📋 Handoff - Sistema de Gestão Comercial B2B

Documento de Entrega para a Equipe MARQUENTE

Versão: 1.0
Data: Dezembro 2025
Projeto: Sistema de Gestão Comercial para Representantes

📑 Índice

---

1. Visão Geral do Sistema

1.1 O que é?

O Sistema de Gestão Comercial B2B é uma plataforma web desenvolvida para **representantes comerciais** gerenciarem suas operações de vendas. Ele permite:

- Gestão completa de clientes

- Catálogo de produtos com preços e estoque
- Criação e acompanhamento de pedidos
- Campanhas promocionais
- Workshops e eventos
- Relatórios e análises
- **Carrinho Inteligente com IA** (sugestões automáticas de produtos)

1.2 Usuários do Sistema

Perfil	Descrição	Permissões
Admin	Administrador do sistema	Acesso total a todas as funcionalidades
Representante	Vendedor/Representante comercial	Acesso aos seus clientes, pedidos e campanhas
Vendedor	Usuário básico	Acesso limitado (visualização)

1.3 URLs de Acesso

- **Supabase Dashboard:**
  `https://supabase.com/dashboard/project/kmgrdxrgymskbmjavdpd`

---

2. Tecnologias Utilizadas

2.1 Frontend

Tecnologia	Versão	Propósito
React	18.x	Framework principal
Vite	5.x	Build tool e dev server
Tailwind CSS	3.x	Estilização
shadcn/ui	-	Componentes de UI

Tecnologia	Versão	Propósito
React Router	7.x	Navegação/Rotas
TanStack Query	5.x	Gerenciamento de estado e cache
Recharts	2.x	Gráficos e visualizações
Framer Motion	12.x	Animações

2.2 Backend (Supabase)

Serviço	Propósito
PostgreSQL	Banco de dados relacional
PostgREST	API REST automática
GoTrue	Autenticação de usuários
Edge Functions	Lógica de backend (Deno/TypeScript)
Storage	Armazenamento de arquivos
Realtime	Atualizações em tempo real

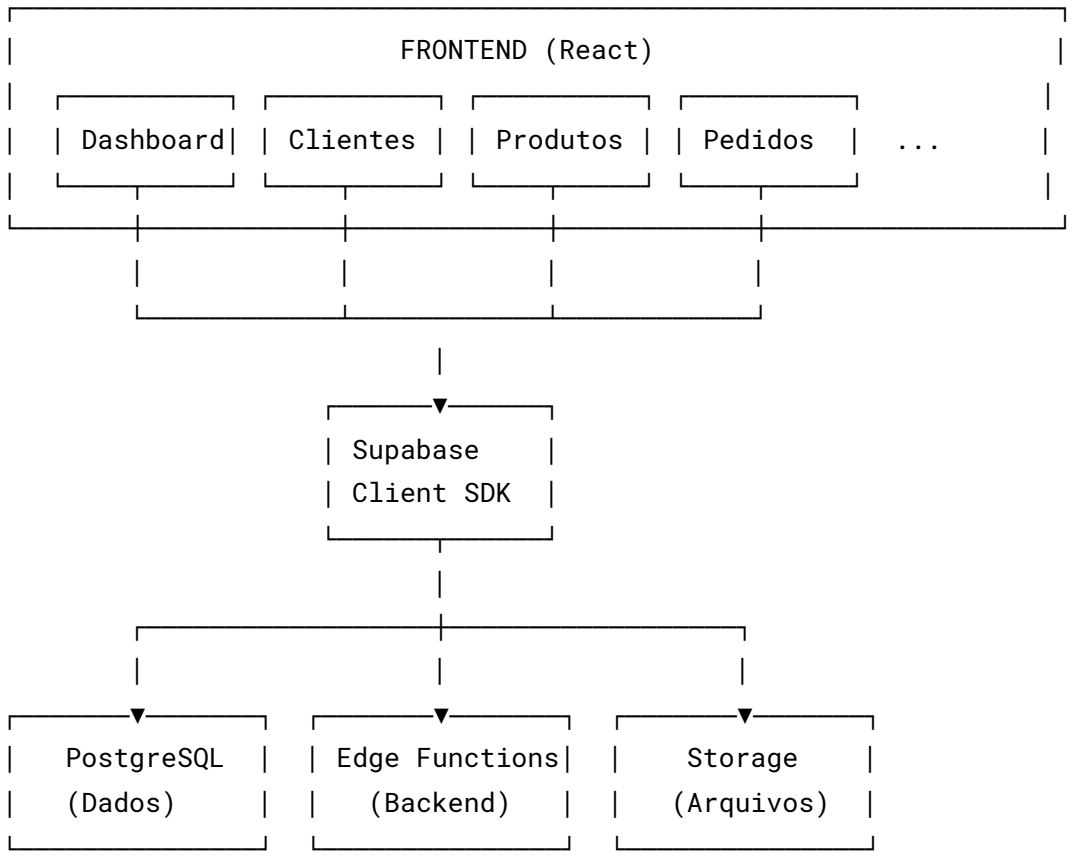
2.3 IA/Machine Learning

Serviço	Propósito
Google Gemini 2.5 Flash	Modelo principal para sugestões
Google Gemini 2.5 Pro	Modelo avançado para análises complexas

---

3. Arquitetura do Sistema

3.1 Diagrama de Arquitetura



3.2 Estrutura de Pastas do Projeto

```
projeto/
├── src/
│   ├── api/                    # Camada de abstração da API
│   │   └── supabase/
│   │       ├── entities.js     # Classes para cada entidade (Cliente, Produt
│   │       └── integrations.js # Funções de integração com Edge Functions
│   ├── components/             # Componentes React
│   │   ├── ui/                 # Componentes base (shadcn)
│   │   ├── dashboard/          # Componentes do dashboard
│   │   ├── clientes/           # Componentes de clientes
│   │   ├── produtos/           # Componentes de produtos
│   │   ├── carrinho/           # Componentes do carrinho inteligente
│   │   └── ...
│   ├── contexts/               # Contextos React (Auth, Theme)
│   ├── hooks/                  # Hooks customizados
│   ├── pages/                  # Páginas/Rotas
│   └── integrations/
│       └── supabase/
│           ├── client.ts       # Cliente Supabase configurado
│           └── types.ts        # Tipos gerados do banco
├── supabase/
│   ├── config.toml             # Configuração do Supabase
```

```
|   ├── functions/              # Edge Functions
|   |   ├── invoke-llm/         # Chamadas de IA
|   |   ├── create-intelligent-cart/  # Carrinho inteligente
|   |   ├── normalize-clients/  # Importação de clientes
|   |   ├── normalize-sales/    # Importação de vendas
|   |   └── ...
|   └── migrations/             # Migrações SQL (histórico)
└── public/                     # Arquivos estáticos
```

---

4. Funcionalidades Principais

4.1 Dashboard

Rota: `/`

O dashboard exibe:

- **Cards de estatísticas:** Total de clientes, vendas do mês, pedidos pendentes
- **Gráficos de vendas:** Evolução mensal, comparativo
- **Top clientes:** Clientes com maior volume de compras
- **Top produtos:** Produtos mais vendidos
- **Alertas inteligentes:** Clientes inativos, oportunidades
- **Mix de marcas:** Distribuição de vendas por marca

4.2 Gestão de Clientes

Rota: `/clientes`

Funcionalidades:

- Listagem paginada com busca e filtros
- Cadastro e edição de clientes
- Visualização de histórico de compras
- Filtros por região, perfil, status
- Importação via CSV
- Próxima compra prevista (IA)

Campos principais do cliente:

- Nome, CNPJ/CPF, Email, Telefone
- Endereço completo (cidade, estado, CEP)
- Representante responsável
- Limite de crédito
- Histórico de compras (datas, valores)
- Participação em workshops

4.3 Gestão de Produtos

Rota: `/produtos`

Funcionalidades:

- Catálogo completo de produtos
- Gestão de preços (preço, custo, revenda)
- Controle de estoque
- Categorização (marca, grupo, subgrupo, griffe)
- Importação via CSV
- Atualização em lote de preços

Campos principais do produto:

- Código, Nome, Descrição
- Marca, Grupo, Subgrupo, Griffe
- Preço, Custo, Revenda
- Estoque disponível
- Coleção, Temporada, Gênero
- Tamanhos e cores disponíveis

4.4 Gestão de Pedidos

Rota: Integrado em várias páginas

Funcionalidades:

- Criação de pedidos manuais
- Pedidos via carrinho inteligente (IA)
- Pedidos via e-catálogo
- Acompanhamento de status
- Histórico completo

Status de pedidos:

- `pendente` - Aguardando processamento
- `aguardando_ecatalogo` - Enviado para aprovação do cliente
- `confirmado` - Aprovado pelo cliente
- `aguardando_alteracao` - Cliente solicitou mudanças
- `faturado` - Pedido faturado
- `entregue` - Pedido entregue

4.5 Carrinho Inteligente (IA)

Rota: `/carrinho-inteligente`

O **Carrinho Inteligente** é uma das funcionalidades mais poderosas do sistema. Ele usa IA para sugerir produtos automaticamente baseado em:

- Histórico de compras do cliente
- Produtos mais vendidos
- Sazonalidade
- Estoque disponível
- Perfil do cliente

Como funciona:

1. Representante seleciona um cliente
2. Sistema analisa histórico e padrões
3. IA sugere produtos com quantidades e preços
4. Representante ajusta sugestões
5. Gera pedido automaticamente

4.6 Campanhas

Rota: `/campanhas`

Gerenciamento de campanhas promocionais:

- Criação de campanhas com período
- Definição de público-alvo
- Descontos e promoções
- Acompanhamento de ROI
- Mensagens personalizadas (IA)

Tipos de campanha:

- Desconto percentual
- Lançamento de coleção
- Reativação de clientes
- Workshop/Evento

4.7 Workshops

Rota: `/workshop`

Gestão de eventos e workshops:

- Criação de eventos
- Gestão de participantes
- Convites automáticos
- Sugestão de clientes (IA)
- Controle de capacidade

4.8 Relatórios

Rota: `/relatorios`

Relatórios analíticos:

- Vendas por período
- Vendas por representante
- Vendas por canal (manual, e-catálogo, carrinho IA)
- KPIs de performance
- Exportação de dados

4.9 Importação de Dados

Rota: `/importacao`

Central de importação de dados (ver seção 8 para detalhes completos):

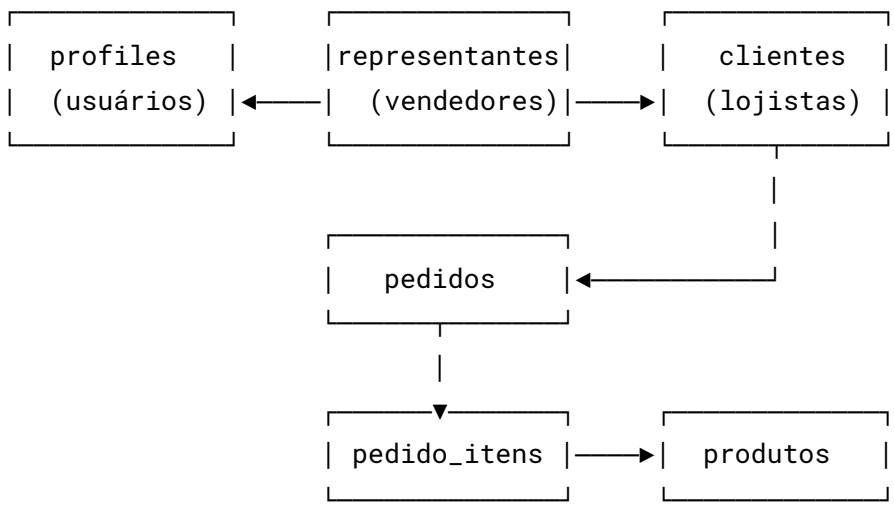
- Upload de arquivos CSV
- Sincronização com SQL Server
- Histórico de importações

- Validação de dados

---

5. Estrutura do Banco de Dados

5.1 Diagrama ER Simplificado



5.2 Tabelas Principais

profiles (Perfis de usuário)

```
- id (uuid, PK)           -- ID do usuário (auth.users)
- nome (text)             -- Nome completo
- email (text)            -- Email
- role (user_role)        -- admin, representante, vendedor
- ativo (boolean)         -- Status ativo/inativo
```

representantes (Vendedores/Representantes)

```
- id (uuid, PK)
- user_id (uuid, FK)      -- Referência ao profile
- nome (text)
- email (text)
- telefone (text)
- cod_externo_vendedor    -- Código no sistema externo (SQL Server)
- regiao_id (uuid, FK)    -- Região de atuação
```

- meta_mensal (numeric)
- comissao_percentual (numeric)

clientes (Lojistas/Clientes)

- id (uuid, PK)
- representante_id (uuid, FK) -- Vendedor responsável
- cod_externo_cliente   -- Código no sistema externo
- nome (text)
- cnpj, cpf (text)
- email, telefone, whatsapp
- endereco, cidade, estado, cep
- limite_credito (numeric)
- total_compras (numeric)
- data_ultima_compra (date)
- perfil (text)   -- VIP, Regular, Novo, etc.
- participou_workshop (boolean)

produtos

- id (uuid, PK)
- codigo (text)   -- Código único
- cod_produto   -- Código no sistema externo
- nome (text)
- marca_id (uuid, FK)
- preco, custo, revenda (numeric)
- estoque_disponivel (integer)
- grupo, sub_grupo, griffe
- tamanhos, cores (array)
- colecao, temporada, genero

pedidos

- id (uuid, PK)
- numero_pedido (text)   -- Número único
- cliente_id (uuid, FK)
- representante_id (uuid, FK)
- data_pedido (date)
- valor_total, valor_final (numeric)
- desconto (numeric)
- status (text)
- origem (text)   -- manual, ecatalogo, campanha, carrinho_ia

pedido_itens

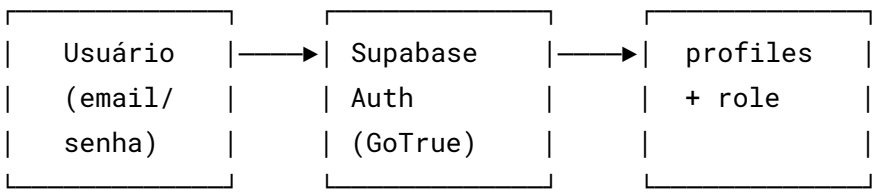
```
- id (uuid, PK)
- pedido_id (uuid, FK)
- produto_id (uuid, FK)
- quantidade (integer)
- preco_unitario (numeric)
- valor_total (numeric)
- tamanho, cor (text)
```

5.3 Tabelas Auxiliares

Tabela	Descrição
`marcas`	Marcas de produtos
`grupos_produto`	Grupos de produtos
`subgrupos_produto`	Subgrupos de produtos
`griffes`	Griffes/designers
`tipos_produto`	Tipos de produtos
`categorias_produto`	Categorias
`regioes`	Regiões geográficas
`lojas`	Lojas/filiais
`estoques`	Estoque por loja/produto
`campanhas`	Campanhas promocionais
`workshops`	Eventos/workshops
`workshop_participantes`	Participantes de workshops
`metas_mensais`	Metas por representante
`import_jobs`	Histórico de importações
`sugestoes_ia`	Cache de sugestões de IA
`carrinho_ia_metricas`	Métricas do carrinho inteligente

6. Autenticação e Autorização

6.1 Fluxo de Login



1. Usuário acessa `/auth` e faz login
2. Supabase Auth valida credenciais
3. Sistema busca perfil e role do usuário
4. Interface carrega baseada nas permissões

6.2 Níveis de Acesso (RLS)

O Supabase usa **Row Level Security (RLS)** para controlar acesso aos dados:

```
-- Exemplo: Representantes só veem seus próprios clientes
CREATE POLICY "Representantes podem ver seus próprios clientes"
ON clientes FOR SELECT
USING (representante_id = get_user_representante_id() OR is_admin());
```

Funções de autorização:

- `is_admin()` - Verifica se usuário é admin
- `is_representante()` - Verifica se é representante
- `get_user_representante_id()` - Retorna ID do representante logado

6.3 Criando Novos Usuários

1. Criar usuário no Supabase Auth (Dashboard > Authentication > Users)
2. O trigger `handle_new_user` cria automaticamente o profile
3. Atualizar role se necessário (admin/representante)
4. Vincular ao representante se aplicável

7. Funcionalidades de IA

7.1 Carrinho Inteligente

Edge Function: `create-intelligent-cart`

Analisa:

- Últimas compras do cliente
- Produtos mais vendidos no geral
- Sazonalidade (época do ano)
- Estoque disponível
- Perfil de consumo

Retorna sugestões de produtos com:

- Código e nome do produto
- Quantidade sugerida
- Preço unitário
- Justificativa da sugestão

7.2 Sugestões de Produtos

Edge Function: `generate-product-suggestions`

Gera sugestões complementares baseadas em:

- Produtos já no carrinho
- Histórico de vendas casadas
- Margem de lucro
- Disponibilidade

7.3 Mensagens de Campanha

Edge Function: `generate-campaign-message`

Gera textos personalizados para campanhas usando IA:

- Tom de voz adequado
- Personalização por cliente

- CTAs efetivos

7.4 Sugestões de Clientes para Workshop

Edge Function: `workshop-client-suggestions`

Sugere clientes ideais para workshops baseado em:

- Perfil de compra
- Localização
- Histórico de participação
- Potencial de conversão

---

8. Alimentação do Sistema (Importação de Dados)

🔴 SEÇÃO CRÍTICA - COMO ALIMENTAR O BANCO DE DADOS

Esta seção explica detalhadamente como os dados entram no sistema.

8.1 Visão Geral

O sistema pode ser alimentado de **duas formas**:

Método	Descrição	Quando Usar
Upload CSV	Arquivos CSV enviados manualmente	Importações pontuais, testes
Sincronização SQL Server	Script automático conecta ao banco	Atualizações frequentes, produção

---

8.2 Método 1: Upload de Arquivos CSV

Rota: `/importacao`

Ordem Recomendada de Importação

⚠️ **IMPORTANTE:** Os arquivos devem ser importados na ordem correta para manter integridade referencial!

```
1. Lojas.csv          → Tabela: lojas
2. Vendedores.csv     → Tabela: representantes
3. Clientes.csv       → Tabela: clientes
4. Produtos.csv       → Tabela: produtos
5. Estoques.csv       → Tabela: estoques
6. Vendas.csv         → Tabela: pedidos
7. Vendas_Produtos.csv → Tabela: pedido_itens
```

Estrutura dos Arquivos CSV

1. Lojas.csv

```
COD_LOJA;NOME;APELIDO
001;Loja Centro;Centro
002;Loja Shopping;Shopping
```

2. Vendedores.csv

```
CPF;NOME;EMAIL;COD_LOJA
12345678901;João Silva;joao@email.com;001
```

3. Clientes.csv

```
COD_CLIENTE;NOME;CNPJ;CPF;EMAIL;TELEFONE;CIDADE;ESTADO;CEP;COD_VENDEDOR;DATA_
1001;Loja ABC;12345678000190;;loja@email.com;11999999999;São Paulo;SP;0131010
```

4. Produtos.csv

```
COD_PRODUTO;NOME;MARCA;GRUPO;SUBGRUPO;PRECO;CUSTO;ESTOQUE;COLECAO;GENERO
P001;Camiseta Básica;Marca X;Vestuário;Camisetas;89.90;35.00;150;2024;M
```

5. Estoques.csv

```
COD_LOJA;COD_PRODUTO;COR;QUANTIDADE;DATA_ATUALIZACAO
001;P001;BRANCO;50;2024-12-01
001;P001;PRETO;30;2024-12-01
```

6. Vendas.csv

```
NUMERO_PEDIDO;COD_CLIENTE;COD_VENDEDOR;DATA_PEDIDO;VALOR_TOTAL;VALOR_FINAL;ST
V2024001;1001;001;2024-11-20;1500.00;1425.00;entregue;manual
```

7. Vendas_Produtos.csv

```
NUMERO_PEDIDO;COD_PRODUTO;QUANTIDADE;PRECO_UNITARIO;VALOR_TOTAL;COR;TAMANHO
V2024001;P001;10;89.90;899.00;BRANCO;M
V2024001;P002;5;105.20;526.00;AZUL;G
```

Como Fazer Upload

1. Acesse `/importacao` no sistema
2. Clique no card do tipo de arquivo
3. Arraste o arquivo ou clique para selecionar
4. Clique em **"Importar Todos os Arquivos"**
5. Acompanhe o progresso
6. Verifique o relatório de erros/avisos

8.3 Método 2: Sincronização Automática com SQL Server

Script: `sync_database.py` (na raiz do projeto)

Este método conecta diretamente ao SQL Server da MARQUENTE e sincroniza os dados automaticamente.

Pré-requisitos

1. **Python 3.8+** instalado
2. **ODBC Driver 17 for SQL Server** instalado
3. Acesso à rede onde está o SQL Server (10.20.0.9)

Instalação

```bash
# 1. Instalar dependências Python
pip install pyodbc requests python-dotenv

# 2. Copiar arquivo de configuração
cp .env.example .env
```

```
# 3. Editar .env com as credenciais corretas
```

Configuração (.env)

```
# SQL Server
SQL_HOST=10.20.0.9
SQL_DATABASE=DW
SQL_USER=terceiro_kpc
SQL_PASSWORD=sua_senha_aqui

# Supabase
SUPABASE_URL=https://kmgrdxrgymskbmjavdpd.supabase.co
SYNC_TOKEN=sk_sync_7f3a9d2e4b1c6f8a5d9e2b4c7f1a6d3e9b2c5f8a1d4e7b3c6f9a2d5e8b
```

Executando a Sincronização

```
# Execução manual
python sync_database.py
```

Saída esperada:

```
🔄 Iniciando sincronização com SQL Server...

📊 Processando Lojas...
  ✓ Lidos 5 registros
  ✓ Criados: 0
  ✓ Atualizados: 5

📊 Processando Vendedores...
  ✓ Lidos 12 registros
  ✓ Criados: 2
  ✓ Atualizados: 10

...

✅ Sincronização concluída!
```

Agendamento Automático

Windows (Agendador de Tarefas):

1. Abrir Agendador de Tarefas

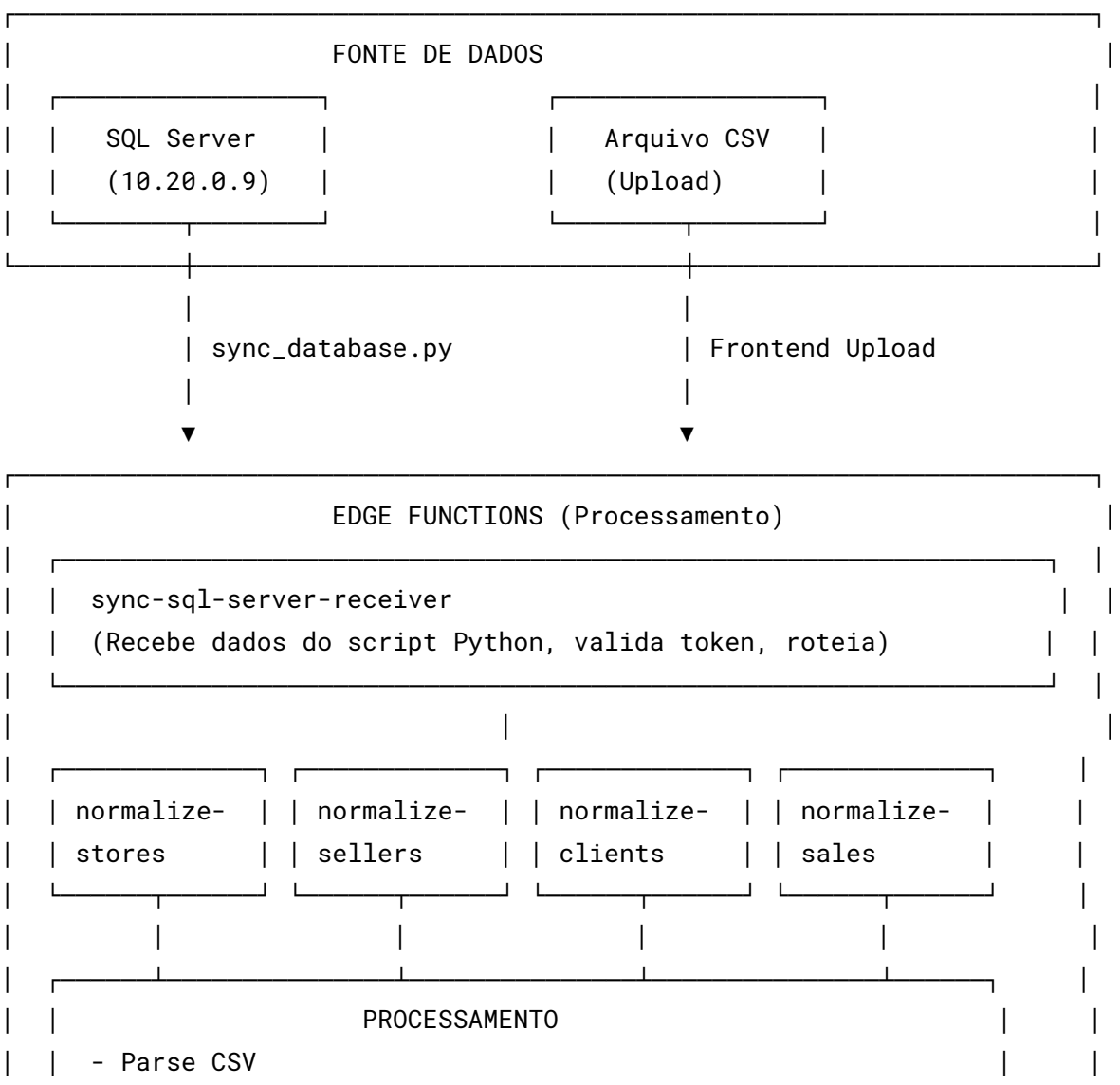
2. Criar Tarefa Básica
3. Configurar gatilho (ex: diariamente às 6h)
4. Ação: Iniciar programa
    - Programa: `python`
    - Argumentos: `C:\caminho\sync_database.py`

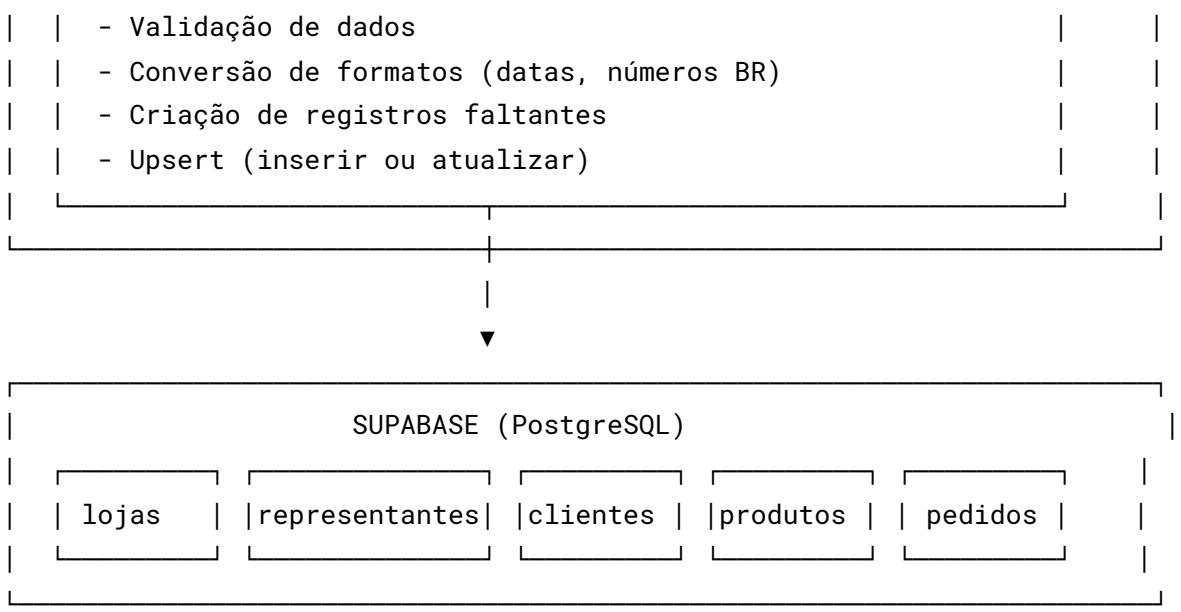
Linux/Mac (Cron):

```
# Editar crontab
crontab -e

# Adicionar linha (todo dia às 6h)
0 6 * * * /usr/bin/python3 /caminho/sync_database.py >> /var/log/sync.log 2>&
```

8.4 Fluxo de Processamento de Dados





8.5 Edge Functions de Importação

Cada tipo de arquivo tem uma Edge Function dedicada:

Arquivo	Edge Function	Tabela Destino
Lojas.csv	`normalize-stores`	`lojas`
Vendedores.csv	`normalize-sellers`	`representantes`
Clientes.csv	`normalize-clients`	`clientes`
Produtos.csv	`process-products-csv`	`produtos` + auxiliares
Estoques.csv	`normalize-inventory`	`estoques`
Vendas.csv	`normalize-sales`	`pedidos`
Vendas_Produtos.csv	`normalize-product-sales`	`pedido_itens`

Funcionalidades das Edge Functions

Todas as funções:

- Fazem parse do CSV
- Convertem formatos brasileiros (datas DD/MM/YYYY, números 1.234,56)
- Tratam valores nulos e vazios

- Fazem upsert (insert ou update se existir)
- Registram estatísticas na tabela `import_jobs`
- Retornam erros e avisos

Recursos específicos:

- `normalize-clients` : Cria representantes placeholder se não existirem
- `normalize-sales` : Cria clientes placeholder se não existirem
- `normalize-product-sales` : Atualiza totais dos pedidos após inserir itens
- `process-products-csv` : Cria marcas, grupos, griffes automaticamente

8.6 Monitoramento de Importações

Tabela import_jobs

Todas as importações são registradas:

```
SELECT
  tipo_arquivo,
  nome_arquivo,
  status,
  total_linhas,
  linhas_criadas,
  linhas_atualizadas,
  linhas_erro,
  tempo_processamento,
  created_at
FROM import_jobs
ORDER BY created_at DESC;
```

Logs de Edge Functions

Para debug, acesse os logs no Supabase Dashboard:

- `https://supabase.com/dashboard/project/kmgrdxrgymskbmjavdpd/functions/[nome-funcao]/logs`

8.7 Troubleshooting de Importação

Problemas Comuns

Problema	Causa	Solução
“Nenhum registro importado”	Arquivo vazio ou formato errado	Verificar encoding (UTF-8) e separador (;)
“Erro de foreign key”	Ordem de importação errada	Importar na ordem correta
“Duplicatas”	Código já existe	Normal - sistema faz update
“Valores zerados”	Formato de número errado	Usar formato brasileiro (1.234,56)
“Datas inválidas”	Formato de data errado	Usar DD/MM/YYYY
“Timeout”	Arquivo muito grande	Dividir em chunks menores

Validação de Dados

Antes de importar, verifique:

```
# Encoding do arquivo
file -i arquivo.csv

# Primeiras linhas
head -5 arquivo.csv

# Contagem de colunas
head -1 arquivo.csv | tr ';' '\n' | wc -l
```

---

9. Edge Functions (Backend)

9.1 Lista de Edge Functions

Função	Propósito	Autenticação
`invoke-llm`	Chamadas de IA (Gemini)	JWT

Função	Propósito	Autenticação
`create-intelligent-cart`	Carrinho inteligente	JWT
`generate-product-suggestions`	Sugestões de produtos	JWT
`generate-campaign-message`	Mensagens de campanha	JWT
`workshop-client-suggestions`	Sugestões para workshop	JWT
`analyze-customer-behavior`	Análise de comportamento	JWT
`process-products-csv`	Importação de produtos	JWT
`normalize-clients`	Importação de clientes	JWT
`normalize-sales`	Importação de vendas	JWT
`normalize-product-sales`	Importação de itens	JWT
`normalize-inventory`	Importação de estoque	JWT
`normalize-stores`	Importação de lojas	JWT
`normalize-sellers`	Importação de vendedores	JWT
`sync-sql-server-receiver`	Recebe dados do script Python	Token customizado

9.2 Secrets Configurados

Os seguintes secrets estão configurados no Supabase:

Secret	Propósito
`SUPABASE_URL`	URL do projeto Supabase
`SUPABASE_ANON_KEY`	Chave pública
`SUPABASE_SERVICE_ROLE_KEY`	Chave de admin (backend)
`SQL_SERVER_SYNC_TOKEN`	Token de autenticação para sync

9.3 Deploy de Edge Functions

As Edge Functions são deployadas automaticamente quando o código é commitado. Para deploy manual:

```
# Via Supabase CLI
supabase functions deploy nome-da-funcao
```

---

10. Configuração e Deploy

10.1 Variáveis de Ambiente

Frontend (.env):

```
VITE_SUPABASE_PROJECT_ID=kmgrdxrgymskbmjavdpd
VITE_SUPABASE_PUBLISHABLE_KEY=eyJhbGci...
VITE_SUPABASE_URL=https://kmgrdxrgymskbmjavdpd.supabase.co
```

10.2 Build e Deploy

O projeto usa **Github/Vercel** para deploy automático. Cada commit na branch principal gera um deploy.

Build manual:

```
npm run build
```

10.3 Acessos Necessários

Serviço	URL	Credenciais
Supabase	https://supabase.com/dashboard/project/kmgrdxrgymskbmjavdpd	Conta com acesso ao projeto

Serviço	URL	Credenciais
GitHub	Arquivo zipado já encaminhado	Repositório do projeto

---

11. Manutenção e Suporte

11.1 Monitoramento

Logs do Supabase:

- Edge Functions: Dashboard > Functions > Logs
- Database: Dashboard > Logs > Postgres

Métricas:

- Uso de banco: Dashboard > Database > Metrics
- Edge Functions: Dashboard > Functions > Metrics

11.2 Backup

O Supabase faz backups automáticos:

- Plano gratuito: 7 dias de retenção
- Plano Pro: 30 dias de retenção

11.3 Contatos

Para suporte técnico:

- Documentação Supabase: https://supabase.com/docs

📋 Checklist de Entrega

- ☑️ Documentação completa do sistema
- ☑️ Acesso ao Supabase Dashboard configurado
- ☑️ Edge Functions deployadas e funcionando
- ☑️ Script de sincronização testado
- ☑️ Usuários de teste criados
- ☑️ Importação inicial de dados realizada

📝 Notas Finais

Este sistema foi desenvolvido para otimizar o trabalho dos representantes comerciais da MARQUENTE. A combinação de funcionalidades tradicionais com **IA** (carrinho inteligente, sugestões, análises) oferece uma vantagem competitiva significativa.

Para dúvidas ou suporte adicional, consulte os logs do sistema e a documentação do Supabase.

Documento gerado em: Dezembro 2025
Versão do sistema: 1.0