

Current table schema.

1. Categories

```
create table public.categories (
    id uuid not null default gen_random_uuid (),
    location_id uuid not null,
    name text not null,
    created_at timestamp with time zone not null default now(),
    updated_at timestamp with time zone not null default now(),
    created_by uuid null,
    constraint categories_pkey primary key (id),
    constraint categories_created_by_fkey foreign KEY (created_by) references auth.users
(id),
    constraint categories_location_id_fkey foreign KEY (location_id) references locations
(id) on delete RESTRICT
) TABLESPACE pg_default;
```

```
create unique INDEX IF not exists ux_categories_location_name on public.categories
using btree (location_id, name) TABLESPACE pg_default;
```

```
create index IF not exists ix_categories_location_id on public.categories using btree
(location_id) TABLESPACE pg_default;
```

```
create trigger trg_categories_updated_at BEFORE
update on categories for EACH row
execute FUNCTION set_updated_at();
```

2. Items

```
create table public.items (
    id uuid not null default gen_random_uuid (),
    location_id uuid not null,
    category_id uuid null,
    unit_id uuid null,
    name text not null,
    barcode text null,
```

```
sale_price numeric(15, 2) not null default 0,  
created_at timestamp with time zone not null default now(),  
updated_at timestamp with time zone not null default now(),  
created_by uuid null,  
constraint items_pkey primary key (id),  
constraint items_category_id_fkey foreign KEY (category_id) references categories (id)  
on delete set null,  
constraint items_created_by_fkey foreign KEY (created_by) references auth.users (id),  
constraint items_location_id_fkey foreign KEY (location_id) references locations (id)  
on delete RESTRICT,  
constraint items_unit_id_fkey foreign KEY (unit_id) references units (id) on delete set  
null  
) TABLESPACE pg_default;
```

create unique INDEX IF not exists ux_items_location_barcode on public.items using
btree (location_id, barcode) TABLESPACE pg_default

where

(barcode is not null);

create index IF not exists ix_items_location_id on public.items using btree (location_id)
TABLESPACE pg_default;

create trigger trg_items_updated_at BEFORE
update on items for EACH row
execute FUNCTION set_updated_at();

3. Locations

```
create table public.locations (  
    id uuid not null default gen_random_uuid (),  
    name text not null,
```

```

created_at timestamp with time zone not null default now(),
updated_at timestamp with time zone not null default now(),
created_by uuid null,
constraint locations_pkey primary key (id),
constraint locations_created_by_fkey foreign KEY (created_by) references auth.users
(id)
) TABLESPACE pg_default;

```

```

create unique INDEX IF not exists ux_locations_name on public.locations using btree
(name) TABLESPACE pg_default;

```

```

create trigger trg_locations_updated_at BEFORE
update on locations for EACH row
execute FUNCTION set_updated_at ();

```

4. Sales_invoice_lines

```

create table public.sales_invoice_lines (
    id uuid not null default gen_random_uuid (),
    location_id uuid not null,
    sales_invoice_id uuid not null,
    item_id uuid not null,
    qty numeric(15, 3) not null default 1,
    unit_price numeric(15, 2) not null default 0,
    line_total numeric(15, 2) not null default 0,
    created_at timestamp with time zone not null default now(),
    updated_at timestamp with time zone not null default now(),
    created_by uuid null,
    constraint sales_invoice_lines_pkey primary key (id),
    constraint sales_invoice_lines_sales_invoice_id_fkey foreign KEY (sales_invoice_id)
references sales_invoices (id) on delete CASCADE,
    constraint sales_invoice_lines_created_by_fkey foreign KEY (created_by) references
auth.users (id),
    constraint sales_invoice_lines_item_id_fkey foreign KEY (item_id) references items
(id) on delete RESTRICT,
    constraint sales_invoice_lines_location_id_fkey foreign KEY (location_id) references
locations (id) on delete RESTRICT,
    constraint chk_unit_price_non_negative check ((unit_price >= (0)::numeric)),
    constraint chk_line_total_non_negative check ((line_total >= (0)::numeric)),

```

```

constraint chk_qty_non_negative check ((qty >= (0)::numeric))
) TABLESPACE pg_default;

create index IF not exists ix_sales_lines_invoice_id on public.sales_invoice_lines using
btree (sales_invoice_id) TABLESPACE pg_default;

create index IF not exists ix_sales_lines_location_id on public.sales_invoice_lines using
btree (location_id) TABLESPACE pg_default;

create trigger trg_sales_invoice_lines_updated_at BEFORE
update on sales_invoice_lines for EACH row
execute FUNCTION set_updated_at ();

```

5. Sales_invoices

```

create table public.sales_invoices (
    id uuid not null default gen_random_uuid (),
    location_id uuid not null,
    invoice_number text not null,
    invoice_date timestamp with time zone not null default now(),
    subtotal numeric(15, 2) not null default 0,
    discount_total numeric(15, 2) not null default 0,
    grand_total numeric(15, 2) not null default 0,
    created_at timestamp with time zone not null default now(),
    updated_at timestamp with time zone not null default now(),
    created_by uuid null,
    constraint sales_invoices_pkey primary key (id),
    constraint sales_invoices_created_by_fkey foreign KEY (created_by) references
    auth.users (id),
    constraint sales_invoices_location_id_fkey foreign KEY (location_id) references
    locations (id) on delete RESTRICT
) TABLESPACE pg_default;

```

```

create unique INDEX IF not exists ux_sales_invoices_location_invoice_number on
public.sales_invoices using btree (location_id, invoice_number) TABLESPACE
pg_default;

```

```

create index IF not exists ix_sales_invoices_location_id on public.sales_invoices using
btree (location_id) TABLESPACE pg_default;

```

```
create trigger trg_sales_invoices_updated_at BEFORE
update on sales_invoices for EACH row
execute FUNCTION set_updated_at();
```

6. Units

```
create table public.units (
    id uuid not null default gen_random_uuid (),
    location_id uuid not null,
    name text not null,
    symbol text null,
    created_at timestamp with time zone not null default now(),
    updated_at timestamp with time zone not null default now(),
    created_by uuid null,
    constraint units_pkey primary key (id),
    constraint units_created_by_fkey foreign KEY (created_by) references auth.users (id),
    constraint units_location_id_fkey foreign KEY (location_id) references locations (id) on
    delete RESTRICT
) TABLESPACE pg_default;
```

```
create unique INDEX IF not exists ux_units_location_name on public.units using btree
(location_id, name) TABLESPACE pg_default;
```

```
create index IF not exists ix_units_location_id on public.units using btree (location_id)
TABLESPACE pg_default;
```

```
create trigger trg_units_updated_at BEFORE
update on units for EACH row
execute FUNCTION set_updated_at();
```

7. User_profiles

```
create table public.user_profiles (
    user_id uuid not null,
    location_id uuid not null,
    role text not null default 'cashier'::text,
    created_at timestamp with time zone not null default now(),
    updated_at timestamp with time zone not null default now(),
    created_by uuid null,
```

```
constraint user_profiles_pkey primary key (user_id),
constraint user_profiles_created_by_fkey foreign KEY (created_by) references
auth.users (id),
constraint user_profiles_location_id_fkey foreign KEY (location_id) references
locations (id) on delete RESTRICT,
constraint user_profiles_user_id_fkey foreign KEY (user_id) references auth.users (id)
on delete CASCADE
) TABLESPACE pg_default;
```

```
create index IF not exists ix_user_profiles_location_id on public.user_profiles using btree
(location_id) TABLESPACE pg_default;
```

```
create trigger trg_user_profiles_updated_at BEFORE
update on user_profiles for EACH row
execute FUNCTION set_updated_at();
```