DATA INGESTION AND CPM LOGIC SPEC DOC

Paid Media Performance Dashboard (V1)

1. Purpose

This document defines exactly how:

- CSV files are pulled from S3
- Data is validated and loaded into the database
- Client specific CPM values are applied
- Spend based metrics are calculated

This is the playbook for any developer building or maintaining the ingestion and calculation pipeline.

2. Ingestion Schedule and Triggers

2.1 Daily Ingestion

- Frequency: Once per day
- Recommended time: 3:00 to 4:00 AM Eastern
- Trigger: Cron job or scheduled Lambda invocation

Daily job responsibilities

1. Connect to Stephen's S3 bucket
2. Locate the latest CSV file for the previous day
3. Download and parse the file
4. Validate structure and data types
5. Write raw records into staging tables
6. Apply CPM and derived metric calculations
7. Promote data into production tables
8. Log success or failure in `ingestion_logs`
9. Send alert to Stephen on failure

2.2 Weekly and Monthly Summary Jobs

- Weekly job: Runs Monday at 5:00 AM
- Monthly job: Runs on the 1st of each month at 5:00 AM

Both read from the database, not directly from S3.

3. S3 Integration Details

3.1 S3 Credentials (provided by Stephen)

- Bucket name
- File path or folder prefix
- Access key ID
- Secret access key
- Optional region

These are stored securely in environment variables or secrets manager.

3.2 File Naming Convention

To keep ingestion predictable, we recommend one of these formats:

- `media-report-YYYY-MM-DD.csv`
- `client-media-YYYYMMDD.csv`

If the partner has an existing naming pattern, we will:

1. Document it
2. Use a prefix plus most recent timestamp rule to select the correct file

3.3 File Discovery Logic

Daily ingestion should:

1. List files in the S3 folder for the target date
2. Choose the file with:
    - Matching date
    - Or the latest timestamp for that date
3. If no file is found:
    - Log missing file
    - Send alert to Stephen

4. CSV Structure and Parsing

4.1 Expected Columns

From the media partner:

Dimensions

- `date`
- `campaign_name`
- `strategy_name`
- `placement_name`
- `creative_name`

Metrics

- `impressions`
- `clicks`
- `ctr`
- `conversions`
- `conversion_revenue`

If the partner provides additional metrics (such as default CPM or spend), they can be stored but the system will still recalculate using client CPM.

4.2 Parsing Rules

1. All numeric fields are parsed as numbers
2. Date field parsed into a standard `YYYY-MM-DD` format
3. Empty or null values default to:
    - Zero for numeric metrics
    - Blank string for text fields

4.3 Validation Rules

The ingestion engine must check:

- All required columns exist
- Row count greater than zero

- Date values are valid and within a reasonable range
- No negative values for impressions, clicks, conversions or revenue

If validation fails:

- Do not load data into production tables
- Record failure in `ingestion_logs`
- Send error email to Stephen

5. Data Staging and Mapping

We use a two step approach:

1. Load CSV rows into a staging table
2. Transform and map staging data into normalized tables

5.1 Staging Table Example

`staging_media_raw` might contain:

- `id`
- `client_id` (if known at ingest time or via mapping rule)
- `date`
- `campaign_name`
- `strategy_name`
- `placement_name`
- `creative_name`
- `impressions`
- `clicks`
- `ctr`
- `conversions`
- `conversion_revenue`
- `ingestion_run_id`

5.2 Normalized Mapping

After staging:

1. Resolve or create related entities
    - Find or create `campaign` by client + campaign_name
    - Find or create `strategy` by campaign + strategy_name
    - Find or create `placement`
    - Find or create `creative`
2. Write final metrics into a `daily_metrics` table keyed by:
    - `client_id`
    - `date`
    - `campaign_id`
    - `strategy_id`
    - `placement_id`
    - `creative_id`

6. Client CPM Configuration

6.1 Storage

Each client has a record in `client_settings` with:

- `client_id`
- `cpm` (numeric, example: 15.00)
- `currency` (optional, example: USD)
- `effective_date` (optional, if CPM can change over time)

6.2 Admin UI

Stephen can:

- Set CPM when creating a client
- Update CPM values later

When CPM changes, future daily calculations will use the new CPM. Historical data can either:

- Remain calculated with the CPM that was active when ingested
- Or be recalculated with the new CPM if we implement a recalculation job

By default the safer approach is to lock historical CPM per date range. We can extend to recalculation later if desired.

7. CPM Based Calculations

The core formula:

> Spend = (Impressions / 1,000) × Client CPM

All other spend based metrics derive from this.

Let:

- `I` = impressions
- `C` = clicks
- `V` = conversions
- `R` = conversion_revenue
- `CPM` = client CPM

7.1 Spend

```
Spend = (I / 1000) × CPM
```

If impressions are zero, spend is zero.

7.2 CTR

If not already provided, calculate:

```
CTR = C / I
```

Guardrail:

- If `I` is zero, set `CTR = 0`.

7.3 CPC

```
CPC = Spend / C
```

Guardrail:

- If `C` is zero, set `CPC = 0` or `NULL` depending on design choice

- We recommend zero for display and avoid division by zero

7.4 CPA

```
CPA = Spend / V
```

Guardrail:

- If `V` is zero, set `CPA = 0` or `NULL`
- We recommend zero for display but can signal "no conversions" via UI state

7.5 ROAS (if enabled)

```
ROAS = R / Spend
```

Guardrail:

- If `Spend` is zero, set `ROAS = 0`

7.6 Rounding and Precision

- Store raw values with at least 4 to 6 decimal places
- Display values rounded to:
    - 2 decimals for currency
    - 2 to 3 decimals for rates (CTR, ROAS)

8. Daily Ingestion Step By Step

1. **Start job**
    - Log start time and target date
2. **Connect to S3**
    - Authenticate with access key ID and secret
    - Navigate to configured path

3. **Locate file**
    - Use naming pattern or last modified time for target date
4. **Download file**
    - Save in temporary storage or stream directly
5. **Basic structural validation**
    - Confirm header row contains all expected columns
6. **Row level parsing and validation**
    - Parse each line
    - Convert to proper types
    - Handle missing or malformed values
7. **Insert into staging table**
    - Insert batch records into `staging_media_raw`
    - Record staging success count
8. **Transform and normalize**
    - Resolve campaigns, strategies, placements, creatives
    - Map each row to IDs
    - Fetch client CPM from `client_settings`
    - Apply CPM based calculations
    - Insert into `daily_metrics`
9. **Mark ingestion success**
    - Write final record into `ingestion_logs` with status = success
10. **On failure**
    - Mark status = failed with error message
    - Send alert to Stephen

9. Weekly and Monthly Summary Logic

9.1 Weekly Summary

Inputs:

- `daily_metrics` for the last full week
- Group by client

For each client:

- Sum impressions, clicks, conversions, revenue, spend
- Compute CTR, CPC, CPA, ROAS for the week
- Identify top campaigns and creatives by:
    - Conversions
    - Revenue

Optionally compare to the previous week and compute deltas.

Results are stored in `weekly_summaries`.

9.2 Monthly Summary

Similar to weekly but for the previous calendar month.

Stored in `monthly_summaries`.

10. Error Handling Rules

10.1 Missing File

If no CSV file is found for a scheduled run:

- Log error: `missing_file`
- Send email:
    - Subject: Missing daily media report
    - Body: Date, bucket, folder, and next steps

10.2 CSV Format Error

If required columns are missing:

- Log error: `invalid_format`
- Send email describing missing columns

10.3 Data Validation Error

If rows contain nonsensical data (for example, negative impressions):

- Skip those rows or mark them as rejected
- Include rejected count in error log
- Email summary to Stephen

10.4 Database Write Error

If database insert fails:

- Roll back current transaction
- Log the reason
- Alert Stephen

11. Reingestion and Backfill

There should be an internal or admin accessible method to:

- Re run ingestion for a specific date
- Backfill data for a historical date range

Typical flow:

1. Admin selects date range
2. System fetches relevant files from S3
3. Re runs ingestion and transformation
4. Either overwrites or appends data based on settings

12. Security Notes

- S3 credentials stored in environment variables or secrets manager
- Use least privilege IAM role for S3 read only access
- All ingestion traffic over HTTPS
- Strict CSV parsing to prevent injection or code execution
- Database credentials stored securely and rotated periodically