.
④ 세션 간 트랜잭션 공유를 위한 동기화 포인트이다.

1번

9.8-46 ROLLBACK TO SAVEPOINT 명령에 대한 설명 중 옳은 것은? ★★★☆☆

① 지정한 세이브포인트 이전으로 되돌린다.
② 세이브포인트 이후의 변경만 취소한다.
③ 세이브포인트 이름이 같으면 오류가 발생한다.
④ 트랜잭션 전체를 롤백한다.

1번, 2번 헷갈림

9.8-47 다음 중 트랜잭션에서 두 개의 DML 문을 수행한 후 커밋하는 경우 발생하지 않는 상황은? ★★☆☆☆

① 첫 번째 DML 문만 커밋되고 두 번째는 커밋되지 않는다.
② 두 DML 문 모두 커밋된다.
③ 두 DML 문 중 어느 한 쪽만 커밋 제외된다.
④ 두 DML 문이 하나의 원자적 단위로 커밋된다.

1번

9.8-48 SAVEPOINT 이름을 여러 번 동일하게 사용하는 경우로 옳은 것은? ★★★☆☆

① 동일 이름 세이브포인트가 있으면 오류가 발생한다.
② 동일 이름 세이브포인트가 있으면 나중에 선언된 위치로 덮어쓴다.
③ 세이브포인트 이름은 무조건 달라야 한다.
④ 동일 이름 세이브포인트를 여러 곳에서 지정할 수 있다.

2번

9.8-49 다음 중 묵시적 COMMIT 이 발생하는 경우는? ★★☆☆☆

① DML 문 실행 시
② 트랜잭션 종료 시

③ DDL 문 실행 시
④ ROLLBACK 문 실행 시

3번

9.9 CLOB 사용하기 여기 부분 답 다시 체크 헷갈림

9.9-50 CLOB 과 VARCHAR2 타입의 최대 길이로 옳은 것은? ★★★☆☆

① VARCHAR2 는 최대 4000 바이트, CLOB 은 최대 32,767 바이트까지 저장 가능하다.
② 오라클 12c 부터는 VARCHAR2 최대 길이를 32,767 바이트까지 확장할 수 있다.
③ CLOB 은 VARCHAR2 보다 항상 더 짧은 문자열만 저장할 수 있다.
④ VARCHAR2 와 CLOB 은 최대 저장 길이가 동일하다.

2번

9.9-51 PL/SQL 에서 CLOB 타입 변수 사용에 대한 설명으로 옳지 않은 것은? ★★★☆☆

① PL/SQL 에서는 CLOB 과 VARCHAR2 타입 변수를 거의 구별 없이 사용할 수 있다.
② VARCHAR2 변수 값을 CLOB 타입 변수에 대입하는 것은 문제 없다.
③ CLOB 타입 변수는 모든 내장 함수에 대해 VARCHAR2 와 동일하게 동작한다.
④ CLOB 과 VARCHAR2 간 형 변환은 PL/SQL 에서 자동으로 처리된다.

1번

9.9-52 다음 중 CLOB 데이터 타입을 사용하는 경우로 적절한 것은? ★★★☆☆

① 1000 바이트 이하의 짧은 문자열 저장 시
② 4000 바이트 이하 문자열만 저장할 때
③ 4,000 바이트를 초과하거나 오라클 12c 이상에서 32,767 바이트를 초과하는 문자열 저장 시
④ 모든 문자열 저장에 CLOB 을 기본으로 사용한다

3번

9.9-53 PL/SQL 에서 CLOB 타입 변수와 VARCHAR2 타입 변수 간의 형 변환에 대한 설명으로 옳은 것은? ★★★☆☆

① CLOB 에서 VARCHAR2 로의 변환은 항상 오류 없이 가능하다.
② VARCHAR2 에서 CLOB 으로의 변환은 오류 없이 자동 처리된다.
③ CLOB 과 VARCHAR2 간 변환 시 수동 변환이 필수적이다.
④ 두 타입 간 변환은 지원되지 않는다.

1번

9.9-54 CLOB 타입 변수에 대해 옳지 않은 설명은? ★★★☆☆

① CLOB 타입 변수에 저장된 문자열은 DBMS_OUTPUT.PUT_LINE 으로 출력할 수 있다.
② CLOB 타입 변수에 대해 모든 PL/SQL 내장 함수를 문제 없이 사용할 수 있다.
③ CLOB 타입 변수는 VARCHAR2 와 거의 동일하게 PL/SQL 에서 처리 가능하다.
④ DML 에서 CLOB 타입 변수는 VARCHAR2 타입 칼럼에 사용할 수 있다.

1번

10 제어문

10.1 제어문의 종류

10.1-1 제어문의 역할로 옳지 않은 것은? ★★☆☆☆

① 프로그램의 실행 속도를 높이는 기능
② 논리적 판단과 반복 구조를 사용하여 순차적 처리를 가능하게 하는 기능
③ 데이터베이스 스키마를 정의하는 기능
④ 네트워크 통신을 제어하는 기능

3번

10.1-2 PL/SQL 제어문의 조건분기문에 속하는 것은? ★☆☆☆☆

① GOTO 문
② IF 문
③ EXIT 문
④ FOR LOOP 문

2번

10.1-3 무조건분기문에 해당하는 것은? ★☆☆☆☆

① WHILE LOOP 문
② FOR LOOP 문

3번

③ GOTO 문
④ CASE 문

10.1-4 PL/SQL 순환문에 포함되지 않는 것은? ★☆☆☆☆

① WHILE LOOP 문
② FOR LOOP 문
③ CONTINUE 문
④ IF 문

4번

10.1-5 제어문이 절차적 프로그래밍에서 가지는 역할로 옳은 것은? ★☆☆☆☆

① 데이터 저장 방식을 결정한다.
② 프로그램의 뼈대를 구성하는 핵심 기능이다.
③ 하드웨어와 직접 통신한다.
④ 네트워크 패킷을 분류한다.

2번

10.2 조건 분기문

10.2.1 IF 문

10.2.1-6 조건 분기문에 대한 설명으로 옳지 않은 것은? ★★☆☆☆

① IF 문과 CASE 문은 모두 논리 조건에 따라 서로 다른 문장을 실행할 수 있게 한다.
② IF 문과 CASE 문은 조건을 순차적으로 평가하다가 TRUE 조건을 만나면 그에 해당하는 블록을 실행한다.
③ CASE 문은 반복 처리를 위한 루프를 구성하는 제어문이다.
④ CASE 문은 특정 값에 따라 서로 다른 문장을 실행하는 데 적합하다.

3번

10.2.1-7 IF 문에 대한 설명으로 옳은 것은? ★★☆☆☆

① IF 문은 항상 ELSE 절이 필요하다.
② IF 문은 반복적인 처리를 위해 사용된다.
③ IF 문은 조건이 TRUE 일 때만 특정 문장을 실행할 수 있다.
④ IF 문은 테이블 생성 여부를 판단할 수 없다.

3번

10.2.1-8 다음 중 단순 IF 문으로 적절한 예는? ★☆☆☆☆

① IF v_cnt > 0 THEN ... ELSE ... END IF
② IF v_cnt > 0 THEN ... END IF
③ IF v_cnt > 0 THEN ... ELSIF v_cnt = 0 THEN ... END IF
④ IF v_cnt = 0 THEN ... ELSE ... ELSIF v_cnt > 0 THEN ... END IF

1번

10.2.1-9 다음 중 ELSIF 가 존재하는 IF 문에 대한 설명으로 틀린 것은? ★★☆☆☆

① 여러 조건을 순차적으로 평가하며 TRUE 인 조건에 따라 분기한다.
② ELSE 가 없어도 문장이 실행되지 않을 수 있다.
③ ELSIF 는 'ELSEIF'처럼 철자에 E 가 포함된다.
④ ELSE 는 모든 조건이 FALSE 일 경우에 실행될 수 있다.

3번

10.2.2 CASE 문

10.2.2-10 다음 중 PL/SQL 의 IF 조건문에 대한 설명으로 옳지 않은 것은? ★★☆☆☆

① IF 조건이 TRUE 일 경우 THEN 블록의 문장이 실행된다.
② IF 조건이 FALSE 이면 ELSE 블록의 문장이 실행된다.
③ ELSIF 는 여러 조건을 순차적으로 검사할 때 사용할 수 있다.
④ ELSE 블록은 반드시 존재해야 한다.

4번

10.2.2-11 다음 중 PL/SQL 에서 IF ~ THEN ~ ELSE 구문의 기본 구조로 알맞은 것은? ★★☆☆☆

① IF condition THEN statements END IF;
② IF condition THEN statements ELSE statements;

4번

③ IF condition THEN statements ELSIF condition THEN statements;
④ IF condition THEN statements ELSEIF condition THEN statements END IF;

10.2.2-12 다음 중 PL/SQL 조건문(IF 구문)에 대한 설명으로 적절한 것은? ★★★☆☆
① IF 문은 반드시 ELSE 절을 포함해야 한다.
② ELSIF 는 ELSE 구문 안에 포함되어야 한다.
③ 여러 개의 ELSIF 를 사용할 수 있다.
④ IF 문 안에는 SELECT 문을 사용할 수 없다.

3번

10.2.2-13 다음 중 PL/SQL 의 IF 조건문 실행 흐름에 대한 설명으로 가장 적절한 것은? ★★★☆☆
① 모든 조건은 ELSIF 가 아닌 ELSE 에서 판단한다.
② 첫 번째 조건이 TRUE 이면 이후 조건도 계속 평가한다.
③ 조건이 만족되는 첫 번째 THEN 블록만 실행된다.
④ 모든 THEN 블록이 순서대로 실행된다.

3번

10.2.2-14 다음 중 PL/SQL 조건문에서 'ELSIF'를 사용하는 목적은 무엇인가? ★★★☆☆
① IF 구문의 가독성을 떨어뜨리기 위해 사용한다.
② 하나의 조건만 평가하고 종료하기 위해 사용한다.
③ 다중 조건을 순차적으로 검사하기 위해 사용한다.
④ ELSE 절을 생략할 수 없게 만들기 위해 사용한다.

3번

10.3 무조건 분기문
10.3.1-15 GOTO 문에 대한 설명으로 옳은 것은? ★★★☆☆
① 조건 없이 순차적으로 실행되는 문장을 의미한다.
② 레이블 없이도 특정 위치로 이동할 수 있다.
③ IF 문과 같은 조건문 없이 사용하는 것이 권장된다.
④ 논리 판단 없이 특정 위치로 실행을 이동시키는 문장이다.

4번

10.3.1-16 다음 중 GOTO 문에 대한 설명으로 틀린 것은? ★★★☆☆
① GOTO 문은 IF 문과 함께 사용되기도 한다.
② GOTO 문은 어셈블리 언어 시절의 유물로 여겨진다.
③ GOTO 문은 현재보다 더 깊은 레벨의 레이블로 점프할 수 있다.
④ GOTO 문은 적절히 사용하면 유용할 수 있다.

3번

10.4 레이블
10.4-17 다음 중 레이블(Label)에 대한 설명으로 틀린 것은? ★★☆☆☆
① 레이블은 GOTO 문에서 다음 실행 위치를 지정하는 데 사용된다.
② 레이블명은 변수명과 동일한 규칙을 따르며, 한글도 사용할 수 있다.
③ 레이블 뒤에는 반드시 블록이나 실행 가능한 문장이 와야 한다.
④ 레이블명은 반드시 영문자와 숫자로만 구성되어야 하며, 따옴표 사용은 불가능하다.

4번 헷갈림;

10.4-18 다음 중 GOTO 문에서 사용할 수 있는 레이블 선언 형식으로 올바른 것은? ★★☆☆☆
① GOTO "문장 레이블" := 실행문;
② «레이블명»
③ LABEL 레이블명: 실행문;
④ BEGIN 레이블명; END;

2번

10.4-19 다음 중 레이블을 반드시 사용해야 하는 상황으로 가장 적절한 것은? ★☆☆☆☆
① LOOP 문 안에서 EXIT WHEN 절 사용 시
② FOR 문에서 조건 반복 시
③ GOTO 문을 사용하여 특정 문장으로 분기할 때
④ SELECT INTO 구문으로 데이터를 조회할 때

1번

10.4-20 다음 중 레이블과 관련하여 오류를 방지하기 위한 올바른 설명은? ★★☆☆☆

① 레이블은 항상 BEGIN 바로 앞에 위치해야 한다.
② 레이블 다음에는 최소한 하나의 실행문이 필요하다.
③ 레이블에는 반드시 숫자만 사용할 수 있다.
④ 레이블은 중복되어도 오류가 발생하지 않는다.

2번

10.4-21 다음 중 레이블을 이용해 다중 LOOP 제어를 할 수 없는 문장은? ★★★☆☆

① EXIT OUTMOST_LOOP
② EXIT OUTMOST_LOOP WHEN 조건
③ CONTINUE OUTMOST_LOOP WHEN 조건
④ GOTO LOOP

4번

10.5 순환문

10.5-22 기본 LOOP 문에 대한 설명으로 옳지 않은 것은? ★★☆☆☆

① 종료 조건이 없으면 무한 루프가 된다.
② LOOP 와 END LOOP 사이에 실행문을 작성한다.
③ CTRL+C 로 무한 루프를 종료할 수 있다.
④ 기본 LOOP 문은 WHILE 이나 FOR 없이 반드시 EXIT WHEN 을 사용해야 한다.

4번

10.5-23 다음 중 PL/SQL 의 기본 구조에 대한 설명으로 옳은 것은? ★☆☆☆☆

① 선언부는 BEGIN 과 END 사이에 위치한다
② 실행부는 반드시 작성해야 한다
③ 예외처리부는 필수적으로 작성해야 한다
④ DECLARE 는 예외처리부 시작 시 사용하는 키워드이다

2번

10.5-24 다음 중 실행 가능한 PL/SQL 문장은? ★☆☆☆☆

① DECLARE v_salary NUMBER;
② BEGIN v_salary := 5000;
③ v_salary NUMBER := 1000;
④ v_name CHAR := '홍길동';

1번

10.5-25 다음 중 PL/SQL 에서 예외 처리에 대한 설명으로 옳지 않은 것은? ★★★☆☆

① WHEN OTHERS THEN 은 모든 예외를 포괄한다
② 예외는 반드시 사용자가 정의한 이름으로 선언해야 한다
③ 예외처리 블록은 EXCEPTION 으로 시작한다
④ RAISE 문으로 사용자 정의 예외를 발생시킬 수 있다

2번

10.5-26 다음 중 PL/SQL 블록에서 변수 선언 시 올바른 문장은? ★☆☆☆☆

① v_name CHAR(10) := 'John';
② v_date := SYSDATE;
③ v_salary NOT NULL NUMBER := 0;
④ v_id NUMBER := ;

1번

10.5-27 다음 중 BEGIN ~ END 블록 안에서 유효하게 사용할 수 있는 문장은? ★☆☆☆☆

① 변수 선언문
② IF 조건문
③ CURSOR 선언문
④ EXCEPTION 선언문

4번

10.5-28 다음 중 WHILE 반복문의 특징으로 옳지 않은 것은? ★★☆☆☆

① 조건이 TRUE 일 때 반복한다
② 루프 시작 전에 조건을 평가한다
③ 반복 횟수를 정해놓고 사용하는 데 적합하다
④ EXIT 문을 사용하여 루프에서 빠져나올 수 있다

3번

10.5-29 다음 중 FOR 루프에 대한 설명으로 옳은 것은? ★★★☆☆

① 루프 변수는 명시적으로 선언해야 한다
② 루프 종료 후 루프 변수는 참조 가능하다
③ 루프 변수는 암시적으로 선언된다
④ FOR 루프는 REVERSE 키워드를 사용할 수 없다

3번

10.5-30 다음 중 LOOP 문에 대한 설명으로 옳은 것은? ★★☆☆☆

① 기본 LOOP 는 루프 조건을 반드시 포함해야 한다
② EXIT 없이도 자동으로 루프가 종료된다
③ 무한 루프가 될 수 있으므로 조건 제어가 필요하다
④ LOOP 는 반드시 FOR 나 WHILE 키워드를 포함해야 한다

3번

10.5-31 다음 중 PL/SQL 에서 IF 조건문에 대한 설명으로 옳은 것은? ★★☆☆☆

① IF 문은 반드시 ELSE 절을 포함해야 한다
② IF 문은 BEGIN 블록 바깥에서만 사용해야 한다
③ IF 조건식은 Boolean 값을 반환해야 한다
④ IF 문 안에는 다른 선언문을 포함할 수 있다

3번

10.6 제어 구조의 중첩

10.6-32 제어 구조 중첩에 관한 설명으로 옳지 않은 것은? ★★★☆☆

① IF 문 내부에 다른 IF 문이나 LOOP 문이 포함될 수 있다.
② LOOP 문 내부에 IF 문이나 다른 LOOP 문이 포함될 수 있다.
③ IF 문과 LOOP 문은 서로 겹쳐질 수 있다.
④ 제어 구조의 중첩 시 들여쓰기는 사람이 이해하기 쉽게 하는 용도이다.

4번

10.6-33 다음 중 중첩된 IF 문을 올바르게 닫는 방법으로 옳은 것은? ★★★☆☆

① 내부 IF 문의 END IF 를 먼저 닫고, 외부 IF 문의 END IF 를 나중에 닫는다.
② 외부 IF 문의 END IF 를 먼저 닫고, 내부 IF 문의 END IF 를 나중에 닫는다.
③ IF 문은 END IF 가 필요 없다.
④ 두 IF 문의 END IF 를 동시에 닫는다.

1번

10.6-34 다음 중 컴파일러가 들여쓰기를 해석하는 방식으로 옳은 것은? ★★★☆☆

① 들여쓰기는 컴파일러가 구조를 이해하는 데 필수적이다.
② 들여쓰기는 문법적으로 의미가 없으며, 단지 사람이 보기 쉽게 하는 용도이다.
③ 들여쓰기가 잘못되면 컴파일 에러가 발생한다.
④ 들여쓰기는 LOOP 문에는 적용되지 않는다.

2번

10.6-35 다음 중 LOOP 문 중첩에 대한 설명으로 옳은 것은? ★★★☆☆

① LOOP 문 내부에 다른 LOOP 문을 중첩할 수 있다.
② LOOP 문은 중첩될 수 없으며, 항상 하나만 사용해야 한다.
③ END LOOP 다음에 오는 변수명은 문법적으로 필수이다.
④ END LOOP 다음에 오는 변수명은 컴파일러가 해석하는 중요한 정보이다.

1번

10.6-36 아래 코드에서 오류가 발생하지 않는 이유로 옳은 것은? ★★★☆☆

pgsql
복사편집
FOR i IN 1 .. 3

```
LOOP
  FOR j IN 1 .. 3
  LOOP
    NULL;
  END LOOP i;
END LOOP j;
```

2번

① END LOOP 뒤에 오는 변수명은 컴파일러에게 변수 구분 정보를 준다.
② 변수명이 붙었지만 문법적으로 무시되고 오류가 발생하지 않는다.
③ 변수명이 틀려서 오류가 발생해야 한다.
④ 이 코드는 컴파일 에러를 발생시킨다.