

# Data Model & Calculation Logic

---

This document explains how raw search analytics events are transformed into meaningful metrics. It covers event sequences, timing calculations, and business rules with clear examples.

---

## Table of Contents

1. Event Types & Sequence
  2. Processing Pipeline Overview
  3. Timing Calculations
  4. Business Rules & Classifications
  5. Output Files & Column Definitions
  6. Power BI Calculated Columns
  7. Power BI Measures
- 

## 1. Event Types & Sequence

### Initialization Events

Events fired when the search interface loads (before any user search):

Event	Description	When Fired
SEARCH_USER_LOGGED_IN_SUCCESS	User authenticated	User successfully authenticated in goto/echo
SEARCH_USER_DETAILS_FETCHED	User profile loaded	User details fetched from User Profile after authentication
SEARCH_USER_DETAILS_FETCHED_FROM_CACHE	User profile from cache	User details fetched from local storage after authentication
SEARCH_USER_PHOTO_FETCHED	Profile picture loaded	User profile pic retrieved from User Profile
SEARCH_DATA_FETCH_STARTED	Suggestions request sent	Request to fetch suggestions and trending searches sent to backend
SEARCH_DATA_FETCH_COMPLETED	Suggestions loaded	Suggestions and trending searches retrieved from backend

### Search Flow Events

Core events in the search execution flow (stored in the `name` column):

Event	Description	When Fired
SEARCH_TRIGGERED	User initiates search	User clicks search button OR presses Enter key
SEARCH_STARTED	Request sent to backend	Search request submitted to search service
SEARCH_COMPLETED	Results returned	Search results returned to user
SEARCH_RESULT_COUNT	Results displayed	Search completed and result count returned to user
SEARCH_FAILED	Search error	Any error occurred during search

### Click Events

Events fired when users interact with search results:

Event	Description	When Fired
SEARCH_TAB_CLICK	Tab clicked	Any tab (All, News, GOTO) is clicked
SEARCH_RESULT_CLICK	Result clicked (legacy)	Any item from search results is clicked
SEARCH_RESULT_CLICKED	Result clicked (new)	Any item from search results is clicked (new event name)
SEARCH_VIEW_MORE_LINK	View more clicked	User clicks "view more" link in results
SEARCH_ALL_TAB_PAGE_CLICK	All tab pagination	User on ALL tab clicks page in pagination
SEARCH_NEWS_TAB_PAGE_CLICK	News tab pagination	User on NEWS tab clicks page in pagination
SEARCH_GOTO_TAB_PAGE_CLICK	GoTo tab pagination	User on GOTO tab clicks page in pagination
SEARCH_PEOPLE_*	People result clicked	User clicks a People tab result
SEARCH_TRENDING_CLICKED	Trending item clicked	User clicks a trending search item
SEARCH_FILTER_CLICK	Filter clicked	Date OR Relevance filter clicked on results page

### Full Event Sequence

```

[Initialization - happens once per session]
|
v
SEARCH_USER_LOGGED_IN_SUCCESS
|
v
SEARCH_USER_DETAILS_FETCHED (or SEARCH_USER_DETAILS_FETCHED_FROM_CACHE)
|
v
SEARCH_USER_PHOTO_FETCHED
|
v
SEARCH_DATA_FETCH_STARTED
|
v
SEARCH_DATA_FETCH_COMPLETED
|
v
[User ready to search]
|
v
SEARCH_TRIGGERED <-- User presses Enter or clicks search (10:30:15.123)
|
v
SEARCH_STARTED <-- Request sent to backend (10:30:15.150)
|
v
SEARCH_COMPLETED <-- Results returned (10:30:15.400)
|
v
SEARCH_RESULT_COUNT <-- Results displayed to user (10:30:15.567)
|
v
[User interacts with results - independent events]
|
+-- SEARCH_TAB_CLICK / SEARCH_RESULT_CLICK / SEARCH_RESULT_CLICKED
+-- SEARCH_VIEW_MORE_LINK
+-- SEARCH_ALL_TAB_PAGE_CLICK / SEARCH_NEWS_TAB_PAGE_CLICK / SEARCH_GOTO_TAB_PAGE_CLICK
+-- SEARCH_TRENDING_CLICKED
+-- SEARCH_FILTER_CLICK

```

### Typical Search Sequence (Simplified)

```

User types "project budget" and presses Enter
|
v
[SEARCH_TRIGGERED] <-- timestamp: 10:30:15.123
|
v
[SEARCH_STARTED] <-- timestamp: 10:30:15.150 (27ms later)
|
v
[SEARCH_COMPLETED] <-- timestamp: 10:30:15.400 (250ms later)
|
v
[SEARCH_RESULT_COUNT] <-- timestamp: 10:30:15.567 (167ms later, 444ms total)
|
v
User sees results, clicks one
|
v
[SEARCH_TAB_CLICK] <-- timestamp: 10:30:18.890 (3.3s after results shown)

```

### Example: Complete Session

```

Session: 2025-01-15_user123_session456

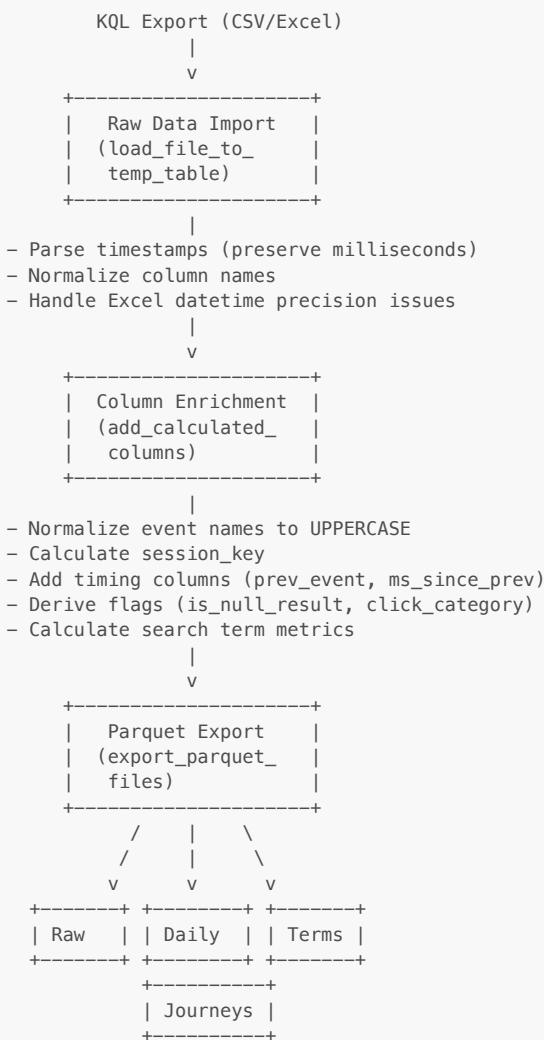
Event 1: SEARCH_TRIGGERED @ 10:30:15.123 (search term: "budget report")
Event 2: SEARCH_STARTED @ 10:30:15.150 (request to backend)
Event 3: SEARCH_COMPLETED @ 10:30:15.400 (results returned)
Event 4: SEARCH_RESULT_COUNT @ 10:30:15.567 (15 results displayed)
Event 5: SEARCH_TAB_CLICK @ 10:30:18.890 (user clicked a result)
Event 6: SEARCH_TRIGGERED @ 10:30:45.000 (user searches again: "2024 budget")
Event 7: SEARCH_STARTED @ 10:30:45.030
Event 8: SEARCH_COMPLETED @ 10:30:45.280

```

```
Event 9: SEARCH_RESULT_COUNT @ 10:30:45.400 (8 results displayed)
Event 10: SEARCH_TAB_CLICK @ 10:30:52.500 (user clicked another result)
```

## 2. Processing Pipeline Overview

### Data Flow



### Key Transformations

#### 1. Event Name Normalization

Raw event names come in mixed case from App Insights. We normalize to uppercase for consistent matching.

```
Input: "Search_completed"
Output: "SEARCH_COMPLETED"
```

#### 2. Session Key Generation

A unique session is identified by combining date + user + session ID:

```
session_key = session_date || '_' || user_id || '_' || session_id
-- Example: "2025-01-15_user123_abc789"
```

#### 3. CET Timezone Conversion

All time-derived columns use Central European Time (CET/CEST) instead of UTC:

```
-- DuckDB:  
timestamp_cet = timezone('Europe/Berlin', timestamp)  
  
-- PostgreSQL:  
timestamp_cet = timestamp AT TIME ZONE 'UTC' AT TIME ZONE 'Europe/Berlin'
```

This automatically handles:

- **CET (UTC+1)**: Standard time (late October to late March)
- **CEST (UTC+2)**: Daylight saving time (late March to late October)

**Columns derived from CET timestamp:**

- **session\_date**: Extracted from CET timestamp (affects session boundaries)
- **session\_key**: Uses CET-based session\_date
- **event\_hour**: Hour (0-23) in CET
- **event\_weekday**: Day name in CET
- **event\_weekday\_num**: ISO day of week in CET
- **searches\_morning/afternoon/evening/night**: Based on CET hour

**Example (Winter - CET):**

```
UTC timestamp: 2025-01-15 23:30:00.000 (late evening UTC)
CET timestamp: 2025-01-16 00:30:00.000 (early morning CET – next day!)
session_date: 2025-01-16 (CET date)
event_hour: 0 (midnight hour in CET)
```

**Example (Summer - CEST):**

```
UTC timestamp: 2025-07-15 22:30:00.000 (late evening UTC)
CEST timestamp: 2025-07-16 00:30:00.000 (early morning CEST – next day!)
session_date: 2025-07-16 (CEST date)
event_hour: 0 (midnight hour in CEST)
```

#### 4. Search Term Normalization

Search terms are cleaned for consistent aggregation:

```
-- Column names vary by event type and nesting level (resolved dynamically):  
-- New structure: CP_searchQuery_queryText (L3) or CP_queryText (L2 for SEARCH_TRIGGERED)  
-- Old structure: CP_searchQuery, searchQuery, query  
search_term_normalized = LOWER(TRIM(COALESCE(CP_searchQuery_queryText, CP_queryText, CP_searchQuery, searchQuery,  
query)))  
-- Input: " Budget Report "  
-- Output: "budget report"
```

### 3. Timing Calculations

**ms\_search\_to\_result** (User-Perceived Latency)

**What it measures:** The time from when a user initiates a search until they see results.

**Event span:** SEARCH\_TRIGGERED --> SEARCH\_RESULT\_COUNT

**How it's calculated:**

```
-- Step 1: Track the most recent SEARCH_TRIGGERED timestamp
last_search_started_ts = LAST_VALUE(
    CASE WHEN name = 'SEARCH_TRIGGERED' THEN timestamp END
    IGNORE NULLS
) OVER (PARTITION BY session_key ORDER BY timestamp)

-- Step 2: Calculate time difference when SEARCH_RESULT_COUNT occurs
ms_search_to_result = DATEDIFF('millisecond', last_search_started_ts, timestamp)
-- Only when name = 'SEARCH_RESULT_COUNT'
```

**Example:**

```

Event: SEARCH_TRIGGERED      @ 10:30:15.123
Event: SEARCH_COMPLETED      @ 10:30:15.234
Event: SEARCH_RESULT_COUNT @ 10:30:15.567

ms_search_to_result = 10:30:15.567 - 10:30:15.123 = 444ms

```

### ms\_result\_to\_click (Decision Time)

**What it measures:** How long the user takes to click a result after seeing search results.

**Event span:** `SEARCH_RESULT_COUNT` --> `Click Event`

**How it's calculated:**

```

ms_result_to_click = ms_since_prev_event
-- Only when click_category IS NOT NULL AND prev_event = 'SEARCH_RESULT_COUNT'

```

### Example:

```

Event: SEARCH_RESULT_COUNT @ 10:30:15.567
Event: SEARCH_TAB_CLICK      @ 10:30:18.890

ms_result_to_click = 10:30:18.890 - 10:30:15.567 = 3,323ms (3.3 seconds)

```

### ms\_since\_prev\_event (Inter-Event Timing)

**What it measures:** Time between any two consecutive events in a session.

```

ms_since_prev_event = DATEDIFF('millisecond',
    LAG(timestamp) OVER (PARTITION BY session_key ORDER BY timestamp),
    timestamp
)

```

### Example:

```

Event 1: SEARCH_TRIGGERED      @ 10:30:15.123 --> ms_since_prev = NULL (first event)
Event 2: SEARCH_COMPLETED      @ 10:30:15.234 --> ms_since_prev = 111ms
Event 3: SEARCH_RESULT_COUNT @ 10:30:15.567 --> ms_since_prev = 333ms
Event 4: SEARCH_TAB_CLICK      @ 10:30:18.890 --> ms_since_prev = 3,323ms

```

### Time Buckets

Timing values are bucketed for easier visualization:

Metric	Bucket	Range
search_to_result	< 0.5s	0-499ms
	0.5-1s	500-999ms
	1-2s	1000-1999ms
	2-5s	2000-4999ms
	> 5s	5000ms+
	No Result	NULL (no SEARCH_RESULT_COUNT event)
result_to_click	< 2s (quick)	0-1999ms
	2-5s	2000-4999ms
	5-10s	5000-9999ms
	10-30s	10000-29999ms
	30-60s	30000-59999ms
	> 60s (browsing)	60000ms+
	No Click	NULL (user didn't click)

## 4. Business Rules & Classifications

### is\_null\_result

**Definition:** The search returned zero results.

```
is_null_result = CASE
    WHEN name = 'SEARCH_RESULT_COUNT' AND CP_totalResultCount = 0 THEN true
    WHEN name = 'SEARCH_RESULT_COUNT' AND CP_totalResultCount > 0 THEN false
    ELSE NULL -- Only meaningful for SEARCH_RESULT_COUNT events
END
```

### Example:

Event: SEARCH\_RESULT\_COUNT with CP\_totalResultCount = 0  
--> is\_null\_result = true (user saw "No results found")

Event: SEARCH\_RESULT\_COUNT with CP\_totalResultCount = 15  
--> is\_null\_result = false (user saw 15 results)

### click\_category

**Definition:** Categorizes click events by type of interaction.

```
click_category = CASE
    WHEN name IN ('SEARCH_RESULT_CLICK', 'SEARCH_RESULT_CLICKED') THEN 'Result'
    WHEN name = 'SEARCH_VIEW_MORE_LINK' THEN 'ViewMore'
    WHEN name = 'SEARCH_TRENDING_CLICKED' THEN 'Trending'
    WHEN name = 'SEARCH_TAB_CLICK' THEN 'Tab'
    WHEN name = 'SEARCH_ALL_TAB_PAGE_CLICK' THEN 'Pagination_All'
    WHEN name = 'SEARCH_NEWS_TAB_PAGE_CLICK' THEN 'Pagination_News'
    WHEN name = 'SEARCH_GOTO_TAB_PAGE_CLICK' THEN 'Pagination_GoTo'
    WHEN name = 'SEARCH_FILTER_CLICK' THEN 'Filter'
    ELSE NULL -- Not a click event
END
```

### is\_success\_click

**Definition:** True only for clicks that indicate the user found content.

```
is_success_click = CASE
    WHEN name IN ('SEARCH_RESULT_CLICK', 'SEARCH_RESULT_CLICKED') THEN true
    ELSE false
END
```

**Note:** **SEARCH\_TRENDING\_CLICKED** is NOT a success click - it's a search initiation via suggestion. **SEARCH\_VIEW\_MORE\_LINK** is also NOT a success click - it's navigation to see more results, classified as **ViewMore** in **click\_category**.

### journey\_outcome (Session-Level)

**Definition:** Classifies how a search session ended based on user engagement level.

```
journey_outcome = CASE
    WHEN success_click_count > 0 THEN 'Success'
    WHEN click_count > 0 AND success_click_count = 0 THEN 'Engaged'
    WHEN result_count > 0 AND null_result_count = result_count THEN 'No Results'
    WHEN result_count > 0 AND click_count = 0 THEN 'Abandoned'
    ELSE 'Unknown'
END
```

### Categories explained:

- **Success:** User clicked on an actual search result (found content)
- **Engaged:** User interacted with tabs, pagination, or filters but didn't click a result (browsed but didn't find)
- **No Results:** All search attempts returned 0 results
- **Abandoned:** Had results displayed but no interaction at all
- **Unknown:** Incomplete session data

**Note:** Uses **success\_click\_count** (SEARCH\_RESULT\_CLICK only) for Success, and **click\_count** (all clicks) for Engaged.

**Example scenarios:**

Scenario	success_click_count	click_count	result_count	null_result_count	Outcome
User searched, clicked a result	1	1	1	0	Success
User clicked tabs/pagination only	0	2	1	0	Engaged
User searched, got 0 results	0	0	1	1	No Results
User searched, saw results but didn't click	0	0	1	0	Abandoned
Incomplete session data	0	0	0	0	Unknown

**session\_complexity**

**Definition:** Categorizes sessions by number of **user actions** (searches + clicks).

This counts only user-initiated events:

- **Searches:** `SEARCH_TRIGGERED` events (user pressed Enter or clicked search)
- **Clicks:** All click events (`SEARCH_RESULT_CLICK`, `SEARCH_TAB_CLICK`, `SEARCH_TRENDING_CLICKED`, pagination clicks, `SEARCH_FILTER_CLICK`)

It excludes backend telemetry events like `SEARCH_STARTED`, `SEARCH_COMPLETED`, `SEARCH_RESULT_COUNT` which inflate counts without representing user engagement.

```
user_actions = search_count_in_session + click_count

session_complexity = CASE
    WHEN user_actions = 1 THEN 'Single Action'
    WHEN user_actions <= 3 THEN 'Simple'
    WHEN user_actions <= 10 THEN 'Medium'
    ELSE 'Complex'
END
```

Complexity	User Actions	Typical Scenario
Single Action	1	Quick search, no click
Simple	2-3	Search + click, or 2 searches
Medium	4-10	Multiple searches and/or clicks
Complex	>10	Extended research session

**had\_reformulation**

**Definition:** Did the user refine/change their search query within the session?

```
had_reformulation = CASE
    WHEN unique_search_terms > 1 THEN true
    ELSE false
END
```

**Example:**

```
Session with searches: "budget", "2024 budget", "budget report Q4"
--> unique_search_terms = 3
--> had_reformulation = true (user refined their search)
```

**recovered\_from\_null**

**Definition:** Did the user eventually find content (click on a result) despite getting zero results initially?

```
recovered_from_null = CASE
    WHEN null_result_count > 0 AND success_click_count > 0 THEN true
    ELSE false
END
```

**Note:** Uses `success_click_count` (`SEARCH_RESULT_CLICK` only), not `click_count`. Navigating tabs/pagination after a null result is not considered "recovery" - only clicking actual content counts.

**Example:**

```
Session: Search "bugdet" (typo) --> 0 results
  Search "budget" --> 15 results --> Click on result
--> null_result_count = 1, success_click_count = 1
--> recovered_from_null = true
```

ApplInsights Identifiers: user\_id and session\_id

The `user_id` and `session_id` values come from Azure Application Insights telemetry. Understanding their behavior is important for interpreting user cohort and session metrics.

#### **user\_id (Cookie-based)**

- ApplInsights uses a browser cookie to generate and persist the `user_id`
- The same user will have the same `user_id` across sessions as long as the cookie exists
- A new `user_id` will be generated if:
  - The user clears their cookies
  - The user switches to a different browser
  - The user uses incognito/private browsing mode
  - The cookie expires

**Implication for "Returning Users":** The `returning_users` metric may undercount actual returning users if they clear cookies or switch browsers. It may also overcount if multiple people share the same browser.

#### **session\_id (Activity-based)**

- A `session_id` persists for the duration of a user's active session
- A session is defined as a **period of activity separated by less than 30 minutes of inactivity**
- After **30 minutes of inactivity**, a new `session_id` is generated for the next activity
- Closing the browser typically ends the session (new session on return)

**Implication for Session Metrics:** Users who take long breaks (>30 min) during research will appear as multiple sessions. Quick tab-switching between searches will remain in the same session.

User Cohort: `is_users_first_session`

**Definition:** Is this the first time we've seen this user search?

```
user_session_number = ROW_NUMBER() OVER (
  PARTITION BY user_id
  ORDER BY session_start
)
is_users_first_session = CASE WHEN user_session_number = 1 THEN true ELSE false END
```

New vs Returning Users (Daily)

**Definition:** Count of users who are new vs returning on each day.

```
-- First, find when each user first appeared
first_seen_date = MIN(session_date) GROUP BY user_id

-- Then classify on each day
new_users = COUNT(DISTINCT CASE WHEN session_date = first_seen_date THEN user_id END)
returning_users = COUNT(DISTINCT CASE WHEN session_date > first_seen_date THEN user_id END)
```

## 5. Output Files & Column Definitions

`searches_raw.parquet`

**Granularity:** One row per event (click, search, result)

**Use case:** Detailed event-level analysis, debugging

Column	Type	Description	Example
<code>timestamp</code>	Timestamp	Event timestamp in UTC (microsecond precision)	2025-01-15 10:30:15.567123
<code>timestamp_cet</code>	Timestamp	Event timestamp in CET/CEST (microsecond precision)	2025-01-15 11:30:15.567123
<code>timestamp_cet_str</code>	String	CET timestamp as string for Power BI	2025-01-15 11:30:15.567
<code>name</code>	String	Event type (normalized to uppercase)	SEARCH_RESULT_COUNT

Column	Type	Description	Example
user_id	String	Anonymous user identifier	user_abc123
session_id	String	Session identifier	sess_xyz789
session_key	String	Composite key: date_user_session (CET date)	2025-01-15_user_abc123_sess_xyz789
session_date	Date	Date of the event (CET-based)	2025-01-15
event_order	Integer	Sequence number within session	3
prev_event	String	Previous event type in session	SEARCH_COMPLETED
ms_since_prev_event	Integer	Milliseconds since previous event	333
search_term_normalized	String	Cleaned search query	budget report
is_null_result	Boolean	True if zero results returned	false
click_category	String	Click type (Result/ViewMore/Trending/Tab/Pagination_*/Filter)	Result
is_success_click	Boolean	True for actual result clicks	true
last_search_started_ts	Timestamp	Most recent SEARCH_TRIGGERED timestamp	2025-01-15 10:30:15.123
clicked_position	Integer	Position of clicked result in result list	3
clicked_tab	String	Which tab was clicked	All
applied_filter	String	Which filter was applied	Date
clicked_result_title	String	Title of clicked result	Budget Report Q4
clicked_result_url	String	URL of clicked result	<a href="https://intranet/">https://intranet/...</a>
news_result_count	Integer	News results in result count	5
query_language	String	Detected query language (UPPER, or "Unknown")	EN
device_type	String	User's device type	Desktop
department	String	User's department	Finance
location	String	User's location	Berlin
job_title	String	User's job title	Analyst
search_latency	Double	Search latency in milliseconds	234.5

searches\_journeys.parquet

**Granularity:** One row per search session

**Use case:** Session-level behavior analysis, funnel metrics

Column	Type	Description	Calculation
session_date	Date	Date of session	
session_start	Timestamp	First event timestamp	MIN(timestamp)
session_start_str	String	Session start as string	STRFTIME for Power BI compatibility
total_events	Integer	Events in session	COUNT(*)
search_count_in_session	Integer	SEARCH_TRIGGERED events	COUNT(SEARCH_TRIGGERED)
result_count	Integer	SEARCH_RESULT_COUNT events	COUNT(SEARCH_RESULT_COUNT)
click_count	Integer	Click events	COUNT(click_category IS NOT NULL)
unique_search_terms	Integer	Distinct queries	COUNT(DISTINCT search_term)
null_result_count	Integer	Zero-result events	SUM(is_null_result)
max_total_results	Integer	Max results shown	MAX(CP_totalResultCount)
sec_search_to_result	Float	Seconds: search to results	MIN(ms_search_to_result) / 1000
sec_result_to_click	Float	Seconds: results to click	MIN(ms_result_to_click) / 1000
total_duration_sec	Float	Session length in seconds	(MAX - MIN timestamp) / 1000
first_event_hour	Integer	Hour of first event (0-23 CET)	MIN(event_hour)
last_event_hour	Integer	Hour of last event (0-23 CET)	MAX(event_hour)
result_clicks	Integer	SEARCH_RESULT_CLICK events	COUNT(click_category='Result')

Column	Type	Description	Calculation
trending_clicks	Integer	SEARCH_TRENDING_CLICKED events	COUNT(click_category='Trending')
tab_clicks	Integer	SEARCH_TAB_CLICK events	COUNT(click_category='Tab')
pagination_clicks	Integer	All pagination clicks	COUNT(click_category LIKE 'Pagination%')
pagination_all_clicks	Integer	All tab pagination	COUNT(click_category='Pagination_All')
pagination_news_clicks	Integer	News tab pagination	COUNT(click_category='Pagination_News')
pagination_goto_clicks	Integer	GoTo tab pagination	COUNT(click_category='Pagination_GoTo')
filter_clicks	Integer	SEARCH_FILTER_CLICK events	COUNT(click_category='Filter')
success_click_count	Integer	Success clicks (SEARCH_RESULT_CLICK only)	COUNT(is_success_click=true)
includes_first_search_of_day	Boolean	Session has day's first search	MAX(is_first_search_of_day)
search_to_result_bucket	String	Latency category	See Time Buckets
result_to_click_bucket	String	Decision time category	See Time Buckets
session_duration_bucket	String	Session length category	< 5s, 5-30s, 30-60s, 1-3 min, etc.
journey_outcome	String	Session result	Success/Engaged/Abandoned/No Results
had_reformulation	Boolean	User changed query	unique_search_terms > 1
session_complexity	String	Session size category	Based on user actions (searches + clicks)
search_to_result_sort	Integer	Sort order for latency bucket	1-6 for Power BI sorting
result_to_click_sort	Integer	Sort order for click time bucket	1-7 for Power BI sorting
session_duration_sort	Integer	Sort order for duration bucket	1-6 for Power BI sorting
journey_outcome_sort	Integer	Sort order for outcome	1=Success, 2=Engaged, 3=Abandoned, 4=No Results
session_complexity_sort	Integer	Sort order for complexity	1-4 for Power BI sorting
had_null_result	Boolean	Had zero-result search	null_result_count > 0
recovered_from_null	Boolean	Success despite null result	null_result > 0 AND success_click > 0
user_session_number	Integer	User's session sequence	ROW_NUMBER per user
is_users_first_session	Boolean	First time user	user_session_number = 1
distinct_click_categories	Integer	Tab types clicked	COUNT(DISTINCT click_category)
had_tab_switch	Boolean	Clicked multiple tabs	distinct_click_categories > 1
viewmore_clicks	Integer	SEARCH_VIEW_MORE_LINK events	COUNT(click_category='ViewMore')
device_type	String	User's device type	MIN(device_type) per session
department	String	User's department	MIN(department) per session
location	String	User's location	MIN(location) per session
job_title	String	User's job title	MIN(job_title) per session
query_language	String	Query language (UPPER, or "Unknown")	MIN(query_language) per session
avg_click_position	Float	Avg position of result clicks	AVG(clicked_position) for Result clicks
min_click_position	Integer	Best (lowest) click position	MIN(clicked_position) for Result clicks
max_news_results	Integer	Max news results shown	MAX(news_result_count)
avg_search_latency_ms	Float	Avg search latency (ms)	AVG(search_latency)
distinct_tabs_clicked	Integer	Unique tabs clicked	COUNT(DISTINCT clicked_tab)
distinct_filters_used	Integer	Unique filters used	COUNT(DISTINCT applied_filter)

searches\_daily.parquet

**Granularity:** One row per day

**Use case:** Daily KPIs, trend analysis

Column	Type	Description	Calculation
date	Date	The day	
total_events	Integer	All events	COUNT(*)

Column	Type	Description	Calculation
unique_sessions	Integer	Distinct sessions	COUNT(DISTINCT session_key)
unique_users	Integer	Distinct users	COUNT(DISTINCT user_id)
unique_search_terms	Integer	Distinct search queries	COUNT(DISTINCT search_term_normalized)
search_starts	Integer	SEARCH_TRIGGERED events	COUNT(SEARCH_TRIGGERED)
result_events	Integer	SEARCH_RESULT_COUNT events	COUNT(SEARCH_RESULT_COUNT)
click_events	Integer	Click events	COUNT(click_category)
null_results	Integer	Zero-result events	SUM(is_null_result)
result_events_with_results	Integer	Results with >0 hits	SUM(is_clickable_result)
sessions_with_results	Integer	Sessions that got results	From session_stats CTE
sessions_with_clicks	Integer	Sessions with clicks	From session_stats CTE
sessions_abandoned	Integer	Results but no click	sessions_with_results - sessions_with_clicks
sum_search_term_length	Integer	Sum of query lengths	SUM(search_term_length) - for weighted avg in DAX
sum_search_term_words	Integer	Sum of word counts	SUM(search_term_word_count) - for weighted avg in DAX
search_term_count	Integer	Count of queries	COUNT(search_term_length IS NOT NULL)
first_searches_of_day	Integer	First searches of day	COUNT(is_first_search_of_day)
success_clicks	Integer	Success clicks (SEARCH_RESULT_CLICK only)	COUNT(is_success_click=true)
clicks_result	Integer	SEARCH_RESULT_CLICK events	COUNT(click_category='Result')
clicks_trending	Integer	SEARCH_TRENDING_CLICKED events	COUNT(click_category='Trending')
clicks_tab	Integer	SEARCH_TAB_CLICK events	COUNT(click_category='Tab')
clicks_pagination	Integer	All pagination clicks	COUNT(click_category LIKE 'Pagination%')
clicks_pagination_all	Integer	All tab pagination	COUNT(click_category='Pagination_All')
clicks_pagination_news	Integer	News tab pagination	COUNT(click_category='Pagination_News')
clicks_pagination_goto	Integer	GoTo tab pagination	COUNT(click_category='Pagination_GoTo')
clicks_filter	Integer	SEARCH_FILTER_CLICK events	COUNT(click_category='Filter')
clicks_viewmore	Integer	SEARCH_VIEW_MORE_LINK events	COUNT(click_category='ViewMore')
sum_click_position	Integer	Sum of click positions	SUM(clicked_position) for Result clicks - for weighted avg in DAX
click_position_count	Integer	Result clicks with position	COUNT(clicked_position IS NOT NULL) for Result clicks
sum_news_result_count	Integer	Sum of news result counts	SUM(news_result_count) for result events
sum_search_latency_ms	Float	Sum of search latency	SUM(search_latency) - for weighted avg in DAX
latency_event_count	Integer	Events with latency data	COUNT(search_latency IS NOT NULL)
day_of_week	String	Day name	DAYNAME(session_date)
day_of_week_num	Integer	ISO day number (1=Mon)	ISODOW(session_date)
searches_night	Integer	Searches 03:00-09:00 CET (APAC)	Hour-based filter (CET)
searches_morning	Integer	Searches 09:00-16:00 CET (CET)	Hour-based filter (CET)
searches_afternoon	Integer	Searches 16:00-22:00 CET (Americas)	Hour-based filter (CET)
searches_evening	Integer	Searches 22:00-03:00 CET (Dead time)	Hour-based filter (CET)
new_users	Integer	First-time users today	Users where first_seen = today
returning_users	Integer	Repeat users today	Users where first_seen < today

searches\_terms.parquet

**Granularity:** One row per search term per day**Use case:** Search term performance analysis, content gap identification

Column	Type	Description	Calculation
session_date	Date	The day	

Column	Type	Description	Calculation
search_term	String	Normalized search query	LOWER(TRIM(query))
word_count	Integer	Words in query	COUNT of spaces + 1
search_count	Integer	Times searched today	COUNT(SEARCH_TRIGGERED)
unique_users	Integer	Users who searched this	COUNT(DISTINCT user_id)
unique_sessions	Integer	Sessions with this term	COUNT(DISTINCT session_key)
result_events	Integer	Result events for term	COUNT(SEARCH_RESULT_COUNT)
null_result_count	Integer	Zero-result count	SUM(is_null_result)
sum_result_count	Integer	Sum of result counts	SUM(cp_total_result_count) - for weighted avg in DAX
click_count	Integer	All clicks from this term	COUNT(click_category)
success_click_count	Integer	Success clicks (SEARCH_RESULT_CLICK only)	COUNT(is_success_click=true)
clicks_result	Integer	SEARCH_RESULT_CLICK events	COUNT(click_category='Result')
clicks_trending	Integer	SEARCH_TRENDING_CLICKED events	COUNT(click_category='Trending')
clicks_tab	Integer	SEARCH_TAB_CLICK events	COUNT(click_category='Tab')
clicks_pagination	Integer	All pagination clicks	COUNT(click_category LIKE 'Pagination%')
clicks_pagination_all	Integer	All tab pagination	COUNT(click_category='Pagination_All')
clicks_pagination_news	Integer	News tab pagination	COUNT(click_category='Pagination_News')
clicks_pagination_goto	Integer	GoTo tab pagination	COUNT(click_category='Pagination_GoTo')
clicks_filter	Integer	SEARCH_FILTER_CLICK events	COUNT(click_category='Filter')
clicks_viewmore	Integer	SEARCH_VIEW_MORE_LINK events	COUNT(click_category='ViewMore')
sum_click_position	Integer	Sum of click positions	SUM(clicked_position) for Result clicks - for weighted avg in DAX
click_position_count	Integer	Result clicks with position	COUNT(clicked_position IS NOT NULL) for Result clicks
sum_news_result_count	Integer	Sum of news result counts	SUM(news_result_count) for result events
sum_search_latency_ms	Float	Sum of search latency	SUM(search_latency) - for weighted avg in DAX
latency_event_count	Integer	Events with latency data	COUNT(search_latency IS NOT NULL)
clicks_with_timing	Integer	Clicks with timing data	COUNT(click after SEARCH_RESULT_COUNT)
sum_sec_to_click	Float	Sum of click times	SUM(ms_result_to_click) / 1000 - for weighted avg in DAX
searches_night	Integer	Searches 03:00-09:00 CET (APAC)	Hour-based filter (CET)
searches_morning	Integer	Searches 09:00-16:00 CET (CET)	Hour-based filter (CET)
searches_afternoon	Integer	Searches 16:00-22:00 CET (Americas)	Hour-based filter (CET)
searches_evening	Integer	Searches 22:00-03:00 CET (Dead time)	Hour-based filter (CET)
first_seen_date	Date	First day term appeared	MIN(session_date) over all time
is_new_term	Boolean	First appearance today	session_date = first_seen_date
month_num	Integer	Month number (1-12)	For seasonality analysis

searches\_term\_clicks.parquet

**Granularity:** One row per search term x clicked content (title + URL) per day

**Use case:** Understanding what content users actually click after searching — maps search intent to content discovery. Enables "What do people click when they search for X?" analysis.

Column	Type	Description	Calculation
session_date	Date	The day	
search_term	String	Normalized search query	Forward-filled from SEARCH_TRIGGERED
clicked_result_title	String	Title of the clicked result	From SEARCH_RESULT_CLICK event
clicked_result_url	String	URL of the clicked result	From SEARCH_RESULT_CLICK event
click_count	Integer	Times this content was clicked for this term	COUNT(*)
unique_users	Integer	Distinct users who clicked this	COUNT(DISTINCT user_id)

Column	Type	Description	Calculation
unique_sessions	Integer	Sessions with this click	COUNT(DISTINCT session_key)
sum_click_position	Integer	Sum of click positions	SUM(clicked_position) - for weighted avg in DAX
click_position_count	Integer	Clicks with position data	COUNT(clicked_position IS NOT NULL)
top_department	String	Most common department	MODE(department)
top_device_type	String	Most common device type	MODE(device_type)

#### Relationship to other tables:

- Joins to `searches_terms` on `session_date + search_term` (many-to-one: each term can have multiple clicked results)
- Joins to `searches_daily` on `session_date` (many-to-one)

#### Key analysis patterns:

- **Content discovery:** "When users search for 'expense report', what do they actually click?"
- **Content gaps:** Terms with many searches in `searches_terms` but few/no rows in `searches_term_clicks`
- **Content consolidation:** Multiple URLs clicked for the same search term suggests scattered content
- **Click position quality:** Low `sum_click_position / click_position_count` = content ranks well for this term

## 6. Power BI Calculated Columns

These columns are created in Power BI using DAX and are not present in the parquet files.

### searches\_terms Table

#### Query\_Length\_Bucket

Categorizes search queries by word count for visualization.

```
Query_Length_Bucket =
SWITCH(
    TRUE(),
    searches_terms[word_count] = 1, "1 word",
    searches_terms[word_count] = 2, "2 words",
    searches_terms[word_count] = 3, "3 words",
    searches_terms[word_count] = 4, "4 words",
    searches_terms[word_count] >= 5, "5+ words",
    "Unknown"
)
```

#### Query\_Length\_Sort

Sort order for Query\_Length\_Bucket. Set "Sort by column" in Power BI.

```
Query_Length_Sort =
SWITCH(
    TRUE(),
    searches_terms[word_count] = 1, 1,
    searches_terms[word_count] = 2, 2,
    searches_terms[word_count] = 3, 3,
    searches_terms[word_count] = 4, 4,
    searches_terms[word_count] >= 5, 5,
    99
)
```

### Term\_Outcome

Classifies search term performance into actionable categories based on **success clicks** (content discovery).

```
Term_Outcome =
VAR nullRate = DIVIDE([null_result_count], [result_events], 0)
VAR ctr = DIVIDE([success_click_count], [search_count], 0)
RETURN
SWITCH(
    TRUE(),
    nullRate = 1, "Zero Results",
    nullRate > 0.5, "Mostly No Results",
    ctr = 0, "No Clicks",
    ctr < 0.2, "Low CTR",
```

```
"Success"
)
```

**Note:** Uses `success_click_count` (SEARCH\_RESULT\_CLICK only), not `click_count` (all clicks).

Category	Meaning	Action
Zero Results	100% null rate	Content gap - add content
Mostly No Results	>50% null rate	Partial gap - improve coverage
No Clicks	Has results but 0 clicks	Poor relevance - tune ranking
Low CTR	<20% click rate	Suboptimal - review content
Success	Good performance	Monitor

### Term Lifecycle Filter

Classifies term age into lifecycle stages. Use as a slicer/filter (measures cannot be used as slicers).

```
Term Lifecycle Filter =
VAR Age =
    DATEDIFF(
        searches_terms[first_seen_date],
        searches_terms[session_date],
        DAY
    ) + 1
RETURN
SWITCH(
    TRUE(),
    Age <= 3, "New (<= 3 days)",
    Age <= 7, "Emerging (4-7 days)",
    Age <= 14, "Establishing (8-14 days)",
    Age <= 30, "Established (15-30 days)",
    "Mature (31+ days)"
)
```

### Term Lifecycle Filter Sort

Sort order for Term Lifecycle Filter. Set "Sort by column" in Power BI.

```
Term Lifecycle Filter Sort =
VAR Age =
    DATEDIFF(
        searches_terms[first_seen_date],
        searches_terms[session_date],
        DAY
    ) + 1
RETURN
SWITCH(
    TRUE(),
    Age <= 3, 1,
    Age <= 7, 2,
    Age <= 14, 3,
    Age <= 30, 4,
    5
)
```

Stage	Days Since First Seen	Sort	Interpretation
New (<= 3 days)	1-3	1	Just emerged, monitor for trending
Emerging (4-7 days)	4-7	2	Past initial spike, validate staying power
Establishing (8-14 days)	8-14	3	Building consistent usage pattern
Established (15-30 days)	15-30	4	Regular part of user vocabulary
Mature (31+ days)	31+	5	Long-standing, stable terms

### searches\_journeys Table

#### Latency\_Bucket

Categorizes search latency for visualization.

```
Latency_Bucket =
SWITCH(
    TRUE(),
    ISBLANK(searches_journeys[avg_search_latency_ms]), "No Data",
    searches_journeys[avg_search_latency_ms] < 500, "< 0.5s",
    searches_journeys[avg_search_latency_ms] < 1000, "0.5-1s",
    searches_journeys[avg_search_latency_ms] < 2000, "1-2s",
    "> 2s"
)
```

### Click\_Position\_Bucket

Categorizes click position for visualization.

```
Click_Position_Bucket =
SWITCH(
    TRUE(),
    ISBLANK(searches_journeys[avg_click_position]), "No Click",
    searches_journeys[avg_click_position] <= 1, "Position 1",
    searches_journeys[avg_click_position] <= 3, "Top 3",
    searches_journeys[avg_click_position] <= 5, "Top 5",
    "Below 5"
)
```

### Journey\_Type

Combines outcome and behavior flags for segmentation.

```
Journey_Type =
searches_journeys[journey_outcome] &
IF(searches_journeys[had_reformulation], " (Refined)", "") &
IF(searches_journeys[recovered_from_null], " (Recovered)", "")
```

## 7. Power BI Measures

These measures are created in Power BI for aggregated calculations.

### Search Effectiveness Score

Combined metric considering both success CTR and null rate. Higher is better.

```
Search Effectiveness Score =
VAR ctr = DIVIDE(SUM(searches_terms[success_click_count]), SUM(searches_terms[search_count]), 0)
VAR nullRate = DIVIDE(SUM(searches_terms=null_result_count), SUM(searches_terms[result_events]), 0)
RETURN
(ctr * 100) - (nullRate * 50)
```

**Note:** Uses `success_click_count` (SEARCH\_RESULT\_CLICK only) for accurate content discovery measurement.

### Score interpretation:

- Positive scores: Good performance (CTR outweighs null rate penalty)
- Near zero: Balanced but could improve
- Negative scores: High null rates hurting performance

### Term Success CTR %

Success click-through rate for search terms (actual content clicks only).

```
Term Success CTR % =
DIVIDE(
    SUM(searches_terms[success_click_count]),
    SUM(searches_terms[search_count]),
    0
) * 100
```

### Term All Clicks Rate %

All clicks rate including navigation (tabs, pagination, filters).

```
Term All Clicks Rate % =
DIVIDE(
    SUM(searches_terms[click_count]),
    SUM(searches_terms[search_count]),
    0
) * 100
```

### Term Null Rate %

Percentage of searches returning zero results.

```
Term Null Rate % =
DIVIDE(
    SUM(searches_terms=null_result_count),
    SUM(searches_terms=result_events),
    0
) * 100
```

### Weighted Avg Search Term Length

Correctly weighted average across days (use instead of AVERAGE on avg\_search\_term\_length).

```
Weighted Avg Search Term Length =
DIVIDE(
    SUM(searches_daily[sum_search_term_length]),
    SUM(searches_daily[search_term_count]),
    0
)
```

### Weighted Avg Search Term Words

Correctly weighted average across days.

```
Weighted Avg Search Term Words =
DIVIDE(
    SUM(searches_daily[sum_search_term_words]),
    SUM(searches_daily[search_term_count]),
    0
)
```

### Weighted Avg Search Latency

Correctly weighted average search latency across days (for daily and terms tables).

```
Avg Search Latency (ms) =
DIVIDE(
    SUM(searches_daily[sum_search_latency_ms]),
    SUM(searches_daily[latency_event_count]),
    BLANK()
)
```

### Avg News Results per Search

Average news results per result event.

```
Avg News Results =
DIVIDE(
    SUM(searches_daily[sum_news_result_count]),
    SUM(searches_daily[result_events_with_results]),
    BLANK()
)
```

### ViewMore Click Rate

Percentage of clicks that are "view more" navigations.

```

ViewMore Click Rate % =
DIVIDE(
    SUM(searches_daily[clicks_viewmore]),
    SUM(searches_daily[click_events]),
    0
) * 100

```

### Weighted Avg Click Position

Correctly weighted average click position across days/terms (for daily and terms tables).

```

Avg Click Position =
DIVIDE(
    SUM(searches_daily[sum_click_position]),
    SUM(searches_daily[click_position_count]),
    BLANK()
)

```

### Weighted Avg Sec to Click

Correctly weighted average click time (for terms aggregation).

```

Weighted Avg Sec to Click =
DIVIDE(
    SUM(searches_terms[sum_sec_to_click]),
    SUM(searches_terms[clicks_with_timing]),
    0
)

```

## Example: Full Data Flow

### Raw Input (from App Insights)

```

timestamp,name,user_Id,session_Id,CP_searchQuery,CP_totalResultCount
2025-01-15 10:30:15.123456,Search_Started,user123,sess456,budget report,
2025-01-15 10:30:15.234567,Search_Completed,user123,sess456,budget report,
2025-01-15 10:30:15.567890,Search_Result_Count,user123,sess456,,15
2025-01-15 10:30:18.890123,Search_Tab_Click,user123,sess456,,

```

### After Processing (searches\_raw.parquet)

timestamp	name	session_key	prev_event	ms_since_prev	search_term	is_null_result	click_category
10:30:15.123	SEARCH_TRIGGERED	2025-01-15_user123_sess456	NULL	NULL	budget report	NULL	NULL
10:30:15.234	SEARCH_COMPLETED	2025-01-15_user123_sess456	SEARCH_TRIGGERED	111	NULL	NULL	NULL
10:30:15.567	SEARCH_RESULT_COUNT	2025-01-15_user123_sess456	SEARCH_COMPLETED	333	NULL	false	NULL
10:30:18.890	SEARCH_TAB_CLICK	2025-01-15_user123_sess456	SEARCH_RESULT_COUNT	3323	NULL	NULL	General

### Aggregated (searches\_journeys.parquet)

session_date	total_events	search_count	click_count	sec_search_to_result	sec_result_to_click	journey_outcome
2025-01-15	4	1	1	0.44	3.32	Success

### Calculation breakdown:

- **sec\_search\_to\_result**: 10:30:15.567 - 10:30:15.123 = 444ms = 0.44s
- **sec\_result\_to\_click**: 10:30:18.890 - 10:30:15.567 = 3323ms = 3.32s
- **journey\_outcome**: click\_count > 0 --> "Success"

### Version History

Version	Date	Changes
1.0	2025-01-15	Initial documentation
1.1	2025-01-16	Added missing parquet columns (click breakdowns, sort columns, timing aggregates), Power BI calculated columns section, Power BI measures section
1.2	2025-01-23	Added CET timezone support: timestamp_cet columns, CET-based session_date/event_hour/event_weekday, updated time distribution documentation
1.3	2025-01-23	Expanded event documentation: added initialization events, SEARCH_STARTED distinction, click event details (SEARCH_RESULT_CLICK, SEARCH_TRENDING_CLICKED, SEARCH_FILTER_CLICK, SEARCH_FAILED)
1.4	2025-01-26	Updated click categories (Result, Trending, Tab, Pagination_*, Filter). Added is_success_click (SEARCH_RESULT_CLICK only - trending clicks are search initiation, not content discovery). Updated journey_outcome to use success_click_count. Changed time distribution to regional alignment (0-8 APAC, 8-12 EMEA, 12-18 overlap, 18-24 Americas).
1.5	2025-01-26	Added "Engaged" journey_outcome category for sessions with navigation clicks but no result clicks. Updated recovered_from_null to use success_click_count. Sort order: 1=Success, 2=Engaged, 3=Abandoned, 4=No Results.
1.6	2025-01-26	Changed session_complexity to use user actions (searches + clicks) instead of all telemetry events. Renamed "Single Event" to "Single Action".
1.7	2025-01-26	Added Applnights Identifiers section explaining user_id (cookie-based) and session_id (30-min inactivity timeout) behavior and implications for metrics.
1.8	2025-01-29	Updated time distribution buckets to align with regional business hours: APAC (03-09 CET), CET (09-16 CET), Americas (16-22 CET), Dead time (22-03 CET). Column names unchanged for Power BI compatibility.
1.9	2025-01-29	Removed pre-calculated rate/average columns that cannot be aggregated: click_rate_pct, null_rate_pct, session_success_rate_pct, session_abandonment_rate_pct, avg_searches_per_session, avg_search_term_length, avg_search_term_words, avg_sec_to_click. Use DAX measures with building block columns instead.
2.0	2025-02-09	Adapted to new App Insights 4-level nesting structure. Added SEARCH_RESULT_CLICKED and SEARCH_VIEW_MORE_LINK events. Added ViewMore click category. Added 12 new fields from dynamic column resolution: clicked_position, clicked_tab, applied_filter, clicked_result_title, clicked_result_url, news_result_count, query_language, device_type, department, location, job_title, search_latency. Updated all three aggregation files (daily, journeys, terms) with new metrics. Added new Power BI measures (Avg Search Latency, Avg News Results, ViewMore Click Rate) and calculated columns (Latency_Bucket, Click_Position_Bucket).
2.1	2025-02-10	Added searches_term_clicks.parquet (term → clicked content mapping). Normalized query_language to UPPER with "Unknown" title case. Added Term Lifecycle Filter and Term Lifecycle Filter Sort calculated columns for slicer usage.