# DM_funnel_session — 설계노트 (Version v1 scope)

**1,** 목적**(Why)**

세션 단위로 funnel 단계(view→click→add_to_cart→checkout→purchase)의 **도달 여부(reach), 발생 시각(first ts), 발생 횟수(count), 순서 정합(strict flags)**을 표준화한다.

이후 funnel 병목 분석(전환율, 단계별 drop-off, strict vs loose 비교)과 Activation stage 정의(초기 14d/30d) 쿼리의 베이스로 사용한다.

**2, Grain**

1 session_id = 1 row

user_id는 세션의 소유자이며, session_start_ts/session_date는 세션 기준 시간이다.

**3, Input tables & Join key**

- users - signup_date 기준일(세션이 가입 후 며칠인지 계산), key - user_id
- sessions - 세션 베이스(세션 시작 시각/유저 속성 포함), key - session_id *(user_id* 포함*)*
- events - funnel 이벤트(view/click/add_to_cart/checkout/purchase) 발생 정보, key - (user_id, session_id)
- 참고(스캔 최적화): bounds(min_signup, max_signup)로 sessions/events를 min_signup ≤ date < max_signup + 180일로 먼저 필터한 뒤 사용한다.
- Scope note: SQL에 promo 관련 컬럼이 포함돼 있으나, 이번 버전에서는 promo 스코프 제외하기로 결정 (문서/스토리에서 사용하지 않음).

4, Partition / Clustering *(bullet point* 스타일*)*

- **PARTITION BY :** session_date 일자/기간별 세션 퍼널 추이가 핵심(일/주/월 slice)
- **CLUSTER BY :** user_id**,** session_id 유저/세션 기준 필터, 조인(events_pivot) 효율을 고려

**5, Window** 정의

- 기준일**: signup_date** (users에서 가져옴)
- 세션 발생일 기준 파생

    - **days_since_signup** = DATE_DIFF(session_date, signup_date, DAY)
    - **is_in_14d/30d**: signup_date ≤ session_date < signup_date+14/30일

- 전처리**(global filter,** 스캔 절감 목적**)**
    - sessions/events: **min_signup ≤ date < max_signup + 180**일로 사전 필터

- 경계 규칙**:** 시작일 포함**,** 종료일 미포함

**6, Main Features and** 계산 로직

A. Session base fields

- session_start_ts(TIMESTAMP), session_date(DATE)

- 유저 속성: user_type, device (+ promo 컬럼은 scope-out)

B. Funnel reach flags (loose)

- has_view/has_click/has_add_to_cart/has_checkout/has_purchase

    - Logic: 해당 이벤트 first timestamp가 NULL이 아니면 TRUE

C. First timestamps (세션 내 첫 발생 시각)

- view_ts/click_ts/add_to_cart_ts/checkout_ts/purchase_ts

    - Logic: MIN(IF(event_type=..., event_ts, NULL))

D. Event counts (세션 내 발생 횟수)

- view_cnt/click_cnt/add_to_cart_cnt/checkout_cnt/purchase_cnt

    - Logic: COUNTIF(event_type=...)

E. purchase - order 정합용

- order_id

    - Logic: purchase 이벤트가 있으면 해당 order_id를 1개 보관(MAX(IF(event_type='purchase', order_id, NULL)))

F. Strict flags (순서 정합성)

- strict_view: view 존재

- strict_click: view & click 존재 + view_ts ≤ click_ts
- strict_add_to_cart: view→click→cart 존재 + timestamp 순서 유지
- strict_checkout: view→click→cart→checkout 존재 + timestamp 순서 유지
- strict_purchase: view→click→cart→checkout→purchase 존재 + timestamp 순서 유지

목적: 이벤트가 있긴 한데 순서가 꼬인 케이스"를 분리해서 funnel KPI를 더 보수적으로 볼 수 있게 함

**7. Sanity checks**

- PK 유일성: row 수 = distinct session_id 수
- 이벤트 피벗 정합성:
    - has_purchase = TRUE이면 purchase_cnt ≥ 1이어야 함
    - purchase_ts NOT NULL이면 order_id가 대부분 NOT NULL이어야 함(생성 정책 검증)

- strict 단조성:
    - strict_purchase ⇒ strict_checkout ⇒ strict_add_to_cart ⇒ strict_click ⇒ strict_view

- 날짜 범위: days_since_signup가 과도하게 음수/매우 큰 값이 없는지(가입일 이후 세션만 주로 있어야 정상)
- 조인 폭발 방지: events_pivot은 (user_id, session_id) 1행이어야 함(피벗 단계 rowcount 확인)

**8. DM**이 꼭 필요한지

- 필요 **-** 세션 단위 **funnel**을 **"loose/strict"**로 동시에 제공해 병목**/**전환율 분석의 베이스가 되고**, downstream KPI DM**에서 반복 계산을 줄인다**.**