Locality Data Architecture & Modeling Documentation

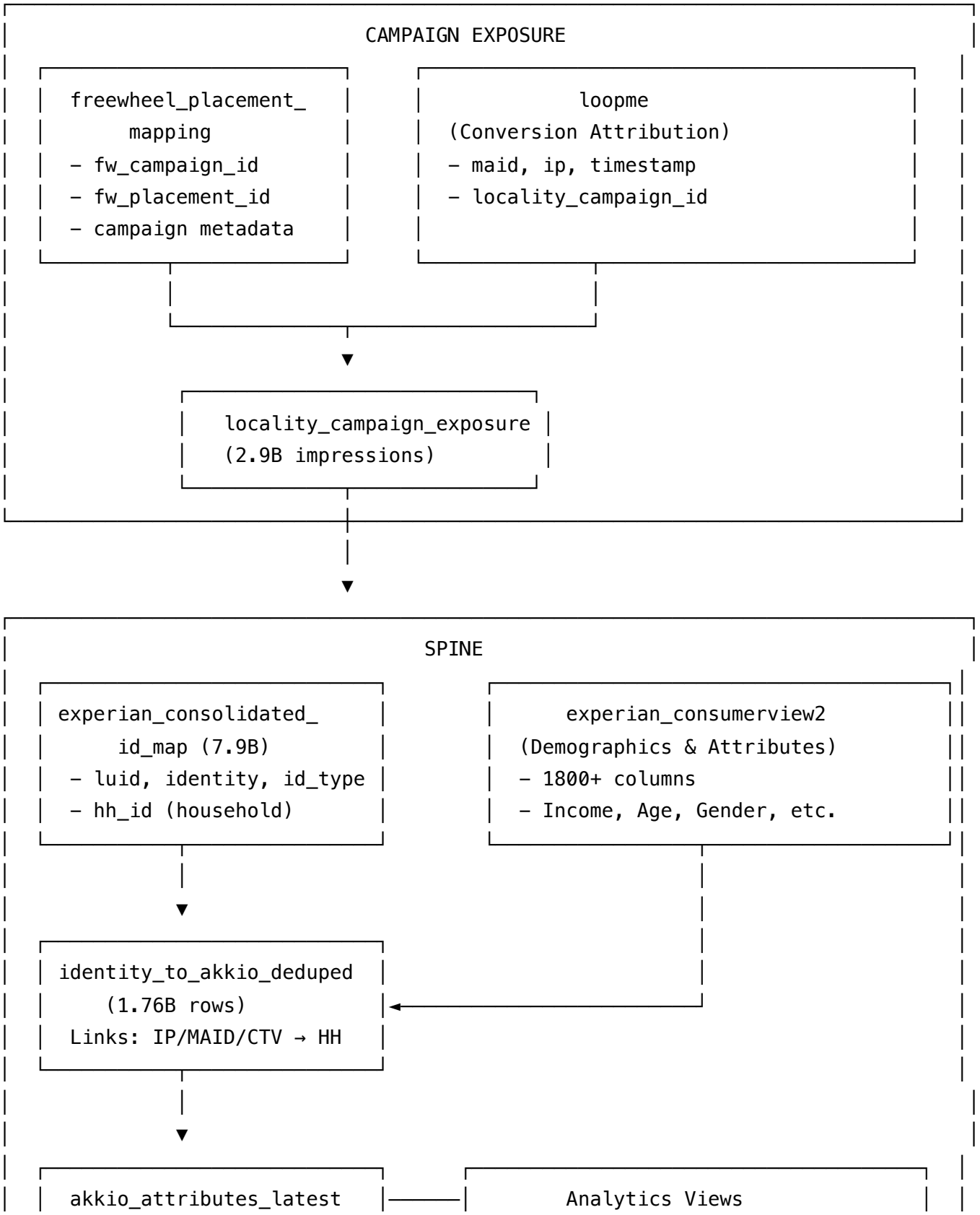
Executive Summary

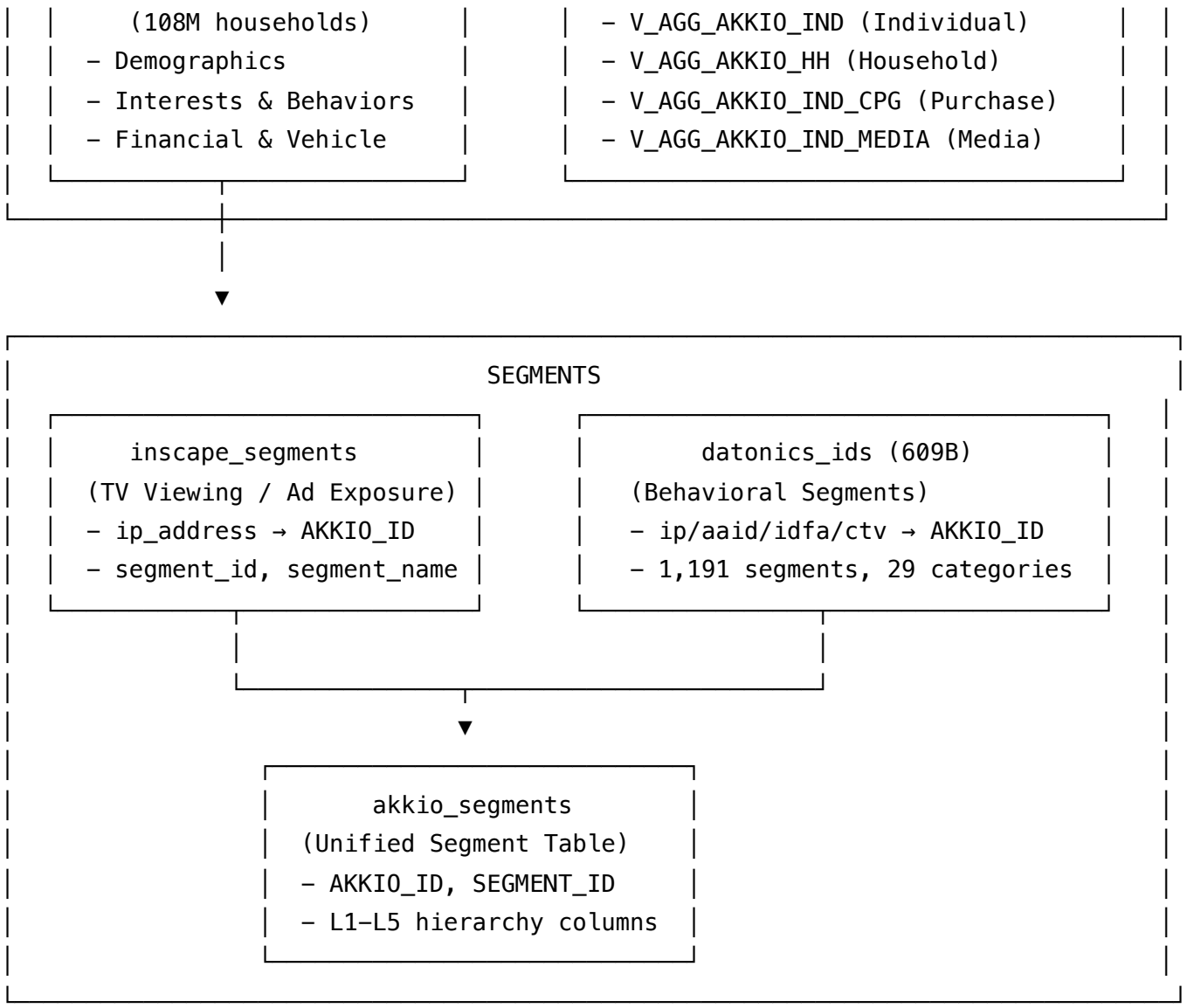
This document provides a comprehensive overview of the Locality data architecture, detailing how we unify multiple data sources to create a **household-centric** analytical foundation. The architecture enables:

- **Campaign Attribution**: Linking ad exposures to household profiles and conversions
- **Audience Enrichment**: Augmenting households with 100+ demographic, behavioral, and interest attributes
- **Segment Analysis**: Mapping households to both TV viewing patterns (Inscape) and behavioral segments (Datonics)
- **Analytics Compatibility**: Structured output tables optimized for the Insights analytics platform

1. Architecture Overview

1.1 Data Flow Diagram





1.2 Core Design Principles

Principle	Implementation
Household-Centric	All data resolves to `AKKIO_ID` (household identifier) as the primary key
Identity Resolution	Multi-device identity graph links IPs, MAIDs, IDFAs, CTVs to households
Hierarchical Segments	Datonics segments stored as L1-L5 columns for flexible querying
Incremental Processing	Campaign exposure uses incremental strategy for efficiency
Materialized for Performance	Critical join tables (identity_to_akkio_deduped) are materialized to avoid repeated scans

2. Data Sources

2.1 Source Systems Summary

Source	Description	Scale	Key Use
Experian Consolidated ID Map	Identity graph linking devices to households	7.9B rows	Identity resolution
Experian ConsumerView2	Demographic & behavioral attributes	288M rows, 1800+ cols	Attribute enrichment
FreeWheel Logs	Ad impression data from CTV campaigns	8.26B rows	Campaign exposure tracking
LoopMe	Conversion attribution data	7.4M rows	Conversion matching
Inscape Segments	TV viewing patterns	692M rows	Audience segmentation
Datonics IDs	Behavioral segment memberships	609B rows	Audience segmentation
Datonics Segments	Segment metadata (hierarchy definitions)	1,191 segments	Segment taxonomy

2.2 Source Schema Configuration

```yaml
sources:
  - name: locality_poc_share_silver
    tables:
      - experian_consolidated_id_map  # Identity graph
      - experian_consumerview2        # Demographics
      - datonics_ids                  # Segment memberships
      - datonics_segments             # Segment metadata
      - inscape_segments              # TV viewing segments
      - loopme                        # Conversion data
      - freewheel_placement_mapping   # Campaign metadata

  - name: locality_poc_share_gold
    tables:
      - freewheel_logs_gold           # Ad impressions
```

3. Identity Resolution Layer

3.1 The Identity Graph Approach

The foundation of the architecture is a **household-centric identity graph** that maps multiple device identifiers to a single household ID (`AKKIO_ID`).

Identity Types Supported:

- `ip` — IP addresses
- `aaid` — Android Advertising IDs
- `idfa` — iOS Identifier for Advertisers
- `ctv` — Connected TV device IDs

3.2 Identity-to-Akkio Mapping (`identity_to_akkio_deduped`)

This is a **critical materialized table** that serves as the join bridge between raw data sources and the household spine.

```sql
-- identity_to_akkio_deduped.sql
SELECT DISTINCT
    e_map.identity AS IDENTITY,
    e_map.id_type AS ID_TYPE,
    attr.AKKIO_ID,
    attr.AKKIO_HH_ID
FROM experian_consolidated_id_map e_map
INNER JOIN akkio_attributes_latest attr
    ON e_map.hh_id = attr.AKKIO_ID
WHERE e_map.id_type IN ('ip', 'ctv', 'idfa', 'aaid')
```

Why Materialized as a Table?

> This table is referenced 3+ times in downstream models. Materializing it avoids re-scanning 7.9B rows each time. Build time is ~5-10 minutes, producing 1.76B deduplicated rows.

Performance Optimization:

```sql
-- Clustering for efficient lookups
ALTER TABLE identity_to_akkio_deduped CLUSTER BY (IDENTITY, ID_TYPE)
```

4. Household Attribute Spine

4.1 Core Attributes Table (`akkio_attributes_latest`)

The household spine contains 108M households enriched with demographic, behavioral, and interest attributes from Experian ConsumerView2.

Key Design Decisions:

1. **One LUID per Household**: When multiple LUIDs exist for a household, we take `MIN(luid)` to ensure deterministic joins.
2. **Code-to-Value Mapping**: Experian data uses letter codes (A-K) that we decode to human-readable values and numeric midpoints.
3. **Multi-Value Aggregation**: Related attributes are concatenated into comma-separated strings for flexibility.

4.2 Attribute Categories

Demographics

```
-- Gender inference with fallback logic
CASE
    WHEN e_cv.PDM_Gender_Male = 'Y' AND COALESCE(e_cv.PDM_Gender_Female, '') != 'Y' THE
    WHEN e_cv.PDM_Gender_Female = 'Y' AND COALESCE(e_cv.PDM_Gender_Male, '') != 'Y' THE
    WHEN e_cv.Person_RC_gndr_gndr_2 = 'M' THEN 'M'
    WHEN e_cv.Person_RC_gndr_gndr_2 = 'F' THEN 'F'
    ELSE NULL
END AS GENDER,

-- Age with bucket mapping (1=18-24 through 7=75+)
CASE
    WHEN e_cv.Age_Range_1820 = 'Y' THEN 19
    WHEN e_cv.Age_Range_2529 = 'Y' THEN 27
    -- ... additional ranges
END AS AGE,

-- Ethnicity decoding
CASE COALESCE(e_cv.Person__RC_Ethnic_-_Group_1, e_cv.Person__RC_Ethnic_-_Group_2)
    WHEN 'A' THEN 'African American'
    WHEN 'O' THEN 'Hispanic'
    -- ... additional mappings
END AS ETHNICITY
```

Income & Financial

```
-- Income midpoint calculation from Experian letter codes
CASE e_cv.RC_Est_Household_Income_V6
    WHEN 'A' THEN 12500   -- $1K-25K
    WHEN 'B' THEN 37500   -- $25K-50K
    WHEN 'J' THEN 300000  -- $250K+
END AS INCOME,

-- Net worth from CFI score
CASE e_cv.CFINet_Asset_Score
    WHEN 'A' THEN 7500000   -- >$5M
    WHEN 'K' THEN 12500     -- <$25K
END AS NET_WORTH
```

Interests (Comma-Separated Aggregations)

```
-- General interests
CONCAT_WS(',',
    CASE WHEN e_cv.RC_ActInt_Fitness_Enthusiast = 'A' THEN 'Fitness' END,
    CASE WHEN e_cv.RC_ActInt_Gourmet_Cooking = 'A' THEN 'Gourmet Cooking' END,
    CASE WHEN e_cv.RC_ActInt_Wine_Lovers = 'A' THEN 'Wine' END
    -- ... 20+ interest flags
) AS GENERAL_INTERESTS,

-- Sports interests
CONCAT_WS(',',
    CASE WHEN e_cv.RC_ActIntNFL_Enthusiast = 'A' THEN 'NFL' END,
    CASE WHEN e_cv.RC_ActIntNBA_Enthusiast = 'A' THEN 'NBA' END,
    CASE WHEN e_cv.RC_ActIntPGA_Tour_Enthusiast = 'A' THEN 'Golf/PGA' END
    -- ... 17+ sports flags
) AS SPORTS_INTERESTS
```

Vehicle Ownership

```
-- Vehicle makes (supports multiple vehicles per household)
CONCAT_WS(',',
    CASE WHEN e_cv.Auto_Seg_Own_Tesla = 'Y' THEN 'Tesla' END,
    CASE WHEN e_cv.Auto_Seg_Own_BMW = 'Y' THEN 'BMW' END,
    CASE WHEN e_cv.Auto_Seg_Own_Toyota = 'Y' THEN 'Toyota' END
    -- ... 28 makes supported
) AS VEHICLE_MAKES,

-- Fuel type
CONCAT_WS(',',
    CASE WHEN e_cv.Own_Electric_Y = 'Y' THEN 'Electric' END,
    CASE WHEN e_cv.Own_Hybrid_Y = 'Y' THEN 'Hybrid' END
) AS FUEL_CODE
```

4.3 Full Attribute Schema

Category	Columns	Description
Identity	AKKIO_ID, AKKIO_HH_ID, LUID	Household identifiers

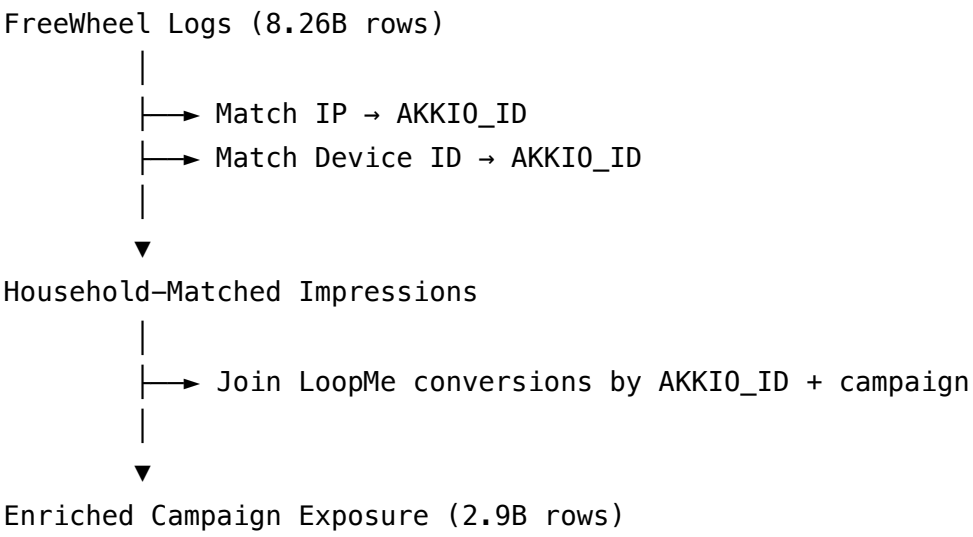
Category	Columns	Description
Demographics	GENDER, AGE, AGE_BUCKET, ETHNICITY, EDUCATION_LEVEL, MARITAL_STATUS	Individual characteristics
Geographic	STATE, ZIP11, COUNTY_NAME	Location data
Employment	OCCUPATION, OCCUPATION_TITLE	Professional information
Household	HOME_OWNERSHIP, NUM_PEOPLE_IN_HOUSEHOLD_GROUP, HOME_VALUE_RANGE	Household composition
Children	PRESENCE_OF_CHILDREN, NUMBER_OF_CHILDREN, CHILD_AGE_GROUP	Family composition
Income/Wealth	INCOME, INCOME_BUCKET, NET_WORTH, NET_WORTH_BUCKET	Financial indicators
Interests	GENERAL_INTERESTS, SPORTS_INTERESTS, READING_INTERESTS, TRAVEL_INTERESTS	Lifestyle preferences
Vehicles	VEHICLE_MAKES, VEHICLE_STYLES, VEHICLE_CLASS, FUEL_CODE	Auto ownership
Financial	FINANCIAL_HEALTH_BUCKET, CREDIT_CARD_INFO, INVESTMENT_TYPE	Financial behavior

5. Campaign Exposure Tracking

5.1 Campaign Exposure Model (`locality_campaign_exposure`)

This model links FreeWheel ad impressions to households and enriches them with LoopMe conversion data.

Processing Flow:



5.2 Identity Matching Logic

```sql
-- Priority: IP match, then device match
WITH freewheel_with_households AS (
    SELECT
        fw.*,
        COALESCE(ita_ip.AKKIO_ID, ita_device.AKKIO_ID) AS AKKIO_ID
    FROM freewheel_logs_gold fw
    LEFT JOIN identity_to_akkio_deduped ita_ip
        ON fw.ip_address = ita_ip.IDENTITY AND ita_ip.ID_TYPE = 'ip'
    LEFT JOIN identity_to_akkio_deduped ita_device
        ON fw.device_id = ita_device.IDENTITY
    WHERE COALESCE(ita_ip.AKKIO_ID, ita_device.AKKIO_ID) IS NOT NULL
)
```

5.3 Conversion Attribution

```
-- LoopMe conversion matching
loopme_conversions AS (
    SELECT DISTINCT
        ita.AKKIO_ID,
        l.locality_campaign_id,
        l.loopme_campaign_id
    FROM loopme l
    INNER JOIN identity_to_akkio_deduped ita
        ON (l.maid = ita.IDENTITY) OR (l.ip = ita.IDENTITY)
)

-- Enrichment with conversion flag
SELECT
    fw.AKKIO_ID,
    conv.loopme_campaign_id AS LOOPME_CAMPAIGN_ID,
    CASE WHEN conv.AKKIO_ID IS NOT NULL THEN TRUE ELSE FALSE END AS HAS_LOOPME_CONVERSI
    -- ... 70+ FreeWheel columns
FROM freewheel_with_households fw
LEFT JOIN loopme_conversions conv
    ON fw.AKKIO_ID = conv.AKKIO_ID
    AND fw.locality_campaign_id = conv.locality_campaign_id
```

5.4 Incremental Strategy

```
config:
    materialized: incremental
    unique_key: ['transaction_id', 'AKKIO_ID']
    incremental_strategy: merge
    partition_by: event_date
```

Incremental Filter:

```
{% if is_incremental() %}
    AND fw.event_date > (SELECT MAX(event_date) FROM {{ this }})
{% endif %}
```

5.5 Key Output Columns

Column	Description
`AKKIO_ID`	Matched household identifier
`HAS_LOOPME_CONVERSION`	Boolean: Did this household convert?
`LOOPME_CAMPAIGN_ID`	LoopMe campaign ID (if converted)
`TRANSACTION_ID`	Unique FreeWheel impression ID
`EVENT_DATE`	Date of ad exposure
`LOCALITY_CAMPAIGN_ID`	Locality campaign identifier
`LOCALITY_ADVERTISER`	Advertiser name
`LOCALITY_CAMPAIGN`	Campaign name
`IP_ADDRESS`	Viewer IP
`DEVICE_ID`	Viewer device ID
`REVENUE`	Impression revenue

6. Audience Segmentation

6.1 Segment Sources

The architecture unifies two distinct segment sources:

Source	Type	Scale	Identity Matching
Inscape	TV Viewing / Ad Exposure	692M rows	IP address only
Datonics	Behavioral & Demographic	609B rows	IP, AAID, IDFA, CTV

6.2 Datonics Hierarchical Segments

Datonics segments follow a hierarchical structure with up to 5 levels:

```
Automotive > Luxury > Owners > Recent Purchasers > New Vehicle
    L1          L2      L3           L4               L5
```

Categories (29 total): Automotive, B2B, CPG, Demographics, Education, Entertainment, Finance, Health, Home, Insurance, Lifestyle, Media, Parenting, Pets, Political, Retail, Sports, Technology, Telecommunications, Travel, and more.

6.3 Segment Metadata Processing

The `int_datonics_segments_metadata` table pre-computes segment hierarchy and parent-child relationships:

```sql
-- Extract hierarchy levels
SELECT
    segment,
    segment_name,
    GET(SPLIT(segment_name, ' > '), 0) as L1,
    GET(SPLIT(segment_name, ' > '), 1) as L2,
    GET(SPLIT(segment_name, ' > '), 2) as L3,
    GET(SPLIT(segment_name, ' > '), 3) as L4,
    GET(SPLIT(segment_name, ' > '), 4) as L5,
    SIZE(SPLIT(segment_name, ' > ')) as depth
FROM datonics_segments

-- Compute parent-child relationships
SELECT
    p.segment as parent_segment,
    c.segment as child_segment
FROM all_segments p
INNER JOIN all_segments c
    ON c.segment_name LIKE p.segment_name || ' > %'
    AND c.depth = p.depth + 1
```

6.4 Segment Processing at Scale

Processing 609B Datonics rows requires special consideration:

```sql
-- datonics_all_segments.sql
-- Process all categories in single pass with DISTINCT for deduplication

SELECT DISTINCT
    ita.AKKIO_ID,
    d.segment AS SEGMENT_ID,
    CONCAT_WS(', ', seg.L1, seg.L2, seg.L3, seg.L4, seg.L5) AS SEGMENT_NAME,
    seg.L1 AS SEGMENT_L1,
    seg.L2 AS SEGMENT_L2,
    seg.L3 AS SEGMENT_L3,
    seg.L4 AS SEGMENT_L4,
    seg.L5 AS SEGMENT_L5,
    seg.segment_description AS SEGMENT_DESCRIPTION,
    'datonics' AS SEGMENT_SOURCE
FROM datonics_ids d
INNER JOIN identity_to_akkio_deduped ita
    ON d.id = ita.IDENTITY AND d.id_type = ita.ID_TYPE
INNER JOIN int_datonics_segments_metadata seg
    ON d.segment = seg.segment
```

Key Optimizations:

- **DISTINCT at Join Time**: Prevents materializing 118B intermediate rows
- **ID Type Matching**: Joins on both identity value AND type for accuracy
- **Metadata Pre-computation**: Segment hierarchy computed once, reused across all queries

6.5 Unified Segments Table (`akkio_segments`)

The final segments table unions Inscape and Datonics:

```sql
-- akkio_segments.sql
WITH inscape_segments AS (
    SELECT DISTINCT
        ita.AKKIO_ID,
        i_seg.segment_id AS SEGMENT_ID,
        normalize_segment_name(i_seg.segment_name) AS SEGMENT_NAME,
        NULL AS SEGMENT_L1, -- Inscape uses flat structure
        -- ...
        'inscape' AS SEGMENT_SOURCE
    FROM inscape_segments i_seg
    INNER JOIN identity_to_akkio_deduped ita
        ON i_seg.ip_address = ita.IDENTITY AND ita.ID_TYPE = 'ip'
),

datonics_segments AS (
    SELECT * FROM datonics_all_segments
)

SELECT * FROM inscape_segments
UNION ALL
SELECT * FROM datonics_segments
```

6.6 Segment Schema

Column	Type	Description
AKKIO_ID	STRING	Household identifier
SEGMENT_ID	STRING	Unique segment identifier
SEGMENT_NAME	STRING	Human-readable segment name
SEGMENT_L1	STRING	Category (Datonics only)
SEGMENT_L2	STRING	Subcategory level 2
SEGMENT_L3	STRING	Subcategory level 3
SEGMENT_L4	STRING	Subcategory level 4
SEGMENT_L5	STRING	Subcategory level 5
SEGMENT_DESCRIPTION	STRING	Segment description

Column	Type	Description
SEGMENT_SOURCE	STRING	'inscape' or 'datonics'

6.7 Querying Segments

```sql
-- Find all households in Automotive category
SELECT DISTINCT AKKIO_ID
FROM akkio_segments
WHERE SEGMENT_L1 = 'Automotive';

-- Find luxury auto owners (any subcategory)
SELECT DISTINCT AKKIO_ID
FROM akkio_segments
WHERE SEGMENT_L1 = 'Automotive' AND SEGMENT_L2 = 'Luxury';

-- Find households in both auto and travel luxury
SELECT AKKIO_ID
FROM akkio_segments
WHERE SEGMENT_L1 = 'Automotive' AND SEGMENT_L2 = 'Luxury'
INTERSECT
SELECT AKKIO_ID
FROM akkio_segments
WHERE SEGMENT_L1 = 'Travel' AND SEGMENT_L2 = 'Luxury';
```

7. Analytics Output Tables

7.1 Table Overview

The architecture produces four analytics-ready tables optimized for the Insights platform:

Table	Grain	Purpose	Key Attributes
V_AGG_AKKIO_IND	Individual (AKKIO_ID)	Core demographics & interests	Gender, Age, Income, Interests
V_AGG_AKKIO_HH	Household (AKKIO_HH_ID)	Household composition	Children, Home Value, Ownership

Table	Grain	Purpose	Key Attributes
V_AGG_AKKIO_IND_CPG	Individual	Purchase behavior	Categories, Spend Levels, Channels
V_AGG_AKKIO_IND_MEDIA	Individual	Media consumption	Streaming, Devices, Genres

7.2 Individual Aggregation (V_AGG_AKKIO_IND)

```
SELECT
    attr.AKKIO_ID,
    attr.AKKIO_HH_ID,
    1.0 AS WEIGHT,  -- For analytics platform

    -- Demographics with NULL handling
    COALESCE(attr.GENDER, 'UNDETERMINED') AS GENDER,
    attr.AGE,
    attr.AGE_BUCKET,
    COALESCE(attr.ETHNICITY, 'Unknown') AS ETHNICITY_PREDICTION,
    COALESCE(attr.EDUCATION_LEVEL, 'Unknown') AS EDUCATION,
    COALESCE(attr.MARITAL_STATUS, 'Unknown') AS MARITAL_STATUS,

    -- Home ownership as numeric
    CASE attr.HOME_OWNERSHIP
        WHEN 'Homeowner' THEN 1
        WHEN 'Renter' THEN 0
        ELSE NULL
    END AS HOMEOWNER,

    -- All interest and vehicle columns
    attr.GENERAL_INTERESTS,
    attr.SPORTS_INTERESTS,
    attr.VEHICLE_MAKES AS MAKE,
    -- ...

    attr.PARTITION_DATE
FROM akkio_attributes_latest attr
```

7.3 Household Aggregation (`V_AGG_AKKIO_HH`)

```sql
SELECT
    attr.AKKIO_HH_ID,
    1.0 AS WEIGHT,
    1.0 AS HH_WEIGHT,  -- Backwards compatibility

    -- Household-specific attributes
    CASE attr.HOME_OWNERSHIP
        WHEN 'Homeowner' THEN 1
        WHEN 'Renter' THEN 0
        ELSE NULL
    END AS HOMEOWNER,
    attr.INCOME,
    attr.INCOME_BUCKET,

    -- Children data
    attr.CHILD_AGE_GROUP,
    attr.NUMBER_OF_CHILDREN,
    attr.PRESENCE_OF_CHILDREN,

    -- Housing
    attr.NUM_PEOPLE_IN_HOUSEHOLD_GROUP,
    attr.HOME_VALUE_RANGE AS MEDIAN_HOME_VALUE_BY_STATE,

    attr.PARTITION_DATE
FROM akkio_attributes_latest attr
```

7.4 CPG Purchase Behavior (`V_AGG_AKKIO_IND_CPG`)

```sql
SELECT
    attr.AKKIO_ID,
    1.0 AS WEIGHT,

    -- Transaction categories
    CONCAT_WS(',',
        CASE WHEN e_cv.TRX_Apparel_Spenders = 'Y' THEN 'Apparel' END,
        CASE WHEN e_cv.TRX_Pets_and_Animals_High_Spenders = 'Y' THEN 'Pets (High)' END,
        -- ... 25+ categories
    ) AS CATEGORIES_PURCHASED,

    -- Spending levels by category
    CONCAT_WS(',',
        CONCAT('Clothing:', e_cv.ConsumerSpend_Clothing),
        CONCAT('Dining:', e_cv.ConsumerSpend_Dining_Out),
        -- ...
    ) AS PURCHASE_BUCKETS,

    -- Discretionary spend estimates
    e_cv.RC_DSE_Discretionary_Spend_Estimate AS TOTAL_DISCRETIONARY_SPEND,
    e_cv.RC_DSE_Apparel AS DSE_APPAREL,
    e_cv.RC_DSE_Entertainment AS DSE_ENTERTAINMENT,

    -- Shopping behavior
    CONCAT_WS(',',
        CASE WHEN e_cv.TRX_InStore_Transactors = 'Y' THEN 'In-Store' END,
        CASE WHEN e_cv.TRX_Online_Transactors = 'Y' THEN 'Online' END
    ) AS SHOPPING_CHANNELS,

    attr.PARTITION_DATE
FROM akkio_attributes_latest attr
LEFT JOIN experian_consumerview2 e_cv ON attr.LUID = e_cv.recd_luid
```

7.5 Media Consumption (`V_AGG_AKKIO_IND_MEDIA`)

```sql
SELECT
    attr.AKKIO_ID,
    1.0 AS WEIGHT,

    -- Streaming services
    CONCAT_WS(',',
        CASE WHEN e_cv.rc_rs_video_brand_netflix = 'Y' THEN 'Netflix' END,
        CASE WHEN e_cv.rc_rs_video_brand_hulu = 'Y' THEN 'Hulu' END,
        CASE WHEN e_cv.rc_rs_audio_brand_spotify = 'Y' THEN 'Spotify' END,
        -- ...
    ) AS APP_SERVICES_USED,

    -- TV providers
    CONCAT_WS(',',
        CASE WHEN e_cv.rc_rs_tv_brand_directv = 'Y' THEN 'DirecTV' END,
        CASE WHEN e_cv.rc_rs_tv_brand_spectrum = 'Y' THEN 'Spectrum' END,
        -- ...
    ) AS NETWORKS_WATCHED,

    -- Devices
    CONCAT_WS(',',
        CASE WHEN e_cv.RC_CompElect_Apple_iPhone_ = 'A' THEN 'iPhone' END,
        CASE WHEN e_cv.tv_brand_samsung = 'Y' THEN 'Samsung TV' END,
        -- ...
    ) AS INPUT_DEVICES_USED,

    -- Viewing behavior
    CONCAT_WS(',',
        CASE WHEN e_cv.tv_ad_avoider = 'Y' THEN 'Ad Avoider' END,
        CASE WHEN e_cv.tv_ad_acceptor = 'Y' THEN 'Ad Acceptor' END
    ) AS AD_BEHAVIORS,

    CASE WHEN e_cv.RC_OBM_cordcuttersV1 = 'Y' THEN 1 ELSE 0 END AS IS_CORD_CUTTER,

    attr.PARTITION_DATE
FROM akkio_attributes_latest attr
LEFT JOIN experian_consumerview2 e_cv ON attr.LUID = e_cv.recd_luid
```

8. Performance Considerations

8.1 Materialization Strategy

Model	Materialization	Reason
`akkio_attributes_latest`	TABLE	Core spine, accessed by all downstream models
`identity_to_akkio_deduped`	TABLE	Critical join table, avoids 7.9B row re-scans
`int_datonics_segments_metadata`	TABLE	Small metadata table (1,191 rows)
`locality_campaign_exposure`	INCREMENTAL	2.9B rows, only process new data
`akkio_segments`	INCREMENTAL	Large table with append-only pattern
`V_AGG_*` tables	TABLE	Analytics outputs, need fast query performance

8.2 Clustering Strategy

Tables are clustered to optimize query patterns:

```
-- Household lookups
ALTER TABLE akkio_attributes_latest CLUSTER BY (AKKIO_ID);

-- Identity resolution
ALTER TABLE identity_to_akkio_deduped CLUSTER BY (IDENTITY, ID_TYPE);

-- Segment queries
ALTER TABLE akkio_segments CLUSTER BY (AKKIO_ID, SEGMENT_SOURCE);

-- Time-based analytics
ALTER TABLE V_AGG_AKKIO_IND CLUSTER BY (PARTITION_DATE, AKKIO_ID);
```

8.3 Scale Metrics

Model	Rows	Build Time	Notes
`akkio_attributes_latest`	108M	~1 min	Full refresh
`identity_to_akkio_deduped`	1.76B	5-10 min	Full refresh
`locality_campaign_exposure`	2.9B	Incremental	Daily append
`akkio_segments` (Inscape)	~410M	Variable	IP-only matching
`akkio_segments` (Datonics)	~60B	Hours	609B source, deduplicated

9. dbt Macros

9.1 Segment Name Normalization

```
{% macro normalize_segment_name(column_name) %}
    regexp_replace(
        regexp_replace(
            regexp_replace(
                regexp_replace(
                    lower({{ column_name }}),
                    ' ', '_'
                ),
                '&', 'and'
            ),
            '\\s*\\/\\s*', '/'
        ),
        '\\s*-\\s*', '-'
    )
{% endmacro %}
```

Transformations:

- Lowercase all characters
- Replace spaces with hyphens
- Replace `&` with `and`
- Normalize spacing around `/` and `-`

9.2 Datonics Category Processing

The `process_datonics_category` macro provides an alternative approach for per-category processing with leaf-filtering (used for debugging/testing):

```
{% macro process_datonics_category(category_name) %}
-- Process by id_type for efficiency
{% for id_type in ['ip', 'aaid', 'idfa', 'ctv'] %}
    -- Filter to category segments for this id_type
    -- Apply leaf filtering (keep deepest segment only)
    -- Join to identity graph
{% endfor %}
{% endmacro %}
```

10. Data Quality & Testing

10.1 dbt Tests

```yaml
models:
  - name: akkio_attributes_latest
    tests:
      - dbt_utils.unique_combination_of_columns:
          combination_of_columns: [AKKIO_ID]
    columns:
      - name: AKKIO_ID
        tests: [unique, not_null]
      - name: INCOME
        tests:
          - dbt_utils.accepted_range:
              min_value: 0
              max_value: 5000000

  - name: locality_campaign_exposure
    tests:
      - dbt_utils.unique_combination_of_columns:
          combination_of_columns: [TRANSACTION_ID, AKKIO_ID]
    columns:
      - name: AKKIO_ID
        tests:
          - relationships:
              to: ref('akkio_attributes_latest')
              field: AKKIO_ID

  - name: akkio_segments
    columns:
      - name: SEGMENT_SOURCE
        tests:
          - accepted_values:
              values: ['inscape', 'datonics']
```

10.2 Referential Integrity

All campaign exposures and segments must link to valid households:

```yaml
- name: AKKIO_ID
  tests:
    - relationships:
        to: ref('akkio_attributes_latest')
        field: AKKIO_ID
```

11. Appendix

A. Model Dependency Graph

```
akkio_attributes_latest
    │
    ├──► identity_to_akkio_deduped
    │        │
    │        ├──► akkio_segments (Inscape)
    │        ├──► akkio_segments (Datonics via datonics_all_segments)
    │        └──► locality_campaign_exposure
    │
    ├──► V_AGG_AKKIO_IND
    ├──► V_AGG_AKKIO_HH
    ├──► V_AGG_AKKIO_IND_CPG
    ├──► V_AGG_AKKIO_IND_MEDIA
    └──► V_AKKIO_ATTRIBUTES_LATEST (view)
```

B. File Structure

```
models/
├── intermediate/
│   ├── int_datonics_categories.sql
│   └── int_datonics_segments_metadata.sql
│
└── locality/
    ├── akkio_attributes_latest.sql      # Core household spine
    ├── akkio_segments.sql               # Unified segments
    ├── identity_to_akkio_deduped.sql    # Identity graph bridge
    ├── locality_campaign_exposure.sql   # Campaign impressions
    │
    ├── v_agg_akkio_hh.sql               # Household analytics
    ├── v_agg_akkio_ind.sql              # Individual analytics
    ├── v_agg_akkio_ind_cpg.sql          # CPG analytics
    ├── v_agg_akkio_ind_media.sql        # Media analytics
    ├── v_akkio_attributes_latest.sql    # Backwards-compatible view
    │
    └── datonics_segments/
        ├── datonics_all_segments.sql    # All category processing
        └── README.md                     # Processing documentation
```

C. Experian Code Mappings Reference

Income Ranges (RC_Est_Household_Income_V6):

Code	Range	Midpoint
A	$1K-25K	$12,500
B	$25K-50K	$37,500
C	$50K-75K	$62,500
D	$75K-100K	$87,500
E	$100K-125K	$112,500
F	$125K-150K	$137,500
G	$150K-175K	$162,500
H	$175K-200K	$187,500

Code	Range	Midpoint
I	$200K-250K	$225,000
J	$250K+	$300,000

Net Worth (CFINet_Asset_Score):

Code	Range
A	>$5M
B	$2.5-5M
C	$1-2.5M
D	$750K-1M
E	$500-750K
F	$250-500K
G	$100-250K
H	$75-100K
I	$50-75K
J	$25-50K
K	<$25K

Ethnicity (Person__RC_Ethnic_-_Group):

Code	Ethnicity
A	African American
B	Southeast Asian
C	South Asian
D	Central Asian
E	Mediterranean
F	Native American

Code	Ethnicity
G	Scandinavian
H	Polynesian
I	Middle Eastern
J	Jewish
K	Western European
L	Eastern European
M	Caribbean Non-Hispanic
N	East Asian
O	Hispanic

Document Version: 1.0
Last Updated: December 2024
dbt Project: locality_poc_databricks