한 번에 끝내는 코딩테스트 369

Part4. SQL 기술면접 대비

[SQL 50 문제 해설집]

목차

1. 문제에 사용되는 테이블

[직원 테이블]

직원코드 (ENumber)	부서코드 (DNumber)	직급코드 (RNumber)	직원명 (EName)	주민번호 (ERRN)	주소 (EAddr)	입사일 (StartDate)	퇴사일 (ResignationDate)
E0001		R0001	김이수	850726-1118323	서울시 강남구	2018-01-02 09:00:00+09	
E0002		R0001	김사원	900513-1136345	부산시 북구	2018-01-02 09:00:00+09	
E0003		R0002	이길동	890206-1112632	대전시 서구	2018-01-02 09:00:00+09	

[생산 테이블]

생산코드 (PNumber)	직원코드 (ENumber)	제품코드 (INumber)	생산량 (PCount)	생산일 (PDate)
P1001	E1001	I1001	620	2020-01-03 11:32:22+09
P1002	E1002	I1002	32	2020-02-16 09:11:07+09
P1003	E1003	I1003	323	2020-02-21 10:07:35+09

[주문 테이블]

주문코드 (ONumber)	고객코드 (CNumber)	생산코드 (PNumber)	주문일자 (ODate)
O1001	C1001	P1001	2021-01-07 09:08:32+09
O1002	C1002	P1002	2021-01-08 10:31:27+09
O1003	C1003	P1004	2021-01-12 09:43:31+09

[반품 테이블]

반품코드 (RNumber)	주문코드 (ONumber)	수량 (RCount)	반품사유코드 (RRNumber)	반품일 (RDate)
R1001	O1002	5	RR0001	2022-01-03 13:32:07+09
R1002	O1007	700	RR0004	2022-01-03 15:17:47+09
R1003	O1009	210	RR0002	2022-01-03 16:22:16+09

[고객 테이블]

고객코드 (CNumber)	고객명 (CName)	고객주소 (CAddr)	계약일 (Contractdate)
C1001	대한문방구	서울시 강남구	2019-01-23 13:32:12+09
C1002	용호문방구	서울시 용산구	2019-01-27 14:10:01+09
C1003	오랜문방구	울산시 중구	2019-02-03 16:08:06+09

[제품 테이블]

제품코드 (INumber)	제품명 (IName)	단가 (Price)	등록일 (RegisterDate)
I1001	가위	600	2019-01-02 14:32:23+09
I1002	풀	500	2019-01-13 15:33:23+09
I1003	공책	1000	2019-01-23 16:34:23+09

[반품사유 테이블]

반품사유코드 (RRNumber)	반품사유명 (RReason)
RR0001	불량
RR0002	단순변심
RR0003	환불
RR0004	배송오류

[직급 테이블]

직급코드 (RNumber)	직급명 (RName)
R0001	사원
R0002	주임
R0003	대리

[부서 테이블]

부서코드 (DNumber)	부서명 (DName)
D1001	문구생산부
D2001	가구생산부
D3001	악세사리생산부
D4001	전자기기생산부
D5001	음료생산부

2. 문제 해설

1. (SFWO)

문구생산부와 가구생산부의 직원정보를 확인하려한다.

위에 해당하는 직원들의 직원명과 입사일을 입사일이 빠른 순서대로 출력하세요.

(문구생산부의 DNumber는 'D1001', 가구생산부는 'D2001'이며 입사일은 연,월,일까지만 출력되어야 한다)

<정답>

<결과 화면>

	직원명 character varying(5)	입사일 text
1	박천이	2001-03-13
2	박총괄	2002-02-06
3	이과장	2010-02-03
4	이해오	2010-05-06
5	이혜연	2013-10-09

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tem.EName AS 직원명, TO_CHAR(tem.StartDate,'YYYY-MM-DD') AS 입사일 ----------①
FROM tEmployee AS tem
WHERE tem.DNumber IN ('D1001', 'D2001') ----------②
ORDER BY tem.StartDate ----------③
```

<해설>

① EName(직원명)을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져오고 StartDate(입사일)을 출력하기 위해 직원테이블인 tEmployee에서 입사일 컬럼인 StartDate를 가져온 후 TO_CHAR함수를 사용하여 연-월-일 형태로 형식을 변경한다.

② 문제의 조건인 'D1001', 'D2001'에 해당하는 값을 가져오기 위해 직원테이블인 tEmployee에서 관련 컬럼인 DNumber(부서코드)와 비교한 후 여러 조건을 비교하기 위해 다중 행 연산자인 IN을 사용한다.

③ 정렬 기준인 StartDate(입사일)를 오름차 순 정렬을 하기 위해 ORDER BY를 사용한다.

<함수설명>

TO_CHAR(tem.StartDate,'YYYY-MM-DD')

- 첫번째 파라미터인 StartDate는 Timestamp타입이고, 두번째 파라미터인 'YYYY-MM-DD' 포맷에 맞춰 '년도-월-일'의 텍스트타입으로 리턴한다.

2. (SFWO)

2020년 크리스마스부터 입사일이 만 2년이 넘어가는 사람에게 보너스를 지급하려고 한다.

위 조건에 해당하는 직원의 직원명과 입사일을 출력하시오.

(단, 정렬은 고려하지 않는다)

<정답>

<결과 화면>

	직원명 character varying(5)	입사일 timestamp with time zone
1	김이수	2018-01-02 09:00:00+09
2	김사원	2018-01-02 09:00:00+09
3	이길동	2018-01-02 09:00:00+09
4	서준수	2018-01-02 09:00:00+09
5	이기영	2018-01-02 09:00:00+09

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tem.EName AS 직원명, tem.StartDate AS 입사일 ---------- ①
FROM tEmployee AS tem
WHERE tem.StartDate < CAST('2020-12-25' AS TIMESTAMP) - CAST('2 year' AS INTERVAL)---- ②
```

<해설>

① EName(직원명)을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져오고 StartDate(입사일)을 출력하기 위해 직원테이블인 tEmployee에서 입사일 컬럼인 StartDate를 가져온다.

② 문제의 조건인 2020년 크리스마스(2020-12-25)를 기준으로 입사일이 2년을 넘어간 조건을 찾기 위해 CAST 함수를 사용해서 해당 날짜에서 2년을 뺀 값 보다 작은 값을 직원테이블인 tEmployee에서 관련 컬럼인 StartDate(입사일)와 비교한다.

<함수설명>

CAST('2020-12-25'as timestamp)

- 텍스트타입인 '2020-12-25'를 TIMESTAMP 타입으로 형 변환한다.

CAST('2 year' AS INTERVAL)

- 텍스트타입인 '2 year'을 INTERVAL 타입으로 형 변환한다.

3. (GROUP BY)

생산량 조정을 위해 2020년 2월의 총 생산량을 알려고 한다.

해당 월에 생산된 제품들의 코드와 해당 제품들의 총 생산량을 출력하시오.

<정답>

<결과 화면>

	제품코드 character varying(5)	총_생산량 bigint
1	I3005	234
2	I4004	235
3	I2005	343
4	I3003	540
5	I1002	32

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tpr.INumber AS 제품코드, SUM(tpr.PCount) AS 총_생산량 ]  -------- ①
FROM tProduction AS tpr
WHERE tpr.PDate BETWEEN CAST('2020-02-01' AS TIMESTAMP) AND CAST('2020-03-01' AS TIMESTAMP) ---- ②
GROUP BY tpr.INumber  ---------- ③
```

<해설>

① INumber(제품코드)를 출력하기 위해 생산테이블인 tProduction에서 제품코드 컬럼인 INumber을 가져오고 총 생산량을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온다. 합을 구하기 위해 SUM함수를 사용한다.

② 문제의 조건인 2020년 2월의 값을 가져오기 위해 BETWEEN과 CAST함수를 사용하여 날짜가 2020-02-01보다 크고 2020-03-01보다 작은 값을 생산테이블인 tProduction에서 관련 컬럼인 PDate(생산날짜)와 비교한다

③ INumber(제품코드)를 GROUP BY를 사용하여 그룹화 한 후 해당 컬럼을 기준으로 ①에서 SUM함수를 사용하여 총 생산량을 구한다

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다.

CAST('2020-02-01' AS TIMESTAMP), CAST('2020-03-01' AS TIMESTAMP)

- 텍스트타입인 '2020-02-01'과 '2020-03-01'를 TIMESTAMP 타입으로 형 변환한다.

4. (GROUP BY)

가구류 제품들의 선호도 조사를 위하여 고객들이 가구류 제품들의 주문을 몇 번 했는지 고객코드별로 출력하시오. (가구류의 생산코드는 P2~로 시작한다)

<정답>

<결과 화면>

	고객코드 character varying(5)	제품주문횟수 bigint
1	C2001	2
2	C2002	4
3	C2003	4
4	C2004	2
5	C2005	1

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tor.CNumber AS 고객코드 , COUNT(tor.PNumber) AS 제품주문횟수 -------- ①
FROM tOrder AS tor
WHERE SUBSTRING(tor.PNumber,1,2) = 'P2'  -------- ②
GROUP BY tor.CNumber  -------- ③
```

<해설>

① CNumber(고객코드)를 출력하기 위해 주문테이블인 tOrder에서 고객코드 컬럼인 CNumber를 가져오고 제품주문횟수를 출력하기 위해 주문테이블인 tOrder에서 생산코드 컬럼인 PNumber를 가져온다. 개수를 구하기 위해 COUNT함수를 사용한다.

② 문제의 조건인 P2와 같은값을 가져오기 위해 SUBSTRING함수를 이용하여 앞 두자리를 잘라 주문테이블인 tOrder에서 관련 컬럼인 PNumber와 비교한다

③ GROUP BY를 사용하여 그룹화 한 CNumber(고객코드)를 기준으로 ①에서 COUNT함수를 사용하여 제품주문횟수를 구한다

<함수설명>

COUNT(tor.PNumber)

- PNumber의 값이 NULL인 값을 제외한 데이터의 개수를 리턴한다

SUBSTRING(tor.PNumber, 1, 2)

- 첫번째 파라미터는 문자열이고 두번째 파라미터는 시작 인덱스입니다. 세번째 파라미터는 가져 올 개수입니다. PNumber에서 1번째부터 2개까지 문자를 가져옵니다.

5. (GROUP BY, HAVING)

2020년 1월의 성실직원을 뽑기 위해 성실직원의 기준인 생산량 500개 이상을 달성한 인원들의 직원코드와 총 생산량을 출력하시오.

<정답>

<결과 화면>

	직원코드 character varying(5)	총_생산량 bigint
1	E5002	2323
2	E5001	1215
3	E1001	620
4	E3003	923
5	E3002	632

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tpr.ENumber AS 직원코드, SUM(tpr.PCount) AS 총_생산량 ]  -------- ①
FROM tProduction AS tpr
WHERE tpr.PDate BETWEEN CAST('2020-01-01' AS TIMESTAMP) AND CAST('2020-02-01' AS TIMESTAMP) --- ②
GROUP BY tpr.ENumber -------- ③
HAVING SUM(tpr.PCount) >= 500  ------------ ④
```

<해설>

① ENumber(직원코드)를 출력하기 위해 직원테이블인 tEmployee에서 직원코드 컬럼인 ENumber를 가져온 후 총 생산량을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져오고 합을 구하기 위해 SUM함수를 사용한다.

② 문제의 조건인 2020년 1월의 값을 가져오기 위해 BETWEEN과 CAST함수를 사용하여 날짜가 2020-01-01보다 크고 2020-02-01보다 작은 값을 생산테이블인 tProduction에서 관련 컬럼인 PDate(생산일자)와 비교한다

③ PCount(생산량)를 GROUP BY를 사용하여 그룹화 한다.
그룹화 한 PCount(생산량)를 기준으로 ①에서 SUM함수를 사용하여 총 생산량을 구한다.

④ 문제의 조건인 500보다 크거나 같은 값을 집계함수인 SUM의 결과값에 조건을 걸기 위해서 HAVING을 사용한다.

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

CAST('2020-01-01' AS TIMESTAMP), CAST('2020-02-01' AS TIMESTAMP)

- 텍스트타입인 '2020-01-01'과 '2020-02-01'를 TIMESTAMP 타입으로 형 변환한다..

6. (CASE WHEN)

판매 가능한 제품들의 재고 파악을 위해 2020년 2월에 생산된 양을 확인하려하는데 우선 문구류 제품들을 먼저 파악하려 한다. 해당 제품명과 제품들의 총 생산량을 출력하시오.

(문구류의 제품코드(INumber)는 I100(1~5)이며 1번은 가위, 2번은 풀, 3번은 공책, 4번은 볼펜, 5번은 지우개이다, ex - I1001 = 가위)

<정답>

<결과 화면>

	제품명 text	총_생산량 bigint
1	풀	32
2	공책	323
3	볼펜	1199

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT                                                    ---------- ①
    CASE
        WHEN tpr.INumber = 'I1001' THEN '가위'
        WHEN tpr.INumber = 'I1002' THEN '풀'
        WHEN tpr.INumber = 'I1003' THEN '공책'
        WHEN tpr.INumber = 'I1004' THEN '볼펜'
        WHEN tpr.INumber = 'I1005' THEN '지우개'
        END AS 제품명
        , SUM(tpr.PCount) AS 총_생산량
FROM tProduction AS tpr
WHERE TO_CHAR(tpr.PDate,'YYYYMM') = '202002'    ---------- ②
    AND SUBSTRING(tpr.INumber, 1, 2) = 'I1'
GROUP BY tpr.INumber     ---------- ③
```

<해설>

① 제품명을 출력하기 위해 생산테이블인 tProduction에서 제품코드 컬럼인 INumber을 가져오고 문제의 조건인 제품코드를 기준으로 제품명을 출력하기 위해 제품코드를 기준으로 CASE-WHEN 함수를 사용한다.
총 생산량을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온다.
합을 구하기 위해 SUM함수를 사용한다.

② 문제의 조건인 2020-02과 같은 값을 가져오기 위해 생산테이블인 tProduction에서 관련 컬럼인 PDate와 비교한다.
문제의 조건인 문구류와 같은값을 가져오기 위해 SUBSTRING함수를 이용하여 앞 두 자리를 잘라

생산테이블인 tProduction에서 관련 컬럼인 INumber(제품코드)와 비교한다

③ INumber(제품코드)를 GROUP BY를 사용하여 그룹화 한다.
그룹화 한 PCount(생산량)를 기준으로 ①에서 SUM 함수를 사용하여 총 생산량을 구한다.

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

TO_CHAR(tpr.PDate,'YYYYMM')

- 첫번째 파라미터인 PDate 는 Timestamp타입이고, 두번째 파라미터인 'YYYYMM' 포맷에 맞춰 '연월'의 텍스트타입으로 리턴한다.

SUBSTRING(tpr.INumber, 1, 2)

- 첫번째 파라미터는 문자열이고 두번째 파라미터는 시작 인덱스입니다. 세번째 파라미터는 가져 올 개수입니다. INumber에서 1번째부터 2개까지 문자를 가져옵니다.

7. (UNION)

제품이 한번이라도 주문됐거나, 반품이 한번이라도 발생한 월의 정보를 알아보려고한다.

위, 조건에 해당하는 월들을 중복을 제거하고 출력하시오.

<정답>

<결과 화면>

	주문_및_반품월 text
1	02
2	03
3	05
4	01
5	07

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT TO_CHAR(tor.ODate, 'MM') AS 주문_및_반품월 ---------- ①
FROM tOrder AS tor
GROUP BY TO_CHAR(tor.ODate, 'MM')  ---------- ②
UNION  ---------- ③
SELECT TO_CHAR(tre.RDate, 'MM')  ---------- ④
FROM tReturn AS tre
GROUP BY TO_CHAR(tre.RDate, 'MM') ---------- ⑤
```

<해설>

① ODate(주문일)를 출력하기 위해 주문테이블인 tOrder에서 주문일 컬럼인 ODate를 가져오고 문제의 조건인 월별로 출력하기 위해 TO_CHAR함수를 사용하여 월 형태로 형식을 변경한다.

② TO_CHAR함수를 사용하여 월 형태로 형식을 변경한 Odate(주문일)를 GROUP BY를 사용하여 그룹화한다

③ 위의 쿼리와 아래의 쿼리를 UNION을 사용하여 합친다.

④ RDate(반품일)를 출력하기 위해 반품테이블인 tReturn에서 반품일 컬럼인 RDate를 가져온다. 문제의 조건인 월별로 출력하기 위해 TO_CHAR함수를 사용하여 월 형태로 형식을 변경한다.

⑤ TO_CHAR함수를 사용하여 월 형태로 형식을 변경한 RDate(반품일)를 GROUP BY를 사용하여 그룹화한다.

<함수설명>

TO_CHAR(tor.ODate, 'MM') , TO_CHAR(RDate,'MM')

- 첫번째 파라미터인 ODate, RDate는 Timestamp타입이고, 두번째 파라미터인 'MM' 포맷에 맞춰 '월'의 텍스트타입으로 리턴한다.

8. (UNION ALL)

여태까지 회사에 입사했던 사람들의 총 인원 수와 연도별 입사한 직원 수를 출력하시오.

<정답>

<결과 화면>

	입사년도 text	입사한_직원_수 bigint
1	총 인원 수 :	57
2	2006	3
3	2002	1
4	2014	3
5	2003	1

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT '총 인원 수 :' AS 입사년도, COUNT(tem1.*) AS 입사한_직원_수 --------- ①
FROM tEmployee AS tem1
UNION ALL ------------ ②
SELECT TO_CHAR(tem2.StartDate,'YYYY'), COUNT(tem2.*) ---------- ③
FROM tEmployee AS tem2
GROUP BY TO_CHAR(tem2.StartDate,'YYYY') ---------------------------- ④
```

<해설>

① 총 직원수를 출력하기 위해 직원테이블인 tEmployee에서 개수를 구하기 위해 COUNT함수를 사용한다.

② 위의 쿼리와 아래의 쿼리를 UNION ALL을 사용하여 중복된 값도 출력되게 합친다.

③ 입사년도를 출력하기 위해 직원테이블인 tEmployee에서 입사일 컬럼인 StartDate를 가져오고 TO_CHAR함수를 사용하여 연 형태로 형식을 변경한다.
직원수를 구하기 위해 직원테이블인 tEmployee에서 개수를 구하기 위해 COUNT함수를 사용한다.

④ TO_CHAR함수를 사용하여 연 형태로 형식을 변경한 StartDate를 GROUP BY를 사용하여 그룹화하여 ①에서 COUNT함수를 사용하여 입사한 총 직원 수를 구한다

<함수설명>

COUNT(tem1.*)

- tem1.* 의 값이 NULL인 값을 제외한 데이터의 개수를 리턴한다.

TO_CHAR(tem2.StartDate,'YYYY')

- 첫번째 파라미터인 StartDate는 Timestamp타입이고, 두번째 파라미터인 'YYYY' 포맷에 맞춰 '연도'의 텍스트타입으로 리턴한다.

9. (서브쿼리 COLUMN)

회사에서 제공해주는 기숙사에 머무를 수 있는 인원의 제한을 위해 부서와 직급, 그리고 현재 거주지를 따져 제한하려고 한다. 이에 따라 부서명과 직급명, 직원명 그리고 현재 직원의 주소를 출력하시오

(부서코드, 직급코드가 아닌 부서명, 직급명으로 출력되어야 한다.)

<정답>

<결과 화면>

부서명 character varying(10)	직급명 character varying(2)	직원명 character varying(5)	직원주소 character varying(10)
	대리	이기영	울산시 동구
문구생산부	사원	박하나	서울시 강남구
문구생산부	주임	김문구	부산시 남구
문구생산부	대리	이과장	부산시 중구
문구생산부	대리	이해오	서울시 도봉구

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT (SELECT tde.DName FROM tDepartment AS tde WHERE tde.DNumber = tem.DNumber) AS 부서명----①
        , tem.EName AS 직원명        -------- ②
        , (SELECT tra.RName FROM tRank AS tra WHERE tra.RNumber = tem.RNumber) AS 직급명----------③
        , tem.EAddr AS 직원주소      --------- ④
FROM tEmployee AS tem
```

<해설>

① tEmployee(직원테이블)에 존재하는 DNumber(부서코드)를 이용하여 DName(부서명)을 출력하기 위해 SELECT절에 서브쿼리를 사용한다.

② EName(직원명)을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져온다.

③ tEmployee(직원테이블)에 존재하는 RNumber(직급코드)를 이용하여 RName(직급명)을 출력하기 위해 SELECT절에 서브쿼리를 사용한다.

④ EAddr(직원주소)를 출력하기 위해 직원테이블인 tEmployee에서 직원주소 컬럼인 EAddr을 가져온다.

10. (서브쿼리 WHERE)

회사 내 전 직원들의 평균 생산량보다 한번이라도 많이 생산한 직원들에게는 상여금을 주려한다.

이에 해당하는 직원명을 출력하시오 (생산량은 tProduction 테이블의 PCount이다.)

<정답>

<결과 화면>

	직원명 character varying(5)
1	김문구
2	이해오
3	김이해
4	박지원
5	김지오

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tem.EName AS 직원명 ] --------- ①
FROM tEmployee AS tem
WHERE tem.ENumber IN ( ------------------ ②
                SELECT tpr1.ENumber ] ---------- ③
                FROM tProduction AS tpr1 ]
                WHERE tpr1.PCount > (  ------------------ ④
                                SELECT AVG(tpr2.PCount) ] ------------- ⑤
                                FROM tProduction AS tpr2 ]
                            )
                    )
```

<해설>

① EName(직원명)을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져온다.

② 문제의 조건인 평균 생산량보다 생산량이 많은 직원의 직원명을 찾기 위해 ENumber(직원코드)를 비교한 후 여러 값을 비교하기 위해 다중 행 연산자인 IN을 사용한다.

③ ENumber(직원코드)를 출력하기 위해 생산테이블인 tProduction 에서 직원코드 컬럼인 ENumber를 가져온다.

④ 문제의 조건인 평균 생산량 보다 많은 직원의 직원코드를 찾기위해 생산량을 비교한다.

⑤ PCount(생산량)를 출력하기 위해 직원테이블인 tEmployee에서 생산량 컬럼인 PCount를 가져온다. 평균을 구하기 위해 AVG함수를 사용한다.

동명이인의 경우가 있을 수 있으므로 고유번호(기본키)인 직원 코드로 비교를 진행한다.

<함수설명>

AVG(tpr2.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 다하고 개수만큼 나누는 계산을 한 후 계산된 값을 리턴.

11. (서브쿼리 WHERE)

2022년 1월에 반품이 발생한 직원을 확인하여 패널티를 부과하려 한다. 이에 해당하는 직원명을 출력하시오.

<정답>

<결과 화면>

	직원이름 character varying(5)
1	박하나
2	김문구
3	박지원
4	김구성
5	박천이

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tem.EName AS 직원명 ------- ①
FROM tEmployee AS tem
WHERE tem.ENumber IN ( ----------②
        SELECT tpr.ENumber ---------- ③
        FROM tProduction AS tpr
        WHERE tpr.PNumber IN (
                SELECT tor.PNumber ---④
                FROM tOrder AS tor
                WHERE tor.ONumber IN (
                        SELECT tre.ONumber
                        FROM tReturn AS tre ---- ⑤
                        WHERE TO_CHAR(tre.RDate,'YYYY-MM') = '2022-01')))
```

<해설>

① 직원명을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져온다.

② 문제의 조건인 반품이 발생한 직원에 해당하는 값을 가져오기 위해 직원테이블인 tEmployee에서 관련 컬럼인 ENumber(직원코드)와 비교하고 여러 조건을 비교하기 위해 다중행 연산자인 IN을 사용한다.

③ 문제의 조건인 반품이 발생한 주문 내역에 해당하는 값을 가져오기 위해 관련 컬럼인 PNumber(생산코드)와 비교한 후 tpr(별칭) 서브쿼리에서 결과물을 출력한 ENumber(직원코드) 컬럼을 가져온다

④ 문제의 조건인 반품 발생 내역에 해당하는 값을 가져오기 위해 관련 컬럼인 ONumber(주문코드)와 비교한 후 tor(별칭) 서브쿼리에서 결과물을 출력한 PNumber(생산코드) 컬럼을 가져온다

⑤ 문제의 조건인 2022년 1월의 반품 내역에 해당하는 값을 가져오기 위해 반품 테이블인 tReturn에서 관련 컬럼인 RDate(반품일자)를 비교 한 후 tre(별칭) 서브쿼리에서 결과물을 출력한 ONumber(주문코드) 컬럼을 가져온다.

<함수설명>

TO_CHAR(tre.RDate,'YYYY-MM')

- 첫번째 파라미터인 RDate 는 Timestamp 타입이고, 두번째 파라미터인 'YYYY-MM' 포맷에 맞춰 '년도-월'의 텍스트타입으로 리턴한다.

12. (서브쿼리 FROM)

2021년부터 판매가 시작됨에 따라 지난 1년(20년 1월 1일 ~ 20년 12월 31일) 동안 생산된 제품들의 제품코드와 총 생산량을 생산량이 많은 순으로 확인하려한다. 위의 조건에 맞춰 출력하시오.

<정답>

<결과 화면>

	제품코드 character varying(5)	총_생산량 bigint
1	I5002	14243
2	I5001	13229
3	I5003	11027
4	I5004	9159
5	I5005	5526

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tBase.INumber AS 제품코드, tBase.PCount AS 총_생산량  ┐ ---------- ①
FROM                                                          ┘
(
    SELECT tpr.INumber, SUM(tpr.PCount) AS PCount  ┐ --------- ②
    FROM tProduction AS tpr                        ┘
    WHERE tpr.PDate BETWEEN CAST('2020-01-01' AS TIMESTAMP)  ┐ -------- ③
                        AND CAST('2021-01-01' AS TIMESTAMP)  ┘
    GROUP BY tpr.INumber    ------------------ ④
) AS tBase
ORDER BY tBase.PCount DESC  ---------------- ⑤
```

<해설>

① 서브쿼리로 구한 결과물인 tBase(별칭)에서 INumber(제품코드)와 PCount(생산량)을 가져온다.

② INumber(제품코드)를 출력하기 위해 생산테이블인 tProduction에서 제품코드 컬럼인 INumber를 가져온다.
생산량을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온다.
합을 구하기 위해 SUM함수를 사용한다

③ 문제의 조건인 2020년 1월부터 2020년 12월의 값을 가져오기 위해 BETWEEN과 CAST함수를 사용하여 날짜가 2020-01-01보다 크고 2021-01-01보다 작은 값을 생산테이블인 tProduction에서 관련 컬럼인 PDate(생산일자)와 비교한다

④ INumber(제품코드)를 GROUP BY를 사용하여 그룹화 한다.
그룹화 한 INumber(제품코드)를 기준으로 ①에서 SUM함수를 사용하여 생산량을 구한다

⑤ 정렬 기준인 총 생산량을 내림차 순 정렬을 하기 위해 ORDER BY ~ DESC를 사용한다

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

CAST('2020-01-01' AS TIMESTAMP), CAST('2021-01-01' AS TIMESTAMP)

- 텍스트타입인 '2020-01-01'과 '2021-01-01'을 TIMESTAMP타입으로 형 변환한다.

13. (RANK)

회사 내 우수직원을 생산량이 많은 직원을 기준으로 하여 상위 10명까지 뽑아 상여금을 주려한다.
이에 해당하는 직원들의 직원코드와 총 생산량을 상위 10명까지 순위를 매겨 출력하시오.
(만약 공동순위(ex. 공동 1등)가 있다면 다음 순위는 중복 된 순위의 수 만큼 떨어진다.)

<정답>

<결과 화면>

	직원코드 character varying(5)	총_생산량 bigint	총_생산량_순위 bigint
1	E5002	13334	1
2	E5010	13334	1
3	E5005	7421	3
4	E5004	5080	4
5	E5008	4653	5

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT   tpr.ENumber AS 직원코드 ------- ①
   , SUM(tpr.PCount) AS 총_생산량
   , RANK() OVER(ORDER BY SUM(tpr.PCount) DESC) AS 총_생산량_순위 ---------- ②
FROM tProduction AS tpr
GROUP BY tpr.ENumber          -------- ③
LIMIT 10                      -------- ④
```

<해설>

① 직원코드를 출력하기 위해 생산테이블인 tProduction에서 직원코드 컬럼인 ENumber를 가져오고 총 생산량을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온다. 합을 구하기 위해 SUM함수를 사용한다

② 문제의 조건에 맞는 결과인 생산량이 많은 직원을 출력하되 동순위가 두 명이상 일 수 있기에 Rank를 사용하였고 ORDER BY에 존재하는 SUM(PCount) (총 판매량)을 기준으로 생산량이 많은 순으로 출력하기 위해 DESC(내림차 순)하여 순위를 출력해준다

③ 그룹화 한 ENumber(직원코드)를 기준으로 ①에서 SUM함수를 사용하여 총 생산량을 구한다

④ 상위 10명까지 출력을 해 주어야 하기 때문에 LIMIT 10을 사용해준다.

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다.

LIMIT 10

- LIMIT로 정한 수인 10 만큼 반환하게끔 제한 할 수가 있다

14. (ROW_NUM)

현재 판매하는 제품들 중 장농의 인기가 많아져 판매 가능한 장농의 재고를 확인하기 위하여
장농 생산이력을 전부 출력하되 생산량이 높은순서대로 생산한 직원의 코드와 제품코드, 생산량을 순위를 매겨 출력하시오.
(공동 순위가 나오지 않게 출력 해야 하며 또한 동률일경우 직원코드(ENumber)가 작은 코드가 우선순위를 가지도록 한다, 장농의 제품코드는 I2003번이다)

<정답>

<결과 화면>

	직원코드 character varying(5)	제품코드 character varying(5)	생산량 integer	생산량_순위 bigint
1	E2010	I2003	543	1
2	E2007	I2003	42	2
3	E2002	I2003	22	3
4	E2003	I2003	22	4
5	E2010	I2003	11	5

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tpr.ENumber AS 직원코드 ┐-------- ①
    , tpr.INumber AS 제품코드  │
    , tpr.PCount AS 생산량     ┘
    , ROW_NUMBER() OVER(PARTITION BY tpr.INumber                    ┐------②
                        ORDER BY tpr.PCount DESC, tpr.ENumber ASC) AS 생산량_순위 ┘
FROM tProduction AS tpr
WHERE tpr.INumber = 'I2003'  -------------- ③
```

<해설>

① ENumber(직원코드)를 출력하기 위해 생산테이블인 tProduction에서 직원코드 컬럼인 ENumber를 가져오고 INumber(제품코드)를 출력하기 위해 생산테이블인 tProduction에서 제품코드 컬럼인 INumber를 가져오고 PCount(생산량)를 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온다.

② 구해진 생산량을 통하여 순위를 출력하는데 동률일 경우 직원코드가 작은 코드에 우선순위를 두기 위해 ROW_NUMBER를 사용하였고 PARTITION BY에 제품코드 컬럼인 INumber를 사용함으로써 각 제품코드마다 ORDER BY에 존재하는 PCount(생산량)이 내림차 순, ENumber(직원코드)가 오름차 순 정렬하여 순위를 매겨준다.

③ 문제의 조건인 I2003과 같은값을 가져오기 위해 생산테이블인 tProduction에서 관련 컬럼인 INumber(제품코드)와 비교한다

15. (DENSE_RANK)

부서별로 연령의 평균을 파악하기 위해서 각 부서마다 속해있는 직원들의 나이로 출생연도를 알아보려한다.
이를 위해 부서명, 직원명, 출생연도를 출력하되 각 부서의 직원들을 출생연도가 빠른 순으로 순위를 매겨라.
(공동순위는 동일하게 부여하고 그 다음 순위는 공동 순위 다음 번호로 순위가 출력되어야 하며 출생연도는 tEmplyee의 ERRN의 앞 2자리로 비교하여 출력할 수 있다. D1001 부서는 문구생산부, D2001은 가구생산부, D3001은 액세서리생산부, D4001은 전자기기생산부, D5001은 음료생산부이다.)

<정답>

<결과 화면>

	부서명 text	직원명 character varying(5)	출생연도 text	출생연도_순위 bigint
1	문구생산부	박충괄	67	1
7	문구생산부	박지원	91	7
8	문구생산부	권종인	91	7
9	문구생산부	김지오	94	8
10	문구생산부	정정인	95	9

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT
    CASE
        WHEN tem.DNumber = 'D1001' THEN '문구생산부'
        WHEN tem.DNumber = 'D2001' THEN '가구생산부'
        WHEN tem.DNumber = 'D3001' THEN '악세사리생산부'
        WHEN tem.DNumber = 'D4001' THEN '전자기기생산부'--------- ①
        WHEN tem.DNumber = 'D5001' THEN '음료생산부'
        ELSE '부서없음'
        END AS 부서명
    , tem.EName AS 직원명                                        --------- ②
    , SUBSTRING(tem.ERRN, 1, 2) AS 출생연도
    , DENSE_RANK() OVER(PARTITION BY tem.DNumber ORDER BY SUBSTRING(tem.ERRN, 1, 2)) AS 출생연도_순
위------ ③
FROM tEmployee AS tem
```

<해설>

① 부서명을 출력하기 위해 직원테이블인 tEmployee에서 부서코드 컬럼인 DNumber를 가져온 후에 CASE WHEN을 이용하여 부서코드마다 부서명을 붙여준다.

② 직원명을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져오고 출생연도를 출력하기 위해 주민번호 컬럼인 ERRN을 가져온 후에 SUBSTRING 함수를 이용하여 앞 두자리를 끊어준다.

③ 문제의 조건에 맞는 결과인 출생연도가 빠른 순을 출력하되 공동순위가 있을때는 다음 번호로 순위가 되어야 하기에 DENSE_RANK 함수를 쓰고 PARTITION BY에 부서코드 컬럼인 DNumber를 이용함으로써 직원들의 연령을 부서별로 파악할 수 있게 하였고 ORDER BY에 존재하는 SUBSTRING(ERRN, 1, 2) (출생연도)을 기준으로 하여 순위를 출력해준다.

<함수설명>

SUBSTRING(tem.ERRN, 1, 2)

- 첫번째 파라미터는 문자열이고 두번째 파라미터는 시작 인덱스입니다. 세번째 파라미터는 가져 올 개수입니다. ERRN에서 1번째부터 2개까지 문자를 가져옵니다.

16. (INNER JOIN)

현재까지 입사했던 모든 직원들의 직원코드, 부서명, 직원명, 직급명, 입사일, 퇴사일을 출력하시오
(부서와 직급의 경우는 코드가 아닌 부서명과 직급명으로 출력하고 입사일과 퇴사일은 연,월,일만 출력되어야한다)

<정답>

<결과 화면>

	직원코드 character varying(5)	부서명 character varying(10)	직원명 character varying(5)	직급명 character varying(2)	입사일 text	퇴사일 text
1	E1001	문구생산부	박하나	사원	2016-01-02	
2	E1002	문구생산부	김문구	주임	2014-03-05	
			⋮			
5	E1005	문구생산부	박총괄	부장	2002-02-06	
6	E1006	문구생산부	김이해	사원	2016-05-02	2021-02-25
7	E1007	문구생산부	박지원	주임	2015-12-11	2021-01-13

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tem.ENumber AS 직원코드, tde.DName AS 부서명, tem.EName AS 직원명, tra.RName AS 직급명 ---①
     , TO_CHAR(tem.Startdate,'YYYY-MM-DD') AS 입사일
     , TO_CHAR(tem.Resignationdate,'YYYY-MM-DD') AS 퇴사일
FROM tEmployee AS tem
JOIN tDepartment AS tde ------- ②
ON tem.DNumber = tde.DNumber
JOIN tRank AS tra ------- ③
ON tem.RNumber = tra.RNumber
```

<해설>

① ENumber(직원코드)를 출력하기 위해 직원테이블인 tEmployee에서 직원코드 컬럼인 ENumber를 가져오고 DName(부서명)을 출력하기 위해 부서테이블인 tDepartment에서 부서명 컬럼인 DName을 가져오고 EName(직원명)을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져오고 RName(직급명)을 출력하기 위해 직급테이블인 tRank에서 직급명 컬럼인 RName을 가져오고 입사일을 출력하기 위해 직원테이블인 tEmployee에서 입사일 컬럼인 StartDate를 가져오고 퇴사일을 출력하기 위해 직원테이블인 tEmployee에서 퇴사일 컬럼인 Resignationdate를 가져온다.
TO_CHAR함수를 사용하여 연-월-일 형태로 형식을 변경한다.

② tDepartment(부서) 테이블에 있는 DName(부서명)을 가져오기 위해 tEmployee(직원)테이블의 DNumber(부서코드)와 tDepartment(부서)테이블의 DNumber(부서코드)를 JOIN한다.

③ tRank(직급) 테이블에 있는 RName(직급명)을 가져오기 위해 tEmployee(직원)테이블의 RNumber(직급코드)와 tRank(직급)테이블의 RNumber(직급코드)를 JOIN한다.

<함수설명>

TO_CHAR(tem.StartDate,'YYYY-MM-DD')

- 첫번째 파라미터인 StartDate는 Timestamp타입이고, 두번째 파라미터인 'YYYY-MM-DD' 포맷에 맞춰 '연-월-일'의 텍스트타입으로 리턴한다.

17. (INNER JOIN)

2021년 1월의 전자기기류 판매금 정산을 위해 전자기기류 제품들의 제품명과 해당 제품의 총 판매량을 출력하세요. (전자기기류의 제품코드는 INumber 번호가 I4로 시작한다.)

<정답>

<결과 화면>

	제품이름 character varying(20)	생산량 bigint
1	노트북	605
2	디지털카메라	93
3	모니터	110
4	컴퓨터	288

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tit.IName AS 제품명, SUM(tpr.PCount) AS 총_판매량 --------- ①
FROM tOrder AS tor
JOIN tProduction AS tpr ---------- ②
ON tor.PNumber = tpr.PNumber
JOIN tItem AS tit ---------- ③
ON tpr.INumber = tit.INumber
WHERE TO_CHAR(tor.ODate,'YYYY-MM') = '2021-01' AND SUBSTRING(tit.INumber,1,2) = 'I4' --------- ④
GROUP BY tit.IName --------- ⑤
```

<해설>

① 제품명을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 총 판매량을 출력하기 위해 제품테이블인 tProduction에서 판매량과 관련된 컬럼인 PCount을 가져온 후 합을 구하기 위해 SUM함수를 사용한다.

② tProduction(생산) 테이블에 있는 PCount(생산량)을 가져오기 위해 tOrder(주문) 테이블의 PNumber (생산코드)와 tProduction(생산)테이블의 PNumber(생산코드)를 JOIN한다.

③ tItem(제품) 테이블에 있는 IName(제품명)을 가져오기 위해 tProduction(생산) 테이블의 INumber(제품코드)와 tItem(제품) 테이블의 INumber(제품코드)를 JOIN한다.

④ 문제의 조건인 2021-01과 같은 값을 가져오기 위해 TO_CHAR함수를 이용하여 주문테이블인 tOrder에서 관련 컬럼인 ODate(주문일자)와 비교했고 전자기기류 제품 구분코드인 I4를 출력해주기 위해 SUBSTRING 함수를 이용하여 제품 테이블인 tItem에서 관련 컬럼인 INumber(제품코드)와 비교한다.

⑤ 그룹화 한 IName(제품명)을 기준으로 ①에서 SUM함수를 사용하여 총 판매량을 구한다

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다.

TO_CHAR(tem. ODate,'YYYY-MM')

- 첫번째 파라미터인 StartDate는 Timestamp타입이고, 두번째 파라미터인 'YYYY-MM' 포맷에 맞춰 '년도-월'의 텍스트타입으로 리턴한다.

SUBSTRING(tit.INumber,1,2)

- 첫번째 파라미터는 문자열이고 두번째 파라미터는 시작 인덱스입니다. 세번째 파라미터는 가져 올 개수입니다. INumber 에서 1 번째부터 2 개까지 문자를 가져옵니다.

18. (INNER JOIN)

2020년 1월에 만들어진 모든 제품의 생산코드, 생산직원명, 제품명, 생산량, 생산일자를 출력하세요
(단, 정렬은 고려하지 않는다)

<정답>

<결과 화면>

	생산코드 character varying(5)	생산직원명 character varying(5)	제품명 character varying(20)	생산량 integer	생산일자 timestamp with time zone
1	P1001	박하나	가위	620	2020-01-03 11:32:22+09
2	P2001	김구성	쇼파	52	2020-01-02 09:23:17+09
3	P2002	박가구	테이블	73	2020-01-07 10:34:53+09
4	P2003	김혜영	장농	22	2020-01-13 13:08:33+09
5	P2004	이은혜	침대	457	2020-01-22 09:07:32+09

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tpr.PNumber AS 생산코드, tem.EName AS 생산직원명, tit.IName AS 제품명,        --------- ①
       tpr.PCount AS 생산량, tpr.PDate AS 생산일자
FROM tProduction AS tpr
JOIN tItem AS tit                                                    --------- ②
ON tpr.INumber = tit.INumber
JOIN tEmployee AS tem                                                ---------- ③
ON tpr.ENumber = tem.ENumber
WHERE TO_CHAR(tpr.PDate,'YYYY-MM') = '2020-01' --------- ④
```

<해설>

① PNumber(생산코드)를 출력하기 위해 생산테이블인 tProduction에서 생산코드 컬럼인 PNumber을 가져오고
EName(직원명)을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져오고
IName(제품명)을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고
PCount(생산량)을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져오고
PDate(생산일자)를 출력하기 위해 생산테이블인 tProduction에서 생산일자 컬럼인 PDate를 가져온다.

② tItem(제품) 테이블에 있는 IName(제품명)을 가져오기 위해 tItem(제품) 테이블의
INumber(제품코드)와 tProduction(생산)테이블의 INumber(제품코드)를 JOIN한다.

③ tEmployee(직원) 테이블에 있는 EName(직원명)을 가져오기 위해 tEmployee(직원) 테이블의
ENumber(직원코드)와 tProduction(생산)테이블의 ENumber(직원코드)를 JOIN한다.

④ 문제의 조건인 2020-01과 같은값을 가져오기 위해 TO_CHAR 함수를 이용하여 생산테이블인 tProduction에서 관련 컬럼인 PDate와 비교한다

<함수설명>

TO_CHAR(tpr.PDate,'YYYY-MM-DD')

- 첫번째 파라미터인 PDate는 TIMESTAMP타입이고, 두번째 파라미터인 'YYYY-MM-DD' 포맷에 맞춰 '연-월-일'의 텍스트타입으로 리턴한다.

19. (INNER JOIN)

2022년 3월 20일 기준으로 현재 판매 가능한 공책의 재고량을 구하시오

(반품되어 돌아온 공책의 경우 재판매 하지 않는다.)

<정답>

<결과 화면>

	제품이름 character varying(20)	재고량 bigint
1	공책	834

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tBase.IName AS 제품명, (tBase2.PCount - tBase.OCount) AS 재고량---------- ①
FROM
(
    SELECT tit.IName, SUM(tpr.PCount) AS OCount ------ ②
    FROM tProduction AS tpr
    JOIN tOrder AS tor ---------- ③
    ON tpr.PNumber = tor.PNumber
    JOIN tItem AS tit ---------- ④
    ON tpr.INumber = tit.INumber
    WHERE tit.IName = '공책' AND tor.ODate < CAST('20220321' AS TIMESTAMP)------ ⑤
    GROUP BY tit.IName ------ ⑥
) AS tBase
JOIN ------ ⑦
(
    SELECT tit.IName, SUM(tpr.PCount) AS PCount------ ⑧
    FROM tProduction AS tpr
    JOIN tItem AS tit ---------- ⑨
    ON tpr.INumber = tit.INumber
    WHERE tit.IName = '공책' AND tpr.PDate < CAST('20220321' AS TIMESTAMP) ----- ⑩
    GROUP BY tit.IName ---------- ⑪
) AS tBase2
ON tBase.IName = tBase2.IName
```

<해설>

① 제품명을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 판매 가능한 재고량을 출력하기 위해 총 생산량을 담당하는 서브쿼리 tBase2(별칭)의 PCount(생산량)에서 총 판매량을 담당하는 서브쿼리 tBase(별칭)의 OCount를 빼준다.

② 제품명을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 총 판매량을 출력하기 위하여 생산테이블인 tProduction에서도 판매내역이 있는 PCount(생산량)를 가져온 후 합을 구하기 위해 SUM함수를 사용한다.

③ tOrder(주문) 테이블에 있는 ODate(판매일자)를 가져오기 위해 tProduction(생산) 테이블의 PNumber(생산코드)와 tOrder(주문) 테이블의 PNumber(생산코드)를 JOIN한다.

④, ⑨ tItem(제품) 테이블에 있는 IName(제품명)을 가져오기 위해 tProduction(생산) 테이블의 INumber(제품코드)와 tItem(제품) 테이블의 INumber(제품코드)를 JOIN한다.

⑤, ⑩ 문제의 조건인 공책을 가져오기 위해 제품테이블인 tItem에서 관련 컬럼인 IName과 비교한 후 문제의 조건인 2022-03-20 이전이기에 CAST를 사용하여 날짜가 이보다 작은 값을 가져온다.

⑥, ⑪ 그룹화 한 IName(제품명)을 기준으로 [SUM]함수를 사용하여 총 판매량(⑥)과 총 생산량(⑩)을 구한다

⑦ tBase(별칭) 서브쿼리에 있는 IName(제품명)과 OCount(총 판매량)와 tBase2(별칭) 서브쿼리에 있는 IName(제품명)과 PCount(총 생산량)를 가져오기 위해 tBase(별칭) 서브쿼리의 IName(제품명)과 tBase2(별칭) 서브쿼리의 IName(제품명)을 JOIN한다.

⑧ 제품명을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 총 생산량을 출력하기 위하여 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온 후 합을 구하기 위해 SUM함수를 사용한다.

현재 재고량이 총 생산량 – 총 판매량이기에 총 생산량과 총 판매량을 가진 값들이 필요하여 생산량 데이터를 가지고 있는 테이블인 tProduction의 PCount와 판매 이력을 가지고 있는 테이블인 tOrder와 생산량을 가지고 있는 테이블인 tProduction을 JOIN하여 판매량 데이터를 가지게 된 컬럼인 PCount를 SUM을 이용하여 각각 총 생산량과 총 판매량으로 합산 후 두 테이블을 JOIN한 후 총 생산량에서 총 판매량 값을 빼주어 재고량을 구해주었다.

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

CAST('20220321' AS TIMESTAMP)

- 텍스트타입인 '20220321'을 TIMESTAMP타입으로 형 변환한다.

20. (OUTER JOIN)

2020년 1월의 제품 별 생산량의 순위를 확인하기 위하여 제품명과 생산량을 순위를 매겨 출력하시오.
(모든 제품이 출력되어야 하며 공동순위가 있다면 다음 순위는 공동순위의 수 만큼 밀려나고 생산되지 않은 제품은 제일 마지막 순위로 결정되어야 한다)

<정답>

<결과 화면>

	제품명 character varying(20)	생산량 bigint	생산량_순위 bigint
1	주스	2323	1
2	커피	1215	2
3	귀걸이	923	3
4	전통차	872	4
5	목걸이	632	5

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tit.IName AS 제품명, tBase.PCount AS 생산량,                                    ---------- ①
       RANK() OVER(ORDER BY tBase.PCount DESC NULLS LAST) AS 생산량_순위
FROM tItem AS tit
LEFT OUTER JOIN ------------------------------ ②
(
    SELECT tpr.INumber, SUM(tpr.PCount) AS PCount ---------- ③
    FROM tProduction AS tpr
    WHERE TO_CHAR(tpr.PDate,'YYYYMM') = '202001' ---------- ④
    GROUP BY tpr.INumber ---------- ⑤
) AS tBase
ON tit.INumber = tBase.INumber
```

<해설>

① tItem(제품)테이블에서 출력된 IName(제품명)을 가져오고 서브쿼리 tBase(별칭)에서 출력된 생산량을 가져온다.
문제의 조건에 맞는 결과인 제품 별 생산량의 순위를 출력하되 동순위가 두 명이상 일수 있기에 RANK 함수를 사용하였고 ORDER BY에 존재하는 tBase(별칭).PCount (제품 별 생산량)을 기준으로 하여 생산량이 많은 순으로 순위를 출력해야 하기에 DESC(내림차 순)을 써주었으며 NULL 값은 제일 마지막에 출력하기 위하여 NULLS LAST를 작성한다.

② tItem(제품)테이블에 있는 INumber(제품코드)에 대응하는 서브쿼리 tBase(별칭)의 생산량을 가져오기 위해 tItem(제품) 테이블의 INumber(제품코드)와 서브쿼리 tBase(별칭)의 INumber(제품코드)를 JOIN하되 생산되지 않은 제품정보까지 출력해주기 위하여 LEFT OUTER JOIN을 사용한다.

③ INumber(제품코드)와 PCount(생산량)을 출력하기 위해 생산테이블인 tProduction에서 제품코드 컬럼인 INumber와 생산량 컬럼인 PCount를 가져온다.
그리고 생산량의 합을 구하기 위해 PCount에 SUM함수를 사용한다.

④ 문제의 조건인 '202001'과 같은값을 가져오기 위해 TO_CHAR함수를 이용하여 생산테이블인 tProduction에서 관련 컬럼인 PDate와 비교한다

⑤ INumber(제품코드)를 GROUP BY를 사용하여 그룹화 한 후 INumber(제품코드)를 기준으로 ③에서 SUM함수를 사용하여 생산량을 구한다

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

TO_CHAR(tpr.PDate,'YYYY-MM-DD')

- 첫번째 파라미터인 PDate는 Timestamp타입이고, 두번째 파라미터인 'YYYY-MM-DD' 포맷에 맞춰 '연-월-일'의 텍스트타입으로 리턴한다.

21. (INNER JOIN)

우리 회사의 고객인 '오랜문방구'의 반품제품명, 주문코드, 주문량, 반품량, 반품사유를 출력하세요

<정답>

<결과 화면>

	반품제품명 character varying(20)	주문코드 character varying(5)	주문량 integer	반품량 integer	반품사유 character varying(4)
1	공책	01007	700	700	배송오류

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tit.IName AS 반품제품명, tre.ONumber AS 주문코드,                    ----①
       tpr.PCount AS 주문량, tre.RCount AS 반품량, trr.RReason AS 반품사유
FROM tReturn AS tre
JOIN tOrder AS tor                                                          ------②
ON tre.ONumber = tor.ONumber
JOIN tCustomer AS tcu                                                       ------③
ON tor.CNumber = tcu.CNumber
JOIN tReturnReason AS trr                                                   ------④
ON tre.RRNumber = trr.RRNumber
JOIN tProduction AS tpr                                                     ------⑤
ON tor.PNumber = tpr.PNumber
JOIN tItem AS tit                                                           ------⑥
ON tpr.INumber = tit.INumber
WHERE tcu.CName = '오랜문방구'  ------⑦
```

<해설>

① 제품명을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 주문코드를 출력하기 위해 주문테이블인 tOrder 에서 주문코드 컬럼인 ONumber를 주문량을 출력하기 위해 생산테이블인 tProduction에서 생산 및 주문량 컬럼인 PCount, 반품량을 출력하기 위해 반품테이블인 tReturn에서 반품량 컬럼인 RCount와 마지막으로 반품사유를 출력하기 위해 반품사유테이블인 tReturnReason에서 반품사유 컬럼인 RReason을 가져온다.

② tOrder(주문) 테이블을 통해서 tReturn(반품) 테이블의 RCount(반품량) 데이터를 가져오기 위하여 tOrder(주문) 테이블의 PNumber(생산코드)와 tProduction(생산) 테이블의 PNumber(생산코드)를 JOIN한다

③ tCustomer(고객) 테이블에 있는 CName(고객명) 데이터를 가져오기 위해 tCustomer(고객) 테이블의 CNumber(고객코드)와 tOrder(주문)테이블의 CNumber(고객코드)를 JOIN한다

④ tReturnReason(반품사유) 테이블에 있는 RReason (반품사유) 데이터를 가져오기 위해 tReturnReason(반품사유) 테이블의 RRNumber(반품사유코드)와 tReturn (주문)테이블의 RRNumber(반품사유코드)를 JOIN한다

⑤ tProduction(생산) 테이블에 있는 PCount(생산량) 데이터를 가져오기 위해 tProduction(생산) 테이블의 PNumber(생산코드)와 tOrder(주문)테이블의 PNumber(생산코드)를 JOIN한다

⑥ tItem(제품) 테이블에 있는 IName(제품명) 데이터를 가져오기 위해 tItem(제품) 테이블의 INumber(제품코드)와 tProduction(생산) 테이블의 INumber(제품코드)를 JOIN한다

⑦ 고객테이블인 tCustomer에서 관련 컬럼인 CName과 비교한다.

22. (INNER JOIN)

퇴사자들이 생산한 제품들 중 반품된 제품들의 정보에 대해 알아보려 한다.

해당 조건에 맞는 제품을 생산한 직원명과 제품명, 해당 제품의 생산량, 반품량, 반품이유를 출력하시오.

<정답>

<결과 화면>

	직원명 character varying(5)	제품명 character varying(20)	생산량 integer	반품량 integer	반품이유 character varying(4)
1	박지원	돌	754	210	단순변심
2	박수혜	반지	62	62	환불
3	이승진	컴퓨터	288	20	불량

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tem.EName AS 직원명, tit.IName AS 제품명, tpr.PCount AS 생산량,        ---------- ①
       tre.RCount AS 반품량, trr.RReason AS 반품사유명
FROM tEmployee AS tem
JOIN tProduction AS tpr                  ---------- ②
ON tem.ENumber = tpr.ENumber
JOIN tOrder AS tor                       ---------- ③
ON tpr.PNumber = tor.PNumber
JOIN tReturn AS tre                      ---------- ④
ON tor.ONumber = tre.ONumber
JOIN tReturnReason AS trr                ---------- ⑤
ON tre.RRNumber = trr.RRNumber
JOIN tItem AS tit                        ---------- ⑥
ON tpr.INumber = tit.INumber
WHERE tem.Resignationdate IS NOT NULL    ------ ⑦
```

<해설>

① EName(직원명)을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져오고 IName(제품명)을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 PCount(생산량)을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount을 가져오고 RCount(반품량)을 출력하기 위해 반품테이블인 tReturn에서 반품량 컬럼인 RCount을 가져오고 RReason(반품사유명)을 출력하기 위해 반품사유테이블인 tReturnReason에서 반품사유명 컬럼인 RReason을 가져온다.

② tEmployee(직원) 테이블에 있는 EName(직원명)을 가져오기 위해 tEmployee(직원) 테이블의 ENumber(직원코드)와 tProduction(생산)테이블의 Eumber(직원코드)를 JOIN한다.

③ tOrder(주문) 테이블에 있는 ONumber(주문코드)를 가져오기 위해 tProduction(생산) 테이블의 PNumber(생산코드)와 tOrder(주문) 테이블의 PNumber(생산코드)를 JOIN한다.

④ tReturn(반품) 테이블에 있는 RCount(반품량)를 가져오기 위해 tOrder(주문) 테이블의 ONumber(주문코드)와 tReturn(반품) 테이블의 ONumber(주문코드)를 JOIN한다.

⑤ tReturnReason(반품사유) 테이블에 있는 RReason(반품사유)를 가져오기 위해 tReturn(반품) 테이블의 RRNumber(반품사유코드)와 tReturnReason(반품사유) 테이블의 RRNumber(반품사유코드)를 JOIN한다.

⑥ tItem(제품) 테이블에 있는 IName(제품명)을 가져오기 위해 tProduction(생산) 테이블의 INumber(제품코드)와 tItem(제품) 테이블의 INumber(제품코드)를 JOIN한다.

⑦ 문제의 조건인 퇴사자를 구하기 위해 직원테이블인 tEmployee에서 관련 컬럼인 Resignationdate(퇴사일자)가 NULL이 아닌 것을 찾는다

23. (INNER JOIN)

부서별로 생산하는 제품들의 총 생산량을 부서명과 함께 순위를 매겨서 출력하세요

(부서명, 부서에서 생산하는 제품의 총 생산량, 순위가 나와야 하며 공동순위(ex 공동 1등)가 있어도 다음 순위는 순차적으로 매겨진다.)

<정답>

<결과 화면>

	부서명 character varying(10)	총_생산량 bigint	총_생산량_순위 bigint
1	음료생산부	53184	1
2	문구생산부	11227	2
3	악세사리생산부	8073	3
4	전자기기생산부	6285	4
5	가구생산부	2845	5

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tde.DName AS 부서명                                    -----①
    , SUM(tpr.PCount) AS 총_생산량
    , ROW_NUMBER() OVER (ORDER BY SUM(tpr.PCount) DESC) AS 총_생산량_순위-------②
FROM tProduction AS tpr
JOIN tEmployee AS tem                                         -----③
ON tpr.ENumber = tem.ENumber
JOIN tDepartment AS tde                                       -----④
ON tem.DNumber = tde.DNumber
GROUP BY tde.DName ----------------------⑤
```

<해설>

① 부서명을 출력하기 위해 부서테이블인 tDepartment에서 부서명 컬럼인 DName을 가져오고 부서별 총 생산량을 출력하기 위해 생산테이블인 tProduction에서 PCount(생산량)를 가져온 다음 합을 구하기 위해 SUM함수를 사용한다.

② 문제의 조건에 맞는 결과인 제품들의 총 생산량을 출력하되 순위가 겹치면 안 되기 때문에 ROW_NUMBER를 사용하여 순위를 매겼고 ORDER BY에 존재하는 SUM(PCount) (총 판매량)을 기준으로 하여 순위를 출력해주되 생산량이 많은 순으로 출력하기 위해 DESC(내림차 순)으로 출력해준다.

③ tEmployee(직원) 테이블에 있는 ENumber(직원코드)를 가져오기 위해 tEmployee(직원) 테이블의 ENumber (직원코드)와 tProduction (생산)테이블의 ENumber (직원코드)를 JOIN한다.

④ tDepartment(부서) 테이블에 있는 DName(부서명)을 가져오기 위해 tEmployee(직원) 테이블의 DNumber (부서코드)와 tDepartment(부서) 테이블의 DNumber (부서코드)를 JOIN한다.

⑤ 그룹화 한 DName(부서명)을 기준으로 ①에서 SUM함수를 사용하여 총 생산량을 구한다

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

24. (OUTER JOIN)

시장조사를 위하여 불량을 제외한 반품내역을 가진 고객들의 주변에 거주하는 직원들의 명단을 출력하시오
(고객들의 주소와 고객명단, 고객 주변에 거주하는 직원명단은 전부 출력되어야 한다)

<정답>

<결과 화면>

	고객주소 character varying(10)	고객명 character varying(20)	직원명 character varying(5)
1	울산시 중구	오랜문방구	박천이
2	서울시 강남구	대한문방구	김이수
3	서울시 강남구	대한문방구	박하나
4	서울시 강남구	대한문방구	이빛나
5	서울시 강남구	대한문방구	김화사

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tBase.CAddr AS 고객주소, tBase.CName AS 고객명, tem.EName AS 직원명 ---------- ①
FROM(
    SELECT *                                    ---------- ②
    FROM tCustomer AS tcu
    JOIN tOrder AS tor                          ---------- ③
    ON tcu.CNumber = tor.CNumber
    JOIN tReturn AS tre                         ---------- ④
    ON tor.ONumber = tre.ONumber
    JOIN tReturnReason AS trr                   ---------- ⑤
    ON tre.RRNumber = trr.RRNumber
    WHERE trr.RReason <> '불량'                 ---------- ⑥
) AS tBase
LEFT OUTER JOIN tEmployee AS tem                ---------- ⑦
ON tBase.CAddr = tem.EAddr
```

<해설>

① 문제에서 제시한 CAddr(고객주소), CName(고객명), EName(직원명)을 서브쿼리 tBase(별칭)에서 가져온다.

② 모든 데이터를 가져오기 위해 *(모든 컬럼 출력) 를 입력한다.

③ tOrder(주문) 테이블에 있는 데이터를 가져오기 위해 tCustomer(고객) 테이블의 CNumber(고객코드)와 tOrder(주문) 테이블의 CNumber(고객코드)를 JOIN한다.

④ tReturn(반품) 테이블에 있는 데이터를 가져오기 위해 tOrder(주문) 테이블의 ONumber(주문코드)와 tOrder(주문) 테이블의 ONumber(주문코드)를 JOIN한다.

⑤ tReturnReason(반품사유) 테이블에 있는 데이터를 가져오기 위해 tReturn(반품) 테이블의 RRNumber(반품사유코드)와 tReturnReason(반품사유) 테이블의 RRNumber(반품사유코드)를 JOIN한다.

⑥ 문제의 조건인 불량이 아닌값을 가져오기 위해 반품사유테이블인 tReturnReason에서 관련컬럼인 RReason(반품사유)과 비교한다.

⑦ 서브쿼리 tBase(별칭)에서 조건에 해당하지 않는 정보까지 출력해주기 위하여 LEFT OUTER JOIN을 사용한다

25. (OUTER JOIN)

고객별 반품 현황을 파악하기 위하여 고객별로 고객명과 제품을 구매한 양과 반품한 양 그리고 이를 구매량 대비 반품량을 반품률로 나타내시오.

(반품률은 높은 순으로 소숫점 2자리까지 반올림되어 출력되어야하며 반품내역이 없는 값(null)은 0으로 대체 되면서 마지막에 출력되어야 한다.)

<정답>

<결과 화면>

	고객이름 character varying(20)	구매량 bigint	반품량 bigint	반품률 numeric
1	오랜문방구	1254	700	55.82
2	강남악세사리	494	272	55.06
3	대한문방구	1374	754	54.88
4	동해가구	124	65	52.42
5	패션악세사리	1559	650	41.69

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tcu.CName AS 고객명 ]---------- ①
    , SUM(tpr.PCount) AS 구매량 ]
    , COALESCE(SUM(tre.RCount), 0) AS 반품량------ ②
    , COALESCE(ROUND(CAST(CAST(SUM(tre.RCount) AS FLOAT)/ ]------ ③
      CAST(SUM(tpr.PCount) AS FLOAT) * 100 AS DECIMAL),2), 0) AS 반품률
FROM tOrder AS tor
JOIN tCustomer AS tcu ]---------- ④
ON tcu.CNumber = tor.CNumber ]
JOIN tProduction AS tpr ]---------- ⑤
ON tpr.PNumber = tor.PNumber ]
LEFT OUTER JOIN tReturn AS tre ]---------- ⑥
ON tre.ONumber = tor.ONumber ]
GROUP BY tcu.CName ---------- ⑦
ORDER BY 반품률 DESC NULLS LAST---------- ⑧
```

<해설>

① 고객명을 출력하기 위해 고객테이블인 tCustomer에서 고객명 컬럼인 CName을 가져오고 구매량을 출력하기 위해 주문테이블인 tOrder와 겹쳐지는 생산이력을 가진 값을 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온 다음 합을 구하기 위해 SUM함수를 사용한다.

② 반품량을 출력하기 위해 반품테이블인 tReturn에서 반품량 컬럼인 RCount를 가져오는데 이 때 NULL값이 있을 수도 있기에 NULL일 때는 0으로 바꿔서 출력하기 위해 COALESCE 함수를 사용한다.

③ 반품률을 출력하기 위해 ①에서 구했던 구매량과 ②에서 구했던 반품량을 나누어주는데 이 때 나누는 값을 FLOAT 형식으로 바꿔주기 위하여 CAST를 사용하고, 소수점 두 자리에서 반올림 해줘야 하기에 ROUND 함수를 사용하고 ②와 마찬가지로 NULL일 때는 0으로 바꿔서 출력하기 위해 COALESCE함수를 사용한다.

④ tCustomer(고객) 테이블에 있는 CName(고객명)을 가져오기 위해 tOrder(주문) 테이블의 CNumber(고객코드)와 tCustomer(고객)테이블의 CNumber(고객코드)를 JOIN한다.

⑤ tProduction(생산) 테이블에 있는 PCount(생산량)를 가져오기 위해 tOrder(주문) 테이블의 PNumber(생산코드)와 tProduction(생산) 테이블의 PNumber(생산코드)를 JOIN한다.

⑥ tReturn(반품) 테이블에 있는 RCount (반품량)를 가져오기 위해 tReturn(반품) 테이블의 ONumber(주문코드)와 tOrder(주문) 테이블의 ONumber(주문코드)를 JOIN하되 반품되지 않은 주문정보까지 출력해주기 위하여 LEFT OUTER JOIN을 사용한다

⑦ 그룹화 한 CName(고객명)을 기준으로 SUM함수를 사용하여 ①, ②에서 SUM(PCount)(구매량)과 반품량을 구하여 출력하게한다.

⑧ 정렬 기준인 반품률을 내림차순정렬을 하기 위해 ORDER BY 를 사용하고 NULL값이 있다면 마지막에 출력하기 위하여 NULLS LAST를 사용한다.

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

COALESCE(SUM(tre.RCount), 0)

- COALESCE 내부의 SUM(RCount) 값이 NULL 일 때 0으로 대체한다.

CAST(SUM(tre.RCount) AS FLOAT), CAST(SUM(tpr.PCount) AS FLOAT)

- NUMERIC 타입인 SUM(tre.RCount)와 SUM(tpr.PCount)의 데이터 타입을 FLOAT 형식으로 바꿔준다.

ROUND(CAST(CAST(SUM(tre.RCount) AS FLOAT) / CAST(SUM(tpr.PCount) AS FLOAT) * 100 AS DECIMAL),2)

- RCount의 모든 데이터를 합친 값과 PCount의 모든 데이터를 합친 값을 FLOAT타입으로 변환하고 100을 곱한 후 DECIMAL타입으로 변환하고 소수점 2자리까지 반올림한다

26. (INNER JOIN)

부서별로 직급이 제일 높은 직원을 확인하려 한다. 이에 해당하는 직원의 부서명과 직급명, 직원명을 출력하시오

<정답>

<결과 화면>

	부서명 character varying(10)	직급명 character varying(2)	직원명 character varying(5)
1	문구생산부	부장	박충괄
2	가구생산부	부장	박천이
3	악세사리생산부	부장	류이연
4	전자기기생산부	부장	정재훈
5	음료생산부	이사	박원선

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tBase.DName AS 부서명, tBase.RName AS 직급명, tBase.EName AS 직원명 ---------- ①
FROM
(
    SELECT tde.DName ┐ ---------- ②
        , tem.EName  │
        , tra.RName  ┘
        , RANK() OVER(PARTITION BY tem.DNumber ORDER BY tem.RNumber DESC) AS ranknum ------- ③
    FROM tEmployee AS tem
    JOIN tDepartment AS tde ┐ ---------- ④
    ON tem.DNumber = tde.DNumber ┘
    JOIN tRank AS tra ┐ ---------- ⑤
    ON tem.RNumber = tra.RNumber ┘
    WHERE tem.DNumber IS NOT NULL ---------- ⑥
) AS tBase
WHERE tBase.ranknum = 1 ---------- ⑦
```

<해설>

① 문제에서 제시한 DName(부서명), RName(직급명), EName(직원명)을 서브쿼리 tBase(별칭)에서 가져온다.

② DName(부서명)을 출력하기 위해 부서테이블인 tDepartment에서 부서명 컬럼인 DName을 가져오고 EName(직원명)을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져오고 RName(직급명)을 출력하기 위해 직급테이블인 tReturn에서 직급명 컬럼인 RName을 가져온다.

③ 직급코드를 통하여 문제의 조건에 맞는 순위를 출력하기 위해 RANK 함수를 사용하였고 PARTITION BY에 부서코드 컬럼인 DNumber를 사용함으로써 각 부서코드 마다 ORDER BY에 존재하는 RNumber(직급코드)를 정렬하여 순위를 매겨준다.
문제의 조건에 맞는 결과인 직급이 제일 높은 직원을 출력하되 동순위가 두 명이상 일수 있기에 RANK 함수를 사용하여 순위를 매겨 1위인 직원들을 뽑을 수 있는 조건을 만들었고 PARTITION BY에 부서코드 컬럼인 DNumber를 사용함으로써 부서 별로 출력해주었으며 ORDER BY에 존재하는 RNumber(직급코드)를 기준으로 하여 순위를 출력해주는데 높은 순으로 출력하기 위해 DESC(내림차 순)하여 순위를 출력해준다

④ tDepartment(부서) 테이블에 있는 DName(부서이름)을 가져오기 위해 tEmployee(직원) 테이블의 DNumber(부서코드)와 tDepartmet(부서) 테이블의 DNumber(부서코드)를 JOIN한다.

⑤ tRank(직급) 테이블에 있는 RName(직급이름)을 가져오기 위해 tEmployee(직원) 테이블의 RNumber(직급코드)와 tRank(직급) 테이블의 RNumber(직급코드)를 JOIN한다.

⑥ 문제의 조건인 부서별 출력을 위해선 부서가 없는 직원은 출력되면 안되므로 tEmployee(직원)테이블의 관련 컬럼인 DNumber가 NULL이 아닌 값을 가져온다.

⑦ 문제의 조건인 부서별 가장 높은 직원을 출력하기 위해 부서별 1등인 값만 가져온다.

27. (INNER JOIN)

입사일이 10년이 넘는 직원들에게는 해당 직원들의 평균판매량을 기준으로 추가수당을 주려한다.

2022년 2월 28일을 기준으로 해당 직원들의 평균 판매 금액을 출력하시오.

(반품 내역은 고려하지 않지만 퇴사자는 제외되어야 한다.)

<정답>

<결과 화면>

	평균판매금액 numeric
1	182318400.00000000

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT AVG(tBase.sell_SUM) AS 평균판매금액 ------------------ ①
FROM
(
    SELECT SUM(tpr.PCount*tit.Price) AS sell_SUM ---------- ②
    FROM tOrder AS tor
    JOIN tProduction AS tpr ------------------ ③
    ON tpr.PNumber = tor.PNumber
    JOIN tItem AS tit ------------------ ④
    ON tit.INumber = tpr.INumber
    JOIN tEmployee tem ------------------ ⑤
    ON tem.ENumber = tpr.ENumber
    WHERE tem.StartDate < CAST('2022-02-28' AS TIMESTAMP) – CAST('10 year' AS INTERVAL) -----⑥
    AND tem.ResignationDate IS NULL
    GROUP BY tem.EName ---------- ⑦
) AS tBase
```

<해설>

① tBase(별칭) 서브쿼리에서 결과물을 출력한 sell_SUM(총 판매금액) 컬럼을 가져와 평균을 나타내기 위하여 집계함수 AVG를 사용한다.

② 총 판매 금액을 출력하기 위해 판매량과 단가를 곱한 후 GROUP BY를 기준으로 SUM을 사용하여 합하였는데 이 때 사용한 판매량을 출력하기 위해 주문테이블인 tOrder 와 겹쳐지는 생산이력을 가진 값을 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져왔고 단가를 출력하기 위해 제품테이블인 tItem에서 단가 컬럼인 Price를 가져온다.

③ tProduction(생산) 테이블에 있는 PCount(생산량)을 가져오기 위해 tProduction(생산) 테이블의 PNumber(생산코드)와 tOrder(주문) 테이블의 PNumber(생산코드)를 JOIN한다.

④ tItem(제품) 테이블에 있는 Price(단가)를 가져오기 위해 tItem(제품) 테이블의 INumber(제품코드)와 tProduction(생산) 테이블의 INumber(제품코드)를 JOIN한다.

⑤ tEmployee(직원) 테이블에 있는 StartDate(입사일)를 가져오기 위해 tEmployee(직원) 테이블의 ENumber (직원코드)와 tProduction(생산) 테이블의 ENumber (직원코드)를 JOIN한다.

⑥ 문제의 조건인 2022-02-28을 기준으로 입사일이 10년 넘어간 조건을 가져오기 위해 CAST 함수를 사용하여 해당 날짜에서 10년을 뺀 값 보다 작은 값을 직원테이블인 tEmployee에서 관련 컬럼인 StartDate와 비교하였고 퇴사자는 제외되어야 하기 때문에 같은 테이블의 퇴사일자 컬럼인 ResignationDate을 비교하여 NULL인 값을 가져온다.

⑦ 그룹화 한 EName(직원명)을 기준으로 ②에서 SUM함수를 사용하여 총 판매량을 구하였다.

<함수설명>

AVG(tBase.sell_SUM)

- sell_SUM의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값의 평균을 계산 한 후 계산된 값을 리턴.

SUM(tpr.PCount*tit.Price)

- (tpr.PCount * tit.Price) 의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다.

CAST('2022-02-28' AS TIMESTAMP)

- 텍트스타입인 '2022-02-28'을 TIMESTAMP 타입으로 형 변환한다.

CAST('10 year' AS INTERVAL)

- 텍스트타입인 '10 year'을 INTERVAL타입으로 형 변환한다.

28. (OUTER JOIN)

직원들의 평균 불량률이 5%라고 가정하였을 때, 전체 직원명과 직원별 전체 생산량과 평균 불량률을 고려한 예상 불량품량과 실제로 나온 실제 불량품량을 직원명과 함께 모두 출력하시오.

<정답>

<결과 화면>

직원명 character varying(5)	생산량 bigint	예상불량품량 numeric	실제불량품량 integer
박수혜	62	3.10	
남궁수영	805	40.25	
이경희	872	43.60	
이승진	288	14.40	20
최양	1307	65.35	

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tBase.EName AS 직원명, tBase.PCount AS 생산량,                    ---------- ①
       tBase.avg_PCount AS 예상불량품량, tBase2.RCount AS 실제불량품량
FROM
(
    SELECT tpr.ENumber, tem.EName, SUM(tpr.PCount) AS PCount,    --------- ②
           SUM(tpr.PCount)*0.05 AS avg_PCount
    FROM tProduction AS tpr
    JOIN tEmployee AS tem                                        ---------- ③
    ON tpr.ENumber = tem.ENumber
    GROUP BY tem.EName, tpr.ENumber ---------- ④
) AS tBase
LEFT OUTER JOIN ---------- ⑤
(
    SELECT tpr.ENumber, tre.RCount                               ---------- ⑥
    FROM tProduction AS tpr
    JOIN tOrder AS tor                                           ---------- ⑦
    ON tor.PNumber = tpr.PNumber
    JOIN tReturn AS tre                                          ---------- ⑧
    ON tor.ONumber = tre.ONumber
    JOIN tReturnReason AS trr                                    ---------- ⑨
    ON tre.RRNumber = trr.RRNumber
    WHERE trr.RReason = '불량'  ---------- ⑩
) AS tBase2
ON tBase.ENumber = tBase2.ENumber
```

<해설>

① 문제에 제시된 EName(직원명), PCount(생산량), 예상불량품량, 실제불량품량을 출력하기 위해 서브쿼리 tBase(별칭)와 서브쿼리 tBase2(별칭)에서 값을 가져온다.

② ENumber(직원코드)를 출력하기 위해 생산테이블인 tProduction에서 직원코드컬럼인 ENumber를 가져오고 PCount(생산량)를 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온다.
합을 구하기 위해 SUM함수를 사용한다.
생산량의 5%를 구하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온다.
합을 구하기 위해 SUM함수를 사용하고 5%출력을 위해 0.05를 곱한다.

③ tEmployee(직원) 테이블에 있는 EName(직원명)을 가져오기 위해 tProduction(생산) 테이블의 ENumber(직원코드)와 tEmployee(직원)테이블의 ENumber(직원코드)를 JOIN한다.

④ EName(직원명), ENumber(직원코드)를 GROUP BY를 사용하여 그룹화 한 후 EName(직원명), ENumber(직원코드)를 기준으로 ②에서 SUM함수를 사용하여 생산량의 합과 예상불량품량을 구한다

⑤ 서브쿼리 tBase2(별칭)에 있는 eName(직원명)을 가져오기 위해 서브쿼리 tBase(별칭)에 있는 ENumber(직원코드) 와 서브쿼리 tBase2(별칭)에 있는 ENumber(직원코드)를 JOIN한다.

⑥ ENumber(직원코드)를 출력하기 위해 생산테이블인 tProduction에서 직원코드컬럼인 ENumber를 가져오고 RCount(반품량)를 출력하기 위해 반품테이블인 tReturn에서 반품량 컬럼인 RCount를 가져온다.

⑦ tOrder(주문) 테이블에 있는 ONumber(주문코드)를 가져오기 위해 tOrder(주문) 테이블의 PNumber(생산코드)와 tProduction(생산)테이블의 PNumber(생산코드)를 JOIN한다.

⑧ tReturn(반품) 테이블에 있는 RCount(반품량)를 가져오기 위해 tOrder(주문) 테이블의 ONumber(주문코드)와 tReturn(반품)테이블의 ONumber(주문코드)를 JOIN한다.

⑨ tReturnReason(반품사유) 테이블에 있는 RReason(반품사유명)을 가져오기 위해 tReturn(반품) 테이블의 RRNumber(반품사유코드)와 tReturnReason(반품사유) 테이블의 RRNumber(반품사유코드)를 JOIN한다

⑩ 문제의 조건인 불량과 같은 값을 가져오기 위해 반품사유테이블인 tReturnReason 테이블에서 관련 컬럼인 RReason(반품사유)과 비교한다.

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

29. (OUTER JOIN)

부서별로 제품의 판매량에 따라 추가혜택을 제공하려 한다. 각 부서별 부서명과 총 판매량을 출력하시오.

<정답>

<결과 화면>

	부서이름 character varying(10)	총_제품_판매량 bigint
1	문구생산부	5482
2	가구생산부	1278
3	악세사리생산부	3960
4	음료생산부	16583
5	전자기기생산부	3019

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tde.DName AS 부서명                    ----------①
    , SUM(tpr.PCount) AS 총_판매량
FROM tOrder AS tor
JOIN tProduction AS tpr                       ----------②
ON tor.PNumber = tpr.PNumber
JOIN tEmployee AS tem                         ----------③
ON tpr.ENumber = tem.ENumber
JOIN tDepartment AS tde                       ----------④
ON tem.DNumber = tde.DNumber
LEFT OUTER JOIN tReturn AS tre                ----------⑤
ON tor.ONumber = tre.ONumber
WHERE tre.ONumber IS NULL------⑥
GROUP BY tde.DName -------------⑦
```

<해설>

① 부서명을 출력하기 위해 부서테이블인 tDepartment에서 부서명 컬럼인 DName을 가져오고 총 판매량을 출력하기 위해 제품테이블인 tProduction에 해당하는 생산이력을 가진 PCount를 가져온 다음 합을 구하기 위해 SUM함수를 사용한다.

② tProduction(생산) 테이블에 있는 PCount(생산량)을 가져오기 위해 tProduction(생산) 테이블의 PNumber(생산코드)와 tOrder(주문) 테이블의 PNumber(생산코드)를 JOIN한다.

③ tEmployee(직원) 테이블에 있는 ENumber(직원코드)를 가져오기 위해 tEmployee(직원) 테이블의 ENumber(직원코드)와 tProduction(생산) 테이블의 ENumber(직원코드)를 JOIN한다.

④ tDepartment(부서) 테이블에 있는 DName(부서명)을 가져오기 위해 tDepartment(부서) 테이블의 DNumber(부서코드)와 tEmployee(직원) 테이블의 DNumber(부서코드)를 JOIN한다.

⑤ tReturn(반품) 테이블에 있는 RNumber(반품코드)를 가져오기 위해 tReturn(반품) 테이블의 ONumber(주문코드)와 tOrder(주문) 테이블의 ONumber(주문코드)를 JOIN하고 반품되지 않은 주문정보까지 출력해주기 위하여 LEFT OUTER JOIN을 사용한다

⑥ 반품된 판매내역은 출력하지 않기 위해 반품테이블인 tReturn에서 관련 컬럼인 ONumber(주문코드)와 비교한다

⑦ 그룹화 한 DName(부서명)을 기준으로 ①에서 SUM함수를 사용하여 총 판매량을 구한다

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다.

30. (OUTER JOIN)

현재(2022년 2월) 회사에 남아있는 제품들의 재고량을 파악하려 한다.

아직 팔리지 않거나 불량을 제외한 사유로 반품된 재고품들의 제품명과 재고량을 출력하시오.

(결과는 제품명을 오름차순으로 정렬하시오.)

<정답>

<결과 화면>

	제품명 character varying(20)	재고량 bigint
1	가위	782
2	공책	1534
3	귀걸이	580
4	노트북	432
5	디지털카메라	280

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tit.IName AS 제품명, SUM(tpr.PCount) AS 재고량 ---------- ①
FROM tProduction AS tpr
LEFT OUTER JOIN tOrder AS tor                                    --------- ②
ON tor.PNumber = tpr.PNumber AND TO_CHAR(ODate,'YYYYMM') < '202202'
LEFT OUTER JOIN tReturn AS tre ---------- ③
ON tre.ONumber= tor.ONumber
LEFT OUTER JOIN tReturnReason AS trr                             -----④
ON tre.RRNumber = trr.RRNumber AND trr.RReason <> '불량'
JOIN tItem AS tit ------------- ⑤
ON tpr.INumber = tit.INumber
WHERE tor.ONumber IS NULL OR trr.RRNumber IS NOT NULL -------- ⑥
GROUP BY tit.IName ------------------ ⑦
ORDER BY tit.IName ------------------ ⑧
```

<해설>

① IName(제품명)을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 PCount(생산량)을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온다. 합을 구하기 위해 SUM함수를 사용한다.

② tOrder(주문) 테이블에 있는 ONumber(주문코드)를 가져오기 위해 tProduction(생산) 테이블의 PNumber(생산코드)와 tOrder(주문) 테이블의 PNumber(생산코드)를 JOIN한다.
문제에 제시된 2022년 2월이전에 주문된 수량에 대해서 제외하기 위해 주문테이블인 tOrder에서 관련컬럼인 ODate를 비교한다.

③ tReturn(반품) 테이블에 있는 RNumber(반품코드)를 가져오기 위해 tOrder(주문) 테이블의 ONumber(주문코드)와 tOrder(주문) 테이블의 ONumber(주문코드)를 JOIN한다.

④ tReturnReason(반품사유) 테이블에 있는 RReason(반품사유명)을 가져오기 위해 tReturn(반품) 테이블의 RRNumber(반품사유코드)와 tReturnReason(반품사유) 테이블의 RRNumber(반품사유코드)를 JOIN한다 문제에 제시된 불량을 제외한 사유와 같은 값을 가져오기 위해 반품사유테이블인 tReturnReason에서 관련 컬럼인 RReason(반품사유)과 비교한다.

⑤ tItem(제품) 테이블에 있는 IName(제품명)을 가져오기 위해 tProduction(생산) 테이블의 INumber(제품코드)와 tItem(제품) 테이블의 INumber(제품코드)를 JOIN한다.

⑥ 문제에 제시된 아직 팔리지 않거나 반품사유가 존재하는 값을 가져오기 위해 주문테이블인 tOrder에서 관련 컬럼인 ONumber(주문코드)를 비교하고, 반품사유 테이블인 tReturnReason에서 관련 컬럼인 RRNumer(반품사유코드)를 비교한다.

⑦ GROUP BY를 사용하여 그룹화 한 IName(제품명)을 기준으로 ①에서 SUM함수를 사용하여 재고량을 구한다

⑧ 정렬 기준인 IName(제품명)을 오름차 순 정렬을 하기 위해 ORDER BY 를 사용한다

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

TO_CHAR(ODate,'YYYYMM')

- 첫번째 파라미터인 ODate는 Timestamp타입이고, 두번째 파라미터인 'YYYYMM' 포맷에 맞춰 '년도월'의 텍스트타입으로 리턴한다.

31. (INNER JOIN)

각 제품별 반품현황을 파악하기 위하여 반품된 제품들 제품명과 반품사유명, 반품날짜를 출력하시오
(반품사유가 여러 개면 제일 먼저 반품된 사유 순으로 나열하여야 한다.)

<정답>

<결과 화면>

	제품명 character varying(20)	반품사유 character varying(4)	반품날짜 timestamp with time zone
1	공책	배송오류	2022-01-03 15:17:47+09
2	노트북	불량	2022-01-08 16:09:20+09
3	디지털카메라	불량	2022-01-13 13:25:53+09
4	모니터	환불	2022-01-02 09:27:11+09
5	목걸이	단순변심	2022-01-20 09:45:33+09

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tit.IName AS 제품명, trr.RReason AS 반품사유, tre.RDate AS 반품날짜 ----- ①
FROM tReturn AS tre
JOIN tReturnReason AS trr ---- ②
ON tre.RRNumber = trr.RRNumber
JOIN tOrder AS tor ---- ③
ON tre.ONumber = tor.ONumber
JOIN tProduction AS tpr ---- ④
ON tor.PNumber = tpr.PNumber
JOIN tItem AS tit ---- ⑤
ON tpr.INumber = tit.INumber
ORDER BY tit.IName, tre.RDate---------- ⑥
```

<해설>

① 제품명을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 반품사유를 출력하기 위해 반품사유테이블인 tReturnReason에서 반품사유명 컬럼인 RReason을 가져오고 반품날짜를 출력하기 위해 반품테이블인 tReturn에서 반품날짜컬럼인 RDate를 가져왔다.

② tReturnReason(반품사유) 테이블에 있는 RReason(반품사유명)을 가져오기 위해 tReturnReason(반품사유) 테이블의 RRNumber(반품사유코드)와 tReturn(반품) 테이블의 RRNumber0(반품사유코드)를 JOIN한다.

③ tOrder(주문) 테이블에 있는 ONumber(주문코드)를 가져오기 위해 tOrder(주문) 테이블의 ONumber(주문코드)와 tReturn(반품) 테이블의 ONumber(주문코드)를 JOIN한다.

④ tProduction(생산) 테이블에 있는 PNumber(생산코드)를 가져오기 위해 tProduction0(생산) 테이블의 PNumber(생산코드)와 tOrder(주문) 테이블의 PNumber(생산코드)를 JOIN한다.

⑤ tItem(제품) 테이블에 있는 IName (제품명)을 가져오기 위해 tItem(제품) 테이블의 INumber(제품코드)와 tProduction(생산) 테이블의 INumber(제품코드)를 JOIN한다.

⑥ 정렬 기준인 IName(제품명)을 오름차 순으로 정렬한 후 여러 반품사유가 존재할 시 RDate(반품날짜)를 기준으로 제일 먼저 반품 된 내역이 나와야 하기에 오름차 순으로 정렬을 하는 ORDER BY 를 사용한다

32. (OUTER JOIN)

각 제품별 판매량 대비 반품의 현황을 파악하려한다.

이를 위하여 제품별로 제품명과 판매량, 반품량 그리고 반품률을 나타내어 출력하시오.

(반품률은 (제품 총 반품량/제품 총 판매량)으로 그리고 소숫점 2자리까지 반올림하여야 하고 반품내역이 없는 값(null)은 0으로 대체 되어 출력되어야한다.)

<정답>

<결과 화면>

	제품명 character varying(20)	판매량 bigint	반품량 bigint	반품률 numeric
1	가위	1740	0	0
2	공책	1394	700	50.22
3	귀걸이	1145	0	0
4	노트북	1191	11	0.92
5	디지털카메라	93	9	9.68

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tit.IName AS 제품명                                          ---------- ①
    , SUM(tpr.PCount) AS 판매량
    , COALESCE(SUM(tre.RCount), 0) AS 반품량
    , COALESCE(ROUND(CAST(CAST(SUM(tre.RCount) AS FLOAT) /
      CAST(SUM(tpr.PCount) AS FLOAT) * 100 AS DECIMAL),2), 0) AS 반품률
FROM tOrder AS tor
JOIN tProduction AS tpr                                             -------- ②
ON tpr.PNumber = tor.PNumber
LEFT OUTER JOIN tReturn AS tre                                      --------- ③
ON tre.ONumber = tor.ONumber
JOIN tItem AS tit                                                   --------- ④
ON tit.INumber = tpr.INumber
GROUP BY tit.IName      ---------------- ⑤
ORDER BY tit.IName      ---------------- ⑥
```

<해설>

① IName(제품명)을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 PCount(생산량)을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져오고 합을 구하기 위해 SUM함수를 사용한다.
RCount(반품량)을 출력하기 위해 반품테이블인 tReturn에서 반품량 컬럼인 RCount를 가져오고 합을 구하기 위해 SUM함수를 사용하고 NULL값을 0으로 치환하기 위해 COALESCE함수를 사용한다
반품률을 구하기 위해 반품량/생산량 을 계산한다. 소수점으로 나누기 위하여 CAST함수를 사용해 데이터 타입을 변경하였고 NULL값을 0으로 치환하기 위해 COALESCE함수를 사용하였다.

② tProduction(생산)테이블에 있는 PCOUNT(생산량)을 가져오기 위해 tOder(주문) 테이블의 ONumber(주문코드)와 tProdunction(생산) 테이블의 ONumber(주문코드)를 JOIN한다.

③ tReturn(반품) 테이블에 있는 RNumber(반품코드)를 가져오기 위해 tOrder(주문) 테이블의 ONumber(주문코드)와 tOrder(주문) 테이블의 ONumber(주문코드)를 JOIN한다.
반품하지 않은 주문정보까지 출력해주기 위하여 OUTER JOIN을 사용한다.

④ tItem(제품) 테이블에 있는 IName(제품명)을 가져오기 위해 tProduction(생산) 테이블의 INumber(제품코드)와 tItem(제품) 테이블의 INumber(제품코드)를 JOIN한다.

⑤ IName(제품명)을 GROUP BY를 사용하여 그룹화 한다.
그룹화 한 IName(제품명)을 기준으로 ①에서 SUM함수를 사용하여 판매량, 반품량, 반품률을 구한다

⑥ 정렬 기준인 IName(제품명)을 오름차 순 정렬을 하기 위해 ORDER BY 를 사용한다

<함수설명>

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

COALESCE(SUM(tre.RCount), 0)

- SUM(tre.RCount) 값이 NULL일 때 0으로 대체한다

CAST(SUM(tpr.PCount) AS FLOAT), CAST(SUM(tre.RCount) AS FLOAT)

- NUMERIC 타입인 SUM(tpr.PCount)와 SUM(tre.RCount)를 FLOAT 타입으로 형변환 한다.

ROUND(CAST(CAST(SUM(tre.RCount) AS FLOAT) / CAST(SUM(tpr.PCount) AS FLOAT) * 100 AS DECIMAL),2)

- RCount의 모든 데이터를 합친 값과 PCount의 모든 데이터를 합친 값을 FLOAT타입으로 변환하고 100을 곱한 후 DECIMAL타입으로 변환하고 소수점 2자리까지 반올림한다

33. (INNER JOIN)

2021년 01월부터 2021년 03월까지 지역별 판매순위를 확인하기 위해서 지역명과 지역별 주문 횟수를 출력하되 주문 횟수가 많은 순서대로 순위를 매겨 출력하시오

(지역은 고객 테이블(tCustomer)의 CAddr(고객주소)로 구분하며, 만약 공동순위(ex. 공동 1등)가 있다면 다음 순위는 중복 된 순위의 수 만큼 떨어진다.)

<정답>

<결과 화면>

	고객주소 character varying(10)	지역별_주문횟수 bigint	주문순위 bigint
1	서울시 강남구	10	1
2	서울시 송파구	6	2
3	부산시 남구	6	2
4	부산시 북구	5	4
5	서울시 동작구	4	5

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tcu.CAddr AS 고객주소                                    ----- ①
    , COUNT(tcu.CAddr) AS 지역별_주문횟수
    , RANK() OVER(ORDER BY COUNT(tcu.CAddr) DESC) AS 주문순위 ----- ②
FROM tOrder AS tor                                              ----------- ③
JOIN tCustomer AS tcu
ON tcu.CNumber = tor.CNumber
WHERE tor.ODate BETWEEN CAST('20210101' AS TIMESTAMP) AND CAST('20210401' AS TIMESTAMP) ----- ④
GROUP BY tcu.CAddr ----- ⑤
```

<해설>

① 고객주소를 출력하기 위해 고객테이블인 tCustomer에서 고객주소 컬럼인 CAddr을 가져오고 지역별로 주문횟수를 출력하기 위해 CAddr(고객주소)을 기준으로 삼아 COUNT 함수를 사용한다.

② 문제의 조건에 맞는 결과인 지역별 판매순위를 출력하되 동순위가 두 명 이상 일수 있기에 RANK 함수를 사용하여 순위를 매겼고 ORDER BY에 존재하는 COUNT(tcu.CAddr) (고객주소 별 주문횟수)를 기준으로 하여 순위를 출력해주는데 횟수가 많은 순으로 출력하기 위해 DESC(내림차 순)하여 순위를 출력해준다

③ tOrder(주문) 테이블과 tReturnReason(반품사유) 테이블을 CNumber(제품코드)를 이용하여 조인하였다.

④ 문제의 조건인 2021년 1월부터 2021년 3월의 값을 가져오기 위해 BETWEEN과 CAST함수를 사용하여 날짜가 20210101보다 크고 20210401보다 작은 값을 주문테이블인 tOrder의 관련 컬럼인 ODate(주문일자)와 비교하였다.

⑤ 그룹화 한 CAddr(고객주소)을 기준으로 COUNT 함수를 사용하여 지역별 주문횟수를 구한다

문제에서 필요로 하는 컬럼은 고객주소와 고객의 주소별 즉 지역별 주문 횟수 및 순위이다.

이를 구하기 위한 테이블은 고객 주소가 있는 테이블인 tCustomer와 주문내역이 있는 테이블인 tOrder 인데, 이 두 테이블을 JOIN 한 후 컬럼들을 출력해주면 되는데 지역별 주문횟수를 출력하기 위해서는 COUNT를 사용하되 지역별로 구분하기에 고객주소를 기준으로 잡아 COUNT 해 준 후 이 값으로 순위를 매기는 RANK 컬럼을 출력하면 된다.

<함수설명>

COUNT(tcu.CAddr)

- CAddr 의 값이 NULL 인 값을 제외한 데이터의 개수를 리턴한다.

CAST('20210101' AS TIMESTAMP), CAST('20210401' AS TIMESTAMP)

- 텍스트타입인 '20210101'과 '20210401'을 TIMESTAMP 타입으로 형 변환한다.

34. (INNER JOIN)

제품들의 분류별로 가장 인기 있는 제품을 확인하기 위하여 자주 판매된 제품을 확인하려한다.

이에 해당하는 분류코드와 제품명을 출력하시오.

(분류는 INumber에서 I1XXX, I2XXX 등과 같이 앞의 두 문자로 구분할 수 있으며 만약 판매 횟수가 같으면 제품 코드가 빠른 순으로 출력하고 1위만 출력한다.)

<정답>

<결과 화면>

	분류코드 text	제품명 character varying(20)
1	I1	가위
2	I2	쇼파
3	I3	반지
4	I4	노트북
5	I5	커피

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tBase.INumber AS 분류코드, tBase.IName AS 제품명 ---------- ①
FROM(
    SELECT SUBSTRING(tit.INumber,1,2) AS INumber ------- ②
        , tit.IName AS IName
        , RANK() OVER(PARTITION BY SUBSTRING(tit.INumber,1,2)
          ORDER BY COUNT(tpr.PNumber) DESC, tit.INumber) AS rank
    FROM tOrder AS tor
    JOIN tProduction AS tpr ---------- ③
    ON tor.PNumber = tpr.PNumber
    JOIN tItem AS tit ---------- ④
    ON tit.INumber = tpr.INumber
    GROUP BY tit.IName, tit.INumber ---------- ⑤
) AS tBase
WHERE tBase.rank = 1 ---------- ⑥
```

<해설>

① 문제에 제시된 분류코드와 IName(제품명)을 출력하기 위해 서브쿼리 tBase(별칭)에서 값을 가져온다.

② INumber(제품코드)를 출력하기 위해 제품테이블인 tItem에서 제품코드 컬럼인 INumber를 가져온 후 문제에서 제시된 출생연도 두 자리를 출력하기 위해 부분적으로 문자를 가져오는 SUBSTRING함수를 사용한다.
IName(제품명)을 출력하기 위해 생산테이블인 tProduction에서 제품명 컬럼인 IName을 가져온다.
문제의 조건에 맞는 순위를 출력하기 위해 RANK함수를 사용하였고 COUNT함수를 사용하여 판매된 개수를 가져오고 판매 횟수가 같을 경우 문제의 조건대로 제품코드가 빠른 순서대로 정렬하기 위해서 INumber(제품코드)를 오름차순으로 정렬한다. 판매 횟수가 같을 경우 INumber를 오름차 순한다.
문제의 조건에 맞는 결과인 가장 인기 있는 제품을 출력하되 동순위가 두 명이상 일수 있기에 RANK 함수를 사용하여 순위를 매겨 1위인 제품들을 뽑을 수 있는 조건을 만들었고 PARTITION BY에 SUBSTRING(tit.INumber,1,2) (제품 분류 코드)를 사용함으로써 제품의 분류별로 출력해주었으며 ORDER BY에 존재하는 COUNT(tpr.PNumber) (주문횟수)를 기준으로 하여 순위를 출력해주는데 주문량이 많은 순으로 출력하기 위해 DESC(내림차 순)하여 순위를 출력하면서도 기존의 제품 순서대로 정렬해주기 위해 INumber(제품코드)를 추가로 작성해준다.

③ tProduction(생산)테이블에 있는 PNumber(생산코드)를 가져오기 위해 tOder(주문) 테이블의 PNumber(생산코드)와 tProdunction(생산) 테이블의 PNumber(생산코드)를 JOIN한다.

④ tItem(제품) 테이블에 있는 IName(제품명)을 가져오기 위해 tProduction(생산) 테이블의 INumber(제품코드)와 tItem(제품) 테이블의 INumber(제품코드)를 JOIN한다

⑤ IName(제품명), INumber(제품코드)를 GROUP BY를 사용하여 그룹화 한다.
그룹화 한 IName(제품명), INumber(제품코드)를 기준으로 ②에서 COUNT함수를 사용하여 판매횟수를 구한다

⑥ 문제의 조건인 가장 판매가 많이된 값을 가져오기 위해 서브쿼리 tBase(별칭)에서 관련 컬럼인 RANK에서 1위만 가져온다.

<함수설명>

COUNT(tpr.PNumber)

- PNumber의 값이 NULL인 값을 제외한 데이터의 개수를 리턴한다.

SUBSTRING(tit.INumber,1,2)

- 첫번째 파라미터는 문자열이고 두번째 파라미터는 시작 인덱스입니다. 세번째 파라미터는 가져 올 개수입니다.
  INumber에서 1번째부터 2개까지 문자를 가져옵니다.

35. (SELF JOIN)

현재 근무중인 '김효식'의 평가점수를 매기려한다.

이를 위하여 '김효식'과 동일한 부서에서 근무하는 직원들의 부서코드와 직원명을 출력하시오.

<정답>

<결과 화면>

	부서코드 character varying(5)	직원명 character varying(5)
1	D2001	김구성
2	D2001	박가구
3	D2001	김혜영
4	D2001	이은혜
5	D2001	박천이

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tem1.DNumber AS 부서코드, tem2.EName AS 직원명------ ①
FROM tEmployee AS tem1                                    ------ ②
JOIN tEmployee AS tem2
ON tem1.DNumber = tem2.DNumber
WHERE tem1.EName = '김효식' AND tem2.EName <> '김효식'------ ③
```

<해설>

① 부서코드와 직원명을 출력하기 위해 직원테이블인 tEmployee에서 부서코드 컬럼인 DNumber와 직원명 컬럼인 EName을 가져온다.

② tEmployee(직원) 테이블을 DNumber(직원코드)를 이용하여 SELF JOIN한다.

③ 두 tEmployee 테이블 중 한 테이블은 직원명이 김효식인 직원을 가져오고 다른 테이블은 직원명이 김효식이 아닌 직원을 가져오게 한다.

부서코드와 직원명이 들어있는 테이블인 tEmployee를 SELF JOIN 하면서 tEmployee(직원)테이블을 나란히 합친 결과를 만들어 냄으로써 왼쪽의 tEmployee(직원) 테이블(tem1)에서 김효식 만을 걸러 낸 후 오른쪽의 tEmployee(직원) 테이블(tem2)의 직원들을 뽑으면서 동시에 ON 절에서 DNumber(부서코드)로 JOIN 해주어 같은 부서가 아닌 인원은 한 행의 결과로 출력할 수 없게 하여 김효식과 같은 부서에 있는 직원들만을 출력할 수 있게 되었다.

36. (SELF JOIN)

반지와 귀걸이를 둘 다 제작해본 경험이 있는 직원들이 있는지 알아보려고한다.

위, 조건에 해당하는 직원들의 직원코드를 출력하시오.

<정답>

<결과 화면>

	직원코드 character varying(5)
1	E3001
2	E3003

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT DISTINCT tpr1.ENumber AS 직원코드 ┐ --------- ①
FROM tProduction AS tpr1                ┘
JOIN tProduction AS tpr2                ┐ ---------- ②
ON tpr1.ENumber = tpr2.ENumber          ┘
JOIN tItem AS tit1                                    ┐ ----- ③
ON tpr1.INumber = tit1.INumber AND tit1.IName = '반지'  ┘
JOIN tItem AS tit2                                    ┐ ----- ④
ON tpr2.INumber = tit2.INumber AND tit2.IName = '귀걸이' ┘
```

<해설>

① ENumber(직원코드)를 출력하기 위해 제품테이블인 tProduction에서 직원코드 컬럼인 ENumber를 가져온다.

② tProduction 테이블과 tProduction 테이블을 ENumber를 이용하여 SELF JOIN한다.

③ tItem(제품) 테이블에 있는 IName(제품명)을 가져오기 위해 tProduction(생산) 테이블의 INumber(제품코드)와 tItem(제품) 테이블의 INumber(제품코드)를 JOIN한다
문제의 주어진 조건인 반지와 같은 값을 찾기 위해 tItem 테이블에서 관련 컬림인 IName을 비교한다.

④ tItem(제품) 테이블에 있는 IName(제품명)을 가져오기 위해 tProduction(생산) 테이블의 INumber(제품코드)와 tItem(제품) 테이블의 INumber(제품코드)를 JOIN 한 후 문제의 주어진 조건인 귀걸이와 같은 값을 찾기 위해 tItem(제품) 테이블에서 관련 컬림인 IName(제품명)을 비교한다

IName(제품명)이 반지인 INumber(제품코드) 컬럼을 이용하여 tItem(제품) 테이블과 tProduction(생산) 테이블을 JOIN하고 IName(제품명)이 목걸이인 INumber(제품코드) 컬럼을 이용하여 tItem(제품) 테이블과 tProduction(생산) 테이블을 JOIN이 하고 tProduction(생산) 테이블을 셀프 조인하여 원하는 컬럼을 가져오는 문제다.

37. (FULL OUTER JOIN)

제품들의 주문 및 반품 여부를 확인하고자 한다. 이에 해당하는 제품을 생산 한 직원명, 생산 된 제품명, 생산량, 주문을 한 고객명, 반품량을 전부 출력하세요.

(반품이 존재하지 않더라도 나머지 이력이 있으면 출력해야하며 존재하지않은 정보가 있으면 null로 출력되어야 한다.)

<정답>

<결과 화면>

	생산자명 character varying(5)	생산제품명 character varying(20)	생산량 integer	주문고객 character varying(20)	반품량 integer
1	김문구	풀	32	용호문방구	5
2	박하나	공책	700	오랜문방구	700
3	김구성	쇼파	52	튼튼가구	10
4	박천이	테이블	65	동해가구	65
5	이빛나	반지	325	패션악세사리	18

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tem.EName AS 생산자명, tit.IName AS 생산제품명, tpr.PCount AS 생산량,        ------ ①
       tcu.CName AS 주문고객, tre.RCount AS 반품량
FROM tProduction AS tpr
JOIN tItem AS tit                                                    -------- ②
ON tpr.INumber = tit.INumber
JOIN tEmployee AS tem                                                -------- ③
ON tpr.ENumber = tem.ENumber
FULL OUTER JOIN tOrder AS tor                                        ------- ④
ON tpr.PNumber = tor.PNumber
FULL OUTER JOIN tCustomer AS tcu                                     ------- ⑤
ON tor.CNumber = tcu.CNumber
FULL OUTER JOIN tReturn AS tre                                       ------- ⑥
ON tor.ONumber = tre.ONumber
```

<해설>

① 생산자명을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져오고 생산제품명을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 생산량을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져오고 주문고객을 출력하기 위해 고객테이블인 tCustomer에서 고객명 컬럼인 CName을 가져오고 반품량을 출력하기 위해 반품테이블인 tReturn에서 반품량 컬럼인 RCount를 가져온다.

② tItem(제품) 테이블을 통해서 IName(제품명) 데이터를 가져오기 위하여 tItem(제품) 테이블의 INumber(제품코드)와 tProduction(생산) 테이블의 INumber(제품코드)를 JOIN한다.

③ tEmployee(직원) 테이블을 통해서 EName(직원명) 데이터를 가져오기 위하여 tEmployee(직원) 테이블의 ENumber(직원코드)를 tProduction(생산) 테이블의 ENumber(직원코드)를 JOIN한다.

④ tOrder(주문) 테이블을 통해서 ONumber(주문코드) 데이터를 가져오기 위하여 tProduction(생산) 테이블의 PNumber(생산코드)를 이용하여 JOIN하되 주문되지 않은 생산정보까지 출력해주기 위하여 FULL OUTER JOIN을 사용한다.

⑤ tCustomer(고객) 테이블을 통해서 CName(고객명) 데이터를 가져오기 위하여 tOrder(주문) 테이블의 CNumber(고객코드)를 이용하여 JOIN하되 주문하지 않은 고객정보까지 출력해주기 위하여 FULL OUTER JOIN을 사용한다.

⑥ tReturn(반품) 테이블을 통해서 RCount(반품량) 데이터를 가져오기 위하여 tReturn(반품) 테이블의 ONumber(주문코드)와 tOrder(주문) 테이블의 ONumber(주문코드)를 이용하여 JOIN하되 주문하지 않은 고객정보까지 출력해주기 위하여 FULL OUTER JOIN을 사용한다.

④, ⑤의 경우 주문되지 않은 생산정보와 같이 JOIN한 두 테이블이 서로 관련되지 않은 결과인 NULL값이 출력되더라도 가져와야 하기 때문에 FULL OUTER JOIN을 사용해주었다.

38. (FULL OUTER JOIN)

각 부서별 직원의 분포를 확인하고 하려한다. 이에 따라 각 부서별 부서명과 부서 내 직원들의 직급명, 직급별 인원 수를 출력하세요. (단, 출력결과는 부서, 직급을 오름차순으로 정렬합니다.)

<정답>

<결과 화면>

	부서명 character varying(10)	직급명 character varying(2)	직급별_인원_수 bigint
1	가구생산부	사원	5
2	가구생산부	주임	2
3	가구생산부	대리	2
4	가구생산부	부장	1
5	문구생산부	사원	5

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tde.DName AS 부서명, tra.RName AS 직급명, COUNT(tem.RNumber) AS 직급별_인원_수 ------ ①
FROM tEmployee AS tem
FULL OUTER JOIN tDepartment AS tde       ------- ②
ON tem.DNumber = tde.DNumber
JOIN tRank AS tra          ------------------- ③
ON tra.RNumber = tem.RNumber
WHERE tde.DNumber IS NOT NULL ---------------- ④
GROUP BY tde.DName, tra.RName, tra.RNumber ---------- ⑤
ORDER BY tde.DName, tra.RNumber   ------------ ⑥
```

<해설>

① DName(부서명)을 출력하기 위해 부서테이블인 tDepartment에서 부서명 컬럼인 DName을 가져오고 RName(직급명)을 출력하기 위해 직급테이블인 tReturn에서 직급명 컬럼인 RName을 가져오고 직급별 인원 수를 출력하기 위해 직급테이블인 tReturn에서 직급코드 컬럼인 RNumber를 가져온다 개수를 구하기 위해 COUNT함수를 사용한다

② tDepartment(부서) 테이블에 있는 DName(부서이름)을 가져오기 위해 tEmployee(직원) 테이블의 DNumber(부서코드)와 tDepartmet(부서) 테이블의 DNumber(부서코드)를 JOIN한다. 부서명이 NULL인 직원정보까지 출력해주기 위하여 FULL OUTER JOIN을 사용한다.

③ tRank(직급) 테이블에 있는 RName(직급이름)을 가져오기 위해 tEmployee(직원) 테이블의 RNumber(직급코드)와 tRank(직급) 테이블의 RNumber(직급코드)를 JOIN한다.

④ 문제에 주어진 조건을 위해 ENumber(직원코드)가 NULL이 아니고 DNumber(부서코드)가 NULL이 아닌 값을 가져온다.

⑤ DName(부서명), RName(직급명), RNumber(직급코드)를 GROUP BY를 사용하여 그룹화 하여 ①에서 COUNT 함수를 사용하여 직급별 인원 수를 구한다

⑥ 문제에 주어진 조건인 DName(부서별), RNumber(직급코드)별로 정렬을 한다.

<함수설명>

COUNT(tor.PNumber)

- PNumber의 값이 NULL인 값을 제외한 데이터의 개수를 리턴한다.

39. (CROSS JOIN)

회사 내 제품의 생산량을 확인하기 위해 2020년에 생산된 제품들의 월별 총 생산량과 각 분기별 생산된 제품들의 합을 출력하시오.

(컬럼은 "월", "1분기", "2분기", "3분기", "4분기" 가 되며, 월별생산은 행으로 출력되고 마지막 행은 "분기합"으로 각 분기별 월의 생산량을 더한 값이 출력되어야한다.)

<정답>

<결과 화면>

	2020년 월 text	1분기 bigint	2분기 bigint	3분기 bigint	4분기 bigint
1	01월	8178	0	0	0
2	02월	9495	0	0	0
3	03월	5912	0	0	0
4	04월	0	2615	0	0
	⋮				
13	분기 합	23585	18018	18090	17841

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT
MIN(CASE WHEN tBase.set = 1 THEN TO_CHAR(tpr.PDate,'MM')||'월' ELSE '분기 합' END) AS "2020년 월"-----①
    , SUM(CASE TO_CHAR(tpr.PDate,'MM') WHEN '01' THEN tpr.PCount --------②
                                       WHEN '02' THEN tpr.PCount
                                       WHEN '03' THEN tpr.PCount
                ELSE 0 END) AS "1분기"
    , SUM(CASE TO_CHAR(tpr.PDate,'MM') WHEN '04' THEN tpr.PCount
                                       WHEN '05' THEN tpr.PCount
                                       WHEN '06' THEN tpr.PCount
                ELSE 0 END) AS "2분기"
    , SUM(CASE TO_CHAR(tpr.PDate,'MM') WHEN '07' THEN tpr.PCount
                                       WHEN '08' THEN tpr.PCount
                                       WHEN '09' THEN tpr.PCount
                ELSE 0 END) AS "3분기"
    , SUM(CASE TO_CHAR(tpr.PDate,'MM') WHEN '10' THEN tpr.PCount
                                       WHEN '11' THEN tpr.PCount
                                       WHEN '12' THEN tpr.PCount
                ELSE 0 END) AS "4분기"
FROM tProduction AS tpr
CROSS JOIN (SELECT 1 AS "set" UNION SELECT 2 AS "set") AS tBase--------③
WHERE TO_CHAR(tpr.PDate,'YYYY') = '2020' ----------------------------------④
GROUP BY CASE WHEN tBase.set = 1 THEN TO_CHAR(tpr.PDate,'MM') END------⑤
ORDER BY "2020년 월"-------------------------------------------------------⑥
```

<해설>

① 생산일자를 출력하기 위해 생산테이블인 tProduction에서 생산일자 컬럼인 PDate를 가져오고 난 후 TO_CHAR 함수를 이용하여 월 단위로 끊어주고 CASE WHEN과 아래의 CROSS JOIN(SELECT 1 AS "set" UNION SELECT 2 AS "set")을 이용하여 월 별 생산량과 분기 별 총합 생산량을 나누어 출력 하였다.

② 생산량을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져오고 난 후 생산량의 총 합을 구하기 위하여 SUM 함수를 사용하고 CASE WHEN을 통하여 각 월 별 생산량 총 합을 출력하도록 하였다.

③ SELECT 1 AS "set" UNION SELECT 2 AS "set"을 CROSS JOIN 해 줌으로써 tProduction(생산) 테이블이 두 번 출력되는 것처럼 결과를 나타냈다.

④ 문제의 조건인 2020년과 같은 값을 가져오기 위해 TO_CHAR 함수를 이용하여 생산테이블인 tProduction에서 관련 컬럼인 PDate와 비교한다

⑤ 그룹화 한 각 월(1월,2월...)을 기준으로 월별 총 생산량과 분기별 총합 생산량을 구한다

⑥ 정렬 기준인 2020년 월 컬럼을 오름차순정렬을 하기 위해 ORDER BY 를 사용한다

우선 생산량을 출력해야 하기에 생산테이블인 tProduction을 가져온 후 FROM 절에서 1과 2 두 값을 넣어준 tBase(별칭)를 CROSS JOIN 해 줌으로써 같은 값을 두 번 반복하여 갖게끔하여 tProduction 데이터를 두 셋트 만드는 것처럼 나타내었다.

그리고 두쌍이 된 tProduction 테이블 중 set컬럼의 1 데이터와 매칭된 값들은 월별 총생산량을 구하기 위해서 group by 절에 case when 구문으로 명시하고 있고 set 컬럼의 2 데이터는 명시하지 않았기 때문에 null이되어서 차후 분기 전체 데이터의 총생산량을 구하기 위한 준비를 하게 되었다.

그럼 select절을 보면, 문제에서 요구한 첫 컬럼명인 월을 표현하기 위해서 별칭으로 2020년 월이라고 적었고 case when 구문으로 group by절에 적었던 것처럼 set컬럼의 1 데이터는 월을 표현하고 있는것을 알 수 있다. 그리고 set컬럼이 1이 아닌 2데이터는 분기들의 각각 합을 표출하기위해서 else로 분기 합이라고 표시 해 두었다.

MIN함수로 묶은 이유는 현재 tBase.set 컬럼이 group by절에 없기 때문에 해당 컬럼을 사용하기 위해서는 집계함수로 표현해야하므로 MIN함수를 사용하여 표현하고 있다.

두번째 컬럼을 보면 문제의 조건대로 1분기라는 별칭의 case when 구문으로 1분기에 해당하는 1, 2, 3월은 sum 함수로 총생산량을 표시하도록 하고 있다. 마찬가지로 3번째 컬럼 4번째 컬럼 5번째컬럼들도 동일한 방식으로 2분기, 3분기, 4분기를 표시하고 있다. 마지막으로 월별로 차례대로 표현하기 위해서 order by 절에 첫번째 열에 해당하는 별칭 2020년 월을 적어 줌으로써 쿼리를 구성하고 있다.

<함수설명>

MIN(CASE WHEN tBase.set = 1 THEN TO_CHAR(tpr.PDate,'MM')||'월' ELSE '분기 합' END)

- 출력 된 TO_CHAR(tpr.PDate,'MM') 중에서 제일 최솟값, 제일 먼저 오는 값을 반환한다.

SUM(CASE TO_CHAR(tpr.PDate,'MM') WHEN '01' THEN PCount

WHEN '02' THEN PCount

WHEN '03' THEN PCount

ELSE 0 END)

- 결과 값인 PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다.

TO_CHAR(tpr.PDate,'YYYY')

- 첫번째 파라미터인 PDate 는 Timestamp타입이고, 두번째 파라미터인 'YYYY' 포맷에 맞춰 '년도'의 텍스트타입으로 리턴한다.

40. (CROSS JOIN)

2020년 연말정산으로써 전 직원의 월별 생산 횟수 및 월별 생산 합계를 파악하려 한다.

위의 내용에 맞춰 결과를 출력하시오.

(컬럼은 "년도", "사원", "1월"~"12월" 이 되며, 행마다 직원명과 월별 생산횟수가 출력되어야 하고 마지막 행은 각 월별 생산횟수 합계가 표시되어야한다.)

<정답>

<결과 화면>

	년도 text	사원 text	1월 bigint	2월 bigint	3월 bigint	4월 bigint	5월 bigint	6월 bigint	7월 bigint	8월 bigint	9월 bigint	10월 bigint	11월 bigint	12월 bigint
1		이정보	1	0	1	0	0	0	0	0	0	0	0	0
2		류승환	0	0	0	0	0	0	1	0	0	0	0	0
3		박기술	1	0	0	0	0	0	0	0	0	0	0	0
4		이은혜	1	0	1	0	0	0	1	0	0	0	0	0
⋮														
52	2020년	월별 합계	14	26	14	5	5	5	6	4	5	4	6	6

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT MIN(CASE WHEN tBase.set = 1 THEN TO_CHAR(tpr.PDate,'YYYY')||'년' ELSE '' END) AS "년도" -------- ①
    , MIN(CASE WHEN tBase.set = 2 THEN tem.EName ELSE '월별 합계' END) AS "사원"
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '01' THEN tem.EName END) AS "1월" ------ ②
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '02' THEN tem.EName END) AS "2월"
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '03' THEN tem.EName END) AS "3월"
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '04' THEN tem.EName END) AS "4월"
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '05' THEN tem.EName END) AS "5월"
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '06' THEN tem.EName END) AS "6월"
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '07' THEN tem.EName END) AS "7월"
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '08' THEN tem.EName END) AS "8월"
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '09' THEN tem.EName END) AS "9월"
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '10' THEN tem.EName END) AS "10월"
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '11' THEN tem.EName END) AS "11월"
    , COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '12' THEN tem.EName END) AS "12월"
FROM tProduction AS tpr
CROSS JOIN (SELECT 1 AS "set" UNION SELECT 2 AS "set") AS tBase -------- ③
JOIN tEmployee AS tem                  ----------------- ④
ON tpr.ENumber = tem.ENumber
WHERE TO_CHAR(tpr.PDate,'YYYY') = '2020'----------- ⑤
GROUP BY CASE WHEN tBase.set = 2 THEN tem.EName END---------------- ⑥
ORDER BY "년도", "사원" ------------ ⑦
```

<해설>

① 생산일자를 출력하기 위해 생산테이블인 tProduction에서 생산일자 컬럼인 PDate를 가져오고 난 후 TO_CHAR 함수를 이용하여 연 단위로 끊어주고 CASE WHEN과 아래의 CROSS JOIN(SELECT 1 AS "set" UNION SELECT 2 AS "set")을 이용하여 월 별 생산횟수와 연 별 총 생산횟수를 나누어 출력하고 직원명을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져오고 난 후 앞의 설명과 같이 사원별 생산 횟수와 사원 전체의 월별 생산횟수로 나누어 출력 해 주었다.

② 생산횟수를 출력하기 위해 생산테이블인 tProduction에서 생산일자 컬럼인 PDate를 가져오고 난 후 생산횟수를 구하기 위하여 COUNT 함수를 사용하고 CASE WHEN을 통하여 월 별 생산횟수를 출력하도록 하였다.

③ SELECT 1 AS "set" UNION SELECT 2 AS "set"을 CROSS JOIN 해 줌으로써 tProduction(생산) 테이블이 두 번 출력되는 것처럼 결과를 나타냈다.

④ tEmployee(직원) 테이블에 있는 EName(직원명)을 가져오기 위해 tEmployee(직원) 테이블의 ENumber(직원코드)와 tProduction(생산)테이블의 Eumber(직원코드)를 JOIN한다.

⑤ 문제의 조건인 2020년인 값만 가져오기 위해 생산 테이블인 tProduction에서 관련 컬럼인 PDate와 비교한다.

⑥ tBase.Set이 2일 경우 직원이름으로 그룹화하여 각 월별 생산횟수를 출력한다. 그리고 Set 컬럼의 1 데이터는 명시하지 않았기 때문에 null이 되어서 차후 월별 전체 데이터의 총 생산횟수를 구하기 위한 준비를 하게 된다.

⑦ 정렬 기준인 년도 별 컬럼을 오름차순으로 정렬하기 위해 ORDER BY를 사용하고 출력된 결과를 직원명 순으로 다시 오름차순 정렬하기 위하여 사원 컬럼을 추가로 사용해주었다.

<함수설명>

MIN(CASE WHEN set = 2 THEN tem.EName ELSE '월별 합계' END)

- 출력 된 EName 중에서 제일 최솟값, 제일 먼저 오는 값을 반환한다.

COUNT(CASE WHEN TO_CHAR(tpr.PDate,'MM') = '01' THEN tem.EName END)

- EName 의 값이 NULL인 값을 제외한 데이터의 개수를 리턴한다.

TO_CHAR(tpr.PDate,'MM')

- 첫번째 파라미터인 PDate는 Timestamp타입이고, 두번째 파라미터인 'MM' 포맷에 맞춰 '월'의 텍스트타입으로 리턴한다.

심화 1.

각 제품의 분류별 1등 고객(가장 구매 횟수가 많은 고객)에게는 해당 부서에서 제공하는 혜택이 주어진다. 각 부서별 부서명과 위의 조건에 해당하는 고객명을 출력하시오.

(단, 부서별로 동률인 고객이 존재할 수 있으므로 1등 고객이 여러명일경우 모두 출력한다.)

<정답>

<결과 화면>

	부서명 character varying(10)	고객명 character varying(20)
1	가구생산부	동해가구
2	가구생산부	그린가구
3	문구생산부	용호문방구
4	문구생산부	오랜문방구
5	문구생산부	최고문방구

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT DName AS 부서명, CName AS 고객명 ---------①
FROM
(
    SELECT *                                                                    ----------②
        , RANK() OVER(PARTITION BY DName ORDER BY tBase.cnt DESC) AS Seq
    FROM
    (
        SELECT   tde.DName AS DName                                             -----------③
            , tcu.CName
            , COUNT(tcu.CName) AS cnt
        FROM tOrder AS tor
        JOIN tCustomer AS tcu                                                   ----------④
        ON tor.CNumber = tcu.CNumber
        JOIN tProduction AS tpr                                                 ----------⑤
        ON tor.PNumber = tpr.PNumber
        JOIN tEmployee AS tem                                                   ----------⑥
        ON tem.ENumber = tpr.ENumber
        JOIN tDepartment AS tde                                                 ----------⑦
        ON tde.DNumber = tem.DNumber
        GROUP BY tde.DName ,tcu.CName ----------------⑧
    ) AS tBase
) AS tBase2
WHERE Seq = 1-----------⑨
ORDER BY DName--------⑩
```

<해설>

① tBase(별칭) 서브쿼리에서 결과물을 출력한 DName (부서명), CName (고객명) 컬럼을 가져온다.

② tBase(별칭) 서브쿼리에서 출력된 결과물인 DName (부서명), CName (고객명), cnt(고객주문횟수) 컬럼을 가져와서 순위를 매길 때 문제의 조건에 맞는 결과인 분류별 1등 고객을 출력하되 1등이 여러 명일수도 있기 때문에 RANK를 사용하였고 PARTITION BY에 부서명 컬럼인 DName을 사용함으로써 각 부서별 결과를 출력해주었으며 ORDER BY에 존재하는 cnt(고객주문횟수)로 순위를 출력해준다.

③ 부서명을 출력하기 위해 부서테이블인 tDepartment에서 부서명 컬럼인 DName을 가져오고 고객명을 출력하기 위해 고객테이블인 tCustomer에서 고객명 컬럼인 CName을 가져온 후 고객별 구매 횟수를 확인하기 위하여 COUNT 함수를 사용한다.

④ tCustomer(고객) 테이블을 통해서 CName(고객명) 데이터를 가져오기 tOrder(주문) 테이블의 CNumber(고객코드)와 tCustomer(고객)테이블의 CNumber(고객코드)를 JOIN한다.

⑤ tProduction(생산) 테이블을 통해서 PNumber(생산코드) 데이터를 가져오기 위하여 tProduction(생산) 테이블의 PNumber(생산코드)와 tOrder(주문) 테이블의 PNumber(생산코드)를 JOIN한다.

⑥ tEmployee(직원) 테이블을 통해서 ENumber(직원코드) 데이터를 가져오기 위하여 tEmployee(직원) 테이블의 ENumber(직원코드)와 tProduction(생산) 테이블의 ENumber(직원코드)를 JOIN한다.

⑦ tDepartment(부서) 테이블을 통해서 DName(부서명) 데이터를 가져오기 위하여 tDepartment(부서) 테이블의 DNumber(부서코드)와 tEmployee(직원) 테이블의 DNumber(부서코드)를 JOIN한다.

⑧ 그룹화 한 DName(부서명)과 CName(고객명)을 기준으로 ③에서 COUNT 함수를 사용하여 총 구매횟수를 구한다

⑨ ②에서 매겨준 순위 중에서 제일 많은 구매 횟수를 가진 고객 1명, 즉 1등 고객을 출력하기 위해 비교한다.

⑩ 부서명을 오름차순 정렬을 하기 위해 ORDER BY 를 사용한다

<함수설명>

COUNT(tcu.CName)

- CName 의 값이 NULL 인 값을 제외한 데이터의 개수를 리턴한다.

심화 2.

제품의 생산량 조절 및 반품에 대한 패널티를 부여하기 위하여

반품율이 가장 높은 부서명과 해당 부서에서 반품률이 가장 높은 제품의 이름을 출력하시오

(반품률 계산 시 반품내역이 없는 값(null)은 0으로 대체 되어 출력되어야 한다.)

<정답>

<결과 화면>

	부서명 character varying(10)	제품명 character varying(20)
1	악세사리생산부	목걸이

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT DName AS 부서명, IName AS 제품명   ---------- ①
FROM
(
    SELECT DName, IName,                                                                    --------- ②
        RANK() OVER(PARTITION BY DName ORDER BY itemrratio DESC) AS itemSeq,
        RANK() OVER(ORDER BY departmentrratio DESC) AS departmentSeq
    FROM
    (
        SELECT DISTINCT tde.DName, tit.IName,                                                   -- ③
            SUM(tpr.PCount) OVER(PARTITION BY tit.IName) AS totCount,
            COALESCE(SUM(tre.RCount) OVER(PARTITION BY tit.IName), 0) AS RCount,
            COALESCE(CAST(CAST(SUM(tre.RCount) OVER(PARTITION BY tit.IName) AS FLOAT) /
            CAST(SUM(tpr.PCount) OVER(PARTITION BY tit.IName) AS FLOAT) AS DECIMAL), 0) AS itemrratio,
            CAST(SUM(tre.RCount) OVER(PARTITION BY tde.DName) AS FLOAT) /
            CAST(SUM(tpr.PCount) OVER(PARTITION BY tde.DName) AS FLOAT) AS departmentrratio
        FROM tOrder AS tor
        JOIN tProduction AS tpr          ---------- ④
        ON tpr.PNumber = tor.PNumber
        LEFT OUTER JOIN tReturn AS tre   ---------- ⑤
        ON tre.ONumber = tor.ONumber
        JOIN titem AS tit                --------- ⑥
        ON tit.INumber = tpr.INumber
        JOIN tEmployee AS tem            ---------- ⑦
        ON tem.ENumber = tpr.ENumber
        JOIN tDepartment AS tde          ---------- ⑧
        ON tde.DNumber = tem.DNumber
        ORDER BY tde.DName, tit.IName    ---------- ⑨
    )AS tBase
```

```
)AS tBase2
WHERE itemseq = 1 AND departmentseq = 1 -------- ⑩
```

<해설>

① 서브쿼리 tBase2(별칭)에서 문제에서 제시한 DName(부서명)과 IName(제품명)을 가져온다.

② 서브쿼리 tBase(별칭)에서 DName(부서명)과 IName(제품명)을 가져온 후 반품률이 가장 높은 제품의 순위를 출력하되 동순위가 두 명이상 일 수 있기에 RANK를 사용하여 1위인 부서와 제품들을 뽑을 수 있는 조건을 만들었고 PARTITION BY에 부서명 컬럼인 DName을 사용함으로써 각 부서별 반품이 된 제품을 출력해주고 ORDER BY에 존재하는 itemrratio(제일 반품이 많이 됨 제품)을 기준으로 순위를 매겨 부서 별 제품의 반품순위 출력해주었고(itemSeq), 반품률을 기준으로 부서의 순위를 출력해주되 1위 부서가 2개 일 수도 있기에 RANK를 사용해주고 departmentrratio(부서 별 반품률)을 기준으로 순위를 출력해 주었다.

③ DName(부서명)을 출력하기 위해 부서테이블인 tDepartment에서 부서명 컬럼인 DName을 가져온다.
DISTINCT를 사용하여 중복을 제거한다.
IName(제품명)을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져온다.
PCount(생산량)을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온다.
합을 위해 SUM함수를 사용한다.
Itemrratio(제품 별 반품률)을 출력하기 위해 SUM(tre.RCount)(총 반품량) / SUM(tpr.PCount)(총 생산량)을 하였고 PARTITION BY에 제품명 컬럼인 IName을 사용함으로써 각 제품명 별로 구하는 GROUP BY 역할을 해주었고 COALESCE함수를 사용해 NULL인 값은 0으로 출력한다.
departmentrratio (부서 별 반품률)을 출력하기 위해 SUM(tre.RCount)(총 반품량) / SUM(tpr.PCount)(총 생산량)을 하였고 PARTITION BY에 부서명 컬럼인 DName을 사용함으로써 각 부서명 별로 구하는 GROUP BY 역할을 해주었고 COALESCE함수를 사용해 NULL인 값은 0으로 출력한다.

④ tProduction(생산) 테이블에 있는 PCount(생산량)을 가져오기 위해 tOrder(주문) 테이블의 PNumber(생산코드)와 tProduction(생산) 테이블에 있는 PNumber(생산코드)를 JOIN한다.

⑤ tReturn(반품) 테이블에 있는 RCount(반품량)을 가져오기 위해 tOrder(주문) 테이블의 ONumber(주문코드)와 tReturn(반품) 테이블에 있는 RNumber(주문코드)를 JOIN한다.
반품하지 않은 주문정보까지 출력해주기 위하여 LEFT OUTER JOIN을 사용한다.

⑥ tItem(제품) 테이블에 있는 INmae(제품명)을 가져오기 위해 tProduction(생산) 테이블의 INumber(제품코드)와 tItem(제품) 테이블에 있는 INumber(제품코드)를 JOIN한다.

⑦ tEmployee(직원) 테이블에 있는 DNumber(부서코드)를 가져오기 위해 tProduction(생산) 테이블의 ENumber(직원코드)와 tEmployee(직원) 테이블에 있는 ENumber(직원코드)를 JOIN한다.

⑧ tDepartment(부서) 테이블에 있는 DName(부서명)을 가져오기 위해 tEmployee(직원) 테이블의 DNumber(부서코드)와 tDepartment(부서) 테이블에 있는 DNumber(부서코드)를 JOIN한다.

⑨ DName(부서명) 과 IName(제품명)을 오름차순으로 정렬한다.

⑩ 문제에서 제시한 반품량이 1등인 부서면서 해당 부서에서 가장 반품률이 높은 제품을 출력한다

<함수설명>

SUM(tpr.PCount), SUM(tre.RCount)

- PCount, RCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

CAST(SUM(tre.RCount) OVER(PARTITION BY tde.DName) AS FLOAT)

- NUMERIC타입인 SUM(tre.RCount)를 DName별로 더한 후 FLOAT 타입으로 형 변환 한다.

COALESCE(SUM(tre.RCount) OVER(PARTITION BY tit.IName), 0)

- IName별로 값을 더한 SUM(tre.RCount) 의 값이 NULL일 때 0으로 출력한다.

심화 3.

월별 제품 선호도 조사를 위하여 2021년의 월별로 제일 많이 판매된 제품을 확인하려 한다.

위의 조건에 맞는 결과를 해당 월, 제품명, 총 판매수량으로 출력하시오.

(단, 반품여부는 생각하지 않는다.)

<정답>

<결과 화면>

	월 text	제품명 character varying(20)	총_판매수량 bigint
1	01월	커피	2746
2	02월	탄산음료	1200
3	03월	홍차	3753
4	04월	팥찌	604
5	05월	전통차	1431

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT Month||'월' AS 월,  ------①
    IName AS 제품명,
    PCount AS 총_판매수량
FROM
(
    SELECT TO_CHAR(tor.ODate,'MM') AS Month,  ------②
        tit.IName,
        SUM(tpr.PCount) AS PCount,
        RANK() OVER (PARTITION BY TO_CHAR(tor.ODate,'MM') ORDER BY SUM(tpr.PCount) DESC) AS Seq -----③
    FROM tOrder AS tor
    JOIN tProduction AS tpr  -------④
    ON tor.PNumber = tpr.PNumber
    JOIN tItem AS tit  -------⑤
    ON tit.INumber = tpr.INumber
    WHERE TO_CHAR(tor.ODate,'YYYY')='2021'----------⑥
    GROUP BY TO_CHAR(tor.ODate,'MM'), tit.IName----⑦
) AS tBase
WHERE Seq = 1--------⑧
ORDER BY Month------⑨
```

<해설>

① tBase(별칭) 서브쿼리에서 결과물을 출력한 MONTH(주문 월), INAME(제품명), PCount(판매량) 컬럼을 가져온다.

② Month(주문 월)을 출력하기 위해 주문테이블인 tOrder에서 주문일자 컬럼인 ODate에서 TO_CHAR()를 사용하여 주문월만 가져오고 IName(제품명)을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 총 판매량을 출력하기 위해 생산테이블인 tProduction에서 주문테이블인 tOrder에 존재하는 PNumber(주문코드)와 일치하는 생산량 컬럼인 PCount를 가져온 후 합을 구하기 위해 SUM함수를 사용한다.

③ 문제의 조건에 맞는 결과인 가장 많이 판매된 물품만을 출력하되 동순위가 두 명이상 일수 있기에 RANK 함수를 사용하여 순위를 매겨 1위인 제품들을 뽑을 수 있는 조건을 만들었고 PARTITION BY에 TO_CHAR(tor.ODate,'MM')(주문 월)를 사용함으로써 각 주문 월별 판매수량 출력해주었으며 ORDER BY에 존재하는 SUM(tpr.PCount) (총 판매량)을 기준으로 하여 순위를 출력해준다.

④ tProduction(생산) 테이블에 있는 PCount(생산량)을 가져오기 위해 tProduction(제품) 테이블의 PNumber(생산코드)와 tOrder(주문)테이블의 PNumber(생산코드)를 JOIN한다.

⑤ tItem(제품) 테이블을 통해서 INumber(제품코드) 데이터를 가져오기 위하여 tItem(제품) 테이블의 INumber(제품코드)와 tProduction(생산) 테이블의 INumber(제품코드)를 JOIN한다.

⑥ 문제의 조건인 2021년과 같은 값을 가져오기 위해 TO_CHAR함수를 이용하여 주문테이블인 tOrder에서 관련 컬럼인 ODate(주문일자)와 비교한다

⑦ 그룹화 한 주문월, IName(제품명)을 기준으로 ②에서 SUM함수를 사용하여 총 판매량을 구한다

⑧ ③에서 매겨준 순위 중에서 제일 많은 구매 횟수를 가진 고객 1명을 출력하기 위해 비교한다.

⑨ 정렬 기준인 주문 월 별로 오름차순으로 정렬하기 위해 ORDER BY 를 사용한다.

<함수설명>

TO_CHAR (tor.ODate,'MM'))

- 첫번째 파라미터인 ODate는 Timestamp타입이고, 두번째 파라미터인 'MM' 포맷에 맞춰 '월'의 텍스트타입으로 리턴한다.

SUM(tpr.PCount)

- PCount 의 데이터 타입이 숫자일 때 NULL 인 값을 제외한 모든 값을 더해 리턴한다.

심화 4.

2021년 음료생산부의 판매실적을 확인하기 위하여 월별로 판매량을 출력하세요. 출력된 결과는 주문일을 기준으로 오름차순 정렬하세요. (음료류의 제품코드(INumber)는 I500(1~5)이다.)

<정답>

<결과 화면>

	월 text	월별_판매량 bigint
1	01월	7153
2	02월	1631
3	03월	6465
4	05월	1431
5	07월	4310

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT TO_CHAR(tor.ODate,'MM')||'월' AS 월,  --------- ①
    SUM(tpr.PCount) AS 월별_판매량
FROM tOrder AS tor
JOIN tProduction AS tpr                      ---------- ②
ON tor.PNumber = tpr.PNumber
JOIN tEmployee AS tem                        ---------- ③
ON tpr.ENumber = tem.ENumber
JOIN tDepartment AS tde                      ---------- ④
ON tem.DNumber = tde.DNumber
WHERE tde.DName = '음료생산부' AND TO_CHAR(tor.ODate,'YYYY') = '2021' -------- ⑤
GROUP BY TO_CHAR(tor.ODate,'MM') ----------- ⑥
ORDER BY TO_CHAR(tor.ODate,'MM') ----------- ⑦
```

<해설>

① ODate(주문일)을 출력하기 위해 주문테이블인 tOrder에서 주문일 컬럼인 ODate를 가져오고 PCount(생산량)을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온다. 합을 위해 SUM함수를 사용한다

② tProduction(생산) 테이블에 있는 PCount(생산량)을 가져오기 위해 tOrder(주문) 테이블의 ONumber(생산코드)와 tProduction(생산) 테이블의 ONumber(생산코드)를 JOIN한다.

③ tEmployee(직원) 테이블에 있는 DNumber(부서코드)을를가져오기 위해 tProduction(생산) 테이블의 ENumber(직원코드)와 tEmployee(직원) 테이블의 ENumber(직원코드)를 JOIN한다.

④ tDepartment(부서) 테이블에 있는 DName(부서이름)을 가져오기 위해 tEmployee(직원) 테이블의 DNumber(부서코드)와 tDepartmet(부서) 테이블의 DNumber(부서코드)를 JOIN한다.

⑤ 문제에 주어진 조건인 음료생산부와 같은 값을 가져오기 위해 부서테이블인 tDepartment에서 관련 컬럼인 DName(부서명)과 비교하고 그 다음 주어진 조건인 2021과 같은 값을 가져오기 위해 주문테이블인 tOrder에서 관련 컬림인 ODate(주문일자)와 비교한다.

⑥ TO_CHAR함수를 사용하여 월 형태로 표시된 주문일을 GROUP BY를 사용하여 그룹화 한다.
그룹화 한 ODate(주문일자)를 기준으로 ①에서 SUM함수를 사용하여 생산량을 구한다

⑦ 정렬 기준인 ODate(주문일자)를 오름차 순 정렬하기 위해 ORDER BY를 사용한다.

<함수설명>

TO_CHAR(tor.ODate,'MM'), TO_CHAR(tor.ODate,'YYYY')

- 첫번째 파라미터인 ODate는 Timestamp타입이고, 두번째 파라미터인 'MM', 'YYYY' 포맷에 맞춰 '월', '연' 의 텍스트타입으로 리턴한다.

SUM(tpr.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

심화 5.

2021년 1월 직급별 평균 월급과 직급별로 월급이 제일 많았던 직원의 직원명, 월급 금액을 쓰시오
(직급별 직급명이 출력되어야 하며 직원들의 월급은 본인이 생산한 물품들의 판매금액 중 1%이고 소숫점 2자리 까지 반올림되어 출력되어야 한다.)

<정답>

<결과 화면>

	직급명 character varying(2)	직급_평균_급여 numeric	직급내_최대_급여자 character varying(5)	최대_급여자_월급 bigint
1	사원	1193167.50	박기술	4988000
2	주임	61888.75	박수인	139380
3	대리	1201646.25	박은혜	4910360
4	과장	8019500.00	박호승	8019500
5	차장	72720.00	김대한	72720

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT RName AS 직급명,                                                    -----①
    davg AS 직급_평균_급여,
    EName AS 직급내_최대_급여자,
    eSalary AS 최대_급여자_월급
FROM
(
    SELECT *,                                                              -----②
        RANK() OVER(PARTITION BY RName ORDER BY eSalary DESC) AS eSeq,
        ROUND(AVG(eSalary) OVER(PARTITION BY RName),2) AS dAvg
    FROM
    (
        SELECT   tra.RNumber,                                              -----③
            tra.RName,
            tem.EName,
            SUM(tpr.PCount*tit.Price)/100 AS eSalary
        FROM tOrder AS tor
        JOIN tProduction AS tpr                                            -----④
        ON tor.PNumber = tpr.PNumber
        JOIN tEmployee AS tem                                              -----⑤
        ON tpr.ENumber = tem.ENumber
        JOIN tRank AS tra                                                  -----⑥
        ON tem.RNumber = tra.RNumber
        JOIN tItem AS tit                                                  -----⑦
        ON tpr.INumber = tit.INumber
        WHERE TO_CHAR(tor.ODate,'YYYY-MM') = '2021-01'-----⑧
```

```
        GROUP BY tra.RNumber, tra.RName, tem.EName ------⑨
    )AS tBase
)AS tBase2
WHERE eSeq = 1------⑩
ORDER BY RNumber---⑪
```

<해설>

① tBase2(별칭) 서브쿼리에서 결과물을 RName(직급명), davg(직급 평균 급여), EName(직급내 최대 급여자), eSalary(최대 급여자 월급) 컬럼을 가져온다.

② tBase(별칭) 서브쿼리에서 결과물을 RNunmber(직급코드), RName(직급명), EName(직원명), eSalary(월급) 컬럼을 가져오고 문제의 조건에 맞는 결과인 가장 많은 월급을 받은 직원을 출력하되 동순위가 두 명이상 일수 있기에 RANK 함수를 사용하여 순위를 매겨 1위인 사람들을 뽑을 수 있는 조건을 만들었고 PARTITION BY에 직급명 컬럼인 RName을 사용함으로써 각 직급별 월급 현황을 출력해주었으며 ORDER BY에 존재하는 eSalary(월급)을 기준으로 하여 순위를 매긴 컬럼인 eSeq와 AVG 함수를 사용해 직급별 평균 월급을 나타낼 컬럼인 davg를 출력해준다.

③ RNumber(직급코드)와 RName(직급명)을 출력하기 위해 직급테이블인 tRank에서 직급코드와 직급명 컬럼인 RNumber, RNumber를 가져오고 EName(직원명)을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져오고 월급을 계산해주기 위해 필요한 PCount(생산량)과 Price(단가)를 출력하기 위해 각각 tProduction(생산), tItem(제품) 테이블에서 생산량 컬럼인 PCount와 단가 컬럼인 Price를 가져와서 합을 구하기 위해 SUM함수를 사용한 후 나누기 100을 해주어 eSalary(월급)을 만들었다.

④ tProduction(생산) 테이블에 있는 PCount(생산량)을 가져오기 위해 tProduction(제품) 테이블의 PNumber(생산코드)와 tOrder(주문)테이블의 PNumber(생산코드)를 JOIN한다.

⑤ tEmployee(직원) 테이블에 있는 EName(직원명)을 가져오기 위해 tEmployee(직원) 테이블의 ENumber (직원코드)와 tProduction(생산) 테이블의 ENumber (직원코드)를 JOIN한다.

⑥ tRank(직급) 테이블에 있는 RNumber(직급코드), RName(직급명)을 가져오기 위해 tRank(직급) 테이블의 RNumber(직급코드)와 tEmployee(직원) 테이블의 RNumber(직급코드)를 JOIN한다.

⑦ tItem(제품) 테이블을 통해서 INumber(제품코드) 데이터를 가져오기 위하여 tItem(제품) 테이블의 INumber(제품코드)와 tProduction(생산) 테이블의 INumber(제품코드)를 JOIN한다.

⑧ 문제의 조건인 2021년 1월과 같은 값을 가져오기 위해 TO_CHAR함수를 이용하여 주문테이블인 tOrder에서 관련 컬럼인 ODate와 비교한다

⑨ 그룹화 한 RNumber(직급코드), RName(직급명), EName(직원명)을 기준으로 ③에서 SUM함수를 사용하여 (총 판매량 * 단가) 를 구한다

⑩ ②에서 매겨준 순위 중에서 직급 별로 월급이 제일 많은 직원 1명을 출력하기 위해 비교한다.

⑪ 정렬 기준인 RNumber(직급코드)를 오름차순정렬을 하기 위해 ORDER BY 를 사용한다

<함수설명>

ROUND(AVG(esalary) OVER(PARTITION BY RName),2)

- AVG(esalary)의 데이터 타입이 숫자 일 때 소수점 2자리까지 반올림한다

AVG(esalary)

- esalary의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더하고 개수만큼 나눠 평균을 리턴한다.

SUM(tpr.PCount*tit.Price)

- SUM(tpr.PCount*tit.Price) 의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다.

TO_CHAR(tor.ODate,'YYYY-MM')

- 첫번째 파라미터인 ODate 는 Timestamp 타입이고, 두번째 파라미터인 'YYYY-MM' 포맷에 맞춰 '연-월'의 텍스트타입으로 리턴한다.

심화 6.

2022년 1월 이전 판매내역과 현재 재고량을 통해서 판매 중인 제품들의 고객선호도를 조사하시오.

(재고량 / 생산량) 을 통하여 수치가 제일 낮은 물품을 고객선호도가 높은 물품으로 판단한다.

(재고량은 팔리지 않았거나 반품 사유가 불량(RR0001)이 아닌 사유로 반품된 물품들을 뜻함, 선호도는 소숫점 2자리 이후는 반올림한 후 출력되어야 한다.)

<정답>

<결과 화면>

	제품명 character varying(20)	고객선호도 numeric
1	팔찌	0.10
2	컴퓨터	0.16
3	침대	0.18
4	홍차	0.24
5	책상	0.27

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tBase.IName AS 제품명, ROUND(CAST(CAST(AVG(tBase.PCount) AS FLOAT) /        -------- ①
        CAST(SUM(tpr.PCount) AS FLOAT) AS DECIMAL),2) AS 고객선호도
FROM tProduction AS tpr
JOIN                                                                                -------- ②
(
    SELECT tit.IName, tit.INumber, SUM(tpr.PCount) AS PCount    ---------- ③
    FROM tProduction AS tpr
    LEFT OUTER JOIN tOrder AS tor                                                   --- ④
    ON tor.PNumber = tpr.PNumber AND TO_CHAR(ODate,'YYYYMM') < '202201'
    LEFT OUTER JOIN tReturn AS tre    ---------- ⑤
    ON tre.ONumber= tor.ONumber
    LEFT OUTER JOIN tReturnReason AS trr                        --------- ⑥
    ON tre.RRNumber = trr.RRNumber AND trr.RReason <> '불량'
    JOIN tItem AS tit                 ---------- ⑦
    ON tpr.INumber = tit.INumber
    WHERE tor.ONumber IS NULL OR trr.RRNumber IS NOT NULL ---------- ⑧
    GROUP BY tit.IName, tit.INumber   ---------- ⑨
)AS tBase
ON tpr.INumber = tBase.INumber
GROUP BY tBase.IName   ---------- ⑩
ORDER BY 고객선호도 ASC   ---------- ⑪
```

<해설>

① 서브쿼리 tBase(별칭)에서 IName(제품명)을 출력한다. (재고량 / 생산량)을 계산한 값을 ROUND함수를 사용하여 반올림 한다.

② 서브쿼리 tBase(별칭)와 생산테이블인 tProduction을 같은 INumber(제품코드)기준으로 합치기 위하여 JOIN한다

③ IName(제품명)을 출력하기 위하여 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져온다.
INumber(제품코드)를 출력하기 위하여 제품테이블인 tItem에서 제품코드 컬럼인 INumber를 가져온다.
PCount(생산량)을 출력하기 위하여 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져온다.
합을 구하기 위해 SUM함수를 사용한다.

④ tOrder(주문)테이블에 있는 ONumber(주문코드)를 가져오기 위해 tProdunction(생산)테이블의 PNumber(생산코드)와 tOrder(주문)테이블의 PNumber(생산코드)를 JOIN한다.
문제에 주어진 조건인 2022년 1월 이전 데이터만 가져온다.

⑤ tReturn(반품)테이블에 있는 RNumber(반품코드)를 가져오기 위해 tOrder(주문)테이블의 ONumber(주문코드) 와 tReturn(반품)테이블의 ONumber(주문코드)를 JOIN한다.

⑥ tReturnReason(반품사유)테이블에 있는 RReason(반품사유)를 가져오기 위해 tReturn(반품) 테이블의 RRNumber(반품사유코드)와 tReturnReason(반품사유)테이블에 있는 RRNumber(반품사유코드)를 JOIN하고 문제에 주어진 조건인 '불량'을 제외한 값만 가져온다.

⑦ tItem(제품)테이블에 있는 IName(제품명)을 가져오기 위해 tItem(제품)테이블의 INumber(제품코드)와 tProduction(생산)테이블의 INumber(제품코드)를 JOIN한다.

⑧ 문제에 제시된 조건인 팔리지 않거나 불량이 아닌 사유로 반품이 된 물품들을 찾기위해 ONumber(주문코드) 가 NULL 이거나 RRNumber(반품사유코드)가 NULL이 아닌 값들을 가져온다.

⑨ IName(물품명)과 INumber(물품코드)을 GROUP BY를 사용하여 그룹화 한다.
그룹화 한 IName(물품명), INumber(물품코드)을 기준으로 ③에서 SUM함수를 사용하여 재고량을 구한다

⑩ 서브쿼리 tBase(별칭)에 있는 IName(물품명)을 GROUP BY를 사용하여 그룹화 한다.
그룹화 한 IName(물품명)을 기준으로 ①에서 SUM함수를 사용하여 고객선호도를 구한다

⑪ 정렬 기준인 고객선호도를 오름차 순 정렬을 하기 위해 ORDER BY 를 사용한다

<함수설명>

SUM(tre.RCount)

- RCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

AVG(tBase.PCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더하고 개수만큼 나눠 평균을 리턴한다

CAST(AVG(tBase.PCount) AS FLOAT)

- NUMERIC타입인 AVG(tBase.PCount)를 FLOAT 타입으로 형 변환한다.

TO_CHAR(tor.ODate,'YYYYMM')

- 첫번째 파라미터인 ODate는 Timestamp타입이고, 두번째 파라미터인 'YYYYMM' 포맷에 맞춰 '년도월'의 텍스트타입으로 리턴한다.

ROUND(CAST(CAST(AVG(tBase.PCount) AS FLOAT) / CAST(SUM(tpr.PCount) AS FLOAT) AS DECIMAL),2)

- tBase.PCount 의 모든 데이터의 평균 값과 tpr.PCount의 모든 데이터를 합친 값을 FLOAT타입으로 변환하고 나눈 후 DECIMAL타입으로 변환하고 소수점 2자리까지 반올림한다

심화 7.

부서별로 생산량 대비 반품률을 구한 후 반품률이 가장 큰 부서 순으로 랭크를 매겨 출력하세요

(공동순위가 있을 경우 다음 순위는 공동순위의 수 만큼 밀려나며 반품률의 소숫점 2자리이후는 반올림한 후 출력되어야 한다.)

<정답>

<결과 화면>

	부서이름 character varying(10)	반품률 numeric	반품률_순위 bigint
1	문구생산부	20.94	1
2	악세사리생산부	17.77	2
3	음료생산부	9.95	3
4	가구생산부	5.38	4
5	전자기기생산부	4.01	5

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT tde.DName AS 부서명,                                          ----------①
    ROUND(CAST(CAST(SUM(tre.RCount) AS FLOAT)/
CAST(SUM(tpr.PCount) AS FLOAT) * 100 AS DECIMAL),2) AS 반품률,
    RANK() OVER(ORDER BY ROUND(CAST(CAST(SUM(tre.RCount) AS FLOAT)/
CAST(SUM(tpr.PCount) AS FLOAT) * 100 AS DECIMAL),2) DESC) AS 반품률_순위-----②
FROM tOrder AS tor
JOIN tProduction AS tpr                                              ------③
ON tpr.PNumber = tor.PNumber
LEFT OUTER JOIN tReturn AS tre                                       ------④
ON tre.ONumber = tor.ONumber
JOIN tItem AS tit                                                    ------⑤
ON tit.INumber = tpr.INumber
JOIN tEmployee AS tem                                                ------⑥
ON tem.ENumber = tpr.ENumber
JOIN tDepartment AS tde                                              ------⑦
ON tde.DNumber = tem.DNumber
GROUP BY tde.DName----------------------⑧
```

<해설>

① DName(부서명)을 출력하기 위해 부서테이블인 tDepartment에서 부서명 컬럼인 DName을 가져오고 반품률을 출력하기 위해 필요한 RCount(반품량)과 PCount(생산량)을 출력하기 위해 반품테이블인 tReturn과 생산테이블인 tProduction에서 각각 반품량 컬럼인 RCount와 생산량 컬럼인 PCount를 가져온 후 RCount(반품량) / PCount(생산량) * 100으로 반품률을 계산하였다.

② 문제의 조건에 맞는 결과인 반품률이 가장 큰 부서를 출력하되 동순위가 두 명이상 일 수 있기에 RANK 함수를 사용하여 순위를 매겨 1위인 부서들을 뽑을 수 있는 조건을 만들었고 ORDER BY에 존재하는 반품률을 기준으로 하여 순위를 출력해준다.

③ tProduction(생산) 테이블에 있는 PCount(생산량)을 가져오기 위해 tProduction(제품) 테이블의 PNumber(생산코드)와 tOrder(주문)테이블의 PNumber(생산코드)를 JOIN한다.

④ tReturn(반품) 테이블을 통해서 RCount(반품량) 데이터를 가져오기 위하여 tReturn(반품) 테이블의 ONumber(주문코드)와 tOrder(주문) 테이블의 ONumber(주문코드)를 이용하여 JOIN하되 주문하지 않은 고객정보까지 출력해주기 위하여 LEFT OUTER JOIN을 사용한다.

⑤ tItem(제품) 테이블을 통해서 INumber(제품코드) 데이터를 가져오기 위하여 tItem(제품) 테이블의 INumber(제품코드)와 tProduction(생산) 테이블의 INumber(제품코드)를 JOIN한다.

⑥ tEmployee(직원) 테이블에 있는 EName(직원명)을 가져오기 위해 tEmployee(직원) 테이블의 ENumber (직원코드)와 tProduction(생산) 테이블의 ENumber(직원코드)를 JOIN한다

⑦ tDepartment(부서) 테이블을 통해서 DName(부서명) 데이터를 가져오기 위하여 tDepartment(부서) 테이블의 DNumber(부서코드)와 tEmployee(직원) 테이블의 DNumber(부서코드)를 JOIN한다.

⑧ 그룹화 한 DName(부서명)을 기준으로 SUM함수를 사용하여 ①, ②에서 반품률과 반품률 순위를 구한다

RCount(반품량)과 PCount(생산량)을 나눌 때 CAST를 통한 형 변환을 하지 않으면 계산이 안 되거나 소수점 자리까지 제대로 출력이 되지 않는다..

<함수설명>

ROUND(CAST(CAST(SUM(tre.RCount) AS FLOAT)/CAST(SUM(tpr.PCount) AS FLOAT) * 100 AS DECIMAL),2)

- RCount의 모든 데이터를 합친 값과 PCount의 모든 데이터를 합친 값을 FLOAT타입으로 변환하고 100을 곱한 후 DECIMAL타입으로 변환하고 소수점 2자리까지 반올림한다

CAST(SUM(tre.RCount) AS FLOAT)

- NUMERIC 타입인 SUM(tre.RCount)를 FLOAT 타입으로 형 변환한다.

SUM (tre.RCount)

- RCount 의 데이터 타입이 숫자일 때 NULL 인 값을 제외한 모든 값을 더해 리턴한다.

심화 8.

현재 근무하는 전체 직원들의 연령대 별로 판매금액을 출력하시오

(현재 연도는 2022년이며 연령대는 20대, 30대 등으로 구분하며 반품된 물품들은 사유 및 반품 개수와 상관없이 해당물품은 제외한다. 따라서, 판매금액에서도 제외된다.)

<정답>

<결과 화면>

	연령대 text	연령대_별_판매금액 bigint
1	40	1791573000
2	50	729189000
3	20	3516671000
4	30	2891781800

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT   SUBSTRING(CAST(122 - CAST(SUBSTRING(tem.ERRN,1,2)      ---------- ①
         AS INTEGER) AS VARCHAR),1,1)||'0' AS 연령대
         , SUM(tpr.PCount * tit.Price) AS 연령대_별_판매금액
FROM tOrder AS tor
LEFT OUTER JOIN tReturn AS tre          ---------- ②
ON tor.ONumber = tre.ONumber
JOIN tProduction AS tpr                 ---------- ③
ON tpr.PNumber = tor.PNumber
JOIN tEmployee AS tem                   ---------- ④
ON tem.ENumber = tpr.ENumber
JOIN tItem AS tit                       ---------- ⑤
ON tit.INumber = tpr.INumber
WHERE tre.RNumber IS NULL   ---------- ⑥
GROUP BY SUBSTRING(CAST(122 - CAST(SUBSTRING(tem.ERRN,1,2) AS INTEGER) AS VARCHAR),1,1)||'0' ------ ⑦
```

<해설>

① ERRN(주민번호)를 출력하기 위해 직원테이블인 tEmployee에서 주민번호 컬럼인 ERRN을 가져오고 연령대가 필요하므로 2022년 기준으로 122에서 탄생년도 만큼 뺄셈을 한다. ERRRN 컬럼에 SUBSTRING을 사용하여 1번째 인덱스부터 2개를 잘라 탄생년도를 추출한다. 해당 년도를 INTEGER타입으로 형변환을 하고 122에서 뺄셈을 실시해 나온 결과값 앞자리만 가져와 0을 붙이면 연령대를 알 수 있다.

② tReturn(반품)테이블에 있는 RNumber(반품코드)를 가져오기 위해 tOrder(주문)테이블의 ONumber(주문코드)와 tReturn(반품)테이블의 ONumber(주문코드)를 JOIN한다.

③ tProduction(생산)테이블에 있는 ENumber(직원코드)를 가져오기 위해 tOrder(주문)테이블의 PNumber(생산코드)와 tProduction(생산)테이블에 있는 PNumber(생산코드)를 JOIN한다.

④ tEmployee(직원)테이블에 있는 ERRN(주민번호)를 가져오기 위해 tProduction(생산)테이블의 ENumber(직원코드)와 tEmployee(직원)테이블에 있는 ENumber(직원코드)를 JOIN한다.

⑤ tItem(제품)테이블에 있는 Price(단가)를 가져오기 위해 tProduction(생산)테이블의 INumber(제품코드)와 tItem(제품)테이블에 있는 INumber(제품코드)를 JOIN한다.

⑥ 문제에 주어진 조건인 환불되지 않은 제품을 구하기 위해 RNumber가 NULL인 값을 찾는다.

⑦ 연령대를 GROUP BY를 사용하여 그룹화 한다.
그룹화한 연령대를 기준으로 ①에서 SUM함수를 사용하여 연령대 별 판매금액을 구한다.

<함수설명>

SUBSTRING(tem.ERRN,1,2)

- 첫번째 파라미터는 문자열이고 두번째 파라미터는 시작 인덱스입니다. 세번째 파라미터는 가져 올 개수입니다. ERRN에서 1번째부터 2개까지 문자를 가져옵니다.

CAST(SUBSTRING(tem.ERRN,1,2) AS INTEGER)

- TEXT 타입인 SUBSTRING(tem.ERRN,1,2)을 INTEGER타입으로 형 변환한다.

SUM(tpr.PCount * tit.Price)

- (tpr.PCount * tit.Price) 의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

심화 9.

2020년 1/4분기에 생산된 모든 제품의 생산량과 생산량 대비 판매비율, 생산량 대비 반품비율을 구하시오. 단, 주문비율과 반품비율은 소수점 둘째자리에서 반올림 하고 null값이 있다면 0으로 대체 하시오.

(1/4분기는 1월~3월)

<정답>

<결과 화면>

	제품이름 character varying(20)	생산량 bigint	판매량 bigint	반품량 bigint	판매비율 numeric	반품비율 numeric
1	가위	1640	1640	0	100.00	0.00
2	공책	1023	700	700	68.43	100.00
3	귀걸이	1463	923	0	63.09	0.00
4	노트북	755	755	11	100.00	1.46
5	디지털카메라	163	93	9	57.06	9.68

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT IName AS 제품명                                                                    -----①
    , SUM(PCount) AS 생산량
    , SUM(OCount) AS 판매량
    , SUM(RCount) AS 반품량
    , ROUND(CAST(COALESCE(CAST(SUM(OCount) AS FLOAT)/NULLIF(CAST(SUM(PCount) AS FLOAT),0),0)
      AS DECIMAL)*100,2) AS 판매비율
    , ROUND(CAST(COALESCE(CAST(SUM(RCount) AS FLOAT)/NULLIF(CAST(SUM(OCount) AS FLOAT),0),0)
      AS DECIMAL)*100,2) AS 반품비율
FROM
(
    SELECT tit.IName                                                                      -----②
        , tpr.PCount AS PCount
        , CASE WHEN tor.ONumber IS NULL then 0 ELSE tpr.PCount END AS OCount
        , CASE WHEN tre.RCount IS NULL then 0 ELSE tre.RCount END AS RCount
    FROM tProduction AS tpr
    JOIN tItem AS tit                                                                     -----③
    ON tpr.INumber = tit.INumber
    LEFT OUTER JOIN tOrder AS tor                                                         -----④
    ON tor.PNumber = tpr.PNumber
    LEFT OUTER JOIN tReturn AS tre                                                        -----⑤
    ON tre.ONumber = tor.ONumber
    WHERE tpr.PDate BETWEEN CAST('20200101' AS TIMESTAMP) AND CAST('20200401' AS TIMESTAMP) -----⑥
) AS tBase
GROUP BY IName ------⑦
ORDER BY IName ------⑧
```

<해설>

① IName(제품명), PCount(생산량), OCount(판매량), RCount(반품량)는 tBase(별칭) 서브쿼리에서 결과물을 출력한 INumber(제품코드), PCount(생산량), OCount(판매량), RCount(반품량) 컬럼을 가져오고 판매비율 컬럼과 반품비율 컬럼의 경우 각각 (판매량) / (생산량), (반품량) / (판매량)을 하되 NULLIF 함수를 사용하여 값이 0이 출력될 때, 즉 생산됐으나 판매되지 않거나 판매됐으나 반품되지 않은 제품이 있을 경우는 NULL로 한 후 COALESCE함수를 사용하여 다시 0으로 바꿔주었고 제대로 값이 나온다면 마지막으로 ROUND 함수를 사용하여 소수점 2자리 이하는 반올림해주었다.

② IName(제품명)을 출력하기 위해 제품테이블인 tItem에서 제품명 컬럼인 IName을 가져오고 PCount(생산량)을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져오고 OCount(판매량)과 RCount(반품량)을 출력하기 위해 tProduction에서 생산량 컬럼인 PCount를 가져오되 주문테이블인 tOrder나 반품테이블인 tReturn의 값이 null일 때는 0으로 치환한다.

③ tItem(제품) 테이블을 통해서 INumber(제품코드) 데이터를 가져오기 위하여 tItem(제품) 테이블의 INumber(제품코드)와 tProduction(생산) 테이블의 INumber(제품코드)를 JOIN한다.

④ tOrder(주문) 테이블을 통해서 ONumber(주문코드) 데이터를 가져오기 위하여 tProduction(생산) 테이블의 PNumber(생산코드)를 이용하여 JOIN하되 주문되지 않은 생산정보까지 출력해주기 위하여 LEFT OUTER JOIN을 사용한다

⑤ tReturn(반품) 테이블을 통해서 RNumber(반품코드) 데이터를 가져오기 위하여 tReturn(반품) 테이블의 ONumber(주문코드)와 tOrder(주문) 테이블의 ONumber(주문코드)를 이용하여 JOIN하되 주문하지 않은 고객정보까지 출력해주기 위하여 LEFT OUTER JOIN을 사용한다.

⑥ 문제의 조건인 2020년 1월부터 2020년 3월의 값을 가져오기 위해 BETWEEN과 CAST함수를 사용하여 날짜가 20200101보다 크고 20200401보다 작은 값을 생산테이블인 tProduction에서 관련 컬럼인 PDate(생산날짜)와 비교한다.

⑦ 그룹화 한 IName(제품명)을 기준으로 ①에서 SUM함수를 사용하여 생산량, 판매량, 반품량, 판매비율, 반품비율을 구한다

⑧ 정렬 기준인 IName(제품명)을 오름차순정렬을 하기 위해 ORDER BY 를 사용한다

판매비율과 반품비율의 경우 (판매량) / (생산량), (반품량) / (판매량)으로 나누어주면 되지만 나눌 때 CAST를 통한 형 변환을 하지 않으면 계산이 안 되거나 소수점 자리까지 제대로 출력이 되지 않는다..

<함수설명>

SUM(PCount) , SUM(OCount) , SUM(RCount)

- PCount의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다

ROUND(CAST(COALESCE(CAST(SUM(OCount) AS FLOAT)/NULLIF(CAST(SUM(PCount) AS FLOAT),0),0) AS DECIMAL)*100, 2)

- OCount 의 모든 데이터를 합친 값과 PCount의 모든 데이터를 합친 값을 FLOAT타입으로 변환하고 나눈 후 나온 값을 DECIMAL타입으로 변환하고 100을 곱한 후 소수점 2자리까지 반올림한다

COALESCE(CAST(SUM(RCount) AS FLOAT)/NULLIF(CAST(SUM(OCount) AS FLOAT),0)

- COALESCE 내부의 (RCount / OCount) 값이 NULL 일 때 0으로 대체한다.

CAST(SUM(OCount) AS FLOAT)

- NUMERIC타입인 SUM(OCount)를 FLOAT 타입으로 형 변환한다.

NULLIF(CAST(SUM(OCount) AS FLOAT),0)

- CAST(SUM(OCount) AS FLOAT)값이 0이면 NULL, 아니면 1을 출력한다.

CAST('20200101' AS TIMESTAMP), CAST('20200401' AS TIMESTAMP)

- 텍스트타입인 '20200101'과 '20200401'을 TIMESTAMP타입으로 형 변환한다.

심화 10.

고객별로 판매금액이 제일 컸던 직원에게 판매금액의 12%의 인센티브를 지급하려한다.

고객별 판매금액이 제일 높은 직원의 직원명과 해당 직원의 인센티브를 고객명과 같이 출력하시오.

<정답>

<결과 화면>

	고객명 character varying(20)	직원명 character varying(5)	인센티브 numeric
1	강남악세사리	박은혜	25779600.00
2	그린가구	이은혜	125052000.00
3	남한전자상가	류승환	112488000.00
4	다있다전자상가	최미리	90698400.00
5	달달음료	김대한	1030320.00

* 결과 화면은 편의를 위해 최대 5줄까지로 제한합니다.

<정답 쿼리>

```
SELECT CName AS 고객명, EName AS 직원명, Incentive AS 인센티브 ---------- ①
FROM
(
    SELECT tcu.CName                                                                                    --②
        , tem.EName
        , SUM(tpr.PCount * tit.Price) * 0.12 AS Incentive
        , RANK() OVER(PARTITION BY tcu.CName ORDER BY SUM(tpr.PCount * tit.Price) * 0.12 DESC) AS seq
    FROM tOrder AS tor
    JOIN tCustomer AS tcu ---------- ③
    ON tor.CNumber = tcu.CNumber
    JOIN tProduction AS tpr ---------- ④
    ON tor.PNumber = tpr.PNumber
    JOIN tEmployee AS tem ---------- ⑤
    ON tpr.ENumber = tem.ENumber
    JOIN tItem AS tit ---------- ⑥
    ON tpr.INumber = tit.INumber
    GROUP BY tcu.CName, tem.EName ---------- ⑦
    ORDER BY 1 ---------- ⑧
) AS tBase
WHERE seq = 1 ---------- ⑨
```

<해설>

① 서브쿼리 tBase(별칭)에서 CName(고객명), EName(직원명), Incentive(인센티브)를 가져와 출력한다.

② CName(고객명)을 출력하기 위해 고객테이블인 tCustomer에서 고객명 컬럼인 CName을 가져오고
EName(직원명)을 출력하기 위해 직원테이블인 tEmployee에서 직원명 컬럼인 EName을 가져오고
PCount(생산량)을 출력하기 위해 생산테이블인 tProduction에서 생산량 컬럼인 PCount를 가져오고
Price(단가)를 출력하기 위하여 제품테이블인 tItem에서 단가 컬럼인 Price를 가져온다
생산량과 단가를 곱한 값들을 모두 SUM함수를 사용해 더하고 문제에서 제시한 12%(0.12)를 곱한다.
문제의 조건에 맞는 결과인 가장 많이 판매된 물품만을 출력하되 동순위가 두 명이상 일수 있기에 RANK 함수를 사용하여 순위를 매겨 1위인 제품들을 뽑을 수 있는 조건을 만들었고
PARTITION BY에 고객이름 컬럼인 CName을 사용함으로써 각 고객별 판매금액을 출력해주었으며 ORDER BY에 존재하는 SUM(tpr.PCount * tit.Price) * 0.12 DESC)을 기준으로 하여 순위를 출력하되 판매금액이 큰 순으로 출력해주기 위해 DESC(내림차 순)으로 순위를 출력해준다.

③ tCustomer(고객)테이블에 있는 CName(고객명)을 가져오기 위해 tOrder(주문)테이블의 CNumber(고객코드)와 tCustomer(고객)테이블에 있는 CNumber(고객코드)를 JOIN한다.

④ tProduction(생산)테이블에 있는 PNumber(생산코드)를 가져오기 위해 tOrder(주문)테이블의 PNumber(생산코드)와 tProduction(생산)테이블에 있는 PNumber(생산코드)를 JOIN한다.

⑤ tEmployee(직원)테이블에 있는 EName(직원명)을 가져오기 위해 tProduction(생산)테이블의 ENumber(직원코드)와 tEmployee(직원)테이블에 있는 ENumber(직원코드)를 JOIN한다.

⑥ tItem(제품)테이블에 있는 Price(단가)를 가져오기 위해 tProduction(생산)테이블의 INumber(제품코드)와 tItem(제품)테이블에 있는 INumber(제품코드)를 JOIN한다.

⑦ CName(고객명), EName(직원명)을 GROUP BY를 사용하여 그룹화 한다.
그룹화 한 CName(고객명), EName(직원명)을 기준으로 ②에서 SUM함수를 사용하여 인센티브를 구한다

⑧ ORDER BY 1의 의미는 제일 앞의 컬럼을 뜻한다. 그리고 이 상황에서는 고객명 기준으로 오름차순 정렬을 한다.

⑨ 문제에 주어진 조건인 고객명 별로 판매 순위가 1등인 직원만 출력한다.

<함수설명>

SUM(tpr.PCount * tit.Price)

- (tpr.PCount * tit.Price) 의 데이터 타입이 숫자일 때 NULL인 값을 제외한 모든 값을 더해 리턴한다