```
--
-- PostgreSQL database dump
--

\restrict BA4A8vVNp25ZcRre0olZleVfrclmRFvu7rvNaZILqcF8Bc88D37llM4yjwfTGIS

-- Dumped from database version 17.7 (bdd1736)
-- Dumped by pg_dump version 17.7 (Ubuntu 17.7-0ubuntu0.25.10.1)

SET statement_timeout = 0;
SET lock_timeout = 0;
SET idle_in_transaction_session_timeout = 0;
SET transaction_timeout = 0;
SET client_encoding = 'UTF8';
SET standard_conforming_strings = on;
SELECT pg_catalog.set_config('search_path', '', false);
SET check_function_bodies = false;
SET xmloption = content;
SET client_min_messages = warning;
SET row_security = off;

SET default_tablespace = '';

SET default_table_access_method = heap;

--
-- Name: admin_sessions; Type: TABLE; Schema: public; Owner: neondb_owner
--

CREATE TABLE public.admin_sessions (
    id uuid DEFAULT public.uuid_generate_v4() NOT NULL,
    user_id uuid NOT NULL,
    token_hash text NOT NULL,
    expires_at timestamp with time zone NOT NULL,
    created_at timestamp with time zone DEFAULT now(),
    last_used_at timestamp with time zone DEFAULT now(),
    is_revoked boolean DEFAULT false
);


ALTER TABLE public.admin_sessions OWNER TO neondb_owner;

--
-- Name: admin_users; Type: TABLE; Schema: public; Owner: neondb_owner
--

CREATE TABLE public.admin_users (
    id uuid DEFAULT gen_random_uuid() NOT NULL,
    username character varying(50) NOT NULL,
    password_hash text NOT NULL,
    role character varying(20) DEFAULT 'admin'::character varying,
    is_active boolean DEFAULT true,
    created_at timestamp with time zone DEFAULT now(),
    updated_at timestamp with time zone DEFAULT now(),
    CONSTRAINT admin_users_role_check CHECK (((role)::text = ANY
((ARRAY['admin'::character varying, 'superadmin'::character varying])::text[])))
);
```

```
  ALTER TABLE public.admin_users OWNER TO neondb_owner;

  --
  -- Name: TABLE admin_users; Type: COMMENT; Schema: public; Owner: neondb_owner
  --

  COMMENT ON TABLE public.admin_users IS 'Admin users for JWT-based authentication
to the Admin Dashboard.';


  --
  -- Name: app_config; Type: TABLE; Schema: public; Owner: neondb_owner
  --

  CREATE TABLE public.app_config (
      id uuid DEFAULT gen_random_uuid() NOT NULL,
      key character varying(100) NOT NULL,
      value text NOT NULL,
      type character varying(20) DEFAULT 'string'::character varying,
      category character varying(50) DEFAULT 'general'::character varying,
      description text,
      is_public boolean DEFAULT false,
      created_at timestamp with time zone DEFAULT now(),
      updated_at timestamp with time zone DEFAULT now(),
      CONSTRAINT app_config_type_check CHECK (((type)::text = ANY
((ARRAY['string'::character varying, 'number'::character varying,
'boolean'::character varying, 'json'::character varying, 'color'::character
varying])::text[])))
  );


  ALTER TABLE public.app_config OWNER TO neondb_owner;

  --
  -- Name: TABLE app_config; Type: COMMENT; Schema: public; Owner: neondb_owner
  --

  COMMENT ON TABLE public.app_config IS 'Global application configuration key-value
store. Single-tenant architecture.';


  --
  -- Name: app_messages; Type: TABLE; Schema: public; Owner: neondb_owner
  --

  CREATE TABLE public.app_messages (
      id uuid DEFAULT gen_random_uuid() NOT NULL,
      code character varying(50) NOT NULL,
      lang character varying(10) DEFAULT 'es'::character varying NOT NULL,
      message text NOT NULL,
      created_at timestamp with time zone DEFAULT now(),
      updated_at timestamp with time zone DEFAULT now()
  );


  ALTER TABLE public.app_messages OWNER TO neondb_owner;

  --
  -- Name: TABLE app_messages; Type: COMMENT; Schema: public; Owner: neondb_owner
  --

  COMMENT ON TABLE public.app_messages IS 'Internationalization (i18n) message
repository. Single-tenant architecture.';


  --
  -- Name: audit_logs; Type: TABLE; Schema: public; Owner: neondb_owner
  --

  CREATE TABLE public.audit_logs (
      id uuid DEFAULT public.uuid_generate_v4() NOT NULL,
      table_name text NOT NULL,
      record_id uuid NOT NULL,
      action public.audit_action NOT NULL,
      old_values jsonb,
      new_values jsonb,
      performed_by text DEFAULT 'system'::text,
      ip_address inet,
      created_at timestamp with time zone DEFAULT now(),
      event_type text,
      event_data jsonb
  );


  ALTER TABLE public.audit_logs OWNER TO neondb_owner;

  --
  -- Name: TABLE audit_logs; Type: COMMENT; Schema: public; Owner: neondb_owner
  --

  COMMENT ON TABLE public.audit_logs IS 'Audit trail for all critical operations.
Tracks INSERT, UPDATE, SOFT_DELETE, HARD_DELETE, and security events.';


  --
  -- Name: bookings; Type: TABLE; Schema: public; Owner: neondb_owner
  --

  CREATE TABLE public.bookings (
      id uuid DEFAULT public.uuid_generate_v4() NOT NULL,
      user_id uuid NOT NULL,
      provider_id uuid NOT NULL,
      service_id uuid,
      start_time timestamp with time zone NOT NULL,
      end_time timestamp with time zone NOT NULL,
      status public.booking_status DEFAULT 'confirmed'::public.booking_status,
      gcal_event_id text,
      notes text,
      created_at timestamp with time zone DEFAULT now(),
      updated_at timestamp with time zone DEFAULT now(),
      deleted_at timestamp with time zone,
      reminder_1_sent_at timestamp with time zone,
      reminder_2_sent_at timestamp with time zone,
      CONSTRAINT valid_booking_time CHECK ((end_time > start_time))
  );


  ALTER TABLE public.bookings OWNER TO neondb_owner;
  --
  -- Name: TABLE bookings; Type: COMMENT; Schema: public; Owner: neondb_owner
  --

  COMMENT ON TABLE public.bookings IS 'Booking transactions. Links users,
resources, and services. Syncs with Google Calendar via gcal_event_id. Single-
tenant architecture.';


  --
  -- Name: notification_configs; Type: TABLE; Schema: public; Owner: neondb_owner
  --

  CREATE TABLE public.notification_configs (
      id uuid DEFAULT gen_random_uuid() NOT NULL,
      reminder_1_hours integer DEFAULT 24,
      reminder_2_hours integer DEFAULT 2,
      is_active boolean DEFAULT true,
      created_at timestamp with time zone DEFAULT now(),
      updated_at timestamp with time zone DEFAULT now(),
      default_duration_min integer DEFAULT 30,
      min_duration_min integer DEFAULT 15,
      max_duration_min integer DEFAULT 120,
      CONSTRAINT notification_configs_check CHECK ((max_duration_min >=
min_duration_min)),
      CONSTRAINT notification_configs_default_duration_min_check CHECK
((default_duration_min > 0)),
      CONSTRAINT notification_configs_min_duration_min_check CHECK
((min_duration_min > 0)),
      CONSTRAINT notification_configs_reminder_1_hours_check CHECK
((reminder_1_hours > 0)),
      CONSTRAINT notification_configs_reminder_2_hours_check CHECK
((reminder_2_hours > 0))
  );


  ALTER TABLE public.notification_configs OWNER TO neondb_owner;

  --
  -- Name: notification_queue; Type: TABLE; Schema: public; Owner: neondb_owner
  --

  CREATE TABLE public.notification_queue (
      id uuid DEFAULT gen_random_uuid() NOT NULL,
      booking_id uuid,
      user_id uuid,
      message text NOT NULL,
      priority integer DEFAULT 0,
      status public.notification_status DEFAULT
'pending'::public.notification_status,
      retry_count integer DEFAULT 0,
      error_message text,
      created_at timestamp with time zone DEFAULT now(),
      updated_at timestamp with time zone DEFAULT now(),
      sent_at timestamp with time zone,
      CONSTRAINT notification_queue_retry_count_check CHECK (((retry_count >= 0)
AND (retry_count <= 10)))
  );
  ALTER TABLE public.notification_queue OWNER TO neondb_owner;

  --
  -- Name: TABLE notification_queue; Type: COMMENT; Schema: public; Owner:
neondb_owner
  --

  COMMENT ON TABLE public.notification_queue IS 'Queue for asynchronous
notifications with retry support. Processed by BB_07 retry worker.';


  --
  -- Name: providers; Type: TABLE; Schema: public; Owner: neondb_owner
  --

  CREATE TABLE public.providers (
      id uuid DEFAULT public.uuid_generate_v4() NOT NULL,
      user_id uuid,
      name text NOT NULL,
      email public.citext,
      google_calendar_id text,
      slot_duration_minutes integer DEFAULT 30,
      min_notice_hours integer DEFAULT 2,
      public_booking_enabled boolean DEFAULT true,
      created_at timestamp with time zone DEFAULT now(),
      deleted_at timestamp with time zone,
      slug text,
      CONSTRAINT check_min_notice_positive CHECK ((min_notice_hours >= 0)),
      CONSTRAINT check_provider_name_not_empty CHECK ((length(TRIM(BOTH FROM name))
> 0)),
      CONSTRAINT check_slot_duration_positive CHECK ((slot_duration_minutes > 0)),
      CONSTRAINT check_slug_format CHECK ((slug ~* '^[a-z0-9-]+$'::text))
  );


  ALTER TABLE public.providers OWNER TO neondb_owner;

  --
  -- Name: TABLE providers; Type: COMMENT; Schema: public; Owner: neondb_owner
  --

  COMMENT ON TABLE public.providers IS 'Service providers (specialists,
practitioners). Modern single-tenant architecture with respectful terminology.';


  --
  -- Name: schedules; Type: TABLE; Schema: public; Owner: neondb_owner
  --

  CREATE TABLE public.schedules (
      id uuid DEFAULT public.uuid_generate_v4() NOT NULL,
      provider_id uuid NOT NULL,
      day_of_week public.day_of_week NOT NULL,
      start_time time without time zone NOT NULL,
      end_time time without time zone NOT NULL,
      is_active boolean DEFAULT true,
      CONSTRAINT valid_shift CHECK ((end_time > start_time))
  );


  ALTER TABLE public.schedules OWNER TO neondb_owner;

  --
  -- Name: TABLE schedules; Type: COMMENT; Schema: public; Owner: neondb_owner
  --

  COMMENT ON TABLE public.schedules IS 'Weekly availability schedules for
professionals. Defines working hours per day of week.';


  --
  -- Name: security_firewall; Type: TABLE; Schema: public; Owner: neondb_owner
  --

  CREATE TABLE public.security_firewall (
      id uuid DEFAULT public.uuid_generate_v4() NOT NULL,
      entity_id text NOT NULL,
      strike_count integer DEFAULT 0,
      is_blocked boolean DEFAULT false,
      blocked_until timestamp with time zone,
      last_strike_at timestamp with time zone DEFAULT now(),
      created_at timestamp with time zone DEFAULT now(),
      updated_at timestamp with time zone DEFAULT now()
  );


  ALTER TABLE public.security_firewall OWNER TO neondb_owner;

  --
  -- Name: services; Type: TABLE; Schema: public; Owner: neondb_owner
  --

  CREATE TABLE public.services (
      id uuid DEFAULT public.uuid_generate_v4() NOT NULL,
      provider_id uuid NOT NULL,
      name text NOT NULL,
      description text,
      duration_minutes integer NOT NULL,
      price numeric(10,2) DEFAULT 0,
      tier public.service_tier DEFAULT 'standard'::public.service_tier,
      active boolean DEFAULT true,
      CONSTRAINT check_duration_positive CHECK ((duration_minutes > 0)),
      CONSTRAINT check_price_non_negative CHECK ((price >= (0)::numeric)),
      CONSTRAINT check_service_name_not_empty CHECK ((length(TRIM(BOTH FROM name))
> 0))
  );


  ALTER TABLE public.services OWNER TO neondb_owner;

  --
  -- Name: TABLE services; Type: COMMENT; Schema: public; Owner: neondb_owner
  --

  COMMENT ON TABLE public.services IS 'Services offered by professionals. Each
service has a duration, price, and tier (standard/premium/emergency).';
```

```
--
-- Name: system_errors; Type: TABLE; Schema: public; Owner: neondb_owner
--

CREATE TABLE public.system_errors (
    error_id uuid DEFAULT public.uuid_generate_v4() NOT NULL,
    workflow_name text,
    workflow_execution_id text,
    error_type text,
    severity text,
    error_message text,
    error_stack text,
    error_context jsonb,
    user_id uuid,
    created_at timestamp with time zone DEFAULT now()
);


ALTER TABLE public.system_errors OWNER TO neondb_owner;

--
-- Name: users; Type: TABLE; Schema: public; Owner: neondb_owner
--

CREATE TABLE public.users (
    id uuid DEFAULT public.uuid_generate_v4() NOT NULL,
    telegram_id bigint NOT NULL,
    first_name text,
    last_name text,
    username text,
    phone_number text,
    rut text,
    role public.user_role DEFAULT 'user'::public.user_role,
    language_code public.supported_lang DEFAULT 'es'::public.supported_lang,
    metadata jsonb DEFAULT '{}'::jsonb,
    created_at timestamp with time zone DEFAULT now(),
    updated_at timestamp with time zone DEFAULT now(),
    deleted_at timestamp with time zone,
    password_hash text,
    last_selected_provider_id uuid,
    CONSTRAINT check_rut_format CHECK (((rut IS NULL) OR (rut ~* '^[0-9]+-[0-9kK]
$'::text)))
);
```