DATABASE

Stratum AI - Database Schema Documentation

Overview

Stratum AI uses PostgreSQL 16 with SQLAlchemy 2.0 ORM. The database implements multi-tenancy through tenant_id columns on all tables, with row-level security enforced at the application layer.

Entity Relationship Diagram

```
┌────────────────────────────────────────────────────────────────────────────
─┐
│                           STRATUM AI DATABASE SCHEMA
│
└────────────────────────────────────────────────────────────────────────────
─┘

┌─────────────┐
│   TENANTS   │──────────────────────────────────────────────────────┐
└──────┬──────┘                                                      │
       │ 1:N                                                         │
       │                                                             │
       ├────────────┬────────────┬────────────┬────────────┐        │
       │            │            │            │            │        │
       ▼            ▼            ▼            ▼            ▼        │
┌──────────┐ ┌──────────┐ ┌──────────┐ ┌──────────┐ ┌──────────┐   │
│  USERS   │ │CAMPAIGNS │ │  RULES   │ │COMPETITORS│ │ WHATSAPP │   │
└────┬─────┘ └────┬─────┘ └────┬─────┘ └──────────┘ │ CONTACTS │   │
     │            │            │                     └────┬─────┘   │
     │            │ 1:N        │ 1:N                      │         │
     │            │            │                          │ 1:N     │
     │            ▼            ▼                          ▼         │
     │      ┌──────────┐ ┌──────────┐              ┌──────────┐    │
     │      │ CAMPAIGN │ │  RULE    │              │ WHATSAPP │    │
     │      │ METRICS  │ │EXECUTIONS│              │ MESSAGES │    │
     │      └──────────┘ └──────────┘              └──────────┘    │
     │            │                                                 │
     │            │ 1:N                                            │
     │            ▼                                                │
     │      ┌──────────┐                                           │
     │      │ CREATIVE │                                           │
     │      │  ASSETS  │                                           │
     │      └──────────┘                                            │
     │                                                              │
     │ 1:N                                                          │
     ▼                                                              │
┌──────────┐                                                        │
│  AUDIT   │◄───────────────────────────────────────────────────────┘
│   LOGS   │  (All tables log to audit)
└──────────┘
```

Core Tables

tenants

Organizations using the platform. Root entity for multi-tenancy.

```sql
CREATE TABLE tenants (
    id                SERIAL PRIMARY KEY,
    name              VARCHAR(255) NOT NULL,
    slug              VARCHAR(100) UNIQUE NOT NULL,
    domain            VARCHAR(255),

    -- Subscription
    plan              VARCHAR(50) DEFAULT 'free' NOT NULL,
    plan_expires_at TIMESTAMP WITH TIME ZONE,
    stripe_customer_id VARCHAR(255),

    -- Settings
    settings          JSONB DEFAULT '{}' NOT NULL,
    feature_flags     JSONB DEFAULT '{}' NOT NULL,

    -- Limits
    max_users         INTEGER DEFAULT 5 NOT NULL,
    max_campaigns     INTEGER DEFAULT 50 NOT NULL,

    -- Soft Delete & Timestamps
    is_deleted        BOOLEAN DEFAULT FALSE,
    deleted_at        TIMESTAMP WITH TIME ZONE,
    created_at        TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
    updated_at        TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

CREATE INDEX ix_tenants_slug ON tenants(slug);
CREATE INDEX ix_tenants_active ON tenants(is_deleted, plan);
```

Columns:

| Column | Type | Description |

id	SERIAL	Primary key
name	VARCHAR(255)	Organization display name
slug	VARCHAR(100)	URL-safe identifier
domain	VARCHAR(255)	Custom domain (optional)
plan	VARCHAR(50)	Subscription plan: free, starter, pro, enterprise
settings	JSONB	Tenant configuration
feature_flags	JSONB	Feature toggles

---

users

User accounts with role-based access control. PII fields are encrypted.

```
CREATE TABLE users (
    id              SERIAL PRIMARY KEY,
    tenant_id       INTEGER NOT NULL REFERENCES tenants(id),

    -- Authentication
    email           VARCHAR(255) NOT NULL,      -- Encrypted PII
    email_hash      VARCHAR(64) NOT NULL,       -- SHA-256 for lookups
    password_hash   VARCHAR(255) NOT NULL,      -- bcrypt

    -- Profile (Encrypted PII)
    full_name       VARCHAR(255),
    phone           VARCHAR(100),
    avatar_url      VARCHAR(500),

    -- Role & Permissions
    role            user_role DEFAULT 'analyst' NOT NULL,
    permissions     JSONB DEFAULT '{}' NOT NULL,

    -- Status
    is_active       BOOLEAN DEFAULT TRUE NOT NULL,
    is_verified     BOOLEAN DEFAULT FALSE NOT NULL,
    last_login_at   TIMESTAMP WITH TIME ZONE,

    -- Preferences
    locale          VARCHAR(10) DEFAULT 'en' NOT NULL,
    timezone        VARCHAR(50) DEFAULT 'UTC' NOT NULL,
    preferences     JSONB DEFAULT '{}' NOT NULL,

    -- GDPR
    consent_marketing   BOOLEAN DEFAULT FALSE,
    consent_analytics   BOOLEAN DEFAULT TRUE,
    gdpr_anonymized_at  TIMESTAMP WITH TIME ZONE,
    -- Soft Delete & Timestamps
    is_deleted      BOOLEAN DEFAULT FALSE,
    deleted_at      TIMESTAMP WITH TIME ZONE,
    created_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
    updated_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

    UNIQUE (tenant_id, email_hash)
);

CREATE TYPE user_role AS ENUM ('admin', 'manager', 'analyst', 'viewer');

CREATE INDEX ix_users_email_hash ON users(email_hash);
CREATE INDEX ix_users_tenant_active ON users(tenant_id, is_active,
is_deleted);
```

Role Permissions:

Role	Description	Capabilities
admin	Full access	All operations
manager	Team management	CRUD on campaigns, rules, team members
analyst	Reporting	Create reports, view data
viewer	Read-only	View dashboards only

---

campaigns

Unified campaign model normalizing data from all ad platforms.

```
CREATE TABLE campaigns (
    id              SERIAL PRIMARY KEY,
    tenant_id       INTEGER NOT NULL REFERENCES tenants(id),

    -- Platform Reference
    platform        ad_platform NOT NULL,
    external_id     VARCHAR(255) NOT NULL,
    account_id      VARCHAR(255) NOT NULL,

    -- Campaign Info
    name            VARCHAR(500) NOT NULL,
    status          campaign_status DEFAULT 'draft' NOT NULL,
    objective       VARCHAR(100),

    -- Budget (in cents to avoid floating point)
    daily_budget_cents      INTEGER,
    lifetime_budget_cents   INTEGER,
    total_spend_cents       INTEGER DEFAULT 0 NOT NULL,
    currency                VARCHAR(3) DEFAULT 'USD' NOT NULL,
```

```sql
    -- Performance Metrics (Aggregated)
    impressions     INTEGER DEFAULT 0 NOT NULL,
    clicks          INTEGER DEFAULT 0 NOT NULL,
    conversions     INTEGER DEFAULT 0 NOT NULL,
    revenue_cents   INTEGER DEFAULT 0 NOT NULL,

    -- Computed Metrics
    ctr             FLOAT,      -- Click-through rate
    cpc_cents       INTEGER,    -- Cost per click
    cpm_cents       INTEGER,    -- Cost per mille
    cpa_cents       INTEGER,    -- Cost per acquisition
    roas            FLOAT,      -- Return on ad spend

    -- Targeting
    targeting_age_min   INTEGER,
    targeting_age_max   INTEGER,
    targeting_genders   JSONB,
    targeting_locations JSONB,
    targeting_interests JSONB,

    -- Demographics Breakdown
    demographics_age      JSONB,
    demographics_gender   JSONB,
    demographics_location JSONB,

    -- Scheduling
    start_date      DATE,
    end_date        DATE,

    -- Organization
    labels          JSONB DEFAULT '[]' NOT NULL,

    -- Raw platform data
    raw_data        JSONB,

    -- Sync metadata
    last_synced_at  TIMESTAMP WITH TIME ZONE,
    sync_error      TEXT,

    -- Soft Delete & Timestamps
    is_deleted      BOOLEAN DEFAULT FALSE,
    deleted_at      TIMESTAMP WITH TIME ZONE,
    created_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
    updated_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

    UNIQUE (tenant_id, platform, external_id)
);

CREATE TYPE ad_platform AS ENUM ('meta', 'google', 'tiktok', 'snapchat',
'linkedin');
CREATE TYPE campaign_status AS ENUM ('draft', 'active', 'paused',
'completed', 'archived');

CREATE INDEX ix_campaigns_tenant_status ON campaigns(tenant_id, status);
CREATE INDEX ix_campaigns_platform ON campaigns(tenant_id, platform);
CREATE INDEX ix_campaigns_date_range ON campaigns(tenant_id, start_date,
end_date);
CREATE INDEX ix_campaigns_roas ON campaigns(tenant_id, roas);
```

Money Handling:
All monetary values are stored as integers in cents to avoid floating-point precision issues:

- `$100.50` → `10050` cents
- Convert on read: `total_spend_cents / 100`

---

campaign_metrics

Daily time-series metrics for campaigns.

```
CREATE TABLE campaign_metrics (
    id                SERIAL PRIMARY KEY,
    tenant_id         INTEGER NOT NULL,
    campaign_id       INTEGER NOT NULL REFERENCES campaigns(id) ON DELETE
CASCADE,
    date              DATE NOT NULL,

    -- Daily Metrics
    impressions       INTEGER DEFAULT 0 NOT NULL,
    clicks            INTEGER DEFAULT 0 NOT NULL,
    conversions       INTEGER DEFAULT 0 NOT NULL,
    spend_cents       INTEGER DEFAULT 0 NOT NULL,
    revenue_cents     INTEGER DEFAULT 0 NOT NULL,

    -- Engagement
    video_views           INTEGER,
    video_completions     INTEGER,
    shares                INTEGER,
    comments              INTEGER,
    saves                 INTEGER,

    -- Demographics snapshot
    demographics    JSONB,

    UNIQUE (campaign_id, date)
);
```

```
CREATE INDEX ix_campaign_metrics_date ON campaign_metrics(tenant_id, date);
CREATE INDEX ix_campaign_metrics_campaign_date ON
campaign_metrics(campaign_id, date);
```

creative_assets

Digital Asset Management for ad creatives.

```
CREATE TABLE creative_assets (
    id              SERIAL PRIMARY KEY,
    tenant_id       INTEGER NOT NULL REFERENCES tenants(id),
    campaign_id     INTEGER REFERENCES campaigns(id) ON DELETE SET NULL,

    -- Asset Info
    name            VARCHAR(500) NOT NULL,
    asset_type      asset_type NOT NULL,
    file_url        VARCHAR(1000) NOT NULL,
    thumbnail_url   VARCHAR(1000),

    -- File Metadata
    file_size_bytes INTEGER,
    file_format     VARCHAR(50),
    width           INTEGER,
    height          INTEGER,
    duration_seconds FLOAT,

    -- Organization
    tags            JSONB DEFAULT '[]' NOT NULL,
    folder          VARCHAR(255),

    -- Performance
    impressions     INTEGER DEFAULT 0 NOT NULL,
    clicks          INTEGER DEFAULT 0 NOT NULL,
    ctr             FLOAT,

    -- Creative Fatigue
    fatigue_score       FLOAT DEFAULT 0.0 NOT NULL,
    first_used_at       TIMESTAMP WITH TIME ZONE,
    times_used          INTEGER DEFAULT 0 NOT NULL,

    -- AI Metadata
    ai_description      TEXT,
    ai_tags             JSONB,
    brand_safety_score  FLOAT,
    -- Soft Delete & Timestamps
    is_deleted      BOOLEAN DEFAULT FALSE,
    created_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
    updated_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

CREATE TYPE asset_type AS ENUM ('image', 'video', 'carousel', 'story', 'html5');

CREATE INDEX ix_assets_tenant_type ON creative_assets(tenant_id, asset_type);
CREATE INDEX ix_assets_fatigue ON creative_assets(tenant_id, fatigue_score);
```

rules

Automation rules engine (IFTTT-style).

```
CREATE TABLE rules (
    id              SERIAL PRIMARY KEY,
    tenant_id       INTEGER NOT NULL REFERENCES tenants(id),

    -- Rule Info
    name            VARCHAR(255) NOT NULL,
    description     TEXT,
    status          rule_status DEFAULT 'draft' NOT NULL,

    -- Condition (IF)
    condition_field         VARCHAR(100) NOT NULL,
    condition_operator      rule_operator NOT NULL,
    condition_value         VARCHAR(255) NOT NULL,
    condition_duration_hours INTEGER DEFAULT 24 NOT NULL,

    -- Action (THEN)
    action_type     rule_action NOT NULL,
    action_config   JSONB DEFAULT '{}' NOT NULL,

    -- Scope
    applies_to_campaigns    JSONB,  -- Campaign IDs or null for all
    applies_to_platforms    JSONB,  -- Platform list or null for all

    -- Execution tracking
    last_evaluated_at   TIMESTAMP WITH TIME ZONE,
    last_triggered_at   TIMESTAMP WITH TIME ZONE,
    trigger_count       INTEGER DEFAULT 0 NOT NULL,
    cooldown_hours      INTEGER DEFAULT 24 NOT NULL,
    -- Soft Delete & Timestamps
    is_deleted      BOOLEAN DEFAULT FALSE,
    created_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
    updated_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

CREATE TYPE rule_status AS ENUM ('active', 'paused', 'draft');
CREATE TYPE rule_operator AS ENUM ('equals', 'not_equals', 'greater_than',
'less_than', 'gte', 'lte', 'contains', 'in');
CREATE TYPE rule_action AS ENUM ('apply_label', 'send_alert',
'pause_campaign', 'adjust_budget', 'notify_slack', 'notify_whatsapp');

CREATE INDEX ix_rules_tenant_status ON rules(tenant_id, status);
CREATE INDEX ix_rules_evaluation ON rules(status, last_evaluated_at);
```

Rule Condition Examples:

```
// ROAS below 1.5 for 48 hours
{
  "field": "roas",
  "operator": "less_than",
  "value": "1.5",
  "duration_hours": 48
}

// Spend over $1000
{
  "field": "total_spend",
  "operator": "greater_than",
  "value": "1000",
  "duration_hours": 24
}
```

rule_executions

Log of rule executions for audit.

```
CREATE TABLE rule_executions (
    id              SERIAL PRIMARY KEY,
    tenant_id       INTEGER NOT NULL,
    rule_id         INTEGER NOT NULL REFERENCES rules(id) ON DELETE CASCADE,
    campaign_id     INTEGER REFERENCES campaigns(id) ON DELETE SET NULL,

    -- Execution Details
    executed_at     TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
    triggered       BOOLEAN NOT NULL,
```

```sql
    condition_result JSONB NOT NULL,
    action_result   JSONB,
    error           TEXT
);

CREATE INDEX ix_rule_executions_rule_date ON rule_executions(rule_id, executed_at);
CREATE INDEX ix_rule_executions_tenant_date ON rule_executions(tenant_id, executed_at);
```

---

competitor_benchmarks

Competitor intelligence data.

```sql
CREATE TABLE competitor_benchmarks (
    id              SERIAL PRIMARY KEY,
    tenant_id       INTEGER NOT NULL REFERENCES tenants(id),

    -- Competitor Info
    domain          VARCHAR(255) NOT NULL,
    name            VARCHAR(255),
    is_primary      BOOLEAN DEFAULT FALSE,

    -- Scraped Metadata
    meta_title          VARCHAR(500),
    meta_description    TEXT,
    meta_keywords       JSONB,
    social_links        JSONB,

    -- Market Intelligence
    estimated_traffic       INTEGER,
    traffic_trend           VARCHAR(20),
    top_keywords            JSONB,
    paid_keywords_count     INTEGER,
    organic_keywords_count  INTEGER,

    -- Share of Voice
    share_of_voice      FLOAT,
    category_rank       INTEGER,

    -- Ad Intelligence
    estimated_ad_spend_cents INTEGER,
    detected_ad_platforms    JSONB,
    ad_creatives_count       INTEGER,

    -- Historical
    metrics_history     JSONB,

    -- Data Source
    data_source         VARCHAR(50) DEFAULT 'scraper' NOT NULL,
    last_fetched_at     TIMESTAMP WITH TIME ZONE,
    fetch_error         TEXT,

    -- Timestamps
    created_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
    updated_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW(),

    UNIQUE (tenant_id, domain)
);

CREATE INDEX ix_competitors_tenant_primary ON
competitor_benchmarks(tenant_id, is_primary);
CREATE INDEX ix_competitors_sov ON competitor_benchmarks(tenant_id,
share_of_voice);
```

audit_logs

Security and compliance audit trail.

```
CREATE TABLE audit_logs (
    id              SERIAL PRIMARY KEY,
    tenant_id       INTEGER NOT NULL,
    user_id         INTEGER REFERENCES users(id) ON DELETE SET NULL,

    -- Action Details
    action          audit_action NOT NULL,
    resource_type   VARCHAR(100) NOT NULL,
    resource_id     VARCHAR(100),

    -- Change Tracking
    old_value       JSONB,
    new_value       JSONB,
    changed_fields  JSONB,

    -- Request Context
    ip_address      VARCHAR(45),
    user_agent      VARCHAR(500),
    request_id      VARCHAR(100),
    endpoint        VARCHAR(255),
    http_method     VARCHAR(10),

    -- Timestamp
    created_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

CREATE TYPE audit_action AS ENUM ('create', 'update', 'delete', 'login',
'logout', 'export', 'anonymize');

CREATE INDEX ix_audit_tenant_date ON audit_logs(tenant_id, created_at);
CREATE INDEX ix_audit_user_date ON audit_logs(user_id, created_at);
CREATE INDEX ix_audit_resource ON audit_logs(resource_type, resource_id);
CREATE INDEX ix_audit_action ON audit_logs(tenant_id, action, created_at);
```

WhatsApp Tables

whatsapp_contacts

```
CREATE TABLE whatsapp_contacts (
    id              SERIAL PRIMARY KEY,
    tenant_id       INTEGER NOT NULL REFERENCES tenants(id),
    user_id         INTEGER NOT NULL REFERENCES users(id) ON DELETE CASCADE,

    -- Contact Info (E.164)
    phone_number    VARCHAR(20) NOT NULL,
    country_code    VARCHAR(5) NOT NULL,
    display_name    VARCHAR(255),

    -- Verification
    is_verified             BOOLEAN DEFAULT FALSE,
    verification_code       VARCHAR(6),
    verification_expires_at TIMESTAMP WITH TIME ZONE,
    verified_at             TIMESTAMP WITH TIME ZONE,

    -- Opt-in Status (Required for WhatsApp Business)
    opt_in_status   whatsapp_opt_in DEFAULT 'pending' NOT NULL,
    opt_in_at       TIMESTAMP WITH TIME ZONE,
    opt_out_at      TIMESTAMP WITH TIME ZONE,
    opt_in_method   VARCHAR(50),

    -- WhatsApp Profile
    wa_id               VARCHAR(50),
    profile_name        VARCHAR(255),
    profile_picture_url VARCHAR(500),

    -- Preferences
    notification_types  JSONB DEFAULT '["alerts", "reports"]',
    quiet_hours         JSONB DEFAULT '{"enabled": false}',
    timezone            VARCHAR(50) DEFAULT 'UTC',
    language            VARCHAR(10) DEFAULT 'en',

    -- Status
    is_active           BOOLEAN DEFAULT TRUE,
    last_message_at     TIMESTAMP WITH TIME ZONE,
    message_count       INTEGER DEFAULT 0,

    -- Timestamps
    created_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
    updated_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

CREATE TYPE whatsapp_opt_in AS ENUM ('pending', 'opted_in', 'opted_out');
```

whatsapp_messages

```
CREATE TABLE whatsapp_messages (
    id              SERIAL PRIMARY KEY,
    tenant_id       INTEGER NOT NULL,
    contact_id      INTEGER NOT NULL REFERENCES whatsapp_contacts(id) ON
DELETE CASCADE,
    template_id     INTEGER REFERENCES whatsapp_templates(id) ON DELETE SET
NULL,

    -- Message Details
    direction       whatsapp_direction DEFAULT 'outbound' NOT NULL,
    message_type    VARCHAR(20) NOT NULL,

    -- Content
    template_name       VARCHAR(100),
    template_variables  JSONB DEFAULT '{}',
    content             TEXT,
    media_url           VARCHAR(500),
    media_type          VARCHAR(50),

    -- WhatsApp API Response
    wamid               VARCHAR(100),
    recipient_wa_id     VARCHAR(50),

    -- Status Tracking
    status              whatsapp_status DEFAULT 'pending' NOT NULL,
    status_history      JSONB DEFAULT '[]',

    -- Error Handling
    error_code      VARCHAR(20),
    error_message   TEXT,
    retry_count     INTEGER DEFAULT 0,
    next_retry_at   TIMESTAMP WITH TIME ZONE,
    -- Timing
    scheduled_at    TIMESTAMP WITH TIME ZONE,
    sent_at         TIMESTAMP WITH TIME ZONE,
    delivered_at    TIMESTAMP WITH TIME ZONE,
    read_at         TIMESTAMP WITH TIME ZONE,
    created_at      TIMESTAMP WITH TIME ZONE DEFAULT NOW()
);

CREATE TYPE whatsapp_direction AS ENUM ('outbound', 'inbound');
CREATE TYPE whatsapp_status AS ENUM ('pending', 'sent', 'delivered', 'read',
'failed');
```

ML Tables

ml_predictions

Cache for ML model predictions.

```
CREATE TABLE ml_predictions (
    id              SERIAL PRIMARY KEY,
    tenant_id       INTEGER NOT NULL,
    campaign_id     INTEGER REFERENCES campaigns(id) ON DELETE CASCADE,

    -- Prediction Info
    model_type      VARCHAR(50) NOT NULL,
    model_version   VARCHAR(50) NOT NULL,
    input_hash      VARCHAR(64) NOT NULL,

    -- Result
    prediction_value    FLOAT NOT NULL,
    confidence_lower    FLOAT,
    confidence_upper    FLOAT,
    feature_importances JSONB,

    -- Metadata
    predicted_at    TIMESTAMP WITH TIME ZONE DEFAULT NOW(),
    expires_at      TIMESTAMP WITH TIME ZONE NOT NULL
);

CREATE INDEX ix_predictions_cache ON ml_predictions(model_type, input_hash);
CREATE INDEX ix_predictions_expiry ON ml_predictions(expires_at);
```

Database Migrations

Migrations are managed with Alembic.

Running Migrations

```bash
# Create new migration
alembic revision --autogenerate -m "Add new column"

# Apply migrations
alembic upgrade head

# Rollback one version
alembic downgrade -1

# View history
alembic history
```

Migration Best Practices

1. **Always review auto-generated migrations** before applying
2. **Include both upgrade and downgrade** functions
3. **Test migrations on a copy** of production data
4. **Batch large data migrations** to avoid locking

Query Patterns

Multi-Tenant Queries

All queries must include tenant_id filter:

```python
# Service layer pattern
async def get_campaigns(db: AsyncSession, tenant_id: int):
    query = select(Campaign).where(
        Campaign.tenant_id == tenant_id,
        Campaign.is_deleted == False
    )
    result = await db.execute(query)
    return result.scalars().all()
```

Soft Delete

Records are never physically deleted:

```python
async def delete_campaign(db: AsyncSession, campaign_id: int):
    campaign = await db.get(Campaign, campaign_id)
    campaign.is_deleted = True
    campaign.deleted_at = datetime.utcnow()
    await db.commit()
```

JSONB Queries

```sql
-- Query campaigns with specific label
SELECT * FROM campaigns
WHERE labels @> '["summer"]';

-- Query users with specific permission
SELECT * FROM users
WHERE permissions->>'can_export' = 'true';
```

Performance Optimization

Indexes Used

Table	Index	Columns	Purpose
campaigns	ix_campaigns_tenant_status	(tenant_id, status)	Filter by status
campaigns	ix_campaigns_roas	(tenant_id, roas)	Sort by ROAS
campaign_metrics	ix_metrics_date	(tenant_id, date)	Time-range queries
users	ix_users_email_hash	(email_hash)	Login lookup
audit_logs	ix_audit_tenant_date	(tenant_id, created_at)	Audit queries

Query Tips

1. **Always filter by tenant_id first** - hits the index
2. **Use date range filters** for metrics queries
3. **Limit JSONB queries** - not efficiently indexed
4. **Paginate large result sets** - use LIMIT/OFFSET

Backup & Recovery

Backup Strategy

```
# Full backup
pg_dump -h localhost -U stratum stratum_ai > backup.sql

# With compression
pg_dump -h localhost -U stratum stratum_ai | gzip > backup.sql.gz

# Automated daily backup (cron)
0 2 * * * pg_dump -h localhost -U stratum stratum_ai | gzip >
/backups/stratum_$(date +\%Y\%m\%d).sql.gz
```

Point-in-Time Recovery

Enable WAL archiving for PITR:

```
ALTER SYSTEM SET wal_level = replica;
ALTER SYSTEM SET archive_mode = on;
ALTER SYSTEM SET archive_command = 'cp %p /archive/%f';
```

Security Considerations

1. **Connection Encryption**: Use SSL/TLS for all connections
2. **Credential Rotation**: Rotate passwords regularly
3. **Access Control**: Use separate accounts for app/admin
4. **Audit Logging**: All changes logged to audit_logs
5. **PII Encryption**: Sensitive fields encrypted with Fernet