<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

4.26 Amenities Master Table (Version 1.5)

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

This document details the structure, purpose, and considerations for the `amenities\_master` table. Version 1.5 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 `amenities\_master` table defines a standardized list of all possible amenities that accommodations (and potentially other types of waypoints) can offer (e.g., "Wi-Fi Internet," "Guest Kitchen Access," "Pets Permitted," "Secure Bicycle Storage"). Its purpose is to ensure data consistency, support robust filtering by amenities, allow for multilingual display of amenity names, and associate icons with each amenity for a richer user interface.

Key user-story touchpoints:

- Pilgrim (Anna): Filtering accommodations by available amenities (e.g., finding places with a kitchen or laundry facilities). (Story A3)

- Pilgrim (Anna): Quickly seeing available amenities (with icons) on an accommodation's listing.

- Accommodation Host (Marco): Selecting the amenities their B&amp;B offers from a predefined list. (Story B1)

- Admin/Content Manager: Managing the global list of recognized amenities, including their names, icons, categories, and active status.

- System/UI: Populating filter options for amenities and displaying them clearly in accommodation details using active amenities.

### 2\. Schema (Markdown Table)

\*(No change to column structure from Version 1.4)\*

| column | data\_type | constraints | description |

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

| amenity\_code | `text` | Unique, Not Null, CHECK (length(amenity\_code) > 0 AND length(amenity\_code) &lt;= 50 AND amenity\_code ~ '^[A-Z0-9\_]+$') | Short, stable, machine-readable code (e.g., "WIFI\_INTERNET", "KITCHEN\_ACCESS"). Uppercase snake\_case. |

| name | `text` | Not Null, CHECK (length(name) > 0 AND length(name) &lt;= 100) | Human-readable name in the primary reference language (English). Translatable via the `translations` table. |

| description | `text` | Nullable | Optional further description of the amenity in the primary reference language (English). Translatable via the `translations` table. |

| 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 amenity. |

| amenity\_category | `text` | Nullable, CHECK (amenity\_category IS NULL OR length(amenity\_category) &lt;= 50) | Broad category for grouping amenities in the primary reference language (English) (e.g., "General", "Room Feature"). Translatable. |

| is\_common\_pilgrim\_need | `boolean` | Not Null, Default false | Flags amenities particularly sought after by pilgrims, can be used for highlighting or default filter suggestions. |

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

| is\_active | `boolean` | Not Null, Default true | True if the amenity is active and available for use; false if retired/archived. Important for RLS and FK integrity checks. |

| 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 amenity 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 amenity record. |

### 3\. PostgreSQL DDL

\*(DDL for table structure, comments, triggers, and indexes remain the same as Version 1.4. 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.amenities\_master (

id INTEGER GENERATED ALWAYS AS IDENTITY PRIMARY KEY,

amenity\_code TEXT UNIQUE NOT NULL CHECK (length(amenity\_code) > 0 AND length(amenity\_code) <= 50 AND amenity\_code ~ '^[A-Z0-9\_]+$'),

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

description TEXT NULL,

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

amenity\_category TEXT NULL CHECK (amenity\_category IS NULL OR length(amenity\_category) <= 50),

is\_common\_pilgrim\_need BOOLEAN NOT NULL DEFAULT FALSE,

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.amenities\_master IS 'Master list of all possible amenities with standardized names, codes, and icon references. Version 1.5';

-- Column comments from Version 1.4 remain unchanged. E.g.:

COMMENT ON COLUMN public.amenities\_master.name IS 'Human-readable name in the primary reference language (English). Translatable via the ''translations'' table. Max 100 chars.';

COMMENT ON COLUMN public.amenities\_master.amenity\_category IS 'Broad category for grouping amenities in the primary reference language (English) (e.g., "General", "Room Feature"). Translatable via the ''translations'' table. Max 50 chars.';

COMMENT ON COLUMN public.amenities\_master.is\_active IS 'True if the amenity is active and available for use; false if retired. Important for RLS and FK integrity checks.';

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

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

-- Indexes (including idx\_am\_name from previous update)

CREATE INDEX IF NOT EXISTS idx\_am\_is\_active ON public.amenities\_master(is\_active);

CREATE INDEX IF NOT EXISTS idx\_am\_sort\_order ON public.amenities\_master(sort\_order);

CREATE INDEX IF NOT EXISTS idx\_am\_amenity\_category ON public.amenities\_master(amenity\_category) WHERE amenity\_category IS NOT NULL;

CREATE INDEX IF NOT EXISTS idx\_am\_is\_common\_pilgrim\_need ON public.amenities\_master(is\_common\_pilgrim\_need);

CREATE INDEX IF NOT EXISTS idx\_am\_name ON public.amenities\_master(name);

CREATE INDEX IF NOT EXISTS idx\_am\_created\_by\_profile\_id ON public.amenities\_master(created\_by\_profile\_id) WHERE created\_by\_profile\_id IS NOT NULL;

CREATE INDEX IF NOT EXISTS idx\_am\_updated\_by\_profile\_id ON public.amenities\_master(updated\_by\_profile\_id) WHERE updated\_by\_profile\_id IS NOT NULL;

-- Trigger for updated\_at

CREATE TRIGGER trigger\_amenities\_master\_set\_updated\_at

BEFORE UPDATE ON public.amenities\_master

FOR EACH ROW

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

COMMENT ON TRIGGER trigger\_amenities\_master\_set\_updated\_at ON public.amenities\_master IS 'Trigger to automatically update updated\_at timestamp on row modification.';

-- Trigger for orphan translation cleanup

CREATE OR REPLACE FUNCTION public.cleanup\_amenities\_master\_translations()

RETURNS TRIGGER AS $$

BEGIN

IF TG\_OP = 'DELETE' THEN

DELETE FROM public.translations

WHERE table\_identifier = 'amenities\_master'

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

END IF;

RETURN OLD;

END;

$$ LANGUAGE plpgsql SECURITY DEFINER;

CREATE TRIGGER trigger\_cleanup\_amenities\_master\_translations

AFTER DELETE ON public.amenities\_master

FOR EACH ROW

EXECUTE FUNCTION public.cleanup\_amenities\_master\_translations();

COMMENT ON TRIGGER trigger\_cleanup\_amenities\_master\_translations ON public.amenities\_master IS 'Cleans up orphaned translations from public.translations when an amenities\_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.amenities\_master (amenity\_code, name, icon\_identifier, amenity\_category, is\_common\_pilgrim\_need, sort\_order, description, is\_active, created\_by\_profile\_id, updated\_by\_profile\_id) VALUES

('WIFI\_INTERNET', 'Wi-Fi Internet', 'wifi', 'Connectivity', true, 10, 'Access to wireless internet.', true, NULL, NULL),

('KITCHEN\_ACCESS', 'Guest Kitchen Access', 'kitchen', 'Food & Drink', true, 20, 'Guests can use kitchen facilities to prepare their own meals.', true, NULL, NULL),

-- ... other seed data from V1.4 ...

('PARKING\_ONSITE', 'On-site Parking', 'local\_parking', 'Services', false, 140, 'Parking available at the accommodation premises.', true, NULL, NULL);

```

### 4\. JSON Schema Mirror

\*(No change from Version 1.4)\*

JSON

```

{

"title": "amenity\_master",

"description": "Master list of all possible amenities with standardized names, codes, and icon references. Version 1.5",

"type": "object",

"properties": {

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

"amenity\_code": { /\* ... \*/ },

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

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

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

"amenity\_category": { /\* ... \*/ },

"is\_common\_pilgrim\_need": { /\* ... \*/ },

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

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

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

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

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

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

},

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

}

```

### 5\. Relationships & Integrity

\*(No change from Version 1.4)\*

- Primary Key: `id` (INTEGER)

- Unique Constraint: `amenity\_code` must be unique.

- Foreign Key References FROM other tables:

- `accommodation\_amenities.amenity\_id` REFERENCES `public.amenities\_master(id)` (ON DELETE RESTRICT recommended on junction).

- 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: Retiring an amenity by `is\_active = false`.

### 6\. Multilingual Strategy

\*(No change from Version 1.4)\*

- Translatable Fields: `name`, `description`, `amenity\_category`.

- 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 be granted rights if amenities are considered part of regional content characteristics.

- Lifecycle: Amenities are made inactive by setting `is\_active = false`. Physical deletion is restricted by FK from `accommodation\_amenities` if an amenity is in use.

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

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

SQL

```

-- Name: Allow public read access to active amenities master list

-- Target: amenities\_master

-- Operation: SELECT

-- Role(s): anon, authenticated

CREATE POLICY "Allow public read access to active amenities master list"

ON public.amenities\_master FOR SELECT

USING (is\_active = true);

```

- Platform Administrators / Content Managers (Full Control):

SQL

```

-- Name: Allow platform content administrators to manage amenities master list

-- Target: amenities\_master

-- Operation: ALL

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

CREATE POLICY "Allow platform content administrators to manage amenities master list"

ON public.amenities\_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.amenities\_master ENABLE ROW LEVEL SECURITY;

```

- Notes: RLS must filter by `is\_active = true` for general read access.

### 8\. ENUM vs Lookup Discussion

\*(No change from Version 1.4)\*

- 🟢 Decision: Correctly a lookup table.

- Reasoning: Metadata richness, auditability, lifecycle.

### 9\. UI/UX Enablement

\*(No change from Version 1.4)\*

- `name` (translated): For display in lists and filters.

- `icon\_identifier`: For icons.

- `is\_common\_pilgrim\_need`: To highlight options.

- `amenity\_category` (translated): For grouping.

- `sort\_order`: For consistent display.

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

### 10\. Key Considerations & Definitions

\*(No change from Version 1.4)\*

- `amenity\_code`: Unique, stable (UPPER\_SNAKE\_CASE).

- Icon Management: Frontend library needed.

- Categorization (`amenity\_category`): Define useful categories; consider promotion to master in V2+.

### 11\. Scalability & Future-Proofing

\*(No change from Version 1.4)\*

- Lookup Table Structure: Highly scalable.

- Flexibility: Easy to add new attributes.

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

### 12\. Next-Action Checklist

\*(No DDL changes required for this specific update beyond what was in V1.4, 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.

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

- 🟢 Translation Entries: Prepare/verify English entries for `name`, `description`, and `amenity\_category` in `public.translations`.