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

✅ 31~40번: 트랜잭션, SELECT, 집계 함수, NULL 처리

- 트랜잭션 흐름, SAVEPOINT, SELECT 조건절
- COUNT, DISTINCT, IS NULL 등 실전 활용

[문제 031]
아래와 같은 테이블에 SQL구문이 실행되었을 경우 최종 출력 값을 작성하시오.

[품목]

품목ID	단가
001	1000
002	2000
003	1000
004	2000

```
[SQL구문]
BEGIN TRANSACTION

INSERT INTO 품목(품목ID, 단가) VALUES('005', 2000)
COMMIT

BEGIN TRANSACTION
DELETE 품목 WHERE 품목ID='002'

BEGIN TRANSACTION
UPDATE 품목 SET 단가=2000 WHERE 단가=1000
ROLLBACK

SELECT COUNT(품목ID) FROM 품목 WHERE 단가=2000
```

① 0
② 2
③ 3
④ 4

정답: ③

🧸 쉬운 해설:
DELETE로 하나 지우고, UPDATE는 ROLLBACK돼서 반영 안 돼!
그래서 단가 2000인 품목은 총 3개야!

📚 전문 해설:

SQL 처리 순서 한 줄씩 요약
1️⃣ BEGIN TRANSACTION → 트랜잭션 시작
2️⃣ INSERT INTO 품목 VALUES('005', 2000) → 품목 '005' 추가됨
3️⃣ COMMIT → 추가된 '005' 확정됨

4️⃣ BEGIN TRANSACTION → 새 트랜잭션 시작
5️⃣ DELETE 품목 WHERE 품목ID='002' → 품목 '002' 삭제됨 (아직 COMMIT 안 됨)

6️⃣ BEGIN TRANSACTION → 또 다른 트랜잭션 시작
7️⃣ UPDATE 품목 SET 단가=2000 WHERE 단가=1000 → 단가 1000 → 2000으로 변경 시도
8️⃣ ROLLBACK → 변경 취소됨, 단가 1000 그대로 유지됨

9️⃣ SELECT COUNT(품목ID) FROM 품목 WHERE 단가=2000 → 단가 2000인 품목 개수 조회

최종 상태 기준 품목 테이블

품목ID	단가
001	1000

품목ID	단가
003	1000
004	2000
005	2000

➡ 단가 2000인 품목: '004', '005', 그리고 '002'는 삭제됨
→ **총 3개 → 정답: ③**

보기 설명:

보기 번호	설명	적절성
①	UPDATE 반영 시	❌
②	DELETE 반영만 고려	❌
③	INSERT 반영 + UPDATE 무효	✅
④	전체 품목 수 오해	❌

🧠 **기억법:**
ROLLBACK = 변경 무효

필요 암기카드:

- 🃏 카드 53: ROLLBACK = 취소

[문제 032]
아래의 상품 테이블의 데이터에 대하여 관리자가 아래와 같이 SQL문장을 실행하여 데이터를 변경하였다.
데이터 변경 후의 상품ID '001'의 최종 상품명을 작성하시오.

[테이블 : 상품]

상품ID	상품명
001	TV

```
[SQL]
BEGIN TRANSACTION:
SAVE TRANSACTION SP1;

UPDATE 상품 SET 상품명 = 'LCD-TV' WHERE 상품ID = '001';
SAVE TRANSACTION SP2;

UPDATE 상품 SET 상품명 = '평면-TV' WHERE 상품ID = '001';
ROLLBACK TRANSACTION SP2;

COMMIT:
```

정답 : LCD-TV

🧸 **쉬운 해설:**

SP2까지 변경했다가 ROLLBACK 했으니까 SP1의 LCD-TV가 남아!

📚 **전문 해설:**

- SP1: LCD-TV로 변경
- SP2: 평면-TV로 변경
- ROLLBACK SP2 → 평면-TV 취소
  → 최종값은 LCD-TV

```
SQL 처리 순서 한 줄씩 요약
1️⃣ BEGIN TRANSACTION: → 트랜잭션 시작
2️⃣ SAVE TRANSACTION SP1; → 저장지점 SP1 설정
3️⃣ UPDATE 상품 SET 상품명 = 'LCD-TV' WHERE 상품ID = '001'; → 상품명 'LCD-TV'로 변경
4️⃣ SAVE TRANSACTION SP2; → 저장지점 SP2 설정
5️⃣ UPDATE 상품 SET 상품명 = '평면-TV' WHERE 상품ID = '001'; → 상품명 '평면-TV'로 변경
6️⃣ ROLLBACK TRANSACTION SP2; → SP2 이후 변경 취소 → 다시 'LCD-TV'로 복원
7️⃣ COMMIT: → LCD-TV 상태로 확정 저장
```

🧠 **기억법:**

SAVEPOINT → ROLLBACK 시점 기준 복원

필요 암기카드:

- 🃏 카드 53: ROLLBACK = 취소

```
[문제 033]
아래의 ① 에 들어갈 내용을 적으시오.

SQL을 사용하여 데이터베이스에서 데이터를 조회할 때 원하는 데이터 만을 검색하기 위해서 SELECT,
FROM 절과 함께 ① 을(를) 이용하여 조회되는 데이터의 조건을 설정하여 데이터를 제한할 수 있다.
```

정답 : WHERE

🧸 **쉬운 해설:**
조건 걸 땐 WHERE 써야지!

📚 **전문 해설:**
WHERE 절은 SELECT 문에서 조건을 지정할 때 사용됩니다.
GROUP BY나 HAVING은 집계 후 조건에 사용됩니다.

🧠 **기억법:**
WHERE = 필터링 조건

필요 암기카드:

- 🃏 카드 22: WHERE = 필터링

```
[문제 034]
다음 중 SQL의 실행 결과로 가장 적절한 것은?
```

[테이블 : EMP_TBL]

EMPNO	SAL
001	1500
002	3000
003	2000

```
SELECT COUNT(*)
FROM EMP_TBL
WHERE EMPNO > 100 AND SAL >= 3000 OR EMPNO = 200;
```

```
① 0
② 1
③ 2
④ 3
```

정답: ②

🧸 **쉬운 해설:**
조건이 두 개인데 OR로 묶여 있어!
EMPNO가 200인 행만 만족하니까 결과는 1개야!

📚 **전문 해설:**

조건절:

```
EMPNO > 100 AND SAL >= 3000 OR EMPNO = 200
```

우선순위에 따라 괄호 없이 해석하면:

```
(EMPNO > 100 AND SAL >= 3000) OR (EMPNO = 200)
```

테이블에서 EMPNO는 001, 002, 003 → EMPNO = 200은 해당 없음
→ EMPNO > 100 AND SAL ≥ 3000 → EMPNO 002만 만족
→ 결과는 1건

```
SQL 처리 순서 한 줄씩 요약
1️⃣ SELECT COUNT(*) → 조건에 맞는 행의 개수를 셈
2️⃣ FROM EMP_TBL → EMP_TBL 테이블에서 조회
3️⃣ WHERE EMPNO > 100 AND SAL >= 3000 OR EMPNO = 200 → 괄호 우선순위에 따라 다음처럼 해석됨:
(EMPNO > 100 AND SAL >= 3000) OR (EMPNO = 200)
4️⃣ EMPNO 002는 조건 만족 → 포함
5️⃣ EMPNO 001, 003은 조건 불만족 → 제외
6️⃣ EMPNO = 200인 행은 없음 → 포함되지 않음
7️⃣ 최종 결과: 조건에 맞는 행은 1개
8️⃣ 출력값: 1
```

🧠 **기억법:**

AND가 먼저, OR는 나중 → 괄호로 우선순위 정리!

필요 암기카드:

- 🃏 카드 22: WHERE = 필터링
- 🃏 카드 33: COUNT(*) = 전체 행

```
[문제 035]
다음 중 SELECT COL1 + COL3 FROM TAB_A; 의 결과로 가장 적절한 것은?
```

TAB_A (레코드 3건)

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

① NULL
②

COL1+COL3
80
10
60

③ 150
④

COL1+COL3
50
NULL
NULL

정답: ④

🧸 **쉬운 해설:**
NULL이 하나라도 있으면 결과도 NULL이야!

📚 **전문 해설:**

- 1행: 30 + 20 = 50
- 2행: NULL + 40 = NULL
- 3행: 50 + NULL = NULL

🧠 **기억법:**
NULL + 값 = NULL

필요 암기카드:

- 🃏 카드 34: COALESCE = NULL 처리

[문제 036]

다음 SQL 문장 중 COLUMN1의 값이 널(NULL)이 아닌 경우를 찾아내는 문장으로 가장 적절한 것은? (ANSI 표준 기준)

① SELECT * FROM MYTABLE WHERE COLUMNI IS NOT NULL
② SELECT * FROM MYTABLE WHERE COLUMNI ◊ NULL
③ SELECT * FROM MYTABLE WHERE COLUMNI != NULL
④ SELECT * FROM MYTABLE WHERE COLUMNI NOT NULL

정답: ①

🧸 **쉬운 해설:**
NULL 비교는 IS NOT NULL로만 가능해!

📚 **전문 해설:**
ANSI SQL 기준에서 NULL은 IS NULL 또는 IS NOT NULL로만 비교합니다.
!= NULL은 오류입니다.

보기 설명:

보기 번호	설명	적절성
①	표준 문법	✅
②	문법 오류	❌
③	비교 불가	❌
④	문법 오류	❌

🧠 **기억법:**
NULL 비교는 IS로만!

필요 암기카드:

- 🃏 카드 34: COALESCE = NULL 처리

[문제 037]
아래와 같은 DDL 문장으로 테이블 생성하고, SQL를를 수행하였을 때 다음 설명 중 옳은 것은?

```
CREATE TABLE 서비스
(
    서비스번호 VARCHAR2(10) PRIMARY KEY,
    서비스명 VARCHAR2(100) NULL,
    개시일자 DATE NOT NULL
);

[SQL]
(가) SELECT * FROM 서비스 WHERE 서비스번호 = 1;
(나) INSERT INTO 서비스 VALUES ('999', '', '2015-11-11');
(다) SELECT * FROM 서비스 WHERE 서비스명 = '';
(라) SELECT * FROM 서비스 WHERE 서비스명 IS NULL;
```

```
① 서비스번호 컬럼에 모든 레코드 중에서 '001'과 같은 숫자형식으로 하나의 레코드만이라도 입력되어 (가)는 오류 없이 실행된다.
② ORACLE에서 (나)과같이 데이터를 입력하였을 때, 서비스명 컬럼에 공백문자 데이터가 입력된다.
③ ORACLE에서 (나)과같이 데이터를 입력하고, (다)과 같이 조회하였을 때, 데이터는 조회된다.
④ SQL Server에서 (나)과같이 데이터를 입력하고, (라)과 같이 조회하였을때, 데이터는 조회되지 않는다.
```

정답: ④

🧸 쉬운 해설:
SQL Server는 빈 문자열을 NULL로 처리해서 조회 안 돼!

📚 전문 해설:
ORACLE은 빈 문자열을 공백으로 처리하지만, SQL Server는 NULL로 인식합니다.
따라서 IS NULL 조건이 맞지 않아 조회되지 않습니다.

보기 설명:

보기 번호	설명	적절성
①	숫자형 비교 오류	❌
②	ORACLE 기준	❌
③	공백 조회 불가	❌
④	SQL Server 기준 정확	✅

🧠 기억법:
빈 문자열 = NULL (SQL Server)

필요 암기카드:

- 🃏 카드 64: NOT NULL = 빈칸 금지

[문제 038]
아래와 같이 월별매출 테이블에 데이터가 입력되어 있다. 다음 중 2014년 11월부터 2015년 03월까지의 매출금액 합계를 출력하는 SQL 문장으로 옳은 것은?

[테이블: 월별매출]

년(PK)	월(PK)	매출금액
2014	01	1000
2014	02	2000
2014	03	3000
2014	11	4000
2014	12	5000
2015	01	6000
2015	02	7000
2015	03	8000
2015	11	9000
2015	12	10000

```
① SELECT SUM(매출금액) AS 매출금액합계
FROM 월별매출
WHERE 년 BETWEEN '2014' AND '2015'
AND   월 BETWEEN '03' AND '12';

② SELECT SUM(매출금액) AS 매출금액합계
FROM 월별매출
WHERE 년 IN ('2014', '2015')
AND   월 IN ('11', '12', '03', '04', '05');

③ SELECT SUM(매출금액) AS 매출금액합계
FROM 월별매출
WHERE (년 = '2014' OR 년 = '2015')
AND (월 BETWEEN '01' AND '03' OR 월 BETWEEN '11' AND '12');

④ SELECT SUM(매출금액) AS 매출금액합계
FROM 월별매출
WHERE 년 = '2014' AND 월 BETWEEN '11' AND '12'
OR    년 = '2015' AND 월 BETWEEN '01' AND '03';
```

정답: ④

🧸 **쉬운 해설:**
연도별로 월 조건을 나눠야 정확하게 구간 잡혀!

📚 **전문 해설:**
①은 월 BETWEEN이 전체 연도에 적용돼서 오류
④는 년도별로 월 조건을 나눠 정확한 구간 지정

🧠 **기억법:**
BETWEEN은 연도/월 분리해서 써야 정확함

필요 암기카드:

- 🃏 카드 22: WHERE = 필터링

```
[문제 039]
아래 테이블 스키마를 참조하여 SQL 문장을 작성하였다. 다음 중 결과가 다른 SQL 문장은?
```

[서비스_가입]

고객ID **서비스ID** **가입일자** **가입 시간**
서비스시작일시 서비스종료일시

(논리)

[SVC_JOIN]

CUST_ID: VARCHAR2(10) NOT NULL **SVC ID VARCHAR2(5) NOT NULL** **JOIN YMD: VARCHAR2(8) NOT NULL** **JOIN HH: VARCHAR2(4) NOT NULL**
SVC START DATE: DATE NULL SVC_END DATE: DATE NULL

(물리)

```
① SELECT SVC_ID, COUNT(*) AS CNT
FROM SVC JOIN
WHERE SVC_END_DATE > TO_DATE('20150101000000', 'YYYYMMDDHH24MISS')
AND SVC_END_DATE <= TO_DATE('20150131235959','YYYYMMDDHH24MISS')
AND CONCAT(JOIN_YMD, JOIN_HH) = '2014120100'
GROUP BY SVC_ID;

② SELECT SVC_ID, COUNT(*) AS CNT
FROM SVC JOIN
WHERE SVC_END_DATE >= TO_DATE('20150101', 'YYYYMMDD')
AND   SVC_END_DATE < TO_DATE('20150201', 'YYYYMMDD')
AND   (JOIN_YMD, JOIN_HH) IN (('20141201', '00'))
GROUP BY SVC_ID;

③ SELECT SVC_ID, COUNT(*) AS CNT
FROM SVC JOIN
WHERE '201501' = TO_CHAR(SVC_END_DATE, 'YYYYMM')
AND JOIN YMD = '20141201'
AND JOIN HH = '00'
GROUP BY SVC_ID;

④ SELECT SVC_ID, COUNT(*) AS CNT
FROM SVC_JOIN
WHERE TO_DATE('201501', 'YYYYMM') = SVC_END_DATE
AND JOIN_YMD||JOIN_HH = '2014120100'
GROUP BY SVC ID
```

정답: ④

🧸 **쉬운 해설:**
④번은 TO_DATE('201501')이 날짜 전체와 일치해야 해서 조건이 너무 엄격해!

📚 **전문 해설:**
①~③은 범위 또는 문자열 비교로 월 단위 조건을 유연하게 처리합니다.
④는 TO_DATE('201501') = SVC_END_DATE 조건이 정확히 2015년 1월 1일 00:00:00과 일치해야 하므로 대부분의 데이터가 제외됩니다.

보기 설명:

보기 번호	설명	적절성
①	범위 조건 + 문자열 연결	✅
②	날짜 범위 + 튜플 비교	✅
③	문자열 변환 비교	✅
④	날짜 전체 일치 조건	❌

🧠 **기억법:**
TO_DATE 비교는 시간까지 포함되므로 조심!

필요 암기카드:

- 🧚 카드 22: WHERE = 필터링
- 🧚 카드 33: COUNT(*) = 전체 행

```
[문제 040]
아래와 같은 내장 함수에 대한 설명 중에서 옳은 것을 모두 묶은 것은?

가) 함수의 입력 행수에 따라 단일행 함수와 다중행 함수로 구분할 수 있다.
나) 단일행 함수는 SELECT, WHERE, ORDER BY, UPDATE의 SET 절에 사용이 가능하다.
다) 1:M 관계의 두 테이블을 조인할 경우 M쪽에 다중행이 출력되므로 단일행 함수는 사용할 수 없다.
라) 단일행 함수는 다중행 함수와 다르게 여러 개의 인수가 입력 되어도 단일 값만을 반환한다.
```

① 가
② 가, 나
③ 가, 나, 라
④ 가, 나, 다, 라

정답: ③

🧸 **쉬운 해설:**
가,나,라까지는 맞지만, 다번은 틀렸어!
단일행 함수는 1:M 조인에서도 사용 가능해!

📚 **전문 해설:**

- 가) 단일행/다중행 함수 구분 가능 → 맞음
- 나) 단일행 함수는 다양한 절에서 사용 가능 → 맞음
- 다) 단일행 함수는 1:M 조인에서도 사용 가능 → 틀림
- 라) 여러 인수 입력해도 단일 값 반환 → 맞음

🧠 **기억법:**
단일행 함수는 어디서든 단일 값 반환!

필요 암기카드:

- 🧚 카드 29: 윈도우 함수 = 행별 분석
- 🧚 카드 34: COALESCE = NULL 처리