<https://gemini.google.com/u/1/app/bf28f389cd093e55?is_sa=1&android-min-version=301356232&ios-min-version=322.0&campaign_id=bkws&utm_source=google&utm_medium=cpc&utm_campaign=2024enUS_gemfeb&pt=9008&mt=8&ct=p-growth-sem-bkws>

<https://gemini.google.com/u/1/app/34bd37922031b95a>

<https://gemini.google.com/u/1/app/21b8e85b3e611787>

### 3\. Updated Production-Ready Specification

4.13 Dietary Option Tags Master Table (Version 1.4)

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

This document details the structure, purpose, and considerations for the `dietary\_option\_tags\_master` table. This table is crucial for pilgrims with specific dietary needs. Version 1.4 updates the RLS policies to align with the platform-wide security and authentication strategy using the `public.has\_role()` helper function.

### 1\. Purpose & Primary Use-Cases

The `dietary\_option\_tags\_master` table provides a standardized list of dietary options that food establishments might cater to (e.g., "Vegetarian Friendly," "Gluten-Free Options," "Vegan Options Available"). Its purpose is to enable accurate tagging of establishments, allowing pilgrims to easily filter and identify places that meet their dietary requirements, and to ensure multilingual consistency for these important attributes.

Key user-story touchpoints:

- Pilgrim: Filtering food establishments by specific dietary needs (e.g., finding vegetarian or gluten-free meals). (Story A4)

- Pilgrim: Quickly identifying if an establishment is likely to accommodate their dietary restrictions.

- Admin/Content Manager: Tagging food establishments with relevant dietary options from a managed list, and managing the lifecycle of these tags (active/inactive).

- System/UI: Populating filter options for dietary needs and displaying these options clearly on establishment listings, potentially with icons, using active tags.

### 2\. Schema (Markdown Table)

\*(No change to column structure from Version 1.3)\*

| column | data\_type | constraints | description |

| id | `integer` | Primary Key (Generated as identity always) | Unique identifier for the dietary option tag. |

| code | `text` | Unique, Not Null, CHECK (length(code) > 0 AND length(code) &lt;= 50 AND code ~ '^[a-z0-9\_]+$') | Short, stable, machine-readable code (e.g., 'vegetarian\_friendly', 'gluten\_free'). Snake\_case. |

| label | `text` | Not Null, CHECK (length(label) > 0 AND length(label) &lt;= 100) | Human-readable name in the primary reference language (English) for UI display and as a base for translation. Translatable. |

| description | `text` | Nullable | Optional description in the primary reference language (English) clarifying the scope of the dietary option. Translatable. |

| icon\_identifier | `text` | Nullable, CHECK (icon\_identifier IS NULL OR length(icon\_identifier) &lt;= 100) | Name, class, or path for a UI icon associated with this dietary option. |

| sort\_order | `integer` | Not Null, Default 0 | Determines the display order in UI lists or filters. |

| is\_active | `boolean` | Not Null, Default true | True if the tag is active and available for use; false if retired/archived. Important for array FK validation. |

| created\_at | `timestamp with time zone` | Not Null, Default `now()` | Timestamp of record creation. |

| updated\_at | `timestamp with time zone` | Not Null, Default `now()` | Timestamp of last update (auto-updated by trigger). |

| created\_by\_profile\_id | `uuid` | Nullable, Foreign Key to `public.profiles(id)` ON DELETE SET NULL | Profile ID of the user/admin who created this dietary option tag record. |

| updated\_by\_profile\_id | `uuid` | Nullable, Foreign Key to `public.profiles(id)` ON DELETE SET NULL | Profile ID of the user/admin who last updated this dietary option tag record. |

### 3\. PostgreSQL DDL

\*(DDL for table structure, comments, triggers, and indexes remain the same as Version 1.3. Only the version in the table comment changes.)\*

SQL

```

-- Assumes public.profiles table exists

-- Assumes public.set\_current\_timestamp\_updated\_at() function exists

-- Assumes public.cleanup\_related\_translations(TEXT, TEXT) function and specific per-table wrapper exist

CREATE TABLE public.dietary\_option\_tags\_master (

id INTEGER GENERATED ALWAYS AS IDENTITY PRIMARY KEY,

code TEXT UNIQUE NOT NULL CHECK (length(code) > 0 AND length(code) <= 50 AND code ~ '^[a-z0-9\_]+$'),

label TEXT NOT NULL CHECK (length(label) > 0 AND length(label) <= 100),

description TEXT NULL,

icon\_identifier TEXT NULL CHECK (icon\_identifier IS NULL OR length(icon\_identifier) <= 100),

sort\_order INTEGER NOT NULL DEFAULT 0,

is\_active BOOLEAN NOT NULL DEFAULT true,

created\_at TIMESTAMPTZ NOT NULL DEFAULT now(),

updated\_at TIMESTAMPTZ NOT NULL DEFAULT now(),

created\_by\_profile\_id UUID NULL REFERENCES public.profiles(id) ON DELETE SET NULL,

updated\_by\_profile\_id UUID NULL REFERENCES public.profiles(id) ON DELETE SET NULL

);

COMMENT ON TABLE public.dietary\_option\_tags\_master IS 'Master list of dietary option tags (e.g., vegetarian, gluten-free). Replaces dietary\_option\_tag\_enum. Version 1.4';

-- Column comments from Version 1.3 remain unchanged. E.g.:

COMMENT ON COLUMN public.dietary\_option\_tags\_master.label IS 'Human-readable name in the primary reference language (English) for UI and as a base for translation. Translatable via the ''translations'' table. Max 100 chars.';

COMMENT ON COLUMN public.dietary\_option\_tags\_master.is\_active IS 'True if the tag is active and available for use; false if retired. Important for array FK validation.';

COMMENT ON COLUMN public.dietary\_option\_tags\_master.created\_by\_profile\_id IS 'Profile ID of the user/admin who created this record.';

COMMENT ON COLUMN public.dietary\_option\_tags\_master.updated\_by\_profile\_id IS 'Profile ID of the user/admin who last updated this record.';

-- Indexes (including idx\_dotm\_label from previous update)

CREATE INDEX IF NOT EXISTS idx\_dotm\_is\_active ON public.dietary\_option\_tags\_master(is\_active);

CREATE INDEX IF NOT EXISTS idx\_dotm\_sort\_order ON public.dietary\_option\_tags\_master(sort\_order);

CREATE INDEX IF NOT EXISTS idx\_dotm\_label ON public.dietary\_option\_tags\_master(label);

CREATE INDEX IF NOT EXISTS idx\_dotm\_created\_by\_profile\_id ON public.dietary\_option\_tags\_master(created\_by\_profile\_id) WHERE created\_by\_profile\_id IS NOT NULL;

CREATE INDEX IF NOT EXISTS idx\_dotm\_updated\_by\_profile\_id ON public.dietary\_option\_tags\_master(updated\_by\_profile\_id) WHERE updated\_by\_profile\_id IS NOT NULL;

-- Trigger for updated\_at

CREATE TRIGGER trigger\_dietary\_option\_tags\_master\_set\_updated\_at

BEFORE UPDATE ON public.dietary\_option\_tags\_master

FOR EACH ROW

EXECUTE FUNCTION public.set\_current\_timestamp\_updated\_at();

COMMENT ON TRIGGER trigger\_dietary\_option\_tags\_master\_set\_updated\_at ON public.dietary\_option\_tags\_master IS 'Trigger to automatically update updated\_at timestamp on row modification.';

-- Trigger for orphan translation cleanup

CREATE OR REPLACE FUNCTION public.cleanup\_dietary\_option\_tags\_master\_translations()

RETURNS TRIGGER AS $$

BEGIN

IF TG\_OP = 'DELETE' THEN

DELETE FROM public.translations

WHERE table\_identifier = 'dietary\_option\_tags\_master'

AND row\_foreign\_key = OLD.id::TEXT;

END IF;

RETURN OLD;

END;

$$ LANGUAGE plpgsql SECURITY DEFINER;

CREATE TRIGGER trigger\_cleanup\_dietary\_option\_tags\_master\_translations

AFTER DELETE ON public.dietary\_option\_tags\_master

FOR EACH ROW

EXECUTE FUNCTION public.cleanup\_dietary\_option\_tags\_master\_translations();

COMMENT ON TRIGGER trigger\_cleanup\_dietary\_option\_tags\_master\_translations ON public.dietary\_option\_tags\_master IS 'Cleans up orphaned translations from public.translations when a dietary\_option\_tags\_master record is deleted.';

-- Initial Data Example (ensure created\_by\_profile\_id and updated\_by\_profile\_id are set appropriately for seed data)

INSERT INTO public.dietary\_option\_tags\_master (code, label, icon\_identifier, sort\_order, description, is\_active, created\_by\_profile\_id, updated\_by\_profile\_id) VALUES

('vegetarian\_friendly', 'Vegetarian Friendly', 'icon-vegetarian', 10, 'Offers dishes suitable for vegetarians (no meat, poultry, or fish). May include dairy or eggs.', true, NULL, NULL),

('vegan\_options\_available', 'Vegan Options Available', 'icon-vegan', 20, 'Offers dishes suitable for vegans (no animal products, including dairy, eggs, and honey).', true, NULL, NULL),

-- ... other seed data from V1.3 ...

('organic\_options', 'Organic Options', 'icon-organic', 110, 'Offers dishes prepared with certified organic ingredients.', true, NULL, NULL);

```

### 4\. JSON Schema Mirror

\*(No change from Version 1.3)\*

JSON

```

{

"title": "dietary\_option\_tag\_master",

"description": "Master list of dietary option tags (e.g., vegetarian, gluten-free). Version 1.4",

"type": "object",

"properties": {

"id": { /\* ... \*/ },

"code": { /\* ... \*/ },

"label": { /\* ... \*/ },

"description": { /\* ... \*/ },

"icon\_identifier": { /\* ... \*/ },

"sort\_order": { /\* ... \*/ },

"is\_active": { /\* ... \*/ },

"created\_at": { /\* ... \*/ },

"updated\_at": { /\* ... \*/ },

"created\_by\_profile\_id": { /\* ... \*/ },

"updated\_by\_profile\_id": { /\* ... \*/ }

},

"required": [ /\* ... \*/ ]

}

```

### 5\. Relationships & Integrity

\*(No change from Version 1.3)\*

- Primary Key: `id` (INTEGER)

- Unique Constraint: `code` must be unique.

- Foreign Key References FROM other tables:

- `food\_water\_sources\_details.dietary\_option\_tag\_ids` (an `INTEGER[]` column) will contain IDs that reference `dietary\_option\_tags\_master.id`.

- Foreign Key References TO other tables:

- `created\_by\_profile\_id` REFERENCES `public.profiles(id)` ON DELETE SET NULL.

- `updated\_by\_profile\_id` REFERENCES `public.profiles(id)` ON DELETE SET NULL.

- Data Integrity Notes:

- 🔴 CRITICAL: Database triggers on referencing tables (e.g., `food\_water\_sources\_details`) \*must\* be implemented to ensure all IDs in their `dietary\_option\_tag\_ids` array correspond to valid `id` entries in `dietary\_option\_tags\_master` AND that `dietary\_option\_tags\_master.is\_active = true`.

- Retiring a dietary option tag should be done by setting `is\_active = false`.

### 6\. Multilingual Strategy

\*(No change from Version 1.3)\*

- Translatable Fields: `label`, `description`.

- Translation Management: Via `public.translations` table and orphan cleanup trigger.

### 7\. Role-Based Workflow & RLS Notes

\*(This section is updated to reflect the new auth strategy)\*

- Content Management: This table is typically managed by users with the `admin\_platform` role. `regional\_content\_manager` might also manage these if dietary options are considered part of regional content characteristics relevant to food sources.

- Lifecycle: Dietary option tags are made inactive by setting `is\_active = false`. Physical deletion should be rare.

- RLS Policies (Assumes `public.has\_role(TEXT)` helper function exists):

- Public Users (Read-Only on active tags):

SQL

```

-- Name: Allow public read access to active dietary option tags

-- Target: dietary\_option\_tags\_master

-- Operation: SELECT

-- Role(s): anon, authenticated

CREATE POLICY "Allow public read access to active dietary option tags"

ON public.dietary\_option\_tags\_master FOR SELECT

USING (is\_active = true);

```

- Platform Administrators / Content Managers (Full Control):

SQL

```

-- Name: Allow platform content administrators to manage dietary option tags

-- Target: dietary\_option\_tags\_master

-- Operation: ALL

-- Role(s): admin\_platform, regional\_content\_manager

CREATE POLICY "Allow platform content administrators to manage dietary option tags"

ON public.dietary\_option\_tags\_master FOR ALL

USING (

auth.role() = 'authenticated' AND

(public.has\_role('admin\_platform') OR public.has\_role('regional\_content\_manager'))

) WITH CHECK (

auth.role() = 'authenticated' AND

(public.has\_role('admin\_platform') OR public.has\_role('regional\_content\_manager'))

);

```

- Enable RLS:

SQL

```

ALTER TABLE public.dietary\_option\_tags\_master ENABLE ROW LEVEL SECURITY;

```

- Notes: RLS must filter by `is\_active = true` for general read access. Array FK validation triggers on referencing tables must also check `is\_active = true`.

### 8\. ENUM vs Lookup Discussion

\*(No change from Version 1.3)\*

- 🟢 Decision: Correctly a lookup table.

- Reasoning: Crucial user filters, translatable names/descriptions, icons, consistency, auditability, lifecycle.

### 9\. UI/UX Enablement

\*(No change from Version 1.3)\*

- `label` (translated): For filter labels, display tags.

- `icon\_identifier`: To show icons.

- `description` (translated): For tooltips.

- `sort\_order`: To list common options first.

- `is\_active`: UI should only use active tags.

### 10\. Key Considerations & Definitions

\*(No change from Version 1.3)\*

- Clarity is Key: Names/descriptions must be unambiguous.

- Scope: `description` can clarify "dedicated prep" vs. "options available."

- "Warning" type tags: UI needs to communicate clearly.

- Array FK Integrity: Referencing table's trigger logic must check `is\_active = true`.

### 11\. Scalability & Future-Proofing

\*(No change from Version 1.3)\*

- Manageable List: Number of common dietary options is manageable.

- Flexibility: Easy to add new preferences.

- Audit Fields & `is\_active` flag: Robust.

### 12\. Next-Action Checklist

\*(No DDL changes required for this specific update beyond what was in V1.3, the main change is to the RLS policy definition in this document)\*

- 🔴 Verify/Implement RLS Helper: Ensure the `public.has\_role(TEXT)` helper function is correctly implemented and available.

- 🔴 Apply RLS Policies: Implement and thoroughly test the updated RLS policies.

- 🔴 Initial Population/Seed Data: Ensure `created\_by\_profile\_id`/`updated\_by\_profile\_id` are correctly set for seed data.

- 🟠 Array FK Integrity: Ensure validation trigger on `food\_water\_sources\_details` (for `dietary\_option\_tag\_ids`) checks `is\_active = true` in this table.

- 🟢 Icon Set Development: Coordinate with UI/UX team.

- 🟢 Translation Entries: Prepare/verify English entries for `label` and `description` in `public.translations`.