Table of Contents

# 📊 CÓRTEX BI - Configuração de Fontes de Dados para Detecção de Anomalias

**Versão:** 2.0  
**Data:** Outubro 2025  
**Desenvolvido em parceria com:** Manus AI

## 🎯 Visão Geral

A configuração das **fontes de dados** é o coração do sistema de detecção de anomalias. É aqui que você define:

* **ONDE** o sistema deve buscar os dados (SQL Server, arquivos CSV, APIs, etc.)
* **O QUE** monitorar (vendas, custos, métricas de performance, etc.)
* **QUANDO** verificar (intervalos de tempo)
* **COMO** interpretar os dados (thresholds, severidade, etc.)

## 📁 Estrutura de Configuração

### Localização dos Arquivos

cortex-bi/  
├── config/  
│ └── anomaly\_rules/  
│ ├── metrics\_config.json # ⭐ ARQUIVO PRINCIPAL  
│ ├── data\_sources.json # Configuração de conexões  
│ ├── false\_positive\_rules.json # Regras de falsos positivos  
│ └── alert\_templates.json # Templates de alertas  
│  
├── data/  
│ ├── raw/ # Dados brutos (CSV, Excel, etc.)  
│ ├── processed/ # Dados processados  
│ └── anomalies/ # Histórico de anomalias detectadas  
│  
└── .env # Credenciais e configurações globais

## ⚙️ Arquivo Principal: metrics\_config.json

Este é o arquivo mais importante. Ele define todas as métricas que serão monitoradas.

### Template Completo

Crie o arquivo config/anomaly\_rules/metrics\_config.json:

{  
 "version": "2.0",  
 "description": "Configuração de métricas para detecção de anomalias",  
   
 "metrics": [  
 {  
 "id": "vendas\_totais\_diarias",  
 "name": "Vendas Totais Diárias",  
 "description": "Soma de todas as vendas do dia",  
 "enabled": true,  
   
 "data\_source": {  
 "type": "sql\_server",  
 "connection\_id": "sql\_principal",  
 "query": "SELECT SUM(valor\_venda) as valor, COUNT(\*) as quantidade FROM vendas WHERE CAST(data\_venda AS DATE) = CAST(GETDATE() AS DATE)",  
 "value\_column": "valor",  
 "timestamp\_column": null  
 },  
   
 "historical\_data": {  
 "query": "SELECT CAST(data\_venda AS DATE) as data, SUM(valor\_venda) as valor FROM vendas WHERE data\_venda >= DATEADD(day, -30, GETDATE()) GROUP BY CAST(data\_venda AS DATE) ORDER BY data",  
 "min\_samples": 30,  
 "max\_samples": 90  
 },  
   
 "anomaly\_detection": {  
 "method": "isolation\_forest",  
 "contamination": 0.1,  
 "threshold": -0.5,  
 "features": ["valor", "dia\_semana", "hora"]  
 },  
   
 "alert\_config": {  
 "severity": "high",  
 "threshold\_type": "percentage",  
 "threshold\_value": 15,  
 "cooldown\_minutes": 60,  
 "channels": ["email", "teams"],  
 "recipients": ["gerente.vendas@empresa.com", "diretor@empresa.com"]  
 },  
   
 "monitoring": {  
 "check\_interval": 3600,  
 "active\_hours": "00:00-23:59",  
 "active\_days": [1, 2, 3, 4, 5, 6, 7]  
 }  
 },  
   
 {  
 "id": "custos\_operacionais",  
 "name": "Custos Operacionais",  
 "description": "Custos operacionais diários",  
 "enabled": true,  
   
 "data\_source": {  
 "type": "csv",  
 "file\_path": "data/raw/custos\_diarios.csv",  
 "value\_column": "custo\_total",  
 "timestamp\_column": "data",  
 "delimiter": ",",  
 "encoding": "utf-8"  
 },  
   
 "historical\_data": {  
 "source": "same\_file",  
 "min\_samples": 30,  
 "max\_samples": 90  
 },  
   
 "anomaly\_detection": {  
 "method": "isolation\_forest",  
 "contamination": 0.1,  
 "threshold": -0.5,  
 "features": ["custo\_total", "dia\_semana"]  
 },  
   
 "alert\_config": {  
 "severity": "medium",  
 "threshold\_type": "absolute",  
 "threshold\_value": 50000,  
 "cooldown\_minutes": 120,  
 "channels": ["email"],  
 "recipients": ["controller@empresa.com"]  
 },  
   
 "monitoring": {  
 "check\_interval": 7200,  
 "active\_hours": "08:00-18:00",  
 "active\_days": [1, 2, 3, 4, 5]  
 }  
 },  
   
 {  
 "id": "taxa\_conversao\_site",  
 "name": "Taxa de Conversão do Site",  
 "description": "Percentual de visitantes que convertem",  
 "enabled": true,  
   
 "data\_source": {  
 "type": "api",  
 "url": "https://api.analytics.empresa.com/conversao/hoje",  
 "method": "GET",  
 "headers": {  
 "Authorization": "Bearer ${API\_ANALYTICS\_TOKEN}",  
 "Content-Type": "application/json"  
 },  
 "response\_path": "data.taxa\_conversao",  
 "timeout": 30  
 },  
   
 "historical\_data": {  
 "api\_url": "https://api.analytics.empresa.com/conversao/historico",  
 "api\_params": {  
 "dias": 30  
 },  
 "response\_path": "data.historico",  
 "min\_samples": 30  
 },  
   
 "anomaly\_detection": {  
 "method": "isolation\_forest",  
 "contamination": 0.1,  
 "threshold": -0.5,  
 "features": ["taxa\_conversao", "hora", "dia\_semana"]  
 },  
   
 "alert\_config": {  
 "severity": "high",  
 "threshold\_type": "percentage",  
 "threshold\_value": 10,  
 "cooldown\_minutes": 30,  
 "channels": ["email", "teams", "webhook"],  
 "recipients": ["marketing@empresa.com"]  
 },  
   
 "monitoring": {  
 "check\_interval": 1800,  
 "active\_hours": "00:00-23:59",  
 "active\_days": [1, 2, 3, 4, 5, 6, 7]  
 }  
 },  
   
 {  
 "id": "tempo\_resposta\_api",  
 "name": "Tempo de Resposta da API",  
 "description": "Tempo médio de resposta das requisições",  
 "enabled": true,  
   
 "data\_source": {  
 "type": "internal",  
 "metric\_key": "avg\_response\_time",  
 "aggregation": "avg",  
 "window\_minutes": 5  
 },  
   
 "historical\_data": {  
 "source": "internal\_logs",  
 "log\_file": "logs/cortexbi.log",  
 "pattern": "execution\_time: (\\d+)",  
 "min\_samples": 100  
 },  
   
 "anomaly\_detection": {  
 "method": "isolation\_forest",  
 "contamination": 0.05,  
 "threshold": -0.3,  
 "features": ["response\_time", "hora"]  
 },  
   
 "alert\_config": {  
 "severity": "critical",  
 "threshold\_type": "absolute",  
 "threshold\_value": 5000,  
 "cooldown\_minutes": 15,  
 "channels": ["teams", "webhook"],  
 "recipients": ["ti@empresa.com", "devops@empresa.com"]  
 },  
   
 "monitoring": {  
 "check\_interval": 60,  
 "active\_hours": "00:00-23:59",  
 "active\_days": [1, 2, 3, 4, 5, 6, 7]  
 }  
 },  
   
 {  
 "id": "estoque\_produto\_critico",  
 "name": "Estoque de Produto Crítico",  
 "description": "Nível de estoque do produto mais vendido",  
 "enabled": true,  
   
 "data\_source": {  
 "type": "excel",  
 "file\_path": "data/raw/estoque.xlsx",  
 "sheet\_name": "Estoque\_Atual",  
 "value\_column": "quantidade",  
 "filter": {  
 "column": "produto\_id",  
 "value": "PROD-001"  
 }  
 },  
   
 "historical\_data": {  
 "file\_path": "data/raw/historico\_estoque.xlsx",  
 "sheet\_name": "Historico",  
 "min\_samples": 30  
 },  
   
 "anomaly\_detection": {  
 "method": "isolation\_forest",  
 "contamination": 0.1,  
 "threshold": -0.5,  
 "features": ["quantidade", "dia\_semana"]  
 },  
   
 "alert\_config": {  
 "severity": "high",  
 "threshold\_type": "absolute",  
 "threshold\_value": 100,  
 "direction": "below",  
 "cooldown\_minutes": 240,  
 "channels": ["email"],  
 "recipients": ["logistica@empresa.com"]  
 },  
   
 "monitoring": {  
 "check\_interval": 14400,  
 "active\_hours": "08:00-18:00",  
 "active\_days": [1, 2, 3, 4, 5]  
 }  
 },  
   
 {  
 "id": "nps\_diario",  
 "name": "NPS Diário",  
 "description": "Net Promoter Score calculado diariamente",  
 "enabled": true,  
   
 "data\_source": {  
 "type": "power\_bi",  
 "workspace\_id": "seu-workspace-id",  
 "dataset\_id": "seu-dataset-id",  
 "table": "NPS\_Diario",  
 "measure": "NPS\_Score",  
 "filter": "Data = TODAY()"  
 },  
   
 "historical\_data": {  
 "source": "power\_bi",  
 "filter": "Data >= TODAY() - 30",  
 "min\_samples": 30  
 },  
   
 "anomaly\_detection": {  
 "method": "isolation\_forest",  
 "contamination": 0.1,  
 "threshold": -0.5,  
 "features": ["nps\_score", "dia\_semana"]  
 },  
   
 "alert\_config": {  
 "severity": "medium",  
 "threshold\_type": "percentage",  
 "threshold\_value": 20,  
 "cooldown\_minutes": 1440,  
 "channels": ["email", "teams"],  
 "recipients": ["atendimento@empresa.com", "qualidade@empresa.com"]  
 },  
   
 "monitoring": {  
 "check\_interval": 86400,  
 "active\_hours": "09:00-09:30",  
 "active\_days": [1, 2, 3, 4, 5]  
 }  
 }  
 ],  
   
 "global\_settings": {  
 "timezone": "America/Sao\_Paulo",  
 "min\_historical\_days": 30,  
 "max\_historical\_days": 90,  
 "confidence\_threshold": 0.8,  
 "default\_cooldown\_minutes": 60,  
 "retry\_on\_error": true,  
 "max\_retries": 3,  
 "retry\_delay\_seconds": 60  
 }  
}

## 🔌 Tipos de Fontes de Dados Suportadas

### 1. SQL Server

**Configuração básica:**

{  
 "data\_source": {  
 "type": "sql\_server",  
 "connection\_id": "sql\_principal",  
 "query": "SELECT SUM(valor) as total FROM vendas WHERE data = CAST(GETDATE() AS DATE)",  
 "value\_column": "total"  
 }  
}

**Configuração da conexão** (em config/anomaly\_rules/data\_sources.json):

{  
 "connections": {  
 "sql\_principal": {  
 "type": "sql\_server",  
 "host": "servidor.database.windows.net",  
 "port": 1433,  
 "database": "vendas\_db",  
 "username": "${SQL\_SERVER\_USERNAME}",  
 "password": "${SQL\_SERVER\_PASSWORD}",  
 "driver": "ODBC Driver 17 for SQL Server",  
 "connection\_timeout": 30,  
 "query\_timeout": 60  
 }  
 }  
}

**Variáveis no .env:**

SQL\_SERVER\_USERNAME=seu\_usuario  
SQL\_SERVER\_PASSWORD=sua\_senha

### 2. Arquivo CSV

**Configuração:**

{  
 "data\_source": {  
 "type": "csv",  
 "file\_path": "data/raw/vendas\_diarias.csv",  
 "value\_column": "valor\_total",  
 "timestamp\_column": "data",  
 "delimiter": ",",  
 "encoding": "utf-8",  
 "skip\_rows": 0,  
 "date\_format": "%Y-%m-%d"  
 }  
}

**Exemplo de arquivo CSV** (data/raw/vendas\_diarias.csv):

data,valor\_total,quantidade,regiao  
2025-10-01,125000.50,450,Sul  
2025-10-02,132000.75,480,Sul  
2025-10-03,118000.00,420,Sul  
2025-10-04,145000.25,510,Sul

### 3. Arquivo Excel

**Configuração:**

{  
 "data\_source": {  
 "type": "excel",  
 "file\_path": "data/raw/relatorio\_mensal.xlsx",  
 "sheet\_name": "Vendas",  
 "value\_column": "Total",  
 "timestamp\_column": "Data",  
 "header\_row": 0,  
 "filter": {  
 "column": "Regiao",  
 "value": "Sul"  
 }  
 }  
}

### 4. API REST

**Configuração:**

{  
 "data\_source": {  
 "type": "api",  
 "url": "https://api.empresa.com/metrics/vendas",  
 "method": "GET",  
 "headers": {  
 "Authorization": "Bearer ${API\_TOKEN}",  
 "Content-Type": "application/json"  
 },  
 "params": {  
 "data": "hoje",  
 "formato": "json"  
 },  
 "response\_path": "data.vendas.total",  
 "timeout": 30  
 }  
}

**Variáveis no .env:**

API\_TOKEN=seu-token-de-api

**Exemplo de resposta da API:**

{  
 "status": "success",  
 "data": {  
 "vendas": {  
 "total": 125000.50,  
 "quantidade": 450,  
 "data": "2025-10-14"  
 }  
 }  
}

### 5. Power BI

**Configuração:**

{  
 "data\_source": {  
 "type": "power\_bi",  
 "workspace\_id": "12345678-1234-1234-1234-123456789abc",  
 "dataset\_id": "87654321-4321-4321-4321-cba987654321",  
 "table": "Vendas",  
 "measure": "Total\_Vendas",  
 "filter": "Data = TODAY()"  
 }  
}

**Configuração de autenticação** (em data\_sources.json):

{  
 "power\_bi": {  
 "tenant\_id": "${POWERBI\_TENANT\_ID}",  
 "client\_id": "${POWERBI\_CLIENT\_ID}",  
 "client\_secret": "${POWERBI\_CLIENT\_SECRET}"  
 }  
}

**Variáveis no .env:**

POWERBI\_TENANT\_ID=seu-tenant-id  
POWERBI\_CLIENT\_ID=seu-client-id  
POWERBI\_CLIENT\_SECRET=seu-client-secret

### 6. Métricas Internas do Sistema

**Configuração:**

{  
 "data\_source": {  
 "type": "internal",  
 "metric\_key": "avg\_response\_time",  
 "aggregation": "avg",  
 "window\_minutes": 5  
 }  
}

**Métricas internas disponíveis:**

* avg\_response\_time - Tempo médio de resposta
* request\_count - Número de requisições
* error\_rate - Taxa de erro
* active\_users - Usuários ativos
* memory\_usage - Uso de memória
* cpu\_usage - Uso de CPU

### 7. Logs do Sistema

**Configuração:**

{  
 "data\_source": {  
 "type": "log\_file",  
 "file\_path": "logs/cortexbi.log",  
 "pattern": "execution\_time: (\\d+)",  
 "aggregation": "avg",  
 "window\_minutes": 5  
 }  
}

## 📝 Arquivo: data\_sources.json

Este arquivo centraliza as configurações de conexão para evitar repetição.

**Criar** config/anomaly\_rules/data\_sources.json:

{  
 "version": "2.0",  
 "description": "Configurações de conexão para fontes de dados",  
   
 "connections": {  
 "sql\_principal": {  
 "type": "sql\_server",  
 "host": "servidor.database.windows.net",  
 "port": 1433,  
 "database": "vendas\_db",  
 "username": "${SQL\_SERVER\_USERNAME}",  
 "password": "${SQL\_SERVER\_PASSWORD}",  
 "driver": "ODBC Driver 17 for SQL Server",  
 "connection\_timeout": 30,  
 "query\_timeout": 60,  
 "pool\_size": 5  
 },  
   
 "sql\_secundario": {  
 "type": "sql\_server",  
 "host": "servidor-backup.database.windows.net",  
 "port": 1433,  
 "database": "vendas\_db\_backup",  
 "username": "${SQL\_SERVER\_USERNAME}",  
 "password": "${SQL\_SERVER\_PASSWORD}",  
 "driver": "ODBC Driver 17 for SQL Server"  
 },  
   
 "api\_analytics": {  
 "type": "api",  
 "base\_url": "https://api.analytics.empresa.com",  
 "auth\_type": "bearer",  
 "token": "${API\_ANALYTICS\_TOKEN}",  
 "timeout": 30,  
 "retry\_count": 3  
 },  
   
 "api\_erp": {  
 "type": "api",  
 "base\_url": "https://erp.empresa.com/api/v2",  
 "auth\_type": "basic",  
 "username": "${ERP\_API\_USER}",  
 "password": "${ERP\_API\_PASSWORD}",  
 "timeout": 60  
 }  
 },  
   
 "power\_bi": {  
 "tenant\_id": "${POWERBI\_TENANT\_ID}",  
 "client\_id": "${POWERBI\_CLIENT\_ID}",  
 "client\_secret": "${POWERBI\_CLIENT\_SECRET}",  
 "authority": "https://login.microsoftonline.com",  
 "scope": ["https://analysis.windows.net/powerbi/api/.default"]  
 },  
   
 "sharepoint": {  
 "site\_url": "${SHAREPOINT\_SITE\_URL}",  
 "client\_id": "${SHAREPOINT\_CLIENT\_ID}",  
 "client\_secret": "${SHAREPOINT\_CLIENT\_SECRET}",  
 "tenant\_id": "${SHAREPOINT\_TENANT\_ID}"  
 },  
   
 "file\_paths": {  
 "data\_root": "data/raw",  
 "backup\_root": "data/backup",  
 "archive\_root": "data/archive"  
 }  
}

## 🔐 Arquivo: .env (Credenciais)

Todas as credenciais sensíveis devem estar no arquivo .env:

# ===== SQL SERVER =====  
SQL\_SERVER\_USERNAME=usuario\_sql  
SQL\_SERVER\_PASSWORD=senha\_super\_secreta\_123  
  
# ===== APIs =====  
API\_ANALYTICS\_TOKEN=eyJhbGciOiJIUzI1NiIsInR5cCI6IkpXVCJ9...  
ERP\_API\_USER=api\_user  
ERP\_API\_PASSWORD=senha\_erp\_456  
  
# ===== POWER BI =====  
POWERBI\_TENANT\_ID=12345678-1234-1234-1234-123456789abc  
POWERBI\_CLIENT\_ID=87654321-4321-4321-4321-cba987654321  
POWERBI\_CLIENT\_SECRET=cliente\_secreto\_powerbi  
  
# ===== SHAREPOINT =====  
SHAREPOINT\_SITE\_URL=https://empresa.sharepoint.com/sites/dados  
SHAREPOINT\_CLIENT\_ID=sharepoint-client-id  
SHAREPOINT\_CLIENT\_SECRET=sharepoint-client-secret  
SHAREPOINT\_TENANT\_ID=tenant-id-sharepoint  
  
# ===== ALERTAS =====  
ALERT\_EMAIL\_TO=gerente@empresa.com,diretor@empresa.com  
SMTP\_USER=cortexbi@empresa.com  
SMTP\_PASSWORD=senha\_email\_789  
TEAMS\_WEBHOOK\_URL=https://outlook.office.com/webhook/seu-webhook

## 🎯 Exemplos Práticos Completos

### Exemplo 1: Monitorar Vendas do SQL Server

**1. Configurar conexão** (data\_sources.json):

{  
 "connections": {  
 "sql\_vendas": {  
 "type": "sql\_server",  
 "host": "vendas.database.windows.net",  
 "database": "VendasDB",  
 "username": "${SQL\_VENDAS\_USER}",  
 "password": "${SQL\_VENDAS\_PASS}"  
 }  
 }  
}

**2. Adicionar credenciais** (.env):

SQL\_VENDAS\_USER=vendas\_user  
SQL\_VENDAS\_PASS=senha123

**3. Configurar métrica** (metrics\_config.json):

{  
 "metrics": [  
 {  
 "id": "vendas\_diarias",  
 "name": "Vendas Diárias",  
 "enabled": true,  
   
 "data\_source": {  
 "type": "sql\_server",  
 "connection\_id": "sql\_vendas",  
 "query": "SELECT SUM(valor\_venda) as total FROM vendas WHERE CAST(data\_venda AS DATE) = CAST(GETDATE() AS DATE)",  
 "value\_column": "total"  
 },  
   
 "historical\_data": {  
 "query": "SELECT CAST(data\_venda AS DATE) as data, SUM(valor\_venda) as total FROM vendas WHERE data\_venda >= DATEADD(day, -30, GETDATE()) GROUP BY CAST(data\_venda AS DATE)",  
 "min\_samples": 30  
 },  
   
 "alert\_config": {  
 "severity": "high",  
 "threshold\_type": "percentage",  
 "threshold\_value": 15,  
 "channels": ["email"],  
 "recipients": ["gerente@empresa.com"]  
 },  
   
 "monitoring": {  
 "check\_interval": 3600  
 }  
 }  
 ]  
}

### Exemplo 2: Monitorar CSV de Custos

**1. Preparar arquivo CSV** (data/raw/custos.csv):

data,custo\_total,categoria  
2025-10-01,45000,Operacional  
2025-10-02,47000,Operacional  
2025-10-03,44500,Operacional

**2. Configurar métrica** (metrics\_config.json):

{  
 "metrics": [  
 {  
 "id": "custos\_operacionais",  
 "name": "Custos Operacionais",  
 "enabled": true,  
   
 "data\_source": {  
 "type": "csv",  
 "file\_path": "data/raw/custos.csv",  
 "value\_column": "custo\_total",  
 "timestamp\_column": "data"  
 },  
   
 "historical\_data": {  
 "source": "same\_file",  
 "min\_samples": 30  
 },  
   
 "alert\_config": {  
 "severity": "medium",  
 "threshold\_type": "absolute",  
 "threshold\_value": 50000,  
 "channels": ["email"],  
 "recipients": ["controller@empresa.com"]  
 },  
   
 "monitoring": {  
 "check\_interval": 7200  
 }  
 }  
 ]  
}

### Exemplo 3: Monitorar API Externa

**1. Configurar conexão** (data\_sources.json):

{  
 "connections": {  
 "api\_analytics": {  
 "type": "api",  
 "base\_url": "https://api.analytics.com",  
 "auth\_type": "bearer",  
 "token": "${ANALYTICS\_TOKEN}"  
 }  
 }  
}

**2. Adicionar token** (.env):

ANALYTICS\_TOKEN=seu-token-aqui

**3. Configurar métrica** (metrics\_config.json):

{  
 "metrics": [  
 {  
 "id": "taxa\_conversao",  
 "name": "Taxa de Conversão",  
 "enabled": true,  
   
 "data\_source": {  
 "type": "api",  
 "url": "https://api.analytics.com/conversao/hoje",  
 "method": "GET",  
 "headers": {  
 "Authorization": "Bearer ${ANALYTICS\_TOKEN}"  
 },  
 "response\_path": "data.taxa"  
 },  
   
 "historical\_data": {  
 "api\_url": "https://api.analytics.com/conversao/historico",  
 "api\_params": {"dias": 30},  
 "response\_path": "data.historico"  
 },  
   
 "alert\_config": {  
 "severity": "high",  
 "threshold\_type": "percentage",  
 "threshold\_value": 10,  
 "channels": ["teams"],  
 "recipients": ["marketing@empresa.com"]  
 },  
   
 "monitoring": {  
 "check\_interval": 1800  
 }  
 }  
 ]  
}

## ✅ Checklist de Configuração

Use este checklist para garantir que tudo está configurado:

### 1. Estrutura de Diretórios

* ☐ config/anomaly\_rules/ existe
* ☐ data/raw/ existe
* ☐ data/anomalies/ existe
* ☐ logs/anomalies/ existe

### 2. Arquivos de Configuração

* ☐ metrics\_config.json criado
* ☐ data\_sources.json criado
* ☐ .env configurado com credenciais

### 3. Fontes de Dados

* ☐ Conexões SQL Server testadas
* ☐ Arquivos CSV/Excel no lugar correto
* ☐ APIs acessíveis e autenticadas
* ☐ Power BI configurado (se aplicável)

### 4. Métricas

* ☐ Pelo menos 1 métrica configurada
* ☐ Query SQL validada (se SQL Server)
* ☐ Colunas corretas especificadas
* ☐ Dados históricos disponíveis (mínimo 30 dias)

### 5. Alertas

* ☐ Email configurado (SMTP)
* ☐ Teams webhook configurado (se aplicável)
* ☐ Destinatários definidos
* ☐ Templates de alerta personalizados (opcional)

### 6. Testes

* ☐ Conexão com fonte de dados testada
* ☐ Query retorna dados válidos
* ☐ Modelo de ML treinado
* ☐ Alerta de teste enviado com sucesso

## 🧪 Scripts de Teste

### Testar Conexão SQL Server

# test\_sql\_connection.py  
  
import pyodbc  
import os  
from dotenv import load\_dotenv  
  
load\_dotenv()  
  
connection\_string = (  
 f"DRIVER={{ODBC Driver 17 for SQL Server}};"  
 f"SERVER={os.getenv('SQL\_SERVER\_HOST')};"  
 f"DATABASE={os.getenv('SQL\_SERVER\_DATABASE')};"  
 f"UID={os.getenv('SQL\_SERVER\_USERNAME')};"  
 f"PWD={os.getenv('SQL\_SERVER\_PASSWORD')}"  
)  
  
try:  
 conn = pyodbc.connect(connection\_string)  
 cursor = conn.cursor()  
   
 # Testar query  
 query = "SELECT SUM(valor\_venda) as total FROM vendas WHERE CAST(data\_venda AS DATE) = CAST(GETDATE() AS DATE)"  
 cursor.execute(query)  
 result = cursor.fetchone()  
   
 print(f"✅ Conexão bem-sucedida!")  
 print(f"📊 Resultado da query: {result[0]}")  
   
 conn.close()  
   
except Exception as e:  
 print(f"❌ Erro na conexão: {e}")

### Testar Leitura de CSV

# test\_csv\_reading.py  
  
import pandas as pd  
  
try:  
 df = pd.read\_csv("data/raw/custos.csv")  
   
 print(f"✅ Arquivo CSV lido com sucesso!")  
 print(f"📊 Linhas: {len(df)}")  
 print(f"📊 Colunas: {list(df.columns)}")  
 print(f"\nÚltimos 5 registros:")  
 print(df.tail())  
   
 # Testar coluna de valor  
 valor\_atual = df['custo\_total'].iloc[-1]  
 print(f"\n💰 Valor atual: {valor\_atual}")  
   
except Exception as e:  
 print(f"❌ Erro ao ler CSV: {e}")

### Testar API

# test\_api\_connection.py  
  
import requests  
import os  
from dotenv import load\_dotenv  
  
load\_dotenv()  
  
url = "https://api.analytics.com/conversao/hoje"  
headers = {  
 "Authorization": f"Bearer {os.getenv('ANALYTICS\_TOKEN')}",  
 "Content-Type": "application/json"  
}  
  
try:  
 response = requests.get(url, headers=headers, timeout=30)  
 response.raise\_for\_status()  
   
 data = response.json()  
   
 print(f"✅ API acessível!")  
 print(f"📊 Status: {response.status\_code}")  
 print(f"📊 Resposta:")  
 print(data)  
   
except Exception as e:  
 print(f"❌ Erro ao acessar API: {e}")

### Testar Configuração Completa

# test\_full\_config.py  
  
import json  
import os  
from pathlib import Path  
  
def test\_configuration():  
 """Testa toda a configuração"""  
   
 print("🔍 Verificando configuração...\n")  
   
 # 1. Verificar estrutura de diretórios  
 print("1️⃣ Verificando diretórios...")  
 dirs = [  
 "config/anomaly\_rules",  
 "data/raw",  
 "data/anomalies",  
 "logs/anomalies"  
 ]  
 for dir\_path in dirs:  
 if Path(dir\_path).exists():  
 print(f" ✅ {dir\_path}")  
 else:  
 print(f" ❌ {dir\_path} - NÃO EXISTE!")  
   
 # 2. Verificar arquivos de configuração  
 print("\n2️⃣ Verificando arquivos de configuração...")  
 files = [  
 "config/anomaly\_rules/metrics\_config.json",  
 "config/anomaly\_rules/data\_sources.json",  
 ".env"  
 ]  
 for file\_path in files:  
 if Path(file\_path).exists():  
 print(f" ✅ {file\_path}")  
 else:  
 print(f" ❌ {file\_path} - NÃO EXISTE!")  
   
 # 3. Validar metrics\_config.json  
 print("\n3️⃣ Validando metrics\_config.json...")  
 try:  
 with open("config/anomaly\_rules/metrics\_config.json", 'r') as f:  
 config = json.load(f)  
   
 metrics = config.get('metrics', [])  
 print(f" ✅ Arquivo válido")  
 print(f" 📊 Métricas configuradas: {len(metrics)}")  
   
 for metric in metrics:  
 status = "✅ Habilitada" if metric.get('enabled') else "⏸️ Desabilitada"  
 print(f" - {metric.get('name')}: {status}")  
   
 except Exception as e:  
 print(f" ❌ Erro ao validar: {e}")  
   
 # 4. Verificar variáveis de ambiente  
 print("\n4️⃣ Verificando variáveis de ambiente...")  
 from dotenv import load\_dotenv  
 load\_dotenv()  
   
 env\_vars = [  
 "SQL\_SERVER\_USERNAME",  
 "ALERT\_EMAIL\_TO",  
 "ANOMALY\_DETECTION\_ENABLED"  
 ]  
 for var in env\_vars:  
 value = os.getenv(var)  
 if value:  
 print(f" ✅ {var} = {'\*' \* len(value)}")  
 else:  
 print(f" ⚠️ {var} - NÃO DEFINIDA")  
   
 print("\n✅ Verificação completa!")  
  
if \_\_name\_\_ == "\_\_main\_\_":  
 test\_configuration()

Execute:

python test\_full\_config.py

## 📞 Resumo

**A configuração da fonte de dados está em:**

1. **config/anomaly\_rules/metrics\_config.json** → Define QUAIS métricas monitorar e ONDE buscar os dados
2. **config/anomaly\_rules/data\_sources.json** → Define conexões reutilizáveis (SQL, APIs, etc.)
3. **.env** → Armazena credenciais sensíveis (senhas, tokens, etc.)

**Cada métrica precisa especificar:** - data\_source → De onde vem o dado atual - historical\_data → De onde vem o histórico para comparação - alert\_config → Como alertar quando anomalia for detectada - monitoring → Com que frequência verificar

**CÓRTEX BI v2.0** - *Cognitive Operations & Real-Time EXpert Business Intelligence*  
Desenvolvido em parceria com **Manus AI** | Outubro 2025

📊 **Fontes de Dados Configuradas = Anomalias Detectadas**