<https://gemini.google.com/u/1/app/5b5f3092ace62463>

\* \* \* \* \*

Production-Ready Specification: Module 8 Views

Version: 1.0

Date: May 18, 2025

This document outlines the specifications for views designed to simplify querying and data access for the Platform Content module (Module 8), focusing on articles and their associated media.

View 1: `public.published\_articles\_view`

----------------------------------------

1\. View Name

public.published\_articles\_view

2\. Version & Date

- Version: 1.0

- Date: May 18, 2025

3\. Purpose & Primary Use-Cases

To provide a denormalized, read-only list of articles that are currently published and not soft-deleted. This view includes key author details and featured image information, making it suitable for populating public-facing article listings or individual article displays. Content fields are presented in the platform's primary reference language (English).

4\. Source Tables

- `public.articles`

- `public.profiles`

- `public.media`

5\. View Definition (DDL)

SQL

```

CREATE OR REPLACE VIEW public.published\_articles\_view AS

SELECT

a.id,

a.title, -- In primary reference language

a.slug,

a.excerpt, -- In primary reference language

a.body\_content, -- In primary reference language (consider if full body is always needed in a list view)

a.publication\_date,

a.author\_profile\_id AS author\_id,

p.public\_display\_name AS author\_public\_display\_name,

p.public\_avatar\_media\_id AS author\_avatar\_media\_id, -- App layer can use this to get actual URL from media table + image\_variants\_json

a.featured\_image\_media\_id,

fm.storage\_bucket\_name AS featured\_image\_bucket\_name,

fm.storage\_object\_path\_original AS featured\_image\_storage\_object\_path, -- App layer can construct URL

fm.image\_variants\_json AS featured\_image\_variants\_json, -- Provides different sizes/formats

fm.default\_alt\_text AS featured\_image\_default\_alt\_text, -- In primary reference language

a.tags,

a.associated\_trail\_id,

a.associated\_region\_id,

a.associated\_town\_id,

a.created\_at AS article\_created\_at,

a.updated\_at AS article\_updated\_at

FROM

public.articles a

LEFT JOIN

public.profiles p ON a.author\_profile\_id = p.id

LEFT JOIN

public.media fm ON a.featured\_image\_media\_id = fm.id

WHERE

a.article\_status = 'published'::public.content\_visibility\_status\_enum

AND a.deleted\_at IS NULL;

COMMENT ON VIEW public.published\_articles\_view IS 'Provides a denormalized view of published articles with author and featured image details, primarily in the reference language. Module 8. Version 1.0.';

```

6\. Column Descriptions

| Column | Source Table(s) / Logic | Description |

| `id` | `articles.id` | Unique identifier of the article. |

| `title` | `articles.title` | Title of the article (in primary reference language). |

| `slug` | `articles.slug` | URL-friendly slug for the article. |

| `excerpt` | `articles.excerpt` | Short summary of the article (in primary reference language). |

| `body\_content` | `articles.body\_content` | Main content of the article (in primary reference language). |

| `publication\_date` | `articles.publication\_date` | Date and time the article was published. |

| `author\_id` | `articles.author\_profile\_id` | UUID of the article's author (references `profiles.id`). |

| `author\_public\_display\_name` | `profiles.public\_display\_name` | Public display name of the author. |

| `author\_avatar\_media\_id` | `profiles.public\_avatar\_media\_id` | Media ID of the author's avatar. Application layer can use this to construct the full URL via `public.media` and `image\_variants\_json`. |

| `featured\_image\_media\_id` | `articles.featured\_image\_media\_id` | Media ID of the article's featured image. |

| `featured\_image\_bucket\_name` | `media.storage\_bucket\_name` (from featured image join) | Storage bucket for the featured image. |

| `featured\_image\_storage\_object\_path` | `media.storage\_object\_path\_original` (from featured image join) | Original storage path for the featured image. |

| `featured\_image\_variants\_json` | `media.image\_variants\_json` (from featured image join) | JSONB object containing paths to different variants (sizes, formats) of the featured image. |

| `featured\_image\_default\_alt\_text` | `media.default\_alt\_text` (from featured image join) | Default alt text for the featured image (in primary reference language). |

| `tags` | `articles.tags` | Array of tags associated with the article. |

| `associated\_trail\_id` | `articles.associated\_trail\_id` | Optional ID of an associated trail. |

| `associated\_region\_id` | `articles.associated\_region\_id` | Optional ID of an associated region. |

| `associated\_town\_id` | `articles.associated\_town\_id` | Optional ID of an associated town. |

| `article\_created\_at` | `articles.created\_at` | Timestamp of when the article record was created. |

| `article\_updated\_at` | `articles.updated\_at` | Timestamp of when the article record was last updated. |

7\. Querying & Usage Notes

- This view is designed for read-only access to articles intended for public display.

- Textual content fields (`title`, `excerpt`, `body\_content`, `featured\_image\_default\_alt\_text`) are presented in the platform's primary reference language (English).

- For displaying content in other languages, the application layer should query the `public.translations` table using the `articles.id` (for article content) or `featured\_image\_media\_id` (for featured image alt text) as the `row\_foreign\_key`, along with the appropriate `table\_identifier` and `column\_identifier`. Alternatively, a dedicated database function accepting a language code is recommended for more complex translation retrieval logic.

- The `author\_avatar\_media\_id`, `featured\_image\_bucket\_name`, `featured\_image\_storage\_object\_path`, and `featured\_image\_variants\_json` columns provide the necessary information for applications to construct full, optimized image URLs.

8\. RLS Considerations

- The view's `WHERE` clause intrinsically filters for articles where `article\_status = 'published'` and `deleted\_at IS NULL`.

- Row-Level Security policies on the underlying tables (`public.articles`, `public.profiles`, `public.media`) will still be enforced when this view is queried.

- For public access, ensure appropriate `SELECT` grants are applied to this view for roles like `anonymous` or `authenticated\_user`. Example:

SQL

```

-- GRANT SELECT ON public.published\_articles\_view TO authenticated\_user;

-- GRANT SELECT ON public.published\_articles\_view TO anonymous\_user;

```

9\. Dependencies

- `public.articles` table

- `public.profiles` table

- `public.media` table

- `public.content\_visibility\_status\_enum` type (used by `public.articles.article\_status`)

10\. Next-Action Checklist

- 🔴 Create View: Execute the `CREATE OR REPLACE VIEW public.published\_articles\_view` DDL.

- 🟢 Test: Thoroughly test the view with various data scenarios (e.g., articles with/without authors, with/without featured images).

- 🟢 Performance: Monitor performance under load. For very high-traffic scenarios, consider if a materialized view would be more appropriate (though this adds complexity for data freshness).

- 🔴 Security: Review and apply necessary `GRANT` permissions for relevant user roles.

- 🟠 Translations Strategy: Confirm application-level or database function strategy for fetching translations for fields served by this view.

\* \* \* \* \*

View 2: `public.article\_media\_details\_view`

-------------------------------------------

1\. View Name

public.article\_media\_details\_view

2\. Version & Date

- Version: 1.0

- Date: May 18, 2025

3\. Purpose & Primary Use-Cases

To provide a denormalized, read-only list of all approved media items associated with articles through the public.article\_media linking table. This includes media role information, display order, paths to media files, and any text overrides (primarily in the reference language). It is useful for constructing image galleries or displaying inline media elements for a specific article.

4\. Source Tables

- `public.article\_media`

- `public.media`

- `public.media\_roles\_master`

5\. View Definition (DDL)

SQL

```

CREATE OR REPLACE VIEW public.article\_media\_details\_view AS

SELECT

am.id AS article\_media\_id,

am.article\_id,

am.media\_id,

am.media\_role\_code,

mrm.default\_display\_name AS media\_role\_name, -- In primary reference language

am.display\_order,

am.caption\_override, -- In primary reference language

am.alt\_text\_override, -- In primary reference language

m.storage\_bucket\_name,

m.storage\_object\_path\_original,

m.image\_variants\_json,

m.default\_alt\_text AS media\_default\_alt\_text, -- Fallback, in primary reference language

m.default\_caption AS media\_default\_caption, -- Fallback, in primary reference language

m.file\_mime\_type,

m.image\_width\_px\_original,

m.image\_height\_px\_original,

am.created\_at AS link\_created\_at,

am.updated\_at AS link\_updated\_at

FROM

public.article\_media am

JOIN

public.media m ON am.media\_id = m.id

JOIN

public.media\_roles\_master mrm ON am.media\_role\_code = mrm.role\_code

WHERE

m.deleted\_at IS NULL

AND m.media\_status = 'published\_approved'::public.media\_status\_enum -- Only show published/approved media items

AND mrm.is\_active = true; -- Only consider active media roles

COMMENT ON VIEW public.article\_media\_details\_view IS 'Provides denormalized details for approved media items linked to articles, including roles and text overrides (in primary reference language). For Module 8. Version 1.0.';

```

6\. Column Descriptions

| Column | Source Table(s) / Logic | Description |

| `article\_media\_id` | `article\_media.id` | Unique ID of the link between an article and a media item. |

| `article\_id` | `article\_media.article\_id` | ID of the article this media is linked to. |

| `media\_id` | `article\_media.media\_id` | ID of the linked media item. |

| `media\_role\_code` | `article\_media.media\_role\_code` | Code for the role this media plays in the article (e.g., `gallery\_image`). |

| `media\_role\_name` | `media\_roles\_master.default\_display\_name` | Human-readable name of the media role (in primary reference language). |

| `display\_order` | `article\_media.display\_order` | Display order for this media item within its role for the article. |

| `caption\_override` | `article\_media.caption\_override` | Context-specific caption for this media in this article (in primary reference language). |

| `alt\_text\_override` | `article\_media.alt\_text\_override` | Context-specific alt text for this media in this article (in primary reference language). |

| `storage\_bucket\_name` | `media.storage\_bucket\_name` | Storage bucket of the linked media item. |

| `storage\_object\_path\_original` | `media.storage\_object\_path\_original` | Original storage path of the linked media item. |

| `image\_variants\_json` | `media.image\_variants\_json` | JSONB object with paths to different variants (sizes, formats) of the media item. |

| `media\_default\_alt\_text` | `media.default\_alt\_text` | Fallback default alt text from the media item itself (in primary reference language). |

| `media\_default\_caption` | `media.default\_caption` | Fallback default caption from the media item itself (in primary reference language). |

| `file\_mime\_type` | `media.file\_mime\_type` | MIME type of the media item. |

| `image\_width\_px\_original` | `media.image\_width\_px\_original` | Original width of the media item in pixels (if an image). |

| `image\_height\_px\_original` | `media.image\_height\_px\_original` | Original height of the media item in pixels (if an image). |

| `link\_created\_at` | `article\_media.created\_at` | Timestamp of when the link record in `article\_media` was created. |

| `link\_updated\_at` | `article\_media.updated\_at` | Timestamp of when the link record in `article\_media` was last updated. |

7\. Querying & Usage Notes

- Typically queried by `article\_id` to retrieve all associated media details for a specific article.

- Text fields (`media\_role\_name`, `caption\_override`, `alt\_text\_override`, `media\_default\_alt\_text`, `media\_default\_caption`) are presented in the platform's primary reference language.

- For displaying content in other languages, the application layer should query the `public.translations` table for the `caption\_override` and `alt\_text\_override` (using `article\_media.id` as `row\_foreign\_key`) and for `media\_role\_name` (using `media\_roles\_master.role\_code` as `row\_foreign\_key`). Fallback media captions/alt text would also need translation lookup using `media.id`. A dedicated database function is recommended for complex translation logic.

- The view's `WHERE` clause filters for media items that are `published\_approved` and media roles that are `is\_active`.

8\. RLS Considerations

- RLS on underlying tables (`public.article\_media`, `public.media`, `public.media\_roles\_master`) will be enforced.

- Access to this view should ideally be controlled based on the user's permission to view the parent article referenced by `article\_media.article\_id`. This might involve:

- A `SECURITY INVOKER` view where RLS policies on `article\_media` (which in turn could check `articles`) handle access.

- Joining with `public.articles` within the view's RLS policy (if view-level RLS is added) to check the parent article's status and RLS.

- Example (conceptual, assuming RLS on `article\_media` checks parent article visibility): <!-- end list -->

SQL

```

-- GRANT SELECT ON public.article\_media\_details\_view TO authenticated\_user;

-- GRANT SELECT ON public.article\_media\_details\_view TO anonymous\_user;

-- (Actual access depends on RLS policies on public.article\_media)

```

9\. Dependencies

- `public.article\_media` table

- `public.media` table

- `public.media\_roles\_master` table

- `public.media\_status\_enum` type (used by `public.media.media\_status`)

10\. Next-Action Checklist

- 🔴 Create View: Execute the `CREATE OR REPLACE VIEW public.article\_media\_details\_view` DDL.

- 🟢 Test: Thoroughly test with various articles, media items, roles, and overrides.

- 🟢 Performance: Monitor performance, especially when querying for articles with many media items. Ensure underlying table indexes are effective.

- 🔴 Security: Review and apply necessary `GRANT` permissions. Define and test RLS policies on the base `public.article\_media` table to ensure this view exposes data correctly according to user permissions for the parent article.

- 🟠 Translations Strategy: Confirm application-level or database function strategy for fetching translations for fields served by this view.