https://gemini.google.com/u/1/app/a3fdff8118a11194

Okay, let's define the specification for the view v\_towns\_list\_localized.

### **Specification:** public.v\_towns\_list\_localized **(View)**

**1. Purpose & Primary Use-Cases**

* **Purpose**: To provide a simplified and denormalized representation of essential town information required for list displays, with textual fields already translated into available languages. Each row in the view represents a town in a specific language.
* **Primary Use-Cases**:
  + Populating lists of towns in a user's selected language on websites and mobile applications (e.g., search results, regional town lists).
  + Simplifying queries for the GET /towns API endpoint by pre-joining towns with translations and key related entities.
  + Providing a basis for filtering towns by various criteria while displaying localized names.

**2. View Definition (SQL DDL)**

SQL

CREATE OR REPLACE VIEW public.v\_towns\_list\_localized

AS

SELECT

t.id AS town\_id,

t.slug AS town\_slug,

-- Core town attributes for list display & filtering

t.latitude\_centroid,

t.longitude\_centroid,

t.is\_major\_stage\_town,

t.content\_visibility\_status AS town\_content\_visibility\_status, -- For admin filtering or specific UI

t.deleted\_at AS town\_deleted\_at, -- For RLS and potential admin UI

-- Language of this localized record

town\_name\_trans.language\_code,

-- Translated Town Fields

town\_name\_trans.translated\_text AS town\_name,

town\_short\_desc\_trans.translated\_text AS town\_short\_description,

-- Linked Region Info (Translated Name & Key Fields)

r.id AS region\_id,

r.slug AS region\_slug,

region\_name\_trans.translated\_text AS region\_name,

-- r.is\_active AS region\_is\_active, -- regions use deleted\_at & status, not is\_active

(r.deleted\_at IS NULL AND r.content\_visibility\_status = 'published') AS region\_is\_visible,

-- Linked Province Info (Translated Name & Key Fields)

p.id AS province\_id,

p.code AS province\_code,

province\_name\_trans.translated\_text AS province\_name,

p.is\_active AS province\_is\_active,

-- Linked Town Type Info (Translated Name & Icon)

ttm.code AS town\_type\_code,

town\_type\_name\_trans.translated\_text AS town\_type\_name,

ttm.icon\_identifier AS town\_type\_icon\_identifier,

ttm.is\_active AS town\_type\_is\_active,

-- Primary Image Thumbnail Info

m.id AS primary\_media\_id,

-- Assuming 'thumbnail\_S' is a defined key in image\_variants\_json

m.image\_variants\_json ->> 'thumbnail\_S' AS primary\_image\_thumbnail\_url,

media\_alt\_text\_trans.translated\_text AS primary\_image\_alt\_text,

(m.media\_status = 'approved') AS media\_is\_approved -- Assuming media has a status like 'approved'

FROM

public.towns t

-- Join for Town Name (drives the language for each view row)

INNER JOIN public.translations town\_name\_trans

ON town\_name\_trans.table\_identifier = 'towns'

AND town\_name\_trans.row\_foreign\_key = t.id::TEXT

AND town\_name\_trans.column\_identifier = 'name'

AND town\_name\_trans.translation\_status = 'published\_live' -- Assuming this status indicates ready-to-use

-- Join for Town Short Description

LEFT JOIN public.translations town\_short\_desc\_trans

ON town\_short\_desc\_trans.table\_identifier = 'towns'

AND town\_short\_desc\_trans.row\_foreign\_key = t.id::TEXT

AND town\_short\_desc\_trans.column\_identifier = 'short\_description'

AND town\_short\_desc\_trans.language\_code = town\_name\_trans.language\_code

AND town\_short\_desc\_trans.translation\_status = 'published\_live'

-- Join for Region (and its translated name)

LEFT JOIN public.regions r ON t.region\_id = r.id

LEFT JOIN public.translations region\_name\_trans

ON region\_name\_trans.table\_identifier = 'regions'

AND region\_name\_trans.row\_foreign\_key = r.id::TEXT

AND region\_name\_trans.column\_identifier = 'name'

AND region\_name\_trans.language\_code = town\_name\_trans.language\_code

AND region\_name\_trans.translation\_status = 'published\_live'

-- Join for Province (and its translated name)

LEFT JOIN public.provinces p ON t.province\_id = p.id

LEFT JOIN public.translations province\_name\_trans

ON province\_name\_trans.table\_identifier = 'provinces'

AND province\_name\_trans.row\_foreign\_key = p.id::TEXT

AND province\_name\_trans.column\_identifier = 'name'

AND province\_name\_trans.language\_code = town\_name\_trans.language\_code

AND province\_name\_trans.translation\_status = 'published\_live'

-- Join for Town Type (and its translated name)

LEFT JOIN public.town\_types\_master ttm ON t.town\_type\_code = ttm.code

LEFT JOIN public.translations town\_type\_name\_trans

ON town\_type\_name\_trans.table\_identifier = 'town\_types\_master'

AND town\_type\_name\_trans.row\_foreign\_key = ttm.code -- PK is TEXT 'code'

AND town\_type\_name\_trans.column\_identifier = 'name'

AND town\_type\_name\_trans.language\_code = town\_name\_trans.language\_code

AND town\_type\_name\_trans.translation\_status = 'published\_live'

-- Join for Primary Media (and its translated alt text)

LEFT JOIN public.media m ON t.primary\_media\_id = m.id

LEFT JOIN public.translations media\_alt\_text\_trans

ON media\_alt\_text\_trans.table\_identifier = 'media' -- Assuming 'media' as table\_identifier for media text

AND media\_alt\_text\_trans.row\_foreign\_key = m.id::TEXT -- Assuming media.id is the FK

AND media\_alt\_text\_trans.column\_identifier = 'alt\_text'

AND media\_alt\_text\_trans.language\_code = town\_name\_trans.language\_code

AND media\_alt\_text\_trans.translation\_status = 'published\_live'

WHERE

-- Ensure only active/published towns are initially considered by the view's base query

t.deleted\_at IS NULL AND t.content\_visibility\_status = 'published';

-- Further filtering for linked entities' visibility (is\_active, deleted\_at, status)

-- should ideally be handled in the WHERE clause of queries \*against\* the view,

-- or embedded here if the business rule is strict that a town list item shouldn't appear

-- if its region is (e.g.) not active.

-- Example: AND (r.id IS NULL OR (r.deleted\_at IS NULL AND r.content\_visibility\_status = 'published'))

-- AND (p.id IS NULL OR p.is\_active = true)

-- AND (ttm.code IS NULL OR ttm.is\_active = true)

-- AND (m.id IS NULL OR m.media\_status = 'approved')

-- For simplicity in the base view, these are exposed as boolean flags (e.g., region\_is\_visible).

-- The API can then decide to filter based on these flags.

COMMENT ON VIEW public.v\_towns\_list\_localized IS 'Provides a denormalized and localized list of towns for display purposes, joining towns with their translations and key related entity names in various languages. Each row represents a town in one specific language. Version 1.0';

**3. Output Columns**

| **Column Name** | **Data Type** | **Description** |
| --- | --- | --- |
| town\_id | INTEGER | The unique ID of the town (from public.towns.id). |
| town\_slug | TEXT | The URL-friendly slug of the town (from public.towns.slug). |
| latitude\_centroid | DOUBLE PRECISION | Latitude of the town's centroid. |
| longitude\_centroid | DOUBLE PRECISION | Longitude of the town's centroid. |
| is\_major\_stage\_town | BOOLEAN | Flag indicating if the town is a major stage town. |
| town\_content\_visibility\_status | content\_visibility\_status\_enum | Visibility status of the base town record. |
| town\_deleted\_at | TIMESTAMPTZ | Soft deletion timestamp for the base town record. |
| language\_code | TEXT | The language code (e.g., 'en', 'it') for the translated textual fields in this row. |
| town\_name | TEXT | The translated name of the town in the specified language\_code. |
| town\_short\_description | TEXT | The translated short description of the town in the specified language\_code. |
| region\_id | INTEGER | The ID of the town's primary region. |
| region\_slug | TEXT | The slug of the town's primary region. |
| region\_name | TEXT | The translated name of the town's primary region in the specified language\_code. |
| region\_is\_visible | BOOLEAN | Indicates if the linked region is currently visible (published and not deleted). |
| province\_id | INTEGER | The ID of the town's province. |
| province\_code | TEXT | The official code of the town's province. |
| province\_name | TEXT | The translated name of the town's province in the specified language\_code. |
| province\_is\_active | BOOLEAN | Indicates if the linked province is active. |
| town\_type\_code | TEXT | The code of the town's type. |
| town\_type\_name | TEXT | The translated name of the town's type in the specified language\_code. |
| town\_type\_icon\_identifier | TEXT | The icon identifier for the town's type. |
| town\_type\_is\_active | BOOLEAN | Indicates if the linked town type is active. |
| primary\_media\_id | INTEGER | The ID of the town's primary media image. |
| primary\_image\_thumbnail\_url | TEXT | URL for a small thumbnail of the town's primary image (e.g., from image\_variants\_json ->> 'thumbnail\_S'). |
| primary\_image\_alt\_text | TEXT | The translated alt text for the town's primary image in the specified language\_code. |
| media\_is\_approved | BOOLEAN | Indicates if the linked media item is approved. |

**4. Example Usage**

To get a list of towns in Italian, ordered by name:

SQL

SELECT

town\_id,

town\_slug,

town\_name,

town\_short\_description,

region\_name,

primary\_image\_thumbnail\_url

FROM

public.v\_towns\_list\_localized

WHERE

language\_code = 'it'

-- AND region\_is\_visible = TRUE -- Example additional filter

-- AND province\_is\_active = TRUE

-- AND town\_type\_is\_active = TRUE

-- AND (media\_is\_approved = TRUE OR primary\_media\_id IS NULL)

ORDER BY

town\_name ASC

LIMIT 20 OFFSET 0;

Filtering by other attributes like region\_id or is\_major\_stage\_town can also be done in the WHERE clause.

**5. Underlying Tables & Key Joins**

* **Primary Table**: public.towns (aliased as t)
* **Key Joins**:
  + public.translations (multiple times for town\_name, town\_short\_description, region\_name, province\_name, town\_type\_name, media\_alt\_text). Joins are based on table\_identifier, row\_foreign\_key, column\_identifier, and language\_code. Crucially, translation\_status = 'published\_live' should be part of these join conditions.
  + public.regions (as r): For region details.
  + public.provinces (as p): For province details.
  + public.town\_types\_master (as ttm): For town type details.
  + public.media (as m): For primary image details.

**6. RLS (Row-Level Security) Considerations**

* The view will be defined with SECURITY INVOKER by default. This means that RLS policies defined on the underlying tables (towns, regions, provinces, town\_types\_master, media, and translations) will be applied based on the querying user's permissions.
* The WHERE clause in the view definition (t.deleted\_at IS NULL AND t.content\_visibility\_status = 'published') provides a baseline filtering for publicly visible towns. Querying users will only see rows they have access to in *all* joined tables.
* The exposed region\_is\_visible, province\_is\_active, town\_type\_is\_active, media\_is\_approved flags allow API consumers to further filter if the view's base WHERE clause is made less restrictive on linked entities for flexibility.

**7. Performance & Optimization Notes**

* **Indexing**:
  + Crucial indexes on public.translations: A composite index on (table\_identifier, column\_identifier, language\_code, row\_foreign\_key, translation\_status) or similar, tailored to these joins, is vital. Also, an index on (row\_foreign\_key, language\_code, translation\_status) can be beneficial if table\_identifier and column\_identifier are usually fixed in queries.
  + Ensure all foreign key columns used in joins (t.region\_id, t.province\_id, t.town\_type\_code, t.primary\_media\_id) are indexed in their respective tables.
  + Indexes on code for town\_types\_master (PK) are already in place.
* **Join Strategy**: The INNER JOIN on town\_name\_trans ensures that only towns with at least a translated name (in any language with 'published\_live' status) will appear in the view. Subsequent LEFT JOINs for other translations ensure the town row is not dropped if a secondary translation (like short description) is missing for that language.
* **Materialized View**: If read performance for very common language queries becomes a bottleneck, this view could be a candidate for conversion to a MATERIALIZED VIEW, refreshed periodically.
* **Filtering**: Queries against the view should always include WHERE language\_code = 'desired\_lang'. Additional filters should leverage indexed columns from the base tables where possible.
* **Column Selection**: Select only necessary columns when querying the view to reduce data transfer.

**8. Assumptions & Dependencies**

* Assumes public.translations.translation\_status column exists and 'published\_live' is the correct value for visible translations.
* Assumes public.media.media\_status column exists and 'approved' is the correct value for visible media.
* Assumes specific keys like 'thumbnail\_S' exist in public.media.image\_variants\_json.
* Assumes table\_identifier for media translations is 'media' and column\_identifier is 'alt\_text'.
* The base tables (regions, provinces, town\_types\_master) have is\_active or equivalent visibility logic (deleted\_at, content\_visibility\_status) which is reflected in the view's boolean visibility flags for these linked entities.

**9. Next-Action Checklist**

* 🟠 **Finalize** translation\_status **and** media\_status **values**: Confirm the exact values to use for 'published' or 'approved' statuses in translations and media tables.
* 🟠 **Confirm** image\_variants\_json **keys**: Verify the specific keys (e.g., thumbnail\_S) that will be reliably present in media.image\_variants\_json.
* 🟠 **Refine Join Conditions for Linked Entity Visibility**: Decide whether to strictly filter out list items in the view's main WHERE clause if linked entities (like region or province) are not active/visible, or to rely on the exposed boolean flags for API-level filtering (current approach leans towards flags for flexibility).
* 🟠 **Test View Definition**: Create and thoroughly test the view with sample data, covering various languages and missing translations/media.
* 🟢 **Create Indexes**: Ensure optimal indexes exist on public.translations and other base tables supporting the view's joins.
* 🟢 **Integrate into API**: Update the GET /towns API endpoint logic to query this view.
* 🟢 **Monitor Performance**: After deployment, monitor the performance of queries using this view and consider materialization if necessary.
* 🟢 **Document for API Consumers**: Clearly document the view's output columns and how to use it effectively (especially language\_code filtering).