SQL BASIC

구간	문제 번호	주제	난이도
1~10	001~010	SQL 명령어 분류 및 기본	⭐ 초급
11~20	011~020	테이블 생성 및 제약조건	⭐⭐ 초~중급
21~30	021~030	관계 설정 및 트랜잭션	⭐⭐ 중급
31~40	031~040	SELECT 조건 및 NULL 처리	⭐⭐⭐ 중~고급
41~50	041~050	고급 함수 및 날짜/CASE	⭐⭐⭐ 고급
51~64	051~064	실전 SELECT + GROUP BY/HAVING + JOIN	⭐⭐⭐⭐ 실무형

✅ 41~50번: 고급 SQL 함수, 날짜 처리, CASE 문

- LENGTH, REPLACE, TO_DATE, TO_CHAR
- CASE 문 변형, GROUP BY 조건

[문제 041]
다음 중 아래와 같은 2건의 데이터 상황에서 SQL의 수행 결과로 가장 적절한 것은? (단, 이해를 돕기 위해 ↓는 줄바꿈을 의미 실제 저장값이 아님, CHR(10): ASCII 값 → 줄바꿈을 의미)

[TAB1]

ROWNUM	C1
1	A ↓ A
2	B ↓ B ↓ B

```
SELECT SUM(CC)
FROM (
   SELECT(LENGTH(C1) - LENGTH(REPLACE(C1, CHR(10))) + 1) CC
   FROM TABI
)
```

① 2
② 3
③ 5
④ 6

정답: ③

🧸 **쉬운 해설:**
줄바꿈 문자(CHR(10)) 개수 + 1이 줄 수야!
A↓A → 1개 + 1 = 2줄
B↓B↓B → 2개 + 1 = 3줄
→ 총합: 2 + 3 = 5줄

📚 **전문 해설:**
REPLACE로 줄바꿈 문자 제거 후 길이 차이를 계산하면 줄 수를 알 수 있습니다.
각 행의 줄 수를 구해 합산하면 총 5줄입니다.

```
SQL 처리 순서 한 줄씩 요약
1️⃣ SELECT(LENGTH(C1) - LENGTH(REPLACE(C1, CHR(10))) + 1) → 각 행의 줄 수 계산
- 줄바꿈 문자(CHR(10)) 개수 + 1 = 줄 수
  ※ 한 줄이 끝나고 다음 줄로 넘어갈 때 들어가는 특수 문자
- 행 1: "A↓A" → CHR(10) 1개 → 1 + 1 = 2줄
- 행 2: "B↓B↓B" → CHR(10) 2개 → 2 + 1 = 3줄

2️⃣ FROM TAB1 → 테이블 TAB1에서 두 행 조회
3️⃣ SELECT SUM(CC) → 각 행의 줄 수(2, 3)를 더함
4️⃣ 최종 결과: 2 + 3 = 5
```

🧠 **기억법:**
LENGTH - REPLACE + 1 = 줄 수

필요 암기카드:

- 🃏 카드 34: COALESCE = NULL 처리

```
[문제 042]
오라클환경에서 날짜형 데이터를 다룰 경우, 아래 SQL 결과로 가장 적절한 것은?
```

```
SELECT   TO_CHAR(
    TO_DATE('2015.01.10 10', 'YYYY, MM, DD HH24') + 1/24/(60/10)
  , 'YYYY,MM,DD HH24:MI:SS')
FROM DUAL;
```

① 2015.01.10 11:01:00
② 2015.01.10 10:05:00
③ 2015.01.10 10:10:00
④ 2015.01.10. 10:30:00

정답: ③

🧸 **쉬운 해설:**
1/24는 1시간, 그걸 6으로 나누면 10분!
→ 10시 + 10분 = 10:10:00

📚 **전문 해설:**
1/24/(60/10) = 10분
TO_DATE로 날짜 생성 후 10분 더한 결과를 TO_CHAR로 포맷 변환

```
SQL 처리 순서 한 줄씩 요약
1️⃣ '2015.01.10 10' 문자열을 'YYYY, MM, DD HH24' 형식으로 날짜로 변환 → TO_DATE   → 결과: 2015-01-10 10:00:00

2️⃣ 1/24/(60/10) 계산 →   - 1/24 = 1시간   - 60/10 = 6   - 1/24 ÷ 6 = 1/144 = 10분

3️⃣ 날짜에 1/144를 더함 → 2015-01-10 10:10:00

4️⃣ 결과 날짜를 'YYYY,MM,DD HH24:MI:SS' 형식으로 문자열로 변환 → TO_CHAR

5️⃣ 최종 출력: '2015,01,10 10:10:00'
```

🧠 **기억법:**
1/24 = 1시간 → 나누면 분 단위

필요 암기카드:

- 🃏 카드 45: TO_CHAR = 날짜 포맷 변환

```
[문제 043]
아래는 SEARCHED_CASE_EXPRESSION SQL문장이다. 이때 사용된 SEARCHED_CASE_EXPRESSION은 SIMPLE_CASE_EXPRESSION을
이용해 똑 같은 기능을 표현할 수 있다. 아래 SQL 문장의 ㉠ 안에 들어갈 표현을 작성하시오. (스칼라 서브쿼리는 제외함)
```

```
[SEARCHED CASE EXPRESSION 문장 사례]
SELECT LOC,
    CASE WHEN LOC = 'NEW YORK' THEN 'EAST'
    ELSE 'ETC'
    END as AREA
FROM DEPT;

[SIMPLE_CASE EXPRESSION 문장 사례]
SELECT LOC,
    CASE ㉠
    ELSE 'ETC'
    END as AREA
FROM DEPT;
```

정답: LOC WHEN 'NEW YORK' THEN 'EAST'

🧸 **쉬운 해설:**
SIMPLE CASE는 비교 대상 먼저 쓰고 WHEN 조건 붙이는 거야!

📚 **전문 해설:**
SIMPLE CASE는 CASE 컬럼명 WHEN 값 THEN 결과 형식
SEARCHED CASE는 CASE WHEN 조건 THEN 결과 형식

🧠 **기억법:**
SIMPLE CASE = CASE 컬럼 WHEN 값 THEN 결과

필요 암기카드:

- 🎉 카드 43: CASE 문 실전 예시

```
[문제 044]
팀별 포지션별 FW, MF, DF, GK 포지션의 인원수와 팀별 전체 인원수를 구하는 SQL을 작성할 때 결과가 다른 것은? (보기 1은 SQL Server 환경이고, 보기 2,3,4는 ORACLE 환경이다)
```

```
① SELECT TEAM ID,
    ISNULL(SUM(CASE WHEN POSITION = 'FW' THEN 1 END), 0) FW,
    ISNULL(SUM(CASE WHEN POSITION = 'MF' THEN 1 END), 0) MF,
    ISNULL(SUM(CASE WHEN POSITION = 'DF' THEN 1 END), 0) DF,
    ISNULL(SUM(CASE WHEN POSITION = 'GK' THEN 1 END), 0) GK,
    COUNT(*) SUM
FROM PLAYER
GROUP BY TEAM ID;

② SELECT TEAM ID,
    NVL(SUM(CASE POSITION WHEN 'FW' THEN 1 END),0) FW,
    NVL(SUM(CASE POSITION WHEN 'MF' THEN 1 END),0) MF,
    NVL(SUM(CASE POSITION WHEN 'DF' THEN 1 END),0) DF,
    NVL(SUM(CASE POSITION WHEN 'GK' THEN 1 END),0) GK,
    COUNT(*) SUM
FROM PLAYER
GROUP BY TEAM ID;

③ SELECT TEAM_ID,
    NVL(SUM(CASE WHEN POSITION = 'FW' THEN 1 END), 0) FW,
    NVL(SUM(CASE WHEN POSITION = 'MF' THEN 1 END), 0) MF,
    NVL(SUM(CASE WHEN POSITION = 'DF' THEN 1 END), 0) DF,
    NVL(SUM(CASE WHEN POSITION = 'GK' THEN 1 END), 0) GK,
    COUNT(*) SUM
FROM PLAYER
GROUP BY TEAM ID;

④ SELECT
    TEAM_ID,
    NVL(SUM(CASE POSITION WHEN 'FW' THEN 1 ELSE 1 END),0) FW,
    NVL(SUM(CASE POSITION WHEN 'MF' THEN 1 ELSE 1 END),0) MF,
    NVL(SUM(CASE POSITION WHEN 'DF' THEN 1 ELSE 1 END),0) DF,
    NVL(SUM(CASE POSITION WHEN 'GK' THEN 1 ELSE 1 END),0) GK,
    COUNT(*) SUM
FROM PLAYER
GROUP BY TEAM_ID;
```

정답: ④

🧸 **쉬운 해설:**
④는 ELSE 1이라서 조건 안 맞아도 1을 더해버려!
그래서 인원 수가 부풀려져!

📚 **전문 해설:**
CASE WHEN 조건만 있을 때는 조건 만족 시만 1을 더함
ELSE 1을 넣으면 조건 불만족 시에도 1을 더하므로 결과가 달라짐

🧠 **기억법:**
CASE ELSE 1 → 조건 안 맞아도 더함

필요 암기카드:

- 🃏 카드 44: DML vs DDL

```
[문제 045]
다음 중아래 TAB1을 보고 각 SQL 실행 결과를 가장 올바르게 설명한 것을 고르시오.
```

[TAB1]

COL1	COL2
a	NULL
b	''
c	3
d	4
e	3

```
① SELECT COL2 FROM TAB1 WHERE COL1 = 'b';
→ 실행 결과가 없다. (공집합)
② SELECT ISNULL(COL2, 'X') FROM TABI WHERE COL1 = 'a':
→ 실행 결과로 'X'를 반환한다.
③ SELECT COUNT(COL1) FROM TABI WHERE COL2 = NULL;
→ 실행 결과는 1이다.
④ SELECT COUNT(COL2) FROM TABI WHERE COL1 IN ('b', 'c');
→ 실행 결과는 1이다.
```

정답: ②

🧸 **쉬운 해설:**
a행의 COL2는 NULL이니까 ISNULL로 'X'로 바꿔서 출력돼!

🧤 **전문 해설:**

- ISNULL 함수는 첫 번째 인자가 NULL이면 두 번째 인자를 반환합니다.
- ①은 빈 문자열('')은 NULL이 아니므로 결과가 있음
- ③은 NULL 비교는 IS NULL로 해야 하므로 잘못됨
- ④는 b는 빈 문자열(집계 포함), c는 3 → COUNT는 NULL 제외하므로 2가 나와야 함

보기 설명:
SQL 처리 순서 및 실행 예시

1️⃣ SELECT COL2 FROM TAB1 WHERE COL1 = 'b';

- **실행:** COL1이 'b'인 행을 찾음 → COL2는 '' (빈 문자열)
- **결과:** 빈 문자열은 NULL이 아님 → 결과 있음
- **예시 출력:** '' (공백 문자열)
- ❌ 설명이 틀림 (공집합 아님)

2️⃣ SELECT ISNULL(COL2, 'X') FROM TAB1 WHERE COL1 = 'a';

- **실행:** COL1이 'a'인 행의 COL2는 NULL
- ISNULL(NULL, 'X') → NULL이면 'X' 반환
- **결과:** 'X'
- ✅ 설명이 맞음 → **정답**

3️⃣ SELECT COUNT(COL1) FROM TAB1 WHERE COL2 = NULL;

- **실행:** COL2 = NULL은 항상 FALSE → 아무 행도 선택 안 됨
- **결과:** 0
- ❌ 설명이 틀림 (결과는 1이 아님)
- 🔑 올바른 비교는 COL2 IS NULL

4️⃣ SELECT COUNT(COL2) FROM TAB1 WHERE COL1 IN ('b', 'c');

- **실행:** 조건에 맞는 행:
  - 'b' → COL2 = '' (빈 문자열) → 포함됨
  - 'c' → COL2 = 3 → 포함됨
- COUNT(COL2)는 NULL 제외하고 개수 셈
- **결과:** 2
- ❌ 설명이 틀림 (결과는 1이 아님)

🧠 기억법:

ISNULL = NULL이면 대체값 반환

필요 암기카드:

- 카드 34: COALESCE = NULL 처리

[문제 046]
사원 테이블에서 MGR의 값이 7698과 같으면 NULL을 표시하고, 같지 않으면 MGR을 표시 하려고 한다.
아래 SQL 문장의 [가] 안에 들어갈 함수명을 작성하시오.

```
SELECT ENAME, EMPNO, MGR, [가](MGR,7698) as NM
FROM EMP;
```

정답: NULIF

🧸 **쉬운 해설:**
7698이면 NULL로 바꾸고, 아니면 원래 값 보여주는 함수는 NULIF야!

📚 **전문 해설:**
NULIF(expr1, expr2)는 expr1과 expr2가 같으면 NULL을 반환하고, 다르면 expr1을 반환합니다.
MGR이 7698이면 NULL, 아니면 MGR 그대로 출력됩니다.

🧠 **기억법:**
NULIF = 같으면 NULL

필요 암기카드:

- 🃏 카드 49: NULIF = 같으면 NULL

```
[문제 047]
다음 중 아래 데이터를 가지고 있는 EMP_Q 테이블에서 세개의 SQL 결과로 가장 적절한 것은?
```

```
SELECT SAL/COMM FROM EMP_Q WHERE ENAME = 'KING';
SELECT SAL/COMM FROM EMP_Q WHERE ENAME = 'FORD';
SELECT SAL/COMM FROM EMP_Q WHERE ENAME = 'SCOTT';

※단, SCOTT의 COMM은 NULL 값임
```

[EMP_Q]

ENAME (문자타입)	SAL(숫자타입)	COMM(숫자타입)
KING	0	300
FORD	5000	0
SCOTT	1000	

```
① 0, NULL, NULL
② 0, 에러 발생, 에러 발생
③ 에러 발생, 에러 발생, NULL
④ 0, 에러 발생, NULL
```

정답: ④

🧸 **쉬운 해설:**
0 나누기는 돼! 0 ÷ 300 = 0

하지만 숫자 ÷ 0은 에러고, 숫자 ÷ NULL은 NULL이야!

📚 **전문 해설:**

- KING: 0 ÷ 300 = 0
- FORD: 5000 ÷ 0 → 0으로 나누기 → 에러
- SCOTT: 1000 ÷ NULL → 결과는 NULL

보기 설명:
SQL 실행 예시 및 처리 순서

1️⃣ **SELECT SAL/COMM FROM EMP_Q WHERE ENAME = 'KING';**

- KING의 SAL = 0, COMM = 300
- 계산: 0 / 300 = 0
- **결과:** 0 (정상 실행)

2️⃣ **SELECT SAL/COMM FROM EMP_Q WHERE ENAME = 'FORD';**

- FORD의 SAL = 5000, COMM = 0
- 계산: 5000 / 0 → **0으로 나누기**는 **오라클에서 오류 발생**
- **결과: 에러 발생**

3️⃣ **SELECT SAL/COMM FROM EMP_Q WHERE ENAME = 'SCOTT';**

- SCOTT의 SAL = 1000, COMM = NULL
- 계산: 1000 / NULL → NULL과의 연산은 결과가 NULL
- **결과:** NULL

4️⃣ **최종 결과 예시**

```
-- KING
0

-- FORD
-- ORA-01476: divisor is equal to zero (에러 발생)

-- SCOTT
NULL
```

- **결과: ④ 0, 에러 발생, NULL**

🧠 **기억법:**
0 ÷ 숫자 = 0
숫자 ÷ 0 = 에러
숫자 ÷ NULL = NULL

필요 암기카드:

- 🃏 카드 34: COALESCE = NULL 처리

```
[문제 048]
다음 중 아래와 같은 데이터 상황에서 SQL의 수행 결과로 가장 적절한 것은?
```

TAB1

C1	C2	C3
1	2	3
	2	3
		3

```sql
SELECT SUM(COALESCE(C1, C2, C3))
FROM TAB1
```

① 0
② 1
③ 6
④ 14

정답: ③

🧸 **쉬운 해설:**
COALESCE는 NULL이 아닌 첫 번째 값을 가져와!
→ 1행: 1, 2행: 2, 3행: 3 → 합계는 6

📚 **전문 해설:**
COALESCE는 인자 중 NULL이 아닌 첫 번째 값을 반환합니다.
각 행별로:

- 1행: C1=1 → 1
- 2행: C1=NULL, C2=2 → 2
- 3행: C1=NULL, C2=NULL, C3=3 → 3
  → 총합: 1 + 2 + 3 = 6

> **SQL 처리 순서 한 줄씩 요약**
> 1️⃣ FROM TAB1 → 테이블 TAB1의 모든 행을 가져옴
> 2️⃣ 각 행에 대해 COALESCE(C1, C2, C3) 실행
> **COALESCE는 여러 값 중에서 NULL이 아닌 첫 번째 값을 반환하는 함수**

- 1행: C1=1 → **1 반환**
- 2행: C1=NULL, C2=2 → **2 반환**
- 3행: C1=NULL, C2=NULL, C3=3 → **3 반환**

3️⃣ 반환된 값들을 SUM()으로 합산
→ 1 + 2 + 3 = **6**

4️⃣ 최종 결과 출력: **6**

🧠 **기억법:**
COALESCE = 첫 번째 NULL 아닌 값

필요 암기카드:

- 🎴 카드 34: COALESCE = NULL 처리

```
[문제 049]
아래의 각 함수에 대한 설명 중 (ㄱ), (ㄴ), (ㄷ)에 들어갈 함수를 차례대로 작성하시오.

(ㄱ)  (표현식1, 표현식2): 표현식1의 결과값이 NULL이면 표현식 2의 값을 출력한다.
(ㄴ)  (표현식1, 표현식2): 표현식1이 표현식2와 같으면 NULL을, 같지 않으면 표현식1을 리턴한다.
(ㄷ)  (표현식1, 표현식2): 임의의 개수 표현식에서 NULL이 아닌 최초의 표현식을 나타낸다.
```

정답: ㄱ - NVL, ㄴ - NULIF, ㄷ - COALESCE

🧸 **쉬운 해설:**
NVL은 NULL이면 대체,
NULIF는 같으면 NULL,
COALESCE는 NULL 아닌 첫 번째 값!

📚 **전문 해설:**

- NVL(expr1, expr2): expr1이 NULL이면 expr2 반환
- NULIF(expr1, expr2): 같으면 NULL, 다르면 expr1
- COALESCE(expr1, expr2, ...): NULL이 아닌 첫 번째 값 반환

보기 설명:

항목	설명	정답
ㄱ	NULL이면 대체	NVL
ㄴ	같으면 NULL	NULIF
ㄷ	NULL 아닌 첫 값	COALESCE

🧠 **기억법:**
NVL = NULL이면 대체
NULIF = 같으면 NULL
COALESCE = 첫 번째 유효값

필요 암기카드:

- 🃏 카드 49: NVL / NULIF / COALESCE

```
[문제 050]
다음 중 아래 각각 3개의 SQL 수행 결과로 가장 적절한 것은?
```

```
SELECT AVG(COL3) FROM TAB_A:
SELECT AVG(COL3) FROM TAB_A WHERE COL1 > 0;
SELECT AVG(COL3) FROM TAB_A WHERE COL1 IS NOT NULL;
```

[TAB_A]

COL1	COL2	COL3
30	NULL	20
NULL	40	0
0	10	NULL

```
① 20, 20, 20
② 20, 10, 10
③ 10, 20, 20
④ 10, 10, 10
```

정답: ③

🧸 **쉬운 해설:**
AVG는 NULL을 제외하고 평균을 내!
조건에 따라 포함되는 행이 달라져서 결과도 달라져!

📚 **전문 해설:**

- 전체 AVG(COL3): 20 + 0 → 평균 = 10
- WHERE COL1 > 0 → COL1=30만 해당 → COL3=20 → 평균 = 20
- WHERE COL1 IS NOT NULL → COL1=30, 0 → COL3=20, NULL → 평균 = 20

1️⃣ **SELECT AVG(COL3) FROM TAB_A;**

- TAB_A 전체 행에서 COL3 값 추출
- NULL 제외하고 평균 계산: 20, 0 → (20 + 0) / 2 = **10**

2️⃣ **SELECT AVG(COL3) FROM TAB_A WHERE COL1 > 0;**

- 조건: COL1 > 0 → 해당 행: COL1 = 30
- 해당 행의 COL3 = 20 → 평균 = **20**

3️⃣ **SELECT AVG(COL3) FROM TAB_A WHERE COL1 IS NOT NULL;**

- 조건: COL1이 NULL이 아닌 행 → COL1 = 30, 0
- 해당 행의 COL3 = 20, NULL → NULL 제외하고 평균 = **20**

🎯 **최종 결과**

- 첫 번째 쿼리: **10**
- 두 번째 쿼리: **20**
- 세 번째 쿼리: **20**

✅ 정답: ③ **10, 20, 20**

🧠 **기억법:**
AVG는 NULL 제외하고 계산됨
조건에 따라 포함되는 행이 달라짐

필요 암기카드:

- 🃏 카드 33: COUNT(*) = 전체 행
- 🃏 카드 34: COALESCE = NULL 처리