DESIGN DE BANCO DE DADOS - WMS ENTERPRISE

1. Princípios de Design

1.1 Multi-tenancy

- Isolamento total de dados por tenant
- Row Level Security (RLS) no PostgreSQL
- Chave `tenant_id` em todas as tabelas principais
- Particionamento lógico e físico por tenant

1.2 Auditoria Completa

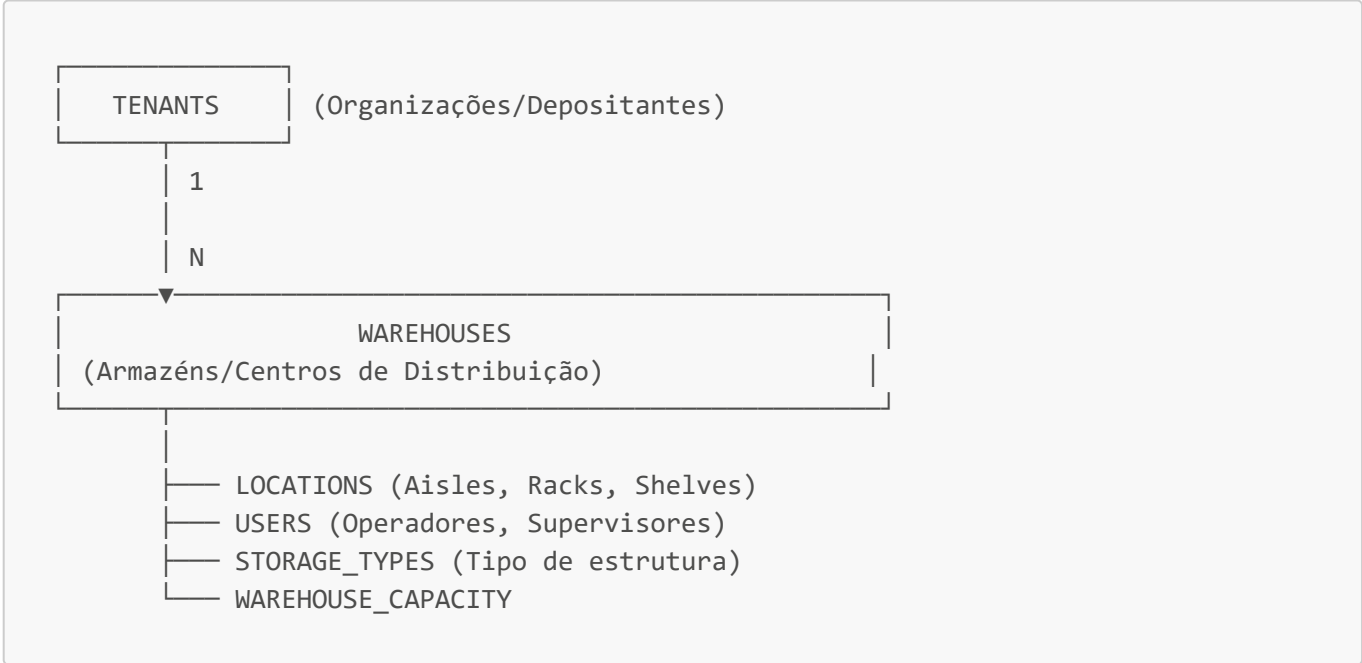
- Todas as alterações registradas em `audit_log`
- Campos: `created_at`, `updated_at`, `created_by`, `updated_by`
- Histórico de mudanças sem perda de dados

1.3 Soft Deletes

- Registros marcados como deletados, não removidos
- Coluna `deleted_at` para controle
- Queries filtram automaticamente deletados

---

2. Diagrama ER (Conceptual)



---

3. Tabelas do Sistema

3.1 Dimensões Organizacionais

TABLE: tenants

```
CREATE TABLE tenants (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    name VARCHAR(255) NOT NULL,
    legal_name VARCHAR(500),
    cnpj VARCHAR(18) UNIQUE,
    email VARCHAR(255),
    phone VARCHAR(20),

    -- Configurações
    max_warehouses INT DEFAULT 5,
    max_users INT DEFAULT 100,
    max_storage_locations INT DEFAULT 100000,
    enable_multi_warehouse BOOLEAN DEFAULT FALSE,
    enable_api_access BOOLEAN DEFAULT TRUE,

    -- Status
    status ENUM ('ACTIVE', 'SUSPENDED', 'DELETED') DEFAULT 'ACTIVE',

    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    deleted_at TIMESTAMP,

    CONSTRAINT check_cnpj_format CHECK (cnpj ~ '^\d{2}\.\d{3}\.\d{3}/\d{4}-\d{2}$'
OR cnpj IS NULL)
);
```

TABLE: warehouses

```
CREATE TABLE warehouses (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,

    -- Informações Básicas
    name VARCHAR(255) NOT NULL,
    code VARCHAR(50) NOT NULL,
    description TEXT,

    -- Localização
    address VARCHAR(500),
    city VARCHAR(100),
    state VARCHAR(2),
    postal_code VARCHAR(10),
    country VARCHAR(3) DEFAULT 'BRA',
    latitude DECIMAL(10,8),
    longitude DECIMAL(11,8),

    -- Capacidade
```

```
    total_positions INT,
    total_weight_capacity DECIMAL(15,2),  -- em kg

    -- Operação
    opening_time TIME,
    closing_time TIME,
    max_workers INT,

    -- Status
    status ENUM ('ACTIVE', 'INACTIVE', 'UNDER_MAINTENANCE') DEFAULT 'ACTIVE',

    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    created_by UUID,
    updated_by UUID,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    UNIQUE(tenant_id, code)
);
```

TABLE: users

```
CREATE TABLE users (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,

    -- Informações Básicas
    email VARCHAR(255) NOT NULL,
    full_name VARCHAR(255) NOT NULL,
    phone VARCHAR(20),
    cpf VARCHAR(14) UNIQUE,

    -- Autenticação
    password_hash VARCHAR(255),
    password_last_changed_at TIMESTAMP,
    mfa_enabled BOOLEAN DEFAULT FALSE,
    mfa_secret VARCHAR(32),

    -- Acesso
    role_id UUID,
    warehouse_ids UUID[] DEFAULT ARRAY[]::UUID[],  -- Array de warehouses
atribuídos

    -- Status
    status ENUM ('ACTIVE', 'INACTIVE', 'PENDING_VERIFICATION', 'LOCKED') DEFAULT
'ACTIVE',
    last_login_at TIMESTAMP,
    login_attempts INT DEFAULT 0,
    locked_until TIMESTAMP,
```

```
    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    deleted_at TIMESTAMP,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    UNIQUE(tenant_id, email)
);

CREATE INDEX idx_users_tenant_id ON users(tenant_id);
```

TABLE: roles

```
CREATE TABLE roles (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,

    name VARCHAR(100) NOT NULL,
    description TEXT,

    -- Permissões armazenadas como JSON para flexibilidade
    permissions JSONB,

    status ENUM ('ACTIVE', 'ARCHIVED') DEFAULT 'ACTIVE',
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    UNIQUE(tenant_id, name)
);

-- Exemplo de roles pré-definidas
-- WAREHOUSE_ADMIN
-- WAREHOUSE_SUPERVISOR
-- INBOUND_OPERATOR
-- PICKING_OPERATOR
-- PACKING_OPERATOR
-- QUALITY_INSPECTOR
-- SHIPPING_MANAGER
-- ANALYTICS_VIEWER
```

3.2 Estrutura de Armazém

TABLE: locations

```
CREATE TABLE locations (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
```

```
    warehouse_id UUID NOT NULL,
    tenant_id UUID NOT NULL,

    -- Identificação
    aisle_code VARCHAR(10),          -- Corredor (A, B, C...)
    rack_code VARCHAR(10),           -- Prateleira (1, 2, 3...)
    shelf_code VARCHAR(10),          -- Nível (1, 2, 3...)
    bin_code VARCHAR(10),            -- Compartimento (A, B, C...)

    location_code VARCHAR(50) GENERATED ALWAYS AS (
        CONCAT(aisle_code, '-', rack_code, '-', shelf_code, '-', bin_code)
    ) STORED,

    -- Características Físicas
    storage_type_id UUID,
    max_weight DECIMAL(12,2),        -- kg
    max_volume DECIMAL(12,2),        -- m³
    height_cm INT,
    width_cm INT,
    depth_cm INT,

    -- Estado
    status ENUM ('AVAILABLE', 'FULL', 'RESERVED', 'BLOCKED', 'MAINTENANCE')
DEFAULT 'AVAILABLE',
    is_available_for_storage BOOLEAN DEFAULT TRUE,
    current_weight DECIMAL(12,2) DEFAULT 0,
    current_volume DECIMAL(12,2) DEFAULT 0,

    -- Regras
    allowed_product_categories UUID[],
    blocked_product_categories UUID[],

    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    FOREIGN KEY (warehouse_id) REFERENCES warehouses(id),
    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (storage_type_id) REFERENCES storage_types(id),

    UNIQUE(warehouse_id, aisle_code, rack_code, shelf_code, bin_code)
);

CREATE INDEX idx_locations_warehouse ON locations(warehouse_id);
CREATE INDEX idx_locations_status ON locations(status);
CREATE INDEX idx_locations_code ON locations(location_code);
```

TABLE: storage_types

```
CREATE TABLE storage_types (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
```

```
    tenant_id UUID NOT NULL,

    name VARCHAR(100),              -- "Rack Convencional", "Cantilever", "Drive-
in"
    code VARCHAR(20) UNIQUE,

    -- Características
    can_store_pallets BOOLEAN DEFAULT TRUE,
    can_store_boxes BOOLEAN DEFAULT TRUE,
    can_store_small_items BOOLEAN DEFAULT FALSE,

    -- Climático
    temperature_min DECIMAL(5,2),
    temperature_max DECIMAL(5,2),
    humidity_control BOOLEAN DEFAULT FALSE,

    -- Operacional
    picking_priority INT,           -- Prioridade de uso (1=máxima)
    utilization_rate DECIMAL(5,2),  -- Taxa de utilização esperada (%)

    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id)
);
```

3.3 Produtos e Inventário

TABLE: skus (Stock Keeping Units)

```
CREATE TABLE skus (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,

    -- Identificação
    sku_code VARCHAR(50) NOT NULL,
    ean13 VARCHAR(13),
    supplier_sku VARCHAR(50),

    -- Descrição
    name VARCHAR(255) NOT NULL,
    description TEXT,
    category_id UUID,

    -- Características Físicas
    weight_kg DECIMAL(12,3),
    volume_m3 DECIMAL(12,6),
    height_cm INT,
    width_cm INT,
    depth_cm INT,
```

```
    -- Classificação ABC
    abc_category ENUM ('A', 'B', 'C') DEFAULT 'C',

    -- Configuração
    storage_type_id UUID,           -- Tipo preferido de armazenagem
    temperature_control_required BOOLEAN DEFAULT FALSE,
    temperature_min DECIMAL(5,2),
    temperature_max DECIMAL(5,2),
    hazmat BOOLEAN DEFAULT FALSE,
    fragile BOOLEAN DEFAULT FALSE,

    -- Validade
    has_expiration_date BOOLEAN DEFAULT FALSE,
    shelf_life_days INT,

    -- Status
    status ENUM ('ACTIVE', 'DISCONTINUED', 'UNDER_REVIEW') DEFAULT 'ACTIVE',

    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (category_id) REFERENCES product_categories(id),
    FOREIGN KEY (storage_type_id) REFERENCES storage_types(id),

    UNIQUE(tenant_id, sku_code)
);

CREATE INDEX idx_skus_ean ON skus(ean13);
CREATE INDEX idx_skus_category ON skus(category_id);
```

TABLE: product_categories

```
CREATE TABLE product_categories (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,

    name VARCHAR(100) NOT NULL,
    code VARCHAR(50),
    description TEXT,
    parent_category_id UUID,

    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (parent_category_id) REFERENCES product_categories(id)
);
```

TABLE: inventory_master

```
CREATE TABLE inventory_master (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,
    warehouse_id UUID NOT NULL,

    sku_id UUID NOT NULL,
    location_id UUID NOT NULL,
    batch_id UUID,

    -- Quantidades
    quantity_on_hand INT DEFAULT 0,
    quantity_reserved INT DEFAULT 0,
    quantity_available INT GENERATED ALWAYS AS (quantity_on_hand -
quantity_reserved) STORED,

    -- Rastreabilidade
    lot_number VARCHAR(100),
    serial_numbers TEXT[],          -- Array de números de série
    expiration_date DATE,
    manufacturing_date DATE,

    -- Status
    status ENUM ('ACTIVE', 'BLOCKED', 'EXPIRED', 'DAMAGED', 'OBSOLETE') DEFAULT
'ACTIVE',

    -- Controle
    last_movement_at TIMESTAMP,
    last_counted_at TIMESTAMP,

    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (warehouse_id) REFERENCES warehouses(id),
    FOREIGN KEY (sku_id) REFERENCES skus(id),
    FOREIGN KEY (location_id) REFERENCES locations(id),

    UNIQUE(warehouse_id, sku_id, location_id, batch_id)
);

CREATE INDEX idx_inventory_sku ON inventory_master(sku_id);
CREATE INDEX idx_inventory_location ON inventory_master(location_id);
CREATE INDEX idx_inventory_expiration ON inventory_master(expiration_date);
CREATE INDEX idx_inventory_status ON inventory_master(status);
```

TABLE: inventory_transactions

```
CREATE TABLE inventory_transactions (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,

    -- Referência
    inventory_master_id UUID NOT NULL,
    document_id UUID,               -- Referência para documento (ASN, Picking
Order, etc)

    -- Tipo de Movimento
    transaction_type ENUM (
        'INBOUND_RECEIPT',
        'OUTBOUND_PICKING',
        'TRANSFER',
        'ADJUSTMENT',
        'COUNT',
        'DISPOSAL',
        'RETURN'
    ),

    -- Valores
    quantity_before INT,
    quantity_after INT,
    quantity_moved INT GENERATED ALWAYS AS (quantity_after - quantity_before)
STORED,

    -- Detalhes
    reason TEXT,
    notes TEXT,
    reference_number VARCHAR(100),

    -- Auditoria
    created_by UUID,
    user_role VARCHAR(50),
    location_from UUID,
    location_to UUID,

    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (inventory_master_id) REFERENCES inventory_master(id),
    FOREIGN KEY (created_by) REFERENCES users(id)
);

CREATE INDEX idx_inventory_transactions_inventory ON
inventory_transactions(inventory_master_id);
CREATE INDEX idx_inventory_transactions_date ON inventory_transactions(created_at
DESC);
CREATE INDEX idx_inventory_transactions_user ON
inventory_transactions(created_by);
```

3.4 Operações Inbound (Recebimento)

TABLE: inbound_asn (Aviso de Recebimento)

```
CREATE TABLE inbound_asn (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,
    warehouse_id UUID NOT NULL,

    -- Identificação
    asn_number VARCHAR(50) NOT NULL,
    po_number VARCHAR(50),
    supplier_id UUID NOT NULL,

    -- Datas
    scheduled_arrival_date DATE,
    actual_arrival_date DATE,

    -- Transportação
    carrier_name VARCHAR(255),
    transport_reference VARCHAR(100),
    vehicle_plate VARCHAR(20),

    -- Quantidades
    expected_lines INT,
    expected_pallets INT,

    -- Status
    status ENUM (
        'DRAFT',
        'SCHEDULED',
        'ARRIVED',
        'RECEIVING_IN_PROGRESS',
        'QUALITY_CHECK',
        'FULLY_RECEIVED',
        'PARTIAL_RECEIVED',
        'CLOSED',
        'CANCELLED'
    ) DEFAULT 'DRAFT',

    -- Controle
    received_lines INT DEFAULT 0,
    rejected_lines INT DEFAULT 0,
    received_quantity INT DEFAULT 0,

    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    received_at TIMESTAMP,
    closed_at TIMESTAMP,
    created_by UUID,
    received_by UUID,
```

```
    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (warehouse_id) REFERENCES warehouses(id),
    FOREIGN KEY (supplier_id) REFERENCES suppliers(id),
    FOREIGN KEY (created_by) REFERENCES users(id),
    FOREIGN KEY (received_by) REFERENCES users(id),

    UNIQUE(tenant_id, asn_number)
);
```

TABLE: inbound_asn_lines

```
CREATE TABLE inbound_asn_lines (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    asn_id UUID NOT NULL,
    tenant_id UUID NOT NULL,

    line_number INT,
    sku_id UUID NOT NULL,

    -- Quantidades
    expected_quantity INT,
    expected_unit VARCHAR(10),
    received_quantity INT DEFAULT 0,
    rejected_quantity INT DEFAULT 0,

    -- Lote/Série
    lot_number VARCHAR(100),
    serial_numbers TEXT[],
    expiration_date DATE,

    -- Status
    status ENUM ('PENDING', 'RECEIVED', 'PARTIALLY_RECEIVED', 'REJECTED',
'CANCELLED') DEFAULT 'PENDING',

    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    received_at TIMESTAMP,

    FOREIGN KEY (asn_id) REFERENCES inbound_asn(id) ON DELETE CASCADE,
    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (sku_id) REFERENCES skus(id)
);
```

TABLE: receiving_operations

```
CREATE TABLE receiving_operations (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
```

```
    asn_id UUID NOT NULL,
    tenant_id UUID NOT NULL,

    -- Operação
    operation_number VARCHAR(50),
    status ENUM ('IN_PROGRESS', 'COMPLETED', 'FAILED', 'CANCELLED') DEFAULT
'IN_PROGRESS',

    -- Qualidade
    quality_check_required BOOLEAN DEFAULT FALSE,
    quality_approved BOOLEAN,
    quality_notes TEXT,
    quality_checked_by UUID,
    quality_checked_at TIMESTAMP,

    -- Localização
    receiving_dock_id UUID,
    staged_location_id UUID,
    final_location_id UUID,

    -- Timestamps
    started_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    completed_at TIMESTAMP,
    started_by UUID,
    completed_by UUID,

    FOREIGN KEY (asn_id) REFERENCES inbound_asn(id),
    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (started_by) REFERENCES users(id),
    FOREIGN KEY (completed_by) REFERENCES users(id)
);
```

3.5 Operações Outbound (Picking, Packing, Shipping)

TABLE: orders

```
CREATE TABLE orders (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,
    warehouse_id UUID NOT NULL,

    -- Identificação
    order_number VARCHAR(50) NOT NULL,
    order_date DATE NOT NULL,
    customer_id UUID NOT NULL,

    -- Datas
    promised_delivery_date DATE,
    ship_by_date DATE,
```

```
    -- Destinatário
    delivery_address VARCHAR(500),
    delivery_city VARCHAR(100),
    delivery_state VARCHAR(2),
    delivery_postal_code VARCHAR(10),

    -- Faturamento
    total_lines INT,
    total_units INT,
    total_weight DECIMAL(12,2),
    total_value DECIMAL(15,2),

    -- Prioridade e Flags
    priority INT DEFAULT 0,          -- 0=normal, 1=urgente, etc
    is_gift BOOLEAN DEFAULT FALSE,
    is_fragile BOOLEAN DEFAULT FALSE,
    requires_signature BOOLEAN DEFAULT FALSE,

    -- Status
    order_status ENUM (
        'NEW',
        'ALLOCATED',
        'READY_TO_PICK',
        'PICKING_IN_PROGRESS',
        'PICKED',
        'PACKING_IN_PROGRESS',
        'PACKED',
        'READY_TO_SHIP',
        'SHIPPED',
        'DELIVERED',
        'CANCELLED',
        'ON_HOLD'
    ) DEFAULT 'NEW',

    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    cancelled_at TIMESTAMP,
    shipped_at TIMESTAMP,
    delivered_at TIMESTAMP,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (warehouse_id) REFERENCES warehouses(id),
    FOREIGN KEY (customer_id) REFERENCES customers(id),

    UNIQUE(tenant_id, order_number)
);

CREATE INDEX idx_orders_status ON orders(order_status);
CREATE INDEX idx_orders_date ON orders(order_date);
CREATE INDEX idx_orders_delivery_date ON orders(promised_delivery_date);
```

TABLE: order_lines

```
CREATE TABLE order_lines (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    order_id UUID NOT NULL,
    tenant_id UUID NOT NULL,

    line_number INT,
    sku_id UUID NOT NULL,

    -- Quantidades
    quantity_ordered INT,
    quantity_allocated INT DEFAULT 0,
    quantity_picked INT DEFAULT 0,
    quantity_packed INT DEFAULT 0,
    quantity_shipped INT DEFAULT 0,

    -- Unit Price
    unit_price DECIMAL(12,2),
    line_total DECIMAL(15,2),

    -- Status
    line_status ENUM ('OPEN', 'ALLOCATED', 'READY_TO_PICK', 'PICKED', 'PACKED',
'SHIPPED', 'CANCELLED') DEFAULT 'OPEN',

    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    updated_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    FOREIGN KEY (order_id) REFERENCES orders(id) ON DELETE CASCADE,
    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (sku_id) REFERENCES skus(id)
);
```

TABLE: picking_orders

```
CREATE TABLE picking_orders (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    order_id UUID NOT NULL,
    tenant_id UUID NOT NULL,
    warehouse_id UUID NOT NULL,

    -- Identificação
    picking_id VARCHAR(50) NOT NULL,
    wave_id UUID,

    -- Atribuição
    assigned_to_user_id UUID,
    assigned_at TIMESTAMP,
```

```
    -- Execução
    started_at TIMESTAMP,
    completed_at TIMESTAMP,

    -- Quantidades
    total_lines INT,
    completed_lines INT DEFAULT 0,

    -- Rota
    suggested_route_coordinates JSONB,  -- JSON com sequência otimizada de
localizações
    distance_meters INT,
    estimated_time_minutes INT,

    -- Status
    status ENUM ('CREATED', 'ASSIGNED', 'IN_PROGRESS', 'COMPLETED', 'CANCELLED')
DEFAULT 'CREATED',

    -- Performance
    actual_distance_meters INT,
    actual_time_minutes INT,
    error_count INT DEFAULT 0,

    FOREIGN KEY (order_id) REFERENCES orders(id),
    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (warehouse_id) REFERENCES warehouses(id),
    FOREIGN KEY (assigned_to_user_id) REFERENCES users(id),

    UNIQUE(tenant_id, picking_id)
);
```

TABLE: picking_lines

```
CREATE TABLE picking_lines (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    picking_order_id UUID NOT NULL,
    order_line_id UUID NOT NULL,
    tenant_id UUID NOT NULL,

    sku_id UUID NOT NULL,
    location_id UUID NOT NULL,

    -- Quantidades
    quantity_required INT,
    quantity_picked INT DEFAULT 0,

    -- Sequência
    sequence_number INT,

    -- Status
    status ENUM ('PENDING', 'PICKED', 'VERIFIED', 'CANCELLED') DEFAULT 'PENDING',
```

```
    -- Auditoria
    picked_at TIMESTAMP,
    verified_at TIMESTAMP,
    picked_by UUID,
    verified_by UUID,

    FOREIGN KEY (picking_order_id) REFERENCES picking_orders(id) ON DELETE
CASCADE,
    FOREIGN KEY (order_line_id) REFERENCES order_lines(id),
    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (sku_id) REFERENCES skus(id),
    FOREIGN KEY (location_id) REFERENCES locations(id),
    FOREIGN KEY (picked_by) REFERENCES users(id),
    FOREIGN KEY (verified_by) REFERENCES users(id)
);
```

TABLE: packages

```
CREATE TABLE packages (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    order_id UUID NOT NULL,
    tenant_id UUID NOT NULL,

    -- Identificação
    package_number VARCHAR(50) NOT NULL,
    tracking_number VARCHAR(100),

    -- Dimensões e Peso
    weight_kg DECIMAL(12,3),
    width_cm INT,
    length_cm INT,
    height_cm INT,
    volume_m3 DECIMAL(12,6),

    -- Embalagem
    package_type VARCHAR(50),       -- 'Box', 'Envelope', 'Pallet', etc

    -- Status
    status ENUM ('PREPARED', 'LABELED', 'CONSOLIDATED', 'SHIPPED', 'DELIVERED',
'RETURNED') DEFAULT 'PREPARED',

    -- Transportação
    carrier_id UUID,
    shipping_method_id UUID,
    shipping_cost DECIMAL(12,2),

    -- Timestamps
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    packed_at TIMESTAMP,
    shipped_at TIMESTAMP,
```

```
    delivered_at TIMESTAMP,

    FOREIGN KEY (order_id) REFERENCES orders(id),
    FOREIGN KEY (tenant_id) REFERENCES tenants(id),

    UNIQUE(tenant_id, package_number)
);
```

TABLE: shipments

```
CREATE TABLE shipments (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,
    warehouse_id UUID NOT NULL,

    -- Identificação
    shipment_number VARCHAR(50) NOT NULL,
    manifest_number VARCHAR(50),

    -- Consolidação
    total_packages INT,
    total_weight DECIMAL(12,2),
    total_volume DECIMAL(12,6),
    total_orders INT,

    -- Transportação
    carrier_id UUID,
    shipping_method_id UUID,
    vehicle_id VARCHAR(50),

    -- Datas
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,
    prepared_at TIMESTAMP,
    manifested_at TIMESTAMP,
    dispatched_at TIMESTAMP,

    -- Status
    status ENUM ('PREPARATION', 'READY', 'MANIFESTED', 'DISPATCHED', 'IN_TRANSIT',
'DELIVERED', 'CANCELLED') DEFAULT 'PREPARATION',

    -- Rastreamento
    tms_reference VARCHAR(100),
    tracking_url TEXT,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    FOREIGN KEY (warehouse_id) REFERENCES warehouses(id),

    UNIQUE(tenant_id, shipment_number)
);
```

3.6 Referências Mestras

TABLE: suppliers

```
CREATE TABLE suppliers (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,

    name VARCHAR(255) NOT NULL,
    legal_name VARCHAR(500),
    cnpj VARCHAR(18),
    email VARCHAR(255),
    phone VARCHAR(20),

    address VARCHAR(500),
    city VARCHAR(100),
    state VARCHAR(2),

    status ENUM ('ACTIVE', 'INACTIVE', 'BLACKLISTED') DEFAULT 'ACTIVE',

    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    UNIQUE(tenant_id, cnpj)
);
```

TABLE: customers

```
CREATE TABLE customers (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,

    name VARCHAR(255) NOT NULL,
    type ENUM ('PJ', 'PF') DEFAULT 'PJ',
    document_number VARCHAR(18),

    email VARCHAR(255),
    phone VARCHAR(20),

    status ENUM ('ACTIVE', 'INACTIVE', 'BLOCKED') DEFAULT 'ACTIVE',

    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id),
    UNIQUE(tenant_id, document_number)
);
```

3.7 Auditoria e Compliance

TABLE: audit_log

```
CREATE TABLE audit_log (
    id UUID PRIMARY KEY DEFAULT gen_random_uuid(),
    tenant_id UUID NOT NULL,

    -- Entidade Afetada
    entity_type VARCHAR(100),       -- 'Order', 'PickingOrder', 'Inventory', etc
    entity_id UUID,

    -- Ação
    action ENUM ('CREATE', 'UPDATE', 'DELETE', 'VIEW', 'EXPORT') NOT NULL,
    action_description TEXT,

    -- Mudanças
    old_values JSONB,
    new_values JSONB,

    -- Usuário
    user_id UUID,
    user_role VARCHAR(100),

    -- Localização
    ip_address INET,
    user_agent TEXT,

    -- Timestamp
    created_at TIMESTAMP DEFAULT CURRENT_TIMESTAMP,

    FOREIGN KEY (tenant_id) REFERENCES tenants(id)
);

CREATE INDEX idx_audit_log_tenant ON audit_log(tenant_id);
CREATE INDEX idx_audit_log_entity ON audit_log(entity_type, entity_id);
CREATE INDEX idx_audit_log_date ON audit_log(created_at DESC);
CREATE INDEX idx_audit_log_user ON audit_log(user_id);
```

---

4. Constraints e Validações

4.1 Data Integrity

```
-- Verificar que quantity_reserved <= quantity_on_hand
ALTER TABLE inventory_master ADD CONSTRAINT
    check_reservation VALIDATE AS
    quantity_reserved <= quantity_on_hand;

-- Verificar que datas de entrega são futuras
```

```
ALTER TABLE orders ADD CONSTRAINT
    check_delivery_date AS
    promised_delivery_date >= order_date;

-- Verificar que peso e volume são positivos
ALTER TABLE skus ADD CONSTRAINT
    check_weight_volume AS
    weight_kg > 0 AND volume_m3 > 0;
```

---

5. Índices por Performance

```
-- Recebimento
CREATE INDEX idx_inbound_asn_status ON inbound_asn(status);
CREATE INDEX idx_inbound_asn_warehouse ON inbound_asn(warehouse_id);
CREATE INDEX idx_inbound_asn_supplier ON inbound_asn(supplier_id);

-- Picking
CREATE INDEX idx_picking_orders_status ON picking_orders(status);
CREATE INDEX idx_picking_orders_user ON picking_orders(assigned_to_user_id);
CREATE INDEX idx_picking_orders_wave ON picking_orders(wave_id);

-- Expedição
CREATE INDEX idx_shipments_status ON shipments(status);
CREATE INDEX idx_shipments_warehouse ON shipments(warehouse_id);
CREATE INDEX idx_shipments_carrier ON shipments(carrier_id);

-- Full-text search
CREATE INDEX idx_skus_search ON skus USING GIN(to_tsvector('portuguese', name || '
' || description));
CREATE INDEX idx_orders_search ON orders USING GIN(to_tsvector('portuguese',
order_number));
```

---

6. Estratégia de Particionamento

Para tabelas grandes (> 500 milhões de registros), implementar particionamento:

```
-- Particionamento por data (inventory_transactions)
CREATE TABLE inventory_transactions_2024_q1 PARTITION OF inventory_transactions
    FOR VALUES FROM ('2024-01-01') TO ('2024-04-01');

CREATE TABLE inventory_transactions_2024_q2 PARTITION OF inventory_transactions
    FOR VALUES FROM ('2024-04-01') TO ('2024-07-01');

-- Particionamento por tenant (para multi-tenancy mais severo)
-- (Alternativa: usar Row Level Security em vez de particionamento físico)
```

---

7. Backup e Disaster Recovery

7.1 Estratégia de Backup

- **Full Backup:** Diário (00:00 UTC)
- **Incremental:** A cada 6 horas
- **Retenção:** 30 dias em cold storage
- **Point-in-Time Recovery:** Até 7 dias

7.2 Replicação

- Read replicas em 3 data centers
- Sincronização em tempo real (RPO < 1 segundo)
- Failover automático com RTO < 5 minutos

Documento Versão: 1.0
Status: Design Proposto
Próximo Passo: Aprovação e implementação