```
/**
* USERS
* Note: This table contains user data. Users should only be able to view and update their own data.
*/
create table users (
  -- UUID from auth.users
  id uuid references auth.users not null primary key,
  full_name text,
  avatar_url text,
  -- The customer's billing address, stored in JSON format.
  billing_address jsonb,
  -- Stores your customer's payment instruments.
  payment_method jsonb
);
alter table users enable row level security;
create policy "Can view own user data." on users for select using (auth.uid() = id);
create policy "Can update own user data." on users for update using (auth.uid() = id);

/**
* This trigger automatically creates a user entry when a new user signs up via Supabase Auth.
*/
create function public.handle_new_user()
returns trigger as $$
begin
  insert into public.users (id, full_name, avatar_url)
  values (new.id, new.raw_user_meta_data->>'full_name', new.raw_user_meta_data->>'avatar_url');
```

```
  return new;
end;
$$ language plpgsql security definer;
create trigger on_auth_user_created
  after insert on auth.users
  for each row execute procedure public.handle_new_user();

/**
* CUSTOMERS
* Note: this is a private table that contains a mapping of user IDs to Stripe customer IDs.
*/
create table customers (
  -- UUID from auth.users
  id uuid references auth.users not null primary key,
  -- The user's customer ID in Stripe. User must not be able to update this.
  stripe_customer_id text
);
alter table customers enable row level security;
-- No policies as this is a private table that the user must not have access to.

/**
* PRODUCTS
* Note: products are created and managed in Stripe and synced to our DB via Stripe webhooks.
*/
create table products (
  -- Product ID from Stripe, e.g. prod_1234.
```

```
  id text primary key,
  -- Whether the product is currently available for purchase.
  active boolean,
  -- The product's name, meant to be displayable to the customer. Whenever this product is sold via a subscription, name will show up on associated invoice line item descriptions.
  name text,
  -- The product's description, meant to be displayable to the customer. Use this field to optionally store a long form explanation of the product being sold for your own rendering purposes.
  description text,
  -- A URL of the product image in Stripe, meant to be displayable to the customer.
  image text,
  -- Set of key-value pairs, used to store additional information about the object in a structured format.
  metadata jsonb
);
alter table products enable row level security;
create policy "Allow public read-only access." on products for select using (true);

/**
* PRICES
* Note: prices are created and managed in Stripe and synced to our DB via Stripe webhooks.
*/
create type pricing_type as enum ('one_time', 'recurring');
create type pricing_plan_interval as enum ('day', 'week', 'month', 'year');
create table prices (
  -- Price ID from Stripe, e.g. price_1234.
```

```
  id text primary key,
  -- The ID of the prduct that this price belongs to.
  product_id text references products,
  -- Whether the price can be used for new purchases.
  active boolean,
  -- A brief description of the price.
  description text,
  -- The unit amount as a positive integer in the smallest currency unit (e.g., 100 cents for US$1.00 or 100 for ¥100, a zero-decimal currency).
  unit_amount bigint,
  -- Three-letter ISO currency code, in lowercase.
  currency text check (char_length(currency) = 3),
  -- One of `one_time` or `recurring` depending on whether the price is for a one-time purchase or a recurring (subscription) purchase.
  type pricing_type,
  -- The frequency at which a subscription is billed. One of `day`, `week`, `month` or `year`.
  interval pricing_plan_interval,
  -- The number of intervals (specified in the `interval` attribute) between subscription billings. For example, `interval=month` and `interval_count=3` bills every 3 months.
  interval_count integer,
    -- Default number of trial days when subscribing a customer to this price using [`trial_from_plan=true`](https://stripe.com/docs/api#create_subscription-trial_from_plan).
  trial_period_days integer,
  -- Set of key-value pairs, used to store additional information about the object in a structured format.
  metadata jsonb
```

```
);
alter table prices enable row level security;
create policy "Allow public read-only access." on prices for select using (true);


/**
* SUBSCRIPTIONS
* Note: subscriptions are created and managed in Stripe and synced to our DB via Stripe webhooks.
*/
create type subscription_status as enum ('trialing', 'active', 'canceled', 'incomplete', 'incomplete_expired', 'past_due', 'unpaid', 'paused');
create table subscriptions (
  -- Subscription ID from Stripe, e.g. sub_1234.
  id text primary key,
  user_id uuid references auth.users not null,
  -- The status of the subscription object, one of subscription_status type above.
  status subscription_status,
   -- Set of key-value pairs, used to store additional information about the object in a structured format.
  metadata jsonb,
  -- ID of the price that created this subscription.
  price_id text references prices,
   -- Quantity multiplied by the unit amount of the price creates the amount of the subscription. Can be used to charge multiple seats.
  quantity integer,
  -- If true the subscription has been canceled by the user and will be deleted at the end of the billing period.
```

```
  cancel_at_period_end boolean,
  -- Time at which the subscription was created.
  created timestamp with time zone default timezone('utc'::text, now()) not null,
  -- Start of the current period that the subscription has been invoiced for.
  current_period_start timestamp with time zone default timezone('utc'::text, now()) not null,
  -- End of the current period that the subscription has been invoiced for. At the end of this period, a new invoice will be created.
  current_period_end timestamp with time zone default timezone('utc'::text, now()) not null,
  -- If the subscription has ended, the timestamp of the date the subscription ended.
  ended_at timestamp with time zone default timezone('utc'::text, now()),
  -- A date in the future at which the subscription will automatically get canceled.
  cancel_at timestamp with time zone default timezone('utc'::text, now()),
   -- If the subscription has been canceled, the date of that cancellation. If the subscription was canceled with `cancel_at_period_end`, `canceled_at` will still reflect the date of the initial cancellation request, not the end of the subscription period when the subscription is automatically moved to a canceled state.
  canceled_at timestamp with time zone default timezone('utc'::text, now()),
  -- If the subscription has a trial, the beginning of that trial.
  trial_start timestamp with time zone default timezone('utc'::text, now()),
  -- If the subscription has a trial, the end of that trial.
  trial_end timestamp with time zone default timezone('utc'::text, now())
);
alter table subscriptions enable row level security;
create policy "Can only view own subs data." on subscriptions for select using (auth.uid() = user_id);

/**
```

```
 * REALTIME SUBSCRIPTIONS

 * Only allow realtime listening on public tables.

 */

drop publication if exists supabase_realtime;

create publication supabase_realtime for table products, prices;
```