공통 조건

주문정보의 테스트주문 (TEST_YN : Y) 제외
레스토랑정보에 없는 레스토랑은 주문이 일어날 수 없으며,
레스토랑정보에 영업시작일 이후의 주문건만 필요 (시작일포함)
예외정보에 들어간 예외시작일 ~ 예외종료일에 해당하는 레스토랑 주문건은 제외
예외정보 테이블은 예외가 발생하는 레스토랑에 대한 정보만 존재.

JOIN 은 ANSI SQL
Scala Subquery 사용금지
수단방법 상관 없음 (SQL 함수, 하드코딩 상관 없음)

Q1. 2021-01-01 ~ 2021-01-10 기간의
일별 주문레스토랑수, 주문성공레스토랑수, 주문실패레스토랑수, 주문성공 주문금액, 주문실패 주문금액 과
전체기간 주문레스토랑수, 주문성공레스토랑수, 주문실패레스토랑수, 주문성공 주문금액, 주문실패 주문금액을 구하시오
레스토랑수는 Unique 수, 정렬은 주문일기준 오름차순, Alias명은 자유, 전체기간의 주문일은 공란 (null)으로 설정
성공주문이 없는 레스토랑은 결과에서 제외
(TABLE : ORDER_INFO, REST_INFO, EXCP_INFO)

Result Sample

주문일	주문 레스토랑 수	성공주문 레스토랑 수	실패주문 레스토랑 수	성공주문 주문금액	실패주문 주문금액
2021-01-01	6	5	2	100000	50000
2021-01-02	7	5	3	150000	100000
2021-01-03	4	4	0	150000	0
'	'	'	'	'	'
'	'	'	'	'	'
2021-01-09	10	8	4	210000	80000
2021-01-10	7	3	7	200000	100000
~~NULL~~	15	9	7	1000000	800000

A1.

```
SELECT
        ORD.ORDER_DATE
        , COUNT(DISTINCT REST.REST_ID)
        , COUNT(CASE WHEN ORDER_RESULT = 'SUCS' THEN REST.REST_ID ELSE NULL END)
        , COUNT(CASE WHEN ORDER_RESULT = 'FAIL' THEN REST.REST_ID ELSE NULL END)
        , SUM(CASE WHEN ORDER_RESULT = 'SUCS' THEN ORD.ORDER_AMT ELSE NULL END)
        , SUM(CASE WHEN ORDER_RESULT = 'FAIL' THEN ORD.ORDER_AMT ELSE NULL END)
FROM ORDER_INFO ORD
        INNER JOIN REST_INFO REST ON ORD.REST_ID = REST .REST_ID --레스토랑정보에 없는 레스토랑은 주문이 일어날 수 없으며,
        LEFT JOIN EXCP_INFO EXCP ON REST.REST_ID = EXCP .REST_ID
WHERE 1=1
        AND ORDER_DATE BETWEEN TO_DATE('2021-01-01') AND TO_DATE('2021-01-10')
        --레스토랑정보에 영업시작일 이후의 주문건만 필요 (시작일포함)
        AND ORD.ORDER_DATE >= REST.START_DATE
        --예외정보에 들어간 예외시작일 ~ 예외종료일에 해당하는 레스토랑 주문건은 제외
```

```
        AND ORD.ORDER_DATE < EXCP.START_DATE
        AND ORD.ORDER_DATE > EXCP.END_DATE
GROUP BY ORD.ORDER_DATE
ORDER BY ORD.ORDER_DATE

UNIONA LL

SELECT
        NULL AS ORDER_DATE
        , COUNT(DISTINCT REST.REST_ID)
        , COUNT(CASE WHEN ORDER_RESULT = 'SUCS' THEN REST.REST_ID ELSE NULL END)
        , COUNT(CASE WHEN ORDER_RESULT = 'FAIL' THEN REST.REST_ID ELSE NULL END)
        , SUM(CASE WHEN ORDER_RESULT = 'SUCS' THEN ORD.ORDER_AMT ELSE NULL END)
        , SUM(CASE WHEN ORDER_RESULT = 'FAIL' THEN ORD.ORDER_AMT ELSE NULL END)
FROM ORDER_INFO ORD
        INNER JOIN REST_INFO REST ON ORD.REST_ID = REST .REST_ID --레스토랑정보에 없는 레스토랑은 주문이
일어날 수 없으며,
        LEFT JOIN EXCP_INFO EXCP ON REST.REST_ID = EXCP .REST_ID
WHERE 1=1
        AND ORDER_DATE BETWEEN TO_DATE('2021-01-01') AND TO_DATE('2021-01-10')
        --레스토랑정보에 영업시작일 이후의 주문건만 필요 (시작일포함)
        AND ORD.ORDER_DATE >= REST.START_DATE
        --예외정보에 들어간 예외시작일 ~ 예외종료일에 해당하는 레스토랑 주문건은 제외
        AND ORD.ORDER_DATE < EXCP.START_DATE
        AND ORD.ORDER_DATE > EXCP.END_DATE
GROUP BY ORDER_DATE
```

Q2. 전체기간의

주문ID별, 주문시점 조리의 순번을 구하시오.
(배달이 시작되지 않은 주문에 대해서 조리중으로 판단하며, 24시간 영업점이라고 가정)
(TABLE : ORDER_INFO, REST_INFO, EXCP_INFO, DELIVERY_INFO)

Result Sample				
주문일	주문번호	레스토랑ID	레스토랑명	조리순번
2021-01-01	20210101001	100001	요기요-서초점	1
2021-01-01	20210101003	100001	요기요-서초점	1
2021-01-01	20210101004	100001	요기요-서초점	2

'	'	'	'	'
'	'	'	'	'
2021-01-09	20210109001	100002	요기요-교대점	8
2021-09-01	20210901001	100002	요기요-교대점	1

A2.

```
SELECT
        ORD.ORDER_DATE
        , ORD.ORDER_ID
        , REST_ID
        , REST_NAME
        , SUM OVER(CASE WHEN ) OVER PARTION(REST_ID)   AS 조리순번
FROM ORDER_INFO ORD
        INNER JOIN REST_INFO REST ON ORD.REST_ID = REST .REST_ID —레스토랑정보에 없는 레스토랑은 주문이 일어날 수 없으며,
        LEFT JOIN EXCP_INFO EXCP ON REST.REST_ID = EXCP .REST_ID
        LEFT JOIN DELIVERY_INFO DEL ON ORD.ORDER_ID = DEL.ORDER_ID

WHERE 1=1
        —레스토랑정보에 영업시작일 이후의 주문건만 필요 (시작일포함)
        AND ORD.ORDER_DATE >= REST.START_DATE
        —예외정보에 들어간 예외시작일 ~ 예외종료일에 해당하는 레스토랑 주문건은 제외
        AND ORD.ORDER_DATE < EXCP.START_DATE
        AND ORD.ORDER_DATE > EXCP.END_DATE
GROUP BY ORD.ORDER_DATE
ORDER BY ORD.ORDER_DATE
;

SELECT
        SUM OVER(CASE WHEN ) OVER PARTION(REST_ID)
FROM ORDER_INFO ORD
        LEFT JOIN DELIVERY_INFO DEL ON ORD.ORDER_ID = DEL.ORDER_ID
```

<u>공통 조건</u>

주문정보의 테스트주문 (TEST_YN : Y) 제외
레스토랑정보에 없는 레스토랑은 주문이 일어날 수 없으며,
레스토랑정보에 영업시작일 이후의 주문건만 필요 (시작일포함)
예외정보에 들어간 예외시작일 ~ 예외종료일에 해당하는 레스토랑 주문건은 제외
예외정보 테이블은 예외가 발생하는 레스토랑에 대한 정보만 존재.

JOIN 은 ANSI SQL
Scala Subquery 사용금지
수단방법 상관 없음 (SQL 함수, 하드코딩 상관 없음)