|  |
| --- |
| We are to implement these indexes in inreamentaly, we shall implement indexes per phase |
|  |

-- Enable required extensions

CREATE EXTENSION IF NOT EXISTS "uuid-ossp";

CREATE EXTENSION IF NOT EXISTS "pg\_trgm";

CREATE EXTENSION IF NOT EXISTS "btree\_gin";

-- Subsystem 1: Core User Identity & RBAC

CREATE TABLE users (

user\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

email VARCHAR(255) UNIQUE NOT NULL,

full\_name VARCHAR(255) NOT NULL,

phone\_number VARCHAR(50),

gps\_location JSONB,

communication\_preferences JSONB DEFAULT '{"email\_notifications": true, "sms\_notifications": true, "push\_notifications": true}'

CHECK (communication\_preferences ? 'email\_notifications'),

is\_active BOOLEAN DEFAULT true,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

last\_login TIMESTAMP

);

CREATE TABLE roles (

role\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

role\_name VARCHAR(50) UNIQUE NOT NULL,

permissions JSONB NOT NULL,

description TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE user\_roles (

user\_role\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_id UUID NOT NULL REFERENCES users(user\_id) ON DELETE CASCADE,

role\_id UUID NOT NULL REFERENCES roles(role\_id) ON DELETE CASCADE,

restaurant\_id UUID, -- NULL for global roles (customer, admin)

assigned\_by UUID REFERENCES users(user\_id),

assigned\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

is\_active BOOLEAN DEFAULT true,

CONSTRAINT fk\_restaurant FOREIGN KEY (restaurant\_id) REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE

);

-- Indexes for Subsystem 1

CREATE INDEX idx\_users\_email ON users(email);

CREATE INDEX idx\_users\_active ON users(is\_active) WHERE is\_active = true;

CREATE INDEX idx\_users\_last\_login ON users(last\_login);

CREATE INDEX idx\_users\_communication\_prefs ON users USING gin(communication\_preferences);

CREATE INDEX idx\_roles\_name ON roles(role\_name);

CREATE INDEX idx\_user\_roles\_user ON user\_roles(user\_id);

CREATE INDEX idx\_user\_roles\_restaurant ON user\_roles(restaurant\_id);

CREATE INDEX idx\_user\_roles\_active ON user\_roles(user\_id, role\_id, restaurant\_id) WHERE is\_active = true;

CREATE UNIQUE INDEX idx\_user\_role\_unique ON user\_roles(user\_id, role\_id, restaurant\_id) WHERE is\_active = true;

-- Subsystem 2: Core Operations & Venue Management

CREATE TABLE restaurants (

restaurant\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

name VARCHAR(255) NOT NULL,

description TEXT,

cuisine\_type VARCHAR(100),

address JSONB NOT NULL

CHECK (address ? 'street' AND address ? 'city' AND address ? 'country'),

contact\_info JSONB NOT NULL

CHECK (contact\_info ? 'phone' AND contact\_info ? 'email'),

operation\_hours JSONB NOT NULL,

social\_media\_links JSONB,

delivery\_options JSONB,

payment\_methods\_accepted JSONB,

average\_rating DECIMAL(3,2) DEFAULT 0.00,

total\_reviews INTEGER DEFAULT 0,

average\_delivery\_time INTEGER,

status VARCHAR(20) DEFAULT 'active' CHECK (status IN ('active', 'inactive', 'suspended')),

local\_server\_id UUID,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

deleted\_at TIMESTAMP,

created\_by UUID REFERENCES users(user\_id),

updated\_by UUID REFERENCES users(user\_id),

deleted\_by UUID REFERENCES users(user\_id)

);

CREATE TABLE restaurant\_tables (

table\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

table\_number VARCHAR(20) NOT NULL,

qr\_code VARCHAR(500) UNIQUE NOT NULL,

capacity INTEGER NOT NULL CHECK (capacity > 0),

table\_status VARCHAR(20) DEFAULT 'available' CHECK (table\_status IN ('available', 'occupied', 'reserved', 'maintenance')),

coordinates JSONB,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE local\_servers (

local\_server\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID UNIQUE NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

server\_name VARCHAR(255),

server\_url VARCHAR(500),

status VARCHAR(20) DEFAULT 'online' CHECK (status IN ('online', 'offline', 'maintenance')),

last\_sync TIMESTAMP,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Indexes for Subsystem 2

CREATE INDEX idx\_restaurants\_status ON restaurants(status);

CREATE INDEX idx\_restaurants\_cuisine ON restaurants(cuisine\_type);

CREATE INDEX idx\_restaurants\_rating ON restaurants(average\_rating);

CREATE INDEX idx\_restaurants\_created\_at ON restaurants(created\_at);

CREATE INDEX idx\_restaurants\_address\_city ON restaurants USING gin ((address->>'city'));

CREATE INDEX idx\_restaurants\_operation\_hours ON restaurants USING gin (operation\_hours);

CREATE INDEX idx\_restaurants\_geo ON restaurants USING gin ((

(address->'coordinates'->>'lat')::float,

(address->'coordinates'->>'lng')::float

));

CREATE INDEX idx\_tables\_restaurant ON restaurant\_tables(restaurant\_id);

CREATE INDEX idx\_tables\_status ON restaurant\_tables(table\_status);

CREATE INDEX idx\_tables\_capacity ON restaurant\_tables(capacity);

CREATE INDEX idx\_tables\_restaurant\_status ON restaurant\_tables(restaurant\_id, table\_status);

CREATE INDEX idx\_tables\_qr\_code ON restaurant\_tables(qr\_code);

CREATE INDEX idx\_local\_servers\_status ON local\_servers(status);

CREATE INDEX idx\_local\_servers\_last\_sync ON local\_servers(last\_sync);

-- Subsystem 3: Menu & Recipe Management

CREATE TABLE menus (

menu\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID UNIQUE NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

name VARCHAR(255) NOT NULL,

description TEXT,

is\_active BOOLEAN DEFAULT true,

version INTEGER DEFAULT 1,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE menu\_items (

menu\_item\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

menu\_id UUID NOT NULL REFERENCES menus(menu\_id) ON DELETE CASCADE,

item\_name VARCHAR(255) NOT NULL,

description TEXT,

sales\_price DECIMAL(10,2) NOT NULL CHECK (sales\_price >= 0),

preparation\_time INTEGER NOT NULL CHECK (preparation\_time > 0),

department VARCHAR(100),

is\_available BOOLEAN DEFAULT true,

display\_order INTEGER DEFAULT 0,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE inventory\_items (

inventory\_item\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

item\_name VARCHAR(255) NOT NULL,

description TEXT,

unit\_of\_measure VARCHAR(50) NOT NULL,

cost\_price DECIMAL(10,2) CHECK (cost\_price >= 0),

current\_stock DECIMAL(10,3) DEFAULT 0,

min\_stock\_threshold DECIMAL(10,3) DEFAULT 0,

max\_stock\_capacity DECIMAL(10,3),

stock\_status VARCHAR(20) DEFAULT 'in\_stock' CHECK (stock\_status IN ('in\_stock', 'low\_stock', 'out\_of\_stock')),

supplier\_id UUID,

last\_restocked TIMESTAMP,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE menu\_item\_ingredients (

menu\_item\_ingredient\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

menu\_item\_id UUID NOT NULL REFERENCES menu\_items(menu\_item\_id) ON DELETE CASCADE,

inventory\_item\_id UUID NOT NULL REFERENCES inventory\_items(inventory\_item\_id) ON DELETE CASCADE,

quantity\_required DECIMAL(10,3) NOT NULL CHECK (quantity\_required > 0),

unit VARCHAR(50) NOT NULL,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

UNIQUE(menu\_item\_id, inventory\_item\_id)

);

-- Indexes for Subsystem 3

CREATE INDEX idx\_menus\_restaurant\_active ON menus(restaurant\_id) WHERE is\_active = true;

CREATE INDEX idx\_menus\_active ON menus(is\_active);

CREATE INDEX idx\_menu\_items\_menu ON menu\_items(menu\_id);

CREATE INDEX idx\_menu\_items\_available ON menu\_items(is\_available);

CREATE INDEX idx\_menu\_items\_price ON menu\_items(sales\_price);

CREATE INDEX idx\_menu\_items\_prep\_time ON menu\_items(preparation\_time);

CREATE INDEX idx\_menu\_items\_department ON menu\_items(department);

CREATE INDEX idx\_menu\_items\_menu\_available\_order ON menu\_items(menu\_id, is\_available, display\_order);

CREATE INDEX idx\_menu\_items\_active\_price ON menu\_items(menu\_id, sales\_price) WHERE is\_available = true;

CREATE INDEX idx\_menu\_items\_name\_search ON menu\_items USING gin (item\_name gin\_trgm\_ops);

CREATE INDEX idx\_inventory\_restaurant ON inventory\_items(restaurant\_id);

CREATE INDEX idx\_inventory\_status ON inventory\_items(stock\_status);

CREATE INDEX idx\_inventory\_supplier ON inventory\_items(supplier\_id);

CREATE INDEX idx\_inventory\_low\_stock ON inventory\_items(restaurant\_id, stock\_status) WHERE stock\_status IN ('low\_stock', 'out\_of\_stock');

CREATE INDEX idx\_inventory\_name\_search ON inventory\_items USING gin (item\_name gin\_trgm\_ops);

CREATE INDEX idx\_ingredients\_menu\_item ON menu\_item\_ingredients(menu\_item\_id);

CREATE INDEX idx\_ingredients\_inventory\_item ON menu\_item\_ingredients(inventory\_item\_id);

-- Subsystem 4: Inventory, Supplier & Supply Chain

CREATE TABLE suppliers (

supplier\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_id UUID REFERENCES users(user\_id),

company\_name VARCHAR(255) NOT NULL,

contact\_person VARCHAR(255),

contact\_info JSONB NOT NULL,

address JSONB,

business\_registration VARCHAR(100),

payment\_terms JSONB,

rating DECIMAL(3,2) DEFAULT 0.00,

is\_active BOOLEAN DEFAULT true,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE restaurant\_suppliers (

restaurant\_supplier\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

supplier\_id UUID NOT NULL REFERENCES suppliers(supplier\_id) ON DELETE CASCADE,

relationship\_status VARCHAR(20) DEFAULT 'active' CHECK (relationship\_status IN ('active', 'suspended', 'inactive')),

is\_preferred BOOLEAN DEFAULT false,

payment\_terms JSONB,

delivery\_lead\_time INTEGER,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

UNIQUE(restaurant\_id, supplier\_id)

);

-- Add foreign key constraint for inventory\_items

ALTER TABLE inventory\_items ADD CONSTRAINT fk\_inventory\_supplier

FOREIGN KEY (supplier\_id) REFERENCES suppliers(supplier\_id);

-- Indexes for Subsystem 4

CREATE INDEX idx\_suppliers\_active ON suppliers(is\_active);

CREATE INDEX idx\_suppliers\_rating ON suppliers(rating);

CREATE INDEX idx\_suppliers\_company\_name ON suppliers USING gin (company\_name gin\_trgm\_ops);

CREATE INDEX idx\_suppliers\_contact\_info ON suppliers USING gin (contact\_info);

CREATE INDEX idx\_restaurant\_suppliers\_restaurant ON restaurant\_suppliers(restaurant\_id);

CREATE INDEX idx\_restaurant\_suppliers\_supplier ON restaurant\_suppliers(supplier\_id);

CREATE INDEX idx\_restaurant\_suppliers\_status ON restaurant\_suppliers(relationship\_status);

CREATE INDEX idx\_restaurant\_suppliers\_preferred ON restaurant\_suppliers(restaurant\_id, is\_preferred) WHERE is\_preferred = true;

-- Subsystem 5: Ordering, Kitchen & Logistics

-- Partitioned Orders Table

CREATE TABLE orders (

order\_id UUID NOT NULL,

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

order\_type VARCHAR(20) NOT NULL CHECK (order\_type IN ('sales', 'supply')),

status VARCHAR(50) NOT NULL CHECK (status IN ('pending', 'confirmed', 'preparing', 'ready', 'in\_delivery', 'delivered', 'cancelled')),

total\_amount DECIMAL(12,2) NOT NULL CHECK (total\_amount >= 0),

notes TEXT,

created\_at TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

created\_by UUID REFERENCES users(user\_id),

updated\_by UUID REFERENCES users(user\_id),

PRIMARY KEY (order\_id, created\_at)

) PARTITION BY RANGE (created\_at);

-- Create monthly partitions for orders

CREATE TABLE orders\_2024\_01 PARTITION OF orders

FOR VALUES FROM ('2024-01-01') TO ('2024-02-01');

CREATE TABLE orders\_2024\_02 PARTITION OF orders

FOR VALUES FROM ('2024-02-01') TO ('2024-03-01');

CREATE TABLE orders\_default PARTITION OF orders DEFAULT;

CREATE TABLE sales\_orders (

sales\_order\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

order\_id UUID UNIQUE NOT NULL,

customer\_user\_id UUID NOT NULL REFERENCES users(user\_id),

order\_subtype VARCHAR(20) NOT NULL CHECK (order\_subtype IN ('dine\_in', 'takeaway', 'delivery')),

table\_id UUID REFERENCES restaurant\_tables(table\_id),

assigned\_waiter\_id UUID,

batch\_id UUID,

delivery\_partner\_id UUID,

customer\_coordinates JSONB,

estimated\_preparation\_time INTEGER,

actual\_preparation\_time INTEGER,

estimated\_delivery\_time INTEGER,

actual\_delivery\_time INTEGER,

preparation\_complexity\_score INTEGER,

otp\_code VARCHAR(6)

);

CREATE TABLE supply\_orders (

supply\_order\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

order\_id UUID UNIQUE NOT NULL,

supplier\_id UUID NOT NULL REFERENCES suppliers(supplier\_id),

expected\_delivery\_date DATE,

delivery\_status VARCHAR(20) DEFAULT 'pending' CHECK (delivery\_status IN ('pending', 'in\_transit', 'delivered', 'cancelled')),

invoice\_total DECIMAL(12,2),

adjusted\_total DECIMAL(12,2),

rejection\_proof\_url VARCHAR(500),

supplier\_invoice\_number VARCHAR(100),

delivery\_notes TEXT,

quality\_rating INTEGER CHECK (quality\_rating BETWEEN 1 AND 5),

on\_time\_rating INTEGER CHECK (on\_time\_rating BETWEEN 1 AND 5)

);

CREATE TABLE order\_items (

order\_item\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

order\_id UUID NOT NULL,

source\_entity\_id UUID NOT NULL,

source\_entity\_type VARCHAR(20) NOT NULL CHECK (source\_entity\_type IN ('menu\_item', 'inventory\_item')),

quantity DECIMAL(10,3) NOT NULL CHECK (quantity > 0),

unit\_price DECIMAL(10,2) NOT NULL CHECK (unit\_price >= 0),

total\_price DECIMAL(10,2) NOT NULL CHECK (total\_price >= 0),

special\_instructions TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

item\_notes TEXT,

customization\_options JSONB,

chef\_special\_instructions TEXT

);

-- NEW: Order Item Rejections Table for individual item rejection proofs

CREATE TABLE order\_item\_rejections (

rejection\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

order\_item\_id UUID NOT NULL REFERENCES order\_items(order\_item\_id) ON DELETE CASCADE,

rejected\_quantity DECIMAL(10,3) NOT NULL CHECK (rejected\_quantity > 0),

rejection\_reason TEXT NOT NULL,

proof\_image\_url VARCHAR(500),

digital\_signature\_data JSONB,

rejected\_by UUID REFERENCES users(user\_id),

rejected\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

supplier\_acknowledged BOOLEAN DEFAULT false,

supplier\_acknowledged\_at TIMESTAMP

);

CREATE TABLE delivery\_batches (

batch\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

assigned\_waiter\_id UUID NOT NULL,

batch\_status VARCHAR(20) DEFAULT 'pending' CHECK (batch\_status IN ('pending', 'in\_progress', 'completed')),

optimized\_route JSONB,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

completed\_at TIMESTAMP,

total\_distance DECIMAL(8,2),

estimated\_completion\_time INTEGER,

actual\_completion\_time INTEGER,

fuel\_cost\_estimate DECIMAL(8,2),

batch\_efficiency\_score DECIMAL(4,2) DEFAULT 0.00

);

CREATE TABLE delivery\_partners (

delivery\_partner\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

partner\_name VARCHAR(255) NOT NULL,

partner\_type VARCHAR(20) NOT NULL CHECK (partner\_type IN ('internal', 'uber\_eats', 'glovo')),

contact\_info JSONB,

is\_active BOOLEAN DEFAULT true,

average\_rating DECIMAL(3,2) DEFAULT 0.00,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE delivery\_tracking (

tracking\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

order\_id UUID NOT NULL,

delivery\_partner\_id UUID NOT NULL REFERENCES delivery\_partners(delivery\_partner\_id),

current\_location JSONB,

status VARCHAR(20) NOT NULL CHECK (status IN ('accepted', 'picked\_up', 'in\_transit', 'delivered')),

estimated\_arrival TIMESTAMP,

actual\_arrival TIMESTAMP,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- NEW: Kitchen Display System Table

CREATE TABLE kitchen\_display\_orders (

display\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

order\_id UUID NOT NULL,

menu\_item\_id UUID NOT NULL REFERENCES menu\_items(menu\_item\_id),

quantity INTEGER NOT NULL CHECK (quantity > 0),

special\_instructions TEXT,

status VARCHAR(20) DEFAULT 'pending' CHECK (status IN ('pending', 'preparing', 'ready', 'served')),

station\_assigned VARCHAR(100),

preparation\_start\_time TIMESTAMP,

preparation\_end\_time TIMESTAMP,

chef\_notes TEXT,

priority INTEGER DEFAULT 1 CHECK (priority BETWEEN 1 AND 5),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Indexes for Subsystem 5

CREATE INDEX idx\_orders\_partition\_restaurant ON orders(restaurant\_id, created\_at);

CREATE INDEX idx\_orders\_partition\_status ON orders(status, created\_at);

CREATE INDEX idx\_orders\_partition\_date ON orders(created\_at);

CREATE INDEX idx\_orders\_active ON orders(restaurant\_id, created\_at) WHERE status NOT IN ('cancelled', 'delivered');

CREATE INDEX idx\_orders\_kitchen\_status ON orders(restaurant\_id, status, created\_at) WHERE status IN ('confirmed', 'preparing');

CREATE INDEX idx\_sales\_orders\_customer ON sales\_orders(customer\_user\_id);

CREATE INDEX idx\_sales\_orders\_subtype ON sales\_orders(order\_subtype);

CREATE INDEX idx\_sales\_orders\_table ON sales\_orders(table\_id);

CREATE INDEX idx\_sales\_orders\_waiter ON sales\_orders(assigned\_waiter\_id);

CREATE INDEX idx\_sales\_orders\_batch ON sales\_orders(batch\_id);

CREATE INDEX idx\_sales\_orders\_customer\_date ON sales\_orders(customer\_user\_id);

CREATE INDEX idx\_sales\_orders\_delivery\_partner ON sales\_orders(delivery\_partner\_id);

CREATE INDEX idx\_supply\_orders\_supplier ON supply\_orders(supplier\_id);

CREATE INDEX idx\_supply\_orders\_delivery\_date ON supply\_orders(expected\_delivery\_date);

CREATE INDEX idx\_supply\_orders\_delivery\_status ON supply\_orders(delivery\_status);

CREATE INDEX idx\_order\_items\_order ON order\_items(order\_id);

CREATE INDEX idx\_order\_items\_source\_entity ON order\_items(source\_entity\_id, source\_entity\_type);

CREATE INDEX idx\_order\_items\_source\_type ON order\_items(source\_entity\_type, source\_entity\_id);

CREATE INDEX idx\_order\_item\_rejections\_order\_item ON order\_item\_rejections(order\_item\_id);

CREATE INDEX idx\_order\_item\_rejections\_date ON order\_item\_rejections(rejected\_at);

CREATE INDEX idx\_delivery\_batches\_restaurant ON delivery\_batches(restaurant\_id);

CREATE INDEX idx\_delivery\_batches\_status ON delivery\_batches(batch\_status);

CREATE INDEX idx\_delivery\_batches\_waiter ON delivery\_batches(assigned\_waiter\_id);

CREATE INDEX idx\_delivery\_batches\_created\_at ON delivery\_batches(created\_at);

CREATE INDEX idx\_delivery\_batches\_efficiency ON delivery\_batches(batch\_efficiency\_score);

CREATE INDEX idx\_delivery\_batches\_distance ON delivery\_batches(total\_distance);

CREATE INDEX idx\_delivery\_batches\_completion ON delivery\_batches(estimated\_completion\_time, actual\_completion\_time);

CREATE INDEX idx\_delivery\_partners\_active ON delivery\_partners(is\_active);

CREATE INDEX idx\_delivery\_partners\_type ON delivery\_partners(partner\_type);

CREATE INDEX idx\_delivery\_partners\_rating ON delivery\_partners(average\_rating);

CREATE INDEX idx\_delivery\_tracking\_order ON delivery\_tracking(order\_id);

CREATE INDEX idx\_delivery\_tracking\_partner ON delivery\_tracking(delivery\_partner\_id);

CREATE INDEX idx\_delivery\_tracking\_status ON delivery\_tracking(status);

CREATE INDEX idx\_delivery\_tracking\_estimated\_arrival ON delivery\_tracking(estimated\_arrival);

CREATE INDEX idx\_delivery\_tracking\_active ON delivery\_tracking(status, estimated\_arrival) WHERE status IN ('accepted', 'picked\_up', 'in\_transit');

CREATE INDEX idx\_kitchen\_orders\_status ON kitchen\_display\_orders(status);

CREATE INDEX idx\_kitchen\_orders\_station ON kitchen\_display\_orders(station\_assigned);

CREATE INDEX idx\_kitchen\_orders\_priority ON kitchen\_display\_orders(priority, created\_at);

CREATE INDEX idx\_kitchen\_orders\_preparation ON kitchen\_display\_orders(preparation\_start\_time, preparation\_end\_time);

-- Subsystem 6: Financial & Accounting Ledger

-- Partitioned Transactions Table

CREATE TABLE transactions (

transaction\_id UUID NOT NULL,

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

source\_entity\_id UUID NOT NULL,

source\_entity\_type VARCHAR(50) NOT NULL CHECK (source\_entity\_type IN ('order', 'booking', 'account\_deposit', 'supplier\_payment', 'refund')),

amount DECIMAL(12,2) NOT NULL,

transaction\_type VARCHAR(10) NOT NULL CHECK (transaction\_type IN ('debit', 'credit')),

category VARCHAR(100) NOT NULL,

payment\_method\_id UUID,

gateway\_transaction\_id VARCHAR(255),

status VARCHAR(20) DEFAULT 'completed' CHECK (status IN ('pending', 'completed', 'failed', 'refunded')),

transaction\_date TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

created\_at TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

notes TEXT,

PRIMARY KEY (transaction\_id, transaction\_date)

) PARTITION BY RANGE (transaction\_date);

-- Create monthly partitions for transactions

CREATE TABLE transactions\_2024\_01 PARTITION OF transactions

FOR VALUES FROM ('2024-01-01') TO ('2024-02-01');

CREATE TABLE transactions\_2024\_02 PARTITION OF transactions

FOR VALUES FROM ('2024-02-01') TO ('2024-03-01');

CREATE TABLE transactions\_default PARTITION OF transactions DEFAULT;

CREATE TABLE billing\_records (

billing\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

order\_id UUID UNIQUE NOT NULL,

subtotal\_amount DECIMAL(12,2) NOT NULL CHECK (subtotal\_amount >= 0),

tax\_amount DECIMAL(12,2) DEFAULT 0 CHECK (tax\_amount >= 0),

service\_charge DECIMAL(12,2) DEFAULT 0 CHECK (service\_charge >= 0),

discount\_amount DECIMAL(12,2) DEFAULT 0 CHECK (discount\_amount >= 0),

total\_amount DECIMAL(12,2) NOT NULL CHECK (total\_amount >= 0),

billing\_status VARCHAR(20) DEFAULT 'pending' CHECK (billing\_status IN ('pending', 'paid', 'partially\_paid', 'refunded')),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE customer\_accounts (

account\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_id UUID NOT NULL REFERENCES users(user\_id) ON DELETE CASCADE,

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

balance DECIMAL(12,2) DEFAULT 0 CHECK (balance >= 0),

account\_type VARCHAR(20) NOT NULL CHECK (account\_type IN ('fiat', 'crypto')),

is\_refundable BOOLEAN DEFAULT false,

crypto\_details JSONB,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE payment\_methods (

payment\_method\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_id UUID NOT NULL REFERENCES users(user\_id) ON DELETE CASCADE,

method\_type VARCHAR(20) NOT NULL CHECK (method\_type IN ('card', 'mobile\_money', 'crypto\_wallet', 'account')),

provider VARCHAR(100) NOT NULL,

last\_four\_digits VARCHAR(4),

is\_default BOOLEAN DEFAULT false,

is\_active BOOLEAN DEFAULT true,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Indexes for Subsystem 6

CREATE INDEX idx\_transactions\_partition\_restaurant ON transactions(restaurant\_id, transaction\_date);

CREATE INDEX idx\_transactions\_partition\_date ON transactions(transaction\_date);

CREATE INDEX idx\_transactions\_partition\_type ON transactions(transaction\_type, transaction\_date);

CREATE INDEX idx\_transactions\_source\_entity ON transactions(source\_entity\_id, source\_entity\_type);

CREATE INDEX idx\_transactions\_restaurant\_category\_date ON transactions(restaurant\_id, category, transaction\_date);

CREATE INDEX idx\_transactions\_amount ON transactions(amount);

CREATE INDEX idx\_billing\_records\_order ON billing\_records(order\_id);

CREATE INDEX idx\_billing\_records\_status ON billing\_records(billing\_status);

CREATE INDEX idx\_billing\_records\_created\_at ON billing\_records(created\_at);

CREATE INDEX idx\_customer\_accounts\_user ON customer\_accounts(user\_id);

CREATE INDEX idx\_customer\_accounts\_restaurant ON customer\_accounts(restaurant\_id);

CREATE INDEX idx\_customer\_accounts\_balance ON customer\_accounts(balance);

CREATE INDEX idx\_customer\_accounts\_type ON customer\_accounts(account\_type);

CREATE INDEX idx\_payment\_methods\_user ON payment\_methods(user\_id);

CREATE INDEX idx\_payment\_methods\_type ON payment\_methods(method\_type);

CREATE INDEX idx\_payment\_methods\_active ON payment\_methods(is\_active);

CREATE INDEX idx\_payment\_methods\_default ON payment\_methods(user\_id, is\_default) WHERE is\_default = true;

-- Subsystem 7: Communication & HR Management

CREATE TABLE restaurant\_staff (

staff\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_role\_id UUID UNIQUE NOT NULL REFERENCES user\_roles(user\_role\_id) ON DELETE CASCADE,

employee\_id VARCHAR(50),

hire\_date DATE NOT NULL,

termination\_date DATE,

salary DECIMAL(10,2),

efficiency\_score DECIMAL(4,2) DEFAULT 0.00,

current\_status VARCHAR(20) DEFAULT 'available' CHECK (current\_status IN ('available', 'busy', 'on\_break', 'offline')),

managed\_by\_id UUID REFERENCES restaurant\_staff(staff\_id),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

total\_orders\_served INTEGER DEFAULT 0,

average\_order\_time INTEGER DEFAULT 0,

customer\_rating\_avg DECIMAL(3,2) DEFAULT 0.00,

last\_performance\_review DATE

);

CREATE TABLE staff\_shifts (

shift\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

shift\_name VARCHAR(100) NOT NULL,

shift\_type VARCHAR(20) NOT NULL CHECK (shift\_type IN ('morning', 'afternoon', 'evening', 'night')),

shift\_start TIME NOT NULL,

shift\_end TIME NOT NULL,

max\_staff\_count INTEGER NOT NULL CHECK (max\_staff\_count > 0),

is\_active BOOLEAN DEFAULT true,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE staff\_shift\_assignments (

assignment\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

staff\_id UUID NOT NULL REFERENCES restaurant\_staff(staff\_id) ON DELETE CASCADE,

shift\_id UUID NOT NULL REFERENCES staff\_shifts(shift\_id) ON DELETE CASCADE,

assignment\_date DATE NOT NULL,

actual\_start\_time TIMESTAMP,

actual\_end\_time TIMESTAMP,

status VARCHAR(20) DEFAULT 'scheduled' CHECK (status IN ('scheduled', 'checked\_in', 'checked\_out', 'cancelled')),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE table\_assignments (

assignment\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

staff\_id UUID NOT NULL REFERENCES restaurant\_staff(staff\_id) ON DELETE CASCADE,

table\_id UUID NOT NULL REFERENCES restaurant\_tables(table\_id) ON DELETE CASCADE,

shift\_assignment\_id UUID NOT NULL REFERENCES staff\_shift\_assignments(assignment\_id) ON DELETE CASCADE,

assignment\_start TIMESTAMP NOT NULL,

assignment\_end TIMESTAMP,

status VARCHAR(20) DEFAULT 'active' CHECK (status IN ('active', 'completed')),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE staff\_performance\_history (

performance\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

staff\_id UUID NOT NULL REFERENCES restaurant\_staff(staff\_id) ON DELETE CASCADE,

metric\_type VARCHAR(50) NOT NULL CHECK (metric\_type IN ('delivery\_time', 'customer\_rating', 'order\_accuracy', 'efficiency', 'sales')),

metric\_value DECIMAL(8,2) NOT NULL,

target\_value DECIMAL(8,2),

measured\_at TIMESTAMP NOT NULL,

period\_type VARCHAR(20) NOT NULL CHECK (period\_type IN ('instant', 'daily', 'weekly', 'monthly')),

notes TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE communication\_groups (

group\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

group\_type VARCHAR(50) NOT NULL CHECK (group\_type IN ('internal\_staff', 'public\_community', 'customer\_support')),

name VARCHAR(255) NOT NULL,

description TEXT,

is\_active BOOLEAN DEFAULT true,

created\_by UUID REFERENCES users(user\_id),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE group\_members (

member\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

group\_id UUID NOT NULL REFERENCES communication\_groups(group\_id) ON DELETE CASCADE,

user\_id UUID NOT NULL REFERENCES users(user\_id) ON DELETE CASCADE,

member\_role VARCHAR(20) DEFAULT 'member' CHECK (member\_role IN ('member', 'admin', 'moderator')),

joined\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

left\_at TIMESTAMP,

UNIQUE(group\_id, user\_id)

);

CREATE TABLE chat\_sessions (

session\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

customer\_user\_id UUID NOT NULL REFERENCES users(user\_id),

assigned\_staff\_id UUID REFERENCES restaurant\_staff(staff\_id),

session\_type VARCHAR(50) NOT NULL CHECK (session\_type IN ('customer\_service', 'order\_support', 'complaint', 'general')),

title VARCHAR(255),

status VARCHAR(20) DEFAULT 'active' CHECK (status IN ('active', 'waiting', 'resolved', 'closed')),

priority VARCHAR(20) DEFAULT 'normal' CHECK (priority IN ('low', 'normal', 'high', 'urgent')),

first\_response\_time INTEGER,

resolution\_time INTEGER,

customer\_satisfaction\_rating INTEGER CHECK (customer\_satisfaction\_rating >= 1 AND customer\_satisfaction\_rating <= 5),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

resolved\_at TIMESTAMP,

closed\_at TIMESTAMP

);

-- Partitioned Chat Messages Table

CREATE TABLE chat\_messages (

message\_id UUID NOT NULL,

session\_id UUID REFERENCES chat\_sessions(session\_id) ON DELETE CASCADE,

sender\_id UUID NOT NULL REFERENCES users(user\_id),

recipient\_entity\_type VARCHAR(20) NOT NULL CHECK (recipient\_entity\_type IN ('user', 'group', 'restaurant')),

recipient\_entity\_id UUID NOT NULL,

message\_content TEXT NOT NULL,

message\_type VARCHAR(20) DEFAULT 'text' CHECK (message\_type IN ('text', 'image', 'video', 'system', 'order\_update')),

priority VARCHAR(20) DEFAULT 'normal' CHECK (priority IN ('normal', 'high', 'urgent')),

is\_edited BOOLEAN DEFAULT false,

edited\_at TIMESTAMP,

read\_receipts JSONB,

delivered\_at TIMESTAMP,

sentiment\_score DECIMAL(3,2),

like\_count INTEGER DEFAULT 0,

reply\_count INTEGER DEFAULT 0,

share\_count INTEGER DEFAULT 0,

is\_active BOOLEAN DEFAULT true,

created\_at TIMESTAMP NOT NULL DEFAULT CURRENT\_TIMESTAMP,

PRIMARY KEY (message\_id, created\_at)

) PARTITION BY RANGE (created\_at);

-- Create monthly partitions for chat\_messages

CREATE TABLE chat\_messages\_2024\_01 PARTITION OF chat\_messages

FOR VALUES FROM ('2024-01-01') TO ('2024-02-01');

CREATE TABLE chat\_messages\_default PARTITION OF chat\_messages DEFAULT;

CREATE TABLE notifications (

notification\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

recipient\_id UUID NOT NULL REFERENCES users(user\_id),

source\_entity\_id UUID NOT NULL,

source\_entity\_type VARCHAR(50) NOT NULL CHECK (source\_entity\_type IN ('order', 'message', 'booking', 'batch', 'feedback', 'promotion')),

notification\_type VARCHAR(100) NOT NULL,

message TEXT NOT NULL,

is\_read BOOLEAN DEFAULT false,

action\_url VARCHAR(500),

sent\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

read\_at TIMESTAMP

);

-- Indexes for Subsystem 7

CREATE INDEX idx\_staff\_user\_role ON restaurant\_staff(user\_role\_id);

CREATE INDEX idx\_staff\_status ON restaurant\_staff(current\_status);

CREATE INDEX idx\_staff\_efficiency ON restaurant\_staff(efficiency\_score);

CREATE INDEX idx\_staff\_managed\_by ON restaurant\_staff(managed\_by\_id);

CREATE INDEX idx\_staff\_hire\_date ON restaurant\_staff(hire\_date);

CREATE INDEX idx\_staff\_active\_optimized ON restaurant\_staff(current\_status, efficiency\_score) WHERE current\_status IN ('available', 'busy');

CREATE INDEX idx\_staff\_performance ON restaurant\_staff(total\_orders\_served, customer\_rating\_avg);

CREATE INDEX idx\_shifts\_restaurant ON staff\_shifts(restaurant\_id);

CREATE INDEX idx\_shifts\_type ON staff\_shifts(shift\_type);

CREATE INDEX idx\_shifts\_active ON staff\_shifts(is\_active);

CREATE INDEX idx\_shifts\_times ON staff\_shifts(shift\_start, shift\_end);

CREATE INDEX idx\_shift\_assignments\_staff ON staff\_shift\_assignments(staff\_id);

CREATE INDEX idx\_shift\_assignments\_shift ON staff\_shift\_assignments(shift\_id);

CREATE INDEX idx\_shift\_assignments\_date ON staff\_shift\_assignments(assignment\_date);

CREATE INDEX idx\_shift\_assignments\_status ON staff\_shift\_assignments(status);

CREATE INDEX idx\_shift\_assignments\_date\_status ON staff\_shift\_assignments(assignment\_date, status);

CREATE INDEX idx\_shift\_assignments\_staff\_date ON staff\_shift\_assignments(staff\_id, assignment\_date);

CREATE INDEX idx\_table\_assignments\_staff ON table\_assignments(staff\_id);

CREATE INDEX idx\_table\_assignments\_table ON table\_assignments(table\_id);

CREATE INDEX idx\_table\_assignments\_shift ON table\_assignments(shift\_assignment\_id);

CREATE INDEX idx\_table\_assignments\_status ON table\_assignments(status);

CREATE INDEX idx\_table\_assignments\_time\_range ON table\_assignments(assignment\_start, assignment\_end);

CREATE INDEX idx\_performance\_staff ON staff\_performance\_history(staff\_id);

CREATE INDEX idx\_performance\_metric\_type ON staff\_performance\_history(metric\_type);

CREATE INDEX idx\_performance\_measured\_at ON staff\_performance\_history(measured\_at);

CREATE INDEX idx\_performance\_period ON staff\_performance\_history(period\_type, measured\_at);

CREATE INDEX idx\_communication\_groups\_restaurant ON communication\_groups(restaurant\_id);

CREATE INDEX idx\_communication\_groups\_type ON communication\_groups(group\_type);

CREATE INDEX idx\_communication\_groups\_active ON communication\_groups(is\_active);

CREATE INDEX idx\_group\_members\_group ON group\_members(group\_id);

CREATE INDEX idx\_group\_members\_user ON group\_members(user\_id);

CREATE INDEX idx\_group\_members\_role ON group\_members(member\_role);

CREATE INDEX idx\_chat\_sessions\_restaurant ON chat\_sessions(restaurant\_id);

CREATE INDEX idx\_chat\_sessions\_customer ON chat\_sessions(customer\_user\_id);

CREATE INDEX idx\_chat\_sessions\_staff ON chat\_sessions(assigned\_staff\_id);

CREATE INDEX idx\_chat\_sessions\_status ON chat\_sessions(status);

CREATE INDEX idx\_chat\_sessions\_priority ON chat\_sessions(priority);

CREATE INDEX idx\_chat\_sessions\_created\_at ON chat\_sessions(created\_at);

CREATE INDEX idx\_chat\_sessions\_active ON chat\_sessions(restaurant\_id, status) WHERE status IN ('active', 'waiting');

CREATE INDEX idx\_chat\_messages\_partition\_session ON chat\_messages(session\_id, created\_at);

CREATE INDEX idx\_chat\_messages\_partition\_sender ON chat\_messages(sender\_id, created\_at);

CREATE INDEX idx\_chat\_messages\_partition\_date ON chat\_messages(created\_at);

CREATE INDEX idx\_chat\_messages\_type ON chat\_messages(message\_type);

CREATE INDEX idx\_chat\_messages\_session\_date ON chat\_messages(session\_id, created\_at);

CREATE INDEX idx\_chat\_messages\_content\_search ON chat\_messages USING gin (message\_content gin\_trgm\_ops);

CREATE INDEX idx\_notifications\_recipient ON notifications(recipient\_id);

CREATE INDEX idx\_notifications\_read ON notifications(is\_read);

CREATE INDEX idx\_notifications\_sent\_at ON notifications(sent\_at);

CREATE INDEX idx\_notifications\_source ON notifications(source\_entity\_type, source\_entity\_id);

CREATE INDEX idx\_notifications\_recipient\_read ON notifications(recipient\_id, is\_read) WHERE is\_read = false;

-- Subsystem 8: Customer Engagement & Analytics

CREATE TABLE customer\_loyalty (

loyalty\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

customer\_user\_id UUID NOT NULL REFERENCES users(user\_id) ON DELETE CASCADE,

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

loyalty\_tier VARCHAR(20) DEFAULT 'bronze' CHECK (loyalty\_tier IN ('bronze', 'silver', 'gold')),

points\_balance INTEGER DEFAULT 0 CHECK (points\_balance >= 0),

lifetime\_spend DECIMAL(12,2) DEFAULT 0 CHECK (lifetime\_spend >= 0),

total\_orders INTEGER DEFAULT 0 CHECK (total\_orders >= 0),

visit\_count INTEGER DEFAULT 0 CHECK (visit\_count >= 0),

first\_visit\_date TIMESTAMP,

last\_visit\_date TIMESTAMP,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

UNIQUE(customer\_user\_id, restaurant\_id)

);

-- NEW: Loyalty Rewards Table

CREATE TABLE loyalty\_rewards (

reward\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

reward\_name VARCHAR(255) NOT NULL,

reward\_description TEXT,

points\_cost INTEGER NOT NULL CHECK (points\_cost >= 0),

tier\_required VARCHAR(20) DEFAULT 'bronze' CHECK (tier\_required IN ('bronze', 'silver', 'gold')),

reward\_type VARCHAR(50) NOT NULL CHECK (reward\_type IN ('discount', 'free\_item', 'priority\_service', 'exclusive\_access')),

discount\_percentage DECIMAL(5,2) CHECK (discount\_percentage BETWEEN 0 AND 100),

free\_menu\_item\_id UUID REFERENCES menu\_items(menu\_item\_id),

is\_active BOOLEAN DEFAULT true,

max\_redemptions\_per\_customer INTEGER DEFAULT 1,

start\_date TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

end\_date TIMESTAMP,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- NEW: Reward Redemptions Table

CREATE TABLE reward\_redemptions (

redemption\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

loyalty\_id UUID NOT NULL REFERENCES customer\_loyalty(loyalty\_id) ON DELETE CASCADE,

reward\_id UUID NOT NULL REFERENCES loyalty\_rewards(reward\_id) ON DELETE CASCADE,

order\_id UUID,

points\_used INTEGER NOT NULL CHECK (points\_used >= 0),

redemption\_value DECIMAL(10,2),

redeemed\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

status VARCHAR(20) DEFAULT 'redeemed' CHECK (status IN ('redeemed', 'applied', 'expired', 'cancelled')),

applied\_at TIMESTAMP,

applied\_by UUID REFERENCES users(user\_id)

);

CREATE TABLE bookings (

booking\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

customer\_user\_id UUID NOT NULL REFERENCES users(user\_id),

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

table\_id UUID NOT NULL REFERENCES restaurant\_tables(table\_id),

booking\_date DATE NOT NULL,

start\_time TIME NOT NULL,

end\_time TIME NOT NULL,

party\_size INTEGER NOT NULL CHECK (party\_size > 0),

status VARCHAR(20) DEFAULT 'pending' CHECK (status IN ('pending', 'confirmed', 'checked\_in', 'completed', 'cancelled', 'no\_show')),

deposit\_amount DECIMAL(10,2) DEFAULT 0 CHECK (deposit\_amount >= 0),

deposit\_status VARCHAR(20) DEFAULT 'pending' CHECK (deposit\_status IN ('pending', 'paid', 'refunded', 'forfeited')),

special\_requests TEXT,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE feedback (

feedback\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

customer\_user\_id UUID NOT NULL REFERENCES users(user\_id),

target\_type VARCHAR(20) NOT NULL CHECK (target\_type IN ('restaurant', 'waiter', 'food', 'app')),

target\_id UUID,

order\_id UUID,

rating INTEGER NOT NULL CHECK (rating >= 1 AND rating <= 5),

title VARCHAR(255),

comments TEXT,

keyword\_tags TEXT[],

is\_verified BOOLEAN DEFAULT false,

helpful\_count INTEGER DEFAULT 0,

status VARCHAR(20) DEFAULT 'active' CHECK (status IN ('active', 'flagged', 'removed')),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE announcements (

announcement\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

title VARCHAR(255) NOT NULL,

body TEXT NOT NULL,

announcement\_type VARCHAR(50) NOT NULL CHECK (announcement\_type IN ('promotion', 'event', 'update', 'maintenance')),

target\_audience VARCHAR(50) DEFAULT 'all' CHECK (target\_audience IN ('all', 'loyalty\_tier', 'specific\_group')),

audience\_parameters JSONB,

start\_date TIMESTAMP NOT NULL,

end\_date TIMESTAMP NOT NULL,

is\_public BOOLEAN DEFAULT true,

view\_count INTEGER DEFAULT 0,

engagement\_count INTEGER DEFAULT 0,

created\_by UUID REFERENCES users(user\_id),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE restaurant\_daily\_summary (

summary\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

restaurant\_id UUID NOT NULL REFERENCES restaurants(restaurant\_id) ON DELETE CASCADE,

summary\_date DATE NOT NULL,

total\_orders INTEGER DEFAULT 0,

total\_revenue DECIMAL(12,2) DEFAULT 0,

average\_order\_value DECIMAL(10,2) DEFAULT 0,

customer\_count INTEGER DEFAULT 0,

peak\_hour INTEGER,

most\_popular\_item UUID REFERENCES menu\_items(menu\_item\_id),

customer\_satisfaction\_score DECIMAL(3,2) DEFAULT 0.00,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

UNIQUE(restaurant\_id, summary\_date)

);

-- Enhanced Content Media Table with Analytics

CREATE TABLE content\_media (

media\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

target\_entity\_id UUID NOT NULL,

target\_entity\_type VARCHAR(50) NOT NULL CHECK (target\_entity\_type IN ('menu\_item', 'announcement', 'order\_item', 'feedback', 'restaurant', 'user')),

media\_type VARCHAR(20) NOT NULL CHECK (media\_type IN ('image', 'video', 'audio', 'document')),

media\_url VARCHAR(500) NOT NULL,

thumbnail\_url VARCHAR(500),

caption VARCHAR(500),

alt\_text VARCHAR(500),

file\_size INTEGER,

duration INTEGER,

display\_order INTEGER DEFAULT 0,

view\_count INTEGER DEFAULT 0,

like\_count INTEGER DEFAULT 0,

share\_count INTEGER DEFAULT 0,

comment\_count INTEGER DEFAULT 0,

is\_active BOOLEAN DEFAULT true,

created\_by UUID REFERENCES users(user\_id),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

conversion\_rate DECIMAL(5,4) DEFAULT 0,

engagement\_rate DECIMAL(5,4) DEFAULT 0,

click\_through\_rate DECIMAL(5,4) DEFAULT 0,

average\_view\_duration INTEGER DEFAULT 0,

media\_quality\_score DECIMAL(3,2) DEFAULT 0.00

);

-- NEW: Media Performance Daily Analytics Table

CREATE TABLE media\_performance\_daily (

performance\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

media\_id UUID NOT NULL REFERENCES content\_media(media\_id) ON DELETE CASCADE,

performance\_date DATE NOT NULL,

view\_count INTEGER DEFAULT 0,

like\_count INTEGER DEFAULT 0,

share\_count INTEGER DEFAULT 0,

comment\_count INTEGER DEFAULT 0,

click\_count INTEGER DEFAULT 0,

order\_count INTEGER DEFAULT 0,

conversion\_rate DECIMAL(5,4) DEFAULT 0,

engagement\_rate DECIMAL(5,4) DEFAULT 0,

average\_view\_duration INTEGER DEFAULT 0,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

UNIQUE(media\_id, performance\_date)

);

CREATE TABLE content\_likes (

like\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_id UUID NOT NULL REFERENCES users(user\_id),

target\_entity\_id UUID NOT NULL,

target\_entity\_type VARCHAR(50) NOT NULL CHECK (target\_entity\_type IN ('media', 'comment', 'announcement', 'feedback', 'restaurant', 'message')),

reaction\_type VARCHAR(20) DEFAULT 'like' CHECK (reaction\_type IN ('like', 'love', 'laugh', 'wow', 'sad', 'angry', 'celebrate', 'support')),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

UNIQUE(user\_id, target\_entity\_id, target\_entity\_type)

);

CREATE TABLE content\_comments (

comment\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

user\_id UUID NOT NULL REFERENCES users(user\_id),

target\_entity\_id UUID NOT NULL,

target\_entity\_type VARCHAR(50) NOT NULL CHECK (target\_entity\_type IN ('media', 'announcement', 'feedback', 'restaurant', 'order')),

comment\_text TEXT NOT NULL,

parent\_comment\_id UUID REFERENCES content\_comments(comment\_id),

depth INTEGER DEFAULT 0,

like\_count INTEGER DEFAULT 0,

reply\_count INTEGER DEFAULT 0,

is\_edited BOOLEAN DEFAULT false,

edited\_at TIMESTAMP,

sentiment\_score DECIMAL(3,2),

is\_active BOOLEAN DEFAULT true,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE tags (

tag\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

tag\_name VARCHAR(100) UNIQUE NOT NULL,

tag\_type VARCHAR(50) NOT NULL CHECK (tag\_type IN ('cuisine', 'ingredient', 'style', 'location', 'sentiment', 'topic', 'custom')),

description TEXT,

color\_hex VARCHAR(7),

is\_system\_tag BOOLEAN DEFAULT false,

is\_active BOOLEAN DEFAULT true,

created\_by UUID REFERENCES users(user\_id),

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

updated\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

CREATE TABLE taggings (

tagging\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

tag\_id UUID NOT NULL REFERENCES tags(tag\_id) ON DELETE CASCADE,

target\_entity\_id UUID NOT NULL,

target\_entity\_type VARCHAR(50) NOT NULL CHECK (target\_entity\_type IN ('media', 'comment', 'message', 'announcement', 'feedback', 'user', 'restaurant')),

context VARCHAR(50) DEFAULT 'user\_generated' CHECK (context IN ('user\_generated', 'ai\_generated', 'system', 'moderator')),

confidence\_score DECIMAL(3,2),

tagged\_by UUID REFERENCES users(user\_id),

tagged\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP,

expires\_at TIMESTAMP

);

CREATE TABLE content\_shares (

share\_id UUID PRIMARY KEY DEFAULT gen\_random\_uuid(),

target\_entity\_id UUID NOT NULL,

target\_entity\_type VARCHAR(50) NOT NULL CHECK (target\_entity\_type IN ('media', 'announcement', 'feedback', 'restaurant', 'message')),

shared\_by UUID NOT NULL REFERENCES users(user\_id),

share\_platform VARCHAR(50) NOT NULL CHECK (share\_platform IN ('facebook', 'twitter', 'instagram', 'whatsapp', 'internal', 'email', 'other')),

share\_destination VARCHAR(500),

share\_message TEXT,

view\_count INTEGER DEFAULT 0,

engagement\_count INTEGER DEFAULT 0,

created\_at TIMESTAMP DEFAULT CURRENT\_TIMESTAMP

);

-- Indexes for Subsystem 8

CREATE INDEX idx\_loyalty\_customer ON customer\_loyalty(customer\_user\_id);

CREATE INDEX idx\_loyalty\_restaurant ON customer\_loyalty(restaurant\_id);

CREATE INDEX idx\_loyalty\_tier ON customer\_loyalty(loyalty\_tier);

CREATE INDEX idx\_loyalty\_points ON customer\_loyalty(points\_balance);

CREATE INDEX idx\_loyalty\_spend ON customer\_loyalty(lifetime\_spend);

CREATE INDEX idx\_loyalty\_customer\_restaurant ON customer\_loyalty(customer\_user\_id, restaurant\_id);

CREATE INDEX idx\_loyalty\_analytics ON customer\_loyalty(restaurant\_id, loyalty\_tier, lifetime\_spend, total\_orders);

CREATE INDEX idx\_loyalty\_rewards\_restaurant ON loyalty\_rewards(restaurant\_id);

CREATE INDEX idx\_loyalty\_rewards\_tier ON loyalty\_rewards(tier\_required);

CREATE INDEX idx\_loyalty\_rewards\_active ON loyalty\_rewards(is\_active) WHERE is\_active = true;

CREATE INDEX idx\_loyalty\_rewards\_dates ON loyalty\_rewards(start\_date, end\_date);

CREATE INDEX idx\_reward\_redemptions\_loyalty ON reward\_redemptions(loyalty\_id);

CREATE INDEX idx\_reward\_redemptions\_reward ON reward\_redemptions(reward\_id);

CREATE INDEX idx\_reward\_redemptions\_order ON reward\_redemptions(order\_id);

CREATE INDEX idx\_reward\_redemptions\_date ON reward\_redemptions(redeemed\_at);

CREATE INDEX idx\_bookings\_restaurant ON bookings(restaurant\_id);

CREATE INDEX idx\_bookings\_customer ON bookings(customer\_user\_id);

CREATE INDEX idx\_bookings\_table ON bookings(table\_id);

CREATE INDEX idx\_bookings\_date ON bookings(booking\_date);

CREATE INDEX idx\_bookings\_status ON bookings(status);

CREATE INDEX idx\_bookings\_restaurant\_date ON bookings(restaurant\_id, booking\_date);

CREATE INDEX idx\_bookings\_customer\_date ON bookings(customer\_user\_id, booking\_date);

CREATE INDEX idx\_bookings\_upcoming ON bookings(restaurant\_id, booking\_date, status) WHERE booking\_date >= CURRENT\_DATE AND status IN ('pending', 'confirmed');

CREATE INDEX idx\_feedback\_restaurant ON feedback(restaurant\_id);

CREATE INDEX idx\_feedback\_customer ON feedback(customer\_user\_id);

CREATE INDEX idx\_feedback\_target ON feedback(target\_type, target\_id);

CREATE INDEX idx\_feedback\_rating ON feedback(rating);

CREATE INDEX idx\_feedback\_verified ON feedback(is\_verified);

CREATE INDEX idx\_feedback\_status ON feedback(status);

CREATE INDEX idx\_feedback\_created\_at ON feedback(created\_at);

CREATE INDEX idx\_feedback\_restaurant\_rating ON feedback(restaurant\_id, rating);

CREATE INDEX idx\_feedback\_keyword\_tags ON feedback USING gin (keyword\_tags);

CREATE INDEX idx\_feedback\_comments\_search ON feedback USING gin (comments gin\_trgm\_ops);

CREATE INDEX idx\_announcements\_restaurant ON announcements(restaurant\_id);

CREATE INDEX idx\_announcements\_type ON announcements(announcement\_type);

CREATE INDEX idx\_announcements\_dates ON announcements(start\_date, end\_date);

CREATE INDEX idx\_announcements\_public ON announcements(is\_public);

CREATE INDEX idx\_announcements\_active ON announcements(start\_date, end\_date) WHERE start\_date <= CURRENT\_TIMESTAMP AND end\_date >= CURRENT\_TIMESTAMP;

CREATE INDEX idx\_announcements\_engagement ON announcements(engagement\_count);

CREATE INDEX idx\_daily\_summary\_restaurant ON restaurant\_daily\_summary(restaurant\_id);

CREATE INDEX idx\_daily\_summary\_date ON restaurant\_daily\_summary(summary\_date);

CREATE INDEX idx\_daily\_summary\_restaurant\_date ON restaurant\_daily\_summary(restaurant\_id, summary\_date);

CREATE INDEX idx\_daily\_summary\_revenue ON restaurant\_daily\_summary(total\_revenue);

CREATE INDEX idx\_content\_media\_target ON content\_media(target\_entity\_id, target\_entity\_type);

CREATE INDEX idx\_content\_media\_conversion ON content\_media(conversion\_rate) WHERE conversion\_rate > 0;

CREATE INDEX idx\_content\_media\_engagement ON content\_media(engagement\_rate) WHERE engagement\_rate > 0;

CREATE INDEX idx\_content\_media\_quality ON content\_media(media\_quality\_score);

CREATE INDEX idx\_content\_media\_analytics ON content\_media(target\_entity\_type, target\_entity\_id, like\_count, view\_count);

CREATE INDEX idx\_media\_performance\_date ON media\_performance\_daily(performance\_date);

CREATE INDEX idx\_media\_performance\_media ON media\_performance\_daily(media\_id);

CREATE INDEX idx\_media\_performance\_conversion ON media\_performance\_daily(conversion\_rate);

CREATE INDEX idx\_content\_likes\_target ON content\_likes(target\_entity\_id, target\_entity\_type);

CREATE INDEX idx\_content\_comments\_target ON content\_comments(target\_entity\_id, target\_entity\_type);

CREATE INDEX idx\_taggings\_target ON taggings(target\_entity\_id, target\_entity\_type);

-- Create materialized views for complex queries

CREATE MATERIALIZED VIEW mv\_restaurant\_performance AS

SELECT

r.restaurant\_id,

r.name,

r.cuisine\_type,

COUNT(DISTINCT so.sales\_order\_id) as total\_orders\_30d,

SUM(o.total\_amount) as revenue\_30d,

AVG(f.rating) as avg\_rating\_30d,

COUNT(DISTINCT so.customer\_user\_id) as unique\_customers\_30d

FROM restaurants r

LEFT JOIN orders o ON r.restaurant\_id = o.restaurant\_id

AND o.created\_at >= CURRENT\_DATE - INTERVAL '30 days'

AND o.order\_type = 'sales'

LEFT JOIN sales\_orders so ON o.order\_id = so.order\_id

LEFT JOIN feedback f ON r.restaurant\_id = f.restaurant\_id

AND f.created\_at >= CURRENT\_DATE - INTERVAL '30 days'

WHERE r.status = 'active'

GROUP BY r.restaurant\_id, r.name, r.cuisine\_type;

CREATE UNIQUE INDEX idx\_mv\_restaurant\_performance ON mv\_restaurant\_performance(restaurant\_id);

-- Refresh function for materialized views

CREATE OR REPLACE FUNCTION refresh\_restaurant\_performance()

RETURNS trigger AS $$

BEGIN

REFRESH MATERIALIZED VIEW CONCURRENTLY mv\_restaurant\_performance;

RETURN NULL;

END;

$$ LANGUAGE plpgsql;

-- Additional composite indexes for analytics

CREATE INDEX idx\_menu\_items\_active\_optimized ON menu\_items(menu\_id, display\_order) WHERE is\_available = true;

CREATE INDEX idx\_notifications\_unread\_optimized ON notifications(recipient\_id, sent\_at) WHERE is\_read = false;

-- Expression indexes for JSONB fields

CREATE INDEX idx\_restaurants\_city\_optimized ON restaurants USING gin ((address->>'city'));

CREATE INDEX idx\_restaurants\_delivery\_optimized ON restaurants

USING gist ((

(address->'coordinates'->>'lat')::float,

(address->'coordinates'->>'lng')::float

));

-- Foreign key constraints for partitioned tables

ALTER TABLE sales\_orders ADD CONSTRAINT fk\_sales\_orders\_order

FOREIGN KEY (order\_id) REFERENCES orders(order\_id) ON DELETE CASCADE;

ALTER TABLE supply\_orders ADD CONSTRAINT fk\_supply\_orders\_order

FOREIGN KEY (order\_id) REFERENCES orders(order\_id) ON DELETE CASCADE;

ALTER TABLE order\_items ADD CONSTRAINT fk\_order\_items\_order

FOREIGN KEY (order\_id) REFERENCES orders(order\_id) ON DELETE CASCADE;

ALTER TABLE billing\_records ADD CONSTRAINT fk\_billing\_records\_order

FOREIGN KEY (order\_id) REFERENCES orders(order\_id) ON DELETE CASCADE;

ALTER TABLE delivery\_tracking ADD CONSTRAINT fk\_delivery\_tracking\_order

FOREIGN KEY (order\_id) REFERENCES orders(order\_id) ON DELETE CASCADE;

ALTER TABLE kitchen\_display\_orders ADD CONSTRAINT fk\_kitchen\_orders\_order

FOREIGN KEY (order\_id) REFERENCES orders(order\_id) ON DELETE CASCADE;

ALTER TABLE reward\_redemptions ADD CONSTRAINT fk\_reward\_redemptions\_order

FOREIGN KEY (order\_id) REFERENCES orders(order\_id) ON DELETE SET NULL;

ALTER TABLE feedback ADD CONSTRAINT fk\_feedback\_order

FOREIGN KEY (order\_id) REFERENCES orders(order\_id) ON DELETE SET NULL;